ADF Manual

Amsterdam Modeling Suite 2019

www.scm.com

Apr 16, 2019

CONTENTS

GENERAL

1.1 Introduction

ADF (Amsterdam Density Functional) is a Fortran program for calculations on atoms and molecules (in gas phase or solution). It can be used for the study of such diverse fields as molecular spectroscopy, organic and inorganic chemistry, crystallography and pharmacochemistry. A separate program BAND is available for the study of periodic systems: crystals, surfaces, and polymers. The COSMO-RS program is used for calculating thermodynamic properties of (mixed) fluids.

The underlying theory is the Kohn-Sham approach to Density-Functional Theory (DFT). This implies a one-electron picture of the many-electron systems but yields in principle the exact electron density (and related properties) and the total energy.

If ADF is a new program for you we recommend that you carefully read the section *Technical remarks, Terminology* (page 7), which presents a discussion of a few ADF-typical aspects and terminology. This will help you to understand and appreciate the output of an ADF calculation.

ADF has been developed since the early 1970s (at that time called HFS, later AMOL, see also Refs. [*308-310* (page 925)]), mainly by the two theoretical chemistry groups of, respectively, the Vrije Universiteit in Amsterdam (http://www.chem.vu.nl/en/research/division-theoretical-chemistry/index.asp) and the University of Calgary, Canada. Other researchers have also contributed. As a major research tool of these academic development groups, ADF is in continuous development and retains a firm basis in the academic world.

Maintenance and distribution of the commercial (export) version of the program is done by Software for Chemistry & Materials B.V. (SCM) (http://www.scm.com), a company based in Amsterdam, formally split off from the theoretical chemistry group in Amsterdam but practically still very much a part of it. Documentation such as User manuals, Installation instructions, Examples, Theoretical documents can be found at the SCM web site.

Publications based on research with ADF should include appropriate references to the program. We recommend that references are made both to the program itself and to publications related to its development and structure. See the *Required Citations* (page 895).

The installation of the Amsterdam Density Functional program package (ADF) is explained in the Installation manual. This User's Guide describes how to use the program, how input is structured, what files are produced, and so on. Some special applications of ADF are described in the *Examples* (page 397).

Where references are made to the operating system (OS) and to the file system on your computer the terminology of UNIX type OSs is used.

The ADF package is in continuous development to extend its functionality and applicability, to increase its efficiency and user-friendliness, and of course to correct errors. We appreciate comments and suggestions for improvement of the software and the documentation.

1.1.1 Functionality

- Single Point calculation

- Geometry Optimization

- Transition States

- Frequencies and thermodynamic properties

- Tracing a Reaction Path

- Computation of any electronic configuration

- Excitation energies, oscillator strengths, transition dipole moments, (hyper)polarizabilities, Van der Waals dispersion coefficients, CD spectra, ORD, MCD, VCD, magnetizabilities, Verdet constants, using Time-Dependent Density Functional Theory (TDDFT)

- ESR (EPR) g-tensors, A-tensors, NQCCs

- NMR chemical shifts and spin-spin coupling constants

- Mössbauer spectroscop

- Transport properties: charge transfer integrals, NEGF

- Various other molecular properties

- Treatment of large systems and environment by the QM/MM (Quantum Mechanics / Molecular Mechanics) hybrid approach.

1.1.2 Applicability

All elements of the periodic table can be used (Z = 1-118). For each of the elements, basis sets of different sizes are available, ranging from minimal to high quality. Special basis sets are provided for relativistic calculations within the ZORA approach and for response calculations that require additional diffuse basis functions.

1.1.3 Model Hamiltonian

- A choice of Density Functionals, both for the Local Density Approximation (LDA), for the Generalized Gradient Approximation (GGA), for (range-separated) hybrid functionals (not for all properties available), and for meta-GGA functionals (not for all properties available) are available.

- Spin: restricted or unrestricted

- Relativistic effects: scalar approximation and spin-orbit (double-group symmetry), using the (now recommended) ZORA or the (previously used) Pauli formalism, X2C

- Environment: Solvent Effects, Homogeneous Electric Field, Point Charges (Madelung Fields), QM/MM method, FDE

- Constrained Density Functional Theory

1.1.4 Analysis

- Decomposition of the bond energy in 'chemical' components (steric interaction, Pauli repulsion, orbital interactions...)

- Natural orbitals for chemical valence (ETS-NOCV)

- Representation of data (Molecular Orbital coefficients, Mulliken Populations) in terms of the constituent chemical fragments in the molecule, along with the conventional representation in elementary basis functions

- Atomic charge determination by Hirshfeld analysis and by Voronoi analysis, multipole derived charges, along with the classical Mulliken populations, and Mayer bond orders

- QTAIM analysis based on local, atomic and non-local descriptors for bonding description.

- Conceptual DFT descriptors including global, atomic, non-local ones and a detailed analysis of the dual descriptor's domains (predominantly electrophilic or nucleophilic regions).

- Bond energy decomposition based on the interacting quantum atoms (IQA) approach and using QTAIM real-space partition. Any atom-atom interaction can be evaluated and decomposed into electrostatic (ionic) and exchange (covalent) contributions.

- Localized molecular orbitals

1.1.5 Technical

- The implementation is based upon a highly optimized numerical integration scheme for the evaluation of matrix elements of the Fock operator, property integrals involving the charge density, etc. The code has been vectorized and parallelized.

- Basis functions are Slater-Type Orbitals (STOs). A database is available with several basis sets for each atom in the periodic table of elements.

- The Coulomb potential is evaluated via an accurate fitting of the charge density.

- A frozen core facility is provided for an efficient treatment of the inner atomic shells.

- Extensive use is made of point group symmetry. Most of the commonly encountered symmetry groups are available.

- Linear scaling techniques are used to speed up calculations on large molecules

1.1.6 Fragments

ADF has a fragment oriented approach: the poly-atomic system to be computed is conceptually built up from fragments, the molecular one-electron orbitals are calculated as linear combinations of fragment orbitals, and final analyzes of e.g. the bonding energy are in terms of fragment properties. The fragments may be single atoms or larger moieties.

When you compute a system in terms of its constituent fragments, these fragments must have been computed before and their properties must be passed on to the current calculation. This is done by attaching *fragment files*, which contain the necessary information. A fragment file is simply the standard result file of an ADF calculation on that fragment.

When using Basic Atoms as fragments, you do not need to create the fragment files yourself. Instead, you may use the Basis key, and ADF will create the required fragment files automatically. We therefore recommend this feature for starting ADF users.

Basic atoms

Obviously there must be a set of fundamental fragments that are not defined in terms of smaller fragments. Therefore ADF has two modes of execution: the normal mode, using fragments, and the create mode, in which a fundamental fragment is generated. Such a fundamental fragment *must* be a single atom, spherically symmetric and spin-restricted (i.e. spin-α and spin-β orbitals are spatially identical, they are equally occupied, and fractional occupations are applied, if necessary, to distribute the electrons equally over symmetry-degenerate states). Such a fundamental fragment is denoted a *basic atom*. The basic atoms are the smallest building blocks from which any 'real' calculations are started.

One should realize that the basic atoms are artificial objects that are convenient in the computational approach but that do not necessarily represent real atoms very well (in fact, usually not at all). The bonding energy of a molecule with respect to basic atoms, for instance, should be corrected for this discrepancy in order to get a decent comparison against experimental data. See ref. [*1* (page 907)] for a discussion and for examples of applicable values.

A basic atom is computed in the conventional way. The one-electron orbitals are determined as linear combinations of basis functions; the frozen core approximation may be applied for the inner atomic states; a particular type of density functional can be chosen, et cetera. You may have, for instance, different basic Copper atoms by using different basis sets, by choosing different levels of frozen core approximations, or by applying different density functionals.

Automatic mode

If you are using 'Basic Atom' fragments only, you do not need to prepare the corresponding fragment files yourself. Instead, add the BASIS block key to the ADF input, and ADF will generate all the required fragment files for you. This makes your job scripts and ADF inputs simpler, it ensures that consistent options for the create runs and molecular runs are used, and you will be sure that the fragment files used have been created by the same release of ADF.

1.1.7 Slater-type basis sets

ADF uses Slater-Type Orbitals (STO's) as basis functions. Slaters can display the correct nuclear cusp and asymptotic decay.

$$f(\mathbf{r}) = Y_{lm} r^n e^{-\zeta r}$$

The center of the function is at a nucleus, the Y_{lm} are spherical harmonics, and the exponential factor ζ (zeta) determines the long-range decay of the function.

ADF comes with a database of thoroughly tested basis set files, ranging in quality from single-zeta to quadruple-zeta basis sets with various diffuse and polarization functions. All-electron and frozen-core basis sets are available for all elements, including lanthanides and actinides. The frozen-core approximation can be used to considerably reduce the computation time for systems with heavy nuclei, in a controlled manner.

1.2 What's new in ADF 2019

1.2.1 New features

In comparison to ADF 2018, the ADF 2019 release offers the following new functionality:

- Spectroscopy: *AOResponse* (page 192): Raman spectra in combination with spin-orbit coupling

- *Restructured QTAIM section* (page 247)

- *New Conceptual DFT section* (page 249) introducing extra reactivity descriptors (global, atomic and non-local ones). A new option is dedicated to the analysis of the dual descriptor's domains (predominantly electrophilic or nucleophilic regions).

- *Optimize Spin Polarization* (page 45)

- *DRF gradients: expert option* (page 85)

1.3 Feature List

1.3.1 Model Hamiltonians

- *XC energy functionals and potentials* (page 55)
    - *LDA* (page 56), *GGA* (page 57), *meta-GGA* (page 59), *model potentials* (page 64)
    - *(meta-)hybrid* (page 59), *range-separated* (page 60)
    - *dispersion corrected* (page 67)
- *Relativistic effects (ZORA and spin-orbit coupling)* (page 72)
- Solvents and other environments
    - *COSMO* (page 74), *SM12* (page 81), QM/MM , pdb2adf, *DIM/QM* (page 84), *SCRF* (page 104), *FDE* (page 95), *3D-RISM* (page 111), QUILD
- *Homogeneous electric field and point charges* (page 117)

1.3.2 Structure and Reactivity

- *Geometry Optimizations* (page 120)
- *Linear Transit* (page 127), *Transition States* (page 125), *CI-NEB* (page 131), *TSRC* (page 126)
- *Intrinsic Reaction Coordinate* (page 129)
- *Excited state optimizations with TDDFT gradients* (page 182)

Optimizations (page 120) can be done in Cartesian, internal, and delocalized coordinates. Various *restraints* (page 138) and constraints (*1* (page 135), *2* (page 133)) can be imposed. *Hessians* (page 139) are available analytically for most GGAs, and numerically otherwise. Preoptimization is possible with DFTB.

1.3.3 Spectroscopic properties

- Vibrational Spectroscopy
    - *IR frequencies and intensities* (page 139)
    - *Mobile Block Hessian (MBH)* (page 143), *Vibrational Circular Dichroism (VCD)* (page 156)
    - *Raman intensities* (page 149)
    - Resonance Raman from *frequency-dependent polarizabilities* (page 151) or *excited state gradients* (page 152)
    - *vibrational Raman optical activity (VROA)* (page 155)
    - *Franck-Condon Factors* (page 184)
- *Excitation energies: UV/Vis spectra, X-ray absorption, CD, MCD* (page 161)
    - *UV/Vis spectra, oscillator strengths* (page 162), *open shell excitations* (page 165), *spin-orbit coupled excitations* (page 170)
    - *core excitations* (page 166), *Quadrupole oscillator strengths* (page 169), *XES* (page 170)
    - *TD-DFT+TB* (page 175), *sTDA, sTDDFT* (page 176)
    - *vibrationally resolved electronic spectra* (page 184)

- *excited state optimizations* (page 182)
- *CD spectra* (page 162), *MCD* (page 178)
- *(Hyper-)Polarizabilities, dispersion coefficients, ORD, magnetizabilities, Verdet constants* (page 190)
  - *frequency-dependent (hyper)polarizabilities* (page 190), *lifetime effects* (page 192)
  - *van der Waals dispersion coefficients* (page 198)
  - *optical rotatory dispersion (ORD)* (page 192)
  - *magnetizability* (page 192)
  - *Verdet constants, Faraday terms* (page 192)
- NMR
  - *chemical shifts* (page 203)
  - *spin-spin couplings* (page 209)
- ESR (EPR)
  - *g-tensors (g-factor)* (page 218)
  - *A-tensor (hyperfine interaction)* (page 218)
  - *zero-field splitting (ZFS, D-tensor)* (page 218)
- *Nuclear quadrupole interaction (EFG), ESR Q-tensor* (page 221)
- *Mössbauer* (page 221), *NRVS* (page 222)

1.3.4 Charge transport properties

- *charge transfer integrals* (page 223)
- *Non-self-consistent Green's function calculation* (page 227)

1.3.5 Analysis

- *Fragments* (page 3)
- *Bond energy decomposition* (page 12), *ETS-NOCV* (page 240)
- Advanced charge density and MO analysis
  - *Mulliken* (page 266), *Multipole-derived charges* (page 267)
  - *Hirshfeld charges, Voronoi deformation density* (page 266), *CM5 charges* (page 267), *FOD* (page 252)
  - *bond orders: Mayer, Nalewajski-Mrozek* (page 267)
  - *Bader (QT-AIM)* (page 247)
  - *NB0 6.0* (page 242)
  - *(partial) DOS* (page 283)
- *Molecular symmetry* (page 9), *Schönfliess symbols and symmetry labels* (page 360)

1.3.6 Accuracy and Efficiency

- *Slater-type basis sets* (page 8)
    - *Z = 1 to 118, all electron, frozen-core, non-relativistic and relativistic* (page 29)
    - *SZ, DZ, DZP, TZP, TZ2P, QZ4P, even-tempered, diffuse* (page 29)
- *Integration scheme* (page 300)
- *Parallelization* (page 14)
- *Linear scaling / distance cut-offs* (page 315)
- *Density fit* (page 305) and *frozen core approximation* (page 8)
- SCF convergence: *simple damping, DIIS, EDIIS, ADIIS, LISTi, ARH* (page 296)

1.4 Technical remarks, Terminology

A few words about ADF as regards its technical setup and the names and abbreviations used in this manual. References to these will be made in the discussion of output and print switches.

1.4.1 Density functional theory

The underlying theory of the ADF package is the Kohn-Sham approach to the Density-Functional Theory (DFT). Kohn-Sham DFT is an important first-principles computational method to predict chemical properties accurately and to analyze and interpret these in convenient and simple chemical terms.

The reasons for its popularity and success are easy to understand. In the first place, the DFT approach is in principle exact. In particular, the Kohn-Sham method implies a one-electron picture of the many-electron systems but yields in principle the exact electron density (and related properties) and the total energy. The exact exchange-correlation (XC) functional is unknown, but the currently available XC functionals provide in most cases already a 'chemical' accuracy of a few kcal/mol for binding energies. Moreover, the quest for more accurate ones based on a more detailed understanding of their essential properties is continuing.

In the past two decades, computational chemistry has evolved from a curiosity of theoreticians into a mainstream tool used by all types of chemists, physicists and engineers who have an interest in research and development. In that time Density Functional Theory has become the dominant method for modeling chemistry at the molecular level.

In the second place, it preserves at all levels of approximation the appealing one-electron molecular orbital (MO) view on chemical reactions and properties. The computed orbitals are suitable for the typical MO-theoretical analyses and interpretations. The KS method effectively incorporates all correlation effects.

In the third place, it is a relatively efficient computational method, and its fundamental scaling properties do not deteriorate when methodological precision is increased, in particular, when a more accurate XC functional is applied. Recent research paves the way to implementations that scale only linearly with the system size. This brings within reach the treatment by fundamental quantum chemical methods of systems with hundreds, maybe even thousands of atoms.

DFT gives superior accuracy to Hartree-Fock theory and semi-empirical approaches, and it is well suited for molecules containing metal atoms. In contrast to conventional ab initio methods (MP2, CI, CC), it enables accurate treatment of systems with several hundreds of atoms (or several thousands with QM/MM).

Text is mostly taken from: *Chemistry with ADF* (page 925), G. te Velde, F.M. Bickelhaupt, E.J. Baerends, C. Fonseca Guerra, S.J.A. van Gisbergen, J.G. Snijders, T. Ziegler J. Comp. Chem. **22** (2001) 931.

1.4.2 The Kohn-Sham MO model

The basic postulate in Kohn-Sham DFT is that we can apply a one-electron formulation to the system of N interacting electrons by introducing a suitable local potential $V_{XC}(r)$, in addition to any external potentials $V_{ext}(r)$ and the Coulomb potential of the electron cloud $V_C(r)$, and solving:

$$[T + V_{ext}(r) + V_C(r) + V_{XC}(r)]\phi_i(r) = \epsilon_i \phi_i(r)$$

Here T is the kinetic energy operator. The potential $V_{XC}(r)$ is the functional derivative with respect to the density ρ of $E_{XC}[\rho]$, the exchange-correlation energy functional. The one-electron molecular orbitals (MOs) ϕ_i with corresponding orbital energies ϵ_i define the exact electronic charge density and give, in principle, access to all properties because these are expressible as functional of the density, in particular the energy. Moreover, they provide an intuitively appealing view of the system as being built from independent-electron orbitals with all ensuing interpretations. The exact form of the exact energy density $E_{XC}(r)$, representing and incorporating all exchange and correlation (XC) effects is unknown. From general principles one can formulate conditions on what $E_{XC}(r)$ should look like, and several, more and more advanced expressions have been advocated for it in the literature. Their application to real systems has been impressively successful, and it seems likely that a further increase of accuracy is a matter of time.

1.4.3 Basis functions and orbitals

Let us make a clear distinction between (basis) *functions* and *orbitals*, even where these phrases are sometimes mixed up in the traditional terminology. Orbitals are always specific combinations of the basis functions. Orbitals are related to the computed eigenfunctions of some Fock operator or Hamiltonian occurring in the run or in a related preceding calculation. Functions are merely the elementary mathematical entities in which the orbitals are expressed. A Slater Type Orbital (STO), for instance is a function, not an orbital.

The physical meaning of one-electron orbitals in DFT has often been questioned. We believe that they are useful quantities for interpretation, just like the HF orbitals. For a recent discussion see [2 (page 907)].

See also

- ADF-GUI tutorial: basis set effects

Cartesian function sets, spurious components

ADF employs Slater-type exponential basis functions centered on the atoms. Such a function consists of an exponential part exp(-ar) and a polynomial pre-factor $r^{kr} x^{kx} y^{ky} z^{kz}$. A function *set* is characterized by its radial behavior (the exponential part and the power of r, kr) and by its angular momentum quantum number *l*. The functions in such a set consist of all possible combinations $x^{kx} y^{ky} z^{kz}$, such that kx+ky+kz=l. These are denoted the *Cartesian* spherical harmonics.

The Cartesian function sets are very suitable for computational manipulations, but they have a drawback. By inspection it is easily verified that a *d*-set consists of 6 Cartesian functions, while there can of course be only 5 true *d*-type functions among them: one (linear combination) of them is in fact an *s*-type function ($x^2 + y^2 + z^2$). Similarly, there are 10 *f*-type Cartesian functions, 3 of which are in fact *p*-functions. And so on. In ADF all such lower-*l* (combinations of) functions are projected out of the basis and not employed. As a consequence the basis set *size* in the sense of the number of degrees of freedom and hence the number of possible eigenfunctions of the Fock operator is smaller than the number of expansion coefficients that refer to the primitive (Cartesian) basis functions.

The abbreviation BAS is used for references to the elementary Cartesian basis functions.

Frozen core: Core Orbitals and Core Functions

To speed up the computation the innermost atomic shells are kept frozen. The frozen Core Orbitals (CO), which are solutions of a large-basis all-electron calculation on the isolated atom, are expressed in an auxiliary set of (Slater-type)

basis functions cor-bas, distinct from the valence set. The core basis set and the COs expansion coefficients are stored in the basis set files.

Orthogonality of the valence Molecular Orbitals (MO) to the COs is achieved with the help of so-called Core Functions (CF). These functions are included in the valence set but they are not additional degrees of freedom. Each of the normal valence functions is combined with a linear combination of all CFs in the molecule in such a way that the transformed function (cbas) is orthogonal to all frozen COs in the molecule. There are exactly as many CFs as COs so the orthogonality condition for all valence basis functions amounts to the solution of a linear system where the number of conditions equals the number of parameters.

This aspect once more increases the discrepancy between the number of expansion coefficients of an MO and the number of MOs: the expansion coefficients in the most elementary bas representation run over all bas functions, including the CFs among them. At some places there may, alternatively, be expansions in the core-orthogonalized BAS functions, CBAS, where the CFs do not count anymore: they are included implicitly in the cbas functions.

Symmetry

The Overlap and Fock matrices become block-diagonal by using symmetry-adapted combination of the (C)BAS functions, such that each such function transforms under the symmetry operators as one of the subspecies of the irreducible representations (irrep) of the symmetry group. Symmetry adapted functions are denoted (C)SBAS.

For a given irrep and subspecies not all elementary basis functions can participate in the symmetry adapted combinations. For instance, for an atom in a reflection plane a basis function that is anti-symmetric with respect to the reflection cannot be part of any symmetric combination of functions. In particular for higher symmetries the number of BAS functions that are relevant for a subspecies may be considerably smaller than the total number of BAS functions. This is used to cut down expansion lengths, both as used internally in the computation and construction of the Fock matrix, and in printed output. The printed expansion coefficients (in the bas representation) refer only to the participating BAS functions. A defining list of them is printed at an early stage of the run for each of the subspecies.

Orthonormal basis

It is often computationally convenient to use an orthonormal basis. This is constructed from the CSBAS basis by a Lowdin orthogonalization procedure. The resulting symmetry-adapted orthonormal basis is denoted low.

The MOs are computed by diagonalization of the Fock matrix in the LOW representation. The resulting eigenvectors are easily transformed back to any other representation whenever suitable, such as for instance to the primitive Cartesian bas representation (including the CFs).

Fragments

Except in Create mode, where a *basic atom* is constructed, the system is built up from fragments and the corresponding fragment files are attached to the run. The program reads from the files the fragment MOs and these are used as (compound) *basis functions* for the molecular calculation. The fragment MOs are called Fragment Orbitals (FO).

FOs belong of course to one of the symmetry representations of the fragment, but not necessarily to a symmetry representation of the new molecule. The FOs are therefore combined into symmetry-adapted combinations, SFOs, to serve as a symmetry-adapted basis in the molecule. These combinations may involve one or more FOs from the same fragment and/or from different fragments. In the latter case the fragments must be symmetry related by one of the operators of the molecule. Symmetry related fragments must of course be identical, apart from their spatial location: they must be of the same fragment *type*.

FOs are naturally orthogonal to the Core Orbitals of their own fragment, but not necessarily to COs of other fragments. By a suitable combination of the SFOs with all CFs in the molecule we obtain the core-orthogonalized symmetry-adapted CSFOs.

The CSFOs can be transformed to an orthonormal basis by a Lowdin transformation. The resulting basis is called low, as above.

Summary of functions and orbitals

In Create mode the (conceptual) approach is:

BAS → (core-orthogonalization) → CBAS → (symmetry) → CSBAS → (orthonormality) → LOW → (Fock diagonalization) → MO

In Fragment mode:

FO (=MO from fragment file) → (symmetry) → SFO → (core-orth.) → CSFO → (orthonormality) → LOW → (Fock diagonalization) → MO

Acronyms

BAS Elementary Cartesian basis functions, consisting of a radial part (exponential factor and power of r) and an angular part (Cartesian spherical harmonic). The complete BAS set contains spurious lower-l combinations; these combinations are projected out and not used in the calculation. The BAS set contains also Core Functions.

SBAS Symmetry-adapted combination of BAS functions.

CF Core Function, part of the bas set. The CFs do not represent degrees of freedom in the basis set but serve only to ensure orthogonalization of the valence space to all frozen Core Orbitals.

CBAS Core-orthogonalized elementary basis functions: the true valence (not-CF) BAS functions transformed by adding a suitable combination of the CFs. The total number of CBAS + the total number of of CFs equals the total number of BAS.

CSBAS Symmetry-adapted combination of cbas functions.

CO Frozen Core Orbitals, expressed as linear combinations of an auxiliary corbas basis set. The corbas set plays no role in the further discussion. The corbas functions are *not* the CFs.

The number of COs equals the number of CFs.

LOW Lowdin orthonormalized symmetry-adapted core-orthogonalized basis. In Create mode they are derived directly from the BAS functions, in Fragment mode from the Fragment Orbitals, which are themselves of course expressible in the BAS set.

FO Fragment Orbital: the MO of a fragment calculation, now used as a *basis function* in the molecule of which the fragment is part.

SFO Symmetry adapted combination of FOs.

CSFO Core-orthogonalized SFO.

1.4.4 Fit functions

Using Slater-type basis functions yields awkward multi-center integrals in the evaluation of the Coulomb potential. We therefore first need to find an approximate density-representation for which the Coulomb integral can be evaluated efficiently. This procedure is commonly referred to as density fitting. The default density fitting procedure in ADF is described in Ref. [*379* (page 929)].

An alternative density fitting approach (STOFIT) employs an auxiliary set of *fit* functions, see also Ref. [*308* (page 925)]. Like the basis functions, the fit functions are Slater-type exponential functions centered on the atoms.

The true density, a sum of *products* of basis functions, is then replaced (approximated) by a linear combination (not products!) of the fit functions. The combination coefficients are called the fit* coefficients*.

$$\rho(r) = \sum_i c_i f_i(r)$$

The Poisson equation for the fit functions is easily solved, yielding the (approximate) Coulomb potential as an expansion in fit *potential* functions $f_i^C(r)$

$$f_i^C(r) = \int f_i(r)/|r - r'|dr$$

$$V_{Coulomb}(\rho(r)) \approx \sum_i c_i f_i^C(r)$$

In the SCF procedure the fit coefficients are computed by a least-squares minimization of

$$\int (\rho_{exact} - \rho_{fit})^2 dr$$

with the constraint that ρ_{fit} contain the correct number of electrons. ρ_{exact} is defined as the sum of occupied orbitals (squared and multiplied by the appropriate occupation number). The accuracy of the fit approximation is important and the fit set plays a role similar to the basis set: too few functions (or badly chosen function characteristics) yield inferior results and there is also such a thing as the fit set limit. The fit functions on an atom are consequently an integral part of the definition of the basic atom and they are included in the Create data files. Fortunately, the size of the fit set does not determine the computational effort in such a drastic way as the size of the basis set does. We have chosen therefore to use always fair (though not extreme) fit sets, with the purpose that the effect of fit-incompleteness should in all cases be small enough to be ignored compared with basis set effects, numerical integration errors and Density Functional deficiencies. This does of course depend somewhat on the computed molecule and the studied properties, so a general guarantee cannot be given and, as with basis set effects, one should always have an open eye for possible problems and check the pertaining information in the output file.

One of the most important properties of a molecule is its energy, or its bonding energy with respect to the constituent fragments. The fit incompleteness introduces two types of errors. The first is that, since the Coulomb potential is only approximated, the SCF solution itself, i.e. the set of self-consistent Molecular Orbitals and their energy eigenvalues may be slightly wrong, yielding an error in the charge density and hence in the energy. Since the energy is to first order stable with respect to changes in the mo coefficients this error in the energy can be assumed very small. The second type of error derives from the computation of the energy from the (self-consistent) charge density, via the Coulomb potential. Let

$$\rho \equiv \rho_{exact}(r) = \rho_{fit}(r) + \delta(r)$$

and

$$V_{fit}(r) = \int \rho_{fit}(r')/|r - r'|dr$$

For the Coulomb energy of the charge density we have

$$2E_{Coul} = \iint \rho(r)\rho(r')/|r - r'|drdr' = \int \rho(r)V_{fit}(r)dr + \iint \rho(r)\delta(r)/|r - r'|drdr'$$

$$= \int V_{fit}(r)(\rho(r) + \delta(r))dr + \iint \delta(r)\delta(r')/|r - r'|drdr'$$

from which we see that the fit error is corrected to first order (by adding the fit deficiency $\delta(r)$ to the exact charge density when integrating against the fit potential) and that only a second order term remains that cannot be evaluated, the last term in the right-hand-side of the last equation.

A fair impression of the fit quality and the importance of the second order error term is obtained by checking

---

- the size of the first order correction term $\int V_{fit}(r)\delta(r)dr$ and

- the norm of the deficiency function, $\int \delta^2(r)dr$.

Both are printed in standard output, at the end of the output of the SCF procedure computational report. They are usually very small, which gives some confidence that the second order fit error can be ignored.

1.4.5 Three-step build-up of the bonding

The approach of ADF is based on fragments. This applies not only in the analysis at the end of the computation but also in the set-up of the program. The computation of the molecule from its constituent fragments takes place in three steps, and these are reflected in the analysis of bond energy components.

First, the (free, unrelaxed) fragments are placed at their positions in the molecule. This implies an *electrostatic* interaction: for each fragment the Coulomb interaction of its undisturbed charge density with the fields of the other fragments.

Next, the Pauli exclusion principle is applied. Even without considering self-consistency the one-electron orbitals of the combined fragments cannot represent a correct one-determinant wave function because the orbitals of different fragments are not orthogonal to one another. The program performs an orthonormalization of the occupied Fragment Orbitals to obtain an antisymmetrized product. This implies a change in the total molecular charge density from the sum-of-fragments to what is called the sum-of-*orthogonalized*-fragments. The corresponding (repulsive) energy term is evaluated separately and is called *Exchange* repulsion, or alternatively *Pauli* repulsion. The phrase *orthogonal(ized) fragments*, if you find it elsewhere in this manual or in the source code of ADF, refers to this aspect. The sum of Pauli repulsion and electrostatic interaction is called the *steric interaction*.

The third phase is the relaxation to self-consistency, with of course the ensuing contributions to the bond energy.

1.4.6 Transition State procedure

This phrase stands for an analysis method described in ref. [3 (page 907)] and has no relation to transition states in chemical reactions. An extensive discussion of bond energy analysis by ADF is given in [4 (page 907), 5 (page 907)]

The energy associated with a change in charge density, say the relaxation to self-consistency from the sum-of-orthogonal-fragments, can be computed by subtracting final and initial energies, each obtained from the corresponding charge density. For purposes of analysis the change in energy dE can be reformulated as

$$dE = \int dr \left((\rho_{\text{final}}(r) - \rho_{\text{initial}}(r)) \int_{\rho_{\text{initial}}}^{\rho_{\text{final}}} d\rho F[\rho(r)] \right) \quad (1.2.8)$$

F(ρ) is the Fock operator belonging to the charge density ρ

By writing the density difference ρ_{final} - ρ_{initial} a summation over contributions from the different irreducible representations Γ of the molecular symmetry group, an expression is obtained that lends itself for a decomposition of the bond energy into terms from the different symmetry representations:

$$dE = \sum_{\Gamma} \int dr \left(\Delta\rho^{\Gamma}(r) \int_{\rho_{\text{initial}}}^{\rho_{\text{final}}} d\rho F[\rho(r)] \right) \quad (1.2.9)$$

The integral of the Fock operator over the charge density is now approximated by a weighted summation (in fact, a Simpson integration):

$$\int_{\rho_{\text{initial}}}^{\rho_{\text{final}}} d\rho F[\rho(r)] \approx \frac{1}{6}F[\rho_{\text{initial}}] + \frac{2}{3}F[\rho_{\text{average}}] + \frac{1}{6}F[\rho_{\text{final}}] \quad (1.2.10)$$

where

$$\rho_{\text{average}} = \frac{1}{2}(\rho_{\text{initial}} + \rho_{\text{final}}) \quad (1.2.11)$$

The term with the Fock operator due to the average charge density has given rise to the phrase *transition state*. To avoid confusion we will often refer to it as to the transition *field*.

The approximate integral (1.2.10) involves two errors. The first, rather obvious, is the approximation of the exact integral in (1.2.9) by the weighted sum in (1.2.10). Except in pathological cases this approximation is highly accurate.

The second error comes from the fact that the Coulomb and XC potentials in the Fock operator are computed from the *fit* density. This is only an approximation to the true density, while in the original bond-energy expression (energy due to the final density minus energy due to the initial density) no potentials occur and the *exact* charge density can be used. As mentioned before, these fit-related errors are usually small. For the XC potential the true density can be used if one includes the keyword EXACTDENSITY.

All such errors in the total bonding energy are easily corrected by comparing the summation over the Γs with the correct value for the total bonding interaction term. The difference is simply added to the total bond energy, so no true error remains. We only have a (correction) term that can't be split in contributions from the distinct symmetry representations. In the printed bond energy analysis such small corrections are 'distributed' over the other terms by scaling the other terms such that their sum is the correct total value.

1.5 Running the program

1.5.1 Execution of ADF

When ADF has been installed you can run it by supplying appropriate input and starting the 'adf' script, located in $ADFBIN. This script sets up some environment variables and parses the input to see if anything special needs to be done (for example, if the BASIS key is used the adf script will also execute commands to make the appropriate fragment files). You can use this run script both for the serial and parallel versions of the program. For other programs in the package, there are similar run scripts ('band', 'dirac', and so on).

Running the program using the run script involves the following steps:

- Construct an ASCII input file, say "in".
- Run the program by typing (under UNIX):

  ```
  $ADFBIN/adf {-n nproc} < in > out
  ```

  The part between curly brackets is optional, so the shortest application has the format

  ```
  $ADFBIN/adf < in > out
  ```

  Note that the run files in the $ADFHOME/examples directory are UNIX scripts which are executed with:

  ```
  run > out
  ```

- Move / copy relevant result files (in particular TAPE21) to the directory where you want to save them, and give them appropriate names.
- Inspect the standard output file out to verify that all has gone well.

During the run you may inspect the logfile, to see how far the program has proceeded, or whether you should interrupt the calculation.

In the above scheme adf is the name of the run script that invokes the adf.exe program executable. During the installation the script has been put in the same directory where the program executables are generated: $ADFBIN. You may have moved it to another place, or renamed it. We recommend that you adjust your $PATH variable so that you can omit $ADFBIN from the execution command.

To run another program from the ADF suite, just use the appropriate program run script.

The input for the program is read from standard input, and output is written to standard output, redirected in the example above to in and out, respectively.

The part between square brackets is optional and is only meaningful for a parallel program version. The -n flag specifies the number of parallel processes (nproc) to use. If omitted the default applies, which is the value of the environment variable $NSCM, if such variable exist, otherwise it is defined by installation parameters (in the $ADFHOME/settings file, see the Installation Manuals).

The program run scripts have, in fact, more flags and arguments, for special usage. You can get a survey by typing

`$ADFBIN/adf -h` If a parallel version of ADF has been installed you should be aware of a few special aspects of running ADF in parallel. Partially this depends on the platform and on the installation settings.

First of all, you may specify (by command-line options in the run-script and/or by defining suitable environment variables) explicitly how many parallel processes are to be used. Secondly, you should realize that most of the files that you would have in a single-node run are in a parallel run distributed over the parallel processes. Some parts of the file may be identical across the processes while other parts are not and would only after a recombination yield the data of the corresponding single-node file. The normal result files, (standard output, the logfile and the binary result file TAPE21) are complete at the master process.

How to set up a parallel calculation can be found in the Installation Manual.

1.5.2 Files

The ADF program may generate several output / result files, along with the standard output file. The most important one is TAPE21 (.t21 file), the general result file. TAPE21 contains relevant information about the outcome of the calculation. You may want to store this file somewhere under an appropriate name, for future usage. The meaning of any other files that are produced are explained later in this User's Guide.

Any files produced by the program are generated in the local (working) directory where the calculation runs. If you want to keep them, make sure to move them after the calculation has finished to wherever you want to store them.

Files attached to the job, such as fragment files, are by default also assumed to exist in the local directory. You must take care to move or copy required files to that directory before starting the calculation, or to provide via input adequate information to the program where to find the files. In many cases you can specify a complete path to the file.

Most files that are generated by the program, in particular the standard result file that can be used as a fragment file in other calculations, are *binary* files. A binary file should usually not be moved from one machine to another, i.e. it may not be readable by another machine than the one that generated the file, unless the two machines are of the same type. The ADF package provides utilities to convert the ADF binary result files from binary to ASCII, and vice versa, so that you don't have to regenerate your fragment libraries when going to another machine. See the Scripting Section for further details regarding such aspects.

TAPE21 and logfile

Two of the files that are produced by ADF deserve special attention. The first is the general result file TAPE21 (.t21 files). It is a binary file that contains a lot of information about the calculation, such as the one-electron orbitals expressed in the basis functions. It can be used as a fragment file for subsequent calculations (although only TAPE21 files from spin-*restricted* calculations can be used as fragment files). Like all files produced by the program, it is generated in the directory where the job runs. Having done a calculation, you will usually store TAPE21 somewhere under a suitable name so that you can later reuse it, as a fragment file, for a restart, to feed it to an analysis program, and so on.

The second is an ASCII log file, called logfile. It accumulates messages from ADF into a (brief) summary of the run. You can inspect it during the calculation to check how far the calculation has proceeded, whether there are important warnings and so on. At the end of the run this file is copied to the tail of the normal standard output file.

Standard output

ADF is a program that lends itself particularly well for chemical analysis. This is a direct result of the fragment-based approach, where properties of the molecule are related to the properties of the constituent fragments, which is precisely how the chemist thinks. Molecular Orbitals are (optionally) analyzed extensively as how they are composed from occupied and virtual fragment orbitals. This inherently implies a large amount of output. Even computations on small molecules may produce startlingly many pages of output. This is not necessarily so because you can regulate the production of output in detail. Obviously, some kind of *default* production of output had to be implemented. The field of ADF users is so wide and diverse that it is hard to satisfy everybody as regards this default level of output. Depending on your purposes the automatic settings, which determine how much output is generated without instructions to the contrary, may yield boringly many numbers that you just skip through in search for the one value you're interested in, or it may be widely insufficient. Therefore, take notice of the possibilities to regulate output.

Above all, however get familiar with the analysis tools that ADF provides to see in what ways these may help to interpret your results. In a later chapter a global description of output is given as it is normally produced. The chapter below gives an introduction in some of the essential features of ADF, which may be sufficiently different from what you are used to in other Quantum Chemistry codes to deserve your attention.

File names during parallel runs

The adf process of a kid normally runs in a separate directory.

Standard output is a special: the parent writes its normal ('print') output to standard output while the *kids* each write to a file KidOutput.

INPUT AND OUTPUT

2.1 Minimal input

Most keys in the input file for ADF are optional. Default values are used for omitted keys. Assuming the default values are sensible, short input files can often be used. We will examine first the minimal input that is required to run ADF. Having read that part, you can start to do calculations.

The following input will run a geometry optimization on water, using a (almost) minimal input:

```
ATOMS
    O   0   0   0
    H   1   1   0
    H  -1   1   0
End

Basis
End

Geometry
End
```

This is the input for the ADF program. You need to store it in a file, and pass it as standard input to ADF.

For example, assume you have stored the above input in a file in. Also assume that the $ADFBIN directory is in your $PATH. Then you run ADF using the following command:

```
adf <in >out
```

ADF will run, and the resulting output will be stored in the file out. If you examine the contents of this file, you will find that ADF has actually run three times: two create runs, and one geometry optimization. The fragment files produced by the create runs are saved in t21.H and t21.O, for hydrogen and oxygen respectively.

2.1.1 Shell script

Like in the *Examples* (page 397), one can put the call to adf inside a shell script, for example, the content of the file 'example.run' could be like:

```
#! /bin/sh

$ADFBIN/adf << eor
ATOMS
    O   0   0   0
    H   1   1   0
    H  -1   1   0
```

```
End
Basis
End
Geometry
End
eor
```

The shell script 'example.run' needs be executable, if it isn't you will need to make it executable, e.g. `chmod u+x example.run`. The 'example.run' file needs to be executed as a shell script, not as input to ADF.

```
./example.run >example.out
```

2.2 Structure of the Input

Much of the general remarks about input for ADF apply also to related property and analysis programs, which are also described in this document.

2.2.1 Units of length and angle

Geometric lengths and angles are in units defined by:

```
UNITS
   length Angstrom / Bohr
   angle Degree / Radian
End
```

Angstrom and Bohr, respectively Degree and Radian, are recognized strings. Each of the subkeys is optional, as is the key UNITS itself. **Defaults:** Angstrom for lengths, and Degree for angles.

2.2.2 Including an external file

You can include an external ASCII file in the input with the `INLINE` key:

```
INLINE FileName
```

The file name should include the path, either absolute or relative to the run-directory. The content of the file `FileName` is included in the input at the point where the `INLINE` key occurs (the `INLINE` key may occur any number of times in the input).

The `INLINE` feature makes it easy to pack your preferred settings in one file and use them in every run with minimum input-typing effort, or can be used to include files containing the atomic coordinates.

2.2.3 Title, comment

```
TITLE Title
```

Title may be any string. The program combines it (that is, the first approximately 50 characters) with date and time of the job to construct the* job identification.* The job identification is used to stamp an identification on result files, which will be read and printed if such a file is used again, for instance as a fragment file.

The job identification will also be echoed in the output header to identify the current run. By default the date and time are combined with a dummy string. In Create mode the title is first read from the data file that supplies the basis functions etc and can then be overwritten via input.

Note that, contrary to some other programs, ADF does *not* take the first input record as a title.

You can put more remarks in the input file to be echoed in the standard output file; these will not become part of the job identification:

```
COMMENT
   text
   ...
end
```

The text records are copied to the output header, directly after the job identification.

The key COMMENT may occur any number of times; all text blocks are printed in the output header with a blank line between any two text blocks.

2.2.4 General remarks on input structure and parsing

- Most keys are optionals. Defaults values will be used for keys that are not specified in the input

- Keys/blocks can either be *unique* (*i.e.* they can appear in the input only once) or *non-unique*. (i.e. they can appear multiple times in the input)

- The order in which keys or blocks are specified in the input does not matter. Possible exceptions to this rule are a) the content of non-standard blocks b) some non-unique keys/blocks)

- Comments in the input file start with one of the following characters: #, !, :::

  ```
  # this is a comment
  ! this is also a comment
  :: yet another comment
  ```

- Empty lines are ignored

- The input parsing is **case insensitive** (except for string values):

  ```
  # this:
  UseSymmetry false
  # is equivalent to this:
  USESYMMETRY FALSE
  ```

- Indentation does not matter and multiple spaces are treaded as a single space (except for string values):

  ```
  # this:
      UseSymmetry     false
  # is equivalent to this:
  UseSymmetry false
  ```

2.2.5 Keys

Key-value pairs have the following structure:

```
KeyName Value
```

Possible types of keys:

bool key The value is a single Boolean (logical) value. The value can be `True` (equivalently `Yes`) or `False` (equivalently `No`.). Not specifying any value is equivalent to specifying `True`. Example:

```
KeyName Yes
```

integer key The value is a single integer number. Example:

```
KeyName 3
```

float key The value is a single float number. For scientific notation, the E-notation is used (e.g. -2.5×10^{-3} can be expressed as $-2.5E-3$). The decimal separator should be a dot (`.`), and **not** a comma (`,`). Example:

```
KeyName -2.5E-3
```

string key The value is a string, which can include white spaces. Only ASCII characters are allowed. Example:

```
KeyName Lorem ipsum dolor sit amet
```

multiple_choice key The value should be a single word among the list options for that key (the options are listed in the documentation of the key). Example:

```
KeyName SomeOption
```

integer_list key The value is list of integer numbers. Example:

```
KeyName 1 6 0 9 -10
```

float_list key The value is list of float numbers. The convention for float numbers is the same as for Float keys. Example:

```
KeywordName 0.1 1.0E-2 1.3
```

2.2.6 Blocks

Blocks give a hierarchical structure to the input, grouping together related keys (and possibly sub-blocks). In the input, blocks generally span multiple lines, and have the following structure:

```
BlockName
   KeyName1 value1
   KeyName2 value2
   ...
End
```

Headers

For some blocks it is possible (or necessary) to specify a *header* next to the block name:

```
BlockName someHeader
   KeyName1 value1
   KeyName2 value2
   ...
End
```

Compact notation

It is possible to specify multiple key-value pairs of a block on a single line using the following notation:

```
# This:
BlockName KeyName1=value1 KeyName2=value2

# is equivalent to this:
BlockName
   KeyName1 value1
   KeyName2 value2
End
```

Notes on compact notation:

- The compact notation cannot be used for blocks with headers.

- Spaces (blanks) between the key, the equal sign and the value are ignored. However, if a value itself needs to contain spaces (e.g. because it is a list, or a number followed by a unit), the entire value must be put in either single or double quotes:

```
# This is OK:
BlockName Key1=value Key2 = "5.6 [eV]" Key3='5 7 3 2'
# ... and equivalent to:
BlockName
   Key1  value
   Key2  5.6 [eV]
   Key3  5 7 3 2
End

# This is NOT OK:
BlockName Key1=value Key2 = 5.6 [eV] Key3=5 7 3 2
```

Non-standard Blocks

A special type of block is the *non-standard block*. These blocks are used for parts of the input that do not follow the usual key-value paradigm.

A notable example of a non-standard block is the `Atoms` block (in which the atomic coordinates and atom types are defined).

2.3 Structure of the Output

ADF produces two ASCII files: standard output and the log file. The latter is a very concise summary of the calculation's progress during the run. Furthermore, ADF produces and reads binary data files. Most of these files have the so-called KF format. KF stands for Keyed File: KF files are keyword oriented, which makes them easy to process by simple procedures. KF files are Direct Access binary files. Consult the Scripting Section for information about how to use some standard utilities for processing KF files.

The graphical user interface ADF-GUI provides graphical representations of calculated data fields:

> **See also:**
> GUI overview tutorials
> ADF-GUI tutorials

2.3.1 Job Characteristics on standard Output

Input Echo, Output Header

- Copy of the input file, except any InLine records: these are expanded and the contents of the inlinefile replaces the InLine command in the echo.

- Header with the program name, the release number and a copyright statement.

- Directly below the header are printed the job identification, title, and any comments that may have been supplied via input (key COMMENT). The job identification is comprised of the ADF release number and the date and time of the calculation.

Main Job Characteristics

- The Model Parameters such as the Density Functional and relativistic options.

- A list of attached files: restart data files and fragment files.

- The run type: Geometry Optimization, Frequencies...

- (Initial) geometric data: atomic positions, atom types, defined fragments, and the inter-atomic distance matrix.

- The point group symmetry, with a list of the irreducible representations and subspecies.

- The electronic configuration: occupation numbers (if specified), their distribution over spin-α and spin-β, and the net charge of the molecule.

Build Info: Fragments and Function Sets

- Correspondence between fragments in the molecule and the corresponding *master* fragments on the pertaining fragment file. (This output is by default off)

- SFOs: the Symmetry combinations of Fragment Orbitals. The SFOs are the basic conceptual entities for the analysis of MOs and other results.

- The elementary basis functions and the frozen-core levels of the atoms.

Technical Parameters

- Parallelization and vectorization characteristics.

- *Direct* versus *Store-On-Disk* options.

- Update strategy parameters for the SCF procedure.

- General precision settings for numerical integration and neglect-of-small function values (in integral evaluations).

Computational Report

- Numerical integration parameter(s) and the number of generated (symmetry unique) integration points, and partitioning of the points in blocks.

- SCF procedure, at each cycle: for each irreducible representation: the one-electron orbital energies and the occupation numbers for a contiguous sequence of orbitals.

- Gross atomic charges, computed from a Mulliken population analysis.

- Property specific output, like geometry updates, spectroscopic properties.

Exit Procedure

- Normal termination or an error message.

- A list of all files that are (still) open when the exit routine is called. The program closes such files at this point.

- Information about buffered I/O processing during the calculation.

- A check of workspace to see whether all dynamically allocated arrays have been cleaned-up.

- Timing Statistics: a survey of CPU, System (I/O) and Elapsed times spend in various sections of the program.

Logfile

At the end of the calculation the log file is copied (optionally, see key PRINT) to the tail of the standard output file. The log file contains a concise summary of the run.

2.3.2 Log file, TAPE21, TAPE13

The log file (logfile) is generated during the calculation and flushed after (almost) each message that is sent to it by the program. Consequently, the user can inspect it and see what is going on without being delayed by potentially large system I/O buffers. Each message contains date and time of the message plus additional info.

A major part of the messages simply states the name of a procedure. Such messages are sent when the procedure is entered. During the SCF procedure, the SCF errors, which are a measure for non-self-consistency, are written at every cycle. In calculations where the geometry is changing (optimization, frequencies...) each set of new coordinates is sent to the log file (Cartesian, in angstrom and also Z-matrix, if a Z-matrix structure was provided in the input file). Other messages should be self-explanatory.

Be alert on error messages. Take them seriously: inspect the standard output carefully and try to understand what has gone wrong. Be also alert to warnings. They are not necessarily fatal but you should understand what they are about before being satisfied with the results of the calculation. Do not ignore them just because the program has not aborted: in some cases the program may not be able to determine whether or not you really want to do what appears to be wrong or suspicious. If you believe that the program displays erratic behavior, then the standard output file may contain more detailed information. Therefore, in such case save the complete standard output file, together with the logfile, in case we need these files for further analysis.

TAPE21

TAPE21 is the general result file of an ADF calculation. It is a KF file: Direct-Access, binary, and keyword driven. It contains information about the calculation. You can use it as a fragment file in a subsequent calculation on a bigger molecule, where the current one may be a part, or in an analysis program. For more information on TAPE21, see *Appendix on TAPE21* (page 362).

TAPE13

TAPE13 is the checkpoint file for restarts after a crash. It is a concise version of TAPE21, containing only the data the program uses for restarting the calculation. See the RESTART keyword. Like TAPE21, TAPE13 is a binary, keyword driven KF file, see *Appendix on TAPE13* (page 383).

COORDINATES, BASIS SETS, FRAGMENTS

See also:

ADF-GUI tutorial: building molecules , basis set effects , fragments

3.1 Atomic coordinates

3.1.1 Cartesian

You can define the input atomic coordinates in the `Atoms` block key:

```
ATOMS
   {N} AtomType Coords {F=Fragment}
   ...
End
```

N This is an optional integer by which you may number the atoms. The numbers should be 1,2,3, et cetera if any reference is made to them in other parts of input. The reason for this restriction is that ADF numbers the atoms internally according to their occurrence in the input file and it applies this internal numbering when any subsequent references are interpreted.

AtomType The name of an *atom type*. It must begin with the standard one- or two-character symbol for the chemical element (*e.g.* H, He, Li,...). Optionally it may be appended by .text, where text is any string (not containing delimiters). Examples: `H`, `Mn.3`, `Cu.dz-new`.

Note: *Atom type* is not identical to *chemical element*. An atom type is defined by all characteristics of the basic atom to which it in fact refers: the nuclear charge, the basis functions, the frozen core, the density functional and any other features that were applied in generating that basic atom.

Coords The xyz coordinates of the atom (see *Units of length and angle* (page 18)). Example:

```
ATOMS
   O   0.000000   0.000000    0.000000
   H   0.758602   0.000000    0.504284
   H   0.758602   0.000000   -0.504284
END
```

F=Fragment Specifies that the atom belongs to a particular fragment. The fragment name must be of the form `fragtype/n`, where fragtype is the name of one of the types of fragments in the molecule. The integer n, after the slash, counts the individual fragments of that type. The numbering suffix /n is not required if there is only one fragment of that type.

When f=fragment is omitted altogether, the fragment type is taken to be the *atom type* that was specified earlier on the same line. (The numbering /n is then added automatically by the program, by counting the number of times that this single-atom fragment type occurs in the list of atoms).

note: Input items are generally case insensitive. Exceptions are the names of files and directories. Since (to be discussed below) the name of the fragment type as it is defined under atoms (explicitly with the f=option, or implicitly as the name of the atom type) might also directly indicate the fragment file, the specification of fragment types is in principle case-sensitive. Errors may occur if you are sloppy in this respect. However, you must not give different fragment types names that differ only by case: at various places in the program fragment type names are compared in a case-insensitive way

The *INLINE* (page 18) key can be used to include a file containing the atomic coordinates:

```
ATOMS
    INLINE MyMolecule.xyz
END
```

Note: *.xyz* files might include headers. Since the *INLINE* (page 18) key simply copies the content of the file `MyMolecule.xyz` to the ADF input, you have to remove the header from your *.xyz* file.

3.1.2 Z-matrix

It is possible to specify the atomic coordinates in a Z-matrix format.

```
ATOMS Zmatrix
    {N} AtomType Coords {F=Fragment}
    ...
End
```

Coords For Z-matrix coordinates you put first the three connection numbers, then the values of the bond length, bond angle and dihedral angle (see *Units of length and angle* (page 18)). Example:

Ge 2 1 5 2.1 95.3 24.8

defines that a Germanium atom is located with a distance 2.1 Angstrom from the second atom in the input list, that the angle (Ge-atom2-atom1) is 95.3 degrees and that the dihedral angle between the planes (Ge-atom2-atom1) and (atom2-atom1-atom5) is 24.8 degrees.

To avoid any confusion as regards the direction (sign) of the dihedral angle, here is the definition used in ADF: Let the connection numbers for an atom P refer to the atoms Q, R and S, in that order. Choose a local coordinate frame such that Q is at the origin, R on the positive z-axis and S in the xz-plane with a positive x-value. The three Z-matrix coordinates bond length, bond angle and dihedral angle of P are then precisely its spherical coordinates r, q, and -f: the distance to the origin, the angle that PQ makes with the positive z-axis $(0..\pi)$ and the negative of the angle that the projection of PQ on the xy-plane makes with the positive x-axis $(0..2\pi$, or $-\pi..+\pi)$.

The connection numbers and internal coordinate values of the first atom in a Z-matrix have no meaning. Similarly, the second atom requires only a bond-length specification and the third atom only a bond length and a bond angle. However, for each atom three connection numbers are read from input and interpreted, and you must therefore supply zeros for them if they don't refer to any atoms. The corresponding meaningless Z-matrix coordinate values can be omitted. More in general: missing coordinate values are set to zero (also for Cartesian coordinates input). Z-matrix values that are meaningless because they correspond to zero connection numbers are ignored, whatever their value is in the input file.

In a Z-matrix definition the three reference atoms, with respectively 3, 2, and 1 connection numbers equal to zero, do not have to be the first three in the input list. The program will scan the list for any atom that has 3 connection numbers zero, then for one that has only a bond length specification, etc. If the Z-matrix is not properly defined, for instance if more than one atom occurs with all three connection numbers equal to zero, or when not every atom is somehow connected to all others, the program will abort.

If a subgroup of the true nuclear symmetry is used and Z-matrix format is used for the coordinates, the program will place the atoms in the standard Z-matrix frame: first atom at the origin, second on the positive x-axis, third in the xy-plane with positive y-value.

Dummy Atoms

Dummy atoms may be useful in the construction of a Z-matrix, for instance to obtain a set of internal coordinates that reflect the symmetry of the molecule better. They may also be useful in a Z-matrix to avoid an ill-defined dihedral angle, which occurs when three (almost) co-linear atoms span either of the two planes that define the angle. In geometry optimizations this must absolutely be avoided if such internal coordinates are used as optimization parameters.

Dummy atoms are input with the chemical symbol xx. XX-type atoms can be inserted in the list of atoms like any other atom types. The name (xx) can have a suffix of the form .*text*. No fragment files must be supplied for dummies.

There are no symmetry constraints on the positions of the dummies. Dummy atoms may be placed asymmetrically. If the atomic coordinates are input as Cartesians, any dummy atoms are irrelevant. Their coordinates will be printed but otherwise they are ignored. The dummies serve only to set up the Z-matrix in a proper way.

3.1.3 Mixed Cartesian and Z-matrix

The key ATOMS can also be used to supply coordinates in a format that gives the values for the Cartesian coordinates *and* the connection matrix, which defines a Z-matrix.

```
ATOMS ZCart
 {N} AtomType Coords {F=Fragment}
 ...
End
```

ZCart Signals this particular format for the coordinates

Coords As for Z-matrix input: three integers and three real values. The integers are the connection numbers that define the Z-matrix structure, but the reals are the *Cartesian* coordinates.

With ZCart input, the z-matrix is internally generated from the Cartesian coordinates and the connection numbers.

This feature is convenient when for instance Cartesian coordinates are easily available but you want to run a Geometry Optimization in *internal* coordinates, for which a Z-matrix structure is required.

The zcart option comes in handy also to satisfy symmetry-related orientation requirements when you basically wish to use Z-matrix coordinates.

With zcart input the program defines the *type* of coordinates in the input file as *Cartesian*. This is significant in Geometry Optimizations, where the optimization variables are by default taken as the input coordinate type.

3.1.4 MOPAC format

It is possible to specify the atomic coordinates in MOPAC format:

```
ATOMS MOPAC
   {N} AtomType Coords {F=Fragment}
   ...
End
```

Coords The MOPAC style input requires that the records in the data block have the following format:

```
AtomType distance idist angle iangle dihedral idehedral
```

The three internal coordinate values (distance, angle, dihedral) are each followed directly by the connection number.

3.1.5 Orientation of Local Atomic Coordinates

As discussed before the atomic positions are input with the key ATOMS. One option has thus far not been mentioned: the possibility to redefine the local coordinate frame of an atom.

```
ATOMS {type of coordinates}
   {n} AtomType coordinates {F=fragment} {Z=xx yy zz} {X=xx yy zz}
   ...
end
```

Except for the z= option all aspects have been examined already before.

z=xx yy zz defines a reorientation of the local atomic z-axis; it is interpreted as a direction vector with components (xx,yy,zz) pointing away from the atom. In the local, reoriented frame the local atomic x-axis will be rotated to the plane defined by the directions of the molecular z-axis and the local atomic z-axis. This feature can be used only for single-atom fragments (otherwise it is ignored). Its purpose is to give more flexibility in the analysis of the final molecular orbitals in terms of the atomic orbitals. In such a case it may be very helpful to redefine the orientation of say the p-orbitals of an atom. For instance, you may orient all p-orbitals towards the origin by specifying for each atom z= -x -y -z (with x,y,z the coordinates of that atom). By default the local and molecular z-axes are identical.

x=xx yy zz defines a reorientation of the local atomic x-axis; it is interpreted as a direction vector with components (xx,yy,zz) pointing away from the atom. Together with the z vector this defines the xz plane. The y axis is then given by the vector product z * x. This is used for analysis (see orientation of the z-axis).

3.1.6 ASCII Output Files with Atomic Coordinates

You may want to have a special result file that contains the atomic coordinates corresponding to all the geometries processed in the calculation, for instance to feed it to a 'movie' generator to display the development of an optimization run. This is regulated with the key FILE:

```
FILE filetype filename { filetype2 filename2 }
```

filetype Specifies the format of the output. Currently supported are three varieties: `mol` and `xyz`

filename The file to which the output is written; the file should not yet exist. The name may include a full or relative path with respect to the directory where the calculation runs.

The same input record may contain any number of pairs-of-arguments, for instance to specify that both a mol-type *and* a xyz-type result file are to be generated. The key may also occur more than once in the input stream, in which case the argument lists are effectively all concatenated (by the program).

3.1.7 Symmetry Key

Together with the point group symmetry, a tolerance parameter can be supplied.

```
SYMMETRY {symbol} {tol=tolerance}
```

symbol The Schönfliess symmetry symbol. A complete list of allowed values for this argument is given in *Appendix Symmetry* (page 360).

To disable symmetry (C_1 symmetry group) you can specify:

```
SYMMETRY NOSYM
```

tolerance The tolerance (absolute deviation in the Cartesian coordinates) for atomic positions being symmetry equivalent. The same tolerance applies to check the mapping of fragments on attached fragment files with the actual fragments. If the tolerance is specified it is interpreted in the chosen unit of length (units). The **default** tolerance is 0.001 Angstrom and the maximum is 0.1 a.u.

Input atomic coordinates that are slightly (within the tolerance) off from their correct positions are adjusted by the program.

The point group symmetry specified in input with a Schönfliess type symbol puts restrictions on the orientation of the atomic system. Unless the input-specified symmetry equals the true symmetry of the nuclear frame (in which case ADF will adjust the orientation of the molecule, if necessary), the user must take care of this by supplying the *Cartesian* coordinates (in the appropriate orientation).

3.2 Basis sets and atomic fragments

See also:
What basis set should I use in ADF? (page 327)
Basis set file format (page 343)

3.2.1 STO basis sets

The basis functions used in ADF are commonly known as Slater Type Orbitals (STOs). The ADF package comes with a database of STO basis sets. The basis sets are simple ASCII files and they are located in the directory `$ADFHOME/atomicdata`. A description of the basis set file format can be found in the appendix *Basis set file format* (page 343).

A basis set can roughly be characterized by its size (single-, double-, triple-zeta; with or without polarization) and by the level of frozen core approximation. The standard basis sets available in ADF are:

- **SZ** Minimal basis sets: single-zeta without polarization. The exponents of the functions correspond to the standard STO-3g basis sets used in programs that employ Gaussian type basis functions. Type-SZ files are provided only for the lighter elements, up to Kr.

- **DZ** Double-zeta basis sets without polarization functions. A triple-zeta set is used for the 3d shells of the first row transition metals, the 4f shells of the Lanthanides, and the 5f shells of the Actinides. In all these cases a double-zeta set provides a rather poor expansion basis for the true (numerically computed) atomic orbital.

- **DZP** Double zeta polarized basis. The basis sets in DZP are derived from DZ, extended with a polarization function. This type of basis sets is thus far provided only for the elements up to Ar, and for the 4p series Ga through Kr.

- **TZP** Triple-zeta basis sets. A polarization function is added for H through Ar and for Ga through Kr (from DZP)

- **TZ2P** Triple-zeta with two polarization functions, for H through Ar and Ga through Kr (from DZP). Note that the TZ2P basis set files are provided only for the lighter elements, up to Kr. The ZORA/TZ2P basis set files are provided for all elements. Typically for all elements one polarization function is added compared to the corresponding TZP basis set. Note, however, that TZ2P will not always give you extra basis functions for most lanthanide and actinide frozen core basis sets.

In addition to the standard basis sets, the database contains directories with special basis sets:

- **TZ2P+** For transition metals Sc-Zn and lanthanides (ZORA) only: as TZ2P, but with extra d-STO (3d metals), and extra f-STO (lanthanides, ZORA)

- **ZORA** contains basis sets designed for *ZORA relativistic calculations* (page 72) (relativistic calculation have special basis set requirements, especially in the core region). ZORA basis sets with frozen core should be used **exclusively** in relativistic calculations with the ZORA approach, while all-electron ZORA basis sets can be used for both relativistic and non-relativistic calculations. The ZORA/QZ4P basis set can be loosely described as core triple zeta, valence quadruple zeta, with four sets of polarization functions.

- **ET** contains several even tempered basis sets which enables one to go to the basis set limit, such as ET/ET-pVQZ, ET/ET-QZ3P, ET/ET-QZ3P-1DIFFUSE, ET/ET-QZ3P-2DIFFUSE, ET/ET-QZ3P-3DIFFUSE. The accuracy of the smallest basis set in this directory can loosely be described as quadruple zeta in the valence with three polarization functions added. This directory also contains basis sets with extra diffuse functions. In Response calculations one should use such large basis sets. Very diffuse functions are absolutely necessary to get good results for excitation energies corresponding to high lying orbitals.

- **AUG** contains several augmented standard basis sets which enables one to get reasonable results for excitation energies with relatively small basis sets, such as AUG/ASZ, AUG/ADZ, AUG/ADZP, AUG/ATZP, AUG/ATZ2P.

Furthermore, in `$ADFHOME/atomicdata` you will find:

- **Special/AE** contains non-relativistic basis sets for all-electron calculations. However, these files cannot be used as such, because they don't contain any fit sets. They serve as starting point for the development of (new) basis sets. For some of the all-electron sets appropriate fit sets have already been generated. The corresponding data base files can be found in the appropriate sub-directories SZ, DZ, DZP, et cetera.

- **Special/Vdiff** contains non-relativistic basis sets that include very diffuse functions. These were recommended to be used for Response calculations. Very diffuse functions are absolutely necessary to get good results for excitation energies corresponding to high lying orbitals. Recommendation: use the even tempered basis sets in the ET directory, since these basis sets are better.

- **Special/MDC** contains non-relativistic basis sets with optimized fit functions especially useful for accurate Multipole Derived Charges. These are available only for a limited number of basis sets.

- **Dirac** contains the input files for the DIRAC auxiliary program (see the RELATIVISTIC keyword).

- **Band** contains input files for the BAND program (see the BAND User's Guide)

- **ForceFields** contains force field files to be used in the QM/MM functionality. Their structure and contents are described in the QM/MM manual. See also the pdb2adf utility (ADF QM/MM documentation), which transforms a PDB file into an ADF input file, for use with QM/MM.

Frozen core

Multiple occurrences of one chemical element in the same basis set sub-directory correspond to different levels of the *frozen core approximation* (page 8). Manganese for instance may have a basis set for an atom with a frozen 2p shell and another one with a frozen 3p shell. The file names are self-explanatory: Mn.2p stands for a data file for Manganese with frozen core shells up to the 2p level. An all-electron basis set would correspond to a file that has no frozen-core suffix in its name.

Electronic configurations specific basis sets

Another type of multiple occurrence of one element in one basis set directory may be found when basis sets have been developed for different electronic configurations: the Slater-type basis sets are fitted then to numerical orbitals from runs with different occupation numbers. Currently this applies only for Ni (in `$ADFHOME/atomicdata/ DZ`, `TZP` and `TZ2P`), where basis sets are supplied for the d8s2 and the d9s1 configurations respectively. Since in earlier releases only the d8s2 variety was available, the names of the basis set files are Ni.2p (for d8s2) and Ni_d9.2p, and likewise Ni.3p and Ni_d9.3p.

References on basis sets

- [*320* (page 925)] E. van Lenthe and E.J. Baerends, *Optimized Slater-type basis sets for the elements 1-118.* Journal of Computational Chemistry 24, 1142 (2003) (https://doi.org/10.1002/jcc.10255)

- [*321* (page 925)] D.P. Chong, E. van Lenthe, S.J.A. van Gisbergen and E.J. Baerends, *Even-tempered Slater-Type orbitals revisited: From Hydrogen to Krypton.* Journal of Computational Chemistry 25, 1030 (2004) (https://doi.org/10.1002/jcc.20030)

- [*322* (page 925)] D.P. Chong, *Augmenting basis set for time-dependent density functional theory calculation of excitation energies: Slater-type orbitals for hydrogen to krypton.* Molecular Physics 103, 749 (2005) (https://doi.org/10.1080/00268970412331333618)

Older references for STO basis sets are Refs. [*336-338* (page 926)]. See also the paper by Raffennetti on design and optimization of even-tempered STO basis sets [*317* (page 925)]. The paper by Del Chong describes completeness profiles as a visual tool in estimating the completeness of a basis set [*318* (page 925)]. Finally, Zeiss and coworkers [*319* (page 925)] describe field-induced polarization functions for STOs. These are useful for defining basis sets with diffuse functions for (hyper)polarizability and other property calculations.

The procedure for the usage and optimization of fit functions is described by Baerends et al. [*308* (page 925)].

For documentation on how to make custom even-tempered basis/fit sets, see the old ADF 2014 documentation (http://www.scm.com/Doc/Doc2014/ADF/ADFUsersGuide/ADFUsersGuide.pdf).

3.2.2 Available basis sets

ADF has optimized STO basis sets for the whole peridic table (Z=1-118).

The next tables give an idication which all electron (ae) and frozen core (fc) standard basis sets are available for the different elements in ADF.

Table 3.1: Available standard basis sets for non-relativistic (non-rel) and ZORA calculations H-Kr (Z=1-36)

Element	ae or fc	SZ, DZ	DZP	TZP, TZ2P	TZ2P+	QZ4P, ET	AUG
H-He (Z=1-2)	ae	Yes	Yes	Yes		Yes	Yes
Li-Ne (Z=3-10)	ae	Yes	Yes	Yes		Yes	Yes
	.1s	Yes	Yes	Yes			non-rel
Na-Ar (Z=11-18)	ae	Yes	Yes	Yes		Yes	Yes
	.1s .2p	Yes	Yes	Yes			non-rel
K-Ca (Z=19-20)	ae	Yes	Yes	Yes		Yes	Yes
	.2p .3p	Yes	Yes	Yes			non-rel
Sc-Zn (Z=21-30)	ae	Yes		Yes	Yes	Yes	Yes
	.2p .3p	Yes		Yes	Yes		non-rel
Ga-Kr (Z=31-36)	ae	Yes	Yes	Yes		Yes	Yes
	.3p .3d	Yes	Yes	Yes			non-rel

Table 3.2: Available standard basis sets for non-relativistic calculations Rb-Cm (Z=37-96)

Element	fc	DZ, TZP
Rb-Sr (Z=37-38)	.3p .3d .4p	Yes
Y-Cd (Z=39-48)	.3d .4p	Yes
In-Ba (Z=49-56)	.4p .4d	Yes
La-Lu (Z=57-71)	.4d .5p	Yes
Hf-Hg (Z=72-80)	.4d .4f	Yes
Tl-Rn (Z=81-86)	.4d .4f .5p .5d	Yes
Fr-Ra (Z=87-88)	.5p .5d	Yes
Ac-Cm (Z=89-96)	.5d	Yes

Table 3.3: Available standard basis sets for ZORA calculations Rb-Og (Z=37-118)

Element	ae or fc	DZ, TZP, TZ2P	TZ2P+	QZ4P
Rb-Sr (Z=37-38)	ae	Yes		Yes
	.3p .3d .4p	Yes		
Y-Cd (Z=39-48)	ae	Yes		Yes
	.3d .4p	Yes		
In-Ba (Z=49-56)	ae	Yes		Yes
	.4p .4d	Yes		
La-Yb (Z=57-70)	ae	Yes	Yes	Yes
	.4d .5p	Yes	Yes	
Lu (Z=71)	ae	Yes		Yes
	.4d .5p	Yes		
Hf-Hg (Z=72-80)	ae	Yes		Yes
	.4d .4f	Yes		
Tl (Z=81)	ae	Yes		Yes
	.4d .4f .5p	Yes		
Pb-Rn (Z=82-86)	ae	Yes		Yes
	.4d .4f .5p .5d	Yes		
Fr-Ra (Z=87-88)	ae	Yes		Yes
	.5p .5d	Yes		
Ac-Lr (Z=89-103)	ae	Yes		Yes
	.4f .5d	Yes		
Rf-Cn (Z=104-112)	ae	Yes		Yes
	.4f .5d .5f	Yes		
Nh-Og (Z=113-118)	ae	Yes		Yes
	.5d .5f	Yes		

For heavier elements, from Rb on, the non-relativistic all electron basis sets are missing. In the ZORA basis sets directory you will find all-electron basis sets for all elements (Z = 1-118), which also could be used in non-relativistic calculations. Note, however, that these basis sets were optimized for ZORA calculations, which means that non-relativistic calculations will not always give you the expected accuracy. Non-relativistically optimized basis sets for the heavier elements are provided in a separate directory AE, which contains basis sets of single-, double- and triple-zeta quality indicated respectively by suffixes 'sz', 'dz', and 'tz'. The files in Special/AE/ are not complete basis set files, because they don't contain fit sets (the usage and relevance of fit functions is explained later).

Basis sets directories

Basis sets can be found in the directory $ADFHOME/atomicdata, for non-relativistic calculations in the sub-directories SZ, DZ, DZP, TZP, TZ2P, TZ2P+, for ZORA calculations in ZORA/SZ, ZORA/DZ, ZORA/DZP, ZORA/TZP,

ZORA/TZ2P, ZORA/TZ2P+, ZORA/TZ2P-J, ZORA/QZ4P, ZORA/QZ4P-J, the augmented basis sets can be found in AUG/ASZ, AUG/ADZ, AUG/ADZP, AUG/ATZP, AUG/ATZ2P, and the even tempered basis sets in ET/ET-pVQZ, ET/ET-QZ3P, ET/ET-QZ3P-1DIFFUSE, ET/ET-QZ3P-2DIFFUSE, ET/ET-QZ3P-3DIFFUSE. All electron basis sets can be used in non-relativistic and ZORA calculations.

Basis sets acronyms

- SZ: single zeta

- DZ: double zeta

- DZP: double zeta + 1 polarization function

- TZP: valence triple zeta + 1 polarization function

- TZ2P: valence triple zeta + 2 polarization function

- TZ2P+: = TZ2P + extra d (3d metals) or extra f (lanthanides)

- QZ4P: valence quadruple zeta + 4 polarization function, relativistically optimized

- ET: even tempered

    - pVQZ, QZ3P: valence quadruple zeta + 3 polarization function, even tempered

    - QZ3P-nD: = QZ3P + n diffuse sets of s, p, d, and f functions, even tempered

- AUG: augmented

    - ASZ, ADZ, ADZP, ATZP, ATZ2P: augmented for use in TDDFT

- TZ2P-J, QZ4P-J: for use in ESR hyperfine or NMR spin-spin couplings

    - TZ2P or QZ4P + extra tight (mainly 1s) functions

All electron or frozen core

- element name (without suffix): all electron

- .1s frozen: 1s

- .2p frozen: 1s 2s 2p

- .3p frozen: 1s 2s 2p 3s 3p

- .3d frozen: 1s 2s 2p 3s 3p 3d

- .4p frozen: 1s 2s 2p 3s 3p 3d 4s 4p

- .4d frozen: 1s 2s 2p 3s 3p 3d 4s 4p 4d

- .4f frozen: 1s 2s 2p 3s 3p 3d 4s 4p 4d 4f

- .5p frozen: 1s 2s 2p 3s 3p 3d 4s 4p 4d 5s 5p (La-Lu)

- .5p frozen: 1s 2s 2p 3s 3p 3d 4s 4p 4d 4f 5s 5p (other)

- .5d frozen: 1s 2s 2p 3s 3p 3d 4s 4p 4d 4f 5s 5p 5d

- .6p frozen: 1s 2s 2p 3s 3p 3d 4s 4p 4d 4f 5s 5p 5d 6s 6p (Ac-Lr)

- .5f frozen: 1s 2s 2p 3s 3p 3d 4s 4p 4d 4f 5s 5p 5d 5f 6s 6p

3.2.3 The Basis Key

The basis set can be specified in the input via the `BASIS` key block:

```
BASIS
   Type bastyp
   Core coretyp
   Path apath
   Atom atompath
   ...
   FitType fittyp
   CreateOutput FileName
End
```

All subkeys are optional. For most calculations you need only to set the `Type` and `Core` subkeys.

Type bastyp `bastyp` is the type of basis set to use, and must correspond to the name of a directory in `$ADFRESOURCES` (= `$ADFHOME/atomicdata`), or within `$ADFRESOURCES/ZORA` for ZORA calculations. Valid standard basis set types are: **SZ**, **DZ**, **DZP**, **TZP**, **TZ2P**, and **QZ4P**. More basis set types can be found in the `$ADFRESOURCES` directory, such as the TZ2P+ basis set, the augmented basis sets AUG/ASZ, AUG/ADZ, AUG/ADZP, AUG/ATZP, AUG/ATZ2P, and the even tempered basis sets ET/ET-pVQZ, ET/ET-QZ3P, ET/ET-QZ3P-1DIFFUSE, ET/ET-QZ3P-2DIFFUSE, ET/ET-QZ3P-3DIFFUSE. ZORA will be included only for the standard basis set types: SZ, DZ, DZP, TZP, TZ2P, and QZ4P, and if the calculation is a ZORA calculation. In case one of the standard basis set types is used, but no basis set of the specified type is available, ADF will try to use a larger basis set.

Default: DZ.

Core coretyp `coretyp` is the type of frozen core to use. Allowed values: **None**, **Small**, **Medium**, **Large**. If no basis set with frozen core is available, an all electron basis set will be used. If there is only one basis set with frozen core, Small, Medium and Large are identical. If there are two basis sets with frozen core, Medium and Large are identical.

Default: Large.

Path apath `apath` is an alternative directory with basis sets to use. ADF looks for appropriate basis sets only within this directory.

Default: $ADFRESOURCES

Atom atompath In this subkey `Atom` should be replaced by the name of the atomic fragment for which you want to specify the basis, for example 'O'.

Use this key to specifically select a basis set for this atom:

- an absolute path to a basis file (for example `$ADFRESOURCES/DZ/O.1s`)

- a relative path to a basis file (for example `DZ/O.1s`)

- a filename within the Type directory (for example `O.1s`)

An absolute path will always be used as specified.

A relative path is relative to the value of the PATH (or PATH/ZORA) subkey.

A filename is always relative to PATH/Type or PATH/ZORA/TYPE directory.

The relative path or filename will automatically switch to a ZORA basis set only in case of the standard basis set types: SZ, DZ, DZP, TZP, TZ2P, and QZ4P, and if the calculation is a ZORA calculation.

You can have one Atom subkey for each basic atom type in your input.

Since you pick explicitly the file to use, you are responsible for choosing a reasonable basis set.

FitType fittyp `fittyp` is the type of auxiliary fit set to use, and must correspond with the name of the directory as used within `$ADFRESOURCES`, or within `$ADFRESOURCES/ZORA` for ZORA calculations. Note that this is an expert option. For all atoms the fit set will be changed if this key is used. The fit set for a given atom

is then taken from the all-electron basis set file for the same element as is the atom. Typical usage: one might want a larger fit set than is present on the basis file, and then fittyp could be: ZORA/QZ4P. More valid fit set types can be found in the $ADFRESOURCES directory. If the requested directory or fit set is not available, then this option might fail without warning. It is absolutely a bad idea to take a fit set from another basis set file, corresponding to some frozen core level, and use that in an all-electron basis set: this will give significant errors and make results worthless.

Default: the auxiliary fit set that is available in the requested basis file.

CreateOutput FileName Use the CreateOutput option to change where the output from ADF create runs and the Dirac program goes. If it is not present, it will go to standard output. The special value 'NONE' for FileName makes it disappear, and any other value will be used as a file name in which to save the output.

Warning: Do not include the *Fragments* (page 39) or *Corepotentials* (page 49) keys when using the Basis key.

3.2.4 Automatic mode

The following input will run a geometry optimization on water, using a (almost) minimal input:

```
ATOMS
    O   0   0   0
    H   1   1   0
    H  -1   1   0
End

Basis
    Type TZP
End

Geometry
End
```

- The `ATOMS` block key specifies the geometry of the system;

- the `GEOMETRY` key instructs ADF to perform a geometry optimization;

- the `BASIS` block key instructs ADF to run the *create runs* automatically, using a TZP basis sets.

The *automatic mode* will be used when the `Basis` key is present in the input.

In *automatic mode* ADF will first create fragment files for all the basic atom fragments found in the `ATOMS` key block. Normally this means that for each atom type in your molecule a fragment file will be created.

You may have different fragments with the same atom: add a dot and a name (without spaces) after the name of the element, as described in the *ATOMS key* (page 25). For example: H.1 and H.2. In this example two fragment files will be created: one for the H.1 fragment and one for the H.2 fragment. Using the ATOM subkey you may assign different basis sets to these fragments. Another consequence is that the H.1 and H.2 atoms will never be symmetry equivalent to each other.

In case of a relativistic calculation, the DIRAC program will also be run automatically, and the create runs will include the correct relativistic key and corresponding basis sets. For **ZORA** calculations, ADF first tries to locate a special ZORA basis set. If this does not succeed it will use a normal basis set if the required basis set does not use a frozen core.

The resulting fragment files will be named t21.atom, with 'atom' replaced by the names of the basic atoms present. In case of a relativistic calculation, the corepotentials will be stored on t12.rel.

3.2.5 Create mode

In Create mode the input file can be extremely simple. First, the geometry is trivial: one atom at the origin. Indeed, no coordinates etc. are read from input; any such items are ignored.

Second, the problem is computationally so simple that default settings for precision aspects, such as convergence criteria and levels of numerical integration accuracy, are internally defined to be much more stringent than in normal calculations. These aspects don't have to be looked after.

In Create mode you need only a one-line input file of the following form:

```
CREATE Atomtype Datafile
```

Create is the keyword. The remainder of the record (atomtype datafile) is the argument.

Atomtype is a name for the basic atom that you want to create. The program reads and interprets this name. Therefore, the name must begin with the standard chemical symbol (H, He, Li, ...) of the element to be created. Optionally the name may have an suffix of the form .text. The suffix begins with a period (.); the part after the period (text) is at your discretion as long as it does not contain a delimiter. A few examples:

Table 3.4: Examples of appropriate and inappropriate atom type names used with the keyword create.

appropriate names	inappropriate names
K	Si-with-core (no period after the chemical symbol)
Li.newbasis	$HOME/atomicdata/C.dzp (not beginning with the chemical symbol)
P.1992/Feb./30	Ga.nocore,smallbasis (contains a comma (a delimiter))
	Sodium.2s (Sodium is not the *symbol* for this element (Na))

Datafile specifies the data file that contains the basis set and related items. It may contain a full path if the file does not reside in the working directory of the job. The datafile part is optional. If you omit it, ADF assumes that the file name is identical to the atom type name, i.e. Create Atomtype is equivalent to and interpreted as Create Atomtype Atomtype In view of the restrictions that apply to the atom type name, the option to use the short form can only be used if the file name has the appropriate format. To make the input file easier to understand for a human reader you may, for Datafile, also type file=Datafile, where file= must be typed as such, and datafile is the name of the file.

So you could have a very simple calculation as follows (the 'creation' of a Carbon atom);

```
$ADFBIN/adf << eor
 Create C.dzp
eor
```

The presence of the keyword `create` sets the computational mode of ADF to: *create a basic atom*. The argument (C.dzp) is then analyzed and found to have as initial part C, telling ADF that we'll be creating a Carbon atom. Since the file-specification part is missing, the data file with the basis set must be a file with the name C.dzp.

More often you will directly address a basis set file that is not local, but located in `$ADFHOME/atomicdata`. The script could then be:

```
$ADFBIN/adf << eor
 Create C $ADFHOME/atomicdata/DZ/C.1s
eor
```

Here you address the file 'C.1s' in the sub-directory DZ/ (this contains basis sets of double-zeta quality). Examine logfile and out to check that everything has gone well.

A considerable number of basis set files are included in the ADF. You can also create basic atoms corresponding to so-called *Alternative Elements*, with for instance a non-integer nuclear charge or a different mass. See the next section.

3.2.6 Ghost Atoms, Non-standard Chemical Elements

The atom type names used under atoms (and in the create record) must begin with the standard chemical element symbol (H, He, Li...). The program uses this to deduce the nuclear charge and other elemental properties.

For the standard elements one can redefine the atomic mass (for instance to define a suitable isotope).

It is also possible to define an artificial chemical element with user-specified properties. Such new elements are denoted *Alternative Elements*; and may for instance have a non-integer nuclear charge. The chemical symbol of an Alternative Elements is Gh (for ghost) or J: either one is okay.

The `BASIS` key recognizes elements denoted with Gh.atom in the ATOMS key as being ghost atoms. If one does not specifically select a basis set for this ghost atom, the all electron basis set for the atom is selected in the creation of the ghost atom using the type of basis set chosen with the BASIS key. The atom name must begin with the standard one- or two-character symbol for the chemical element: Gh.H, Gh.He, Gh.Li, and so on. Optionally it may be appended by .text, where text is any string (not containing delimiters). Examples: Gh.H, Gh.Mn.3, Gh.Cu.dz-new.

You can create J-type or Gh-type basic atoms and use them subsequently as fragments in a molecule.

Automatic mode

To make different isotopes or elements one can use the `ATOMPROPS` key in combination with the `BASIS` key. Typical use would be for the nuclear mass.

```
ATOMPROPS
   Atom.name {m=mass} {q=Q}
End
```

mass The atomic mass, in atomic mass units, which will then override the default value for the indicated chemical element. Can be used, for example, for the calculation of isotopes. If not supplied for a J-element it will be set to the atomic mass of the standard chemical element with nuclear charge A, where A equals Q rounded to the nearest integer, but not smaller than 1 and not larger than 118.

Q The nuclear charge. The q= option *must* be used for a J-element. It must *not* be used for *standard* chemical elements.

name In Atom.name the first letter after the dot should be a capital.

Example: three different hydrogen isotopes:

```
Atoms
   N        0.000000    0.000000    0.010272
   H       -0.471582   -0.816803    0.407861
   H.D      0.943163    0.000000    0.407861
   H.T     -0.471582    0.816803    0.407861
End
AtomProps
   H.D m=2.014101778
   H.T m=3.01604927
End
Basis
   Type TZP
End
```

Create mode

For the standard elements one can redefine the atomic mass (for instance to define a suitable isotope).

```
CREATE H {m=value} datafile
```

value The atomic mass, which will then override the default value for the indicated chemical element.

The nuclear charge of an Alternative Element is not pre-defined, and *must* therefore be specified in the Create run. The atomic mass is optionally supplied.

```
CREATE J.NewElement q=Q {m=mass} datafile
```

J.NewElement The atom type name, beginning with the *alternative* chemical element symbol J. It has an (optional) suffix of the form .text, completely similar to the construction of atom type names from standard chemical element symbols.

Q The nuclear charge. The q= option *must* be used for a J-element. It must *not* be used for *standard* chemical elements.

mass The atomic mass, in atomic mass units. If not supplied it will be set to the atomic mass of the standard chemical element with nuclear charge A, where A equals Q rounded to the nearest integer, but not smaller than 1 and not larger than 118.

datafile The Create data file.

If you want to use the Alternative Element feature you'll have to construct your own Create data file, suited to the Alternative Element you have in mind. The appendix *Basis set file format* (page 343) describes the format of such a file.

Use as fragment

J-type basic atoms can be used like any other basic atoms to build up larger fragments and molecules. In fact, J (or Gh) can be considered just one more chemical symbol along with the 118 traditional ones. The element J has no pre-defined properties. Therefore you have to specify them where appropriate (c.f. the nuclear charge and atomic mass).

You may have different J-elements in a molecule, with different nuclear charges for instance. Yet, they must be denoted with the same chemical symbol J; the difference can only be made clear by the .text suffix in the atom type name. It's convenient to make this suffix a concise but clear description of the main characteristics.

3.2.7 Nuclear Model

By default in ADF a point charge model is used for the nuclear charge distribution. Alternatively, one can use a spherical Gaussian nuclear charge distribution model, see Ref. [*270* (page 922)]. Nuclear finite size effects can have large effects on hyperfine interactions (ESR A-tensor, NMR spin-spin coupling) if heavy atoms like, for example, Mercury (Hg), are involved. In Ref. [*270* (page 922)] it was asserted that the isotropic J-couplings (parameters in NMR spin-spin coupling) are typically reduced in magnitude by about 10 to 15 % for couplings between one of the heaviest NMR nuclei and a light atomic ligand, and even more so for couplings between two heavy atoms. This Ref. [*270* (page 922)] gives more details on the parameters used in the Gaussian nuclear charge distribution model. Note that one needs basis sets with very tight functions to see any effect of using a finite size of the nucleus instead of a point nucleus. Such basis sets can be found for all elements in `$ADFRESOURCES/ZORA/TZ2P-J` and `$ADFRESOURCES/ZORA/QZ4P-J`, and for some elements in `$ADFRESOURCES/ZORA/jcpl`, which are basis sets especially designed for ESR hyperfine and NMR spin-spin coupling calculations.

A Gaussian nuclear charge distribution will be used if one uses the `NUCLEARMODEL` key with:

```
NUCLEARMODEL gaussian
```

NUCLEARMODEL nuclearmodel The argument nuclearmodel of the key NUCLEARMODEL can be 'pointcharge' (default) or 'gaussian'. It should be included in the Create run of an atomic calculation and in the molecular calculation. If the BASIS key is used it will be automatically added in the Create run of the atoms. If this key is absent a point charge nuclear model is used.

In the ADF output parameters will be shown for the Gaussian nuclear charge distribution if one includes in the input for ADF:

```
PRINT Nuclei
```

Starting from ADF2013 ADF also uses a finite distribution of the nuclear magnetic dipole moment for the calculation of the A-tensor.

3.3 Molecular fragments

3.3.1 Fragment mode

In Fragment mode you have to specify: (1) the atomic positions and (2) how the total system is built up from fragments. We recommended to specify also (3) the point group symmetry.

Example of an input file for the C2H4 molecule:

```
ATOMS
   C    0       0    0.668
   C    0       0   -0.668
   H    0.927   0   -1.203
   H   -0.927   0   -1.203
   H    0.927   0    1.203
   H   -0.927   0    1.203
end

fragments
   C TAPE21c.dzp
   H TAPE21h.dzp
end

symmetry D(2h)
```

Three keys are used: *atoms* (page 25), *fragments* (page 41) and *symmetry* (page 28). The first two are block keys.

atoms defines the atomic positions: each record in the data block contains the chemical symbol of an atom followed by its Cartesian coordinates in Angstroms (Z-matrix type input of atomic positions is also possible).

fragments lists the fragment files each record contains a fragment *type* followed by the corresponding fragment *file*. In the example the files are *local* files. Files in other directories are addressed by giving the complete file path. Note: if a *parallel* calculation is performed, be sure that each 'kid' finds the specified fragment files. This will usually require that the files are *not* local to the job, but first be moved to some shared volume, and that the references to the fragment files in the input contain the full path. An alternative is to ensure that the (local) files in the parent directory are copied first to the 'kid' directories before the parallel calculation starts.

symmetry specifies the point group symmetry by a Schönfliess type symbol. *Symmetry* (page 360) contains a complete list of all Schönfliess symbols that are recognized by ADF. If no symmetry is specified ADF will use the true symmetry of the nuclear frame. If *(electric) fields* (page 117) are used, see later, the computed symmetry will take this into account. Note that the computed symmetry may not occur in the list of allowed symmetries

(see *Symmetry* (page 360)), in which case you have to explicitly specify the (lower) point group symmetry you wish to apply.

The atomic coordinates must conform to the point group symmetry; the program will check this and abort if the atomic system does not have the specified symmetry. It is allowed, however, to specify a *lower* symmetry than what is actually present in the set of atomic positions. The *specified* symmetry determines how results are analyzed and how irreducible representations and subspecies are labeled. It also determines various algorithmic aspects: the program runs more efficiently with the highest possible symmetry.

The spatial orientation of the molecular coordinate system is not arbitrary. ADF requires for each point group symmetry a specific standard orientation. In axial groups for instance, the main rotation axis must be the z-axis. This implies a restriction on how you can define the atomic coordinates under atoms. The orientation requirements for all point groups are listed in *Symmetry* (page 360). If the specified symmetry equals the true symmetry of the nuclear frame ADF will adjust the input orientation of the molecule to the requirements (if necessary). If you have specified a subgroup of the true nuclear symmetry, no such orientation adjustment is carried out and the user has to make sure that his input data yield the correct orientation, lest an error will occur.

Restrictions apply to the symmetry (as specified) of the molecule, related to the symmetries of the fragments as they were stipulated in the preceding fragment calculations. All symmetry operators of the molecule that internally rotate or reflect a fragment but leave it at the same position in the molecule, must also be operators of the symmetry group in which the fragment has been computed. Furthermore, two fragments must not be symmetry-equivalent in the molecule only by an improper rotation. The implied internal reflection of the fragment must be one of the symmetry operators in the point group symmetry that is used in the fragment calculation *and* the molecular symmetry group must also contain a proper rotation that maps the two fragments onto each other.

The example of the C2H4 molecule implicitly assumes that all fragments are *single atom* fragments. When the fragments are larger the data records in the atoms key have to be extended: you must specify which atoms belong together in one fragment.

```
Atoms
   Ni   0      0      0
   C  -1.06 -1.06   1.06 f=CO/1
   C  -1.06   1.06 -1.06 f=CO/2
   C  -1.06   1.06 -1.06 f=CO/3
   C   1.06 -1.06 -1.06 f=CO/4
   O   1.71   1.71   1.71 f=CO/1
   O  -1.71 -1.71   1.71 f=CO/2
   O  -1.71   1.71 -1.71 f=CO/3
   O   1.71 -1.71 -1.71 f=CO/4
End

Fragments
   CO TAPE21co.yesterday
   Ni t21ni.dzp
End

SYMMETRY T(D)
```

Another sample input file; using a single atom Ni fragment and four molecular CO fragments. The keys symmetry and fragments operate as before. Again we have two types of fragments (here: Ni and CO); for each of them, the fragment file is specified.

Under the key ATOMS the chemical symbols and the nuclear coordinates are listed. Added is the f=...-part; f stands here for fragment and tells the program that the carbon and oxygen atoms belong to CO fragments. The last part /n enumerates the individual CO fragments: here you define which C and O belong together in one CO fragment.

The record for Ni contains no f= part, implying the *default* for this atom: it is a fragment on its own. In the C2H4 example before the default applied to all atoms.

Note that one should use the f= part for symmetry equivalent fragments. In the next example, ADF assumes the fragments CO1, CO2, CO3, and CO4, to be of different fragment types, even though they are coming from the same TAPE21. Therefore ADF will assume symmetry NOSYM in the next calculation, and will not run in T(D) symmetry.

```
Atoms
   Ni  0      0      0
   C  −1.06 −1.06   1.06  f=CO1
   C  −1.06   1.06 −1.06  f=CO2
   C  −1.06   1.06 −1.06  f=CO3
   C   1.06 −1.06 −1.06  f=CO4
   O   1.71   1.71   1.71  f=CO1
   O  −1.71 −1.71   1.71  f=CO2
   O  −1.71   1.71 −1.71  f=CO3
   O   1.71 −1.71 −1.71  f=CO4
End

Fragments
   CO1 TAPE21co.yesterday
   CO2 TAPE21co.yesterday
   CO3 TAPE21co.yesterday
   CO4 TAPE21co.yesterday
   Ni t21ni.dzp
End
```

There are more possibilities with the keys atoms and fragments. This is worked out later. The purpose of this section was to provide a quick and easy start.

3.3.2 Fragment files

The TAPE21 result files from the ADF computations on the fragments that constitute a molecule completely characterize these fragments. The fragment TAPE21 files must be attached as *fragment files*. This is achieved with the key FRAGMENTS. See also the next section for the relation between Atom type, Fragment type and Fragment file names.

```
Fragments {Directory}
   FragType FragFile
   FragType FragFile
end
```

FragType One of the fragment *types* defined under atoms, either explicitly (f=fragtype/n) or implicitly (fragment type=atom type, if the f= option is not used).

FragFile The fragment file: the standard TAPE21 result file from the computation of that fragment. The file name must contain the complete path relative to Directory (the argument of the key). By default, when no Directory is specified, this is the local directory where the job runs. You may therefore omit the directory and give simple (local) file names if all the files are present in the working directory of the job.

Obviously, FragFile is **case sensitive**. However, FragType is also treated as case sensitive; see also the *ATOMS key* (page 25) discussion (f= option). The reason is that there are shortcuts possible to the effect that the FragType name (in the atoms block) is immediately interpreted as the name of the fragment file.

The key FRAGMENTS may be used any number of times in the input file. This is convenient if you employ a sizable number of fragment files, with subsets located in different directories. You can then use the key separately for each directory, to avoid typing long path names for all the files. Fragtypes that occur in the fragments block(s), but that are not referred by atoms are ignored. No fragment files must be specified for dummy atoms (xx).

It is allowed to use one and the same fragment file for different fragment types. Example:

```
ATOMS
   C.1 x1 y1 z1
   C.2 x2 y2 z2
end

Fragments
   C.1 TAPE21.c
   C.2 TAPE21.c
end
```

Two different atom types (and fragment types) C.1 and C.2 are defined. The properties of the two fragment types are now identical since they are characterized by the same fragment file, but from the program's point of view they are different and can therefore not be symmetry equivalent.

The reason you may want to specify different atom types will usually be related to analysis, in particular symmetry aspects. If you know in advance that the two atom types are not symmetry equivalent, or more generally, that they play a rather different role in the molecule, it can enhance clarity of printed output to assign different atom type names to them. However, see the notes below.

A fragment file must not be the result file of a spin-unrestricted calculation. When you try to use such a fragment file, the program will detect it and abort with an error message. If you want to analyze a molecule in terms of unrestricted fragments, you should use restricted fragment files and apply the key FRAGOCCUPATIONS.

Suppose that you have done a calculation on a molecule *mol*, in which you have defined two different atom types for atoms of the same chemical element. Suppose furthermore, that you want to use that molecule now as a fragment in a new calculation.

You list under atoms all atoms of the molecule and you specify which atoms belong to the various fragments, among which the molecular fragment *mol*. The program will then have a problem deciding which atoms in your system are associated with the different atom types in the fragment. Normally, ADF analyzes this by comparing the chemical elements. That is not sufficient here because one chemical element corresponds with more than one type of atom in the *mol fragment* type. In such a case it is imperative to use *the same atom type names* in your new calculation as you used in the generation of the fragment. These names are stored in the fragment file, and they are printed in the output file of the calculation of *mol*.

The names of three items may be related to each other, depending on how you specify input: the *atom type*, the *fragment type*, and the *fragment file*.

The atom type is defined in the data block to atoms.

The fragment type is defined also in the data block to atoms: with the f= option. For records in the data block that don't have the f= option, the fragment type name is by definition identical to the atom type name.

The fragment file is defined in the data block to fragments, each record consisting of a fragment *type* name, followed by the fragment *file*. If a fragment type is not listed in the data block to fragments, so that no fragment file name is specified, the fragment *file* is by definition identical to the fragment *type* name.

MODEL HAMILTONIANS

See also:

ADF-GUI tutorial: multi-level calculations, spin-orbit coupling

QM/MM manual: QM/MM manual

Examples: *special XC functionals* (page 398), *relativistic effects* (page 418), *solvents, other environments* (page 427)

4.1 Electronic Configuration

The next few keys can be used to specify the electronic configuration. If you don't specify any such keys, certain defaults will apply. In principle, the program will (by default) attempt to find the lowest-energy spin-restricted (one-determinant) state. If SCF convergence is problematic the program may wind up at an excited state, by which (in this context) we mean a one-determinant state with a higher energy than some other one-determinant state with the same net spin polarization. In worse cases the program may fail to converge to any state at all. It is good practice to *always* verify which configuration you actually have computed.

When you specify a particular configuration and/or net charge and/or net spin-polarization of the system, the program will try to compute accordingly, even if the data have no physical or chemical meaning. The program has no knowledge about the existence of materials and will simply try to carry out what you tell it to do.

4.1.1 Charge and Spin

Spin: restricted vs. unrestricted

```
UNRESTRICTED
```

Specifies that spin-α and spin-β MOs may be spatially different and may have different occupation numbers. The default (absence of the key) is spin-restricted. The key has no argument. In the case of Spin-Orbit coupling it means that Kramer's symmetry does not have to be satisfied, in which case the key UNRESTRICTED should be used in combination with the key NONCOLLINEAR or COLLINEAR.

The unrestricted mode roughly doubles the computational effort. The actual numbers of spin-α and spin-β electrons respectively are controlled by the keys *CHARGE* (page 45) and *OCCUPATIONS* (page 45). Not e carefully, that using *only* the keyword unrestricted, without either Charge or Occupations (or both) would not result in any spin polarization. This implies that you would effectively perform a spin-restricted calculation, but with increased computational effort. Therefore, the program will check that in an unrestricted calculation at least one of the keys Charge and Occupations is applied.

The unrestricted feature is equivalent with, in *ab-initio* terminology, (Spin-)Unrestricted-Hartree-Fock (UHF); the N-particle wave function is a single determinant and not necessarily an eigenfunction of the spin operator S^2.

A *restricted* calculation implies that the (spatial) orbitals *and* the occupation numbers are identical for spin-α and spin-β.

The Fock operator, both in an unrestricted and in a restricted run, commutes with the spin operator Sz, but not (unless accidentally) with S^2. The obtained one-determinant wave function may for instance be a mixture of a singlet and a triplet state.

In an unrestricted calculation the expectation value of S^2 is now computed in ADF (note 29 in [*98* (page 912)]). The implementation of an evaluation of S^2 is not quite trivial. DFT is essentially a one-particle formalism, so the S-operator for the n-particle system has to be written out in single-particle operators [*99* (page 912)]. The equations used in ADF to calculate the expectation value of S^2 can be found in Szabo and Ostlund [*100* (page 912)]. Note that the so called exact value $(S_{exact})^2$, which is printed in the ADF output, is defined as $(S_{exact})^2 = (|N_a - N_b|/2)(|N_a - N_b|/2+1)$, where N_a and N_b are the number of spin-α and spin-β electrons, respectively. The expectation value of S^2 is not calculated in a Spin-Orbit coupled calculation.

Molecules that have been calculated using the unrestricted formalism cannot be employed as fragments. ADF will abort when you attach the TAPE21 result file from an unrestricted calculation as a fragment file.

A fair approximation to a computation with unrestricted fragments can be achieved with the key *FRAGOCCUPATIONS* (page 51). See also the examples *Example: Spin-unrestricted Fragments: H2* (page 813), *Example: Bond Energy analysis meta-GGA, (meta-)hybrids: Zn2, Cr2, CrH* (page 828) and *Example: Bond Energy analysis open-shell fragments: PCCP* (page 817).

Unrestricted and Spin-Orbit Coupling

In the case of Spin-Orbit coupling there are two ways to do spin-polarized calculations, either using the collinear approximation or the noncollinear approximation [*101* (page 912), *102* (page 912)]. Using the unrestricted feature in order to assign different numbers of electrons to a and b spin, respectively, cannot be applied as such, if one includes Spin-Orbit coupling, since the electrons are not directly associated with spin-α and spin-β. For the collinear and noncollinear approximation one should use `Symmetry NOSYM` (see *Symmetry Key* (page 28)), and each level can allocate 1 electron. Note that with the key `CHARGE` one should only specify one value, namely the total charge. One should not specify the spin-polarization.

```
COLLINEAR
```

This key is only relevant in the case of Spin-Orbit coupling. The key has no argument. See also the key `NONCOLLINEAR`.

In the collinear approximation in each point in space the spin-polarization has the same direction (default is in the direction of the z-axis). Kramer's symmetry does not have to be satisfied. Symmetry used in the calculation should be NOSYM. The default direction of the spin-polarization can be overruled using the key `SOUX` (this key has no argument) for spin-polarization only in the direction of the x-axis, and the key `SOUY` (this key has no argument) for spin-polarization only in the direction of the y-axis. Both keys `SOUX` and `SOUY` are only relevant in the case of Spin-Orbit coupling in combination with the key `COLLINEAR`.

```
NONCOLLINEAR
```

This key is only relevant in the case of Spin-Orbit coupling. The key has no argument. See also the key `COLLINEAR`.

In the noncollinear approximation in each point in space the spin-polarization can have a different direction. Kramer's symmetry does not have to be satisfied. Symmetry used in the calculation should be NOSYM.

Net Charge and Spin polarization

The net charge of the molecule and the net spin polarization can be controlled with the key CHARGE:

```
CHARGE {NetQ {ab}}
```

NetQ The net total charge of the molecule

ab The net total spin polarization: the number of spin-α electrons in excess of spin-β electrons. Specification is only meaningful in a spin-unrestricted calculation. However, specification is not meaningful in an unrestricted Spin-Orbit coupled calculation using the (non-)collinear approximation.

If the key is used, the first value in the argument is assigned to netQ, the net total charge, and the second to ab. If the key is not used at all, default values apply. The default for the net total charge is the *sum of fragment charges: not necessarily neutral!!* The fragment charges are the net total charges that were used in the fragment runs; this information is stored in the fragment files.

The default spin polarization is zero.

An unrestricted calculation with ab=0 (for instance by not specifying the spin polarization at all) is, in the case one does spin break the spin symmetry, in fact a restricted run: it should give exactly the same as the restricted calculation, but it will use more CPU time. If one does break the spin symmetry, for example with the key MODIFYSTARTPOTENTIAL or the SPINFLIP option in the key *RESTART* (page 320), the solution may also be a broken spin symmetry solution. For example one may want to start a calculation in broken symmetry with spin-α density on one fragment and spin-β density on another, e.g. in a spin-unrestricted calculation of H_2 at large separation.

4.1.2 Orbital occupations: electronic configuration, excited states

With the key OCCUPATIONS and IRREPOCCUPATIONS you can specify in detail the assignment of electrons to MOs

```
OCCUPATIONS Options
```

Aufbau, smearing, freezing

```
OCCUPATIONS Options
```

Options May contain IntegerAufbau, Keeporbitals, ElectronicTemperature, Smear, Freeze, or Steep:

IntegerAufbau Electrons are assigned to MOs according to the Aufbau principle, trying to use integer occupations, also in case of degeneracy at the Fermi level. ADF normally may use fractional occupation numbers in that case. Note that for multi-dimensional irreps, using the subkey IntegerAufbau may stil not prevent fractional occupation numers to be used.

Keeporbitals=NKeep Until SCF cycle Nkeep electrons are assigned to MOs according to the Aufbau principle, using at each cycle the then current orbital energies of the MOs. Thereafter the KeepOrbitals feature is applied. As soon as this is activated the program will on successive SCF cycles assign electrons to the MOs that maximally resemble - in spatial form - those that were occupied in a 'reference cycle number'. The default for Nkeep is 20, except:

1. When orbital occupations for MOs are specified explicitly in the data block of the occupations key, these apply throughout.

2. In a Create run fixed occupations are derived from a database in the program.

3. When electron smearing is explicitly turned on by the user (see the Smear option below) Nkeep is by default 1,000,000 so the program will 'never' compare the spatial forms of MOs to determine the occupation numbers. The 'reference cycle number' is by default the previous cycle, which will suppress jumps in the spatial occupations during the SCF development while at the other hand allowing the system to let the more-or-less-frozen configuration relax to self-consistency.

Freeze Occurrence of this word in the option list specifies that the 'reference cycle number' will be the cycle number on which the KeepOrbitals feature is activated: during all subsequent SCF cycles the program will assign electrons to MOs that resemble the MOs of that specific SCF cycle. This may be used when the MOs of that cycle are already reasonably close to the final ones, and it will suppress unwanted step-by-step charge-transfers from occupied to empty orbitals that are very close in energy. By default this option is not active.

ElectronicTemperature=T Invokes orbital population using the Fermi-Dirac distribution with temperature `T`. This may be used to achieve SCF convergence in an otherwise problematically converging system. Note that the default SCF acceleration method A-DIIS is not compatible with non-aufbau orbital occupations so it is disabled automatically and the classical Pulay DIIS scheme is used. For this reason it is strongly recommended to use either MESA or one of the methods from the LIST family in combination with `ElectronicTemperature` or `Smear`.

Smear=Smear1[,Smear2,Smear3,...,Smear10] Another form of specifying the electronic temperature. The smearing parameter Smear$N$ is related to the `ElectronicTemperature` in such a way that the occupation number for any orbial outside the smearing interval will deviate from an integer value (0 and 1 for spin-unrestricted, 0 and 2 for spin-restricted) by no more than 1% (0.01 electron for spin-unrestricted, 0.02 for spin-restricted). More precisely, a smearing value of 1 Hartree is equivalent to an electronic temperature of 34285 Kelvin. When a comma-delimited list of values is specified, the next value from the list is picked after SCF has converged, and the SCF is continued. This way one can specify a list of gradually decreasing values to get sort of annealing effect. NOTE: No spaces are allowed when specifying the list of values for Smear.

Steep=Lambda[,Nmax] The occupation number for each orbitals are updated according to steepest-descent method (Ref: F. W. Averill and G. S. Painter, Phys. Rev. B **46**, 2498 (1992)). During an SCF cycle, the occupation number for each new orbital is initially determined by decomposing the old charge density with new orbitals. Then, the occupation numbers are modified so that the total energy of the system will decrease. The Lambda parameter gives the coefficient for the charge transfer in 1/au unit. The second parameter, Nmax, is an additional limit for the amount of the charge transfer. Nmax would be useful for early steps of cycle when the Lambda parameter gives too large charge transfer. Too small Nmax results in irregular behavior in SCF convergence. In the case of difficult SCF convergence, you should make mixing and Lambda smaller. From our experience, Nmax=0.1 or 0.2 is usually OK. This method should be used with turning off DIIS method (DIIS N=0), and the choice of the mixing parameter in SCF cycle is also important. This option is especially useful for systems with many quasi-degenerate orbitals around Fermi level. For instance, cluster models of surface systems usually suffer from dangling bonds and should be converged with this method. Note though that slow convergence is an intrinsic feature of this method so one should specify a large limit for the number of SCF cycles, say 500 or even 1000, depending on the cluster size.

OptimizeSpin=Delta, OptimizeSpinRound=Delta The spin polarization can be optimized by minimizing the difference between Fermi level values for spin-alpha and spin-beta MOs. This method works reliably only when a reasonably large finite electron temperature is used (300K is found to be sufficiently large for small transition metal clusters). The polarization value is optimized up to a specified tolerance Delta (fractional values are allowed) by step-wise reduction of the root-braketing interval for the objective function f=FermiLevel(alpha)-FermiLevel(beta). The SCF procedure is converged for each spin polarization. When the interval becomes smaller than Delta the optimization is considered converged. After that the SCF may optionally be repeated with the spin polarization set to the nearest even or odd number (depending on the total number of electrons in the system) or, in other words, with the total number of electrons per spin rounded to the nearest integer. The method is invoked using either the OptimizeSpin or the

OptimizeSpinRound Occupation option. As mentioned above, it must be used together with the Electronic-Temperature option. The OptimizeSpin and OptimizeSpinRound argument specifies the tolerance value. The OptimizeSpinRound option specifies that after polarization has converged the SCF must be completed with the total number of electrons per spin rounded to the nearest integer. It should be noted that, as usual, the Unrestricted and Charge keys must be specified as for any other spin-unrestricted calculation. The second argument of the Charge key sets the initial spin polarization value from which the optimization is started. The result of the optimization may depend on the initial value because the objective function may have more than one root.

Notes about the occupations options:

- When occupation numbers are explicitly defined via the block `IRREPOCCUPATIONS` (see next section), the Smear option cannot be used.

- The aufbau principle does not determine or adjust the distribution of electrons over spin-α versus spin-β in an unrestricted calculation. This aspect is controlled by the key `CHARGE` and by any explicit occupations in the data block of occupations.

- Smearing cannot be used in combination with the keeporbitals option.

Explicit occupation numbers

```
IRREPOCCUPATIONS
  irrep orbitalnumbers
  irrep orbitalnumbers
  ...
End
```

irrep The name of one of the irreducible representations (not a subspecies) of the point group of the system. See the *Symmetry* (page 360) for the irrep names as they are used in ADF.

orbitalnumbers A series of one or more numbers: the occupation numbers for one-electron *valence* orbitals in that irrep. The orbitals are ordered according to their energy eigenvalue; higher states than those listed get an occupation number zero.

For degenerate representations such as the 2-dimensional E-representations or the 3-dimensional T-representations, you must give the *total* occupation, i.e. the sum over the partner representations; ADF assigns each partner an occupation equal to the appropriate fraction of what appears here.

In an unrestricted calculation, two sequences of numbers must be specified for each irrep; the sequences are separated by a double slash (//). The first set of numbers is assigned to the spin-α orbitals, the second set to the spin-β orbitals. Example unrestricted calculation in symmetry NOSYM with two unpaired electrons:

```
IRREPOCCUPATIONS
   A 28 // 26
End
CHARGE 0 2
SYMMETRY NOSYM
```

Note that this is not meaningful in an unrestricted Spin-Orbit coupled calculation using the (non-)collinear approximation, where one should use one sequence of occupation numbers for each irrep.

Notes about the occupations data block:

- When specifying electron configurations, all valence electrons in the calculation must be explicitly assigned to MOs and the IRREPOCCUPATIONS keyword must be used. In this context the concept *valence electrons* and hence *valence orbitals* is not necessarily identical to what you may normally assume to be the valence space of an atom or molecule. The meaning of *valence* is here strictly defined as whatever electrons are outside the

frozen core. It depends therefore on the level of frozen core approximation applied in the calculation. This traces back to the Create runs in which the basic atoms were generated that are now used to build the molecule.

- When for some irrep there is a rather long list of occupation numbers, corresponding to * consecutive fully occupied* states, you can combine these numbers and enter their sum instead: ADF knows the maximum occupation for an irrep, and when you put a larger number the program will split it up. For instance, if you give for the *p*-representation (in a single atom calculation):

```
P 17 3
```

ADF will interpret this as:

```
P 6 6 5 3
```

i.e. the occupation number 17 is interpreted as denoting two fully occupied p-shells and the remaining five electrons in the next higher shell. This example also illustrates how to specify an excited state: here we have defined a hole in the third p-shell.

- Fractional occupation numbers in input are allowed. For a discussion of the interpretation of fractional occupation numbers see [*103* (page 913)]. The program even allows you (technically) to use a non-integer total number of electrons, whatever the physical meaning of such a calculation is.

- The data block of occupations is not parsed (see the section *Structure of the Input* (page 18)). The program does not replace expressions by their value and it does not recognize constants or functions defined with the define key.

- In a numerical frequencies run (without the Symm argument) the symmetry used internally in the program is NOSYM, irrespective of any Schönfliess symbol in the input file. As a consequence the program will recognize only the A representation (the only irrep in nosym), but not the representations belonging to the input point group symmetry. (The symmetry in the equilibrium geometry, defined by the input Schönfliess symbol, is used to enhance efficiency and stability in the construction of the matrix of Force constants).

CHARGE vs. IRREPOCCUPATIONS

The contents of the data block of IRREPOCCUPATIONS, if used, defines the total number of valence electrons and hence the net total charge. In an unrestricted run it also defines the net spin polarization. If the key `CHARGE` is also used, the program will check that both specifications are consistent.

We strongly recommend to employ this and always specify the net total charge and spin polarization with charge whenever explicit occupation numbers are supplied with IRREPOCCUPATIONS, to that the program will check that your occupation numbers result in the total charge and spin polarization that you have in mind.

Create mode

In Create mode occupation numbers are predefined (see Appendix *Elements of the Periodic Table* (page 347)), and these are applied unless you specify occupations in input yourself. Conceivably this may result in a non-aufbau configuration. In Create mode the program always operates as if occupations were set in input.

4.1.3 Frozen core approximation

Frozen core vs. pseudopotentials

Pseudopotentials are not supported. The frozen core approximation is automatic in a normal (Fragment mode) calculation and is defined by the basic atomic fragments. The data file used in the Create run specifies the frozen core for the atom, which is then used in all molecules that incorporate that atomic fragment.

Core Potentials

In the standard approach the Coulomb potential and the charge density due to the atomic frozen core are computed from the frozen one-electron orbitals. ADF stores the computed core density and core potential for each atom type in the molecule on a file TAPE12. Alternatively, you may attach a file with (core) potentials and densities. The file must have the same structure as the standard TAPE12. It should contain one or more sections, each with the core information for one type of atom. With the block COREPOTENTIALS you specify the core file and (optionally) which sections pertain to the distinct atom types in the molecule.

```
COREPOTENTIALS corefile
  {atomtype index}
  {atomtype index}
  ...
end
```

corefile The file with core potentials and charge densities. The name may contain a path.

atomtype One of the atom type names as defined by atoms.

index Points to the core section on the attached file that applies to the atom type. Different atom types may use the same section. A non-positive index tells the program that the atoms of that type don't have a frozen core. If the information on the corresponding fragment file (or data file in Create mode) indicates the contrary the program will abort with an error message.

If the key is used as a simple key (specifying only the core file) the sections on the file are associated with the atom types in order: the first section is used for the first atom type, et cetera. This is overruled by applying the block form. However, since the key *must* have the core file as argument, the block form requires that you apply the continuation symbol: an ampersand (), separated from the core file name by at least one blank.

If you omit an atom type from the data block it gets a zero index (no core).

The attached file may contain more sections than used in the calculation, and the indices specified in the data block don't have to be in ascending order, consecutive, or cover a specific interval.

When a file with non-standard (e.g. relativistic) cores is attached and used in the calculation of an atom or molecule, and the result is used as fragment in a subsequent calculation, you should attach and use the same core potentials again. Otherwise, the program will internally compute the standard core potentials and hence implicitly employ another fragment than you may think, i.e. a fragment with other properties. ADF will not check anything in this respect and corepotentials should therefore be handled with great care.

The primary application of the corepotentials option is to include (scalar) relativistic corrections in the (frozen core part of the) Fock operator. The relativistic core potentials can be computed with the auxiliary program dirac (see the RELATIVISTIC keyword).

4.1.4 Spin-polarized start-up potential

The Coulomb and XC (exchange + correlation) potentials are computed from the fit approximation of the charge density (see *Density fitting* (page 304)).

The fit coefficients of this approximation for the first SCF cycle, needed to compute the first Fock matrix, are read from the fragment files: the start-up density is chosen as a sum-of-fragment-densities (fit approximations) and this combined density defines the initial potential.

In the SCF restart run the fit coefficients may be read from the attached TAPE21 file, see the key *RESTART* (page 320).

Spin-flip method for broken symmetries

Starting from ADF2009.01 it is possible to exchange alpha and beta electrons for selected atoms when performing a restart from a previous spin-unrestricted calculation.

In many cases, one wishes to perform a calculation of a low-spin complex where spin-density is positive on some atoms and negative on the others. It is usually very difficult to achieve SCF convergence if one starts from scratch. Sometimes, the MODIFYSTARTPOTENTIAL feature (see next section) helps with this problem but sometimes it does not. A more robust way is to first perform a high-spin calculation and then modify the resulting t21 file by "flipping" the spin on some atoms. This file then can be used to restart a subsequent low-spin calculation.

Such a "flipping" can now be performed during restart by specifying a SPINFLIP keyword in the RESTART input block as shown below:

```
RESTART high-spin.t21
   ! SpinFlip keyword is followed by atom numbers for
   ! which the flipping will be performed
   SPINFLIP 1
END
```

An example demonstrating the feature may be found in the examples *Broken spin-symmetry: Fe4S4* (page 600).

Modify the starting potential

In some applications you may want to modify the initial fit coefficients (from the restart file or the fragment files), see also the previous section. This is achieved with the block MODIFYSTARTPOTENTIAL. It allows you to scale them in some way so as to represent user-chosen amounts of spin-α and spin-β fit density on some or all of the fragments. This will adjust the spin-α and spin-β initial potentials.

This option applies only to *unrestricted* calculations of course. It may be used to help the program find a particular state. This might, for instance, be hard to find otherwise due to the a-b symmetry in the start-up situation. It may also be useful to speed up the SCF convergence in case you know what the final distribution of spin-α and spin-β density over the molecule will approximately be.

```
MODIFYSTARTPOTENTIAL {specification}
{ frag alfa // beta
 frag alfa // beta
 ...}
end
```

specification Must be *two numbers*, ASPIN and BSPIN, if provided at all. They specify the (relative) amounts of spin-α and spin-β fit density to define the spin-dependent potential at the first SCF cycle. The coefficients retrieved from the fragment files (or from the restart file in case of a SCF restart) are scaled accordingly. This will not affect the *total* amount of fit density: the absolute values of ASPIN and BSPIN play no role, only their ratio. In case of a restart run the restart file must have been generated in a *restricted* calculation, while the continuation run must be an *unrestricted* one.

If no argument is given a data block must be supplied with records frag alfa // beta. This is very much similar to the main option with ASPIN and BSPIN: you specify ASPIN and BSPIN now separately for each fragment. This involves somewhat more input but increases the possibilities to tune the initial potential. Again this can be applied only in an unrestricted calculation. It cannot be used in a restart: the affected fit coefficients are those from the fragment files, while in an SCF restart run these are ignored and replaced by the coefficients on the TAPE21 restart file.

Each line specifies a frag with its corresponding ASPIN and BSPIN fit partitioning. If frag is the name of a fragment *type*, the specified ASPIN-BSPIN is applied to all individual fragments of that type. Alternatively an *individual* fragment can be specified, using the format fragtype/n, where *n* is an index between one and the total number of

fragments of that type. In such a case the ASPIN-BSPIN data applies only to that particular fragment while different values may be supplied for the other fragments of the same type.

It is allowed to specify for certain fragment types individual fragments and for other fragment types only the type. Duplicate specifications are not allowed; an individual fragment must not be specified if its fragment type is also specified as a whole. If the data block form is used, only the fit coefficients of the referenced fragments are affected. For the not-referenced fragments the fit densities are used as they are defined on the corresponding fragment files.

The SCF convergence of a spin-unrestricted calculation usually improves when you start with potentials that correspond to the correct ratio of spin-α and spin-β electrons. By default ASPIN=BSPIN=0.5, as implied by the spin-restricted start density of the fragments or restricted molecule.

The total amount of fit density used on the first iteration is defined by the sum-of-fragment densities (or the density on the restart file). This may be different from the total nr. of electrons in the actual calculation. On the second SCF cycle the fit density will internally be normalized so as to represent the correct number of electrons.

The block-form of the key makes the start up of broken symmetry calculations easy. For example one may want to start a calculation in broken symmetry with spin-α density on one fragment and spin-β density on another, e.g. in a spin-unrestricted calculation of H$_2$ at large separation. It is particularly useful for larger systems, e.g. for magnetic coupling between spin-polarized magnetic centers, as in Fe-S complexes [*111* (page 913)]: start with oppositely polarized Fe centers, but with, for instance, the remaining bridge and terminal ligands unpolarized. See also the N2+ sample run in the examples.

4.1.5 Unrestricted fragments

The fragments from which the molecule is built must be spin-restricted, that is: the fragment files must be result files of spin-restricted calculations. For purposes of analysis, however, it may be desirable in some applications to build your molecule from *un*restricted fragments. This can be simulated as follows.

You tell ADF that you want to *treat* the fragments as if they were unrestricted; this causes the program to duplicate the one-electron orbitals of the fragment: one set for spin-α and one set for spin-β. You can then specify occupation numbers for these spin-unrestricted fragments, and occupy spin-α orbitals differently from spin-β orbitals.

Of course, the unrestricted fragments that you use in this way, are not self-consistent: different numbers of spin-α and spin-β electrons usually result in different spatial orbitals and different energy eigenvalues for spin-α and spin-β when you go to self-consistency, while here you have spatially identical fragment orbitals. Nevertheless it is often a fair approximation which gives you a considerable extension of analysis possibilities.

Typically an unrestricted electron configuration for the fragments is used, such that the Pauli repulsion between the fragments is minimal. For example if one has two fragments which both have one unpaired electron, one would put the unpaired electron of the first fragment in the spin-α orbital and the unpaired electron of the second fragment in the spin-β orbital. If one wants to calculate separately the electron pair bonding see key *REMOVEALLFRAGVIRTUALS* (page 52).

In ADF2012 for hybrids, metaGGA's, and metahybrids the calculation of the Pauli repulsion term is implemented if one is simulating an unrestricted fragment with the key FRAGOCCUPATIONS.

```
FRAGOCCUPATIONS
 fragtype
   irrep spin-a // spin-b
   irrep spin-a // spin-b
   ...
 subend
 fragtype
   irrep spin-a // spin-b
   ...
 subend
end
```

fragtype One of the fragment types and functions as a (block type) subkey. The data block for the subkey ends with the standard end code for block type subkeys (subend).

irrep One of the irreducible representations (irreps) for the point group symmetry that was used in the computation of that fragment.

spin-a // spin-b Two sequences of occupation numbers, which will be applied to the spin-α and spin-β versions of the Fragment Orbitals. The sequences must be separated by a double slash (//). See for comparison the specification of occupation numbers for the overall system (key CHARGE).

The sum of spin-α and spin-β occupations must, for each fragment orbital in each irrep separately, be equal to the total (restricted) occupation of that orbital as it is stored on the fragment file. In other words: you can only change the distribution over spin-α and spin-β electrons within one orbital.

(Without this restriction the spatial distribution of the total (sum over spins) fragment charge density would be changed, leading to an incorrect bonding energy analysis after the calculation).

The data block of FRAGOCCUPATIONS is not parsed for expressions and constants or functions defined under define. Any such items will not be recognized and not be replaced by their values.

Be aware that in more-dimensional irreps (e, t) the number of electrons in a fully occupied orbital is input as the dimension of the irrep times the one-electron orbital occupation. Compare the key IRREPOCCUPATIONS.

For irreps that are not mentioned in this input block, and hence for all irreps of fragment(type)s that are not mentioned at all, the spin-α and spin-β occupations will be set equal, which is of course what they in fact are on the (restricted) fragment file.

For an example of applying this option see [*112* (page 913)].

4.1.6 Remove Fragment Orbitals

By default all fragment orbitals (the MOs of the fragment computation), which are stored on the fragment file, are used as basis functions for the overall molecule. You can remove one or more of these fragment orbitals from the basis set of the molecule. This may be useful for special analyzes, like calculating the electron pair bonding in case one has open shell fragments, or for instance to study the effect of deleting all virtual MOs of a particular fragment (CSOV analysis, Constrained Space Orbital Variation). It may also enhance the efficiency since you effectively reduce the size of the basis set, but you should be aware of the potential effects on the results.

The pure orbital interaction effect of forming electron pair bonding between open shell molecules can approximately be calculated with a bond energy analysis in which all virtual orbitals are removed from the fragments, see Ref. [*489* (page 935)]. For calculating the effect of electron pair bonding best is to specify an unrestricted electron configuration for the fragments using the key FRAGOCCUPATIONS, such that the Pauli repulsion is minimal. Note that the fragments in ADF need to be calculated spin-restricted. Removing of all virtuals from an open shell fragment means that all fragment orbitals with zero occupation are removed. Thus, for example, a singly occupied fragment orbital will not be removed. This singly occupied fragment orbital will result in a spin-α and a spin-β fragment orbital. In combination with other singly occupied fragment orbitals they may form an electron pair bonding combination, but also an anti-bonding combination. In practice this means that the orbital interaction calculated with a bond energy analysis in which all virtual orbitals are removed from open shell fragments might be due to more than electron pair bonding.

If one wants to remove all virtual fragment orbitals use the key REMOVEALLFRAGVIRTUALS.

```
REMOVEALLFRAGVIRTUALS
```

If one does not want to remove all virtual fragment orbitals then one should use the block key REMOVEFRAGOR-BITALS.

```
REMOVEFRAGORBITALS
 fragtype
    subspecies nremove
    subspecies nremove
    ...
 subend
 fragtype
    subspecies nremove
    ...
 subend
 ...
 (etc.)
 ...
end
```

fragtype One of the fragment types in the system. Any subset of the available fragment types can be used here as subkey. The subkeys are block type keys; their data blocks end subend.

subspecies One of the subspecies of the irreducible representations of the point group symmetry that was used in the calculation of the fragment itself. This requires of course that one knows the symmetry that has been used for the fragment calculation.

nremove The number of fragment orbitals of the pertaining representation that will not be used as basis functions for the overall system. The *highest* (in energy eigenvalue) nremove orbitals are discarded. You must not remove *occupied* fragment orbitals.

By default (omission of the key) all fragment orbitals are used in the basis set for the system.

Important Note

It is imperative that any removal of fragment orbitals will not break the symmetry of the molecule. This consideration is relevant when for instance two different subspecies of a fragment irrep contribute to different partner subspecies in one of the irreps of the molecule. In such a case, when one removes an orbital in such a fragment subspecies, its partner orbital should also be removed. If this is violated an error may occur or the results will simply be wrong. Quite likely, the program will detect the error, but this may occur only in the final (analysis) stage of the calculation so that a lot of CPU time may have been wasted.

Example: consider a single-atom fragment, computed in atom symmetry, used as fragment in a c(lin) molecule and assume that the p:x and p:y fragment orbitals contribute to respectively the pi:x and pi:y subspecies of the molecule. Then, when you remove one or more p:x fragment orbitals, you should also remove the same number of p:y fragment orbitals. Practical cases may be more complicated and whenever you use this key, make sure that you've fully analyzed and understood how the fragment irreps combine into the molecular symmetry representations. Hint: run the molecule, without removing any fragment orbitals, and stop at an early stage after the program has computed and printed the build-up of the molecular SFOs from the fragment orbitals. To control early aborts via input, use the key STOPAFTER.

4.1.7 CDFT: Constrained Density Functional Theory

CDFT is a tool for carrying out DFT calculations in the presence of a constraint. The nature of the constraint is general in theory, however, in the current implementation the user can constrain the CHARGEs or the SPINs of a set of moieties (as identified by sets of atoms) to be a specific real number given in input. Note that the use of CDFT as implemented in ADF is an **expert option**, and it is a **work in progress**. Implemented in ADF by M. Pavanello and P. Ramos [*410* (page 931)], based on the method described in Ref. [*409* (page 931)]. At the moment SYMMETRY NOSYM and an all electron basis set are required.

The simplest way to run CDFT is using the following keyword combination

```
CDFT
  NCONSTRAINTS 1
  NATOMSPERSET N
  THEATOMS atom1 atom2 ... atomN
  CONSTRAINTS charge
END
```

All the CDFT block key options are

```
CDFT
  NCONSTRAINTS Nc
  NATOMSPERSET n1 n2 n3 ... n_Nc
  CONSTRAINTS charge_1 charge_2 charge_Nc
  {ALLATOMS | THEATOMS atom1_1 atom2_1 ... atomn_1 atom1_2 ...  atomn_Nc}
  {INITIALMULTIPLIERS lambda_1  lambda_2 ... lambda_Nc}
  {POPTYPE population_analysis_type}
  {ONLYCHARGE, CHARGEANDSPIN, EXCITEDCDFT}
  {METRIC}
  {DONOTOPTIMIZE}
  {ANALYTICALHESSIAN N_scf}
  {THRESHOLD threshold}
  {STEPSIZE stepsize}
  {MAXITER maxiter}
  {PRINT print_level}
END
```

NCONSTRAINTS Nc This specifies the number of sets of atoms to be considered. For example, if the user wishes to constrain a positive charge on one part of the system, and a negative charge on another part, NCONSTRAINTS should be set to two. There is no limit on the number of constraints. However, SCF convergence becomes an inssue with more than 2 constraints. Default: 1. Note: NCONSTRAINTS>1 is untested.

NATOMSPERSET n1 n2 n3 ... n_Nc The number of atoms in each moiety (set). No default value.

CONSTRAINTS charge_1 charge_2 charge_Nc The values of the constraints. If CHARGEANDSPIN, constraints to the alpha and beta electrons need to be specified sequentially. One more electron => CONSTRAINTS -1.0. One less electron => CONSTRAINTS 1.0. If the CDFT type is EXCITEDCDFT, CONSTRAINTS=1.0 is recommended. Other values are technically possible but have not been tested yet. No default value.

ALLATOMS | THEATOMS atom1_1 atom2_1 ... atomn_1 atom1_2 ... atomn_Nc The atom numbers of the moieties in the input geometry order. If NCONSTRAINTS is larger than 1, the sets of atoms are entered as a single list. If ALLATOMS is specified, then THEATOMS is overridden and all the atoms in the active fragment are included in the set. If ALLATOMS is specified then NCONSTRAINTS=1. No default value.

INITIALMULTIPLIERS lambda_1 lambda_2 ... lambda_Nc If available, a guess for the Lagrange multipliers can be entered. Default: 0.0.

POPTYPE population_analysis_type The population analysis chosen for determining the constraint. Default: YUKAWALIKE, as is the only one compatible with an FDE calculation (at the moment). Available options are:

- YUKAWALIKE (atom-centered Yukawa potentials),
- FUZZYVORONOIBECKE (Becke population analysis), and
- FUZZYVORONOIFERMI (atom-centered polynomial decay)

ONLYCHARGE, CHARGEANDSPIN, EXCITEDCDFT The type of CDFT calculation. Default: ONLYCHARGE

- ONLYCHARGE will constrain only the charge, letting spin relax (and potentially delocalize)

- CHARGEANDSPIN will constrain both the charge and the spin

- EXCITEDCDFT will generate an excited state with CONSTRAINTS number of ALPHA electrons constrained to occupy the virtual space of a ground state reference calculation. This is the essence of the eXcited Constrained DFT (XCDFT) method [*492* (page 935)] for the calculation of low-lying single excitations. XCDFT is found to correctly reproduce the energy surface topology at conical intersections between the ground state and the first singly excited state and can also accounts for the condensed phase effects in solvated chromophores where typical Delta SCF methods variationally collapse.

METRIC Relevant for XCDFT. In the XCDFT method orthogonality is not imposed between the KS-orbitals of ground and excited states. If METRIC is specified, the degree of mixing of the single excited state with the ground state or high-order excitations is calculated. Three parameters are calculated: p, m and d. The parameters p and m will give information about the amount of mixing with the ground state, while parameter d will determine the mixing with high order excitations. Additional information about the origin of these parameters can be found in the literature [*492* (page 935)].

DONOTOPTIMIZE If this is specified, the multipliers chosen in INITIALMULTIPLIERS will not be optimized and will be constant throughout the entire SCF procedure. Default: not invoked.

ANALYTICALHESSIAN N_scf This will calculate the analytical derivative of the energy w.r.t. the Lagrange multiplier up to the specified SCF iteration. This key is not recommended due to the high computational cost that comes with it. The calculation is equivalent to a ground state Hessian, and it is carried out with the full sum-over-states formula. Default: not invoked.

THRESHOLD threshold The threshold for convergence of the CDFT constraints. The tighter the SCF convergence criteria, the tighter the THRESHOLD should be. Default: 1.0e-10.

STEPSIZE stepsize The amount of the Lagrange multiplier's step taken in each CDFT iteration. Default: 0.2.

MAXITER maxiter Maximum number of CDFT iterations. CDFT carries out a loop nested inside the SCF cycle. Default: 200.

PRINT print_level Print level and debugging. Default: LOW Available options:

- LOW, minimal printing and no debug checks,

- MEDIUM, intermediate print level, no debug checks,

- HIGH, massive printing and debug checks enabled (slow)

4.2 Density Functionals (XC)

The Density Functional, also called the exchange-and-correlation (XC) functional, consists of an LDA, a GGA part, a Hartree-Fock exchange part (hybrids), and a meta-GGA part (meta-GGA or meta-hybrid). LDA stands for the Local Density Approximation, which implies that the XC functional in each point in space depends only on the (spin) density in that same point. GGA stands for Generalized Gradient Approximation and is an addition to the LDA part, by including terms that depend on derivatives of the density. A hybrid GGA (for example B3LYP) stands for some combination of a standard GGA with a part of Hartree Fock exchange. A meta-GGA (for example TPSS) has a GGA part, but also depends on the kinetic energy density. A meta-hybrid (for example TPSSh) has GGA part, a part of Hartree-Fock exchange and a part that depends on the kinetic energy density. For these terms ADF supports a large number of the formulas advocated in the literature. For post-SCF energies only, ADF supports also various other meta-GGA functionals and more hybrid functionals.

The key that controls the Density Functional is XC. All subkeys are optional.

```
XC
  {LDA LDA {Stoll}}
  {GGA GGA}
  {MetaGGA metagga}
  {Model MODELPOT [IP]}
  {HartreeFock}
  {OEP fitmethod {approximation}}
  {HYBRID hybrid {HF=HFpart}}
  {MetaHYBRID metahybrid}
  {XCFUN}
  {RANGESEP {GAMMA=X} {ALPHA=a} {BETA=b}}
  (LibXC functional}
  {DISPERSION [s6scaling] [RSCALE=r0scaling] [Grimme3] [BJDAMP] [PAR1=par1] ␣
→[PAR2=par2]  [PAR3=par3]  [PAR4=par4] }
  {DISPERSION dDsC}
  {DISPERSION UFF}
end
```

If the `XC` key is omitted, the program will apply only the Local Density Approximation (no GGA terms). The chosen LDA form is then VWN.

4.2.1 LDA

```
XC
  LDA {functional} {Stoll}
End
```

LDA Defines the LDA part of the XC functional. If `functional` is omitted, VWN will be used (also if LYP is specified in the GGA part).

Available LDA functionals:

- **Xonly**: The pure-exchange electron gas formula. Technically this is identical to the Xalpha form (see next) with a value 2/3 for the X-alpha parameter.

- **Xalpha**: The scaled (parametrized) exchange-only formula. When this option is used you may (optionally) specify the X-alpha *parameter* by typing a numerical value after the string Xalpha (separated by a blank). If omitted this parameter takes the default value 0.7

- **VWN**: The parametrization of electron gas data given by Vosko, Wilk and Nusair (ref [*20* (page 908)], formula version V). Among the available LDA options this is the more advanced one, including correlation effects to a fair extent.

  - `Stoll`: For the VWN variety of the LDA form you may include Stoll's correction [*21* (page 908)] by typing Stoll on the same line, after the main LDA specification. You must not use Stoll's correction in combination with the Xonly or the Xalpha form for the Local Density functional. The Stoll formula is considered to be a *correlation* correction to the *Local* Density Approximation. It is conceptually not correct to use the Stoll correction *and* apply gradient (GGA) corrections to the correlation. It is the user's responsibility, in general and also here, to avoid using options that are not solidly justified theoretically.

- **PW92**: the parametrization of electron gas data given by Perdew and Wang (ref [*288* (page 923)]).

4.2.2 GGA

```
XC
   GGA functional
End
```

GGA Specifies the GGA part of the XC Functional (in earlier times often called the 'non-local' correction to the LDA part of the density functional). It uses derivatives (gradients) of the charge density.

Available GGA functionals:

- **BP86**: Exchange: Becke, Correlation: Perdew

- **PW91**: Exchange: pw91x, Correlation: pw91c

- **mPW**: Exchange: mPWx, Correlation: pw91c

- **PBE**: Exchange: PBEx, Correlation: PBEc

- **RPBE**: Exchange: RPBEx, Correlation: PBEc

- **revPBE**: Exchange: revPBEx, Correlation: PBEc

- **mPBE**: Exchange: mPBEx, Correlation: PBEc

- **PBEsol**: Exchange: PBEsolx, Correlation: PBEsolc

- **HTBS**: Exchange: HTBSx, Correlation: PBEc

- **BLYP**: Exchange: Becke, Correlation: LYP

- **OLYP**: Exchange: OPTX, Correlation: LYP

- **OPBE**: Exchange: OPTX, Correlation: PBEc [*175* (page 917)]

- **BEE**: Exchange: BEEx, Correlation: PBEc

- **XLYP**: Exchange: XLYPx [*172* (page 917)] (exchange, not available separately from LYP) + LYP

- **SSB-D**: Dispersion corrected functional by Swart-Solà-Bickelhaupt [*286,287* (page 923)]. The SSB-D functional by definition already includes a dispersion correction by Grimme (factor 0.847455). There are some numerical issues with the GGA implementation in ADF of SSB-D (Ref. [*286,287* (page 923)]) for some systems. Because of this, the GGA SSB-D option is only available for single-points (and NMR). Geometry optimizations (etc.) are still possible by using instead:

```
XC
   METAGGA SSB-D
END
```

  This METAGGA implementation is only possible with all-electron basis sets. Use GGA SSB-D for NMR calculations.

- **S12g**: Dispersion corrected (Grimme-D3) functional by Swart, successor of SSB-D [*367* (page 928)].

- **LB94**: By Van Leeuwen and Baerends [*15* (page 907)].

- **KT1**: By Keal and Tozer [*171* (page 916)].

- **KT2**: By Keal and Tozer [*171* (page 916)].

If only a GGA part is specified (omitting the `LDA` sub key) the LDA part defaults to VWN, except when the LYP correlation correction is used: in that case the LDA default is Xonly: pure exchange. The reason for this is that the LYP formulas assume the pure-exchange LDA form, while for instance the Perdew-86 correlation correction is a correction to a *correlated* LDA form. The precise form of this correlated LDA form assumed in

the Perdew-86 correlation correction is not available as an option in ADF but the VWN formulas are fairly close to it.

Separate choices can be made for the GGA exchange correction and the GGA correlation correction respectively. Both specifications must be typed (if at all) on the same line, after the GGA subkey.

For the **exchange** part the options are:

- **Becke**: Becke (1988) [*22* (page 908)].

- **PW86x**: Perdew-Wang (1986) [*23* (page 908)].

- **PW91x**: Perdew-Wang (1991) [*24* (page 908)]

- **mPWx**: Modified PW91 by Adamo-Barone (1998) [*25* (page 908)]

- **PBEx**: Perdew-Burke-Ernzerhof (1996) [*26* (page 908)]

- **RPBEx**: revised PBE by Hammer-Hansen-Norskov (1999) [*27* (page 908)]

- **revPBEx**: revised PBE by Zhang-Wang (1998) [*28* (page 908)]

- **mPBEx**: Modified PBE by Adamo-Barone (2002) [*174* (page 917)]

- **PBEsolx**: Perdew-Ruzsinszky-Csonka-Vydrov-Scuseria (2008) [*285* (page 923)]

- **HTBSx**: [*437* (page 932)]

- **OPTX**: Handy-Cohen (2001) [*29* (page 908)]

- **BEEx**: Mortensen-Kaasbjerg-Frederiksen-Nørskov-Sethna-Jacobsen (2005) [*284* (page 923)]

For the **correlation** part the options are:

- **Perdew**: Perdew (1986) [*30* (page 908)].

- **PBEc**: Perdew-Burke-Ernzerhof (1996) [*26* (page 908)] .

- **PBEsolc**: The PBEsol correlation correction by Perdew-Ruzsinszky-Csonka-Vydrov-Scuseria (2008) [*285* (page 923)]

- **PW91c**: Perdew-Wang (1991), see [*24* (page 908)].

- **LYP**: Lee-Yang-Parr (1988) correlation correction [*31-33* (page 908)].

The string GGA must contain not more than one of the exchange options and not more than one of the correlation options. If options are applied for both they must be separated by a blank or a comma. Example:

```
XC
  GGA Becke Perdew
End
```

is equivalent to

```
XC
  GGA BP86
End
```

It is questionable to apply gradient corrections to the *correlation*, while not doing so at the same time for the exchange. Therefore the program will check this and stop with an error message. This check can be overruled with the key ALLOW.

4.2.3 MetaGGA

```
XC
  MetaGGA functional
End
```

MetaGGA Specifies that a meta-GGA should be used during the SCF. **All electron basis sets should be used** (see *Basis key* (page 33)).

Available meta-GGA functionals:

- **M06-L**: Functional by Yan-Truhlar [*223,224* (page 919)]
- **TPSS**: Functional by Tao-Perdew-Staroverov-Scuseria [*246,247* (page 921)]
- **revTPSS**: Revised TPSS functional [*438* (page 932)]
- **SSB-D**: Dispersion corrected GGA functional by Swart-Solà-Bickelhaupt [*286,287* (page 923)]. Use GGA SSB-D for NMR calculations.
- **MVS**: Functional by Sun-Perdew-Ruzsinszky [*395* (page 930)]
- **MS0**: Functional by Sun et al. [*505* (page 936)]
- **MS1**: Functional by Sun et al. [*506* (page 936)]
- **MS2**: Functional by Sun et al. [*506* (page 936)]
- **SCAN**: Functional by Sun et al. [*507* (page 936)]

The M06-L functional needs high integration accuracy (at least *BeckeGrid quality good* (page 300)) for reasonable gradients. For TPSS moderate integration accuracy for reasonable gradients is sufficient. For heavier elements (Z>36) and if one uses the M06-L functional it is also necessary to include the following keyword

```
FragMetaGGAToten
```

Using this key FRAGMETAGGATOTEN the difference in the meta-hybrid or meta-GGA exchange-correlation energies between the molecule and its fragments will be calculated using the molecular integration grid, which is more accurate than the default, but is much more time consuming. Default is to calculate the meta-GGA exchange-correlation energies for the fragments in the numerical integration grid of the fragments.

4.2.4 Hartree-Fock

```
XC
  HartreeFock
End
```

HartreeFock Specifies that the Hartree-Fock exchange should be used during the SCF.

4.2.5 Hybrid

```
XC
  HYBRID functional {HF=HFpart}
End
```

HYBRID Specifies that a hybrid functional should be used during the SCF.

Available Hybrid functionals:

- **B3LYP**: ADF uses VWN5 in B3LYP. functional (20% HF exchange) by Stephens-Devlin-Chablowski-Frisch [*176* (page 917)].

- **B3LYP***: Modified B3LYP functional (15% HF exchange) by Reiher-Salomon-Hess [*177* (page 917)].

- **B1LYP**: Functional (25% HF exchange) by Adamo-Barone [*178* (page 917)].

- **KMLYP**: Functional (55.7% HF exchange) by Kang-Musgrave [*179* (page 917)].

- **O3LYP**: Functional (12% HF exchange) by Cohen-Handy [*180* (page 917)].

- **X3LYP**: Functional (21.8% HF exchange) by Xu-Goddard [*172* (page 917)].

- **BHandH**: 50% HF exchange, 50% LDA exchange, and 100% LYP correlation.

- **BHandHLYP**: 50% HF exchange, 50% LDA exchange, 50% Becke88 exchange, and 100% LYP correlation.

- **B1PW91**: Functional by (25% HF exchange) Adamo-Barone [*178* (page 917)].

- **mPW1PW**: Functional (25% HF exchange) by Adamo-Barone [*25* (page 908)].

- **mPW1K**: Functional (42.8% HF exchange) by Lynch-Fast-Harris-Truhlar [*181* (page 917)].

- **PBE0**: Functional (25% HF exchange) by Ernzerhof-Scuseria [*211* (page 919)] and by Adamo-Barone [*212* (page 919)], hybrid form of PBE.

- **OPBE0**: Functional (25% HF exchange) by Swart-Ehlers-Lammertsma [*175* (page 917)], hybrid form of OPBE.

- **S12H**: Dispersion corrected (Grimme-D3) functional (25% HF exchange) by Swart [*367* (page 928)].

HFpart Specifies the amount of HF exchange that should be used in the functional, instead of the default HF exchange percentage for the given hybrid. Example HF=0.25 means 25% Hartree-Fock exchange.

4.2.6 Meta-Hybrid

```
XC
  MetaHYBRID functional
End
```

MetaHYBRID Specifies that a meta-hybrid functional should be used during the SCF.

Available meta-hybrid functionals:

- **M06**: Functional (27% HF exchange) by Yan-Truhlar [*223,224* (page 919)]

- **M06-2X**: Functional (54% HF exchange) by Yan-Truhlar [*223,224* (page 919)]

- **M06-HF**: Functional (100% HF exchange) by Yan-Truhlar [*223,224* (page 919)]

- **TPSSH**: Functional (10% HF exchange) by Tao-Perdew-Staroverov-Scuseria [*246,247* (page 921)]

4.2.7 Range separated hybrids

In ADF there are two (mutually exclusive) ways of specifying range separated hybrids functionals:

- Through the `RANGESEP` and `XCFUN` keys. This will use the Yukawa potential as switching function, see Ref. [*355* (page 928)];

- By specifying a range separated functional via the `LibXC` key.

See also the advanced tutorial: Tuning the range separation in LC-wPBE for organic electronics (https://www.scm.com/news/tuning-range-separation-lc-wpbe-organic-electronics/)

RangeSep + XCFun: Yukawa-range separated hybrids

RANGESEP {GAMMA=X} {ALPHA=a} {BETA=b} If RANGESEP is included, by default a long-range corrected (LC) functional is created with range separation parameter GAMMA of 0.75. As switching function in ADF the Yukawa potential is utilized, see Ref. [*355* (page 928)]. Range separated functionals require XCFUN and are limited to GGA, meta-GGA, and CAMY-B3LYP. The CAMY-B3LYP functional is not the same as the CAM-B3LYP functional, since a different switching function is used. No other hybrids or meta-hybrids are supported. The special CAMYB3LYP functional is defined by three parameters, ALPHA, BETA and the attenuation parameter GAMMA. For CAMYB3LYP by default ALPHA is 0.19, BETA is 0.46, and GAMMA is 0.34.

Range-separated functionals make use of a modified form of the Coulomb operator that is split into pieces for exact exchange and DFT. As switching function in ADF the Yukawa potential is utilized, see Ref. [*355* (page 928)]. Global hybrids can be thought of as a special case of a range-separated functional where the split is independent of the interelectronic distance and is a simple X% exact and 1-X% DFT in all space.

In a general RS-functional the split depends on the interelectronic distance. How the split is achieved depends on the functional in question but it is achieved using functions that smoothly go from 1 to 0. In ADF an exponential function is used (the error function is common in Gaussian based codes). In a range-separated function the potential goes from a Coulomb interaction to a sum of Coulomb functions attenuated by an exponential function.

In practical terms, this means that a range-separated functional looks very much like a hybrid (or meta-hybrid) functional but with additional integrals over the attenuated interaction with fit functions on the exact exchange side and a modified functional on the DFT side.

DFT part of RS-functionals

Using Hirao's approach for creating RS-functionals, the RS form of a given exchange functional is created by multiplying the standard energy density by a factor that depends on the energy density. The factor is the same for all functionals and the only difference is introduced by the individual energy densities.

The range-separation comes in at the level of the integrals over the operator with fit functions. They are very similar to the standard Coulomb integrals.

RS-functionals

An RS-functional is described by a series of regions describing each of the pieces of the Coulomb operator. The total function is built up by looping over the regions and adding up all the pieces. Currently, simple LC functionals can be defined where the exact exchange goes from 0 to 1 as the interelectronic distance increases and the DFT part does the reverse. In addition CAMY-B3LYP type functionals can be defined. More general functionals are not possible yet.

Functionality/Limitations

RS functionals with XCFUN are limited to the GGA and meta-GGA functionals and one hybrid CAMY-B3LYP. The following functionals can be evaluated with range-separation at the present time:

- LDA: VWN5, X-ALPHA PW92

- GGA exchange: Becke88, PBEX, OPTX, PW91X, mPW, revPBEX

- GGA correlation: LYP, Perdew86, PBEC

- MetaGGA: TPSS, M06L, B95

- Hybrids: CAMY-B3LYP

The following functionality has been tested: XC potential, energy, ground state geometry, TDDFT. Starting from ADF2018 singlet-triplet excitation calculations and excited state geometry optimizations are possible. See for possible

limitations in case of excitation calculations or excited state geometry optimizations the corresponding part of the ADF manual.

Numerical stability

The range-separated implementation requires that the range-separation parameter is not too close to the exponent of a fit function. In practice this means that values of the separation parameter between 1.0 and 50 can cause numerical problems. Typical useful values are in the range 0.2 to 0.9 so this should not be too serious a limitation.

```
XC
  XCFUN
  RANGESEP {GAMMA=X} {ALPHA=a} {BETA=b}
END
```

Range separation is activated by putting RANGESEP in the XC block. Inclusion of XCFUN is required, see the *XCFUN description* (page 65). By default a long-range corrected (LC) functional is created with range separation parameter of 0.75. The parameter can be changed by modifying X in GAMMA=X in the RANGESEP card. Range separation typically will be used in combination with a GGA or METAGGA functional.

Range separation can not be included with a hybrid or meta-hybrid, with one exception, the special RS functional: CAMY-B3LYP. This is entered as HYBRID CAMY-B3LYP and must be used in combination with XCFUN (see *XCFUN description* (page 65)) and RANGESEP. The CAMY-B3LYP functional is defined by three parameters, alpha, beta and the attenuation parameter gamma. The gamma parameter can be modified as for the LC functionals. For CAMY-B3LYP it defaults to 0.34. The alpha and beta parameters can be modified through ALPHA=a and BETA=b in the RANGESEP card. They default to 0.19 and 0.46 respectively.

```
XC
  HYBRID CAMY-B3LYP
  XCFUN
  RANGESEP GAMMA=0.34 ALPHA=0.19 BETA=0.46
END
```

List of the most important functionals, for which one can use range separation:

```
LDA VWN
GGA BLYP
GGA BP86
GGA PBE
HYBRID CAMY-B3LYP
```

Range-separated hybrids with LibXC

One can simply specify a range separated hybrid functional in the *LibXC* (page 65) key, *e.g.*:

```
XC
  LibXC CAM-B3LYP
End
```

ADF will automatically use the (non default) *new Hartree-Fock RI scheme* (page 306). See the *LibXC* (page 65) section for a list of available range separated hybrid functionals.

For the HSE03 and HSE06 short range-separated hybrids you can (optionally) specify the switching parameter omega, *e.g.*:

```
XC
  LibXC HSE06 omega=0.1
End
```

4.2.8 Notes on Hartree-Fock and (meta-)hybrid functionals

If a functional contains a part of Hartree-Fock exchange then the LDA, GGA, metaGGA, or MODEL key should not be used in combination with this key, and one should only specify one of HartreeFock, HYBRID or MetaHYBRID. Dispersion can be added. Note that it is not recommended to use (part of the) Hartree-Fock exchange in combination with frozen cores, since at the moment the frozen core orbitals are not included in the Hartree Fock exchange operator. In ADF one can do unrestricted Hartree-Fock (or hybrid or meta-hybrid) calculations, as long as one has integer occupation numbers (ROHF is not implemented in ADF, only UHF). Note that the DEPENDENCY key is switched on for Hartree-Fock exchange in order to circumvent numerical problems, see also the ADDDIFFUSEFIT key. You also need to the same XC-potential in the create run of the atoms, which is done automatically if you use the BASIS key.

Starting from ADF2009.01 the meta-hybrids M06, M06-2X, M06-HF, and TPSSH can be used during the SCF. Also starting from ADF2009.01 Hartree-Fock and the (meta-)hybrid potentials can be used in combination with geometry optimization, TS, IRC, LT, and numerical frequencies; hybrids can be used in calculating *NMR chemical shift* (page 203); PBE0 can be used in calculating *NMR spin-spin coupling* (page 209); Hartree-Fock and (meta-)hybrid can be used in calculating excitation energies, in which the kernel consists of the Hartree-Fock percentage times the Hartree-Fock kernel plus one minus the Hartree-Fock percentage times the ALDA kernel (thus no (meta-)GGA kernel). Hartree-Fock and the (meta-)hybrid potentials still can not or should not be used in combination with analytical frequencies, the (AO)RESPONSE key, EPR/ESR g-tensor, and frozen cores. Starting from ADF2010 it is possible to use Hartree-Fock and hybrids to calculate CD spectra.

In ADF one can do unrestricted Hartree-Fock (or hybrid or meta-hybrid) calculations, as long as one has integer occupation numbers (ROHF is not implemented in ADF, only UHF).

Starting from ADF2009.01 it is possible to change the amount of HF exchange in the input for hybrids (not for meta-hybrids and Hartree-Fock). For many hybrid functionals the sum of the amount of Hartree-Fock exchange and the amount of LDA exchange (or GGA exchange) is one. If that is the case, then if one changes the amount of Hartree-Fock exchange in the input the amount of LDA exchange (or GGA exchange) will also be changed, such that the sum remains one. Example:

```
XC
  Hybrid B3LYP HF=0.25
END
```

In this case the amount of Hartree-Fock for the B3LYP functional will be changed to 25% (instead of 20%), and the amount of LDA exchange to 75% (instead of 80%). The LDA correlation and GGA exchange and correlation part will be left unaltered.

An accuracy issue is relevant for some of the meta-GGA functionals, in particular the M06 functionals. These need high integration accuracy (at least BeckeGrid quality good) for reasonable gradients. For TPSSH moderate integration accuracy for reasonable gradients is sufficient. For heavier elements (Z>36) and if one uses one of the M06 functionals it is also necessary to include the following keyword

```
FragMetaGGAToten
```

Using this key `FRAGMETAGGATOTEN` the difference in the metahybrid or metagga exchange-correlation energies between the molecule and its fragments will be calculated using the molecular integration grid, which is more accurate than the default, but is much more time consuming. Default is to calculate the meta-hybrid or meta-GGA exchange-correlation energies for the fragments in the numerical integration grid of the fragments.

For benchmark calculations one would like to use a large basis set, like the QZ4P basis set. In such cases it is recommended to use a good numerical quality. Thus for accurate hybrid calculations of **small** molecules one could use:

```
basis
  type QZ4P
```

```
end
AddDiffuseFit
Dependency bas=1e-4
NumericalQuality good
```

4.2.9 Model Potentials

Several asymptotically correct XC potentials have been implemented in ADF, such as the (now somewhat outdated) LB94 potential [*15* (page 907)], the gradient-regulated asymptotic correction (GRAC) [*16* (page 907)], and the statistical average of orbital potentials (SAOP) [*244* (page 921), *17* (page 908)]. These can currently be used only for response property calculations, not for geometry optimizations. For spectroscopic properties, they usually give results superior to those obtained with LDA or GGA potentials, (see Ref.[*18* (page 908)] for applications to (hyper)polarizabilities Cauchy coefficients, etc. of small molecules). This is particularly true if the molecule is small and the (high-lying) virtual orbitals are important for the property under study.

It was also shown that, simply using the orbital energies of the occupied Kohn-Sham orbitals of a SAOP calculation, quite good agreement with experiment vertical ionization potentials is obtained. This is true not only for the HOMO orbital energy, which should be identical to (minus) the experimental ionization potential with the exact XC potential, but also for lower-lying occupied orbital energies. The agreement becomes worse for deep-lying core orbital energies. A theoretical explanation and practical results are given in Ref. [*19* (page 908)].

```
XC
  Model ModelPotential [IP]
End
```

MODEL Specifies that one of the less common XC potentials should be used during the SCF. These potentials specify both the exchange and the correlation part. No LDA, GGA, MetaGGA, HartreeFock, HYBRID or MetaHYBRID key should be used in combination with these keys. It is also not advised to use any energy analysis in combination with these potentials. For energy analysis we recommend to use one of the GGA potentials. It is currently not possible to do a Create run with these potentials. It is possible to do a one atom regular ADF calculation with these potentials though, using a regular TAPE21 file from an LDA or GGA potential as input. Available model potentials:

- **LB94**: This refers to the XC functional of Van Leeuwen and Baerends [*15* (page 907)]. There are no separate entries for the Exchange and Correlation parts respectively of LB94. Usually the GRAC or SAOP potentials give results superior to LB94.

- **GRAC**: The gradient-regulated asymptotic correction, which in the outer region closely resembles the LB94 potential [*16* (page 907)]. It requires a further argument: the ionization potential [IP] of the molecule, in hartree units. This should be estimated or obtained externally, or calculated in advance from two GGA total energy calculations.

- `IP`:Should be supplied only if GRAC is specified.

- **SAOP**: The statistical average of orbital potentials [*244* (page 921), *17* (page 908)]. It can be used for all electron calculations only. It will be expensive for large molecules, but requires no further parameter input.

The LB94, GRAC, and SAOP functionals have only a SCF (=Potential) implementation, but **no Energy** counterpart.

The LB94, GRAC, and SAOP forms are density functionals specifically designed to get the correct asymptotic behavior. This yields much better energies for the highest occupied molecular orbital (HOMO) and better excitation energies in a calculation of response properties (Time Dependent DFT). Energies for lower lying orbitals (sub-valence) should improve as well (in case of GRAC and SAOP, but not LB94). The energy expression underlying the LB94 functional is very inaccurate. This does not affect the response properties but it does imply that the energy and its derivatives (gradients) should not be used because LB94-optimized geometries will be wrong, see for instance [*34* (page 908)].

The application of the LB94 functional in a runtype that involves the computation of energy gradients is disabled in ADF. You can override this internal check with the key *ALLOW* (page 310).

In case of a GRAC calculation, the user should be aware that the potential in the outer region is shifted up with respect to the usual level. In other words, the XC potential does not tend to zero in the outer region in this case. The size of the shift is the difference between the HOMO orbital energy and the IP given as input. In order to compare to regular GGA orbital energies, it is advisable to subtract this amount from all orbital energies. Of course, orbital energy differences, which enter excitation energies, are not affected by this shift in the potential.

4.2.10 Optimized effective potentials

```
XC
  OEP fitmethod {approximation}
End
```

OEP Defines the optimized effective potential expanded into a set of the fit functions. The subkeyword fitmethod can be any of the following: BARTLETT [*248* (page 921)], SCUSERIA [*249* (page 921)]. In the case of SCUSERIA one of the following approximations needs to be specified: CEDA, KLI or SLATER. An application of OEP in ADF can be found in Ref.[*250* (page 921)].

4.2.11 XCFun

XCFUN XCFun is a library of approximate exchange-correlation functionals, see Ref. [*354* (page 927)], for which functional derivatives can be calculated automatically. For example, with XCFUN the full (non-ALDA) kernel can be evaluated and this has been implemented in the calculation of TDDFT excitations. The Full kernel can not be used in combination with symmetry or excited state geometry optimizations. The following functionals can be evaluated with XCFUN at the present time:

- LDA: VWN5, X-ALPHA, PW92

- GGA exchange: Becke88, PBEX, OPTX, PW91X, mPW, revPBEX

- GGA correlation: LYP, Perdew86, PBEC

- MetaGGA: TPSS, M06L, B95

- MetaHybrids: M06, M05, M062X, M06HF

- Hybrids: PBE0, B3LYP, BHandH, B1LYP, B3LYP*, PBEFALFX

- Yukawa range separated Hybrids: CAMY-B3LYP and more, see *Yukawa RS hybrids with XCFUN* (page 61)

Here MetaGGA B95 means Becke88 exchange + B95c correclation. The Metahybrids PW6B95 and PWB6K have been removed from this list, since they do not agree with the LibXC implementation.

4.2.12 LibXC

LibXC functional LibXC is a library of approximate exchange-correlation functionals, see Ref. [*392* (page 930), *488* (page 935)]. **All electron basis sets should be used** (see *Basis key* (page 33)). Version 4.2.1 of LibXC is used. The interface in ADF to LibXC is incompatible with LibXC version 2. The following functionals can be evaluated with LibXC (incomplete list):

- LDA: LDA, PW92, TETER93

- GGA: AM05, BGCP, B97-GGA1, B97-K, BLYP, BP86, EDF1, GAM, HCTH-93, HCTH-120, HCTH-147, HCTH-407, HCTH-407P, HCTH-P14, PBEINT, HTBS, KT2, MOHLYP, MOHLYP2, MPBE, MPW, N12, OLYP, PBE, PBEINT, PBESOL, PW91, Q2D, SOGGA, SOGGA11, TH-FL, TH-FC, TH-FCFO, TH-FCO, TH1, TH2, TH3, TH4, XLYP, XPBE, HLE16

- MetaGGA: M06-L, M11-L, MN12-L, MS0, MS1, MS2, MVS, PKZB, TPSS, HLE17

- Hybrids: B1LYP, B1PW91, B1WC, B3LYP, B3LYP*, B3LYP5, B3LYP5, B3P86, B3PW91, B97, B97-1 B97-2, B97-3, BHANDH, BHANDHLYP, EDF2, MB3LYP-RC04, MPW1K, MPW1PW, MPW3LYP, MPW3PW, MPWLYP1M, O3LYP, OPBE, PBE0, PBE0-13, REVB3LYP, REVPBE, RPBE, SB98-1A, SB98-1B, SB98-1C, SB98-2A, SB98-2B, SB98-2C, SOGGA11-X, SSB, SSB-D, X3LYP

- MetaHybrids: B86B95, B88B95, BB1K, M05, M05-2X, M06, M06-2X, M06-HF, M08-HX, M08-SO, MPW1B95, MPWB1K, MS2H, MVSH, PW6B95, PW86B95, PWB6K, REVTPSSH, TPSSH, X1B95, XB1K

- Range-separated: CAM-B3LYP, CAMY-B3LYP, HJS-PBE, HJS-PBESOL, HJS-B97X, HSE03, HSE06, LRC_WPBE, LRC_WPBEH, LCY-BLYP, LCY-PBE, M11, MN12-SX, N12-SX, TUNED-CAM-B3LYP, WB97, WB97X

One of the acronyms in the list above can be used, or one can also use the functionals described at the LibXC website http://www.tddft.org/programs/Libxc. Note that ADF can not calculate VV10 dependent LibXC functionals, like VV10, LC-VV10, B97M-V, WB97X-V. Example usage for the BP86 functional:

```
XC
  LibXC BP86
End
```

Alternative

```
XC
  LibXC XC_GGA_X_B88 XC_GGA_C_P86
End
```

In case of LibXC the output of the ADF calculation will give the reference for the used functional, see also the LibXC website http://www.tddft.org/programs/Libxc.

Do not use any of the subkeys LDA, GGA, METAGGA, MODEL, HARTREEFOCK, OEP, HYBRID, METAHYBRID, XCFUN, RANGESEP in combination with the subkey LIBXC. One can use the DISPERSION key icw LIBXC. For a selected number of functionals the optimized dispersion parameters will be used automatically, please check the output in that case.

The LibXC functionals can not be used icw frozen cores, NMR calculations, the (AO)RESPONSE key, EPR/ESR g-tensor. Most LibXC functionals can be used in combination with geometry optimization, TS, IRC, LT, numerical frequencies, and excitation energies (ALDA kernel used). For a few GGA LibXC functionals analytical frequencies can be calculated, and one can use the full kernel in the calculation of excitation energies (if FULLKERNEL is included as subkey of the key EXCITATIONS). In case of LibXC (meta-)hybrids and calculating excitation energies, the kernel consists of the Hartree-Fock percentage times the Hartree-Fock kernel plus one minus the Hartree-Fock percentage times the ALDA kernel (thus no (meta-)GGA kernel). For the LibXC range separated functionals, like CAM-B3LYP, starting from ADF2016.102 the kernel consists of range separated ALDA plus the kernel of the range separated exact exchange part. In ADF2016.101 the kernel for LibXC range separated functionals, like CAM-B3LYP, was using a 100% ALDA plus range separated exact exchange kernel (the ALDA part was not range-separated corrected). For the range separated functionals WB97, WB97X, and WB97X-V one can use the full kernel in the calculation of excitation energies.

4.2.13 Dispersion corrections

DISPERSION Grimme3 BJDAMP If DISPERSION Grimme3 BJDAMP is present a dispersion correction (DFT-D3-BJ) by Grimme [*436* (page 932)] will be added to the total bonding energy, gradient and second derivatives, where applicable. This is the latest dispersion correction in the DFTB-D family. Parametrizations are implemented e.g. for B3LYP, TPSS, BP86, BLYP, PBE, PBEsol, and RPBE. For SCAN parameters from Ref. [*508* (page 936)] are used. The parametrization has four parameters. One can override this using *PAR1=.. PAR2=..*, etc. In the table the relation is shown between the parameters and the real parameters in the Dispersion correction.

variable	variable on Bonn website (https://www.chemie.uni-bonn.de/pctc/mulliken-center/software/dft-d3/functionalsbj)
PAR1	s6
PAR2	a1
PAR3	s8
PAR4	a2

DISPERSION Grimme3 If DISPERSION Grimme3 is present a dispersion correction (DFT-D3) by Grimme [*292* (page 924)] will be added to the total bonding energy, gradient and second derivatives, where applicable. Parametrizations are available e.g. for B3LYP, TPSS, BP86, BLYP, revPBE, PBE, PBEsol, and RPBE, and will be automatically set if one of these functionals is used. There are also parameters directly recognized for S12g and S12h. For SCAN parameters from Ref. [*508* (page 936)] are used. For all other functionals, PBE-D3 parameters are used as default. You can explicitly specify the three parameters.

variable	variable on Bonn website (https://www.chemie.uni-bonn.de/pctc/mulliken-center/software/dft-d3/functionals)
PAR1	s6
PAR2	sr,6
PAR3	s8

DISPERSION {s6scaling]} {RSCALE=r0scaling} If the DISPERSION keyword is present (without the argument Grimme3) a dispersion correction (DFT-D) by Grimme [*226* (page 920)] will be added to the total bonding energy, gradient and second derivatives, where applicable. The global scaling factor with which the correction is added depends on the exchange-correlation functional used at SCF but it can be modified using the *s6scaling* parameter. The following scaling factors are used (with the XC functional in parantheses): 1.20 (BLYP), 1.05 (BP), 0.75 (PBE), 1.05 (B3LYP). In all other cases a factor 1.0 is used unless modified via the s6scaling parameter. The SSB-D functional includes the dispersion correction (factor 0.847455) by default.

The van der Waals radii used in this implementation are hardcoded in ADF. However, it is possible to modify the global scaling parameter for them using the *RSCALE=r0scaling* argument. The default value is 1.1 as proposed by Grimme [*226* (page 920)]. Please also see *additional documentation* (page 67) for more information about this topic.

DISPERSION dDsC The DISPERSION dDsC key invokes the density dependent dispersion correction [*316* (page 925)], which has been parametrized for the functionals BLYP, PBE, BP, revPBE, B3LYP, PBE0 and BHANDHLYP.

DISPERSION UFF The DISPERSION UFF key invokes the universal correction of density functional theory to include London dispersion (DFT-ulg) [*366* (page 928)], which has been parametrized for all elements up to Lr (Z=103), and for the functional PBE, PW91, and B3LYP. For other functionals the PBE parameters will be used.

DISPERSION MBD The DISPERSION MBD key invokes the MBD@rsSCS method [*377* (page 929)], which is designed to accurately describe long-range correlation (and thus dispersion) in finite-gap systems, including at the same time a description of the short-range interactions from the underlying DFT computation of the electronic structure.

DFT-D3 functionals

In ADF2012 Stefan Grimme's latest dispersion correction is implemented. Grimme and his coworkers at the Universität Münster outlined the parametrization of this new correction, dubbed DFT-D3, in Ref. [*292* (page 924)]. A slightly improved version with a more moderate BJ damping function appeared later, and was called DFTB-D3-BJ. [*436* (page 932)] Here they list the advantages of the new method as the following:

- It is less empirical, i.e., the most important parameters are computed from first principles by standard Kohn-Sham (KS)-(TD)DFT.

- The approach is asymptotically correct with all DFs for finite systems (molecules) or nonmetallic infinite systems. It gives the almost exact dispersion energy for a gas of weakly interacting neutral atoms and smoothly interpolates to molecular (bulk) regions.

- It provides a consistent description of all chemically relevant elements of the periodic system (nuclear charge Z = 1-94).

- Atom pair-specific dispersion coefficients and cutoff radii are explicitly computed.

- Coordination number (geometry) dependent dispersion coefficients are used that do not rely on atom connectivity information (differentiable energy expression).

- It provides similar or better accuracy for "light" molecules and a strongly improved description of metallic and "heavier" systems.

DFT-D3-BJ is invoked with the XC block, for example

```
XC
  GGA BLYP
  Dispersion Grimme3 BJDAMP
END
```

Parametrizations are available for: B3LYP, TPSS, BP86, BLYP, revPBE, PBE, PBEsol, RPBE, and some more functionals, and will be automatically set if one of these functionals is used. Otherwise PBE parameters will be used. The parameters can be set manually, see the XC key block. In ADF2016 parameters for Grimme3 and Grimme3 BJDAMP were updated according to version 3.1.1 of the coefficients, available at the Bonn website (https://www.chemie.uni-bonn.de/pctc/mulliken-center/software/dft-d3/dft-d3)

DFT-D functionals

An implementation for dispersion corrections based, called DFT-D is available starting from ADF2008. Like DFT-D3 this implementation is easy to use and is also supported by the GUI.

This DFT-D implementation is based on the paper by Grimme [*226* (page 920)] and is extremely easy to use. The correction is switched on by specifying *DISPERSION*, possibly with parameters, in the XC input block. See *description of the XC input block* (page 55) for details about the DISPERSION keyword.

Energies calculated Post-SCF using different DFT-D or GGA-D functionals are also present in table printed when METAGGA keyword is specified. These include: BLYP-D, PBE-D, BP86-D, TPSS-D, B3LYP-D, and B97-D. NOTE: this option does not require specifying a DISPERSION keyword in the XC block and thus there is **no correction added to the energy gradient** in this case. Please also note that although the original B97 functional includes HF exchange (and is thus a hybrid functional), the B97-D is a pure GGA. B3LYP-D is, however, a hybrid functional. The following functional-dependent global scaling factors s_6 are used: 1.2 (BLYP-D), 0.75 (PBE-D), 1.05 (BP86-D), 1.0 (TPSS-D), 1.05 (B3LYP-D), and 1.25 (B97-D). These are fixed and cannot be changed.

Regarding performance of different functionals, testing has shown that BLYP-D gives good results for both energies and gradients involving VdW interactions. Post-SCF energy-only calculations at fixed geometries showed that also B97-D gives good binding energies compared to high-level reference data. Thorough comparison of different DFT-D functionals can be found in ref. [*227* (page 920)]

Note: The original paper by Grimme included parameters for elements H throughout Xe. In ADF2009.01 values for dispersion parameters for DFT-D functionals for heavier elements (Cs-Rn) have been added. These new values have not been tested extensively. Thus, in this implementation, no dispersion correction is added for interactions involving atoms heavier than Radon.

DFT-D is invoked with the XC block, for example

```
XC
  GGA BLYP
  Dispersion
END
```

MM dispersion (old implementation)

For the documentation on MM-dispersion see the old ADF 2014 documentation (http://www.scm.com/Doc/Doc2014/ADF/ADFUsersGuide/ADFUsersGuide.pdf).

dDsC: density dependent dispersion correction

The DISPERSION dDsC key invokes the density dependent dispersion correction [*316* (page 925)], which has been parametrized for the functionals BLYP, PBE, BP, revPBE, B3LYP, PBE0 and BHANDHLYP.

```
XC
  GGA BLYP
  Dispersion dDsC
END
```

For other functionals one can set the dDsC parameters ATT0 and BTT0 with

```
XC
  ...
  DISPERSION dDsC ATT0=att0 BTT0=btt0
END
```

The dispersion dDsC in ADF can not be used icw fragments larger than 1 atom. The reason is that ADF uses the Hirshfeld partitioning on fragments for dDsC, which is only correct if the fragments are atoms.

DFT-ulg

The DISPERSION UFF key invokes the universal correction of density functional theory to include London dispersion (DFT-ulg) [*366* (page 928)], which has been parametrized for all elements up to Lr (Z=103), and for the functional PBE, PW91, and B3LYP. For other functionals the PBE parameters will be used. Example:

```
XC
  GGA PBE
  Dispersion UFF
END
```

DFT-MBD functionals

The DISPERSION MBD key invokes the MBD@rsSCS method [*377* (page 929)], which is designed to accurately describe long-range correlation (and thus dispersion) in finite-gap systems, including at the same time a description of the short-range interactions from the underlying DFT computation of the electronic structure. The MBD (many-body

dispersion) method [*376* (page 929)] obtains an accurate description of van der Waals (vdW) interactions that includes both screening effects and treatment of the many-body vdW energy to infinite order. The revised MBD@rsSCS method [*377* (page 929)] employs a range-separation (rs) of the self-consistent screening (SCS) of polarizabilities and the calculation of the long-range correlation energy. It has been parametrized for the elements H-Ba, Hf-Rn, and for the functional PBE and PBE0. Note that the MBD@rsSCS method depends on Hirshfeld charges. In calculating forces the dependence of the Hirshfeld charges on the actual geometry is neglected. The MBD method is implemented in case the BeckeGrid is used for the numerical integration. Example for PBE MBD@rsSCS:

```
XC
  GGA PBE
  Dispersion MBD
END
```

One can use user defined values with:

```
XC
  Dispersion MBD {RSSCS|TS} {BETA=beta}
END
```

MBD {RSSCS|TS} {BETA=beta} The default method for MBD is MBD@rsSCS. Optionally one can use MBD@TS or change the used parameter β with setting beta.

4.2.14 Self-Interaction Correction

For the documentation on Self-Interaction Corrections see the old ADF 2014 documentation (http://www.scm.com/Doc/Doc2014/ADF/ADFUsersGuide/ADFUsersGuide.pdf).

4.2.15 Post-SCF energy functionals

GGA energy functionals

In principle you may specify different functionals to be used for the *potential,* which determines the self-consistent charge density, and for the *energy* expression that is used to evaluate the (XC part of the) energy of the charge density. To be consistent, one should generally apply the same functional to evaluate the potential and energy respectively. Two reasons, however, may lead one to do otherwise:

- The evaluation of the GGA part in the *potential* is more time-consuming than LDA. The effect of the GGA term in the potential on the self-consistent charge density is often not very large. From the point of view of computational efficiency it may, therefore, be attractive to solve the SCF equations at the LDA level (i.e. not including GGA terms in the potential), and to apply the full expression, including GGA terms, to the energy evaluation *a posteriori*: post-SCF.

- A particular XC functional may have only an implementation for the potential, but not for the energy (or vice versa). This is a rather special case, intended primarily for fundamental research of Density Functional Theory, rather than for run-of-the-mill production runs.

One possibility is to calculate a whole list of post-SCF energy functionals using the METAGGA keyword, see next section. For some functionals the next possibility is enough. One has to specify different functionals for potential and energy evaluations respectively, using:

```
XC
 {LDA {Apply} LDA {Stoll}}
 {GGA {Apply} GGA}
end
```

Apply States whether the functional defined on the pertaining line will be used self-consistently (in the SCF-potential), or only post-SCF, i.e. to evaluate the XC energy corresponding to the charge density. The value of apply must be SCF or Energy. A value postSCF will also be accepted and is equivalent to Energy. A value Potential will also be accepted and is equivalent to SCF. For each record separately the default (if no Apply value is given in that record) is SCF. For each of the two terms (LDA, GGA) in the functional: if no record with Energy specification is found in the data block, the evaluation of the XC energy will use the same functional as is applied for the potential.

LDA, GGA See the XC potential section for all possible values.

Meta-GGA and hybrid energy functionals

The post SCF energy calculation is an easy and cheap way to get a reasonable guess for the bond energies for different XC functionals at the same time. Note that post-SCF energy calculations for a certain XC functional will not be so accurate if the functional form of the XC functional used in the SCF is very different from the XC functional used post SCF. The relative accuracy of post-SCF energies may not be so high if one looks at small energy differences. For accurate energy calculations it is recommended to use the same XC functional during the SCF as for the energy.

The calculation of a large, pre-specified list of LDA, GGA, and meta-GGA energy functionals is invoked by specifying

```
METAGGA
```

as a separate keyword. The following (incomplete) list gives an idea of the (meta-)GGA density functionals that will then be calculated (the t-MGGA functional is the θ-MGGA functional of Ref.[*388* (page 929)]):

```
BP, PW91, mPW, BLYP, PBE, RPBE, revPBE, mPBE, OLYP, OPBE, KCIS, PKZB, VS98, FT97,␣
→BLAP3,
HCTH, tau-HCTH, BmTau1, BOP, OLAP3, TPSS, KT1, KT2, B97, M06-L, t-MGGA.
```

The hybrid GGA and hybrid meta-GGA energy functionals are calculated if in addition to the METAGGA key, the key

```
HARTREEFOCK
```

is included. The following (incomplete) list gives an idea of the extra hybrid (meta-)GGA density functionals that will then be calculated:

```
B3LYP, B3LYP*, B1LYP, KMLYP, O3LYP, X3LYP, BHandH, BHandHLYP, B1PW91, MPW1PW, MPW1K,
PBE0, OPBE0, TPSSh, tau-HCTH-hybrid, B97, M05, M05-2X, M06, M06-2X.
```

The keys METAGGA and HARTREEFOCK can be used in combination with any XC potential. Note that at the moment hybrid functionals can not be used in combination with frozen cores. Also most METAGGA functionals will give wrong results if used in combination with frozen cores. Thus it is best to use an all electron basis set if one of the keywords METAGGA or HARTREEFOCK is used. One should include the `HARTREEFOCK` keyword also in the create runs of the atoms. In ADF the hybrid energies only make sense if the calculation is performed with completely filled orbitals (ROHF is not implemented in ADF, only UHF).

The Examples document describes an application to the OH molecule for the METAGGA option. More output, on the total XC energy of the system, can be obtained by specifying

```
PRINT METAGGA
```

This latter option is intended for debugging purposes mainly and is not recommended for general use.

The implementation calculates the total XC energy for a system and writes it to a file. This is always done in Create runs. If the basic fragments are atoms, the keyword

```
ENERGYFRAG
   ATOM [filename]
   ATOM [filename]
   ... ...
END
```

specifies that different atomic fragment files are to be used in the meta-GGA energy analysis than the regular atomic fragment files from the create runs. This keyword cannot be used for molecular fragment files. In order to compare meta-GGA energy differences between molecular fragments and the total molecule, results from the various calculations need to be combined by hand.

In such situations, it is advisable to use a somewhat higher integration accuracy than one would normally do, at least for the smaller fragments, as there is no error cancellation as in a regular ADF bond energy analysis.

A general comment is that some functionals show a more stable behavior than others (at least in our current implementation). In general, the functionals which are dependent on the Laplacian of the density may display a large variation with respect to basis set changes or different numerical integration accuracy. For this reason we currently recommend FT97 in favor of FT98. Similarly, the results with the BmTau1 functional should still be carefully checked. In our test calculations on the G2 set of molecules, the VS98 showed best performance, both for the average error and for the maximum error. The G2 set consists only of small molecules with elements up to Cl. The relative performance for transition metals and heavy elements is unknown and may well be very different from the ordering for the G2 set.

Post Hartree-Fock energy functionals

This is mostly taken from text by the authors of Ref. [*381* (page 929)]: In the early days of DFT, non-self-consistent Kohn-Sham energy was often evaluated upon Hartree-Fock (HF) densities as a way to test new approximations. This method was called HF-DFT. It has been discovered that in some cases, HF-DFT actually gave more accurate answers when compared to self-consistent DFT calculations. In Ref. [*381* (page 929)], it was found that DFT calculations can be categorized into two different types of calculations. The error of an approximate functional can be decomposed into two parts: error from the functional (functional error), and error from the density (density-driven error). For most calculations, functional error is dominant, and here self-consistent DFT is usually better than non-self consistent DFT on more accurate densities (called density corrected DFT (DC-DFT)). Unlike these 'normal' calculations, there is a class of calculations where the density-driven error is much larger, so DC-DFT give better a result than self-consistent DFT. These calculations can be classified as 'abnormal'. HF-DFT is a simple implementation of DC-DFT and a small HOMO-LUMO gap is an indicator of an 'abnormal' calculation, thus, HF-DFT would perform better in such cases.

In ADF one can do HF-DFT with:

```
XC
  HartreeFock
END
MetaGGA
```

This will produce a large, pre-specified list of LDA, GGA, meta-GGA, hybrid, and metahybrid energy functionals.

4.3 Relativistic effects

The key RELATIVISTIC instructs ADF to take relativistic effects into account. By default (omission of the key) relativistic effects are **not** taken into account.

```
RELATIVISTIC {level} {formalism} {potential}
```

Level Scalar (default: scalar relativistic effects), or SpinOrbit (using double group symmetry).

Formalism Pauli (default), ZORA (ZORA is recommended!), X2C, or RA-X2C

Potential MAPA (default) or SAPA.

MAPA and SAPA

The SAPA method is described in Ref.[*50* (page 909)] for the BAND program. The same potential was used in the ADF program. However, starting from ADF 2017 instead of SAPA (the Sum of neutral Atomical potential Approximation) MAPA is used by default for ZORA. The MAPA (the Minumium of neutral Atomical potential Approximation) at a point is the minimum of the neutral Atomical potentials at that point. Advantage of MAPA over SAPA is that the gauge dependence of ZORA is reduced. The ZORA gauge dependency is small for almost all properties, except for the electron density very close to a heavy nucleus. The electron density very close to a heavy nucleus can be used for the interpretation of isomer shifts in Mössbauer spectroscopy

Recommendations:

```
Relativistic Scalar ZORA MAPA
```

or

```
Relativistic SpinOrbit ZORA MAPA
```

4.3.1 Pauli

Specification of the Pauli formalism means that the first order relativistic corrections (the Pauli Hamiltonian) will be used [*51-60* (page 909)]. In a *scalar* relativistic run ADF employs the single point group symmetry and only the so-called *scalar* relativistic corrections, Darwin and Mass-Velocity. The treatment is not strictly first-order, but is *quasi*-relativistic, in the sense that the first-order scalar relativistic Pauli Hamiltonian is diagonalized in the space of the non-relativistic solutions, i.e. in the non-relativistic basis set.

The quasi-relativistic approach improves results considerably over a first-order treatment. There are, however, theoretical deficiencies due to the singular behavior of the Pauli Hamiltonian at the nucleus. This would become manifest in a complete basis set but results are reasonable with the normally employed basis sets. However, this aspect implies that it is not recommended to apply this approach with an all-electron basis set for the heavy atoms, and for very heavy elements even a frozen core basis set often fails to give acceptable results. The problems with the quasi relativistic approach of the Pauli Hamiltonian are discussed for example in Ref.[*61* (page 910)].

4.3.2 ZORA

The ZORA approach gives generally better results than the Pauli formalism. For all-electron calculations, and in fact also for calculations on very heavy elements (Actinides), the Pauli method is absolutely unreliable. Therefore the ZORA method is the recommended approach for relativistic calculations with ADF.

ZORA refers to the Zero Order Regular Approximation [*61-65* (page 910)]. This formalism requires special basis sets, primarily to include much steeper core-like functions; applying the ZORA method with other, not-adapted basis sets, gives unreliable results. The ZORA basis sets can be found in subdirectories under the $ADFHOME/atomicdata/ZORA directory.

The ZORA formalism can also be used in Geometry Optimizations. However, there is a slight mismatch between the energy expression and the potential in the ZORA approach, which has the effect that the geometry where the gradients are zero does not exactly coincide with the point of lowest energy. The differences are very small, but not completely negligible, in the order of 0.0001 Angstrom.

4.3.3 X2C and RA-X2C

X2C stands for an exact transformation of the 4-component Dirac equation to 2-components [*447-448* (page 933)]. RA-X2C also stands for an exact transformation of the 4-component Dirac equation to 2-components, in this case using a regular approach to calculate the transformation matrix. In practice, however, often approximations are made, and this is also true for the implementation in ADF. In the X2C and RA-X2C method implemented in ADF, first the 4-component Dirac equation for a model potential (MAPA) of the molecule is calculated for the given ADF basis set, using the modified Dirac equation by Dyall [*430* (page 932)] for X2C, or using the regular approach [*445* (page 933)] to the modified Dirac equation for RA-X2C.

With the method used in ADF in the basis set limit X2C and RA-X2C should lead to the same results for the model potential (MAPA). However, since in practice always a finite basis set is used, even for the model potential the results for X2C and RA-X2C will differ. Next the (electronic) 4-component Dirac solutions are transformed exactly to 2-components [*446* (page 933)]. The transformation used is also used to calculate an effective one-electron 2-component kinetic energy operator in a basis set representation. This kinetic energy matrix is then assumed to be constant in further ADF SCF calculation on the molecule. The full Hamiltonian is approximated using the electron density in the 2-component picture, the so called Foldy-Wouthuysen picture.

In a similar way a spin-free (scalar relativistic) form of X2C and RA-X2C are calculated. The spin-free form of X2C and RA-X2C will be different even in the basis set limit, see Ref. [*431* (page 932)].

X2C and RA-X2C in ADF can be used in single point calculations only. An all electron basis set is needed. For bond energies, polarizabilities and TD-DFT excitation energies the Foldy-Wouthuysen picture will be used. For the calculation of the EFG, ESR g-value, and ESR A-tensor, the Dirac picure will be used. X2C and RA-X2C is not implemented for frozen cores, optimization of nuclear coordinates, frequencies, NMR properties.

4.3.4 Spin-Orbit coupling

The Spin-Orbit option uses double-group symmetry. The symmetry-adapted orbitals are labeled by the quantum number J rather than L and any references in input to subspecies, such as a specification of occupation numbers, must refer to the double group labels.

Create runs must *not* use the Spin-Orbit formalism. The SFO analysis of Molecular Orbitals for a Spin-Orbit calculation is only implemented in the case of a scalar relativistic fragment file, which is the whole molecule.

In a Spin-Orbit run each level can allocate 2 electrons (times the dimension of the irreducible representation) as in a normal restricted calculation. However, contrary to the normal case these two electrons are not directly associated with spin-α and spin-β, but rather with the more general Kramer's symmetry. Using the unrestricted feature in order to assign different numbers of electrons to a and b spin respectively cannot be applied as such. However, one can use the unrestricted option in combination with the collinear or non-collinear approximation. In that case one should use symmetry NOSYM, and each level can allocate 1 electron.

4.4 Solvents and other environments

4.4.1 COSMO: Conductor like Screening Model

You can study chemistry in solution, as contrasted to the gas phase, with the implementation in ADF [*66* (page 910)] of the Conductor like Screening Model (COSMO) of solvation [*67-69* (page 910)]. The energy derivatives can also be calculated, so geometry optimization, harmonic frequencies, et cetera are available within this model.

The COSMO model is a dielectric model in which the solute molecule is embedded in a molecule-shaped cavity surrounded by a dielectric medium with given dielectric constant ϵ. Energy-related terms are computed for a conductor

first, then scaled by the function

$$f(\epsilon) = \frac{\epsilon - 1}{\epsilon + x} \qquad (2.1.1)$$

The empirical scaling factor *x* is specified in the input data block for the SOLVATION key. The block key SOLVATION turns the solvation calculation on. In most cases default values are available for the involved parameters. In order to construct the cavity atomic cavity radii are used. The default COSMO atomic cavity radii used in ADF are the Van der Waals radii from the MM3 method by Allinger (Ref. [*290* (page 924)]) divided by 1.2.

It is also possible to include a linear parametrization of non-electrostatic terms as a function of surface area. To include such term can be specified in the input data block for the SOLVATION key. Starting from ADF2012 the default is to include only the part of this term that is proportional to the surface area (default CAV0=0.0, CAV1=0.0067639):

$$E_{\text{non-elst}} = f(\epsilon)(CAV0 + CAV1 \times \text{area}) \qquad (2.1.2)$$

If a calculations was done on a fragment, then the wave function obtained would be optimal for the fragment in solution, but not optimal for gas phase. The energies with the gas phase Hamiltonian would be higher, and the apparent solvation contribution to bonding would also be higher. The net point is that normally the COSMO procedure reports the energy of Esolv(AB), but to get the solvation energy, you need to subtract the E(AB, solv) from E(AB,gas) because the wave function changes (unless you are doing it post-SCF.) For this you need to have the same reference fragments in each case A(g) and B(g).

Because of precision issues in ADF2016 the default surface has been changed to Delley, instead of Esurf.

```
SOLVATION
 {SURF Surface {NOKEEP}}
 {SOLV {Name=solvent} {Eps=78.4} {Del=1.4} {Rad=1.4}
       {Neql=1.9}{Emp=0.0}{Cav0=0.0}{Cav1=0.0067639} }
 {DIV {Ndiv=3} {NFdiv=1} {Min=0.5} {OFAC=0.8}
      {leb1=23} {leb2=29}{rleb=1.5} }
 {NOASS}
 {RADII
   name1=value1
   name2=value2
   ...
 subend }
 {CHARGED {Method=meth} {Conv=1e-8} {Omega=1.0} {Iter=1000}
          {Corr} {LoCosmo LoCosmoDist}}
 {C-MAT How {SCF} tol=1e-10 }
 {DISC {SC=0.01} {LEG=4} {TOL=0.1} }
 {SCF {When} {How}
 {NOCSMRSP}
 {LPRT}
End
```

Presence of the SOLVATION key block triggers the solvent calculation and does not require additional data. With subkeys you can customize various aspects of the model, for instance to specify the type of solute. None of the subkeys is obligatory. Follows a description of the subkeys

SURF Surface {NOKEEP} Defines the type of cavity to be used. `Surface` can be one of the following: Wsurf, Asurf, Esurf, Klamt, or Delley. **Default: Delley**. The Wsurf, Asurf, and Esurf surfaces are constructed with the GEPOL93 algorithm [*70* (page 910)]

   Wsurf Wsurf triggers the Van der Waals surface (VdW), which consists of the union of all atomic spheres. Not recommended to be used.

   Asurf Asurf gives the Solvent-Accessible-Surface (SAS). This is similar to VdW but consists of the path traced by the center of a spherical solvent molecule rolling about the VdW surface or, equivalently, a VdW

surface created by atomic spheres to which the solvent radius has been added. These two surface types contain cusps at the intersection of spheres. Not recommended to be used.

Esurf Esurf gives the Solvent-Excluding-Surface (SES), which consists of the path traced by the *surface* of a spherical solvent molecule rolling about the VdW surface. Primarily, this consists of the VdW surface but in the regions where the spheres would intersect, the concave part of the solvent sphere replaces the cusp.

Klamt The fourth surface option is Klamt as described in [*67* (page 910)]. It excludes the cusp regions also. Note that this surface might give an incomplete COSMO surface in case of more complicated molecules. Not recommended to be used.

Delley The fifth surface (default surface) is the so called Delley surface, see also Ref. [*245* (page 921)]. This Delley type of cavity construction is recommended to be used in COSMO calculations, which results are used as input for COSMO-RS calculation, see the corresponding manual for COSMO-RS. This Delley surface is the default COSMO surface in ADF.

NOKEEP The optional parameter NOKEEP controls surface creation during calculation of frequencies by numerical differentiation. By default, the surface is constructed only once at the central geometry and is used for the rest of the calculation. If the NOKEEP is specified then ADF will construct a new surface at each displaced geometry.

DIV The actual construction of the surface involves a few technical parameters controlled with the subkey DIV

Ndiv, NFdiv Ndiv controls how fine the spheres that in fact describe the surface are partitioned in small surface triangles, each containing one point charge to represent the polarization of the cavity surface. Default division level for triangles Ndiv=3. Default final division level for triangles NFdiv=1 (NFdiv $\leq$ Ndiv). Not used in the Delley surface.

Min Min specifies the size, in angstrom, of the smallest sphere that may be constructed by the SES surface. For VdW and SAS surfaces it has no meaning. Default Min=0.5. Not used in the Delley surface.

Ofac Ofac is a maximum allowed overlap of new created spheres, in the construction procedure. Default Ofac=0.8. Not used in the Delley surface.

leb1, leb2, rleb Only used in the Delley surface. For the Delley type of construction one needs to set the variables leb1 (default value 23), leb2 (default value 29), and rleb (default value 1.5 Angstrom) to set the number of surface points. If the cavity radius of an atom is lower than rleb use leb1, otherwise use leb2. These values can be changed: using a higher value for leb1 and leb2 gives more surface points (maximal value leb1, leb2 is 29). A value of 23 means 194 surface points in case of a single atom, and 29 means 302 surface points in case of a single atom Typically one could use leb1 for the surface point of H, and leb2 for the surface points of other elements.

NOASS By default all new spheres that are created in the surface-construction are assigned to atoms, for the purpose of gradient computations (geometry optimization). Specifying the noass subkey turns this off. It has no argument.

SOLV Solvent details.

Eps, Rad Eps specifies the dielectric constant (the default relates to water). In ADF an infinite value for Eps is chosen if Eps is specified to be lower than 1.0. Rad specifies the radius of the (rigid sphere) solvent molecules, in angstrom. Instead of specifying Eps and Rad one can specify a solvent name or formula after 'name='. The following table lists names and formulas that are recognized with the corresponding values for Eps and Rad. The Rad in this table is calculated from the density, the molar mass, and a spherical approximation for the solvent. The names and formulas are case-insensitive.

Name	Formula	Eps	Rad
AceticAcid	CH3COOH	6.19	2.83
Acetone	CH3COCH3	20.7	3.08
Acetonitrile	CH3CN	37.5	2.76
Ammonia	NH3	16.9	2.24
Continued on next page			

Table 4.1 – continued from previous page

Aniline	C6H5NH2	6.8	3.31
Benzene	C6H6	2.3	3.28
BenzylAlcohol	C6H5CH2OH	13.1	3.45
Bromoform	CHBr3	4.3	3.26
Butanol	C4H9OH	17.5	3.31
isoButanol	(CH3)2CHCH2OH	17.9	3.33
tertButanol	(CH3)3COH	12.4	3.35
CarbonDisulfide	CS2	2.6	2.88
CarbonTetrachloride	CCl4	2.2	3.37
Chloroform	CHCl3	4.8	3.17
Cyclohexane	C6H12	2	3.5
Cyclohexanone	C6H10O	15	3.46
Dichlorobenzene	C6H4Cl2	9.8	3.54
DiethylEther	(CH3CH2)2O	4.34	3.46
Dioxane	C4H8O2	2.2	3.24
DMFA	(CH3)2NCHO	37	3.13
DMSO	(CH3)2SO	46.7	3.04
Ethanol	CH3CH2OH	24.55	2.85
EthylAcetate	CH3COOCH2CH3	6.02	3.39
Dichloroethane	ClCH2CH2Cl	10.66	3.15
EthyleneGlycol	HOCH2CH2OH	37.7	2.81
Formamide	HCONH2	109.5	2.51
FormicAcid	HCOOH	58.5	2.47
Glycerol	C3H8O3	42.5	3.07
HexamethylPhosphoramide	C6H18N3OP	43.3	4.1
Hexane	C6H14	1.88	3.74
Hydrazine	N2H4	51.7	2.33
Methanol	CH3OH	32.6	2.53
MethylEthylKetone	CH3CH2COCH3	18.5	3.3
Dichloromethane	CH2Cl2	8.9	2.94
Methylformamide	HCONHCH3	182.4	2.86
Methypyrrolidinone	C5H9NO	33	3.36
Nitrobenzene	C6H5NO2	34.8	3.44
Nitrogen	N2	1.45	2.36
Nitromethane	CH3NO2	35.87	2.77
PhosphorylChloride	POCl3	13.9	3.33
IsoPropanol	(CH3)2CHOH	19.9	3.12
Pyridine	C5H5N	12.4	3.18
Sulfolane	C4H8SO2	43.3	3.35
Tetrahydrofuran	C4H8O	7.58	3.18
Toluene	C6H5CH3	2.38	3.48
Triethylamine	(CH3CH2)3N	2.44	3.81
TrifluoroaceticAcid	CF3COOH	8.55	3.12
Water	H2O	78.39	1.93

Del Del is the value of Klamt's delta_sol parameter, only relevant in case of Klamt surface.

Neql If Neql = ϵ_{NEQL} is included a non-equilibrium solvation is used, i.e. that the dielectric constant ϵ_{NEQL} used in RESPONSE is different from the ground state dielectric constant ϵ. Only relevant in case of TDDFT calculations. Default $\epsilon_{\mathrm{NEQL}} = \epsilon$. The reason for using two different dielectric constants is that the electronic transition can so fast that only the electronic component of the solvent dielectric can respond, i.e., one should use the optical part of the dielectric constant. This is typically referred to as non-equilibrium

solvation. The optical dielectric constant can be obtaining from the (frequency dependent) refractive index n of the solvent as: $\epsilon_{\text{NEQL}} = n^2$.

Emp Emp addresses the empirical scaling factor x in the formula 2.1.1 above.

Cav0, Cav1 Other options specify a linear parametrization of non-electrostatic terms as a function of surface area, see the formula 2.1.2 above. Possible values for CAV0 and CAV1 are CAV0 = 1.321 and CAV1 = 0.0067639, see Ref. [*299* (page 924)]). In ADF2010 the default values for CAV0 and CAV1 are CAV0 = 0.0 and CAV1 = 0.0. However, starting from ADF2012 the default values for CAV0 and CAV1 are CAV0 = 0.0 and CAV1 = 0.0067639, If CAV0 is not zero, Esolv(AB) is not the same as Esolv(A) + Esolv(B) if A and B are far apart. This is the reason why CAV0 is set to zero, by default. By default CAV1 is not set to zero, thus by default there is a solvation energy term that does depend on the size of the cavity (surface area).

COSMO Radii In order to construct the surface you have to specify the atomic ('Van der Waals') radii. There are three ways of doing this. In the first method you append 'R=value' to the atomic coordinates record, in the ATOMS key block. This would look like, for instance:

```
C 1 2 3 CC CCO CCOH f=C.dz R=2.0
```

It assigns a radius of 2.0 to the Carbon atom. In the second method you apply the same format, but specify a symbol (identifier) rather than a value:

```
C 1 2 3 CC CCO CCOH f=C.dz R=C-sp3
```

The identifiers must be defined in the (optional) RADII subkey block in the Solvation data block (see next). In the third method, you don't modify the Atoms block at all. In this case, the RADII subkey must be used and the 'identifiers' in it must be exactly the atom type names in the Atoms block.

RADII This subkey is block type. Its data block (if the subkey is used) must terminate with a record subend. In the Radii data block you give a list of identifiers and values:

```
SOLVATION
   ...
   Radii
      name1=value1
      name2=value2
      ...
   Subend
   ...
End
```

The values are the radii of the atomic spheres, in the same units of length as used in the Atoms block (angstrom or bohr). The names specify to which atoms these values apply. As discussed for the Solv subkey this depends on the Atoms block. If in the specification of atomic coordinates you have used the 'R=' construct to assign radii, with identifiers rather than values for the R-value, these identifiers must be defined in the Radii sub block. If no 'R=' construct was applied in the Atoms block, you must use the atom type names as they occurred in the Atoms data block. Referring to the example given in the Solv subkey discussion, you might have:

```
Radii
   C-sp3=2.0
   ...
Subend
```

A simple atom type reference might look like:

```
Radii
   C=2.0
```

```
    ...
Subend
```

When no radius specified a default value is used. The default value for an atom is the corresponding Van der Waals radius from the MM3 method by Allinger (Ref. [*290* (page 924)]) divided by 1.2.

Another popular choice are the so called Klamt atomic cavity radii (Ref. [*485* (page 935)]), which are optimized for a few elements:

```
Radii
    H   1.30
    C   2.00
    N   1.83
    O   1.72
    F   1.72
    Si  2.48
    P   2.13
    S   2.16
    Cl  2.05
    Br  2.16
    I   2.32
Subend
```

CHARGED {Method=meth} {Conv=1e-8} {Omega=1.0} {Iter=1000} {Corr} {LoCosmo LoCosmoDist}
This addresses the determination of the (point) charges that model the cavity surface polarization. In COSMO calculations you compute the surface point charges q by solving the equation A*q*=-f, where f is the molecular potential at the location of the surface charges q and A is the self-interaction matrix of the charges. The number of charges can be substantial and the matrix A hence very large. A direct method, i.e. inversion of A, may be very cumbersome or even impossible due to memory limitations, in which case you have to resort to an iterative method. Meth specifies the equation-solving algorithm. Meth=INVER requests direct inversion. Meth=GAUS calls for the Gauss-Seidel iterative method. Meth=Jacobi activates another standard iterative procedure. The latter two methods require a positive-definite matrix (which may fail to be the case in an actual calculation) and can be used with a relaxation technique, controlled by the relaxation parameter OMEGA (1.0=no relaxation). Meth=CONJ (default) uses the preconditioned biconjugate gradient method. This is guaranteed to converge and does not require huge amounts of memory. CONV and ITER are the convergence criterion and the maximum number of iterations for the iterative methods. Some of the molecular electronic charge distribution may be located outside the cavity. This affects the assumptions underlying the COSMO equations. LOCOSMO can only be used icw FDE.

Specifying the CORR option to the CHARGED subkey constrains the computed solvent surface charges to add up to the negative of the molecular charge. In ADF2016 this outlying charge correction is also done during the SCF. Note that before ADF2016 it was only a post-SCF correction on the COSMO charges. In ADF2018 a bug that was introduced in ADF2016 in case of this outlying charge correction was fixed for the energy (typically a small effect).

C-MATRIX How: For the potential f we need the Coulomb interaction between the charges q and the molecular electronic density (and nuclei). Three methods are available, specified by the first option to the C-Matrix subkey.

1. EXACT: compute the straightforward Coulomb potential due to the charge q in each point of the molecular numerical integration grid and integrate against the electronic charge density. This is, in principle, exact but may have inaccuracies when the numerical integration points are very close to the positions of a charge q. To remedy this, the point charges q can be 'smeared out' and represented by a disc, see the next subkey DISC.

2. FIT: same as EXACT, but the q-potentials are now integrated not against the exact electronic charge density, but against the (much cheaper-to-compute) fitted density. The same DISC considerations apply.

3. POT: evaluate the molecular potential at the position of the charge q and multiply against the q-strength.

Since the molecular Coulomb potential is computed from the fit density, any difference in results between the FIT and the POT approach should be attributed to the DISC issue. POT is the default, because it is faster, and is only inadequate if the fit density is very inaccurate, which would be a problem anyway.

SCF: If you specify this option, the computation of the Coulomb interaction matrix (between electrons and surface charges) is carried out during the SCF procedure, but this turns out to hamper the SCF convergence behavior. Therefore: not recommended. *IF* you use it, the program will switch to one of the other 3 methods, as given by the 'How' option, as soon as the SCF convergence error drops below TOL: (applies only to the SCF option, which is not recommended).

DISC Applies only when the C-matrix method is EXACT or FIT. Note, however, that the default for the C-matrix method is POT, in which case the DISC subkey has no meaning. The DISC key lets the program replace the point charges q by a solid uniformly charged spherical surface disc whenever the numerical integration accuracy requires so, i.e. for those charges that are close to numerical integration points. Options: SC defines a shrinking factor, by which the actual disc radius used is reduced from its 'normal' value: an inscribed disc in the triangular surface partitions that define the distribution of surface charges, see the subkey DIV. LEG gives the polynomial expansion order of the disc potentials. The Legendre expansion converges rapidly and the default should be adequate.

TOL is a tolerance parameter to control the accuracy of the disc potential evaluations.

SCF In COSMO calculations you can include the surface charges in the Fock operator self-consistently, i.e. by re-computing the charges q at every SCF cycle and include them in the equations, or in a perturbational manner, i.e. post-SCF. This is controlled with the first option. The When option must be either VAR or PERT, for variational and perturbational, respectively. Default is VAR. The second (HOW) option applies only to the WHEN=VAR case and may affect the speed of SCF convergence. The COSMO calculation implies a considerable increase in CPU time! Values for HOW:

- ALL: This includes it in all SCF cycles (except for the first SCF cycle, which is gas-phase)

- LAST: This lets the program first converge the SCF completely without any solvent effects. Thereafter, the COSMO is turned on, hopefully converging in fewer cycles now, to compensate for the 'double' SCF effort.

- TOL=0.1 (or another value) is an in-between approach: converge the gas-phase SCF until the SCF error is below TOL, then turn on COSMO.

NOCSMRSP Relevant only in combination with the time-dependent DFT (TDDFT) applications: the EXCITATION, the RESPONSE, or the AORESPONSE key. If this subkey NOCSMRSP is included the induced electronic charges which are present in the TDDFT calculations, will not influence the COSMO surface charges. No dielectric constant in the response might be closer to the optical dielectric constant than using the full dielectric constant, see also sub-argument NEQL of the subkey SOLV of the key SOLVATION. By default, in absence of this subkey NOCSMRSP, the induced electronic will influence the COSMO surface charges. If one does geometry optimization of the excited state this makes sense, since then the solvent dielectric has time to fully respond.

LPRT This is a debug switch and triggers a lot more output related to the cavity construction etc.

Warning

Numerical frequencies calculated with COSMO should be checked for stability with respect to the *disrad*, the numerical differentiation step size. The problem is that the COSMO surface changes slightly when a nucleus is moved from its equilibrium position. The change is usually small but in some cases it may result is creation or annihilation of surface points, which will lead to discontinuities in the potential energy surface and may result in inaccurate frequencies.

Thus, when calculating vibrational frequencies numerically with COSMO, one should try decreasing the *disrad* value until no changes in frequencies are observed. However, the value should not be too small because then the total numerical noise may become too large compared to the generated forces. A general recommendation would be to try to decrease disrad by a factor of 2 at a time. Of course, this procedure may be very expensive for a large molecule. If

this the case, one should use the SCANFREQ keyword and recalculate only a small number of frequencies. It should be noted that generally frequencies that have a small force constant are more sensitive to the numerical noise.

4.4.2 SM12: Solvation Model 12

Continuum solvation can be done with the Minnesota's Solvation Model 12 (SM12) [*399* (page 930)]. Details on the implementation of SM12 in ADF can be found in Ref. [*439* (page 932)]. The energetics of solvation is calculated using:

$$\Delta G_S^{\otimes} = \Delta E_E + G_P + G_{CDS} + \Delta G_N + \Delta G_{conc}^{\otimes}$$

where the symbol $\otimes$ denotes an arbitrary choice of standard states, ΔE_E is the change in the solute's internal electronic energy in transferring from the gas phase to the liquid phase at the same geometry, G_P is the polarization free energy of the solute–solvent system when the solute is inserted, G_{CDS} is the component of the free energy that is nominally associated with cavitation, dispersion, and solvent structure, ΔG_N is the change in ΔG_S^0 due to a change in nuclear coordinates, and $\Delta G_{conc}^{\otimes}$ accounts for the difference in concentrations, if any, in the gas-phase standard state and the solution-phase one. In case of 1 M concentration in both solution and gas, then $\Delta G_S^{\otimes}$ = 0 kcal/mol, which yields ΔG_S^*. If the same geometry is used in solution and gas phase calculation, then ΔG_N is zero.

SM12 makes use of the Generalized Born approximation to calculate the bulk electrostatic contribution. This is comprised of several terms that are together known as the ENP (Electronic, Nuclear, and Polarization) term G_P. The SM12 model in ADF uses *CM5 charges* (page 267) [*378* (page 929)]. CM5 is a class 4 charge model, making use of both empirical and density related terms. It is comprised of Hirshfeld charges [*125* (page 914)], a simple bond order calculation, atomic distances, and atom specific parameters [*378* (page 929)]. The covalent radii utilized are based on the atomic covalent radius from the Handbook of Chemistry and Physics [*402* (page 930)]. The Coulomb integral is described with the use of an approximation from Still et al. [*400* (page 930)]. Several parameters go into describing this, which include: inter atomic distance, an empirical Born constant, and the Born area, which is calculated with the Analytical Surface Area (ASA) algorithm [*401* (page 930)]. The Born area is calculated using Legendre-Gauss quadrature from the atomic radii to a sphere that encapsulates the entire molecule.

The ASA algorithm is also used to calculate the solvent accessible surface area (SASA), which is computed within the CDS (Cavitation, Dispersion, Solvation) term of SM12. The CDS term depends on three terms:

- SASA (ASA Algorithm)

- Atomic surface tension

- Macroscopic surface tension

Atomic surface tension is based on atom-atom distances and the solvent. Macroscopic surface tension is solvent specific. The SM12 implementation in ADF reports energies in an atom specific way. You can attribute exact CDS and polarization energies to each atom in your solute. The parameters for SM12 are derived to explicitly incorporate organic elements (N, C, O, F, Si, P, S, Cl, Br, I), with less emphasis on non-organics. Also, while most solvents have a generic atomic surface tension reliance for atoms, water has its own explicit set of parameters to better describe it.

```
SOLVATION SM12
  {SOLV NAME=solvent}
  {CUST {NAME=solvent}}
  {REF=1.33}
  {ACID=0.82}
  {BASE=0.35}
  {TENS=103.62}
  {EPS=78.36}
  {ARO=0.0}
  {HALO=0.0}
  {RADSOLV=0.4}
  {CHGAL=2.474}
```

```
  {BORNC=3.70}
  {PRINTSM12}
End
```

Presence of the SOLVATION SM12 key block key triggers the solvent calculation and does not require additional data. If no other keywords are present the default is water. With subkeys you can customize various aspects of the model, for instance to specify the type of solute. None of the subkeys is obligatory. Follows a description of the subkeys

SOLV NAME = solvent Defines which solvent to be used. Each solvent has parameters set for each of the variables defined below. All of which can be found in the SM12 Ref. [*399* (page 930)]. Any variable can be changed with the inclusion of its key. `solvent` can be one of the following:

- ACETICACID
- ACETONITRILE
- ACETOPHENONE
- ANILINE
- ANISOLE
- BENZENE
- BENZONITRILE
- BENZYLALCOHOL
- BROMOBENZENE
- BROMOETHANE
- BROMOFORM
- BROMOOCTANE
- N-BUTANOL
- SEC-BUTANOL
- BUTANONE
- BUTYLACETATE
- N-BUTYLBENZENE
- SEC-BUTYLBENZENE
- T-BUTYLBENZENE
- CARBONDISULFIDE
- CARBONTETRACHLORIDE
- CHLOROBENZENE
- CHLOROFORM
- CHLOROHEXANE
- M-CRESOL
- CYCLOHEXANE
- CYCLOHEXANONE
- DECALIN
- DECANE
- DECANOL
- 1-2-DIBROMOETHANE
- DIBUTYLETHER
- O-DICHLOROBENZENE
- 1-2-DICHLOROETHANE
- DIETHYLETHER
- DIISOPROPYLETHER
- N-N-DIMETHYLACETAMIDE
- N-N-DIMETHYLFORMAMIDE
- 2-6-DIMETHYLPYRIDINE
- DIMETHYLSULFOXIDE
- DODECANE
- ETHANOL
- ETHOXYBENZENE

- ETHYLACETATE
- ETHYLBENZENE
- FLUOROBENZENE
- 1-FLUORO-N-OCTANE
- HEPTANE
- HEPTANOL
- HEXADECANE
- HEXADECYLIODIDE
- HEXANE
- HEXANOL
- IODOBENZENE
- ISOBUTANOL
- ISOOCTANE
- ISOPROPANOL
- ISOPROPYLBENZENE
- P-ISOPROPYLTOLUENE
- MESITYLENE
- METHANOL
- METHOXYETHANOL
- METHYLENECHLORIDE
- N-METHYLFORMAMIDE
- 2-METHYLPYRIDINE
- 4-METHYL-2-PENTANONE
- NITROBENZENE
- NITROETHANE
- NITROMETHANE
- O-NITROTOLUENE
- NONANE
- NONANOL
- OCTANE
- OCTANOL
- PENTADECANE
- PENTANE
- PENTANOL
- PERFLUOROBENZENE
- PHENYLETHER
- PROPANOL
- PYRIDINE
- TETRACHLOROETHENE
- TETRAHYDROFURAN
- TETRAHYDROTHIOPHENEDIOXIDE
- TETRALIN
- TOLUENE
- TRIBUTYLPHOSPHATE
- TRIETHYLAMINE
- 1-2-4-TRIMETHYLBENZENE
- UNDECANE
- WATER
- XYLENE
- 1-2-DIBROMOETHANE_WATER
- 1-2-DICHLOROETHANE_WATER
- BENZENE_WATER
- CARBONTETRACHLORIDE_WATER
- CHLOROBENZENE_WATER

- CHLOROFORM_WATER
- CYCLOHEXANE_WATER
- DIBUTYLETHER_WATER
- DIETHYLETHER_WATER
- ETHYLACETATE_WATER
- HEPTANE_WATER
- HEXANE_WATER
- NITROBENZENE_WATER
- OCTANOL_WATER

CUST {NAME=solvent} A solvent with user specified parameters can be used. A user specified name of the solvent `solvent` can be provided, but should not match one of the solvent names listed above. Any unspecified parameters will be set to zero, with the exception of the dielectric constant, which will be set to the dielectric constant of water.

REF Index of refraction at 293K (n).

ACID Abrahams hydrogen bond acidity (α).

BASE Abrahams hydrogen bond basicity (β).

TENS Macroscopic surface tension of the solvent at the air interface cal mol^{-1} Å^{-2})(γ).

EPS Dielectric constant. If not specified it will be set to the default of water. (ϵ).

ARO Square root of the fraction of non-hydrogen atoms in the solvent molecule that are aromatic carbon atoms (carbon aromaticity) (ϕ).

HALO Square of the fraction of non-hydrogen atoms in the solvent molecule that are halogens (Electronegative halogenicity)(ψ).

RADSOLV Solvent radius, this is added to the van der Waals radius used for the cavity calculation.

BORNC Empirical Born constant, see Ref. [*399* (page 930)]

PRINTSM12 Print more information on the SM12 calculation: CDS atom terms, ENP charges.

4.4.3 QM/MM: Quantum mechanical and Molecular Mechanics model

ADF supports the QM/MM method to handle large systems or environment effects by treating only part of the atoms quantum-mechanically and the other ones by molecular mechanics. Use of this feature is invoked by the QM/MM keyword (block type). The functionality and all details of the keyword, involving quite a few options and aspects, are described in the separate QM/MM manual.

See also the *pdb2adf* utility, described in detail in the ADF QM/MM document, which transforms a PDB file into an ADF input file, for use with QM/MM.

4.4.4 Quild

The QUILD (Quantum-regions Interconnected by Local Description) program has been developed for enabling calculations through multi-level approaches, in which different computational treatments are used for different regions of the system under study, see the separate Quild manual.

4.4.5 DIM/QM: Discrete Interaction Model/Quantum Mechanics

The Discrete Interaction Model/Quantum Mechanics (DIM/QM) method facilitates calculating the optical properties of molecules under the influence of a discrete solvent or a metal nanoparticle, see for example Ref. [*362* (page 928)].

DIM/QM relies on one of three descriptions of the system: Discrete Reaction Field (DRF), where the atoms interact via induced dipoles and static charge, Capacitance Polarizability Interaction Model (CPIM), where the atoms interact via induced dipoles and induced charges, and Polarizability Interaction Model (PIM), where the atoms interact via induced dipoles only. DRF is best for solvents, CPIM is best for small metal nanoparticles, and PIM is best for large metal nanoparticles.

The DRF module is now a part of DIM/QM, so DRF is now a submodule of DIM/QM.

To perform a DIM/QM calculation, two block keys are required. The first is the DIMQM block which activates the DIM/QM module. The parameters for each DIM atom must also be given, and they can be given with either the DIMPAR block which is most convenient for metal nanoparticles, or with the EXTERNALS block which is designed for DRF.

In ADF2019 the bond energies calculated with DIM/QM in ADF have been corrected.

DRF

The Discrete Solvent Reaction Field (DRF) model is a hybrid Quantum mechanical and Molecular Mechanics (QM/MM) model for studying solvation effects on (time-dependent) molecular properties such as dipole moments, excitation energies and (hyper)polarizabilities[*141-145* (page 915)]. The classical solvent molecules are represented using distributed atomic charges and polarizabilities.

With the ADF-GUI it is simple to setup a DRF calculation. Another possibility is to use and/or modify a python script that is explained in the example *DIMQM_DRF_plams* (page 503). Both the ADF-GUI and the DRF python script have default settings which can be easily modified if this is required. In the simplest case the user just needs to provide the QM and DRF regions. By default, atomic charges that are used, in case of the ADF-GUI or the DRF python script, for the DRF are computed with LDA functional, DZP basis set and normal numerical quality. Also by default, atomic polarizabilities (computed with the Thole's model) are taken from a inner database including H, C, N, O, F, S, Cl, Br, I atoms [*146* (page 915)]. If one does not use the ADF-GUI or the DRF python script, the user has to specify this by hand.

Within the Discrete Solvent Reaction Field model the QM/MM operator is

$$H_{\text{QM/MM}} = \sum_i v_{\text{DRF}}(r_i, \omega) = \sum_i V^{\text{el}}(r_i) + V^{\text{pol}}(r_i, \omega)$$

where the first term, v^{el}, is the electrostatic operator and describes the Coulombic interaction between the QM system and the permanent charge distribution of the solvent molecules. The second term, v^{pol}, is the polarization operator and describes the many-body polarization of the solvent molecules, i.e. the change in the charge distribution of the solvent molecules due to interaction with the QM part and other solvent molecules. The charge distribution of the solvent is represented by atomic point charges and the many-body polarization by induced atomic dipoles at the solvent molecules. The induced atomic dipole at site s is found by solving a set of linear equations

$$\mu_{s,\alpha}^{\text{ind}}(\omega) = \alpha_{s,\alpha\beta} \left(F_{s,\beta}^{\text{init}}(\omega) + \sum_{t \neq s} T_{st,\beta\gamma}^{(2)} \mu_{t,\gamma}^{\text{ind}}(\omega) \right)$$

where $\alpha_{s,\alpha\beta}$ is a component of the atomic polarizability tensor at site s. The screened dipole interaction tensor is given by

$$T_{st,\beta\gamma}^{(2)} = 3 f_{st}^T R_{st,\alpha} R_{st,\beta} / R_{st}^5 - f_{st}^E \delta_{\alpha\beta} / R_{st}^3$$

where the damping functions $f^T{}_{st}$ and $f^E{}_{st}$ have been introduced, see also [*146* (page 915)]. A smeared-out point charge model [*147* (page 915)] is used for short-range damping of the QM/MM operator

$$1/R_{st} \rightarrow 1/S_{st} = erf(R_{st})/R_{st}$$

The scaled distance, S_{st}, then replaces the normal distance, R_{st}, in the QM/MM operator.

In order to perform a DRF calculation two types of parameters (model atomic charges and atomic polarizabilities) for each type of atom in the MM part are required. The point charges should represent at least the permanent molecular dipole moment, and the distributed atomic polarizabilities the full molecular polarizability tensor. The atomic charges can straightforward be obtained using e.g. Multipole Derived Charges (MDC) [See *section on MDC* (page 267)] and the distributed polarizabilities by adopting standard parameters or refitting them to match the calculated polarizability tensor [*146,147* (page 915)]. This allows for a simple procedure to obtain the solvent model parameters which subsequently can be used in the DRF calculation.

DRF gradients can be calculated by using the DRF2 keyword in the DIMQM block. This is an expert option and should be used with great caution. The DRF2 keyword uses DRF but with Gaussian screening like that of the PIM method. It also allows for input of parameters by atom type like the PIM method. The user must specify a radius and charge for the atom type like seen below.

```
DIMPAR
  O
    rad=0.9028
    char=-0.6690
    STATIC
  SUBEND
  H
    rad=0.2479
    char=0.3345
    STATIC
  SUBEND
  Ag
    rad=1.445
    char=0.0000
  SUBEND
END
```

Here, the STATIC keyword is used to tell DIM/QM to ignore the frequency dependent response for those atom types.

Surface-enhanced response properties

In addition to the calculation of solvation effects on response properties, DIM/QM can also be used for simulating nonlinear optical properties of molecules near metal surfaces of a nanoparticle. A TDDFT description of the molecule is coupled to an electrodynamical treatment of the nanoparticle, including electrostatics and polarization of the atoms. In this way DIM/QM can be used to study surface-enhance Raman scattering (SERS) [*362* (page 928)], surface-enhanced Raman optical activity (SEROA) [*382* (page 929)], and plasmon-enhanced photochemistry.

Starting from ADF2018 the Polarizability Interaction Model (PIM) can be used to calculate damped first hyperpolarizabilities β and damped second hyperpolarizabilities γ. Different types of β-tensors can be calculated, such as STATIC, OPTICALR, EOPE, and SHG. This allows for simulation of surface-enhanced hyper-Raman scattering (SEHRS) [*487* (page 935)]. All the (sub)keywords for regular damped β, such as BETA and QUADRATIC, can also be used. Also different types of γ-tensors can be calculated, such as STATIC, EFIOR, OKE, IDRI, EFISHG, THG and TPA. This allows for simulation of plasmon-enhanced two-photon absorption (PETPA). Except CUBIC, all the other (sub)keywords for regular damped γ can also be used.

Input options

```
DIMQM
  <DRF|PIM|CPIM>
  NOPOL
  NOCHAR
  NOCROSS
```

```
   DDA
   FULLGRID
   LOCALFIELD <Resp|All|Static>
   EFIELD x y z
   SCREEN <ERF|EXP|ESP|GAU|NONE> {length}
   :: FREQUENCY-DEPENDENT PARAMETERS
   FREQUENCY
   OM_H value
   OM_C value
   OM_O value
   OM_N value
   :: CONTROL OVER SOLVER
   ALGORITHM <BEST|DIRECT|BRUTE|SINGLE|MULTI>
   TOLERANCE tol
   NITER iterations
   VOLUME vol_in_nm^3
   MULTIPLIER aterm bterm cterm
   GRID <COARSE|MEDUIM|FINE>
   :: GRADIENT OPTIONS
   FORCEFIELD
   CHEMBOND qmindex dimindex
   COORDDEPEND
   CHEMISORPTION
   COORDPAR atomtype e0 e1 r0 r1 CNmax Rmax Rmin
   CHEMPAR atomtype e0 e1 r0 r1 cutoff
   PROJECTIONMATRIXPOINTS <ALL|CUTOFF radius|OFF>
   :: OUTPUT CONTROL
   PRINTATOMICDIPOLES
   PRINTLJPAR
   DEBUG
   LOCALDIM
   LOCALDIRECT
   DIMBUFFER
   QMBUFFER
   CUTOFFDIST
END
```

<DRF|CPIM|PIM> DIM/QM relies on one of three descriptions of the system: Discrete Reaction Field (DRF), where the atoms interact via induced dipoles and static charge, Capacitance Polarizability Interaction Model (CPIM), where the atoms interact via induced dipoles and induced charges, and Polarizability Interaction Model (PIM), where the atoms interact via induced dipoles only. DRF is best for solvents, CPIM is best for small metal nanoparticles, and PIM is best for large metal nanoparticles. One and only one of these three keys must be included in every DIM/QM calculation. However, in ADF2019 there is also the expert DRF2 possibility, which allows for DRF gradients but should be used cautiously.

NOPOL The NOPOL key turns off the polarization terms, and thus all induced dipoles are zero. This key is only valid for DRF or CPIM calculations.

NOCHAR The NOCHAR key turns off the charge terms, and thus all induced or static charges are zero. This key is only valid for DRF or CPIM calculations.

NOCHOSS The NOCROSS key turns off the charge-dipole interactions. This key is only valid for CPIM calculations.

DDA By default, the dipole-dipole, charge-dipole and charge-charge interactions are screened to take into account that atoms are not point charges. The DDA key will turn off this screening so that the results can be compared directly to the discrete dipole approximation (DDA).

FULLGRID This is used in conjunction with the frozen density approximation.

LOCALFIELD <Resp|All|Static> Default is All, when no option is specified. When the molecule interacts with a (for example) metal nanoparticle, there are two types of interactions: the image field and the local field. The image field is caused by the dipoles induced into the nanoparticle by the molecule's electron density. This is always taken into account in a DIM/QM calculation. The local field arises by direct interactions of the nanoparticle with an external field. Addition of the LOCALFIELD key causes the DIM/QM calculation to include this effect, but by default this is not included in a DIM/QM calculation. In ADF2016 the correct behavior with local fields for the electric dipole-dipole, electric dipole-quadrupole, and electric dipole-magnetic dipole response tensors in the velocity gauge is implemented. This allows for correct calculation of surface-enhanced Raman optical activity (SEROA) (Ref. [*382* (page 929)]) and plasmonic circular dichroism (Ref. [*383* (page 929)]).

Default option should be used in most situations as other options are expert options. In effort to speed up calculation times, one can request to just calculate the Local Fields during the response section of the calculation by using the Resp keyword after local fields. All will calculate the local field for both ground state calculations and the response calculations, while static will calculate the local field during the ground state only.

EFIELD x y z The EFIELD key is used to include an external static electric field in the vector x y z. Internally, the static charges used in a DRF calculation use ADF's EFIELD block, and therefore use of the EFIELD block is not allowed with the DIMQM block. This key is included so the user may include an electric field that would normally be included by the EFIELD block.

SCREEN <ERF|EXP|ESP|GAU|NONE> {length} The SCREEN key indicates what functional form is used to screen the interactions between each DIM atom and the QM density. The choices are ERF (error function), EXP (exponential), ESP (error function for potential operator only), GAU (Gaussian), or NONE. For CPIM and PIM the default is GAU; for DRF, the default is ESP. In all cases, the default screening length is 1.0, but this may be changed with the optional length parameter.

FREQUENCY The FREQUENCY key turns on frequency-dependent parameters.

OM_[HCON] value The OM_H, OM_C, OM_O, and OM_N keys provide the resonance frequency (in atomic units) for the elements H, C, O and N, respectively. These keys are only for use with DRF and only when the older EXTERNALS block is used.

ALGORITHM <BEST|DIRECT|BRUTE|SINGLE|MULTI> DIM/QM can choose between several solver algorithms. The DIRECT method solves the linear system of equations directly with a LAPACK routine; this should be considered the most robust method, but scales poorly with the number of atoms ($O(N^3)$ where N is the number of atoms in the system). The other three methods use an iterative technique, The BRUTE method (brute force) takes into account all atoms in the matrix-vector multiply step, and scales as $O(N^2)$. The SINGLE method uses the single-level cell-multipole-method (CMM), wherein dipoles that are spatially similar are collected into a multipole moment which effectively reduces the system size. This also scales as $O(N^2)$ but with a lower prefactor than BRUTE. The MULTI method uses the multi-level cell-multipole-method, which uses larger and large multipole the farther apart the dipoles are. This is the fastest method and scales as $O(N \log N)$.

Due to technical limitations, CPIM can only use DIRECT. Further, depending on the system size DIRECT or SINGLE may be more efficient than MULTI. To simplify choosing the solving algorithm, there is a BEST option that chooses the best algorithm for the particular system. BEST is the default option for algorithm.

TOLERANCE tol The TOLERANCE key allows the user to specify a tolerance for the iterative solver. By default the tolerance is based on ADF's INTEGRATION key. This has no effect with the DIRECT solver.

NITER iterations The NITER key allows the user to specify the maximum number of iterations for the iterative solver. By default this is MAX(N/100, 200) where N is the number of DIM atoms.

VOLUME vol_in_nm3 The VOLUME key is used to specify the DIM system volume. The volume is used to determine how to partition the system for the cell multipole method (ALGORITHM options SINGLE or MULTI), and is also use to determine the scattering efficiencies for frequency-dependent polarizability calculations. The volume does not need to be supplied; if it is missing, it will be calculated based on each atom's radius and the MULTIPLIER key.

MULTIPLIER aterm bterm cterm An efficient way to get a approximation for the volume of the system is to sum the volume of each atom in the system, modified to account for the space between the atoms. This is done by modifying the atomic radius by a formula that takes into account the number of DIM atoms so that the effective radius changes with surface-to-bulk ratio. This formula is given by

$r_{eff} = -ar/N^b + c$

where r is the atomic radius, r_{eff} is the effective radius, N is the number of atoms, and the a, b, and c terms are the three parameters defined by the MULTIPLIER key. If the MULTIPLIER key is missing, the default values are 0.7, 0.5, and 1.13, respectively.

GRID <COARSE|MEDIUM|FINE> In the cell multipole method (ALGORITHM options SINGLE or MUTLI), a certain number of the closest atom interactions must be calculated explicitly. The GRID key controls how many atoms must be calculated this way, with COARSE being the least and FINE being the most. COARSE will be the fastest to calculate but may be numerically unstable. FINE is slowest to calculate but is the most stable. If the GRID key is missing, the default is MEDIUM.

FORCEFIELD The FORCEFIELD key indicates that the DIM/QM calculation will include the DIM/QM force field. Currently the only maintained potential is the Lennard-Jones 12-6 potential (see Ref. [*362* (page 928)]). This key is required to perform a DIM/QM geometry optimization and vibrational frequencies. By default, the DIM/QM force field is not included into the calculation. DRF gradients are not implemented.

Currently, DIM/QM geometry optimizations must be done in Cartesian coordinates which is specified in the GEOMETRY block. The user should be aware that ADF's default convergence criterion for a geometry optimization are relatively low, thus it is strongly suggested for a DIM/QM calculation to set the numerical integration quality (BeckeGrid) to good and change the convergence criterion of the max gradient to 1E-4. It is not possible to do geometry optimization with DRF.

COORDDEPEND The COORDDEPEND key indicates that the DIM/QM force field will be coordination dependent. This only effects the Lennard-Jones parameters for DIM atoms (see Ref. [*362* (page 928)]). By default, the DIM/QM force field is not coordination dependent.

CHEMISORPTION The CHEMISORPTION key will include chemisorption corrections for all atoms that have chemisorption parameters within a given cutoff radius. By default, the DIM/QM force field does not included chemisorption corrections.

COORDPAR atomtype e0 e1 r0 r1 CNmax Rmax Rmin The COORDPAR key allows the user to add additional coordination dependent parameter for a selected element type. atomtype specifies the element type (i.e., Ag for silver) for the given parameter set. e0, e1, r0, and r1 are the coordination dependent Lennard-Jones parameters; see Ref. [*362* (page 928)]) for more details. The coordination numbers of the DIM atoms are computed as an effective coordination number. This scheme requires a maximum and minimum cutoff distances, Rmax and Rmin respectively, and a maximum coordination number, CNmax. All parameters need to be in atomic units.

CHEMPAR atomtype e0 e1 r0 r1 CUTOFF The CHEMPAR key allows the user to add additional chemisorption dependent parameter for a selected element type. atomtype specifies the element type (i.e., N for nitrogen) for the given parameter set. e0, e1, r0, and r1 are the coordination dependent Lennard-Jones parameters; see Ref. [*362* (page 928)]) for more details. The code determines if there is a chemical bond by a cutoff distance parameter, CUTOFF. If the QM-DIM bond is within the cutoff, the code uses the chemisorption parameter; otherwise, the code uses the standard parameter set. All parameters need to be in atomic units.

CHEMBOND qmindex dimindex The CHEMBOND key indicates that there will a chemisorption correction used for the bond between the specified QM and DIM atoms. The user may repeat the CHEMBOD key up to 50 times to specify up to 50 different chemical bonds for the force field. The qmindex is an integer based on the order of atoms in the ATOMS block; i.e. the fifth QM atom in the ATOMS block would have qmindex = 5. The dimindex is the same but corresponds to the DIM atom involved in the bond. This key should only be used when the CHEMISORPTION key is also specified. When using CHEMBOND, the cutoff distance parameter for chemisorption correction parameter key will be ignored. It is suggested to use CHEMBOND if the user is generating a potential energy surface with a chemisorbed QM system.

PROJECTIONMATRIXPOINTS <ALL|CUTOFF radius|OFF> The PROJECTIONMATRIXPOINTS key specifies what DIM atoms to include for the projection matrix when removing rigid motions out of the gradient. The methods available are ALL, CUTOFF, or OFF. The ALL option causes PROJECTIONMATRIXPOINTS to include all DIM atoms. OFF will turn off the removal of rigid motions. CUTOFF includes any DIM atom points within a cutoff radius from the center of mass of the QM system to the DIM atom points and requires a cutoff radius to be given in Angstrom. This key only applies to a geometry optimization. If the PROJECTIONMATRIXPOINTS key is not given, the option CUTOFF with a cutoff radius of 25.4 Angstrom is assumed.

PRINTATOMICDIPOLES The PRINTATOMICDIPOLES key causes all the induced dipole moments of each DIM atom to be printed at the conclusion of each SCF cycle and each RESPONSE or AORESPONSE polarizability calculation. Because DIM/QM is typically used with many thousands of atoms, this can result in a large output file, but they may be useful for debugging purposes or to calculate electric fields. By default these are not printed.

PRINTLJPAR The PRINTLJPAR key specifies that all Lennard-Jones parameters used for the calculation will be printed in the output file. The QM atoms' Lennard-Jones parameters are also printed with the DEBUG key.

DEBUG The DEBUG key will print out extra information in the process of the calculation.

LocalDIM LocalDIM performs the calculation fully for a specified CutOffDist (in Bohr) but ignores the image dipoles from atoms outside of this distance. This option should converge to the full simulation and the distance should be chosen wisely. Must have LocalDirect keyword as well.

LocalDirect Tells LocalDIM calculation to use the direct solver for the DIM dipoles. This solver is generally much faster than the other solvers available when the system is small. Currently, LocalDIM only works with the intention of using this faster solver, so this option must be used with LocalDIM

DIMBUFFER Default value = 0.85. This keyword tells DIM how to decide for DRF2 calculations which atoms are in the same molecular grouping. This grouping decides if the charges polarize other atomic dipoles. For example, the charge for atoms in the same molecule will not polarize each other but they will polarize atomic dipoles in other molecules. This value should be adjusted based on the XYZ you supply and checked in the output file to ensure proper grouping behavior.

QMBUFFER Default value = 0.85. This keyword tells DIM how to decide if the QM atoms are "bonded" to one another to determine the force field parameters needed calculations requiring DIM/QM gradient terms. As with the DIMBUFFER keyword, this value should be adjusted for each system and checked in the output file to ensure proper behavior.

CutOffDist Distance in Bohr that determines if the image dipoles of a DIM atom is included in a LocalDIM calculation. Any DIM atom that is inside the supplied distance from any QM atom will be fully included in the calculation.

EXTERNALS key for DRF

The EXTERNALS block key controls the input data for the MM atoms. The EXTERNALS block is designed for DRF calculations. For each MM atom the following data are required:

```
EXTERNALS
 atm num grp-nam grp-num, char, x, y, x, pol
 ...
 GROUP
 {...}
end
```

atm Type of atom, i.e., H, O, ...

num number of atoms (optional)

grp-nam Name of the group to which the atom belongs

grp-num Number of the group to which the atom belong

char atomic charge (in atomic units)

x x-coordinate

y y-coordinate

z z-coordinate

pol atomic polarizability (in atomic units)

GROUP Indicates the end of group

The separation of molecules into GROUP's are important. Since in the many-body polarization operator only intermolecular interactions, i.e. only interaction between sites which do not belong to the some group, are included. Therefore, it is important that the combined string (grp-nam + grp-num) is unique for each GROUP.

An example of a EXTERNALS block for two water molecules:

```
EXTERNALS
 O 4 water 2, -0.6690, -11.380487, -11.810553, -4.515226, 9.3005
 H 5 water 2,  0.3345, -13.104751, -11.837669, -3.969549, 0.0690
 H 6 water 2,  0.3345, -10.510898, -12.853311, -3.320199, 0.0690
 GROUP
 O 7 water 3, -0.6690,  -1.116350,   9.119186, -3.230948, 9.3005
 H 8 water 3,  0.3345,  -2.822714,   9.717033, -3.180632, 0.0690
 H 9 water 3,  0.3345,  -0.123788,  10.538199, -2.708607, 0.0690
 GROUP
 {...}
end
```

In this block, the parameters for the DIM atoms are defined.

```
DIMPAR
  Element
    RAD val
    POL val
    CAP val
    CHAR val
    OM val
    OM1 val
    OM2 val
    GM1 val
    GM2 val
    SIZE val
    BOUND val
    STATIC
    EXP /path/to/experimental/dielectric/file
    DRUDE plasma damping {EV}
    FERMI val
    <LRTZ|LRTZ1> osc res damp {EV}
    LRTZ2 osc pls res damp {EV}
    LRTZ3 pls res damp {EV}
  SUBEND
  XYZ
    {/absolute/path/to/coordinates.xyz}
    {natoms
     elem x.xxx y.yyy z.zzz
     elem x.xxx y.yyy z.zzz
```

```
    elem x.xxx y.yyy z.zzz
    ...}
  SUBEND
  LCLFLD /absolute/path/to/localfields.txt
    {freq1 freq2 freq3 ..
    .....
    }
  GROUP groupname
    natoms
    elem x.xxx y.yyy z.zzz {pol=pol} (cap=cap) {char=char} {..}
    elem x.xxx y.yyy z.zzz {pol=pol} (cap=cap) {char=char} {..}
    elem x.xxx y.yyy z.zzz {pol=pol} (cap=cap) {char=char} {..}
    ...
  SUBEND
END
```

Element Within the DIMPAR block, you will need a sub-block that defines the parameters for each element that is in your DIM system. You will need to replace 'Element' with the element you are assigning parameters to, as in:

```
Ag
  ...
SUBEND
```

if you are assigning parameters to Ag. Note that the first letter MUST be capitalized and the second MUST be lowercase.

RAD val RAD specifies the atomic radius in the unit defined by the input file. RAD is required for all PIM calculations, all calculations with ALOGORITHM options SINGLE or DIRECT, and all frequency-dependent calculations where the AORESPONSE LIFETIME key is given.

POL val POL specifies the polarizability parameter (in a.u.) used in DRF or CPIM.

CAP val CAP specifies the capacitance parameter (in a.u.) used in CPIM.

CHAR val CHAR specifies the atomic charge (in a.u) used in DRF.

OM val OM specifies the resonance frequency (in a.u) used in DRF. This replaces the OM_[HCON] key in the DIMQM block.

OM1 val OM1 specifies the ω_1 parameter (in a.u) used in CPIM.

OM2 val OM2 specifies the ω_2 parameter (in a.u) used in CPIM.

GM1 val GM1 specifies the γ_1 parameter (in a.u) used in CPIM.

GM2 val GM2 specifies the γ_2 parameter (in a.u) used in CPIM.

SIZE val SIZE specifies the size-dependent parameter used in CPIM.

STATIC Calculate the response properties for this atom type as static during a frequency dependent calculation.

EXP /absolute/path/to/experimental/dielectric/file In PIM, the atomic polarizabilities are calculated from the dielectric constant. If you have access to the experimental dielectric constant, this may be supplied directly to DIM/QM. The values will be splined, so it is not necessary to ensure that each frequency at which you are calculating be in the file. DIM/QM expects the file to be formatted with the wavelength (in nm) in the first column, the real part of the dielectric in the second column, and the imaginary part of the dielectric in the third column. All other columns that may exist will be ignored, as well as lines beginning with the hash (#) symbol.

BOUND val A Drude function is typically written as

$$\epsilon_\infty - \frac{\omega_p^2}{\omega(\omega + i\gamma)}$$

with the second term being the Drude function, and the first term accounting for bound electrons. For a conductor with no bound electrons, $\epsilon_\infty = 1$ which is the default value for BOUND. To account for bound electrons you may set BOUND to a value greater than 1. This key only affects PIM.

DRUDE plasma damping {EV} The formula for a Drude function is

$$\epsilon_\infty - \frac{\omega_p^2}{\omega(\omega + i\gamma)}$$

where ϵ_∞ represents the bound electrons (as discussed for BOUND), ω_p (plasma) is the plasma frequency, and γ (damping) is the damping parameter (decay rate). Optionally, EV may be added to specify the values be read in units of electron volts, otherwise they are read in units of a.u. This key only affects PIM.

FERMI val The FERMI key is used to specify a Fermi velocity (in m/s) so that the Drude function may be size-corrected using a modified Drude function:

$$\epsilon_\infty - \frac{\omega_p^2}{\omega(\omega + i(\gamma + v_{\text{fermi}}/R_{\text{eff}}))}$$

where $+v_{\text{fermi}}$ is the Fermi velocity and R_{eff} is the effective nanoparticle radius. This can also be used in conjunction with EXP and DRUDE to size-correct experimental dielectric parameters. This key only affects PIM:

```
<LRTZ|LRTZ1> osc res damp {EV}
LRTZ2 osc pls res damp {EV}
LRTZ3 pls res damp {EV}
```

There are three forms of the Lorentzian function seen in the literature:

$$\sum_n \frac{f_n \Omega_{0,n}^2}{\Omega_{0,n}^2 - \omega^2 - i\Gamma_n \omega}$$

$$\sum_n \frac{f_n \omega_p^2}{\Omega_{0,n}^2 - \omega^2 - i\Gamma_n \omega}$$

$$\sum_n \frac{\Omega_{p,n}^2}{\Omega_{0,n}^2 - \omega^2 - i\Gamma_n \omega}$$

where $\Omega_{0,n}$ (res) is a bound electron resonance frequency, f_n (osc) is a bound electron oscillator strength, Γ_n (damp) is a bound electron excited state decay rate (or damping parameter), ω_p (pls) is the free electron plasma frequency, and $\Omega_{p,n}$ (pls) is the bound electron plasma frequency. You may choose the Lorentzian for against which your parameters were parametrized. The top form is LRTZ1, the middle is LRTZ2, and the bottom is LRTZ3. Because LRTZ1 is the most common, it is also aliased as LRTZ. Optionally, EV may be added to specify the values be read in units of electron volts, otherwise they are read in units of a.u. This key only affects PIM. You may give any of the form of the LRTZ key up to 50 times supply up to 50 Lortenzian functions to make a Drude-Lorentz function.

XYZ The XYZ sub-block is where the DIM atom coordinates are given. Two methods of supplying coordinates are allowed.

In-File Coordinates As an example of how to supply coordinates in-file, imagine you wish to calculate a Au dimer system on the Z-axis. You might define your coordinates as:

```
XYZ
 2
 Au 0.0 0.0 0.0
 Au 0.0 0.0 3.0
SUBEND
```

The first line gives the number of atoms to follow. Every line after that contains the element in the first column (first letter MUST be capitalized, second MUST be lowercase), then the x-component, then the y-component, then the z-component. You may not number the atoms.

External File Coordinates When the DIM system size becomes large, it is often more convenient to keep the DIM coordinates in a separate file. The XYZ block would then look like:

```
XYZ
 /absolute/path/to/coordinates.xyz
SUBEND
```

Note that you MUST include the absolute path to your file.

The .xyz file is set up identically to the in-file table, except that there is a space between the number of atoms and the first coordinate in case a comment need be added. The .xyz file for our dimer system would be:

```
2
A gold dimer (this line will be ignored)
Au 0.0 0.0 0.0
Au 0.0 0.0 3.0
```

LCLFLD /absolute/path/to/localfields.txt Text file of the local field induced dipoles that can be read in to reduce calculation time. The file must be in the following format:

```
freq1 freq2 freq3  ...
1
X
Atom1R-DipoleX    Atom1R-DipoleY    Atom1R-DipoleZ
Atom1I-DipoleX    Atom1I-DipoleY    Atom1I-DipoleZ
Atom2R-DipoleX    Atom2R-DipoleY    Atom2R-DipoleZ
Atom2I-DipoleX    Atom2I-DipoleY    Atom2I-DipoleZ
...
Y
Atom1R-DipoleX    Atom1R-DipoleY    Atom1R-DipoleZ
Atom1I-DipoleX    Atom1I-DipoleY    Atom1I-DipoleZ
Atom2R-DipoleX    Atom2R-DipoleY    Atom2R-DipoleZ
Atom2I-DipoleX    Atom2I-DipoleY    Atom2I-DipoleZ
...
```

Here freq1 freq2 freq3 ... are the frequencies that are being used in the response calculation and 1 tells the program that the following dipoles are to be used with the first frequency. X and Y tell the program that these are the induced dipoles from perturbations in the X and Y Cartesian direction respectively.

GROUP groupname The GROUP sub-block is where the DIM atom coordinates and parameters are given. Combines the XYZ subblock and Element subblock. A (unique) groupname is required (maximum 10 characters).

Example for a water molecule:

```
GROUP water1
3
O  0.00000  0.00000  0.59372  pol=5.7494  char=-0.6690
```

```
H   0.00000   0.76544  -0.00836   pol=2.7927   char=0.3345
H   0.00000  -0.76544  -0.00836   pol=2.7927   char=0.3345
SUBEND
```

The first line gives the number of atoms to follow. Every line after that contains the element in the first column (first letter MUST be capitalized, second MUST be lowercase), then the x-component, then the y-component, then the z-component. You may not number the atoms. At the end of the line for each element you can specify the DIM/QM parameters, as in the format in the example. The DIM/QM parameters can be 'POL', 'CAP', 'CHAR', 'OM1', 'OM2', 'GM1', 'GM2', 'SIZE', 'RAD', 'SPILLOUT', 'BOUND', 'DRUDE', 'FERMI', 'LRTZ', or 'EXP'.

4.4.6 FDE: Frozen Density Embedding

The Frozen-Density-Embedding (FDE) option invokes calculation of the effective embedding potential introduced by Wesolowski and Warshel [*184* (page 917)] in order to take into account the effect of the environment on the electronic structure of an embedded system. The embedding potential (Eq. 3 in Ref. [*240* (page 921)]) depends explicitly on electron densities corresponding to the embedded subsystem (e.g. a solvated molecule) and its environment (e.g. solvent). For a detailed review, see Ref. [*205* (page 918)]. The ADF implementation of the method is described in detail in Ref. [*185* (page 917), *217* (page 919)].

A time-dependent linear-response generalization of this embedding scheme was derived in Ref. [*186* (page 917)]. Its implementation in an approximate form, which assumes a localized response of the embedded system only (uncoupled FDE), is described in the supplementary material to Ref. [*187* (page 917)]. For possible drawbacks and pitfalls in connection with this approximation, see Refs. [*185* (page 917), *190* (page 918), *193* (page 918)].

The theory of coupled excited states for subsystems is described in Refs. [*296,297* (page 924)], and extended for general response properties in Ref. [*298* (page 924)]. This theory (subsystem TDDFT, coupled FDE) allows to treat the mutual response of several subsystems, including the ones that are considered environment.

A generalization of the FDE scheme to the calculation of NMR shieldings has been given in Ref. [*218* (page 919)], where also the approximations involved and possible problems are discussed.

With the exception of interaction energies, the current implementation in ADF only allows the calculation of molecular properties that only depend on the electron density and of response properties using TDDFT. For an application to the calculation of several molecular properties in solution and a comparison to the DRF model also available in ADF, see Ref. [*190* (page 918)]. For further applications of the ADF implementation, see Ref. [*189* (page 918)] (weakly interacting complexes), Refs. [*185* (page 917), *190-192* (page 918)] (solvent effects), and Refs. [*206-207* (page 918)] (other environment effects). To invoke a frozen-density embedding calculation, two additional specifications in the input are required. First, one or more frozen fragments have to be included in the FRAGMENTS block, and second, the block key FDE has to be included. In the simplest case, this input should look like this:

```
FRAGMENTS
   ...
   FragType FragFile type=FDE
   ...
END

FDE
   PW91K
end
```

In the FRAGMENTS block, for any fragment it is possible to specify the option type=FDE to indicate that the density of this fragment is kept frozen. This density is imported from the file FragFile. The frozen fragments have to be included in addition to the usual, non-frozen fragments. The atoms of the frozen fragments have to be included in the ATOMS block. As with normal fragments, the fragment found in the file will be rotated and translated to its position specified in the ATOMS block. For more details on specifying fragments, see the section 'fragment files'. In the FDE

input block, the recommended PW91k (also known as GGA97) approximant is recommended for the non-additive kinetic energy (the default is the local density approximant). A recommended alternative is NDSD. For all other options the defaults will be used.

Please note that throughout the FDE part of the documentation, the word "approximant" is used instead of the more usual "functional" to emphasize that the exact functional is not known, also in the case of the kinetic energy functional. In the literature one may encounter both words used interchangeably.

By including more than one frozen fragment, it is possible to use a frozen fragment that is a superposition of the densities of isolated molecules (this was possible in the previous version of ADF using the DENSPREP option). For a discussion and tests of the use of such approximate environment densities, see Ref. [*185* (page 917)].

There is no restriction on the use of symmetry in FDE calculations, and usually the correct symmetry will be detected automatically. However, in the preparation of frozen fragments that will be rotated and/or translated in the FDE calculation, one has to include the keyword NOSYMFIT for technical reasons.

In the current implementation, only the electron density of the embedded (non-frozen) system is calculated. Therefore, with the exception of interaction energies, only properties that depend directly on the electron density (e.g. dipole moments) are available. In particular, the calculation of energy gradients is not implemented yet. All quantities given in the output refer (unless explicitly specified otherwise) to the non-frozen system only.

To employ the extension of FDE to the calculation of NMR shieldings, the file TAPE10 has to be used in the FDE calculation (by including the option SAVE TAPE10), and subsequently the NMR shielding has to be calculated using the program NMR (not with EPR). The TDDFT extension of the FDE formalism allows the calculation of electronic excitation energies and polarizabilities. This extension is automatically activated if FDE is used in combination with the EXCITATIONS or the RESPONSE key. To allow the mutual response of several subsystems, see the section on [*subsystem TDDFT* (page 102)].

Fragment-specific FDE options

For each frozen fragment, several additional options can be applied. To do this, the fragment specification is used as a subblock key by appending a & sign. The subblock is terminated with SubEnd. This subblock key looks, in the most general form, as follows:

```
FRAGMENTS
  ...
  FragType FragFile type=FDE &
    {FDEOPTIONS [USEBASIS] [RELAX or FREEZEANDTHAW] [OPTIMIZE]}
    {FDEDENSTYPE [SCF | SCFexact | SCFfitted ]}
    {RELAXCYCLES n or FREEZEANDTHAWCYCLES n}
    {XC [LDA | GGA ggapotx ggapotc | MODEL SAOP]}
  SubEnd
  ...
END
```

FDEOPTIONS

 FDEOPTIONS USEBASIS If the USEBASIS option is specified, the basis functions of this frozen fragment will be included in the calculation of the embedded subsystem. This allows to expand the density of the embedded subsystem using not only atom-centered basis sets localized in the embedded subsystem but also the ones in the environment Ref. [*238* (page 920)]. In large-scale simulations using the embedding potential, this option is recommended to be used in the preparation stage to investigate the basis set dependence of the results (chapter 5.3 in Ref. [*205* (page 918)]). This option is also an indispensable element in the procedure introduced in Ref. [*238* (page 920)] to test approximants to the kinetic-energy component of the embedding potential introduced by Wesolowski and Warshel.

 FDEOPTIONS RELAX or FREEZEANDTHAW If the RELAX option (or equivalent FREEZEANDTHAW option) is specified, the density of this frozen fragment will be relaxed in freeze-and-thaw cycles [Ref.

240 (page 921)], i.e., the embedded subsystem is frozen, while this fragment is thawed. This is repeated, until convergence is reached or until the maximum number of iterations has been performed. By relaxing frozen fragments, it is possible to improve a given approximate environment density by including the polarization of the environment due to the embedded system. This option is recommended to be used in the preparation stage of a large-scale numerical simulation. The freeze-and-thaw calculations lead to a pair of electron densities (embedded system and environment) that minimizes the total energy. As a consequence, the electron density of the environment derived from the freeze-and-thaw calculations can be used as a reference to verify the adequacy of the assumed electron density for the environment in a large-scale simulation. Due to technical restrictions, freeze-and-thaw is not possible if an open-shell (unrestricted) fragment is present.

FDEOPTIONS USEBASIS RELAX or FDEOPTIONS USEBASIS FREEZEANDTHAW It is further possible to combine USEBASIS and RELAX or FREEZEANDTHAW. In this case, the basis functions of the non-frozen fragment will be included when the density of the fragment is relaxed. This allows fully relaxed calculations with supermolecular expansion of the electron density of each subsystem. This option is to be used to test approximants to the kinetic-energy component of the embedding potential introduced by Wesolowski and Warshel by means of the procedure introduced in [Ref. *238* (page 920)].

FDEOPTIONS OPTIMIZE If the key `FDEOPTIONS OPTIMIZE` is not specified, then only the active fragment is optimized in a Geometry Optimization run (the FDE fragments are not optimized). If you include `FDEOPTIONS OPTIMIZE`, then the corresponding FDE fragment will also be optimized. The `OPTIMIZE` key implicitly activates the option `RELAX`, which is necessary to evaluate the nuclear gradients of the respective fragment [Ref. *486* (page 935)]. The `OPTIMIZE` key only supports mono-molecular basis set expansions and can therefore NOT be combined with the `USEBASIS` option. For Geometry Optimizations the usage of the `FULLGRID` option is recommended. See the example *Example: Geometry optimization ICW FDE/sSDFT* (page 468).

Note:: When performing *geometry optimization* (page 121) ICW FDE one must:

- Disable *symmetry* (page 360) (`Symmetry NOSYM`)
- Use the *old optimizer branch* (page 121) (`Branch Old`)
- Use Cartesian coordinates in the *optimization procedure* (page 121) (`Optim Cartesian`)

FDEDENSTYPE The FDEDENSTYPE option can be used to specify which density is read from the fragment file. The possible options are:

FDEDENSTYPE SCF (or FDEDENSTYPE SCFexact) The exact density (not calculated using the fit functions) is used. This is the default.

FDEDENSTYPE SCFfitted The fitted density is used. This is less accurate but can be significantly faster.

RELAXCYCLES n or FREEZEANDTHAWCYCLES n This gives the maximum number of freeze-and-thaw cycles that are performed for this fragment. If the maximum number given in the FDE block is smaller, or if convergence is reached earlier, then fewer cycles are performed. For historical reasons, two equivalent keywords are available.

XC The XC option can be used to select the exchange-correlation potential that is used for this fragment when it is relaxed. By default, the same potential as for the non-frozen system is used, but in some cases it might be preferable to use another approximation for certain fragments. An example is given in Ref. [*189* (page 918)].

XC LDA This option selects LDA as exchange-correlation potential for relaxing this fragment.

XC GGA ggapotx ggapotc This selects a GGA potential for relaxing this fragment. The GGA potential is specified by giving the name of the exchange potential, followed by the name of the correlation potential. The available potentials are listed in the documentation for the XC key.

XC MODEL SAOP This selects the model potential SAOP for relaxing this fragment.

Kinetic energy approximants

The approximants to the kinetic energy dependent component of the embedding potential are described here.

```
FDE
  {approximants to the kinetic energy dependent
             component of the embedding potential}
  {CJCORR [rho_cutoff]}
  {GGAPOTXFD exchange approximant}
  {GGAPOTCFD correlation approximant}
end
```

approximants to the kinetic energy dependent component of the embedding potential
Several approximants to the kinetic-energy-dependent component of the effective potential given in Eq. (21) of [Ref. *184* (page 917)] are available. None of them is applicable if the embedded system is covalently bound to its environment. The user is recommended to look at the numerical value of the TSNAD(LDA) parameter which is given in the units of energy and can be considered as a measure of the overlap. The following rule of thumb should be applied: if this parameter is smaller than the estimated interaction energy between the embedded subsystem and the environment, then the available approximants are most probably adequate. If it exceeds this limit, the results can be less reliable. Printing TSNAD(LDA) is not done by default, as it can be quite time-consuming. Its printing is switched on by including "EXTPRINTENERGY", and "PRINTRHO2", and "FULLGRID" in the FDE input block. If no kinetic energy approximant is specified, by default the local-density approximation (Thomas-Fermi approximant) is used. For an assessment of approximants for weakly overlapping pairs of densities see Refs. [*238* (page 920), *239* (page 921), *188* (page 917), *241* (page 921)]. Based on these studies, the use of PW91k (= GGA97) is recommended.

APPROXIMANTS TO BE USED IN NORMAL APPLICATIONS

THOMASFERMI (default) Local-density-approximation form of vt[rhoA,rhoB] [*237* (page 920)] derived from Thomas-Fermi expression for Ts[rho] [*194* (page 918), *195* (page 918)].

GGA97 (or PW91K) Generalized-gradient-approximation form of vt[rhoA,rhoB] [*239* (page 921)] derived from the Lembarki-Chermette [*197* (page 918)] approximant to Ts[rho]. This approximant is currently the recommended one based on the numerical analysis of its accuracy [*188* (page 917), *239* (page 921)] and the fact that the used enhancement factor disappears at large reduced density gradients, i.e. where the second-order gradient-expansion approximation fails [*238* (page 920), *241* (page 921)].

NDSD Similarly to GGA97, the NDSD approximant is constructed by taking into account the asymptotic behavior of the functional vt[rhoA,rhoB] at small density gradients. In the construction of NDSD, the exact property of vt[rhoA,rhoB] at rho_A → 0 and for $\int$ rhoB = 2 given in Eq. A6 of Ref. [*279* (page 923)] is also taken into account. The analysis of the accuracy of this potential [*279* (page 923)] shows that NDSD is of the same or superior quality as GGA97. NDSD is, therefore, recommended as the successor of GGA97 to be used anywhere where the quality of the results depends directly on the accuracy of the potential vt[rhoA,rhoB], i.e., for obtaining electronic-structure-dependent properties. The analytical form of the corresponding approximant to the functional Ts^{nad} [rho_A,rho_B]$ exists (Eq. 23 in Ref. [*279* (page 923)]). It is not possible, however, to obtain the analytical form of the corresponding parent functional for the kinetic energy Ts[rho]. To reflect this and the fact that, similarly to the GGA approximants to vt[rhoA,rhoB], the numerical values of only first- and second derivatives of density are needed, the label NDSD (Non-Decomposable Second Derivatives) is used.

OBSOLETE APPROXIMANTS (can be used but GGA97 leads usually to a better embedding potential see [*238* (page 920), *239* (page 921)])

LLP91 Generalized-gradient-approximation form of vt[rhoA,rhoB] [*238* (page 920)] derived from Lee-Lee-Parr [Ref. *198* (page 918)] approximant to Ts[rho].

PW86k Generalized-gradient-approximation form of vt[rhoA,rhoB] [*238* (page 920)] derived from the Fuentealba-Reyes approximant to Ts[rho] [*242* (page 921)].

THAKKAR92 Generalized-gradient-approximation form of vt[rhoA,rhoB] [*239* (page 921)] derived from the Thakkar approximant to Ts[rho] [*201* (page 918)].

APPROXIMANTS WHICH MIGHT BE USEFUL ONLY FOR THEORY DEVELOPMENT

The accuracy of **some** of these approximants was investigated in detail [*239* (page 921), *238* (page 920), *188* (page 917), *241* (page 921)]. Each of them was shown to lead to a qualitatively incorrect embedding potential. They shouldn't be used in practical applications.

COULOMB Neglecting completely vt[rhoA,rhoB] (vt[rhoA,rhoB] equals zero) together with the exchange-correlation component of the embedding potential introduced by Wesolowski and Warshel.

TF9W The approximant to vt[rhoA,rhoB] [*184* (page 917)] derived from the second-order gradient expansion [*242* (page 921)]] for Ts[rho].

WEIZ The approximant to vt[rhoA,rhoB] [*241* (page 921)] derived from the von Weizsäcker approximant to Ts[rho] [*186* (page 917)].

OL91A Generalized-gradient-approximation form of vt[rhoA,rhoB] [*238* (page 920)] derived from the first Ou-Yang and Levy approximant to Ts[rho] [*200* (page 918)].

OL91B Generalized-gradient-approximation form of vt[rhoA,rhoB] [*239* (page 921)] derived from the second Ou-Yang and Levy approximant to Ts[rho] [*200* (page 918)].

E00 Generalized-gradient-approximation form of vt[rhoA,rhoB] [*263* (page 922)] derived from a kinetic energy functional by Ernzerhof [*264* (page 922)] which represents the gradient expansion approximation up to the fourth order.

P92 Generalized-gradient-approximation form of vt[rhoA,rhoB] [*263* (page 922)] derived from a kinetic energy functional by Perdew [*265* (page 922)] which represents the gradient expansion approximation up to the sixth order.

LONG DISTANCE CORRECTIONS TO THE EFFECTIVE POTENTIAL

CJCORR Option to switch on a long-distance correction. By default this option is not used. As was shown in Ref. [*220* (page 919)], with the available approximate kinetic-energy approximants, the embedding potential has the wrong form in the limit of a large separation of the subsystems. In particular, it was shown that this can have serious consequences in the case of "supermolecular expansion of electron density of each subsystem" calculations (USEBASIS option). In Ref. [*220* (page 919)], a correction is proposed that enforces the correct long-distance limit. (See also this reference for limitations of this correction.)

CJCORR [rho_cutoff] This option switches on the long-distance correction. This option has to be used in combination with one of the above kinetic-energy approximants. By default, a density cut-off of 0.1 is employed.

NONADDITIVE EXCHANGE-CORRELATION APPROXIMANT

GGAPOTXFDGGAPOTCFD Option to specify the non-additive exchange-correlation approximant. By default, in the construction of the effective embedding potential the exchange-correlation approximant that was specified in the XC block is used. It is possible to specify a different approximant with the GGAPOTXFD and GGAPOTCFD options. This is particularly useful in combination with the use of model potentials like SAOP, that can not be used in the embedding potential because of their orbital dependence. (For a discussion, see Ref. [*189* (page 918)].)

GGAPOTXFD exchange approximant The exchange approximant is used in the construction of the embedding potential. The same exchange approximants as in the XC key are available.

GGAPOTCFD correlation approximant The correlation approximant is used in the construction of the embedding potential. The same correlation approximants as in the XC key are available.

General FDE options

In addition to the fragment-specific options and the kinetic energy approximants, there are also a number of options available in FDE calculations that will be described in the following.

```
FDE
  {FULLGRID}
  {RELAXCYCLES n or FREEZEANDTHAWCYCLES n}
  {RELAXPOSTSCF or FREEZEANDTHAWPOSTSCF}
  {EXTPRINTENERGY}
  {PRINTRHO2}
  {ENERGY}
  {SDFTENERGY}
  {DIPOLE}
end
```

FULLGRID By default, FULLGRID is not used, and in FDE calculations the integration grid is generated as described in Ref. [*185* (page 917)] by including only atoms of the frozen subsystem that are close to the embedded subsystem in the generation of the integration grid. The distance cutoff used is chosen automatically, based on the extent of the basis functions of the embedded subsystem. (It can also be chosen manually, see the option qpnear in the INTEGRATION key) This scheme results in an efficient and accurate integration grid. However, it is possible that the default integration scheme is not accurate enough. This can be the case for weakly interacting systems and when the distance between the frozen and the embedded system is large. It is therefore recommended to check the quality of the default integration grid by comparing to results obtained using the full supermolecular grid (FULLGRID option).

If the subkey FULLGRID is included, all atoms of the frozen system are included in the generation of the integration grid. This results in the same grid that would be used in a supermolecular calculation of the combined frozen and embedded system. The integration grid generated by this option might be much larger than the default grid. This option should be used to check the quality of the default integration grid.

RELAXCYCLES n or FREEZEANDTHAWCYCLES n Specifies the maximum number n of freeze-and-thaw iterations [Ref. *240* (page 921)] that are performed (for frozen fragments with the RELAX) option. If a smaller number of iterations is specified as a fragment-specific option, for this fragment this smaller number is used. Furthermore, if convergence is reached earlier, no more iterations will be performed.

RELAXPOSTSCF or FREEZEANDTHAWPOSTSCF If this option is included, several post-SCF properties will be calculated after each freeze-and-thaw cycle [Ref. *240* (page 921)]. These are otherwise only calculated in the last cycle.

EXTPRINTENERGYPRINTRHO2 If the options EXTPRINTENERGY and PRINTRHO2 are included (both are needed and should be listed on separate lines), several additional quantities will be printed, including TSNAD(LDA). In order to obtain meaningful numbers, also the FULLGRID keyword (see above) has to be used.

ENERGY Option to switch on the calculation of the FDE energy as the sum of the energy E[rhoA] of the active, embedded system and the interaction energy Eint[rhoA,rhoB] of the embedded system with the frozen environment. This relies on the calculation of the total energy for the embedded system and all caveats and restrictions for total energy evaluations apply (see keyword *TOTALENERGY* (page 235)). All energy contributions are evaluated on the grid of the active subsystem. Some contributions to the interaction energy Eint[rhoA,rhoB] require an accurate integration grid in the region of the environment. Thus, in pure embedding calculations (without fragment-specific option RELAX), an accurate calculation of the FDE energy requires a full supermolecular integration grid (FULLGRID option). Details on the implementation and the performance of kinetic energy functionals for interaction energies are documented in Ref. [*263* (page 922)]

The calculation of the full, variationally minimized subsystem DFT energy, that is, the sum of the energy of two subsystems E[rhoA] and E[rhoB] and their interaction energy Eint[rhoA,rhoB] in the framework of FDE, is invoked if then the fragment densities are relaxed in freeze-and-thaw cycles (option RELAXCYCLES and

fragment-specific FDE option RELAX). In this case the supermolecular integration grid is not required. Instead, in each step of the freeze-and-thaw cycle, the critical energy terms are taken from the previous freeze-and-thaw step of the presently frozen fragment. The convergence of the energy contributions with the number of freeze-and-thaw iterations should be carefully monitored. Due to conceptual problems for the evaluation of the non-additive kinetic energy contribution, only two subsystems, that is, one frozen fragment, is supported for FDE energy calculations with freeze-and-thaw.

SDFTENERGY This is a generalization of the original ENERGY implementation that allows for the evaluation of the FDE/sDFT total energy. Usage of a supermolecular integration grid is recommended (FULLGRID option), because of the non-additive kinetic and XC contributions. EXTPRINTENERGY can be combined with this option to print the subsystem DFT energy after each freeze-and-thaw cycle. For pure embedding calculations, the SDFTENERGY implementation reads the total energy from each frozen fragment's TAPE21 file. The total energy of frozen fragments is not (re-)evaluated during FDE calculations. It is therefore mandatory to add the *TOTALENERGY* (page 235) keyword to the preparatory isolated fragment calculations.

DIPOLE The dipole of the supersystem is calculated as the sum of analytically integrated fragment dipole moments. Note that the dipole moment becomes origin dependent for charged (sub-)fragments. The current implementation does NOT take care of this.

Frozen Density Embedding with External Orthogonality

An implementation of External Orthogonality (EO) into the FDE framework in ADF can be found in Ref. [*384* (page 929)]. In this method in the fragment calculation one need to specify ghost atoms at the positions of the atoms in all the other fragments. Thus each fragment is calculated in the supermolecular basis.

```
AOMat2File
IgnoreOverlap
FDE
  EXTERNALORTHO {factor}
  ...
END
```

EXTERNALORTHO Used to specify the use of external orthogonality (EO) in the FDE block, with an optional factor (default factor is 1e6). Note that the overlap region is defined by ghost atoms, and the general keyword IGNOREOVERLAP is necessary. Additionally, the general keyword AOMAT2FILE is required to save some important fragment AO matrices to TAPE21 for use in an EO calculation. it is recommended that one uses the STOFIT keyword for the STOFIT density fitting method.

FDE and (localized) COSMO

COSMO solvation can be included in combination with FDE. This means that a COSMO cavity will be created that holds both the active and frozen fragments.

For very large systems the solution of the COSMO equations can become the bottleneck of the calculation. A local COSMO variant, which exploits the subsystem nature of the underlying electronic description, is implemented in ADF, see Ref. [*449* (page 933)]. This method, called LoCOSMO, is an approximation to regular COSMO for fragment-based electronic structure methods. If a given fragment is active (i.e. only its own density changes over the SCF cycles), the surface charges outside a given radius around this fragment will be kept fixed in magnitude (excluded from the COSMO optimization). However, no interactions are ignored by adding a corresponding constant term to the COSMO solution vector.

```
SOLVATION
  ..
  CHARGED LoCosmo LoCosmoDist
END
```

LoCosmo LoCosmoDist If LoCosmo is included the local COSMO will be used in the calculation. LoCosmoDist is a cutoff radius (Angstrom), which must be specified by the user, if LoCOSMO is included. All surface charges within this radius from any atom of the active fragment are included in the active charge space. A sensible value for LoCosmoDist is 5.0 Angstrom.

To be really useful, Freeze-and-Thaw cycles should be carried out with LoCOSMO. After each fragment calculation, the KF file LOCSURCH is written to disk. If present, this file will be automatically read in during the next (Lo-COSMO) fragment calculation. This means that if the internal FT cycles are used by specifying RELAXCYCLES, the code creates and reads in the file in a fully automatic fashion. LOCSURCH is neither created nor read during a regular COSMO calculation.

In case one calculates excitation energies with the EXCITATION block key, and LoCOSMO is specified, only the solvent response due to surface charges within the specified cutoff radius will be taken into account, see also Ref. [*450* (page 933)]. It should be noted that in a regular (uncoupled) FDE TDDFT calculation, the response due to the frozen density is neglected.

Subsystem TDDFT, coupled FDE

The linear-response subsystem TDDFT code implements the theory of coupled excited states for subsystems as described in Refs. [*296,297* (page 924)]. This theory is based on the FDE extension to excited states [*186* (page 917)], which is implemented in ADF in a local response approximation, i.e., neglecting the dynamic response of the environment [*187* (page 917)].

The subsystem TDDFT code allows to treat the mutual response of several subsystems, including the ones that are considered environment. A more typical situation would be a system composed of several equivalent chromophores treated as individual subsystems. In this case, the local response approximation leads to uncoupled excited states of the subsystems (hence the acronym FDEu is employed often), while the subsystem TDDFT code couples the monomer excitations to obtain the excited states of the total system (often denoted as coupled frozen density embedding, FDEc). This can be related to excitonic couplings between the monomers [*297* (page 924)].

The current implementation is restricted to NOSYM calculations and Singlet-Singlet excitations without frozen core approximation. It makes use of the ALDA kernel (including a Thomas-Fermi part for the contribution arising from the non-additive kinetic energy) for consistency with the uncoupled FDE implementation for excited states. Some features have not or not extensively been tested and should be used with great care, e.g., linear dependencies in the basis set. Details on the calculation of transition moments, oscillator and rotational strengths are described in Ref. [*298* (page 924)].

Subsystem TDDFT (FDEc) calculations can be invoked with the SUBEXCI key.

SUBEXCI input

FDEc calculations on coupled excited states first require that an uncoupled FDE-TDDFT calculation has been performed for every subsystem that should be included in the coupled calculation, and that the corresponding TAPE21 files, in which the considered subsystems are "active", have been saved (see the separate FDE input description). This means that it is not possible to use the information on frozen/inactive fragments from a TAPE21 file of a previous uncoupled FDE calculation, which contains all subsystems.

Although it is technically possible to use TAPE21 files from non-FDE calculations on the separate subsystems, this would lead to results that are inconsistent with the subsystem TDDFT methodology from Ref. [*296* (page 924)]. In any case, a previous TDDFT calculation for each subsystem that should be included in the coupling procedure is necessary. If that is not the case, the subsystem will still be considered in the calculation of the total electron density (needed in the setup of the exchange-correlation kernel), but will not be included in the coupling procedure.

The first subsystem should always be one of the coupled subsystems. The input will then look like the corresponding input for an uncoupled FDE-TDDFT calculation, but in addition should contain the following block:

```
SUBEXCI
  {LOWEST nlowest}
```

```
  {OPTSTATES list_of_optstates}
  {CTHRES coupling_threshold}
  {SFTHRES solutionfactor_threshold}
  {COUPLBLOCK}
  {TDA}
  {CICOUPL}
END
```

LOWEST nlowest The selection of the excited states to be coupled consists of two steps. First, a number of reference states are selected. As a default, the nlowest (default: 10) lowest excited states present on the fragment file for the first subsystem are considered.

OPTSTATES list_of_optstates If the keyword OPTSTATES is given, only those excited states of the first subsystem are considered as reference states that are given in the list_of_optstates (numbers of states separated by blanks).

CTHRES coupling_threshold Second, all excitations of all subsystems (present on the fragment TAPE21 files) with an excitation energy that differs by less than coupling_threshold (to be given in units of eV; default: 30 eV) from one of the reference states are selected to be included in the coupling. Note that additional excited states of system 1 may be included here.

COUPLBLOCK If COUPLBLOCK is specified in the input, all couplings between all of these local excited states are included. Otherwise (default), the coupling_threshold is also applied to select pairs of states for which couplings are calculated. I.e., couplings are not calculated if the two particular states to be coupled differ in energy by more than coupling_threshold.

SFTHRES solutionfactor_threshold To reduce the computational effort, it is possible to ignore the effect of orbital pairs with coefficients less than solutionfactor_threshold in the solution factors (TDDFT eigenvectors) of the underlying uncoupled calculation in the construction of the exact trial densities during the calculation of the coupling matrix elements. These orbital pair contributions are not ignored in the subsequent calculation of transition moments, oscillator, and rotational strengths. The default value of 0.00001 typically leads to a precision of the coupled excitation energies of about 0.0001 eV.

TDA TDA specifies the use of the Tamm-Dancoff-Approximation (*Tamm-Dancoff approximation* (page 163)) in the underlying uncoupled FDE-TDDFT calculations (Ref. [*364* (page 928)]). Contrary to the full SUBEXCI-TDDFT variant, SUBEXCI-TDA allows for the usage of hybrid functionals in the underlying uncoupled FDE-TDDFT calculations.

CICOUPL Within the Tamm-Dancoff Approximation, the couplings between localized excited states on different subsystems correspond directly to so-called exciton couplings (see Ref. [*364* (page 928)]). The CICOUPL keyword, in conjunction with TDA, prints these exciton couplings. It is also possible to use CICOUPL with full FDEc-TDDFT. In that case, the excitonic couplings between monomers are reconstructed from an effective 2x2 CIS-like eigenvalue problem, as e.g. done in Ref. [*297* (page 924)].

In addition, the input file may contain either an EXCITATION block or the keyword DIFFUSE. Both options lead to a slight adaption of the integration grid. Apart from this, the EXCITATION block will be ignored.

The key ALLOW PARTIALSUPERFRAGS is currently necessary to be able to use subsystem information for only one subsystem from a TAPE21 file of a previous FDE calculation:

```
ALLOW PARTIALSUPERFRAGS
```

Restrictions and pitfalls

In the current implementation, only the electron density of the embedded system is calculated. Therefore, with the exception of interaction energies, only properties that depend directly on the electron density (e.g., dipole moments)

are available. In addition, the TDDFT extension allows the calculation of electronic excitation energies and polarizabilities, and NMR shieldings can be calculated.

> **Warning:** EVERYTHING ELSE IS NOT YET IMPLEMENTED. THE RESULTS OBTAINED FOR OTHER PROPERTIES MIGHT BE MEANINGLESS.

Kinetic energy approximant: Although the effective embedding potential is derived from first principles using universal density approximants, the ADF implementation relies on approximations. Currently, two implemented approximations are recommended [*188* (page 917)]: PW91k (also known as GGA97) which uses electron densities and the corresponding gradients to express the non-additive kinetic energy component of the embedding potential, or TF (Thomas-Fermi LDA approximant), which does not use gradients at all. Either approximation is applicable only in cases where the overlap between electron densities of the corresponding interactions is small. Note: so far, no approximation has been developed for the strong-overlap case - two subsystem linked by covalent bonds for instance.

4.4.7 SCRF: Self-Consistent Reaction Field

The following sections describe the SCRF method and explain the related ADF input options. SCRF may not be available in the standard distribution on all platforms, contact SCM (support@scm.com) to request SCRF on your platform.

Introduction

by Donald Bashford, St. Jude Children's Research Hospital Memphis, updated October 26, 2011.

SCRF (Self-Consistent Reaction Field) is a method of accounting for the effect of a polarizable solvent (and optionally, a classical macromolecular system) on the quantum system. Consider first the case with only a polarizable solvent. The solvent is modeled as a dielectric continuum with a dielectric constant, EPSSOL, that fills the space outside the quantum system. The boundary between the interior (where the dielectric constant is unity) and the higher-dielectric exterior is the molecular surface, as defined by Connolly [*251* (page 921)]. The charges of the quantum system cause polarization of this continuum, giving rise to a reaction field which then acts back on the quantum system potentially altering its charge distribution. The SCRF algorithm calculates the reaction field through solutions to the Poisson or Poisson–Boltzmann equation, and iteratively obtains self-consistency between the reaction field and charge distribution of the quantum system. These ideas have their roots in Onsager's consideration of a dipole and a molecule with point-dipole polarizabilities inside a sphere [*252* (page 921)], and have also been developed in the Polarizable Continuum Model of Tomasi and co-workers [*253* (page 921), *254* (page 921)]. The first use of these ideas in DFT calculations using ADF was by Chen et al. [*255* (page 921)] and Mouesca et al. [*256* (page 922)]. At around the same time the Tomasi group also used the PCM model with DFT [*257* (page 922)]. In the past, SCRF calculations with ADF were done in the research groups of Noodleman, Bashford and Case using custom modifications of ADF. Now the method is available in a standardized form.

More recently, the ability to include a classical representation of a macromolecular environment, such as a protein around an enzyme active site, has been added to ADF. In what follows, we refer to the macromolecule surrounding the QM system as 'the protein' for historical reasons. It should be understood that the QM region and the protein region are distinct. For example, if the QM region includes some protein sidechain moieties, these are part of the QM system, and **not** the protein.

The protein exerts its effect in two ways: (1) The protein region can be assigned a dielectric constant (EPSPROT) that is different from either the QM region (1.0) or the solvent region (EPSSOL). This changes the reaction field that arises from the polarization of the environment by the charges of the QM atoms. (2) The protein has charges of it's own, which interact with the QM region in a way that is mediated by the complex dielectric environment described above. Calculations of this kind were first done by Li et al. [*343* (page 927)].

The specific algorithm is:

- Solve for the electronic structure in vacuum by the usual QM methods.

- Nucleus-centered partial charges are derived from the electronic structure using a least-squares potential-fitting SVD algorithm [*256* (page 922)], which is somewhat similar to the CHELPG algorithm [*258* (page 922)].

- *Only in calculations with a surrounding protein:* Calculate the electrostatic potential due to the partial charges of the protein in the three-dielectric environment defined by the protein and QM-region coordinates and radii. We refer to this field as the 'protein field' ϕ_{prot} .

- Using the nucleus-centered partial charges:

- Solve the Poisson equation (or, if an ionic strength is specified, the Poisson-Boltzmann equation) for ϕ_{sol} , the electrostatic potential in the presence of the above-described solvent dielectric environment.

- Solve the Poisson equation for ϕ_{vac} , the electrostatic potential in a uniform vacuum environment.

- The reaction field potential ϕ_{rf} is the difference between these two potentials: $\phi_{rf} = \phi_{sol}$ - ϕ_{vac} .

- Recalculate the electronic structure in the presence of the reaction field, and the protein field (if a protein field is present). This is done by adding the potentials ϕ_{rf} (and possibly ϕ_{prot}) to the total potential evaluated on the numerical integration grid when calculating Fock matrix elements.

- Check changes in energy for convergence, and if not converged, return to step 4. Note that because ϕ_{prot} does not depend on electronic structure in the QM region, there is no need to return to step 3.

The algorithm also includes a correction for the small difference between the fit and true electron densities.

At this time, the SCRF procedures are only available for single-point calculations. No geometry optimization or frequency calculations can be done together with SCRF.

Several user-settable options can affect the SCRF procedure.

It is important the the QM calculation be done without use of molecular symmetry, as the finite-difference Poisson solver involves approximations that do not respect symmetry. Therefore, one should specify SYMMETRY NOSYM in the general input.

The choice of atomic radii and the probe radius determines the location of the dielectric boundary. This molecular surface comprises spherical patches of the contact surface generated by a solvent-sized probe rolling over the atomic radii, the toroidal surfaces swept out as the probe roles in grooves between pairs of atoms, and the inverse spherical patches generated when the probe simultaneously touches three or more [*251* (page 921), *259* (page 922)]. In volumetric terms, the In volumetric terms, the molecular surface divides the space of all points that are accessible to any part of a probe sphere that cannot penetrate into any of the atomic spheres, for the space that is not accessible [*260* (page 922)]. The solvent accessible volume is assigned the solvent dielectric constant, while the inaccessible volume is assigned a dielectric constant of unity (the vacuum dielectric constant). Smaller radii move the dielectric surface closer to the atomic nuclei which typically leads to stronger calculated solvent effects.

The routine for calculating atomic partial charges chooses charges that best reproduce the potential outside the molecule that is generated by the nuclei and the electron density. It sets up a grid of potential-sampling points in a region outside the molecule, calculates the potential on this grid due to the electron density and nuclei, and finds the set of nucleus-centered charges that provides the best fit, in a least-squares sense, to the potential on the sampling points. The charge optimization is done using a singular value decomposition (SVD) method described by Mouesca et al. [*256* (page 922)]. These calculations can be affected by user options concerning constraints on total charge and dipole, charge-fitting grid spacing and SVs to be deleted.

The solution of the Poisson or Poisson–Boltzmann equation utilizes libraries from Donald Bashford's MEAD programming suite. These use a finite-difference method that involves setting up cubic lattices around the molecule. Finer grids can be nested inside coarser ones to help manage trade-offs between accuracy and computational cost. The finest grid should cover the entire quantum system (that is, regions of significant electron density), and for good accuracy of ϕ_{rf} should be no coarser than about 0.15 Å. The outermost grid should extend 10 to 15 Å into the space beyond the model so that boundary conditions are accurate. A reasonable scheme for grid selection based on atomic coordinates is implemented as the default, but the MEAD grids are also user-adjustable.

The SCRF method should not be used in the same ADF run with other solvation modeling methods such as COSMO or DRF. However, it is often useful to do geometry optimization with COSMO to get the single-point geometry for an SCRF run.

Additional considerations with a protein environment

The special considerations that apply when a protein environment is specified are similar to those in QM/MM calculations. The classical macromolecular region is specified by a .pqr file that provides the Cartesian coordinates, partial charges and radii of all the atoms of the region, as well as atom identifiers such as residue and atom numbers, in a format similar to PDB format. The QM atoms are specified in the ADF input file in the usual way. In contrast to the QM/MM situation, the ATOM block in the ADF input should specify only QM atoms, while the .pqr file should specify only the atoms of the protein. It is essential that no symmetry be used in the QM calculation (i.e. SYMMETRY NOSYM) even if the QM system possesses symmetries. The protein surroundings, as well as the numerical asymmetries of the finite-difference Poisson solver, will break any symmetry that might exist, but the QM part of the calculation has no way to know in advance about this broken symmetry.

Classical–QM bonds

If there is no covalent linkage between atoms in the QM region and the protein, the situation is straightforward. If there is such a linkage, some of the same considerations regarding capping atoms and link atoms as in QM/MM calculations apply [ref. Sec. 1.2 of ADF QM/MM manual]. One must typically add capping hydrogen atoms to the QM system to satisfy valence requirements. Because the interaction between the regions is purely electrostatic, we do not encounter the classical-side valency issues or van der Waals interaction issues that one finds in the QM/MM case. This is because this model involves no concept of bonds among the protein atoms. However, the link atom will typically be only a few tenths of an Angstrom from the capping hydrogen atom, so if the link atom has a significant partial charge, it could perturb the QM system in strange ways. Therefore we typically choose the link atom to be one that would normally only have a small partial charge (such as an aliphatic carbon) and then set its charge to zero in the .pqr file, perhaps with some minor adjustments of nearby partial charges to maintain the correct total charge.

Generation of the .pqr file

Typically one has at hand a PDB-format file of the macromolecule that one would like to include as the 'protein' part of the environment, and one needs to generate a suitable pqr file from this. PDB files usually lack hydrogen atoms, and often have missing heavy atoms at chain termini and in flexible loops. These problems need to be addressed in much the same way as for standard molecular mechanics calculations, although the requirements for a pqr file are less rigorous given the lack of classical bonds in the model. Although there are no specific tools for generating PQR files in the ADF distribution, a number of tools are available in the molecular modeling community. Amber users can use the usual LEaP tool to prepare prmtop and coordinate files, and then the utility, ambpdb can be used with the -pqr flag to generate a pqr file. Users of CHARMM can generate a structure file (PSF file) for the protein and a CHARMM-generation PDB file (containing hydrogens and using CHARMM naming conventions) and then use a perl script provided by the MEAD suite (available from http://stjuderesearch.org/site/lab/bashford) to generate a pqr file from the CHARMM-generated files. Another option is to use pdb2pqr either as a downloaded program or as web server. Information about pdb2pqr can be found at http://www.poissonboltzmann.org/pdb2pqr.

Once the initial version of the pqr file is generated, it is likely that some hand editing will be needed, particularly if there are covalent linkages between the protein and QM atoms. In cases with no linkages, such as a (QM) drug in a (classical) protein binding site, the most straightforward procedure is to leave the drug out during generation of the pqr file. Then no pqr-editing needs to be done, but it is necessary that the QM coordinates and the coordinates in the pqr file be compatible, e.g., that the drug is correctly positioned in the binding site. If there are covalent links between the QM and protein regions, for example if some sidechains in an enzyme active site are part of the QM region, then the pqr file will probably have been prepared with these sidechains included. *Any atoms that will be part of the QM region must be removed from the pqr file.* If this is not done, the QM region will *feel* protein partial charges that sit on top of the QM-region nuclei; this will generate unphysical results or outright program crashes. There is no need to worry about dangling valences in the pqr file, but it may be necessary to edit link-atom charges, as noted above. The QM atom set may need to include capping atoms, (typically hydrogen) to satisfy valence requirements where bonds cross the QM–protein boundary.

Format of the pqr file

A pqr file is similar to a PDB (Protein Data Bank) file but with atomic charge and radii in the occupancy and B-factor columns, respectively. More specifically, lines beginning with either 'ATOM' or 'HETATM' (no leading spaces) are interpreted as a set of tokens separated by one or more spaces or TAB characters. Other lines are ignored. The tokens (including the leading ATOM or HETATM are interpreted as follows:

```
ignored ignored atName resName resNum x y z charge radius chainID
```

The first two (ignored) tokens must be present, or the line will not be parsed correctly. The chainID token is optional, and any tokens beyond that are ignored. Tokens can be of arbitrary length, but must not contain spaces or tabs. Lines that do not begin with "ATOM" or "HETATM" are ignored. The programs make no distinction between ATOMs and HETATMs. *No atname-resnum-chainid combination can occur more than once.*

Note that the .pqr format does not support some PDB-isms such as a altLoc fields, and a one-character chainID between resName and resSeq. Doing so would break the whitespace separated tokens convention that allows for easy processing with perl scripts, etc. Instead we put 'chainID' in a position more or less analogous with the PDB segID. (Note that the pdb2pqr program differs on this point, and pqr files with chainIDs between resName and resSeq may need to be modified.

SCRF Input

The SCRF input is contained in an SCRF input block as shown below, optional keywords being surrounded by curly brackets.

```
SCRF
  MEADGRID string integer real
  RADIUS string real
  {CYCLES integer}
  {TOLERANCE real}
  {ATOM_MAXR real}
  {CHGFIT_CONSTRAIN string}
  {DELATOM N Iat1 Iat2 ... IatN}
  {GRID_SPACING real}
  {SVD_CONDITION real}
  {SV_DELETE integer}
  {EPSSOL real}
  {IONIC_STR real}
  {SOLRAD real}
  {PROTEIN string}
  {EPSPROT real}
  {SAVESTATE string}
END
```

The SCRF block contains two mandatory keys: MEADGRID and RADIUS. All other keys are optional.

MEADGRID string integer real Specifies the centering style, dimension and spacing for the MEAD grid. Recognized centering styles are "ON_ORIGIN" and "ON_GEOM_CENT". The grid dimension specifies the number of points on one edge of a cubic grid. The grid spacing is given in Angstroms. The edge length of the grid is the product of the dimension minus 1 and the spacing. Multiple records may be used to specify sequentially finer grid levels, but finer grids must fit within the coarsest grid.

RADIUS string real Specifies the radius in Angstroms for an atom type. Used in fitting the ADF electronic structure to partial atomic charges and for defining the boundary between regions of low and high dielectric in MEAD. The atom types should be the same as those used in the ATOMS input block. There must be one RADIUS record for each atom type in the ATOMS input block.

CYCLES integer Specifies the maximum number of cycles of SCRF to perform. Whether or not the SCRF run has converged, it will terminate when the number of cycles exceeds the value specified by CYCLES.

TOLERANCE real Specifies convergence criterion in kcal/mol for SCRF. For each cycle of SCRF the sum of ADF

---

energy, reaction energy, the energy correction and nuclear reaction energy is calculated. If the difference in subsequent sums is less than the TOLERANCE value, SCRF is considered to have converged. Defaults to 0.01.

ATOM_MAXR real Specifies the outer atomic radius in Angstroms for the system. For each atom, grid points that lie between the atomic radius specified by the RADIUS keyword and the outer atomic radius specified here are included in charge fitting. Defaults to 5.0.

CHGFIT_CONSTRAIN string Specifies the type of constraints to be used in charge fitting. Recognized constraints are "NONE", "CHARGE" or "DIPOLE". NONE specifies no constraints will be applied, CHARGE specifies that only the molecular charge will be constrained and DIPOLE specifies that both the molecular charge and dipole will be constrained. Default is DIPOLE.

DELATOM N Iat1 Iat2 ... IatN Specifies which atoms should be excluded from the charge fitting procedure. N is the number of atoms to be excluded, followed by a space separated list of atoms numbers Iat1, .., IatN, that should be excluded from the charge fitting procedure. The input order in the ATOMS input block is used to identify the excluded atoms.

GRID_SPACING real Specifies the grid spacing in Angstroms for the charge fitting grid. The default is 0.2.

SVD_CONDITION real Specifies a condition number threshold for inclusion of singular values (SV) in singular value decomposition (SVD) during charge fitting. The default is 0.000001.

SV_DELETE integer Instead of using a condition number threshold for deciding which SV to include in charge fitting, SV_DELETE may be used to specify how many SV should be excluded. The smallest SV are excluded first. The default is to use a condition number threshold. If both SV_DELETE and SVD_CONDITION are specified, the SV_DELETE value will take precedence.

EPSSOL real Specifies the solvent dielectric for MEAD. Defaults to the dielectric of water: 80.0.

IONIC_STR real Specifies an ionic strength in mol/L for the solvent in MEAD. Defaults to 0.0.

SOLRAD real Specifies the radius in Angstroms of a solvent-sized probe that rolls along the surface of the molecular system to define the dielectric boundary. Defaults to a water-sized probe size of 1.4.

PROTEIN string Use of this keyword turns on the SCRF solinprot option by specifying the prefix of a pqr file containing the protein definition for MEAD solinprot. The filename suffix must be pqr. Pqr format contains one line per atom and begins with the ATOM keyword followed by 10 fields separated by white space and in the order: atom number, atom name, residue name, residue number, x, y and z coordinates, partial atomic charge and atomic radius. The atoms of the quantum mechanical system should NOT be included in the pqr file.

EPSPROT real Specifies the protein dielectric for MEAD. Defaults to 4.0.

SAVESTATE string Specifies a filename in which ground state data for a subsequent VSCRF calculation will be saved. The data is saved in a binary KF file.

4.4.8 VSCRF: Vertical Excitation Self-Consistent Reaction Field

Introduction

by Donald Bashford, St. Jude Children's Research Hospital Memphis, October 26, 2011.

VSCRF is a method of accounting for the effect of the solvent environment (and possibly a macromolecular environment) on the energy of absorption/emission transitions between electronic states. It is built on the *Delta* SCF method of calculating electronic transition energies and a continuum dielectric model of the environment in a way that is consistent with the Franck–Condon principle. The original development and application of the procedure can be found in the papers of Liu et al. [*344* (page 927)] and Han et al. [*345* (page 927)].

Suppose one is calculating the energy for a transition between an initial state i with total density (including nuclei) ρ_i and a final state f with total density ρ_f. State i has existed long enough that the dielectric environment surrounding it has had time to fully respond, including changes in the distributions of nuclear coordinates and solvent-molecule dipole

moments. The density ρ_i has therefore given rise to a reaction field $\phi^{(r)}_{i,eq}$ and is self-consistent with it according to the ideas described in the SCRF part of the documentation, therefore the energy in solvent of the initial system is

$$G^i(\rho_i) = E_0(\rho_i) + 1/2 \int \rho_i(x)\phi^{(r)}_{i,eq}(x)d^3x$$

where E_0 is the vacuum density functional. During the excitation process, the change in density $\Delta \rho_{fi} = \rho_f - \rho_i$ must work against the non-optical component of the pre-existing equilibrium reaction field as well as the optical reaction field arising in response to $\Delta \rho$ itself. The full theoretical development is given in the paper of Liu et al. [*344* (page 927)] and the result is that the excitation energy is:

$$\Delta G^{if}_{ex} = E_0(\rho_i) - E_o(\rho_f)1/2 \int \Delta\rho_i(x)[2\phi^{(r)}_{i,eq}(x)\Delta\phi^{(r)}_{op}(x)]d^3x$$

where $\Delta \phi^{(r)}_{op}$ is the reaction field that arises from the *optical* dielectric constant which is related to the index of refraction: $\epsilon_{op} = n^2 \approx 2$ for organic liquids.

In the following description of the VSCRF algorithm, we adopt the language of an optical absorption calculation. The initial state is therefore referred to as the ground state, and the final state as the excited state. For emission calculations, these roles would be reversed. If there is a macromolecular environment surrounding the protein that modeled by macroscopic dielectric theory as described in the SCRF documentation in this manual, this is referred to as the 'protein'. With these definitions in mind, the VSCRF algorithm is:

- An SCRF calculation for the ground state, either with or without a surrounding protein, is a prerequisite for a VSCRF calculation.

- The electronic structure of the excited state (with occupations initially selected by the user) is calculated in vacuum.

- The ground-state reaction field, the protein field (if any) and electron density are read in from the results of the previous ground-state calculation. It is essential that the excited-state and ground state calculations use exactly the same geometry, ADF grids, basis functions, XC functionals, MEAD grids and so on.

- The excited-state electron density calculated in step 1 is subtracted from the stored ground-state density to obtain $\Delta\rho$.

- Nucleus-centered partial charges are derived from $\delta\rho$ using a least-squares potential-fitting SVD algorithm [*256* (page 922)] which is somewhat similar to the CHELPG algorithm [*258* (page 922)].

- Set the exterior dielectric environment to the optical dielectric constant of the solvent, and the optical dielectric constant of the protein. These are equal to the square of the index of refraction of these materials, and are typically ≈ 2 (the default value).

- Using the nucleus-centered partial charges for $\Delta\rho$, solve the Poisson equation using the finite-difference method in the above dielectric environment to obtain $\Delta\phi_{sol}$.

- Set the exterior dielectric constant to 1 everywhere and solve the Poisson equation by the finite-difference method to obtain $\Delta\phi_{vac}$

- The optical reaction field to the excitation, $\Delta\phi^{(r)}_{op}$ is the difference between $\Delta\phi_{sol}$ and $\Delta\phi_{vac}$. In this subtraction, artifacts related to the finite difference method cancel out.

- Recalculate the electronic structure in the presence of the optical reaction field, as well as the previously stored ground-state reaction field and the protein field (if any). This is done by placing these potentials on the ADF numerical integration grid when calculating Fock matrix elements.

- Check changes in energy for convergence, and if not converged, return to step 6.

It should be noted that some additional calculations may be needed to obtain a proper excited-state energy. For example, if the first excited singlet state is prepared by promoting one of the α electrons from the HOMO to the LUMO, the resulting (single-determinant) wave function is not a pure singlet state but actually a mixture of singlet and triplet states [*323* (page 926)]. Calling this mixed state S'$_1$, the resolution of the problem is to also calculate

the energy of a corresponding triplet state, T_1 (prepared, for example, by adding one α electron to the LUMO and removing one β electron from the HOMO) and using the formula [*323* (page 926)]:

$$E_{S_1} = 2_{S_1'} - E_{T_1}$$

See Liu et al. [*344* (page 927)] or Han et al. [*345* (page 927)]. for specific examples of this in the ADF/VSCRF context.

VSCRF Input

The VSCRF input is contained in a VSCRF input block as shown below, optional keywords being surrounded by curly brackets.

```
VSCRF
  MEADGRID string integer real
  RADIUS string real
  INITIAL_STATE string
  {CYCLES integer}
  {TOLERANCE real}
  {ATOM_MAXR real}
  {CHGFIT_CONSTRAIN string}
  {DELATOM N Iat1 Iat2 ... IatN}
  {GRID_SPACING real}
  {SVD_CONDITION real}
  {SV_DELETE integer}
  {EPS_OPT_SOL real}
  {SOLRAD real}
  {PROTEIN string}
  {EPS_OPT_PROT real}
END
```

The VSCRF block contains three mandatory keys: MEADGRID, RADIUS and INITIAL_STATE. All other keys are optional. It is highly recommended that the parameters specifying the MEAD grid and the details of the charge-fitting SVD procedure be the same for both the SCRF initial state and the VSCRF final state calculations. These procedures involve some numerical error, and keeping parameters the same will promote cancellation of these errors as energy differences are taken as the final results.

MEADGRID string integer real Specifies the centering style, dimension and spacing for the MEAD grid. Recognized centering styles are "ON_ORIGIN" and "ON_GEOM_CENT". The grid dimension specifies the number of points on one edge of a cubic grid. The grid spacing is given in Angstroms. The edge length of the grid is the product of the dimension minus 1 and the spacing. Multiple records may be used to specify sequentially finer grid levels, but finer grids must fit within the coarsest grid. It is highly recommended that the parameters specifying the MEAD grid be the same for both the SCRF initial state and the VSCRF final state calculations.

RADIUS string real Specifies the radius in Angstroms for an atom type. Used in fitting the ADF electronic structure to partial atomic charges and for defining the boundary between regions of low and high dielectric in MEAD. The atom types should be the same as those used in the ATOMS input block. There must be one RADIUS record for each atom type in the ATOMS input block.

INITIAL_STATE string Specifies the filename of a binary KF file containing the ground state data for VSCRF. The KF file is created by a preliminary SCRF calculation using the SAVESTATE option. It is highly recommended that the parameters specifying the MEAD grid and the details of the charge-fitting SVD procedure be the same for both the SCRF initial state and the VSCRF final state calculations.

CYCLES integer Specifies the maximum number of cycles of VSCRF to perform. Whether or not the VSCRF run has converged, it will terminate when the number of cycles exceeds the value specified by CYCLES.

TOLERANCE real Specifies convergence criterion in kcal/mol for VSCRF. For each cycle of VSCRF the sum of

ADF energy, Potential Term, and Response Term is calculated. If the difference in subsequent sums is less than the TOLERANCE value, VSCRF is considered to have converged. Defaults to 0.01.

ATOM_MAXR real Specifies the outer atomic radius in Angstroms for the system. For each atom, grid points that lie between the atomic radius specified by the RADIUS keyword and the outer atomic radius specified here are included in charge fitting. It is highly recommended that the parameters specifying the details of the charge-fitting SVD procedure be the same for both the SCRF initial state and the VSCRF final state calculations. Defaults to 5.0.

CHGFIT_CONSTRAIN string Specifies the type of constraints to be used in charge fitting. Recognized constraints are "NONE", "CHARGE" or "DIPOLE". NONE specifies no constraints will be applied, CHARGE specifies that only the molecular charge will be constrained and DIPOLE specifies that both the molecular charge and dipole will be constrained. It is highly recommended that the parameters specifying the details of the charge-fitting SVD procedure be the same for both the SCRF initial state and the VSCRF final state calculations. Default is DIPOLE.

DELATOM N Iat1 Iat2 ... IatN Specifies which atoms should be excluded from the charge fitting procedure. N is the number of atoms to be excluded, followed by a space separated list of atoms numbers Iat1, .., IatN, that should be excluded from the charge fitting procedure. The input order in the ATOMS input block is used to identify the excluded atoms. It is highly recommended that the parameters specifying the details of the charge-fitting SVD procedure be the same for both the SCRF initial state and the VSCRF final state calculations.

GRID_SPACING real Specifies the grid spacing in Angstroms for the charge fitting grid. It is highly recommended that the parameters specifying the details of the charge-fitting SVD procedure be the same for both the SCRF initial state and the VSCRF final state calculations. The default is 0.2.

SVD_CONDITION real Specifies a condition number threshold for inclusion of singular values (SV) in singular value decomposition (SVD) during charge fitting. It is highly recommended that the parameters specifying the details of the charge-fitting SVD procedure be the same for both the SCRF initial state and the VSCRF final state calculations. The default is 0.000001.

SV_DELETE integer Instead of using a condition number threshold for deciding which SV to include in charge fitting, SV_DELETE may be used to specify how many SV should be excluded. The smallest SV are excluded first. The default is to use a condition number threshold. If both SV_DELETE and SVD_CONDITION are specified, the SV_DELETE value will take precedence. It is highly recommended that the parameters specifying the details of the charge-fitting SVD procedure be the same for both the SCRF initial state and the VSCRF final state calculations.

EPS_OPT_SOL real Specifies the dielectric constant of the solvent at optical frequencies for MEAD. This value is equal to the square of the index of refraction of the solvent. Defaults to the value for water: 2.0.

SOLRAD real Specifies the radius in Angstroms of a solvent-sized probe that rolls along the surface of the molecular system to define the dielectric boundary. Defaults to a water-sized probe size of 1.4.

PROTEIN string Use of this keyword turns on the VSCRF solinprot option by specifying the prefix of a pqr file containing the protein definition for MEAD solinprot. This option can only be used if the ground state SCRF calculation also used a protein. The filename suffix must be pqr. Pqr format contains one line per atom and begins with the ATOM keyword followed by 10 fields separated by white space and in the order: atom number, atom name, residue name, residue number, x, y and z coordinates, partial atomic charge and atomic radius. The atoms of the quantum mechanical system should NOT be included in the pqr file.

EPS_OPT_PROT real Specifies the dielectric constant of the protein at optical frequencies for MEAD. This value is equal to the square of the index of refraction of the protein. Defaults to the value for organic liquids: 2.0.

4.4.9 3D-RISM: 3D reference Interaction Site Model

Introduction

The three-dimensional reference interaction site model with the closure relation by Kovalenko and Hirata (3D-RISM-KH) provides the solvent structure in the form of a 3D site distribution function, $g_\gamma^{UV}(r)$, for each solvent site, γ. Note that the use of 3D-RISM as implemented in ADF is an expert option.

Warning: We have no workflow that consistently produces accurate results.

It enables, at modest computational cost, the calculations of thermodynamics, electronic properties and molecular solvation structure of a solute molecule in a given molecular liquid or mixture. Using 3D-RISM, one can study chemical reactions, including reaction coordinates and transition state search, with the molecular solvation described from the first principles. The method yields all of the features available by using other solvation approaches. The 3D-RISM part of ADF has not been parallelized.

Details on the implementation of 3D-RISM-KH in ADF can be found in Ref. [*300* (page 924)], with applications in Ref. [*301* (page 924)]. The theory of 3D-RISM-KH in combination with DFT can be found in Refs. [*302-304* (page 924), *349* (page 927)]. A combination of 3D-RISM-KH with FDE (frozen-density embedding) can be found in Ref. [*305* (page 924)].

Similar to explicit solvent simulations, 3D-RISM properly accounts for chemical peculiarities of both solute and solvent molecules, such as hydrogen bonding and hydrophobic forces, by yielding the 3D site density distributions of the solvent. Moreover, it readily provides, via analytical expressions, all of the solvation thermodynamics, including the solvation free energy potential, its energetic and entropic decomposition, and partial molar volume and compressibility. The expression for the solvation free energy (and its derivatives) in terms of integrals of the correlation functions follows from a particular approximation for the so-called closure relation used to complete the integral equation for the direct and total correlation functions.

Solvation free energy

The solvation free energy can be calculated as:

> Solvation Free Energy =
> [(Total Bonding Energy) + (Internal Energy)
>   - T * (Entropy) + (Excess Chemical Potential)]_3D-RISM
> - [(Total Bonding Energy) + (Internal Energy) - T * (Entropy)]_Gas-Phase

If one assumes that the internal energy and the vibrational and rotational entropy of the solute molecule is the same in solution

> Solvation Free Energy =
> [(Total Bonding Energy) + (Excess Chemical Potential)]_3D-RISM
> - [(Total Bonding Energy)]_Gas-Phase

However, a formally accurate calculation should include the difference between the thermal corrections from frequency calculations produced by ADF in the SCF calculation with 3D-RISM-KH solvation and in gas phase.

Input

When performing 3D-RISM simulations, each atom in the **ATOMS** block must have two parameters specified, SigU and EpsU, for example:

```
ATOMS
  C      0.00     0.00     0.00     SigU=3.50     EpsU=0.066
  ...
END
```

The SigU and EpsU parameters have the same meaning as Sigma_alpha and Eps_alpha for atoms of the solvent in the SOLVENT sub-block below. They can be obtained from a Lennard-Jones force-field parameter sets.

All 3D-RISM-related input keys are contained in a **RISM** input block. Below, only the mandatory keywords are shown. Optional keywords are described in the next section.

```
RISM title
  RISM1D
    FLUIDPARAM temper=298. DielConst=78.497 UTotDens=1/A3 0.03333
  SUBEND
  SOLVENT ArbitrarySolventName
    UNITS uWeight=g/mol  ULJsize=A  ULJenergy=kcal/mol Ucoord=A Udens=1/A3
    PARAMETERS Weight NAtomTypes
      N1    Z_alpha1   Sigma_alpha1   Eps_alpha1   X1_1 Y1_1 Z1_1
                                                   X1_2 Y1_2 Z1_2
                                                   ...  ...  ...
      N2    Z_alpha2   Sigma_alpha2   Eps_alpha2   X2_1 Y2_1 Z2_1
                                                   X2_2 Y2_2 Z2_2
                                                   ...  ...  ...

      ...
    DENSPE=density
  SUBEND
  SOLUTE ArbitrarySoluteName
    BOXSIZE    sizeX   sizeY   sizeZ
    BOXGRID     npX     npY     npZ
  SUBEND
END
```

The **RISM1D** sub-block contains general parameters for the 1D-RISM calculation of the solvent(s). Even though all RISM1D sub-keys have reasonable defaults, the FLUIDPARAM sub-key deserves a special attention because its default values are only applicable if the solvent is water. Thus, you may need to change some of these values when modeling a different solvent, at least the dielectric constant and the density. Note that even when using all default values from the RISM1D sub-block the sub-block itself must be specified, even if empty. See below for complete description of the RISM1D sub-block.

The **SOLVENT** sub-block can be repeated if the solvent is a mixture. Each SOLVENT sub-block contains parameters for one solvent. First, each solvent has a name, which is specified on the SOLVENT keyword's line and is arbitrary. The first line in the SOLVENT sub-block must contain the UNITS key. You should leave it at the default values. Then follow the actual solvent parameters. In principle, each solvent consists of multiple atoms and functional groups. For simplicity, we will call each of them an atom. For example, in 3D-RISM therms, methanol consists of 3 "atoms": CH_3 , O, and H. Each such atom has a set of three parameters, shared between all atoms of the same type, and the coordinates. These parameters follow the PARAMETERS keyword. The line with the PARAMETERS keyword itself must specify the molecular weight of the solvent and the number of atom types that follow. The first line for each atom type contains, in this order: number of atoms of this type, z_α , σ_α , ϵ_α , three coordinates for the first atom of this type. If there is more than one atom of this type then the coordinates for the 2nd and other atoms follow on subsequent lines. The SOLVENT sub-block is concluded by the specific density of this solvent, by default, in molecules per cubic angstrom. This number should be equal to the total density for mono-component solvents.

For example, the SOLVENT block for water would typically look as:

```
SOLVENT water
  UNITS      uWeight=g/mol  ULJsize=A  ULJenergy=kcal/mol Ucoord=A Udens=1/A3
  PARAMETERS Weight=18.015   nAtoms=2
    1   -0.8476   3.166    0.1554      0.000000  0.00000  0.000000
    2    0.4238   1.000    0.0460     -0.816490  0.00000  0.577359
                                       0.816490  0.00000  0.577359
  DENSPE=0.03333
SUBEND
```

The **SOLUTE** sub-block specifies 3D-RISM parameters for your molecule. The BOXSIZE and BOXGRID sub-keys specify dimensions of the simulation box, in Angstrom, and the number of points of grid in each direction. The box

should be twice as large as the molecule and the BOXGRID values must be a power of 2. The size/np ratio defines the grid spacing in each direction and this should be not larger than 0.5 angstrom.

The **optional** 3D-RISM keys for the RISM1D and SOLUTE sub-blocks are listed below together with their defaults.

```
RISM1D Theory=RISM Closure=KH
! optional RISM1D subkeys with their default values:
  FLUIDPARAM temper=298. DielConst=78.497 UTotDens=1/A3 0.03333
  OUTPUT PrintLev=5 File=solvent OutList=GXT
  GRID   Nr=8192  dR=0.025 MaxRout=100.0 MaxKout=0.0
  MDIIS  N=20 Step=0.5 Tolerance=1.e-12
  ELSTAT LRsmear=1.0 Adbcor=0.5
  ITER   Ksave=-1 Kshow=1 Max=10000
SUBEND
```

OUTPUT PrintLev - print level;

>   File - common base name for output files;

>   OutList - which information will be written to output files: G - distribution function, C - direct correlation function, H - total correlation function, T - thermodynamics.

GRID Nr - the size of the 1D-RISM grid, must be a power of 2;

>   dR - mesh size in Angstrom;

>   MaxRout - plot range in direct space;

>   MaxKOut - plot range in reciprocal space.

MDIIS N - number of vectors in the DIIS space;

>   Step - step size;

>   Tolerance - convergence criterion.

ELSTAT LRsmear - smearing parameters for coulomb potential;

ITER Ksave - save the current solution every Ksave steps (0 - do not save);

>   Kshow - print convergence progress every Kshow steps;

>   Max - maximum number of iterations.

FLUIDPARAM Temper - temperature;

>   DielConst - dielectric constant;

>   UTotDens - units for total density: g/cm3, kg/m3, 1/A3 are valid units followed by the density value.

```
SOLUTE ArbitrarySoluteName
  outlist=M  closure=KH  xvvfile=solvent.xvv outfile=rism3d
  Nis=10  DELOZ=0.5 TOLOZ=2.0e-6
  Ksave=-1  Kshow=1  Maxste=10000
  Output=4
  CHRGLVL=MDCq
SUBEND
```

Outlist - output requested: G - distribution function, C - direct correlation function, H - total correlation function

Closure - closure for the 3D problem: KH – Kovalenko-Hirata, HNC - hypernetted chain approximations, PY - Perkus-Yevik

Xvvfile - name of the file with the results of the 1D-RISM calculation specified in the RISM1D keyword above, with .xvv appended to it;

Outfile - name of the output text file;

Output - print level;

CHRGLVL - which charges computed by ADF to use. This can be MDCq (default), MDCd, MDCm, or EXACT. The Nis, DELOZ, and TOLOZ have the same meaning for 3D-RISM as parameters of the MDIIS keyword of the RISM1D block. Likewise, Ksave, Kshow, and Maxste are analogous to the parameters of the ITER key in RISM1D.

Parameters for some solvents

Table 4.2: **Water Weight=18.015 nAtoms=3**

Atom	z_α / a.u.	σ_α / Å	ϵ_α / kcal/mol	X/Å	Y/Å	Z/Å
O	-0.8476	3.166	0.1554	-0.0646	0.0	0.0
H	0.4238	0.7	0.046	0.5127	0.8165	0.0
H	0.4238	0.7	0.046	0.5127	-0.8165	0.0

Table 4.3: **Methanol Weight=32.042 nAtoms=3**

Atom	z_α / a.u.	σ_α / Å	ϵ_α / kcal/mol	X/Å	Y/Å	Z/Å
CH3	0.265	3.775	0.207	-1.43	0.0	0.0
O	-0.7	3.07	0.17	0.0	0.0	0.0
H	0.435	0.7	0.046	0.2998	0.8961	0.0

Table 4.4: **Ethanol**

Atom	z_α / a.u.	σ_α / Å	ϵ_α / kcal/mol	X/Å	Y/Å	Z/Å
CH3	0.0	3.775	0.207	-1.9028	-1.4551	0.0
CH2	0.265	3.905	0.118	-1.43	0.0	0.0
O	-0.7	3.07	0.17	0.0	0.0	0.0
H	0.435	0.7	0.046	0.2998	0.8961	0.0

Table 4.5: **Acetonitrile Weight=41.0520 nAtoms=3**

Atom	z_α / a.u.	σ_α / Å	ϵ_α / kcal/mol	X/Å	Y/Å	Z/Å
CH3	0.269	3.6	0.38	1.46	0.0	0.0
C	0.129	3.4	0.099	0.0	0.0	0.0
N	-0.398	3.3	0.099	-1.17	0.0	0.0

Table 4.6: **Dimethylsulfoxide Weight=78.135 nAtoms=4**

Atom	z_α / a.u.	σ_α / Å	ϵ_α / kcal/mol	X/Å	Y/Å	Z/Å
CH3	0.160	3.81	0.16	0.0	1.7167	-0.5413
CH3	0.160	3.81	0.16	-1.6653	-0.4168	-0.5413
S	0.139	3.56	0.395	0.0	0.0	0.0
O	-0.459	2.93	0.28	0.0	0.0	1.53

Table 4.7: **Isopropanol Weight=60.09502 nAtoms=5**

Atom	z_α / a.u.	σ_α / Å	ϵ_α / kcal/mol	X/Å	Y/Å	Z/Å
O	-0.700	3.0700	0.1700	1.32393	0.14660	-0.01452
H	0.435	0.7000	0.0460	1.84501	0.01837	-0.79238
CH	0.265	3.8500	0.0800	0.84520	1.49406	-0.00612
CH3	0.000	3.7750	0.2070	1.38816	2.18026	1.24895
CH3	0.000	3.7750	0.2070	-0.68415	1.45191	-0.02015

Table 4.8: **Benzene**

Atom	z_α / a.u.	σ_α / Å	ϵ_α / kcal/mol	X/Å	Y/Å	Z/Å
C	-0.115	3.55	0.07	1.4	0.0	0.0
C	-0.115	3.55	0.07	0.7	-1.2124	0.0
C	-0.115	3.55	0.07	-0.7	-1.2124	0.0
C	-0.115	3.55	0.07	-1.4	0.0	0.0
C	-0.115	3.55	0.07	-0.7	1.2124	0.0
C	-0.115	3.55	0.07	0.7	1.2124	0.0
H	0.115	2.42	0.03	2.48	0.0	0.0
H	0.115	2.42	0.03	1.24	-2.1477	0.0
H	0.115	2.42	0.03	-1.24	-2.1477	0.0
H	0.115	2.42	0.03	-2.48	0.0	0.0
H	0.115	2.42	0.03	-1.24	2.1477	0.0
H	0.115	2.42	0.03	1.24	2.1477	0.0

OPLS-AA Parameters for common atoms and atom group

The table below contains sigma and epsilon parameters for some of the most common solvent groups collected from [*433, 434, 435* (page 932)]. These parameters are kindly provided by Leonardo Costa.

Table 4.9: **All atoms model**

Atom	Example	σ_α / Å	ϵ_α / kcal/mol	Reference
C (SP3)	methane	3.73	0.294	433
C (SP3)	neopentane	3.80	0.050	433
C (SP2)	isobutene	3.75	0.105	433
C (SP)	acetonitrile	3.40	0.099	434
C (arom)	benzene	3.55	0.07	434
C	chloroform	4.10	0.05	435
H	O—H	0.7	0.046	434
H	hydrocarbons	2.42	0.03	434
O	water	3.166	0.1554	434
O	alcohol	3.07	0.17	434
O	sulfoxide	2.93	0.28	434
N (SP)	acetonitrile	3.3	0.099	434
S	sulfoxide	3.56	0.395	434
Cl	chloroform	3.40	0.300	435

Table 4.10: **United atom model**

Atom	Example	σ_α / Å	ϵ_α / kcal/mol	Reference
CH (SP3)	isobutane	3.850	0.080	433
CH (SP2)	2-butene	3.800	0.115	433
CH (arom)	benzene	3.750	0.110	433
CH2 (SP3)	n-butane	3.905	0.118	433
CH2 (SP2)	1-butene	3.850	0.140	433
CH3	hydrocarbon	3.775	0.207	433
CH3	acetonitrile	3.6	0.38	434
CH3	sulfoxide	3.81	0.16	434
NH2	amine	3.3	0.17	434

4.5 Electric Field: Homogeneous, Point Charges, Polarizability

A homogeneous external electric field and/or the field due to point charges can be included in the Fock operator. Either can be applied in both a Single-Point calculation (or a Create run) or in geometry optimization. When applied in geometry optimization, it will allow for the molecule to rotate with respect to point charges or the field vector but not translate. Rigid translation is explicitly disabled to avoid drifting of the molecule in the external electric field.

Homogeneous electric field:

```
EField ex ey ez
```

ex,ey,ez Define a homogeneous electric field in atomic units: Hartree/(e bohr) = 27.211 V/bohr; the relation to SI units is: 1 Hartree/(e bohr) = 5.14 ... * 10^{11} V/m. The units applied by ADF for the interpretation of homogeneous field values are not affected by any units used for specifying atomic coordinates. By default no homogeneous E-field is included.

Point charges:

```
PointCharges
   x y z q
   x y z q
   ...
End
```

x, y, z, q The Cartesian coordinates and strength of a point charge (in elementary charge units, +1 for a proton). Each point charge must be specified on a separate line in the data block. The Cartesian coordinates are in the units defined in the Units block (by default, Angstrom). By default no point charges are included.

Orientation of the fields

When the atomic coordinates are input in z-matrix format, the direction of the homogeneous field and the location of the point charges as specified in input are interpreted as referring to the standard Cartesian frame associated with z-matrix input. The standard frame means: the first atom at the origin, the second on the positive x-axis, the third in the xy-plane with positive y.

If the program rotates (and translates, as the case might be) the atoms from the input frame - or the auto-generated frame in case of z-matrix input - to some other frame, for instance to accommodate the internal ADF symmetry orientation requirements, the fields are transformed along with the atoms.

Symmetry

The homogeneous electric field and the point charge fields may polarize the electronic charge density. This must be accounted for in the point group symmetry. If symmetry is not specified in input, the program computes the symmetry from the nuclear frame and the fields.

Bonding energy

The bonding energy is computed as: the energy of the molecule in the field minus the energy of the constituent fragments in the same field. Of course, the fragments may not be polarized and hence not be self-consistent in this field. This depends on how the fragments themselves were computed.

Polarizability and hyperpolarizability

ADF supports a direct calculation of the (hyper) polarizability (see section on Spectroscopic Properties). The static (hyper) polarizabilities can also be computed by applying a small homogeneous field and comparing the results with the field-free data.

STRUCTURE AND REACTIVITY

See also:

ADF-GUI tutorials: geometry optimization, vibrational frequencies, HCN isomerization reaction, spin coupling in Fe4S4

Examples: *geometry optimization* (page 557), *reactivity* (page 571)

5.1 Run Types

The different *run types* are characterized by how the geometry is manipulated:

SinglePoint The SCF solution is computed for the input geometry.

GeometryOptimization **(page 120)** The atomic coordinates are varied in an attempt to find a (local) energy minimum. One may let all coordinates free or only a subset, keeping the others frozen at their initial values.

TransitionState **(page 125)** Search for a saddle point. Similar to a GeometryOptimization, but now the Hessian at the stationary point presumably has one negative eigenvalue.

LinearTransit **(page 127)** The geometry is modified step by step from an initial to a final configuration. All of the coordinates or only a subset of them may be involved in the transit. The coordinates to be modified are the LinearTransit *parameters*. The LinearTransit feature can be used for instance to sketch an approximate reaction path in order to obtain a reasonable guess for a transition state, from where a true TransitionState search can be started.

IRC or IntrinsicReactionCoordinate **(page 129)** Tracing a reaction path from a transition state to reactants and/or products. A fair approximation of the transition state must be input. The end-point(s) - reactants / products - are determined automatically.

Frequencies **(page 139)** Calculation of normal vibrational modes and harmonic frequencies via numerical differentiation. It is also possible to compute the harmonic frequencies *analytically* (page 140).

CINEB **(page 131)** Calculation of the reaction path and transition state search using Climbing-Image Nudged Elastic Band method. This method is further referrer to as NEB or CI-NEB. Using this method one can find a transition state between two known states, further referred to as initial and final states. The choice which state is initial and which is final is arbitrary. During calculation with this method, a number of replicas, or images, of the system is calculated. These images can be considered as being linked by an elastic band. Each image is optimized in such a way that on each step the forces parallel to the reaction path are removed and spring forces are added that keep distances to this image's neighbors equal. At the end of the optimization the images are evenly distributed along the reaction path, the image highest in energy being the transition state (if the climbing-image option is on, the default).

For all features that involve changes in geometry, i.e. all run types except the SinglePoint, it is imperative that you use **single-atom fragments**. Larger molecular fragments can only be applied in SinglePoint calculations.

Four keys are involved in the specification of the geometry and its manipulation:

Atoms **(page 25)** Sets the atomic (starting) positions.

Geometry **(page 120)** Controls the run type and strategy parameters, such as convergence thresholds and the maximum number of geometry steps to carry out.

Constraints **(page 133)** May be used to impose constraints for geometry optimizations, LT, and TS, in the new branch for optimization, which is the default for these optimizations. This key can not be used for IRC, NEB, or for the old branch of optimizations.

Geovar **(page 135)** May be used to impose constraints, for instance when only a subset of all coordinates should be optimized. This key should be used for IRC, NEB, or for the old branch of optimizations, to impose constraints. GeoVar may also be used in a LinearTransit or NEB run to define the LinearTransit or NEB parameters, respectively, and their initial and final values.

Constraints and LinearTransit parameters in the old branch of optimizations may also be controlled within the atoms block if a MOPAC-style input format is used, see below.

5.1.1 RunType control

With the block key GEOMETRY you define the runtype and strategy parameters.

```
GEOMETRY {RunType {RunTypeData}}
  RunType {RunTypeData}
End
```

RunType Can be:

- SinglePoint or SP

- GeometryOptimization or GeoOpt or GO

- TransitionState or TS

- IntrinsicReactionCoordinate or IRC

- LinearTransit or LT

- Frequencies or FREQ

- CINEB

If omitted the run type is GeometryOptimization. If the key GEOMETRY is not used at all the run type is SinglePoint. The run type specification can be given as argument to the geometry key, or in the data block, but not both. For some run types additional data may be given after the run type specification.

RunTypeData (Optional) further specifications, depending on the run type. See the following sections.

Omission of the GEOMETRY key altogether effectuates a SinglePoint calculation. A straightforward optimization, with all features that can be set with geometry at their default values, is activated by supplying the key with an empty block:

```
GEOMETRY
End
```

More subkeys are available in the geometry block than just the run type specification. They are used to control strategy parameters such as convergence criteria. All subkeys are optional: default values take effect for those omitted. Some of the subkeys are only meaningful for certain run types. They will be ignored for other run types.

The initial approximation of the Hessian matrix may affect the number of optimization steps that are carried out to reach convergence. See the section *Initial Hessian* (page 133).

5.2 Geometry Optimization

The Geometry Optimizations procedure in ADF is based on a quasi Newton approach [*6-8* (page 907)], using the Hessian for computing changes in the geometry so as to reach a local minimum. The Hessian itself is *initialized* (page 133) and updated in the process of optimization. The default optimization strategy uses delocalized coordinates and it is mainly based on the work by Marcel Swart [*225* (page 920)].

See also:

In case of geometry convergence problems, see *Geometry Optimization troubleshooting* (page 335)

You can perform a Geometry Optimization using default settings by simply specifying the GEOMETRY key block:

```
GEOMETRY
End
```

Several subkeys in the GEOMETRY block can be used for control of the Geometry Optimization procedure and related parameters.

```
GEOMETRY
   Converge {E=TolE} {Grad=TolG} {Rad=TolR} {Angle=TolA}
   Iterations Niter
   Inithess inithessian.file
   Hessupd HessUpdate
   Optim {Delocal / Cartesian / Internal} {All / Selected}
   GEigenvalueThreshold threshold
   Branch {New / Old}
   Step {Rad=MaxRadStep} {Angle=MaxAngleStep} {TrustRadius=MaxRadius}
   DIIS {N=NVect} {CYC=Ncyc}
   Externprogram externprog.exe coord=coords.inp energy=energy.out grad=grads.out
End
```

5.2.1 Convergence

A geometry optimization is considered *converged* when all the following criteria are met (this information is also printed to the *logfile* (page 21)):

1. the difference between the bond energy at the current geometry and at the previous geometry step is smaller than TolE

2. the maximum Cartesian nuclear gradient is smaller than TolG

3. the root mean square (RMS) of the Cartesian nuclear gradients is smaller than $\frac{2}{3}$TolG

4. the maximum Cartesian step is smaller than TolR

5. the root mean square (RMS) of the Cartesian steps is smaller than $\frac{2}{3}$TolR

note: if the maximum and RMS gradients are 10 times smaller than the convergence criterion, then criteria 4 and 5 are ignored

The convergence criteria can be changed via the Converge key:

```
GEOMETRY
   Converge {E=TolE} {Grad=TolG} {Rad=TolR} {Angle=TolA}
END
```

TolE The criterion for changes in the energy, in Hartree. **Default: 1e-3**.

TolG The criterion for the nuclear gradients, in Hartree/Angstrom. **Default: 1e-3**.

TolR Refers to changes in the Cartesian coordinates or bond lengths, depending on in what coordinates you optimize, in angstrom. **Default: 1e-2**.

TolA Refers to changes in bond- and dihedral angles, in degrees. This is only meaningful if optimization takes place in Z-matrix coordinates. **Default: 0.5 degree**.

Remarks:

- Molecules may differ very much in the stiffness around the energy minimum. Using the standard convergence thresholds without second thought is therefore not recommended. Strict criteria may require a large number of steps, a loose threshold may yield geometries that are far from the minimum as regards atom-atom distances, bond-angles etc. even when the total energy of the molecule might be very close to the value at the minimum. It is good practice to consider first what the objectives of the calculation are. The default settings in ADF are intended to be reasonable for most applications, but inevitably situations may arise where they are inadequate.

- The technical numerical accuracy of the calculation (e.g. *numerical integration* (page 300)) should somehow match the required level of convergence in gradients (strict convergence criteria may require high numerical accuracy). A simple way of changing the numerical accuracy is through the *NumericalQuality* (page 295) keyword.

- The convergence threshold for the coordinates (`TolL`, `TolA`) is *not a* reliable measure for the precision of the final coordinates. Usually it yields a reasonable estimate (order of magnitude), but to get accurate results one should tighten the criterion on the gradients, rather than on the steps (coordinates). The reason for this is that the program-estimated uncertainty in the coordinates is related to the used Hessian, which is updated during the optimization. Quite often it stays rather far from an accurate representation of the true Hessian. This does usually not prevent the program from converging nicely, but it does imply a possibly incorrect calculation of the uncertainty in the coordinates.

Iterations

```
GEOMETRY
   Iterations Niter
END
```

Iterations

Niter The maximum number of geometry iterations allowed to locate the desired structure. The default is max(30,2*Nfree), where Nfree is the number of free variables, which is typically close to 3*N(atoms). This is a fairly large number. If the geometry has not converged (at least to a reasonable extent) within that many iterations you should sit down and consider the underlying cause rather than simply increase the allowed number of cycles and try again.

5.2.2 Optimization strategy

Optim

```
GEOMETRY
   Optim {Delocal / Cartesian / Internal} {All / Selected}
   GEigenvalueThreshold threshold
END
```

Optim

Delocal Optimization in **delocalized coordinates** [*225* (page 920)] (Delocal) is the **default** in geometry optimizations, transition state searches, and (linear) transits. You can use constraints (see the *CONSTRAINTS key* (page 133)) and restraints, and you can supply the atoms in Z-matrix coordinates. You can not use dummy atoms, ghost atoms, or alternative elements.

Cartesian / Internal Cartesian or Zmatrix (equivalently: internal) specifies the type of coordinates in which the minimization is carried out. By default the coordinate type is applied that was used in the ATOMS key for the input of the (initial) atomic positions. (Cartesian if atoms were input in zcart format). Note that Zmatrix optimizations can only be done with the old branch of optimizations.

Cartesian optimization is allowed if the atoms were input in Z-matrix format, but no constraints (see the key GEOVAR) can then be used: *all* coordinates are optimized. An attempt to explicitly freeze variables may result in an error abort. Optimization in Z-matrix coordinates is not allowed if only Cartesian were supplied in atoms: the program does not construct a Z-matrix by itself. One should then use the zcart format: give Cartesian coordinates and supply the structure of the Z-matrix. Again, in this case you cannot use constraints.

Selected For use with old branch of optimizations. Only those coordinates are optimized that are defined with the key GEOVAR.

All (The default value) means that in principle all atomic coordinates will be varied. With the new branch of optimization the key CONSTRAINTS one may set constraints on the optimization. With the old branch of optimization one can use the key GEOVAR to set constraints.

GEigenvalueThreshold

threshold When performing optimization in delocalized coordinates a singular value decomposition of the Wilson B-matrix is obtained. Left-eigenvectors of the B-matrix corresponding to non-zero singular values are used to determine the delocalized coordinates. 'Non-zero' in this case means larger than sqrt(threshold). The square root is taken because left-eigenvectors of the B-matrix correspond to eigenvectors of the $G = B\,B^t$ symmetric matrix and the singular values of B are related to eigenvalues of G as $\lambda_i(G) = \sigma_i(B)^2$. The default value is 10^{-15}. Note: in rare cases, ADF may generate a "Suspicious delocalized coordinates" warning. In such a case, check the output file for more details. For non-linear molecules there should be no more than 3N-6 eigenvalues above the threshold. There may be fewer than that if block constraints are used. If the number of delocalized coordinates is higher than 3N-6 then it may be useful to increase the threshold until the warning disappears. Check the "G-matrix eigenvalues" table in the output file for a suitable value.

Branch

```
GEOMETRY
 Branch {New / Old}
END
```

Branch

Old / New Expert option. Specifies which branch of the code to use for making steps. Default the branch of code used depends on the optimization used. Optimization in delocalized coordinates can only be done with the new branch. Optimization in Z-matrix coordinates can only be done with the old branch. In case of Cartesian optimization default the new branch is used, but the old branch can also be used. The new branch can be used (and is default) in geometry optimizations, transition state searches, and in LT. The old branch is default in IRC and NEB, for which one can not use the new branch.

Inithess

```
GEOMETRY
   Inithess inithessian.file
END
```

Inithess Can only be used in combination with the new branch. With this INITHESS subkey it is possible to read a Hessian from a text files. The only argument is the name of the file containing the initial Hessian. The Hessian must be given in full, in non-mass-weighted Cartesian coordinates, and in atomic units (Hartree/Bohr^2).

Hessupd

```
GEOMETRY
 Hessupd HessUpdate
END
```

Hessupd

> **HessUpdate** Specifies how the Hessian matrix is updated, using the gradient values of the current and the previous geometry. The methods available depend on the optimization branch being used. For both the old and new branches, the following options are available: (i) BFGS : Broyden-Fletcher-Goldfarb-Shanno (ii) MS : Murtagh-Sargent (iii) FARKAS : Farkas-Schlegel, Eq. (15) and (16) of Ref. [*139* (page 915)] (iv) FARKAS-BOFILL : Farkas-Schlegel-Bofill, Eq. (15) and (14) of Ref. [*139* (page 915)] In the old branch, the following extra options are available: (i) DFP : Davidon-Fletcher-Powell (ii) FS : Fletcher switch (iii) HOSHINO : Hoshino In the new branch, the following extra option is available: (i) BAKKEN-HELGAKER: Bakken-Helgaker, see Ref. [*219* (page 919)] The default is BFGS for geometry optimizations.

Step

```
GEOMETRY
 Step {Rad=MaxRadStep} {Angle=MaxAngleStep} {TrustRadius=MaxRadius}
END
```

Step Controls that changes in geometry from one cycle to another are not too large:

> **MaxRadStep** Can only be used in combination with the old branch. An upper bound on changes in Cartesian coordinates or bond lengths, as the case may be. Default: 0.3 angstrom when optimization is carried out in internal coordinates, 0.15 angstrom for Cartesian optimizations.

> **MaxAngleStep** Can only be used in combination with the old branch. Similarly this option limits changes in bond angles and dihedral angles. Default: 10 degrees. Input for MaxRadStep, MaxAngleStep is in angstrom and degrees respectively, independently of the units used for atomic coordinates input.
>
> Note: Optimization of ring structures carried out in internal (z-matrix) coordinates is sometimes tricky due to the ill-defined last segment of the ring. When problems arise, try Cartesian optimization or consider using smaller limits on the steps (in particular the angles) so as to prevent the program from breaking the ring beyond repair.

> **MaxRadius** Can only be used in combination with the new branch. By default, the trust radius is set to 0.01 Bohr times the number of atoms with a minimum of 0.2 Bohr. Using the key, the user can override this, setting a constant value. A conservative value is 0.2. A large system (eg 100 atoms) typically needs a larger trust radius (eg 0.8).

DIIS

```
GEOMETRY
 DIIS {N=NVect} {CYC=Ncyc}
END
```

DIIS N=NVect CYC=Ncyc Can only be used in combination with the new branch. NVect is the number of vectors used by the DIIS interpolation method. NCYC is the number of geometry cycles run before the DIIS starts to modify the geometry steps. Default DIIS is used and default N=5 and CYC=0.

Externprogram

```
GEOMETRY
 Externprogram externprog.exe coord=coords.inp energy=energy.out grad=grads.out
END
```

Externprogram Expert option, can only be used in combination with the new branch. Subkey EXTERNPRO-GRAM has been added to to allow energies and gradients to be calculated by an external program for use in a geometry optimization.

Note that you need to supply information about atomic fragments, such as the basis set, even though these are not actually used in the calculations.

When ADF is ready to perform an energy and gradient calculation, it writes the current cartesian coordinates to the file name given in the input. The format is similar to the ATOMS block in the ADF input file: it has one atom per line, with the element symbol given, followed by the x, y, and z coordinates.

ADF will then run the executable program, and then read in the energy and gradients from the file names given in the input file. The external program is thus responsible for reading the coordinates (in atomic units) written by ADF from file, generating the corresponding energy and gradients (in atomic units), and writing these to the appropriate files. ADF will then take another geometry step, and the process will repeat.

5.3 Transition State

A transition state (TS) search is very much like a minimization: the purpose is to find a stationary point on the energy surface, primarily by monitoring the energy gradients, which should vanish. The difference between a transition state and a (local) minimum is that at the transition state the Hessian has a negative eigenvalue.

Because of the similarities between a minimization and a TS search most subkeys in geometry are applicable in both cases, see the Geometry Optimization section. However, practice shows that transition states are much harder to compute than a minimum. For a large part this is due to the much stronger anharmonicities that usually occur near the ts, which threaten to invalidate the quasi-Newton methods to find the stationary point. For this reason it is good advice to be more cautious in the optimization strategy when approaching a Transition State and for some subkeys the default settings are indeed different from those for a simple optimization. In addition, certain additional aspects have to be addressed.

```
GEOMETRY
  TransitionState
    {Mode Mode}
    {NegHess=NegHess}
  End
End
```

NegHess The number of negative eigenvalues that the Hessian should have at the saddle point. In the current release it is a rather meaningless key, which should retain its default value (1).

Mode Controls the first step from the starting geometry towards the saddle point: it specifies in which direction the energy is to be *maximized* while the optimization coordinates will otherwise be varied so as to *minimize* the energy. A positive value means that the eigenvector #mode of the (initial) Hessian will be taken for the maximization direction. This means: put all Hessian eigenvalues in ascending order, ignoring those that correspond to impossible movements (rigid rotations and translations, symmetry breaking) and then take the eigenvector of #mode in the remaining list.

Default: mode=1. Generally the program performs best with this default: it will simply concentrates on the mode with the lowest eigenvalue, which should of course finally be the path over the transition state (negative eigenvalue).

After the first geometry step, the subsequent steps will attempt to maximize along the eigenvector that resembles most (by overlap) the previous maximization direction.

As mentioned before, the other subkeys have the same functionality as for minimizations, but different defaults or options may apply:

Hessupd

> **HessianUpdate** Different (fewer) options apply now. The methods available depend on the optimization branch being used. For both the old and new branches, the following options are available:
>
> > 1. Powell: Powell
> >
> > 2. BFGS: Broyden-Fletcher-Goldfarb-Shanno
> >
> > 3. BOFILL : Bofill, Eq. (13) and (14) of Ref. [*139* (page 915)]
> >
> > 4. MS : Murtagh-Sargent In the old branch, the following extra option is available: (i) DFP : Davidon-Fletcher-Powell The default is Powell for transition state searches.

Step

> **MaxRadStep** Default: 0.2 angstrom for Z-matrix optimization, 0.1 angstrom for Cartesian optimization. Not used in the new branch.
>
> **MaxAngleStep** Default: 5 degrees. Not used in the new branch.

Note: in Transition State searches precision is often much more critical than in minimizations. One might consider using NumericalQuality Good or better.

5.3.1 Transition State Reaction Coordinate (TSRC)

A reaction coordinate for transition state search can be specified via the TSRC input block, similar to QUILD. The principle is very similar to the eigenvector-following (EF) method. In the EF method, the specified Hessian eigenvector is stored on the 1st iteration. On later iterations it is compared with eigenvectors of the updated Hessian. The eigenvector that matches the stored one the best is selected as the reaction coordinate and the step is made in the up-hill direction along this vector and down-hill along all other vectors. With TSRC, the specified TSRC vector is used instead of a Hessian eigenvector.

This feature is especially useful when the accurate Hessian is not available. In such a case ADF uses an approximate Hessian that can be poor when weak interactions and/or transition metals are involved. What then happens is that the mode with the lowest Hessian eigenvalue does not correspond to the reaction coordinate along which transition state is searched for, thus leading the optimization in the wrong direction.

This problem can now be solved by specifying a reaction coordinate along which the transition state should be searched for. Such a reaction coordinate can consist of one or more distance, valence or dihedral angle, or just a combination of vectors on certain atoms.

```
TSRC
  {ATOM i x y z {fac}}
  {DIST i j {fac}}
  {ANGLE i j k {fac}}
  {DIHED i j k l {fac}}
end
```

i, j, k, l, x, y, z, fac Here, i, j, k, and l are atom indices, x, y, and z are corresponding components of a TSRC vector for atom i, and fac is the factor (and thus also the sign) of a particular coordinate in the TSRC. By default $fac = 1.0$.

Restrictions and notes

The TSRC feature does **not** work in combination with the old optimization branch. In general, the old branch is no longer developed so all new features related to geometry optimization work with the new branch only. The DIST, ANGLE and DIHED specifications should be used in combination with optimization in delocalized coordinates only (i.e. not with Cartesian).

Only one type of the keyword is allowed in a TSRC block. That is, the keys must be either all ATOM or all DIST, etc. Thus, mixing of different keywords is not allowed.

One should be careful when specifying more than one bond or angle as a TSRC. For example, suppose atom 2 is located between atoms 1 and 3. Then the following TSRC block:

```
TSRC
  DIST 1 2  1.0
  DIST 3 2 -1.0
END
```

means that the TSRC consists of two distances: R(1-2) and R(2-3). The positive direction of the TSRC is defined as increase of the distance R(1-2) and decrease of the distance R(2-3), which, in turn, means that this TSRC corresponds to atom 2 is moving along the R(1-3) axis.

5.4 Linear Transit

In a Linear Transit (LT) run you define a number of atomic coordinates (at least one) to be the LT *parameters*: these get an initial and a final value. The LT is defined as the simultaneous linear change of these parameters from their initial to their final values. This is carried out in a number of equidistant steps. In a non-linear transit calculation these may be not equidistant steps. The total number of LT *points* is specified on input. At each LT point the remaining atomic coordinates - those that are not LT parameters - may or may not be optimized: the (final) structure and energy at each LT point are computed. A Linear Transit (LT) run is therefore just a sequence of (related) constrained Geometry Optimizations.

The LT scan may be used for instance to sketch an approximate path over the transition states between reactants and products. From this a reasonable guess for the Transition State can be obtained which may serve as starting point for a true transition state search for instance.

Whenever a geometry subkey is applicable in a Geometry Optimization, it will apply in a Linear Transit run in each of the optimizations that are carried out at the distinct Linear Transit points, and the same default values apply.

5.4.1 Linear Transit (new branch)

A transit calculation option has been added in the new optimization branch. This is capable of performing both linear transits, and non-linear transits, and is the default when the LINEARTRANSIT or TRANSIT subkey is included in the GEOMETRY block key.

The new transit code works differently to the old: the transit is represented as a sequence of constrained optimizations. The CONSTRAINTS block key should be used to delineate the constraints applied at each stage of the transit.

To perform a linear transit, start and end values are supplied.

```
Constraints
  angle 2 1 3 start=100.0 end=120.0
End

Geometry
  Transit 4
  Optim Deloc
End
```

In the example above, 4 stages are required; ADF will interpolate the start and end values supplied for the angle between atoms 2, 1, and 3. Note that TRANSIT can now be used in place of LINEARTRANSIT, due to the more general nature of the new transit calculations.

Non-linear transits are possible, and can even be combined with linear transits in other coordinates. To perform a non-linear transit in a particular coordinate, explicit values must be given.

```
Constraints
  dist 1 2 0.8 0.9 1.1 1.15
  angle 2 1 3 start=100.0 end=120.0
End

Geometry
  Transit 4
  Optim Deloc
End
```

In the example above, 4 values are given for the distance between atoms 1 and 2. This distance constraint will be applied simultaneously with the linear transit constraints for the angle, with other degrees of freedom optimized at each stage of the transit.

It is worth noting that fixed constraints can also be used in a transit.

```
Constraints
  dist 1 2 0.8 0.9 1.1 1.15
  angle 2 1 3 100.0
End

Geometry
  Transit 4
  Optim Deloc
End
```

In this example, the angle between atoms 2, 1, and 3 will be fixed at 100.0 degrees at all stages of the transit.

Finally, it should be pointed out that fully converged constraints are used by default in the transit calculations. They do not have to be fully met in the input, but if the input geometry is far from meeting the constraints, a large, erratic first geometry step may result.

You can avoid fully enforcing constraints, by adding a CONSTRAINTS subkey to the geometry block key:

```
Geometry
  CONSTRAINTS partialconverge
End
```

In this case the constraints are not required to be fully met at each intermediate geometry, but are fully met at the converged geometries,

5.4.2 Linear Transit (old branch)

The LINEARTRANSIT runtype has to be specified. Additional specifications are optional.

```
GEOMETRY
 Branch Old
 LinearTransit {NPoints}
end
```

NPoints The number of LT points for which an optimization will be carried out If no value is supplied the default takes effect: 5.

There are a few obvious differences between a single optimization and a LT run. Most important is that the coordinate(s) that describe the LT path, the LT *parameters*, cannot be optimized: at each of the LT points they are frozen.

This implies that technically speaking at each LT point a *constrained* optimization is carried out. One of the consequences is that the atoms coordinate type - Cartesian or Z-matrix - must also be the optimization coordinate type. The LT parameters themselves must be defined with the key GEOVAR, see below.

It is possible to freeze *all* coordinates so that the LT run is similar to a sequence of Single Point runs. However, energy gradients will be computed at each step, so that more CPU time is spent at each LT point than for just a Single Point calculation.

The number of LT points by which the path is traced is defined by the npoints argument to the subkey LinearTransit. It is possible to execute only a subset of these points, usually with the purpose to complete the calculation by using the restart facility of ADF. In this way you can break down a very large calculation into several smaller ones, or have the opportunity to check how things have been going for the first few LT points before deciding whether a continuation is useful. This may be achieved of course by simply defining different start- and end-values for the LT parameters in a related series of calculations, but it is more comfortable to specify the complete path once and just execute parts of it at a time. This is accomplished by giving a *second* value to the *iterations* subkey in the geometry block.

```
iterations Niter Niter2
```

Niter The first argument of the subkey iterations in the GEOMETRY block, controlling the maximum number of iterations allowed to reach convergence, applies now for each LT point separately.

Niter2 The second argument specifies the maximum number of LT *points* to calculate in this run. If omitted (default) the whole LT scan is completed. Doing only part of the scan may be combined with the restart feature, so that the remainder can be done in a continuation run. See the restart key. A too large value of LT points is automatically adjusted: no more LT points are computed than required to complete the LT path as defined by the lineartransit subkey. A negative or zero value is not accepted and internally reset to one (1).

WARNING: if you use the QMMM functionality in combination with a Linear Transit, then only the coordinates of the true QM atoms can be used as LT parameters, no MM atoms must be involved in the LT parameter set.

5.4.3 Symmetry in a Linear Transit

In a Linear Transit run it is imperative that the complete Linear Transit path as defined by the parameters conforms to the specified symmetry. If such is not the case, an error will occur or possibly the program will continue but not produce correct results. Note that when no symmetry is specified in input, the initial geometry defines the specified symmetry.

5.5 Intrinsic Reaction Coordinate

The path of a chemical reaction can be traced from the Transition State to the products and/or to the reactants, using the Intrinsic Reaction Coordinate method (IRC) [*9* (page 907), *10* (page 907)]. The starting coordinates should be a fair approximation of the Transition State. The final values at the endpoint(s) - reactants, products - are computed. The IRC path is defined as the steepest-descent path from the Transition State down to the local energy minimum. The energy profile is obtained as well as length and curvature properties of the path, providing the basic quantities for an analysis of the reaction path. Additional properties along the path (dipole moment, atomic charges) are computed.

Technically speaking the path is computed by taking small steps along the path meanwhile optimizing all atomic coordinates *orthogonal to it* so that, like in a Linear Transit run, a sequence of constrained optimizations is carried out. The total number of steps along the path is not known in advance. The maximum number of such steps can be set in input. If the path is not completed in the run, a Restart can be used to finish it. Each of the constrained optimizations in the run is treated as it would be in a Linear Transit run: convergence thresholds, maximum numbers of optimization iterations et cetera are set with subkeys in the geometry block.

You can set the IRC runtype by typing it in the geometry block

```
GEOMETRY
 IRC {Forward} {Backward} {Points=Points} {Step=Step}
     {StepMax=StepMax} {StepMin=StepMin} {Start=Start}
End
```

IRC The runtype IntrinsicReactionCoordinate would also be recognized.

Forward, Backward Specifies execution of the two possible paths from the Transition State to the adjacent local minima. By default both are computed. If Forward is specified only, the other path is turned off and similarly for Backward. For the definition of which of the two directions down from the Transition State to an adjacent minimum is 'forward' see below.

Points The maximum number of IRC points computed in the run, for both paths together and including the initial (TS) central point (as far as applicable). Default 100.

Step The (initial) step length when proceeding from one IRC point to another along the path. The difference between two geometries, to which the step quantity applies, is measured in mass-weighted coordinates. The default value for step is 0.2 $(amu)^{1/2}$ bohr. Larger steps reduce, in principle, the required number of IRC points from the transition state to the minimum, but usually at the expense of more optimization steps at each of the points so the net gain in computation time may not be very large, or even negative. The default size is rather conservative and in many cases you may increase it to save a few steps. However, to some extent you can leave that to the program. When going from one point to the next, the program will increase or decrease the step size depending on whether or not the previous point to a large number of geometry cycles to converge. The adjusting algorithm also tends to be more cautious when the successive IRC points show more drastic changes in the atomic geometrical configuration. In all cases the IRC step sizes remain between pre-set maximum and minimum values, see the next items.

StepMax The maximum step length that the program will select in the step-adjusting algorithm. Default: 1.0 or 10 times the initial step length, whichever is larger.

StepMin The minimum step length that the program will select in the step-adjusting algorithm. Default: 0.03 or 0.3 times the initial step length, whichever is smaller.

Start Defines how the initial direction of the path is chosen to move away from the Transition State. It does *not* imply whether the first step along this direction is taken positively or negatively. See for this aspect the section about Forward/Backward IRC paths. The admissible values for start are: *Grad*: compute the gradient and take that direction right from the start. Obviously, if we start perfectly at the Transition State this will be meaningless since the gradient vanishes there completely. *Read* : the initial path direction is read in with the key IRCstart, see the section IRC Start Direction. *Hess n* : the initial path coincides with the n-th Hessian eigenvector (ordered by ascending eigenvalues); *n* must be an integer in the appropriate range. The default (omission of any start specification at all) is the first Hessian eigenvector, presumably corresponding to the path over the Transition State (negative Hessian eigenvalue!).

5.5.1 IRC start direction

As mentioned above, the IRC path is initialized by a first step away from the Transition State. If perfect information is available this should be along the unique Hessian eigenvector with a negative eigenvalue. Therefore, it is preferable to supply (with a restart file) a good approximation of the Hessian at the Transition State. This can be computed in a Frequencies run. In many cases the automatic internally generated (force field based) Hessian will not severely disturb the procedure and may only require a few more initial search steps for the right direction to take, while saving a potentially expensive Frequencies calculation.

If you decide to use a precalculated Hessian, then usually the approximate Hessian resulting from a Transition State run will be good enough. The latter approach is more attractive of course since the TS run will usually be done anyway, as a preliminary to the IRC run, while an additional Frequencies run would be very demanding. At the other hand, Transition State runs often require a preceding Frequencies run. In such case, the Frequencies result file may be used

both for the TS run and for the IRC run. The fact that the Frequencies run may have been performed not at the exact TS may affect slightly the adequacy for using it as a start-up for the IRC run, but this is likely not significant.**

In some case you may want to specify the initial direction of the IRC path explicitly. This is done as follows:

```
IRCSTART
 data
 data
 ...
end
```

IRCstart A block-type key. The data in the data block are values for *all* atomic coordinates (Cartesian or Z-matrix, as the case may be) that are not frozen and not (by geovar) explicitly instructed to remain equal. All such coordinate data together define a direction vector in the space of all (free) coordinates, which then serves as the initial segment of the IRC path.

Note that only a direction vector is defined here: the *size* of the total vector plays no role.

Furthermore, the initial step may be in the positive or negative direction along the so-defined initial path, see the section Forward / Backward IRC paths.

5.5.2 Forward / Backward IRC paths

Obviously there are two IRC paths down the transition state: Forward and Backward. We would have liked to chemically define forward and backward by determining (in advance!) which of the endpoints is reactants and which products. This is not well doable in practice. Therefore we define the directions in terms of the initial path vector: select simply the atomic coordinate with the largest (absolute) change in the initial vector and define *Forward* as the direction in which this coordinate *increases* and *Backward* as the direction in which it *decreases*.

5.6 Climbing-Image Nudged Elastic Band

The reaction path can be found by simultaneous optimization of a number of replicas of the system in question starting from some rough approximation [*159* (page 916)]. In the simplest case, implemented in ADF, the initial approximation is just a polynomial interpolation between initial and final states (see keyword geovar). The images are optimized not independently of each other but, in fact, forces on each image depend on its neighbors. At each step the forces parallel to the reaction path are eliminated and a so-called spring force is added which keeps the image in the middle between its neighbors. This does not let images slide to the initial or final reaction state and ensures that they are evenly distributed along the reaction path. There are also options to distribute images more densely near the transition state (energy-dependent spring force). Below is the list of NEB options:

```
GEOMETRY
  CINEB {NumImages}
  {NEBSPRING  Nspring  Spring Spring2 Spower}
  {NEBOPT OptMethod}
  {NEBECONO}
  {NOCLIMB}
  {NONEBOPTENDS}
End
```

CINEB The runtype. Nudged will also be recognized.

NumImages The number of NEB images excluding initial and final stated. The default is 8.

NEBSPRING Nspring Spring Spring2 Spower Nspring determines the type of spring used, which, in turn, determines which of the spring parameters are used:

1. constant spring, spring=Spring

2. exponential scaling, spring = Spring+Spring2*exp((dE-dEmax)**Spower)

3. power scaling, spring = Spring+Spring2*(dE/dEmax)**Spower

4. another exponential with different meaning of Spower, spring = Spring+Spring2*exp((dE-dEmax)*Spower)

5. another exponential scaling very close to #4, spring = Spring+Spring2*(2**(dE-dEmax))**Spower)

Units for Spring and Spring2 are Hartree/bohr. Default values, when NEBSPRING is not present in the input, are 1 for Nspring and 0.1 for Spring. If NEBSPRING is specified with Nspring 1 then the Spring parameter is required. If Nspring>1 is specified then also Spring2 and Spower are required.

NEBOPT OptMethod Specifies the optimization procedure. Since NEB is conceptually different from simple optimization, not all or not always options used in simple geometry optimization applicable to NEB. There are two optimization modes available for NEB: global (covering all images simultaneously) and local (that is, local to each image). Each method has its pro's and con's. The global method usually converges in fewer steps than local because its Hessian takes into account all degrees of freedom at once. On the other hand, the size of the matrix may become too large for moderate-size system, which might lead to problems (one dimension of the Hessian matrix may be as large as N(atoms)*3*N(images)). There are two geometry update methods available for both global and local optimization: Quasi-Newton and Conjugate-Gradient. Quasi-Newton is the preferred method at all times. NEB optimization in Z-matrix coordinates is not available at this time.

OptMethod can take any of the following values: GLOBALQN: global Quasi-Newton.

QN: Local Quasi-Newton. The preferred (and default) method.

NEBECONO (local optimization only) Requests that when at some point an image's geometry converges this image will not be recalculated in subsequent steps. This option can be used to speed up calculation in the end when some images have already converged. Please note though that even if an image has converged at one point it may become "un-converged" at a subsequent point due to increase in spring force (which is determined by position of the image with respect to its neighbors). This option is irrelevant in case of the global optimization because then the convergence state of a single image is not determined.

NOCLIMB Switches off the climbing-image feature. This option is generally not recommended and exists for debugging and troubleshooting purposes.

NONEBOPTENDS Do not optimize geometries the initial and final reaction states during NEB optimization.

5.6.1 Recommendations concerning the NEB method

Preparing input Please pay attention to the following points when preparing input for a NEB calculation:

- If an approximated transition state is known then try to use this information when preparing the input: do not just specify the initial and final state coordinates in the GEOVAR input section but also add some values in between to take advantage of the higher-order interpolation of the initial reaction path approximation.

- Try to optimize geometries of the initial and final reaction states (the end-points) as good as possible. ADF will by default optimize them too but doing this in advance can save quite a bit of time.

- If you do not want to optimize the end-points during NEB optimization you can use the NoNEBOptEnds input keyword and you know that the end-point geometries do **not** correspond to local minima, then you **must** make sure that they lie on the reaction path. If one (or both) of the end-points lie off the path then this may result in the images next to them sliding downhill behind the end-points, which will inevitably break the optimization.

- **Choosing an optimization method.** There are several optimization methods used in NEB but you should probably use one of the two Quasi-Newton methods: local (default) or global (*NEBOPT GlobalQN* input option). The main difference in functionality between the two methods is that the Climbing image technique is possible only with the GlobalQN method. In the default (local) method, the images are optimized in such a way that the

Cartesian distance between them is always constant. This means that the image with the highest energy is not necessarily the transition state. With the GlobalQN method, you are guaranteed that the image with the highest energy is at the point of the maximum energy on the reaction path.

Problems during optimization

Many problems may be avoided if you follow the recommendations above. If, however, you did follow the recommendations and still have problems then please read below. Here follows a list of common problems with possible solutions: **Optimization stops with a message that the angle has become too small.**

- Provided that the end-points **are** local minima, this may still happen if the initial guess for the reaction path was too rough an approximation. This usually result in very large forces on some of the images, which may result in very large steps. This is not a problem in itself but a problem may be that neighboring images get significantly different steps. If this happens, the NEB chain becomes jagged. This may get quickly out of hands if the Cartesian distances between images are comparable with the steps taken during optimization. The cure in this case is to either reduce the number of images (to increase distances between them) or to decrease the max step size (the *STEP RAD=* parameter).

- Another reason for the angle becoming too sharp is that the reaction path is very complex. In this case, it may help to use more images to "smoothen" it.

In all other cases it is recommended to contact the SCM support and specify exactly what went wrong and send along the input and output files. It is recommended to use the *DEBUG NEB* input keyword, which produces extra debugging information. Doing so will speed things up a lot because we won't have to repeat your calculation.

5.7 Special Features

5.7.1 Initial Hessian

In a Geometry Optimization (or Transition State search) the Hessian matrix - second derivatives of the energy with respect to changes in coordinates - is updated while the program steps around in an attempt to find the (local) energy minimum. The quality of the initial Hessian may have a considerable impact on the required number of steps to reach geometric convergence.

By default the initial Hessian is read from a restart file (see the key RESTART) if a restart file is present. Otherwise it is constructed from a force field [*11* (page 907)] that is implemented in the program. When using the new branch, the initial Hessian is calculated by the same method as in QUILD [*225* (page 920)]. In case of the old branch of optimization one can modify this with the key HESSDIAG. With the new branch of optimization it is possible to read an Hessian from an input file, see subkey INITHESS of the key GEOMETRY.

5.7.2 Constrained optimizations, LT (new branch)

The key CONSTRAINTS can only be used in case of the New branch for optimization of coordinates. The input for this key is very similar to that of the RESTRAINT keyword. The key CONSTRAINTS can, however, also be used to constrain Cartesian coordinates. Note that the key RESTRAINT and freezing of coordinates with the GEOVAR key can also be used in the New branch for optimization of coordinates. Currently, the New branch for optimization can only be used in geometry optimizations and transition state searches.

The CONSTRAINTS keyword allows geometry optimizations with constraints for the distance between two atoms, an angle defined by three atoms, a dihedral angle defined by four atoms, or to freeze a block of atoms:

```
CONSTRAINTS
  ATOM Ia1 {Xa1 Ya1 Za1}
  COORD Ia1 Icoord {valcoord}
  DIST Ia1 Ia2 Ra
```

```
  ANGLE Ib1 Ib2 Ib3 Rb
  DIHED Ic1 Ic2 Ic3 Ic4 Rc
  SUMDIST Ic1 Ic2 Ic3 Ic4 Rc
  DIFDIST Ic1 Ic2 Ic3 Ic4 Rc
  BLOCK bname
end
```

The ATOM, COORD, DIST, ANGLE, and DIHED constraints do not have to be satisfied at the start of the geometry optimization. The ATOM and COORD constraints can only be used in Cartesian optimizations while all other constraints may only be used with delocalized coordinates.

Important note about constraints and symmetry: if the system has some symmetry, which is going to be maintained, that is if a non-NOSYM symmetry is specified on input or if the initial geometry has non-NOSYM symmetry and "Symmetry NOSYM" is not specified, then the constraints specified here must also be symmetric. This means that if, for example, two distances are equivalent due to symmetry then and are going to be constrained then both must be specified in the CONSTRAINTS input block. It is incorrect to specify only one of them and hope that ADF will take care of the other automatically.

ATOM When *ATOM* is specified, the Cartesian coordinates of atom Ia1 are constrained to: Xa1 Ya1 Za1. The atom number should be given in *Input order*; the value for the coordinates in Angstrom. Optionally one can give the three Cartesian coordinates a value. This key can only be used in Cartesian optimizations.

COORD When *COORD* is specified, the Icoord Cartesian coordinate of atom Ia1 is constrained to: valcoord The atom number should be given in *Input order*; the value for the coordinate in Angstrom. Icoord is 1 means the x-coordinate. Icoord is 2 means the y-coordinate. Icoord is 3 means the z-coordinate. Optionally one can give the Cartesian coordinate a value. This key can only be used in Cartesian optimizations.

DIST When *DIST* is specified, the distance between atoms Ia1 and Ia2 is constrained to the value Ra in Angstrom.

ANGLE When *ANGLE* is specified, the angle between atoms Ib1, Ib2 and Ib3 (Ib1-Ib2-Ib3) is constrained to the value Rb in degrees.

DIHED When *DIHED* is specified, the dihedral angle between atoms Ic1, Ic2, Ic3 and Ic4 (Ic1-Ic2-Ic3-Ic4) is restrained to the value Rc in degrees. The dihedral angle Ic1-Ic2-Ic3-Ic4 is defined in the same way as for the Z-matrix in ADF. The dihedral angle is projected onto the [0,2:math:*pi*] interval, so there should be no difference between specifying -30° or 330°.

SUMDIST When *SUMDIST* is specified, the sum of the distances R(Ia1,Ia2)+R(Ia3,Ia4) is constrained to the value Ra in Angstrom.

DIFDIST When *DIFDIST* is specified, the difference R(Ia3,Ia4)-R(Ia1,Ia2) bitween the distances R(Ia3,Ia4) and R(Ia1,Ia2) is constrained to the value Ra in Angstrom.

> **Note**: The *SUMDIST* and *DIFDIST* constraints should be used with a *Constraints PartialConverge* keyword in the *Geometry* input block.

BLOCK Block constraints allow the internal degrees of freedom of a block of atoms to be frozen, so that the block moves as a whole. To apply block constraints, you add block labels to atoms in the Atoms block, and then add the block constraint in the Constraints input block:

```
ATOMS
   1.C       -0.004115   -0.000021    0.000023 b=b1
   2.C        1.535711    0.000022    0.000008 b=b2
   3.H       -0.399693    1.027812   -0.000082 b=b1
   4.H       -0.399745   -0.513934    0.890139 b=b1
   5.H       -0.399612   -0.513952   -0.890156 b=b1
   6.H        1.931188    0.514066    0.890140 b=b2
   7.H        1.931432    0.513819   -0.890121 b=b2
   8.H        1.931281   -1.027824    0.000244 b=b2
END
```

```
CONSTRAINTS
  BLOCK b1
  BLOCK b2
END
```

Note: The following restrictions apply to optimization with block constraints:

- block constraints may only be used with delocalized coordinates;

- there should be no other constrained coordinates used together with block constraints although this may work in many situation;

- the user should absolutely avoid specifying other constraints that include atoms of a frozen block.

5.7.3 Constrained optimizations, IRC, NEB, LT (old branch)

GEOVAR

The block key GeoVar is used

- In case of the old branch of optimizations

- To put restrictions on the number of coordinates that are varied and

- To define Linear Transit or NEB parameters and assign them initial and final (and in case of NEB - also intermediate) values.

geovar can also be used to assign (initial) values to coordinates without other implications, but this feature is accidental.

In the input section of atomic coordinates (key ATOMS) identifiers (names) may be used rather than numerical values wherever coordinate values are expected: x, y, z in case of Cartesian coordinate input; r, q, f in case of internal coordinates. All such identifiers must then be specified under geovar and assigned a value.

```
GEOVAR
  Name Data
  ...
end
```

Name An identifier that can be used in place of a numerical value for one or more of the atomic coordinate values under atoms.

Data Either of the following three formats:

1. A single value simply assigns the value to the corresponding atomic coordinate(s).

2. Two or more values (separated by a delimiter) imply that the corresponding atomic coordinate is a Linear Transit or a Nudged Elastic Band parameter. For Linear Transit, only two values are allowed in which case they specify initial and final values of the LT path, respectively. In case of a NEB calculation one can provide more than just initial and final values to get a better initial approximation of the reaction path. It is generally recommended (and in some cases necessary) to use more values. Intermediate images will be obtained by polynomial interpolation of degree N-1, where N is the number of values.

3. A single value followed by a letter F assigns the value to the corresponding atomic coordinates *and* specifies that these coordinates are frozen: they will not be optimized.

As regards the optimization of coordinates other than the frozen ones and the LT or NEB parameters, the meaning and effect of the input under geovar depends on the subkey *optim* in the geometry block:

If selected has been set, optimizations are carried out only for the coordinates that are referred to under geovar (and that are not Linear Transit parameters or Frozen). All coordinates that were input as simple numerical data under atoms are kept frozen then.

Alternatively, if selected has *not* been set (: all, the default) all atomic coordinates are optimized (except the Linear Transit parameters and the explicitly frozen coordinates). In that case, each assignments under geovar other than to freeze the coordinate or to define it as a Linear Transit parameter simply assigns an initial value to the pertaining coordinates. In this respect it is not different from typing the numerical value directly in the atoms block, except for the next aspect. Please note that whereas during a linear transit run the LT parameters are *never* optimized, the NEB parameters specified in the *geovar* section are always optimized.

The same identifier may be used for two or more coordinates in atoms. If they are varied (i.e. if they are not frozen) they will forcibly be kept equal throughout the optimization so that they constitute only one degree of freedom. Don't use the same geovar variable for coordinates that belong to atoms of different chemical types or to different *types* of coordinates (an angle and a bond length for instance). It is not sensible to do so and it will very probably lead to an error abort or to stupid results.

It is allowed to put as atomic coordinate under atoms *minus* a geovar variable name, i.e. the name preceded directly by a minus sign (without a blank in between!). The coordinate will then be kept equal, but with opposite sign, to coordinates that are defined by the same variable without the minus sign. The initial (and final, in case of a LT or NEB run) value for that coordinate is the negative of the geovar value.

Coordinate types

Restricted optimizations are performed by freezing certain coordinates, by explicitly referring to one and the same geovar identifier for different coordinates, or by using the selected option. In addition, they are implicit in each Linear Transit or IRC run. All restricted optimizations demand that the *type* of optimization variables (Cartesian or Z-matrix) equal the type of coordinates used in atoms. zcart input under atoms is considered to be *Cartesian* in this respect.

If this is violated in a Linear Transit calculation the program will abort. If you apply the Frozen option under geovar, while not using the same coordinate type for atoms as for optimization, an error will occur. If you refer to the same geovar identifier for distinct coordinates while the atoms and the optimization types of variables do not match, the program will continue and assume that you only have assigned the same *starting* values to the pertaining coordinates. No equality constraints will be in effect then during the optimization. **Linear combinations of constraint**

It is often desirable to carry out a geometry optimization in internal coordinates where two or more of the coordinates are required to maintain constant values relative to each other. The most simple case, where two internal coordinates a kept equal can be achieved by referencing both coordinates to a single variable in the GEOVAR block. Ensuring a different relationship, such as forcing one bond length in a molecule to be 0.5 Angstrom longer than another is more difficult to achieve. These kind of constraints can often be be managed through the creative use of dummy atoms but this is generally laborious and not always possible at all.

This key can not be used in case of optimization in delocalized coordinates.

The LINEARCONSTRAINTS keyword allows geometry optimizations with constraints defined by arbitrary linear combinations of internal coordinates to be performed quite straightforwardly. The keyword allows the linear combination to be constrained or used as part of a linear transit calculation with the constrained value being stepped as would a variable from the GEOVAR block.

```
LINEARCONSTRAINTS
  Name1 Data1
  VAR11 Coef11
  VAR12 Coef12
  ...
  SUBEND
  Name2 Data2
  VAR21 Coef21
  VAR22 Coef22
  ...
```

```
   SUBEND
end
```

Namex Identifier of the xth linear constraint.

Datax Either of two formats

1. A single number, giving the value of the xth constraint

2. Two numbers, the first as in 1) and the second the final value in a linear transit calculation. The LIN-EARTRANSIT keyword must be present in the GEOMETRY block.

Varxy Name of the yth variable in that is part of the xth constraint. Varxy must be defined in the GEOVAR block.

Coeffxy Coefficient of Varxy in the linear combination defining the constraint. Thus: Coeff11*Var11 + Co-eff12*Var12 ...= Data1 The summation must be consistent with the initial values of the Varxy.

This procedure is only possible when the geometry is defined in terms of internal coordinates. Although the program will not complain, it makes no sense to have linear combinations containing both bonds and angles of course.

The number of linear constraints must be less than or equal to the number of entries in the GEOVAR block. Only internal coordinates involving QM atoms can be included at this stage.

As a geometry optimization is run, the force acting on the linear constraints will be printed immediately after the forces on the internal coordinates. The constraint forces may be useful in the search for a transition state for instance.

Z-matrix and symmetry

If the structure of the Z-matrix does not reflect the symmetry of the molecule *and* constraints are applied the program may encounter algorithmic problems to match all demands. As a result some of the *frozen* coordinates may be found to change. Usually these changes are very small. To cure this: build the Z-matrix in a symmetric way.

Summary of geovar, optim, and atoms

For *unconstrained* optimization: don't use geovar, apply *optim* if *Cartesian* optimization is required while the data in the atoms block was in z-matrix format or when *z*-matrix* optimization is required while the atoms input was in zcart format. Provide the atomic coordinates (atoms) directly as numerical data.

For optimizations where only very few coordinates are frozen: use geovar to set a few coordinates to frozen and/or to enforce equality of optimization coordinates whose values should remain equal. Don't use *optim*: the type of optimization coordinates - Cartesian or internal - must be identical to what is used in the atoms input part because you're using constraints now. In the atoms section, use identifiers for the frozen coordinates and for those that should satisfy equality conditions; use numerical input for all other (optimization) coordinates.

For very limited optimization: turn on the selected option with *optim* and assign with geovar initial values to the coordinates that you want to optimize. In the atoms input use identifiers for these coordinates. The numerical input coordinates are kept frozen automatically now.

Initial Hessian

By default the initial Hessian is read from a restart file - see the key RESTART - or constructed from a force field [*11* (page 907)] that is implemented in the program. In the latter case the user can modify the so-generated initial Hessian in four ways:

- By setting all diagonal elements to some constant.

- By defining *three* constants, one for distances (or Cartesian displacements, as the case may be), one for bond angles, and one for dihedral angles. All diagonal elements of the Hessian are adapted accordingly.

- By giving diagonal-Hessian values for one or more specific coordinates.

For each element *i* for which a diagonal Hessian value Hii is supplied the off-diagonal elements Hij, (all j i $\neq$ j) are set to zero.

A combination of the above options is possible. The rules of how combinations are interpreted by the program are:

- The program first initializes the Hessian using the force field (or restart data).

- If a single constant (*1*) or three constants (*2*) are supplied, all diagonal elements are adjusted (and all off diagonal elements are set to zero).

- If a list of diagonal values is supplied (*3*), this overrides the first so many values of the diagonal. Such a list is not required to cover *all* diagonal elements. If the list is shorter than the dimension of the Hessian, i.e. the number of atomic coordinates, only the first so many elements will be adjusted.

- If any individual elements are supplied specifically (*4*), their values are replaced in the diagonal defined thus far.

All input values of the Hessian are in units of Hartree/bohr2 for Cartesian coordinates and bond lengths. Hartree/radian2 for bond angles and dihedral angles.

The first 2 options are controlled by the key HESSDIAG:

```
HESSDIAG {rad=rval} {ang=rval} {dih=rval}
```

HESSDIAG rad=radvalue to assign a value to all Hessian diagonal elements that refer to distance coordinates (bond length in case of z-matrix coordinates, Cartesian coordinates otherwise), ang=angvalue to assign a value to all elements that refer to bond angles, and finally dih=dihvalue for dihedral angles. ang and dih are not significant in Cartesian optimizations.

The force field derived initial Hessian can be printed for inspection. Type in input:

```
HESSTEST
```

Hessian values for selected coordinates

ADF will construct and print the initial Hessian and then abort.

The diagonal elements for selected free coordinates can be given if these free variables are named in the geovar block.

```
GEOVAR
  Varname Data H=HessValue
end
```

Varname, Data The name of the variable and any data as discussed in the sections above: assignment of initial value, final value (in case of a Linear Transit run), or a Frozen specification.

HessValue The value for the diagonal element of the Hessian associated with that variable. All atomic coordinates that are defined by this variable will get the HessValue as diagonal element in the initial force field. Specification of a HessValue for a frozen coordinate or a Linear Transit parameter is meaningless.

5.7.4 Restrained optimizations

With the old branch of optimizations, the only way to constrain distances, angles or dihedral angles within a geometry optimization is by using a Z-matrix, and freezing that particular coordinate. With the key RESTRAINT it is possible to select **any** coordinate (distance, angle, dihedral), irrespective of the coordinates used, and restrain this coordinate.

Note the difference between *constrained* and *restrained* coordinates. At every step in the geometry optimization, the value of a *constrained* coordinate should match exactly a predefined fixed value. On the other hand, with *restraints*, a potential is added to the potential energy in order to satisfy the *restraint*, which means that the *restraint* does not have to be satisfied exactly. For example, one can start with a geometry in a geometry optimization run in which the restraint is not satisfied.

The RESTRAINT keyword allows geometry optimizations with restraints for the distance between two atoms, an angle defined by three atoms, a dihedral angle defined by four atoms, and (or) a distance difference defined by four atoms:

```
RESTRAINT
  DIST Ia1 Ia2 Ra {[Aa] [Ba]}
  ANGLE Ib1 Ib2 Ib3 Rb {[Ab] [Bb]}
  DIHED Ic1 Ic2 Ic3 Ic4 Rc {[Ac] [Bc]}
  DD Id1 Id2 Id3 Id4 R0 [{Ad} {Bd}]
end
```

DIST When *DIST* is specified, the distance between atoms Ia1 and Ia2 is restrained to the value Ra. The atom numbers should be given in *Input order*; the value for the distance in Angstrom. The Aa and Ba values are mere technical values, that don't have to be specified (in fact, recommended not to change these values); the default values of 2.0 resp. 0.1 have been chosen on sensible grounds.

ANGLE When *ANGLE* is specified, the angle between atoms Ib1, Ib2 and Ib3 (Ib1-Ib2-Ib3) is restrained to the value Rb. The atom numbers should be given in *Input order*; the value for the angle in degrees. The Aa and Ba values are mere technical values, that don't have to be specified (in fact, recommended not to change these values); the default values of 1.0 resp. 0.1 have been chosen on sensible grounds.

DIHED When *DIHED* is specified, the dihedral angle between atoms Ic1, Ic2, Ic3 and Ic4 (Ic1-Ic2-Ic3-Ic4) is restrained to the value Rc. The atom numbers should be given in *Input order*; the value for the angle in degrees. The Aa and Ba values are mere technical values, that don't have to be specified (in fact, recommended not to change these values); the default values of 0.5 resp. 0.1 have been chosen on sensible grounds. The dihedral angle Ic1-Ic2-Ic3-Ic4 is defined in the same way as for the Z-matrix in ADF. The dihedral angle is projected onto the [0,2:math:*pi*] interval, so there should be no difference between specifying -30° or 330°.

DD When *DD* is specified, it is possible to restrain a difference between two distances: R0 = (r1 - r2) - (r3 - r4), where r1..r4 are positions of four atoms Id1..Id4 and R0 is the final restrained property. The functional form and meaning of Ad and Bd is the same as with plain distance restraints. Atoms Id1..Id4 need not all to be different.

5.7.5 Symmetry versus constraints

The symmetry of the atomic system defined by the input Schönfliess symbol is preserved during optimization. If the input information (which coordinates are kept frozen and which are optimized) conflicts with the symmetry, the result is unpredictable. For example, if two distances are equivalent due to symmetry then and are going to be constrained then both must be specified in the CONSTRAINTS input block. It is incorrect to specify only one of them and hope that ADF will take care of the other automatically.

Input specifications that are in conflict with the point group symmetry may lead to an error or a non-converging optimization.

5.8 Frequencies

Harmonic frequencies can be computed in ADF either numerically or analytically. The frequencies are computed numerically by differentiation of energy gradients in slightly displaced geometries [*12* (page 907), *13* (page 907)]. The analytical second derivatives implementation in ADF is based on [*208* (page 919), *209* (page 919), *210* (page 919)].

5.8.1 Analytical Frequencies

The frequencies are calculated analytically by specifying the **AnalyticalFreq** block keyword (see below for more details). The analytical frequencies are as accurate as the numerical frequencies for the same integration accuracy, but

can be up to 3 to 5 times quicker to compute, depending on the molecule, integration grid parameters, and choice of basis set. The analytical frequencies are fully parallelized and linearly scaled.

Calculating the analytical frequencies requires the solution of the Coupled Perturbed Kohn-Sham (CPKS) equations, which is an iterative process. This part of the process is of order 3 x *number of atoms*, and is generally the main bottle neck in calculating the frequencies. (The immediate result of the solution of the CPKS equations is the U1 matrix, the components of which are closely related to the derivatives of the MO coefficients. One of the adjustable parameters in the input of an analytical frequencies calculation can be used to control the accuracy of the U1 matrix components.)

One disadvantage in calculating analytical frequencies is that the range of exchange-correlation functionals is limited. (This is because derivative formulas have to be derived for each exchange-correlation functional in ADF, which is not an straight forward task). Here are the currently available functionals:

LDA: XONLY, VWN, STOLL, PW92

Exchange GGA: Becke88, OPTx, PBEx, rPBEx, revPBEx

Correlation GGA: LYP, Perdew86, PBEc

XC GGA shortcuts: BP86, PBE, RPBE, revPBE, BLYP, OLYP, OPBE

Any functional not mentioned above is not implemented, including PW91 and Hartree-Fock.

To calculate the frequencies analytically, include the block key word ANALYTICALFREQ. Subkeys are available, but in general, to calculate the frequencies, no subkeys are required, and including the following in your run file is sufficient:

```
AnalyticalFreq
End
```

Unlike the numerical frequencies, the analytical frequencies can be computed immediately after a geometry optimization by including both block keywords in the same input file:

```
Geometry
  ... Geometry optimization options here ...
End

AnalyticalFreq
End
```

A note of caution: For accurate frequencies it is especially important to also have an accurately optimized geometry. During a geometry optimization the integration accuracy is set by default to "Normal", and so the resulting frequencies will also have this level of integration accuracy while it may be desirable to have frequencies computed with a higher accuracy. One might consider using Good NumericalQuality (or BeckeGrid quality) and set the convergence criteria for the geometry optimization tighter.

The format for the AnalyticalFreq block key is:

```
AnalyticalFreq
    PRINT {eigs} {u1} {parts} {raw_freq}
    DEBUG {fit} {hessian} {b1} {densities} {numbers} {symmetry} {all}
    MAX_CPKS_ITERATIONS Niter
    CHECK_CPKS_FROM_ITERATION  N
    U1_ACCURACY  x
    NUC N1 N2 ... Nk
End
```

An explanation of the subkeys follow.

PRINT This is primarily for debugging purposes. Choosing EIGS results in the print out of the MO eigenvectors, while U1 results in the print out of the U1 matrices. Except for small molecules this will result in a lot of data

being output, and so they are not recommended. Choosing PARTS results in the print out of various sub-hessians that add up to give the final analytical hessian. RAW_FREQ gives the eigenvalues of the initial force matrix, which are essentially the frequencies before rotational and translational degrees of freedom have been removed from the force matrix.

DEBUG This is for debugging purposes. The choice FIT results in the print out of information related to the calculation of the fit coefficients and their derivatives. HESSIAN results in the printing out of many of the sub-Hessians that add ups to give the final Hessian. Many more Hessians are printed out with this option that with the print parts subkey option (mentioned above). Choosing B1 gives data related to the frozen core orthogonalization coefficients and their derivatives. DENSITIES gives the integrals and moments of various densities computed during the calculation of the frequencies. Including NUMBERS results in the print out of numbers of basis functions, fit functions etc, as well as various integer arrays that are crucial to the calculation of the analytical second derivatives. SYMMETRY results in symmetry information and symmetry matrices being printed out. ALL can be used to print out all debug information.

MAX_CPKS_ITERATIONS Niter Calculating the analytical frequencies requires the solution of the Coupled Perturbed Kohn-Sham (CPKS) equations, which is an iterative process. For most systems tested so far, convergence to the required accuracy in the U1 matrix is achieved within Niter=20 iterations, which is the default. If convergence is not achieved (a warning will be printed in the output if this is the case) then this subkey can be used to increase the number of iterations, although convergence is not guaranteed. The user required accuracy of the U1 matrix, as well as the ADF integration accuracy, can effect the rates of convergence.

CHECK_CPKS_FROM_ITERATION N Solution of the CPKS equations is an iterative process, and convergence is achieved if the difference between U1 matrix of successive iterations falls below a certain threshold. This key can be used to determine at which iteration the checking should start taking place. The default is 1.

U1_ACCURACY x Solution of the CPKS equations is an iterative process, and convergence is achieved if the difference between U1 matrix of successive iterations falls below a certain threshold. This subkey can be used to set the threshold. The accuracy of the U1 will be 10**(-x). So, the higher the number the more accurate the U1 will be. The default is 4. While this parameter effects the accuracy of the frequencies, other factors also effect the accuracy of the frequencies, especially the ADF integration accuracy.

NUC N1 N2 ... Nk By default, when calculating the frequencies analytically, the derivatives of the energy with respect to all nuclei are calculated. This gives a complete Hessian (second derivative) matrix, from which the vibrational frequencies of the molecule can be calculated. However, there may be certain cases where only derivatives with respect to a subset of all the nuclei are required. In this case it is a considerable saving in time if only a partial Hessian is calculated. With this subkey, a list of the nuclei for which the derivatives are required can be specified. However, the frequencies in this case are not the vibrational frequencies of the molecule, but may be used in guiding certain transition state searches.

Restarting Analytical Frequency jobs

Analytical frequency jobs can be restarted if a previous job did not finish. A restart can be done if the file TAPE21 has been saved without any corruption to the file. Upon doing a restart for analytical frequencies, the ADF program will check for the presence of an incomplete Hessian and/or for the presence of an incomplete U1 matrix, then attempt to figure out what more needs to be done.

The restart option may also be useful in combination with the atom selection option (i.e. by specifying a list of atoms following the keyword nuc in the analyticalfreq block key, see previous notes in this section). So for instance, you could calculate the partial Hessian for a subset of atoms of a molecule, and at a later time add another subset of atoms, or the rest of the atoms of the molecule to complete the Hessian.

Simplified interface for isotopic shift calculations in ADF

It is now possible to calculate an IR isotopic shift without modifying the atom type. The alternative mass is specified using the IsotopicShift keyword as follows:

```
IsotopicShift n1=m1 n2=m2 ...
```

n1=m1, n2=m2 Here, n1, n2, etc. are sequential numbers of atoms in the input geometry and m1, m2, etc. are their alternative masses.

All frequencies and intensities are calculated as usual with the original atomic masses taken from the corresponding fragment files and the isotopic shifts are calculated and printed afterwards. The Hessian is not broken down into irreducible representations and the rigid motions are not projected out prior to the normal mode analysis. Normal modes (eigenvectors) corresponding to the original and the shifted frequencies are matched against each other based on their overlap, which allows to calculate an isotopic shift for each frequency. This analysis may, however, fail for highly symmetric molecules with degenerate modes.

This feature currently works in conjunction with analytical frequencies only.

5.8.2 Numerical Frequencies

Input options

Calculation of the numerical frequencies is specified using the FREQUENCIES block in the GEOMETRY block. Most of the subkeys in the geometry block are meaningless for the calculation of frequencies. Indeed, a Frequencies calculation is not a variation on optimization, but rather a sequence of Single Point runs for the equilibrium geometry and a series of slightly different geometries. By comparison of the computed gradients the force constants and hence the frequencies are computed (in the harmonic approximation of the energy surface).

```
GEOMETRY
  Frequencies
    {Symm}
    {Allowed}
    {Numdif Numdif}
    {Disrad drad}
    {Disang dang}
    {SCANALL}
    {NOSCAN}
  End
  iterations Niter
end
```

Symm This switch requests that frequencies are calculated in symmetric displacements. During such a calculation first symmetric atomic displacements are constructed. The number of such displacements in each irreducible representation corresponds to the number of frequencies with the corresponding symmetry. All displaced geometries within one representation have the same symmetry, which enables us to use it to speed up the computation significantly. This is a new option and for now it only works with geometries specified in Cartesian coordinates. This option does not work correctly with a restart file. **This option does not work correctly when symmetry is explicitly specified in the input file.**

Allowed Another advantage of the symmetric displacements is that only a subset of frequencies can be calculated. The ALLOWED option requests computation of only IR-visible frequencies. This option is only useful for symmetric molecules where it can be a big time-saver.

Numdif Must have the value between 1 and 4 and specifies the type of numerical differentiation that is applied to compute the force constants from gradients in slightly displaced geometries: 1-, 2-, 3-, or 4-point numerical differentiation. In the case of 1-point differentiation the gradients of the displaced geometry are compared with the gradients at the input (equilibrium) geometry. In 2-point case both a negative and a positive displacement are applied, yielding much more accurate results but at the expense of more computations. This option is the default. In certain cases the 3-point differentiation method gives better result than 2-point because it also takes gradients in the middle point into account. This is the case when geometry has not completely converged and the residual

gradients are not quite close to zero. In this method, a formula is used that interpolates the second derivative (i.e. force constant) at the zero-force point. This way, the error due to a small deviation from the minimum geometry is decreased. The requirement is that the residual forces are small enough, more precisely, less that forces at displaced geometries (that is, using numdif=3 for arbitrary geometries is a bad idea). When Numdif=4 is specified, force constants matrix will be computed by making two displacements in each direction, the standard (see drad, dang below) and twice as short. The force constant is then computed using the Romberg formula that reduces the higher-order and noise components: H(tot)=(4*H(dx/2)-H(dx))/3. Although this method requires twice as many single-point evaluations, one can probably get reliable results using lower integration accuracy, which might be faster than the default.

dang and drad The displacements of the coordinates that will be varied. Dang applies to angles (bond and dihedral) in degrees and drad applies to Cartesian (x, y, z) coordinates and to bond lengths, in angstrom. Defaults: 1 degree and 0.01 angstrom.

Niter In a calculation of frequencies it is the total number of (displaced) geometries for which gradients are computed. By default this is internally determined such that the calculation of frequencies can be completed. If you reduce it, the run will only partially build the matrix of force constants and a restart is required to complete the computation.

WARNING: you cannot combine a Frequencies calculation with the QM/MM feature.

SCANALL and NOSCAN ADF can scan some or all normal modes after a frequency calculation to verify the corresponding frequencies. By default, the normal modes corresponding all found imaginary frequencies are scanned. This can be switched off by specifying NOSCAN. Specifying SCANALL will tell ADF to scan along **all** normal modes. These options apply only to calculations in atomic displacements, that is when SYMM is not specified. These two options are mutually exclusive, the one specified last taking precedence.

Cartesian versus Z-matrix displacements

Cartesian displacements yield usually a higher accuracy than *Z-matrix* displacements because in the former case cancellation of numerical integration errors between the different geometries is (almost always) larger.

If Z-matrix coordinates are used as the displacement variables, then make sure that no bond angles of 180 (or zero) degrees are among them. They will very probably be treated incorrectly. If your molecule has such bond angles, use dummies to redefine the coordinates or use Cartesian displacements.

Frequencies and GEOVAR keyword

The use of the GEOVAR keyword in combination with a Frequencies run implies that constraints may be applied to the displacements, even if no coordinates are explicitly frozen. If different coordinates are connected to the same variable in the GEOVAR block, only combined displacements of the atoms will be allowed that correspond to a small change in the GEOVAR variable. For this reason, the combination of the GEOVAR and Frequencies keywords is to be handled with extreme caution. If no constraints are intended, it is recommended not to use the GEOVAR keyword, but to use DEFINE instead, or to specify the coordinates explicitly

5.8.3 Mobile Block Hessian (MBH)

The mobile block Hessian (MBH) method [*282* (page 923), *283* (page 923)] is useful when calculating vibrational frequencies of a small part of a very large system (molecule or cluster). Calculation of the full spectrum of such a system may be inefficient and is unnecessary if one is interested in one particular part. Besides, it may be difficult to extract normal modes related to the interesting sub-system out of the whole spectrum. Using MBH it is possible to treat parts of the system as rigid blocks. Each block will usually have only six frequencies related to its rigid motions compared to 3*N for when each atom of the block is treated separately.

The calculation of frequencies using mobile blocks is invoked by specifying FREQUENCIES and MBH keywords at the same time in the GEOMETRY input block:

```
GEOMETRY
  FREQUENCIES
  End
  MBH blockname1 blockname2 ...
End
```

The names of blocks must correspond to the ones specified in the *b=* parameter in the ATOMS input block.

The second derivatives with respect to block motions are calculated by numerical differentiation. Since the number of degrees of freedom is reduced, the number of second derivatives is reduced as well. Therefore the MBH can realize a speed-up in the calculation of the Hessian compared to a full numerical frequency calculation.

MBH for partially optimized structures

MBH is suitable to calculate frequencies in partially optimized structures. Assume a geometry optimization is performed with the BLOCK subkey in the CONSTRAINTS section [see constrained geometry optimizations]. During the geometry optimization, the shape of the block is not changed. The internal geometry of the block is kept fixed, but the block as a whole can still translate or rotate.

At the end of such a partial geometry optimization, the position and orientation of the block are optimized. The total force on the block is zero. However, there might be still some residual forces within a block, since those degrees of freedom were not optimized.

A traditional frequency calculation performed on this partially optimized structure might result in non-physical imaginary frequencies without a clear interpretation. Therefore one should use an adapted formulation of normal mode analysis: the Mobile Block Hessian method. The MBH does not consider the internal degrees of freedom of the block (on which residual forces) apply, but instead uses the position/orientation of the block as coordinates. In the resulting normal mode eigenvectors, all atoms within the same block move collectively.

Of course, MBH can also be applied on a fully optimized structure.

Accuracy

The second derivatives with respect to Cartesian displacements (3 translations) of the free atoms (atoms not belonging to any block) and those with respect to block motions (3 block translation/3 block rotations) are calculated by numerical differentiation of the gradient. The accuracy of the second derivatives is mainly influenced by the accuracy of the gradient evaluation (e.g. accuracy numerical integration) and the step size in the numerical differentiation. The parameters DISRAD and DISANG can be specified to set the step size for Cartesian displacements (translations) and block rotations respectively. The step size for angles is automatically scaled with the block size.

```
FREQUENCIES
  {DISRAD drad}
  {DISANG dang}
End
```

The default values for drad and dang are the same as in the case of a standard numerical frequency run.

MBH Notes

Blocks consisting of 1 or 2 atoms are not supported and are ignored. This means that each atom of such a block is treated separately.

At this moment, it is not possible to calculate IR intensities with the MBH method.

The printed output of the MBH normal mode analysis lists all frequencies, including the ones corresponding to rotations and translation of the molecule as the whole. Note that while frequencies corresponding to translations should always be close to zero, the magnitude of the ones corresponding to rotations depends on how well the geometry is optimized.

Debug information can be obtained if one includes DEBUG MBHNormalModes.

5.8.4 Thermodynamics

At the end of a completed Frequencies calculation, a survey is given of thermodynamic properties: Heat Capacity, Internal Energy, Entropy. The computed results assume an ideal gas, and electronic contributions are ignored. The latter is a serious omission if the electronic configuration is (almost) degenerate, but the effect is small whenever the energy difference with the next state is large compared to the vibrational frequencies.

```
THERMO {P=pressure} {TMin=temp1} {TMax=temp2} {nT=nT}
```

pressure The Pressure in atmospheres. Default value: 1.0. A zero or negative pressure is adjusted by the program to a (very) small number 1e-4

TMin, TMax The endpoints of the Temperature range (in K), for which the results are computed. By default only room temperature is considered (298.15 K). If you only specify `TMin` (and not `TMax`), then TMax is assumed to be equal to `TMin`. A zero or negative temperature is adjusted by the program to a (very) small number 1e-4

nT The number of steps by which the temperature interval is scanned. By default it is computed by the program from the temperature range (temp1, temp2), such that the step size is as close as possible to 10 K. Note that the number of temperatures for which the calculations are done is one more than the number of temperature *steps*.

The thermal analysis is based on the temperature dependent partition function. The energy of a (non-linear) molecule is (if the energy is measured from the zero-point energy)

$$\frac{E}{NkT} = \frac{3}{2} + \frac{3}{2} + \sum_j^{3N-6} \left(\frac{h\nu_j}{2kT} + \frac{h\nu_j}{kT(e^{h\nu_j/(kT)} - 1)} \right) - \frac{D}{kT} \qquad (5.1.2)$$

The summation is over all harmonic ν_j, h is Planck's constant and D is the dissociation energy

$$D = D_0 + \sum_j \frac{h\nu_j}{2} \qquad (5.1.3)$$

Contributions from low (less than 20 1/cm) frequencies to entropy, heat capacity and internal energy have always been excluded from the total values for thermodynamical quantities listed before. In ADF2013 they are also listed separately so the user can add them if they wish.

Gibbs free energy change for a gas phase reaction

Here an example is given how to calculate the free energy change for a reaction. In the ADF output of a frequency calculation you can find the electronic bonding energy and nuclear kinetic energies, at room temperature. Example part of the ADF output of a nonlinear molecule:

```
                                     hartree            eV          kcal/
→mol          kJ/mol
                                --------------------   ----------   ----------
→        -----------

Total Bonding Energy:           -0.744356253597793     -20.2550      -467.
→091          -1954.31


Zero-Point Energy :      0.033172 a.u.
===================      0.902646 eV

    Temp                                           Transl    Rotat    ␣
→Vibrat     Total
    ----                                           ------    -----    --
→----      -----
```

```
    298.15   Entropy (cal/mole-K):                              34.441    11.494    ␣
→0.126    46.061
             Internal Energy (Kcal/mole):                        0.889     0.889    ␣
→20.847    22.624
             Constant Volume Heat Capacity (cal/mole-K):         2.981     2.981    ␣
→0.533     6.495
```

The nuclear internal energy = zero point energy + 3 kT + small correction term = 22.624 kcal/mol. 3 kT = 3/2 kT for rotation, and 3/2 kT for translation (i.e. 1/2 kT for each degree of freedom). The small correction term is a term due to the vibration partition function, depending on the temperature not only the ground state vibrational levels are occupied, see also Eq. (5.1.2).

The electronic internal energy = -467.091 kcal/mol. In ADF the electronic internal energy is normally calculated with respect to (artificial) spherical averaged neutral atoms.

The electronic + nuclear internal energy U = -467.091 + 22.624 = -444.467 kcal/mol

Gas phase pV/n = RT = 8.314472 * 298.15 / 4184 = 0.592 kcal/mol

Enthalpy H = U + pV = -444.467 + 0.592 = -443.875 kcal/mol

Gibbs free energy G = H - TS = -443.875 - 298.15*46.061/1000 = -457.608 kcal/mol

For a calculation of the free energy change for reaction (Δ G), you will of course have to to this for the reactant and product molecules, and add and subtract these energies, for each molecule proportional to the number of molecules that take place in the reaction. Application of ADF for obtaining enthalpy, entropy and Gibbs free energy can for instance be obtained in Refs. [*347,348* (page 927)]

5.8.5 Accuracy

Accuracy is a crucial aspect in the computation of frequencies, in particular for modes with low frequencies: the gradients at the geometries displaced along that mode will hardly change - analytically - from their equilibrium values, so numerical integration noise may easily affect the reliability of the computed *differences* in gradients. It is worthwhile to consider carefully the *size* of the displacements. At one hand they should be small in order to suppress the effect of higher order (anharmonic) terms in the energy surface around the minimum, at the other hand they should be large enough to get significant differences in gradients so that these are computed reliably.

High precision calculations where low frequency modes are involved may require high integration settings [*14* (page 907)]. It is sometimes advisable to increase the numerical accuracy of the calculation:

```
NumericalQuality {Good|VeryGood}
```

Using 2-point differentiation rather than 1-point differentiation implies two-sided displacements of the atoms. This doubles the computational effort but in the so-computed force constants all anharmonic terms of *odd* order are eliminated. Since in general the lowest anharmonicity is third order this eliminates the first anharmonicity. Again, this is a feature directed primarily at obtaining highly accurate and reliable results.

The 3- and 4-point methods are intended to assist in special cases and as an extra check when the results obtained with the 2-point formula are not satisfactory. The 3-point formula should be used when residual forces after geometry optimization are between 0.01 and 0.0001 a.u./angstrom. In this case frequencies obtained with the 3-point formula are much closer to those that would be computed at the exact optimum geometry.

If a Frequencies calculation is carried out only to construct a good start-up Hessian for a TS search (see the restart key), accurate results are not crucial. The most important thing in such a situation is to get a fair guess for the negative eigenvalue and its associated mode, and to avoid spurious additional negative eigenvalues. We recommend to avoid the rather time-consuming standard Hessian-computing preparation run for a TS search and to lower the precision of the Frequencies run. A reasonable value should be 4.0.

5.8.6 Isotope Shifts of Vibrational Frequencies

To calculate isotopic shifts using ADF do the following:

- Calculate frequencies and save TAPE21 with a different name, say result.t21

- Modify the input file as follows:

    - add "RESTART result.t21" anywhere in the input file

    - create new fragment file with different mass

    - specify the fragment file in FRAGMENTS section

- Run ADF with the new input

Alternatively one can use the the key *ATOMPROPS* (page 37) to change the masses of the atoms.

Please note that if you change the fragment file for an atom that has symmetry-equivalent ones then the new fragment file will be applied to all of the atoms. Example: first calculate the NH3 frequencies in the C(3v) symmetry and then change H to D. This will mean that one calculates the frequencies of a ND3 molecule and not of NH2D as one might want to do. If one wants to calculate the frequencies of NH2D one first has to do a calculation with lower symmetry, say C(s), to be able to change isotope of only one of the hydrogens.

5.8.7 Scanning a Range of Frequencies

In ADF2006 it was already possible to request a full scan of all frequencies obtained by finite differences. This was originally done to help identify spurious imaginary frequencies that sometimes appear where one would expect a very low (nearly zero) frequency. Most frequently this happens when there is a barrier-free rotation of, for example, methyl groups.

Starting from ADF2007.01, it is possible to scan any range of frequencies calculated in the same run or found in a restart file. Note: while SCANFREQ works in combination with `SYMMETRY NOSYM`, it might not if the user explicitly specifies any other symmetry via the *SYMMETRY* (page 28) key. The input keyword used to request the scan is as follows:

```
SCANFREQ low high {NUM=num DISRAD=disrad}
```

low, high Two values defining an interval of frequencies to scan. Frequencies that fall within the interval will be recalculated by numerical differentiation of the gradient along the respective frequency's normal mode. This means that $2**N*$ single-point calculations with gradients will be performed, where *N* is the number of frequencies within the range. Imaginary frequencies are specified using negative values, which is consistent with the notation adopted within ADF.

num Num is an integer number specifying how many points are to be used for numerical differentiation: 2, 4, or 6. The default value is 2.

disrad Disrad specifies the step size (in Angstrom) to be made in each direction for 2-point differentiation. For the 4-point differentiation the maximum deviation from the equilibrium geometry will be twice as large as for the 2-point one and so on. The default value for disrad is the same as used for numerical frequencies.

The main advantage of this method is that single-point calculations used to obtain a force constant are performed within the same symmetry and, usually, with the same numerical integration grid, which significantly reduces the level of numerical noise and thus increases accuracy of the calculated frequency.

5.8.8 Moments of inertia

In case frequencies are calculated in ADF, ADF also reports the moments of inertia of the molecule in units of amu bohr^2 (amu = atomic mass unit).

- See also section: *Excited state (geometry) optimizations* (page 182)

SPECTROSCOPIC PROPERTIES

See also:

ADF-GUI tutorials: excitation energies, vibrational frequencies

Examples: *IR spectra* (page 606), *excitation energies* (page 636), *response properties* (page 680), *NMR* (page 718), *ESR* (page 736), *EFG* (page 749)

6.1 IR spectra, (resonance) Raman, VROA, VCD

In ADF infrared and Raman spectroscopy can be studied for molecular vibrations. In the Born-Oppenheimer and harmonic approximations the vibrational frequencies are determined by the normal modes corresponding to the molecular electronic ground state potential energy surface. In resonance Raman spectroscopy the molecule is excited to near one of its electronic excited states, to improve the sensitivity compared to traditional Raman spectroscopy. Vibrational circular dichroism (VCD) is the differential absorption of left and right circularly polarized infrared light by vibrating molecules.

6.1.1 IR spectra

The IR frequencies can be calculated with the FREQUENCIES sub-block of the key GEOMETRY (numerical frequencies),

```
GEOMETRY
  FREQUENCIES
  End
END
```

or with the block key ANALYICALFREQ (analytical frequencies),

```
ANALYICALFREQ
END
```

These keys are described more extensively here: *IR Frequencies* (page 139).

6.1.2 Raman scattering

Raman scattering intensities and depolarization ratios for all molecular vibrations at a certain laser frequency can be calculated in a single run. The run type must be Frequencies, which is arranged with the FREQUENCIES subkey of the key GEOMETRY (numerical frequencies), or with the block key ANALYICALFREQ (analytical frequencies), see *IR Frequencies* (page 139).

Here the RESPONSE key is used to specify that Raman intensities are computed. In case of spin-orbit coupling or hybrids one needs to use the AORESPONSE key. The frequency dependent Raman scattering can be calculated for 1 laser frequency at the time.

```
RESPONSE
  RAMAN
  Nfreq 1
  FrqBeg Laserfreq
END
```

Frequencies or wavelengths The number of frequencies Nfreq should be 1. With subkey Frqbeg the value of the Laser frequency value (Laserfreq) can be given. The unit of `Frqbeg` is eV.

For static Raman scattering ($\omega = 0$) use:

```
RESPONSE
 RAMAN
END
```

The Raman scattering calculation is very similar to an IR intensity calculation. In fact, all IR output is automatically generated as well. At all distorted geometries the dipole polarizability tensor is calculated. This is very time-consuming and is only feasible for small molecules. More details on the RESPONSE key can be found *here* (page 190).

There are a few caveats:

- Numerical integration (BeckeGrid) accuracy must be high

- A calculation in which only a subset of the atoms is displaced is not possible for Raman calculations.

- For good results, a well converged (with the same basis and functional) equilibrium geometry must be used.

Because of this last point, it is wise to always start the RAMAN calculation with a TAPE13 restart file from a previous geometry optimization with the same basis, accuracy parameters, and density functional.

Atomic coordinate displacements in a RAMAN calculation must be Cartesian, not Z-matrix. Furthermore, the current implementation does not yet support constrained displacements, i.e. you must use *all* atomic coordinate displacements. However, one can calculate Raman for selected frequencies, see next section.

The alternative Raman implementation with the *AORESPONSE* (page 192) offers some unique features like lifetime options, hybrids support, spin-orbit coupling.

```
AORESPONSE
 RAMAN
END
```

6.1.3 Raman Intensities for Selected Frequencies

The RAMANRANGE keyword can be used to calculate Raman intensities for a range of frequencies only. Recommended to be used in the case one use a t21 as a restart file, which has frequencies on them. Using this option is a fast alternative for the existing method of calculating Raman intensities. By default RAMANRANGE uses the RESPONSE key, however, in case of hybrids or spin-orbit coupling RAMANRANGE will use the AORESPONSE key. Raman icw hybrids icw spin-orbit coupling is not supported in ADF. Note: while RAMANRANGE works in combination with `SYMMETRY NOSYM`, it might not if the user explicitly specifies any other symmetry via the *SYMMETRY* (page 28) key. The input keyword is as follows:

```
RAMANRANGE low high {NUM=num DISRAD=disrad}
```

low, high Two values defining an interval of frequencies to calculate the Raman intensities for. The Raman intensities are calculated by numerical differentiation of the polarizability tensor. Only frequencies within the interval that are known to be Raman-active will be included. This means that 2**N* single-point TDDFT calculations will be performed, where *N* is the number of Raman-active frequencies within the range. Imaginary frequencies are specified using negative values, which is consistent with the notation adopted within ADF.

num Num is an integer number specifying how many points are to be used for numerical differentiation: 2, 4, or 6. The default value is 2.

disrad Disrad specifies the step size (in Angstrom) to be made in each direction for 2-point differentiation. For the 4-point differentiation the maximum deviation from the equilibrium geometry will be twice as large as for the 2-point one and so on. The default value for disrad is the same as used for numerical frequencies.

Main advantages of the method:

- Full symmetry at each displaced geometry is used, which does not only speeds the calculation up but also makes it more accurate.

- Only Raman-active modes are included in the calculation, which may save a lot of time for molecules with symmetry.

- There is no need to recalculate frequencies if you already have a t21 file with them as it can be used as a restart file.

For static Raman scattering ($\omega = 0$) one does not need to add the RESPONSE (or AORESPONSE) block key. However, for the calculation of the frequency dependent Raman scattering the following RESPONSE block key is needed in the input:

```
RESPONSE
 RAMAN
 Nfreq 1
 FrqBeg Laserfreq
END
```

Frequencies or wavelengths The number of frequencies Nfreq should be 1. With subkey Frqbeg the value of the Laser frequency value (Laserfreq) can be given. The unit of `Frqbeg` is eV.

For the AORESPONSE key (see also next section), this would be:

```
AORESPONSe
 RAMAN
 Frequency 1 freq1 units
 {LIFETIME width}
END
```

Frequency 1 freq1 units To calculate time-dependent properties, one needs to specify the frequency of perturbation field. Here 1 frequency freq1 is used. The last item on the line specifies the units and is one of EV, HARTREE, ANGSTROM.

LIFETIME width' Optional. See also next section, and the *AORESPONSE documentation* (page 192).

6.1.4 Resonance Raman: excited-state finite lifetime

In this method (Ref.[*266* (page 922)]) the resonance Raman-scattering (RRS) spectra is calculated from the geometrical derivatives of the frequency-dependent polarizability. The polarizability derivatives are calculated from resonance

polarizabilities by including a finite lifetime (phenomenological parameter) of the electronic excited states using time-dependent density-functional theory.

It is similar to the simple excited-state gradient approximation method (see next section) if only one electronic excited state is important, however, it is not restricted to only one electronic excited state. In the limit that there is only one possible state in resonance the two methods should give more or less the same results. However, for many states and high-energy states and to get resonance Raman profiles (i.e., Raman intensities as a function of the energy of the incident light beam) this approach might be more suitable. The resonance Raman profiles in this approach are averaged profiles since vibronic coupling effects are not accounted for. At the moment this method needs numerically calculated frequencies in Cartesian coordinates. The method described in the next section can use a basis of normal coordinates rather than Cartesian coordinates, so that in that method the calculation can be restricted to a couple of modes.

```
GEOMETRY
 FREQUENCIES
 End
END
AORESPONSE
 RAMAN
 FREQUENCY  1 freq1 units
 LIFETIME width
END
```

The RRS is not calculated if one uses symmetric displacements in the numerically calculated frequencies or if one uses analytically calculated frequencies. However, one can use RAMANRANGE, see previous section. The *documentation of the AORESPONSE key* (page 192) explains in more detail the meaning of the subkeywords in the block key AORESPONSE, which are required to calculate RRS. Similarly to the RESPONSE-Raman module, the AOREPONSE-Raman only works with one frequency.

6.1.5 Resonance Raman: excited-state gradient

According to a the time-dependent picture of resonance-Raman (RR) scattering the relative intensities of RR scattering cross sections are, under certain assumptions, proportional to the square of the excited-state energy gradients projected onto the ground-state normal modes of the molecule (see Ref. [*202* (page 918)]). For an alternative implementation of RR scattering using a finite lifetime of the excited states, and a discussion of some of the differences, see the previous section.

The excited-state gradients which are needed in this method can be computed numerically by ADF's VIBRON module, which is invoked by selecting the VIBRON runtype in the GEOMETRY block key, the use of the VIBRON block key and the EXCITATION block key:

```
GEOMETRY
   VIBRON
END
```

```
VIBRON
 NMTAPE filename
 RESRAMAN
 {...
 ..}
END
```

```
EXCITATIONS
   LOWEST nlowest
END
```

The VIBRON module always requires an EXCITATIONS input block, in which the total number of excited states to be calculated must be specified. NMTAPE is the only obligatory keyword for the VIBRON module. It specifies the name of a TAPE21 file from a previous frequency calculation. This TAPE21 file is needed to read the normal modes w.r.t. which the derivatives are computed. I.e., a separate frequency calculation must be carried out first. The second subkeyword RESRAMAN invokes the resonance Raman calculation. Note that the VIBRON module is not suited for open-shell TDDFT.

Resonance Raman for several excited states

The numerical evaluation of resonance Raman intensities has the advantages that

- Relative Intensities can be computed for several excited states at a time, since all excitation energies are determined simultaneously.

- Intensities can be computed for a selected no. of modes.

The intensities calculated for two different states cannot directly be compared, since the excited-state gradients only provide relative intensities for each excited state in resonance. For RR intensities from different excited states, also other quantities play a role. The most important one is the transition dipole moment to the excited state in resonance, which enters the intensity expression with to the fourth power.

Restrictions: (avoided) crossings between excited-states

The numerical calculation of excited-state gradients has a number of advantages, but also a possible problem: If the step size is chosen too large, or if there are close-lying excited-states, then the order of the excited states can change. For such cases, the excited-state gradient method to estimate relative RR intensities is not reliable: If states with different electronic character (but of the same symmetry) are close in energy, this will cause an avoided crossing. If the numerical derivatives are, in this case, computed w.r.t. the adiabatic states, they will probably not reflect the true situation. Especially if the coupling matrix elements between the two excited states is small, the spectroscopic properties often behave as if there is no avoided crossing, i.e., according to the diabatic states. Such cases should be handled with extreme care, since it is often not possible in advance to see whether the adiabatic or the diabatic picture should be invoked.

Because of the possible (avoided) crossings the user must make sure that always enough excited states are calculated to include the state(s) of interest. E.g., if the resonance Raman intensities are required for the first excited state, also some higher excited states have to be included in the excitation calculation, as the first excited state at the ground-state equilibrium might be higher in energy for displaced structures.

Restrictions: results not trustworthy for higher excited states

Users should be aware of another technical point: Excited states are usually calculated from a Davidson diagonalization procedure, i.e., only a small number of eigenvalues and eigenvectors describing the lowest excitations are obtained. During finite displacements, some of the higher calculated states might leave the calculated energy window, while others enter it. Hence, the character of some of the higher calculated states can change. In such a case, the numerical differentiation based on a (simple) diabatic pictures will fail for the higher states, since no mapping between the excited states for reference (equilibrium) and displaced structure can be carried out.

The solution is rather simple: Users should always ask for more excited states than they are actually interested in, and discard the data for higher states, in particular for those which could not successfully be mapped for displaced structures (look for messages in the output like 'State No. X cannot be expressed in terms of reference states').

For advanced users it should be mentioned that it is possible to set an energy window within the range of states calculated, and only the states within this energy window will be taken into account in the evaluation. See the subkey ELTHRESH and EUTHRESH of the block key VIBRON.

Furthermore, it is possible to pick out certain states from this energy window, and only perform the mapping (and diabatization, if requested) and differentiation for them. See the subkey SELSTATE of the block key VIBRON.

Advanced Restarts

In some cases, it only becomes obvious which states have to be included in a (simple) diabatization after excitation energies for all displaced structures are calculated. Therefore, the selected states and the energy window settings can also be adjusted in a restart (with the usual restart key) after all single-point calculations are done. However, this is only possible if all raw data are saved to TAPE21, which might be an enormous amount of data.

Therefore, the user has to specify the subkeyword SAVERAWDAT of the key VIBRON in the production run, and the subkeyword USERAWDAT of the key VIBRON in the (evaluation) restart. Such a restart does not invoke any new SCF and will, therefore, typically only take a couple of seconds or minutes. If SAVERAWDAT is not specified, restarts are still possible, but the energy window cannot be adjusted differently, and no new state selection can be performed.

Resonance Raman Input options

A number of options are available for the VIBRON module, most of which are for special applications. All the options mentioned below have to appear in the VIBRON block. The VIBRON module always requires an EXCITATION input block, in which the total number of excited states to be calculated must be specified.

```
VIBRON
 NMTAPE filename
 RESRAMAN
 {DISPTYPE disptype}
 {STPSIZE stpsize}
 {ONLYSYM}
 {NOTONLYSYM}
 {DOMODES list}
 {DONTMODES list}
 {DSCHEME dscheme}
 {EUTHRES euthres}
 {ELTHRES elthres}
 {SELSTATE list}
 {SAVERAWDAT}
 {USERAWDAT}
END
```

The most important ones in connection with RR calculations are:

NMTAPE filename NMTAPE is the only obligatory keyword for the VIBRON module. It specifies the name of a TAPE21 file from a previous frequency calculation. This TAPE21 file is needed to read the normal modes w.r.t. which the derivatives are computed. I.e., a separate frequency calculation must be carried out first.

RESRAMAN The second subkeyword RESRAMAN invokes the resonance Raman calculation.

DISPTYPE disptype Select type of displacement steps; possible values are:

- MASSWE: steps in terms of mass-weighted normal mode vectors [default]

- CARTES: steps in terms of Cartesian normal mode vectors

- REDUCE: steps in terms of reduced normal modes

- ENERGY: steps in terms of expected energy change (according to harmonic approximation)

- ZPELEV: like energy, but energy is expressed in ZPE units

STPSIZE stpsize Sets the step size for the numerical differentiation in the default unit for the given DISPTYPE

ONLYSYM Calculate derivatives only for totally symmetric modes (this is useful since this RR estimate only holds for Franck-Condon type Raman scattering, which is zero for non-symmetric modes). This option is ON per default in RR calculations, it can be switched off with the key NOTONLYSYM.

DOMODES list Calculate derivatives only for the normal modes with numbers mentioned in list.

DONTMODES list Calculate derivatives for all normal modes except the ones with numbers mentioned in list.

DSCHEME dscheme The type of differentiation to be used can be set. Three different values for dscheme are available:

ELCHAR A simple diabatic picture in which adiabatic states are mapped to the adiabatic states for the references structure based on a maximum transition density overlap criterion [default].

EIGVEC A diabatic picture in which a diabatization is carried out as explained in Ref. [203 (page 918)]. I.e. the energies used here are the diagonal elements of the potential energy matrix for the nuclear Schrödinger Equation.

ADIABS The adiabatic picture; can only be used if the symmetry of the excited states is supplied from a separate calculation, since the VIBRON module cannot check which states are allowed to cross as no symmetry is used in the excitation calculations. Consult the ADF-VIBRON manual [204 (page 918)] for information.

ELTHRESH elthresh EUTHRESH euthresh For advanced users it is possible to set an energy window within the range of states calculated, and only the states within this energy window will be taken into account in the evaluation. - elthresh: lower bound in eV, default 0 eV. - euthresh: upper bound in eV, default 1.0E10 eV.

SELSTATE list For advanced users it is furthermore possible to pick out certain states from the energy window, and only perform the mapping (and diabatization, if requested) and differentiation for them. Here list includes the number (in ascending excitation energy) of the excited state at the reference (equilibrium) structure.

SAVERAWDAT All raw data are saved to TAPE21, which might be an enormous amount of data. The selected states and the energy window settings can then be adjusted in a restart (with the usual restart key) and the inclusion of the subkey USERAWDATA after all single-point calculations are done.

USERAWDAT All raw data are read from a previous calculation. The selected states and the energy window settings can now be adjusted. You need to invoke the usual restart key. Such a restart does not invoke any new SCF and will, therefore, typically only take a couple of seconds or minutes. If SAVERAWDAT is not specified in the previous calculation, restarts are still possible, but the energy window cannot be adjusted differently, and no new state selection can be performed.

6.1.6 VROA: (Resonance) vibrational Raman optical activity

In ADF2010 a method is implemented to calculate both on- and off-resonance vibrational Raman optical activities (VROAs) of molecules using time-dependent density functional theory, see Ref. [306 (page 925)]. This is an extension of a method to calculate the normal VROA by including a finite lifetime of the electronic excited states in all calculated properties. The method is based on a short-time approximation to Raman scattering and is, in the off-resonance case, identical to the standard theory of Placzek. The normal and resonance VROA spectra are calculated from geometric derivatives of the different generalized polarizabilites obtained using linear response theory which includes a damping term to account for the finite lifetime. Gauge-origin independent results for normal VROA have been ensured using either the modified-velocity gauge or gauge-included atomic orbitals. In ADF2016 the velocity gauge tensors required for the calculation of VROA are now correctly calculated with the life time damping parameter. With these complex tensors fixed, resonance VROA intensities are now origin invariant in the velocity gauge, see also Ref. [382 (page 929)].

For the normal VROA use numerical frequencies, and the subkey VROA of the key AORESPONSE. Example input:

```
GEOMETRY
 Frequencies
 End
END
AORESPONSE
 VROA
 scf  converge 1d-6 iterations 100
 frequency 1 5145 Angstrom
 ALDA
 EL_DIPOLE_EL_DIPOLE VELOCITY
```

```
EL_DIPOLE_EL_QUADRUPOLE VELOCITY
 EL_DIPOLE_MAG_DIPOLE VELOCITY
END
```

For the resonance VROA use numerical frequencies, and the subkey VROA and LIFETIME of the key AORESPONSE. Example input:

```
GEOMETRY
 Frequencies
 End
END
AORESPONSE
 VROA
 scf  converge 1d-6 iterations 100
 frequency 1 5.15462 eV
 lifetime 0.0037
 ALDA
 EL_DIPOLE_EL_DIPOLE VELOCITY
 EL_DIPOLE_EL_QUADRUPOLE VELOCITY
 EL_DIPOLE_MAG_DIPOLE VELOCITY
END
```

In combination with the DIM/QM model one might simulate surface-enhanced Raman optical activity (SEROA), see *section on DIM/QM* (page 84) and Ref. [*382* (page 929)].

If you use the (deprecated) STO fit option, it's recommendend to include the keyword `FitAOderiv` in the `AORESPONSE` block.

6.1.7 Vibrational Circular Dichroism (VCD) spectra

The following keyword enables calculation of rotational strength during an analytical frequencies calculation:

```
VCD
```

It is important to note that the VCD keyword only works in combination AnalyticalFreq and symmetry NOSYM. The frozen core approximation and open shell systems are not supported.

```
AnalyticalFreq
End
SYMMETRY NOSYM
Basis
  type TZP
  core NONE
End
```

The VCD intensities are calculated using Stephens' equations for VCD. For the calculation of the atomic axial tensors (AATs), analytical derivatives techniques and London atomic orbitals (the so called GIAO) are employed. As a result the calculated rotational strengths are origin independent, and therefore the common origin gauge is used [*216* (page 919)].

Calculation of the AATs requires an analytical frequencies calculation. This limits the choice of functionals that can be used for VCD calculations. See the ANALYTICALFREQ keyword for a complete list of the available functionals. The VCD calculations can be done immediately after a geometry optimization, just like analytical frequencies calculations.

The accuracy of the vibrational rotational strengths are determined by the accuracy of the harmonic force field, atomic polar tensors (APTs) and AATs. The most critical parameter being the harmonic force field. Thus, for a fair comparison with experimental data, accurate geometries and functionals that yield accurate force fields (e.g. BP86, OLYP, etc)

should be used. Our tests showed that the BP86 functional in combination with TZP basis sets is always a safe choice. For a comparison of VCD spectra calculated with various functionals (e.g BP86, OLYP, BLYP, B3PW91 and B3LYP) see [*216* (page 919)]. Regarding the geometries, we recommend the following strict settings, 10^{-4} for the geometry convergence of the gradients, and BeckeGrid quality good. The default settings should be used for the calculation of the frequencies.

By default, only the vibrational rotational strengths are printed in the ADF output file. The AATs can also be printed by specifying the keyword:

```
PRINT VCD
```

VCDtools (page 289) is a program that can be used to do an analysis of the VCD spectrum. *VCDtools* (page 289) can be used with the ADF-GUI module ADFspectra.

6.1.8 Vibrationally resolved electronic spectra

See *the section on vibrationally resolved electronic spectra* (page 184).

6.2 Time-dependent DFT

Excitation energies, frequency-dependent (hyper) polarizabilities, Van der Waals dispersion coefficients, higher multipole polarizabilities, Raman scattering intensities and depolarization ratios of closed-shell molecules are all available in ADF [*71* (page 910), *72* (page 911)] as applications of time-dependent DFT (TDDFT) ; see [*73* (page 911)] for a review.

New in ADF2004.01 is the calculation of circular dichroism (CD) spectra, and the calculation of the optical rotation (dispersion).

Starting from the ADF2005.01 version it is possible to calculate excitation energies for open-shell systems with TDDFT, including spin-flip excitation energies. New in ADF2005.01 is the possibility to use time-dependent current-density functional theory (TDCDFT).

New in ADF2006.01 is the possibility to calculate excitation energies for closed-shell molecules including spin-orbit coupling.

New in ADF2008.01 is the possibility to calculate lifetime effects in (dynamic) polarizabilities (AORESPONSE key).

The input description for these properties is split in three parts: (a) general advice and remarks, (b) excitation energies, and (c) frequency-dependent (hyper) polarizabilities (two alternative implementation: RESPONSE key and AORESPONSE key) and related properties.

Tip: See also the ADF Advanced TDDFT tutorial: TDDFT Study of 3 different Dihydroxyanthraquinones

---

6.2.1 General remarks on the Response and Excitation functionality

`Symmetry` As in calculations without TDDFT the symmetry is automatically detected from the input atomic coordinates and need not be specified, except in the following case: infinite symmetries cannot be handled in the current release (ATOM, C(lin), D(lin)). For such symmetries a subgroup with finite symmetry must be specified in the input. The usual orientation requirements apply. If higher multipole polarizabilities are required, it may also be necessary to use a lower subgroup (the program will stop with an error message otherwise). For verification of results one can always compare to a NOSYM calculation.

Closed-shell The current implementation often supports only closed-shell molecules. If occupation numbers other than 0 or 2 are used the program will detect this, (but only at a later stage of the calculation) and abort. All 'RESPONSE' calculations must be spin-restricted.

Open-shell Excitation energies can be obtained for open-shell systems in a spin-unrestricted TDDFT calculation. Spin-flip excitation energies can only be obtained in a spin-unrestricted TDDFT calculation.

Atomic coordinates in a RAMAN calculation Atomic coordinate displacements in a RAMAN calculation must be Cartesian, not Z-matrix. Furthermore, the current implementation does not yet support constrained displacements, i.e. you must use *all* atomic coordinate displacements.

Use of diffuse functions The properties described here may require diffuse functions to be added to the basis (and fit) sets. Poor results will be obtained if the user is unaware of this. As a general rule, diffuse functions are more important for smaller than for larger molecules, more important for hyperpolarizabilities than for normal polarizabilities, more important for high-lying excitation energies (Rydberg states) than for low-lying excitations, more important for higher multipole polarizabilities than for dipole polarizabilities. The user should know when diffuse functions are required and when they are not: the program will not check anything in this respect. For example, in a study on low-lying excitation energies of a large molecule, diffuse functions will usually have little effect, whereas a hyperpolarizability calculation on a small molecule is pointless unless diffuse functions are included. Diffuse even tempered basis sets are included in the ET/ directory in $ADFHOME/atomicdata), for the elements H-Kr. Somewhat older basis sets can be found in the Special/Vdiff directory in $ADFHOME/atomicdata. For other atoms, the user will have to add diffuse basis and fit functions to the existing data base sets. It is not necessary to start from basis V as was done for the basis sets in Special/Vdiff. For example, for heavier elements it may be a good idea to start from the ZORA/QZ4P basis sets. It may be expected that even more extensive basis sets will come available in the future, when usage and experience increase.

Linear dependency in basis If large diffuse basis sets are used, or if diffuse functions are used for atoms that are not far apart the calculation may suffer from numerical problems because of (near-) linear dependencies in the basis set. The user should be aware of this danger and use the DEPENDENCY key to check and solve this.

The LINEARSCALING input keyword For reasons of numerical robustness and safety rather strict defaults apply for the neglect of tails of basis and fit functions (see the key LINEARSCALING) in a Response or Excitation calculation. This may result in longer CPU times than needed for non-TDDFT runs, in particular for larger molecules. Possibly this precaution is not necessary, but we have not yet tested this sufficiently to relax the tightened defaults.

Relativistic effects The Response and Excitations options can be combined with scalar relativistic options (ZORA or Pauli). The one-electron relativistic orbitals and orbital energies are then used as input for the property calculation. Spin-orbit effects have been incorporated only in this part of the code (excitation energies). In case of a ZORA calculation, the so-called 'scaled' orbital energies are used as default.

Choice of XC potential For properties that depend strongly on the outer region of the molecule (high-lying excitation energies, (hyper) polarizabilities), it may be important to use a XC potential with correct asymptotic behavior (approaching -1/r as r tends to infinity). Finally, several asymptotically correct XC potentials have been implemented in ADF, like the LB94 potential [15 (page 907)] and the statistical average of orbital potentials SAOP [244 (page 921), 17 (page 908)]. SAOP is recommended. Because of the correct asymptotic behavior the SAOP potential (and the LB94 potential) can describe Rydberg states correctly. The potentials based upon the so-called GRadient regulated seamless connection of model potentials (GRAC) for the inner and the outer region [16 (page 907), 18 (page 908)] have similar performance to SAOP, but have the disadvantage that the ionization energy of the molecule has to be used as input.

With the SAOP and GRAC functionals for the potential (as well as for LB94), the XC potential is computed from the exact charge density for reasons of stability and robustness (whereas for other functions the (cheaper) fit density is used). This implies that computation times may be longer. Another 'side effect' is that, since there is no energy expression corresponding to these potentials, the final (bonding) energy of such calculations uses another GGA and hence the energy result is not (exactly) consistent with the SCF procedure. Note, finally,

that these potentials have been found to be not suitable for geometry optimizations because they maybe are not sufficiently accurate in the bonding region, see the discussion of the XC input key. Applications with SAOP to (hyper)polarizabilities and excitation energies, also for Rydberg transitions, can be found in [*17* (page 908)] and with SAOP and Becke-Perdew-GRAC in [*16* (page 907), *18* (page 908)]. Applications with the old LB94 potential to response calculations can be found in [*74* (page 911)] (polarizabilities), [*75-77* (page 911)] (hyper-polarizabilities), [*78* (page 911)] (high-lying excitation energies), [*79* (page 911)] (multipole polarizabilities and dispersion coefficients).

XC kernel If most cases the adiabatic local density approximated (ALDA) kernel is used in the TDDFT functionality. In case of calculating excitation energies with a hybrid functionals the Hartree-Fock percentage times the Hartree-Fock kernel plus one minus the Hartree-Fock percentage times the ALDA kernel is used. For the LibXC range separated functionals, like CAM-B3LYP, starting from ADF2016.102 the kernel consists of range separated ALDA plus the kernel of the range separated exact exchange part. In ADF2016.101 the kernel for LibXC range separated functionals, like CAM-B3LYP, was using a 100% ALDA plus range separated exact exchange kernel (the ALDA part was not range-separated corrected).

For excitation energy calculations in some cases the full (non-ALDA) kernel can be evaluated, see the *full XC kernel description* (page 164). The Full kernel can not be used in combination with symmetry or excited state geometry optimizations.

COSMO The COSMO model for solvation is a cheap method to include solvation effects in the TDDFT applications, see the SOLVATION key. Note that in TDDFT calculations one may have to use two dielectric constants. The reason is that the electronic transition is so fast that only the electronic component of the solvent dielectric can respond, i.e., one should use the optical part of the dielectric constant. This is typically referred to as non-equilibrium solvation. The optical dielectric constant can be obtaining from the (frequency dependent) refractive index n of the solvent as: $\epsilon_{opt} = n^2$. One can use the argumen NEQL in the subkey SOLV of the key SOLVATION to set a value for the optical dielectric constant. No dielectric constant in the response might be closer to the optical dielectric constant than using the full dielectric constant, This can be achieved if one includes the subkey NOCSMRSP in the block key SOLVATION, such that the induced electronic charges do not influence the COSMO surface charges. However, if one does geometry optimization then the full dielectric constant should be used in the TDDFT simulations since the solvent dielectric now has time to fully respond. The default is that the full dielectric constant is used in the TDDFT calculations.

Accuracy check list As mentioned before, the TDDFT module is relatively new and not extensively tested for a wide range of applications. Therefore, we strongly recommend the user to build experience about aspects that may affect the accuracy of TDDFT results. In particular we advise to 'experiment' with

- Varying integration accuracy
- Varying the SCF convergence
- Varying the ORTHONORMALITY and TOLERANCE values in an Excitation calculation
- Varying the linearscaling parameters
- Using diffuse functions
- Using the Dependency key
- Applying the ZORA relativistic corrections for molecules containing heavy nuclei
- Using an asymptotically correct XC potential such as SAOP

6.2.2 Analysis options for TDDFT (excitation energies and polarizabilities)

Several options are available to obtain more detailed results than a few bare numbers for excitation energies, oscillator strengths, transition dipole moments, and (hyper)polarizabilities. For a zero-order understanding of which occupied and virtual orbitals play an important in the polarizability or intensity of an absorption peak,

it may be useful to know the values of the dipole matrix elements between (ground-state) occupied and virtual Kohn-Sham orbitals. If these dipole matrix elements are large for a particular occupied-virtual orbital pair, then this pair is almost certainly of great importance for the whole spectrum or polarizability. This information can be obtained by specifying somewhere in the input file (but NOT inside the RESPONSE or EXCITATION block keys):

```
PRINT DIPOLEMAT
```

6.2.3 Time-dependent Current DFT

The time-dependent current-density-functional (TDCDFT) implementation is built entirely upon the normal TDDFT implementation. Therefore all general remarks that are made for the TDDFT part of the program are also valid for TDCDFT. Only the polarizability and excitation energies of closed shell molecules can be calculated with TDCDFT in the present implementation.

If TDCDFT is used together with the ALDA functional (NOVK option) it will give the same results for the polarizability and excitation energies as TDDFT in a complete basis set. TDCDFT in ADF by default uses the VK functional [*160* (page 916), *161* (page 916)], since this is the only current dependent functional that is known presently. Many aspects of the functional are still unknown and the functional should therefore be used with caution. The user is referred to the references for more information on when the VK functional gives good results and when not.

For more information on the implementation and applications of the TDCDFT and the VK functional please read the references: [*162-165* (page 916)]. For more details on the theory and implementation in ADF see: [*166* (page 916)].

To activate TDCDFT and the VK functional one should add the following block key to the input file:

```
CURRENTRESPONSE
END
```

To calculate the polarizability the keyword Response can be used with the following options:

```
RESPONSE
  ALLCOMPONENTS
  Nfreq Nfreq
  FrqBeg FirstFreq
  FrqEnd LastFreq
END
```

The block key EXCITATION can be used with all of its options.

In ADF2012 the block keyword AORESPONSE can also be used with the Vignale-Kohn functional. The current-density is generated on the fly but otherwise the computation is based on the time dependent density response.

In default the VK functional will be applied where the NCT parametrization [*167* (page 916)] is chosen for the transverse exchange-correlation kernel for the polarizability and singlet excitation energies (giving the best results for the systems studied so far). For triplet excitation energies the only available parametrization will be used [*168* (page 916)]. This option is not tested much and the results are in general much worse than ALDA [*166* (page 916)]. It is therefore suggested that VK is not used to calculate triplet excitation energies.

In the output the polarizability tensor (in case of an ALLCOMPONENTS calculation) has a different shape, the results are printed in the more intuitive order x, y, z, instead of y, z, x that the TDDFT implementation uses.

The following subkeys are available within the datablock of CURRENTRESPONSE

```
CURRENTRESPONSE
  QIANVIGNALE
  NOVK
END
```

QIANVIGNALE The QV parametrization [*168* (page 916)] will be used for the transverse exchange-correlation kernel instead of NCT.

NOVK TDCDFT will be applied with the ALDA functional instead of the VK functional. In a complete basis this will give the same results as a TDDFT calculation.

Magnetic properties within TDCDFT

It is now possible to calculate several magnetic properties via TDCDFT, among which magnetizabilities, rotational g-tensors, NMR shielding constants, specific rotations and circular dichroism. This alternative method is based on the diamagnetic-current sum rule, which consists of rewriting the diamagnetic current in terms of the response functions. This is the magnetic analogue of the conductivity sum rule, which is invalid when an external magnetic field is applied.

Details on the method can be found in these references: [*442* (page 933), *443* (page 933)]. Further details regarding the implementation can be found in the following PhD thesis as well: [*444* (page 933)].

In order to obtain these properties, the following subkeys can be used:

```
CURRENTRESPONSE
  MAGNET
  GTENSOR
  NMRSHIELDING
  CDSPEC
  DAMPING Damping
END
```

Depending on which property is calculated, the program chooses a reference point for the external vector potential (see references). For the magnetizability, the center of electronic charges is chosen, while for the rotational g-tensor, the center of nuclear charges is picked up. This choice has no influence in the static case, and has no influence neither for the specific rotations nor for the circular dichroism. It actually only affects the dynamical magnetizabilities (the rotational g-tensor being always computed at zero frequency). Below is a brief description of each subkey and how to use them.

MAGNET The (static or dynamical) magnetizability tensor.

GTENSOR The rotational g-tensor, calculated directly from the magnetizability.

NMRSHIELDING The NMR shielding tensor. Care should be taken when using this keyword, though, as it was observed that the results failed to match experimental and other theoretical studies when using standard basis sets except for the lightest elements. Only when using very large basis sets should the results be trusted.

CDSPEC Calculates both the specific rotation and the circular dichroism. This keyword should be used in combination with DAMPING if one wants to simulate CD spectra. The output spectrum will exhibit relatively sharp peaks. Linear-interpolating the results with a third-tier program is then necessary. This is done on purpose, so that the final spectrum could be broadened easily (only the absolute amplitudes of the peaks will be affected by this broadening).

DAMPING Damping Adds a (fixed) damping factor to the response function. This allows one to avoid divergences that may arise from the response function when close to resonance, hence helping the self-consistent cycle converge. The value for Damping should be given in eV. A value of 0.1 eV seems to be reasonable.

6.3 Excitation energies: UV/Vis, X-ray, CD, MCD

Ultraviolet-visible (UV/Vis) spectroscopy studies electronic excitations of valence electrons, whereas X-ray spectroscopy studies electronic excitations of core electrons. Excitation energies and oscillator strengths are all available in ADF as applications of time-dependent DFT (TDDFT). Excitation energies can be calculated for closed-shell as

well as for open-shell molecules. It is also possible to include spin-orbit coupling and to calculate core excitations (X-ray absorption spectra). Circular dichroism (CD) is the differential absorption of left- and right-handed circularly polarized light.

Tip: See also the ADF Advanced TDDFT tutorial: TDDFT Study of 3 different Dihydroxyanthraquinones

---

6.3.1 Excitation energies, UV/Vis spectra

You can perform a calculation of singlet-singlet and singlet-triplet excitation energies of a closed-shell molecule by supplying in the input file the block key EXCITATION. See the next sections for settings of technical parameters, the calculation of excitation energies for open shell molecules, inclusion of spin-orbit coupling, and the calculation of CD spectra.

```
EXCITATIONS
 EXACT
   {IRREP1 N1}
   {IRREP2 N2}
 END
 DAVIDSON
   {IRREP3 N3}
   {IRREP4 N4}
 END
 ALLOWED
 ONLYSING
 ONLYTRIP
 LOWEST nlowest
End
```

Several options can be addressed with subkeys in the data block. This functionality is based on TDDFT and consequently has a different theoretical foundation than the SCF techniques described elsewhere in this User's Guide. Two possible ways are available to solve the eigenvalue equation from which the excitation energies and oscillator strengths are obtained, of which the iterative Davidson procedure is the default. In this case, the program needs to know how many excitation energies are needed per irrep, what accuracy is required, and what type of excitation energies are required (singlet-singlet or singlet-triplet). Suitable defaults have been defined for all of these. Each of these points is discussed below.

Exact diagonalization vs. iterative Davidson procedure The most straightforward procedure is a direct diagonalization of the matrix from which the excitation energies and oscillator strengths are obtained. Since the matrix may become very large, this option is possible only for very small molecules. It can be activated by specifying the block EXACT as one of the subkeys in the Excitations data block. The default is the iterative Davidson method. A few of the lowest excitation energies and oscillator strengths are then found within an error tolerance. An advantage of the EXACT option is that additional information is produced, such as the Cauchy coefficients that determine the average dipole polarizability. The EXACT option can not be used in unrestricted calculations.

Singlet versus triplet By default, the singlet-singlet and singlet-triplet excitation energies are both calculated. The singlets are handled first, then the corresponding triplet excitation energies. One can skip one of these two parts of the calculation by specifying either ONLYSING or ONLYTRIP as a subkey in the data block.

In case of a calculation including spin-orbit coupling one can not separate the singlet-singlet and singlet-triplet excitations. The subkeys ONLYSING and ONLYTRIP are misused in this case to do a spin-restricted calculation, or a spin-polarized calculation, respectively. One should in fact only use the results of the spin-polarized calculation.

Dipole-allowed versus general excitations. If you are interested in the optical absorption spectrum, you may not want to compute singlet-triplet excitation energies, nor singlet-singlet excitation energies which, by symmetry, have zero oscillator strengths. This subkey should not be used in case of spin-orbit coupling. The subkey ALLOWED tells ADF to treat only those irreducible representations for which the oscillator strengths will be nonzero. Of course, the oscillator strengths may still be negligibly small. The ALLOWED subkey automatically implies ONLYSING. The simplest, fastest, and recommended way to obtain information about the ten lowest dipole-allowed excitation energies would be:

```
EXCITATIONS
 ALLOWED
 LOWEST 10
END
```

Which excitation energies and how many? The user can specify how many excitation energies per irrep should be calculated. If no pertaining input is available the program determines these numbers from the smallest differences between occupied and virtual Kohn-Sham orbital energies. By default it looks at the 10 lowest orbital energy differences. This number can be modified, by specifying inside the Excitation block key, for example:

```
LOWEST 30
```

One should be aware that this procedure does not guarantee that the lowest 10 (or 30) excitation energies will actually be found, since the orbital energy difference approximation to the excitation energy is rather crude. However, if the program decides on the basis of this procedure to calculate 4 excitation energies in a certain irreducible representation, these 4 excitation energies are certainly the lowest in that particular irrep.

The user has more control when the number of excitations per irrep is explicitly specified within the EXCITATION block key by the Davidson subkey:

```
DAVIDSON
 E'' 5
 T1.u 2
END
```

The DAVIDSON sub block may contain any number of records and must end with a record END. In the subkey data block a list of irreps, followed by the number of requested excitation energies is specified. Note that the irrep name may not be identical to the usual ADF name. For example E" is called EEE in ADF. The Excitation code will skip an irrep if the label is not recognized. For multidimensional irreps, only the first column is treated, because the other would produce identical output. This implies that the oscillator strengths for E-irreps have to be multiplied by 2 and the oscillator strengths for T-irreps by 3. The ALLOWED subkey should not be used if irreps are specified with the Davidson block subkey, however, the subkey ONLYSING (or ONLYTRIP) can be used in this case.

The EXACT sub-block, mentioned already above, can also be used to treat only a few irreps instead of all. The number of excitation energies does not have to be specified then.

Some early applications of the Excitation feature in ADF can be found in Refs.[*82-87* (page 911)].

Tamm-Dancoff approximation

Excitation energies can be calculated using the Tamm-Dancoff approximation (TDA) [*158* (page 916)] if one includes, besides the EXCITATION block key, the key TDA:

```
TDA
```

Full XC kernel

With XCFUN the full (non-ALDA) kernel can be evaluated, see the *XCFUN description* (page 65). To use the non-ALDA kernel the keyword FULLKERNEL should be put in the EXCITATIONS block. FULLKERNEL can be used with GGAs (including hybrids and RS functionals) but not meta-GGAs or meta-hybrids. For some of the LibXC functionals (mostly GGA's and hybrids) it also possible to use the Full (non-ALDA) kernel. FULLKERNEL can not be used in combination with excited state geometry optimizations or other response properties. If symmetry is used, recommended is to check that SYMMETRY NOSYM gives the same results.

```
XC
  ...
  XCFUN
End
EXCITATIONS
  ...
  FullKernel
END
```

```
XC
  LibXC ...
End
EXCITATIONS
  ...
  FullKernel
END
```

Singlet-triplet excitations are now also possible with FULLKERNEL.

Plasmons in Molecules

In Ref. [*482* (page 935)] a scaling approach was used to identify plasmons in molecules. To identify plasmons in general molecules, they proposed to analyze the excitations by scaling the electron-electron interaction in the response calculation. While single-particle excitations are only slightly affected by this scaling, plasmonic excitations are sensitive to variations in the scaling parameter λ. More specifically, their squared excitation energies should increase linearly with the scaling parameter λ. Scaling of Coulomb, XC, and HF parts of the response kernel in TDDFT, can be performed using the subkeywords SCALECOUL, SCALEXC, SCALEHF in the EXCITATIONS block.

```
EXCITATIONS
    SCALECOUL  scalecoul
    SCALEXC    scalexc
    SCALEHF    scalehf
END
```

SCALECOUL scalecoul Scaling of Coulomb kernel with scale parameter scalecoul.

SCALEXC scalexc Scaling of the XC-kernel (excluding a possible HF-part) with scale parameter scalexc.

SCALEHF scalehf Scaling of the HF part of the kernel with scale parameter scalehf.

Accuracy and other technical parameters

For details regarding the (near linear scaling and parallelized) implementation, please check Refs.[*71* (page 910), *88* (page 912)].

A summary of technical parameters with their defaults is:

```
EXCITATIONS
 VECTORS 40
 TOLERANCE 1e-6
 ORTHONORMALITY 1e-8
 ITERATIONS 200
 KFWRITE 3
END
```

VECTORS vectors The maximum number of trial vectors in the Davidson algorithm for which space is allocated. If this number is small less memory will be needed, but the trial vector space is smaller and has to be collapsed more often, at the expense of CPU time. The default if usually adequate.

TOLERANCE tolerance Specifies the error tolerance in *the square* of the excitation energies in Hartree units. The default is probably acceptable but we recommend that you verify the results against a stricter default (e.g. 1e-8) for at least a few cases.

ORTHONORMALITY orthonormality The Davidson algorithm orthonormalizes its trial vectors. Increasing the default orthonormality criterion increases the CPU time somewhat, but is another useful check on the reliability of the results.

ITERATIONS iterations The maximum number of attempts within which the Davidson algorithm has to converge. The default appears to be adequate in most cases.

KFWRITE kfwrite If kfwrite is 0 then do not write contributions, transition densities, and restart vectors to TAPE21, since this can lead to a huge TAPE21, especially if many excitations are calculated. Default value kfwrite is 3, which means that contributions, transition densities, and restart vectors are written to TAPE21.

6.3.2 Excitation energies for open-shell systems

Excitation energies can be obtained for open-shell systems in a spin-unrestricted TDDFT calculation [*154* (page 915)]. To perform an open-shell TDDFT calculation one just needs to do an unrestricted SCF calculation and use the EXCI-TATION keyword. Presently the excitation energies can only be found with Davidson's procedure. In case of spin-orbit coupling, see the section on *approximate spin-orbit coupled excitation energies open shell molecule* (page 170).

The printed symmetry in the output in TDDFT calculations is actually the symmetry of transition density. For closed-shell systems, the symmetry of the excited state is the same as the symmetry of the transition density, while for open-shell systems, the symmetry of the excited states is the direct product between the symmetry of the transition density and the ground state symmetry. Note that the ground state symmetry of an open shell molecule is not necessarily A1.

For degenerate representations such as the 2-dimensional E-representations or the 3-dimensional T-representations, the occupation should be either fully occupied or zero. For example, for an orbital in an E-representation the α and β occupation number should be either 2 or 0. The α occupation number can of course be different from the β occupation number.

As for the spin-state, the general rule is that if the excited state mainly results from transitions from the singly occupied orbitals to virtual orbitals or from fully occupied orbitals to the singly occupied orbitals, the spin state of the excited state should roughly be the same as that of the ground state. However, if the excited state mainly comes from transitions from fully occupied orbitals to virtual orbitals, the spin state of the excited state are usually a mixture since TDDFT can only deal with single excitations within adiabatic approximation for the XC kernel [*155* (page 916)]. Sometimes we just suppose the spin state of this kind of excited states to be the same as that of ground state [*154* (page 915)]. In the MO $\rightarrow$ MO transitions part for the excitations of the output file, the spin of each molecular orbitals are also specified to help assign the spin state of the excited states. The transitions are always from α spin-orbital to α spin-orbital or from β spin-orbital to β spin-orbital.

6.3.3 Spin-flip excitation energies

Spin-flip excitation energies [*156* (page 916), *157* (page 916)] can only be obtained in a spin-unrestricted TDDFT calculation. This can not be used in case of spin-orbit coupling. At present, the spin-flip excitation energies can only be calculated with Tamm-Dancoff approximation (TDA) [*158* (page 916)] and Davidson's method.

To calculate spin-flip excitation energies, one must specify two keys:

```
SFTDDFT
TDA
```

anywhere in the input file in addition to the EXCITATION block keyword.

In spin-flip TDDFT, the XC kernel can be calculated directly from the XC potential. To use the LDA potential for the XC kernel, which roughly corresponds to the ALDA in ordinary TDDFT, one must specify the key

```
FORCEALDA
```

anywhere in the input file. Only calculations using the LDA potential in the SCF are fully tested. Using other GGA potentials in the SCF and using the FORCEALDA key at the same time may introduce unreasonable results, while using LB94 or SAOP potential in the SCF without the FORCEALDA key may give unstable results. Unstable results have been reported for the PW91 functional.

For open-shell molecules, spin-flip transition can result in transition to the ground state with a different S_z value, while the symmetry of the transition density is A1. The excitation energy of this transition should be zero and this can be used to test the reliability of spin-flip TDDFT.

The symmetry of the excited states can be determined in the same way as that in spin-unrestricted TDDFT calculations. As for the spin state, similar to that in the spin-unrestricted TDDFT calculations, some states may be more or less pure spin states, others may just be mixtures. The users can interpret the excited state through the transitions that contribute to this state. Note that the transitions are always from α spin-orbital to β spin-orbital in spin-flip calculations, or from β spin-orbital to α spin-orbital.

6.3.4 Select (core) excitation energies, X-ray absorption

Two methods can be used to reduce the computational costs of, for example, core excitation energies, or some other high lying excitation energy. In the state selective method scheme a guess vector for the orbital transition has to be provided. An overlap criterion is used to follow the wanted eigenvector. In this scheme the one-electron excited state configuration space remains complete, see Ref [*346* (page 927)]. In the second scheme, the range of excitations that are calculated is modified, which means that the one-electron excited state configuration space is reduced to the interesting part, see Ref. [*169* (page 916)]. The calculated excited states are more accurate with the state selective method if convergence is reached, however, the second scheme is more robust, and it is easier to find convergence.

These selection methods can, for example, also be used in case one calculates excitation energies as Kohn-Sham orbital energy differences, see subkey *SINGLEORBTRANS* (page 169) of the key EXCITATIONS.

For X-ray spectra it can be important to include quadrupole intensities.

State selective optimization excitation energies

The state selective method (key SELECTEXCITATION) can be used to reduce the computational costs of, for example, core excitation energies. In this scheme a guess vector for the orbital transition has to be provided. It should be used in combination with the Davidson method to calculate excitation energies. An overlap criterion is used to follow the wanted eigenvector. This method for state selective optimization of excitation energies is based on the method by Kovyrshin and Neugebauer, see Ref. [*346* (page 927)]. This key can also be used in case of spin-orbit coupling. The use of the key SELECTEXCITATION is similar as the use of the key MODIFYEXCITATION. However, the key

SELECTEXCITATION can not be used in combination with the key MODIFYEXCITATION. In the state selective method (key SELECTEXCITATION) the one-electron excited state configuration space remains complete, whereas it is reduced in case the scheme with the MODIFYEXCITATION key.

The starting guess vector(s) for the excitation energies can be selected, for example by selecting 1 occupied orbital and 1 virtual orbital.

```
SELECTEXCITATION
   OscStrength oscstrength
   UseOccVirtRange elowoccvirt ehighoccvirt
   UseOccVirtNumbers nrlowoccvirt nrhighoccvirt
   UseOccRange elowocc ehighocc
   UseVirtRange elowvirt ehighvirt
   UseOccupied
      irrep orbitalnumbers
      irrep orbitalnumbers
      ...
   SubEnd
   UseVirtuaL
      irrep orbitalnumbers
      irrep orbitalnumbers
      ...
   SubEnd
   UseScaledZORA
end
```

OscStrength oscstrength Use only pairs of an occupied and virtual orbital as guess vectors, for which the oscillator strength of the single-orbital transition is larger than oscstrength.

UseOccVirtRange elowoccvirt ehighoccvirt Use only pairs of an occupied and virtual orbital as guess vectors, for which the orbital energy difference is between elowoccvirt and ehighoccvirt (in hartree). If one is interested in the lowest excitation energies, use for elowoccvirt a value smaller than the HOMO-LUMO gap, and for ehighoccvirt a value larger than the energy range one is interested in.

UseOccVirtNumbers nrlowoccvirt nrhighoccvirt Use only pairs of an occupied and virtual orbital as guess vectors, for which in the sorted list of the orbital energy differences, the number of the single-orbital transition is between nrlowoccvirt and nrhighoccvirt.

UseOccRange elowocc ehighocc Use only occupied orbitals in the guess vectors which have orbital energies between elowocc and ehighocc (in hartree).

UseVirtRange elowvirt ehighvirt Use only virtual orbitals in the guess vectors which have orbital energies between elowvirt and ehighvirt (in hartree).

UseOccupied Use only the occupied orbitals in the guess vectors which are specified.

UseVirtual Use only the virtual orbitals in the guess vectors which are specified.

irrep The name of one of the irreducible representations (not a subspecies) of the point group of the system. See the Appendix for the irrep names as they are used in ADF.

orbitalnumbers A series of one or more numbers: include all numbers of the orbitals in the guess vectors that are to be used. In an unrestricted calculation the same numbers are used for the spin-α orbitals and the spin-β orbitals.

Modify range of excitation energies

The key MODIFYEXCITATION can be used to reduce the computational costs of, for example, core excitation energies. This key can also be used in case of spin-orbit coupling. The use of the key MODIFYEXCITATION is similar

---

as the use of the key SELECTEXCITATION. However, the key MODIFYEXCITATION can not be used in combination with the key SELECTEXCITATION. In the state selective method (key SELECTEXCITATION) the one-electron excited state configuration space remains complete, whereas it is (effectively) reduced in case the scheme with the MODIFYEXCITATION key.

One possibility is to allow only selected occupied orbitals and or selected virtual orbitals in the TDDFT calculations. In this scheme the complete one-electron excited state configuration space is reduced to the subspace where only the core electrons are excited, see Stener et al. [*169* (page 916)]. In the actual implementation this is done by artificially changing the orbital energies of the uninteresting occupied orbitals to a large negative value (default -1d6 Hartree), and by by artificially changing the orbital energies of the uninteresting virtual orbitals to a large positive value (default 1d6).

In ADF2010 an extra possibility is added with the new subkey UseOccVirtRange, which restricts the space of excitation energies, by allowing only pairs of occupied and virtual orbitals, for which the difference in orbital energy is between a certain range.

```
MODIFYEXCITATION
   OscStrength oscstrength
   UseOccVirtRange elowoccvirt ehighoccvirt
   UseOccVirtNumbers nrlowoccvirt nrhighoccvirt
   UseOccRange elowocc ehighocc
   UseVirtRange elowvirt ehighvirt
   UseOccupied
      irrep orbitalnumbers
      irrep orbitalnumbers
      ...
   SubEnd
   UseVirtuaL
      irrep orbitalnumbers
      irrep orbitalnumbers
      ...
   SubEnd
   SetOccEnergy esetocc
   SetLargeEnergy epsbig
   UseScaledZORA
end
```

OscStrength oscstrength Use only pairs of an occupied and virtual orbital as guess vectors, for which the oscillator strength of the single-orbital transition is larger than oscstrength.

UseOccVirtRange elowoccvirt ehighoccvirt Use only pairs of an occupied and virtual orbital, for which the orbital energy difference is between elowoccvirt and ehighoccvirt (in hartree).

UseOccVirtNumbers nrlowoccvirt nrhighoccvirt Use only pairs of an occupied and virtual orbital as guess vectors, for which in the sorted list of the orbital energy differences, the number of the single-orbital transition is between nrlowoccvirt and nrhighoccvirt.

UseOccRange elowocc ehighocc Use only occupied orbitals which have orbital energies between elowocc and ehighocc (in Hartree).

UseVirtRange elowvirt ehighvirt Use only virtual orbitals which have orbital energies between elowvirt and ehighvirt (in Hartree).

UseOccupied Use only the occupied orbitals which are specified.

UseVirtual Use only the virtual orbitals which are specified.

irrep The name of one of the irreducible representations (not a subspecies) of the point group of the system. See the Appendix for the irrep names as they are used in ADF.

orbitalnumbers A series of one or more numbers: include all numbers of the orbitals that are to be used. In an unrestricted calculation the same numbers are used for the spin-α orbitals and the spin-β orbitals.

SetOccEnergy esetocc All occupied orbitals that have to be used will change their orbital energy to esetocc. In practice only useful if one has selected one occupied orbital energy, and one want to change this to another value. Default: the orbital energies of the occupied orbitals that are used are not changed.

SetLargeEnergy epsbig The orbital energies of the uninteresting occupied orbitals are changed to -epsbig Hartree, and the orbital energies of the uninteresting virtual orbitals are changed to epsbig Hartree (Default: epsbig = 1d6 Hartree).

UseScaledZORA Use everywhere the scaled ZORA orbital energies instead of the ZORA orbital energies in the TDDFT equations. This can improve deep core excitation energies. Only valid if ZORA is used. Default: use the unscaled ZORA orbital energies.

Excitations as orbital energy differences

Instead of the relative expensive TDDFT calculation of excitation energies, sometimes just calculating Kohn-Sham orbital energy differences may already be useful. The subkeyword SINGLEORBTRANS in the block key EXCITATIONS, will calculate excitation energies as Kohn-Sham orbital energy differences. The subkeyword SINGLEORB-TRANS in the block key EXCITATIONS replaces the obsolete key KSSPECTRUM. For a given excitation from an occupied orbital to a virtual orbital the oscillator strength is calculated from the the dipole transition moment between this occupied orbital and this virtual orbital. This method is best suited if LDA or a GGA is used in the SCF.

Especially useful for core excitation energy calculations. If SINGLEORBTRANS is used, it is possible to use fractional occupation numbers in the SCF, like is used in the DFT transition state (DFT-TS) scheme, see, for example, Ref. [*359* (page 928)]. Note: for fractional occupation numbers, typically an orbital is treated in the excitation calculation as if it is fully occupied if the occupation number is 1.5 or more, and it is treated as if it is fully unoccupied if the occupation number is 0.5 or less. In the transition state procedure for core excitations half an electron is moved from an initial core orbital to final virtual orbital and the SCF KS eigenvalue difference is taken as excitation energy. This is the original Slater formulation and is justified doing a Taylor expansion of total energy with respect to occupation numbers, it can be shown that errors arise from third order terms which are small. This, however, is not very practical: to converge SCF by putting half an electron in virtual orbitals is very hard, usually it works only for the lowest virtuals. In order to avoid this problem one neglects the half an electron in virtual, this scheme (only half an electron removed from occupied orbital) is called Transition Potential (TP) and is widely employed in XAS from K-edges.

```
EXCITATIONS
  SingleOrbTrans
END
```

SingleOrbTrans keyword to use only orbital energy differences

The value for the subkeyword KFWRITE in the EXCITATIONS block key is set by default to 0 in case of SINGLE-ORBTRANS. If kfwrite is 0 then do not write contributions, transition densities, and restart vectors to TAPE21, since this can lead to a huge TAPE21, especially if many excitations are calculated.

Quadrupole intensities in X-ray spectroscopy

For electronic excitations in the ultraviolet and visible range of the electromagnetic spectrum, the intensities are usually calculated within the dipole approximation, which assumes that the oscillating electric field is constant over the length scale of the transition. For the short wavelengths used in hard X-ray spectroscopy, the dipole approximation may not be adequate. In particular, for metal K-edge X-ray absorption spectroscopy (XAS), it becomes necessary to include higher-order contributions. An origin-independent calculation of quadrupole intensities in XAS was implemented in ADF by Bernadotte et al., see Ref.[*385* (page 929)]. These quadrupole intensities may also be important for the calculation of X-ray emission spectroscopy, see *section on XES* (page 170). Can not be used in combination with spin-orbit coupling.

Usage

```
EXCITATION
   ...
   XAS
   {ALLXASMOMENTS}
   {ALLXASQUADRUPOLE}
END
```

XAS Use XAS within the EXCITATION block to initiate the calculation of the higher oder multipole moment integrals and the calculation of the quadrupole oscillator strengths. This will only print the total oscillator strength and the excitation energy.

ALLXASMOMENTS Use ALLXASMOMENTS within the EXCITATION block in combination with the XAS keyword. This will print out all the individual transition moments used within the calculation of the total oscillator strength.

ALLXASQUADRUPOLE Use ALLXASQUADRUPOLE within the EXCITATION block in combination with the XAS keyword. This will print out the individual oscillator strength components to the total oscillator strength.

The *MODIFYEXCITATION* (page 168) or *SELECTEXCITATION* (page 167) keyword could be used to select a core orbital.

6.3.5 XES: X-ray emission spectroscopy

For the calculation of XES (X-ray emission spectroscopy) spectra the approach of Ref.[*386* (page 929)] is implemented in the ADF program by Atkins et. al., see Ref.[*387* (page 929)]. This is a frozen orbital, one-electron ΔDFT approach which uses orbital energy differences between occupied orbitals to model the X-ray emission energies. Even though it is the simplest possible approximation for the calculation of XES spectra, it has been shown to work well for V2C-XES (valence-to-core X-ray emission spectroscopy) spectra of transition metal complexes.

The XES keyword initiates the calculation of X-ray emission energies to a core orbital. In addition to dipole oscillator strengths this keyword also triggers the calculation of the higher order moment integrals and the calculation of the quadrupole oscillator strengths. By default it calculates the emission to the first orbital in the first symmetry, often the deepest core orbital, and only prints the total oscillator strength and the excitation energy. Can not be used in combination with spin-orbit coupling.

```
XES
   {COREHOLE irrep number}
   {ALLXESMOMENTS}
   {ALLXESQUADRUPOLE}
END
```

COREHOLE irrep number This allows the selection of the acceptor orbital for the calculation of the emission oscillator strengths. For example 'COREHOLE A1 2' calculates oscillator strengths to the orbital 2 in irrep A1.

ALLXESMOMENTS Use ALLXESMOMENTS within the XES block. This will print out all the individual transition moments used within the calculation of the total oscillator strength.

ALLXESQUADRUPOLE Use ALLXESQUADRUPOLE within the XES block. This will print out the individual oscillator strength components to the total oscillator strength.

6.3.6 Excitation energies and Spin-Orbit coupling

Spin-orbit coupling can be included in the TDDFT calculation of excitation energies for closed-shell molecules. Two methods can be used in ADF. The first one includes spin-orbit coupling as a perturbation to a scalar relativistic calculation of excitation energies, in which spin-orbit coupling matrix elements (SOCMEs) between excited states are

calculated. The second one includes spin-orbit coupling self-consistently in the ground state calculation. If spin-orbit coupling is large, the second one is more accurate, but is also more time-consuming.

The results of these spin-orbit coupled TDDFT calculations include the calculation of the zero field splitting (ZFS) of triplet excited states and the calculation of radiative rate constants, which could be used to calculate radiative phosphorescence lifetimes.

Perturbative inclusion of spin-orbit coupling

```
SOPERT
  {NCALC ncalc}
  {ESHIFT eshift}
End
RELATIVISTIC SCALAR ZORA
EXCITATIONS
END
```

The perturbative method, which is described in Ref.[*280* (page 923)], is an approximate time-dependent density-functional theory (TDDFT) formalism to deal with the influence of spin-orbit coupling effect on the excitation energies for closed-shell systems. In this formalism scalar relativistic TDDFT calculations are first performed to determine the lowest single-group excited states and the spin-orbit coupling operator is applied to these single-group excited states to obtain the excitation energies with spin-orbit coupling effects included. The computational effort of the present method is much smaller than that of the two-component TDDFT formalism. The compositions of the double-group excited states in terms of single-group singlet and triplet excited states are obtained automatically from the calculations. In Ref.[*280* (page 923)] it was shown that the calculated excitation energies based on the present formalism affords reasonable excitation energies for transitions not involving 5p and 6p orbitals. For transitions involving 5p orbitals, one can still obtain acceptable results for excitations with a small truncation error, while the formalism will fail for transitions involving 6p orbitals, especially 6p1/2 spinors.

Although this method is not completely correctly implemented for (meta-)hybrids or Hartree-Fock, it still gives reasonable excitation energies, and can thus be useful also in that case. Note that SYMMETRY C(2H) is not implemented for spin-orbit coupled excitations, use SYMMETRY C(S), C(I) or NOSYM, instead.

NCALC=ncalc Number of spin-orbit coupled excitation energies to be calculated. Default (and maximum) value: 4 times the number of scalar relativistic singlet-singlet excitations.

ESHIFT=eshift The actually calculated eigenvalues are calculated up to the maximum singlet-singlet or singlet-triplet scalar relativistic excitation energy plus eshift (in Hartree). Default value: 0.2 Hartree.

Some extra information about the spin-obit matrix is written to the output if one includes:

```
SOPERT
   {NCALC ncalc}
   {ESHIFT eshift}
End
PRINT SOMATRIX
```

If one includes PRINT SOMATRIX the spin-orbit matrix (in Hartree) on basis of singlet and triplet excited states will be printed. Note that a triplet has three sublevels. The numbering of the basis of singlets and triplets is given above the spin-orbit matrix. The spin-orbit matrix has a real and imaginary part. On the diagonal the singlet or triplet energies is added. This spin-orbit matrix is the one that is diagonalized to get the spin-orbit coupled excitation energies. For example, for the effect of spin-orbit coupling between T1 and S1 one finds 3 complex values, between the 3 sublevels of T1 and the one level of S1. For a magnitude one could take the square root of the (real part squared + imaginary part squared).

Starting from ADF2018 these spin-orbit coupling matrix elements (SOCMEs) are printed in an easier format in the ouput. The SOCME that is printed is calculated as a root mean square: square root of (the sum of squares of spin-orbit

coupling matrix elements of all sublevels of the uncoupled states).

```
GSCORR
```

If one includes GSCORR the singlet ground state is included, which means that spin-orbit coupling can also have some effect on energy of the ground state. The spin-orbit matrix in this case is on basis of the ground state and the singlet and triplet excited states.

Self-consistent spin-orbit coupling

```
RELATIVISTIC SPINORBIT ZORA
EXCITATIONS
   {ALSORESTRICTED}
END
```

Starting from the ADF2006.01 version in ADF the relativistic TDDFT formalism, including spin-orbit coupling, is implemented for closed-shell molecules with full use of double-group symmetry [*182* (page 917)]. This relativistic time-dependent density-functional theory (TDDFT) is based on the two-component zeroth-order regular approximation (ZORA) and a noncollinear exchange-correlation (XC) functional. This two-component TDDFT formalism has the correct non-relativistic limit and affords the correct threefold degeneracy of triplet excitations.

In case of a calculation including spin-orbit coupling one can not separate the singlet-singlet and singlet-triplet excitations. By default the spin-polarized excitation energies are calculated (the noncollinear scheme is used for the spin-dependent exchange-correlation kernel). The subkeys ALSORESTRICTED can be used to include also excitation energies in which a spin-restricted exchange-correlation kernel is used. One should in fact only use the results of the spin-polarized calculation, which is based on the noncollinear exchange-correlation (XC) functional. For the same reason, the ALLOWED subkey should not be used if spin-orbit coupling is included. Note that SYMMETRY C(2H) is not implemented for spin-orbit coupled excitations, use SYMMETRY C(S), C(I) or NOSYM, instead.

To perform a spin-orbit coupled TDDFT calculation one just needs to do a spin-orbit coupled SCF calculation and use the EXCITATION keyword. The molecule needs to be closed shell, and should be calculated spin-restricted. Thus do not use the UNRESTRICTED, COLLINEAR, or NONCOLLINEAR keyword. See, however, also next section.

The contribution to the double group excited states in terms of singlet and triplet single group excited states can be estimated through the inner product of the transition density matrix obtained from two-component and scalar relativistic TDDFT calculations to better understand the double group excited states [*183* (page 917)]. In order to get this analysis one needs to perform a scalar relativistic TDDFT calculation of excitation energies on the closed shell molecule first, and use the resulting TAPE21 as a fragment in the spin-orbit coupled TDDFT calculation of excitation energies, including the keyword STCONTRIB (Singlet and Triplet CONTRIButions):

```
STCONTRIB
```

This STCONTRIB analysis is not performed for (meta-)hybrids, unless one uses the Tamm-Dancoff approximation (TDA) approximation, but then it may also fail. If one wants a similar analysis for (meta-)hybrids one may consider to the perturbative inclusion of spin-orbit coupling in the calculation of excitation energies.

Note that if hybrids are used, the dependency key is automatically set, and this may effectively reduce the number of excitations, which may give problems in the STCONTRIB analysis. A workaround for these problems is to first calculate the scalar relativistic fragment without the EXCITATIONS keyword. Use the TAPE21 of this calculation as fragment in a scalar relativistic calculation with the EXCITATIONS keyword. Use the TAPE21 of the second calculation as fragment in the spin-orbit coupled calculation, including the STCONTRIB keyword.

Highly approximate spin-orbit coupled excitation energies open shell molecule

Excitation energies can be obtained for open-shell systems in a spin-unrestricted TDDFT calculation including spin-orbit coupling. This approximate method uses a single determinant for the open shell ground state. The Tamm-Dancoff approximation (TDA) is needed and symmetry NOSYM should be used. Best is to use the noncollinear approximation. For analysis it is advised to calculate the molecule also with the scalar relativistic spin-restricted method and use it as fragment in the spin-orbit coupled calculation. This will make it easier to identify the excitations.

```
Relativistic spinorbit ZORA
Unrestricted
NonCollinear
Symmetry Nosym
TDA
Excitations
End
```

Note that this approximate method for open shell molecules is not able to show the subtle effects of spin-orbit coupling. Some of the reasons are the approximate nature of the XC functionals for open shell molecules, the single determinant that is used for the open shell ground state, and that only single excitations are included in the excitation. If one does not include spin-orbit coupling the spin-unrestricted TDDFT approach introduces spin-contamination such that the result does not represent transitions between pure spin states. Inclusion of spin-orbit coupling will not simplify this. However, if spin-orbit coupling is large, then this method may help to identify excitations.

Note that the approximations made in this approximate method are much worse than for spin-orbit coupled TDDFT for closed shell systems. In that case one can get a reasonable description of the subtle effects of spin-orbit coupling, for example, for the zero-field splitting of a triplet excited state.

6.3.7 CV(n)-DFT: Constricted Variational DFT

In the constricted nth order variational density functional method (CV(n)-DFT) [*393* (page 930), *394,* (page 930) *413* (page 931)] the occupied excited state orbitals are allowed to relax in response to the change of both the Coulomb and exchange-correlation potential in going from the ground state to the excited state. This theory is not time-dependent nor is it based on response theory. It is instead variational in nature and has been termed constricted variational DFT or CV(n)-DFT.

Due to bugs in older versions it is important to use ADF2016.105 or later. In ADF2017 the relaxation density is a bit differently calculated than in ADF2016, which will slightly modify the results compared to ADF2016. In ADF2017 singlet-triplet excitations are added [*479* (page 935)] There have been different working equations implemented in different modified versions of ADF. In ADF2016 and ADF2017 all CV-DFT excitation energies are calculated consistently, this is why some energies may differ from previously published values.

The CVNDFT block key regulates the execution of the CV(n)-DFT code, which calculates the singlet or triplet electronic excitations for the closed shell molecules. Note that one has to choose either singlet-singlet excitations (subkey ONLYSING in EXCITATIONS) or singlet-triplet excitations (subkey ONLYTRIP in EXCITATIONS), in case of CV(n)-DFT, one can not calculate them both in one run. The parameter n in (n) describes the order of this theory. There are two limiting cases implemented in the CV(n)-DFT code: n=2 and n= ∞. Since CV(n)-DFT coincides in the second order with TDDFT, CV(n)-DFT code is designed as an extension of the EXCITATIONS module. Therefore, the EXCITATIONS block has to be present in the input file together with CVNDFT. CVNDFT takes all the information about the number of excitations and their symmetry from the EXCITATIONS block. Moreover, the TDDFT transition density vectors are used as an initial guess for the CV(∞)-DFT calculations.

There are a few approximations within the CV(∞)-DFT approach, therefore there are a few corresponding sub-blocks in the CVNDFT block. In the simplest case the TDDFT transition density U-vector is substituted into the infinite order CV(∞)-DFT excitation energy. This approximation corresponds to the CV_DFT sub-block, so the input fragment would look like this:

```
CVNDFT
  CV_DFT &
  SUBEND
END
EXCITATIONS
  ONLYSING|ONLYTRIP
  ...
END
```

In general, the U-vector has to be optimized with respect to the infinite order CV(∞)-DFT excitation energy. It is accomplished iteratively in the SCF-CV(∞)-DFT method [*412* (page 931)], which is invoked when the SCF_CV_DFT sub-block is present. Therefore, there are input parameters that control the SCF procedure, i.e. the total number of iterations and tolerance. The corresponding input fragment would look like this:

```
CVNDFT
  ITERATION iter
  TOLERANCE tol
  SCF_CV_DFT &
    DAMPING mix
  SUBEND
END
EXCITATIONS
  ONLYSING|ONLYTRIP
  ...
END
```

iter iter is the maximum number of iterations. Default 50.

tol tol is the convergence criterion, i.e. the SCF-CV(∞)-DFT procedure stops when the given accuracy is achieved. Default 1e-4.

mix mix is the relative weight of the new U-vector that is added to the one from the previous iteration (default value is mix=0.2).

Another feature of CV(∞)-DFT is a possibility to optimize (or relax) the molecular orbitals for the particular excitation. This is accomplished in the R-CV(∞)-DFT method, see for an aplication Ref.[*411* (page 931)]. In the last case, the transition U-vector is frozen, while the orbitals are relaxed. For example, the input fragment which employs the U-vector from TDDFT would look like this:

```
CVNDFT
  ITERATION iter
  TOLERANCE tol
  R_CV_DFT &
    RELAXALPHA start_a_iter
    RELAXBETA  start_b_iter
    DAMPORBRELAX mix_relax
  SUBEND
END
EXCITATIONS
  ONLYSING|ONLYTRIP
  ...
END
```

start_a_iter start_a_iter is the SCF cycle number at which the relaxation of α orbitals starts. Default value 1.

start_b_iter start_b_iter is the SCF cycle number at which the relaxation of β orbitals starts. Default value 1.

Since there is no optimization of the U-vector in R-CV(∞)-DFT, the relaxation of orbitals (either α or β or both) should be turned on at the very first iteration.

mix_relax The mix_relax parameter defines the relative weight of the new relaxation vector that is added to the one from the previous iteration (the default value is mix_relax=0.2).

Note that all tree methods can be used in the input file. The corresponding input fragment would look like this:

```
CVNDFT
  ITERATION iter
  TOLERANCE tol
  CV_DFT &
  SUBEND
  SCF_CV_DFT &
    DAMPING mix
  SUBEND
  R_CV_DFT &
    RELAXALPHA start_a_iter
    RELAXBETA  start_b_iter
    DAMPORBRELAX mix_relax
  SUBEND
END
EXCITATIONS
  ONLYSING|ONLYTRIP
  ...
END
```

In this case the U-vector will be optimized with respect to the infinite order CV(∞)-DFT excitation energy by the SCF-CV(∞)-DFT code and then supplied to the R-CV(∞)-DFT code.

CVNDFT prints in the output the excitation energies as well as the maximum value of the lambda parameter λ. This number corresponds to the largest singular value of the singular value decomposition of the U matrix/vector (i.e. the U-vector can be considered as a matrix with *nvir* rows and *nocc* columns, where *nvir* is the number of virtual orbitals and *nocc* is the number of occupied orbitals). If this number is close to $\pi/2$ (i.e. $\lambda_{max} \approx 1.57$), then the corresponding electronic excitation can be approximated by a single natural transition orbital (NTO) to another single NTO transition. This type of transitions are obtained in Δ SCF. Therefore, this information can be used, for example, to compare the CV(∞)-DFT results to the Δ SCF excitation energies.

6.3.8 TD-DFT+TB

The basis idea of TD-DFT+TB [*424* (page 932)] is to use the molecular orbitals from a DFT ground state calculation as input to an excited state calculation with TD-DFTB coupling matrices. TB means tight binding. If many excitations are required, this method will speed up the calculation drastically in comparison to the standard time needed for TDDFT calculations of excitation energies. This method is best suited if a (meta-)GGA or LDA is used in the SCF. This method can not be used in excited state geometry optimizations.

```
SYMMETRY NOSYM
EXCITATIONS
 TDDFTB
END
```

One can use this, for example, in combination with the *MODIFYEXCITATION key* (page 167) to use only single-orbital transitions that have a minimal oscillator strength:

```
SYMMETRY NOSYM
EXCITATIONS
 TDDFTB
 ONLYSING
END
MODIFYEXCITATION
```

```
   OscStrength 0.001
End
```

In TD-DFT+TB reducing the space of single-orbital transitions with the key MODIFYEXCITATION will reduce the memory needed in ADF. Starting from ADF2016.102 for TD-DFT+TB only the lowest (in energy) 100000 single orbital transitions are included by default. One can override this by using a different value for the subkey UseOccVirt-Numbers of the key MODIFYEXCITATION.

```
MODIFYEXCITATION
 UseOccVirtNumbers 1 100000
END
```

The value for the subkeyword KFWRITE in the EXCITATIONS block key is set by default to 0 in case of TD-DFT+TB. If kfwrite is 0 then do not write contributions, transition densities, and restart vectors to TAPE21, since this can lead to a huge TAPE21, especially if many excitations are calculated.

6.3.9 sTDA, sTDDFT

The simplified Tamm-Dancoff approach (sTDA) [*389* (page 930)] and simplified time-dependent DFT approach (sTDDFT) [*390* (page 930)] by Grimme et al. are implemented in ADF. In these approaches the time-dependent DFT equations are simplified by the evaluation of the two-electron integrals as short-range damped Coulomb interactions between (transition) charge density monopoles and a truncation of the single excitation expansion space. These methods are best suited if a (meta-)hybrid or a range-separated-hybrid is used in the SCF. These methods will speed up the calculation drastically in comparison to the standard time needed for TDA or TDDFT calculations of excitation energies for hybrids. Theses methods can not be used in excited state geometry optimizations.

An example application of sTDDFT is given in the TDDFT advanced tutorial

For sTDA use:

```
SYMMETRY NOSYM
EXCITATIONS
 sTDA
END
```

For sTDDFT use:

```
SYMMETRY NOSYM
EXCITATIONS
 sTDDFT
END
```

For global hybrids ADF will use the parameters α and β that depend on the amount of exact exchange a_x as:

$$\alpha = \alpha_1 + \alpha_2 a_x$$
$$\beta = \beta_1 + \beta_2 a_x$$

where $\alpha_1 = 1.42$, $\alpha_2 = 0.48$, $\beta_1 = 0.2$, $\beta_2 = 1.83$ are used in ADF. These values are fitted parameters by Grimme [*389* (page 930)]. For range-separated hybrids one should set the parameters α, β and a_x in the input of ADF with the keyword MODIFYEXCITATION. See Ref. [*391* (page 930)] for parameters that are used by Risthaus et al. for a few range-separated functionals. However the α and β parameters are mixed up in Ref. [*391* (page 930)]. Thus use:

Functional	a_x	α	β
CAM-B3LYP	0.38	0.90	1.86
WB97	0.61	4.41	8.00
WB97X	0.56	4.58	8.00

For example, for CAM-B3LYP use

```
MODIFYEXCITATION
  GrimmeAlpha 0.9
  GrimmeBeta 1.86
  GrimmeAex 0.38
END
```

If one uses the keyword MODIFYEXCITATION one may also set more parameters that are used in the sTDA and sTDDFT approach.

```
MODIFYEXCITATION
  GrimmeAlpha Alpha
  GrimmeBeta Beta
  GrimmeAex Aex
  GrimmeDEmax DEmax
  GrimmeTPmin TPmin
  GrimmePertC|NoGrimmePertC
END
```

GrimmeAlpha Alpha To set the α parameters, should be used icw GrimmeBeta and GrimmeAex.

GrimmeBeta Beta To set the β parameters, should be used icw GrimmeAlpha and GrimmeAex.

GrimmeAex Aex To set a_x that is used in the sTDA or sTDDFT part of the calculation, should be used icw GrimmeAlpha and GrimmeBeta.

GrimmeDEmax DEmax Single orbital transitions that have an orbital energy difference less than DEmax (in Hartree) are included. Default value DEmax = 0.4 Hartree. See also the meaning of E_{max} in Grimme [*389* (page 930)].

GrimmeTPmin TPmin Single orbital transitions that would have a cumulative perturbative energy contribution larger than TPmin is included. Default value TPmin = 10^{-4}. See also the meaning of t_p in Grimme [*389* (page 930)].

GrimmePertC|NoGrimmePertC In case of GrimmePertC, which is the default, if the cumulative perturbative energy contribution for a single orbital transition is smaller TPmin, this contribution is used to modify the diagonal value of a matrix that is used in the sTDA or the sTDDFT method. If NoGrimmePertC is included such contributions are neglected. See Grimme [*389* (page 930)] for more details on the truncation of the single excitation space.

Starting from ADF2016.102 for sTDA or sTDDFT only the lowest (in energy) 10000 single orbital transitions are included by default. One can override this by using a different value for the subkey UseOccVirtNumbers of the key MODIFYEXCITATION.

```
MODIFYEXCITATION
 UseOccVirtNumbers 1 10000
END
```

The value for the subkeyword KFWRITE in the EXCITATIONS block key is set by default to 0 in case of sTDA or sTDDFT. If kfwrite is 0 then do not write contributions, transition densities, and restart vectors to TAPE21, since this can lead to a huge TAPE21, especially if many excitations are calculated.

6.3.10 CD spectra

Circular dichroism (CD) is the differential absorption of left- and right-handed circularly polarized light. Starting from ADF2010 Hartree-Fock and hybrids can also be used to calculate CD spectra.

```
EXCITATIONS
 CDSPECTRUM
 ANALYTICAL
 VELOCITY
End
```

CDSPECTRUM If the subkey *CDSPECTRUM* is included in the key EXCITATIONS the rotatory strengths for the calculated excitations are calculated, in order to simulate Circular Dichroism (CD) spectra [*80* (page 911), *81* (page 911)]. Interesting for chiral molecules. This subkey should not be used in case of spin-orbit coupling. For accuracy reasons you should also use the subkey *ANALYTICAL* in the block key EXCITATIONS, otherwise the results may be nonsense.

ANALYTICAL If the subkey *ANALYTICAL* is included the required integrals for the CD spectrum are calculated analytically, instead of numerically. Only used in case of CD spectrum.

Velocity If the subkey *VELOCITY* is included ADF calculates the dipole-velocity representation of the oscillator strength. If applicable (use of subkey *CDSPECTRUM*) the dipole-velocity representation of the rotatory strength is calculated. Default the dipole-length representation of the oscillator strength and rotatory strength is calculated.

6.3.11 MCD

MCD or magnetic circular dichroism is the differential absorption of left and right circularly polarized light in the presence of a magnetic field. MCD intensity is usually described in terms of different contributions called A, B and C terms, see Refs. [*273* (page 923), *274* (page 923)]. A further parameter D is often discussed in MCD studies. D is proportional to the intensity of an absorption band and is closely related to the oscillator strength. A and B terms for closed and open-shell molecules and C terms of open-shell molecules induced by spin-orbit coupling can be calculated. Starting from ADF2010 C terms related to spatially degenerate states, i.e. breaking of degeneracies can be calculated. The A term is only calculated in case the molecule has symmetry and then only for excited states that belong to multi-dimensional irreps.

For MCD calculations for molecules that have C(2) or D(2) symmetry use SYMMETRY NOSYM. **Input options**

```
EXCITATIONS
   MCD {options}
   ONLYSINGLET
   {SELECT transition number}
   {DTENSOR {Dxx Dxy Dyy Dxz Dyz|D E/D}}
End

ALLPOINTS
{RELATIVISTIC ZORA}
{SOMCD}
{ZFS}
```

MCD If the subkey MCD is included in the key EXCITATIONS the MCD parameters of some or all of the excitations considered in the TDDFT procedure are calculated [*275-278* (page 923)]. This subkey should not be used with spin-orbit coupling (but, see below). Several other keywords could be important.

ALLPOINTS: required for an MCD calculation.

ONLYSINGLET: this keyword should be used in combination with a MCD calculation.

RELATIVISTIC ZORA: required for a calculation of temperature-dependent C terms. In this case the keyword SOMCD must also be added as a key by itself, and the calculation must be unrestricted. If only A and B terms are calculated then ZORA is not needed but can be included if desired.

ZFS: If the ZFS keyword and MCD with SOMCD are also included then the influence of the calculated zero-field splitting (ZFS) on the temperature-dependent MCD is evaluated. The MCD in the presence of ZFS is described as anisotropic in the output because the Zeeman splitting becomes orientation dependent in the presence of ZFS.

In ADF2010 the temperature-dependent MCD due to the breaking of degeneracies of excited states by spin-orbit coupling can be calculated. Although all temperature-dependent MCD is typically called "C terms", the parameters associated with the MCD are labeled "CE" to distinguish them from the MCD due to mixing between states caused by spin-orbit coupling that is labeled "C". The CE terms have a derivative shape like A terms. They have the same temperature-dependence as normal C terms. If they are present, CE terms are calculated automatically along with C terms if the keyword SOMCD is included in the input.

MCD {options} Options include NMCDTERM, NMIX, DCUTOFF, MCDOUT, CGOUT, NANAL, NANAL2, FULLOMEGA, NOAB, NODIRECT, NOCG, CONVCG, ITERCG, ITER2CG, BMIN, BMAX, TMIN, TMAX and NTEMP.

NMCDTERM=nmcdterm Number of excitations for which MCD parameters are to be calculated. The nmcdterm lowest energy excitations are treated. The default is the number of transitions considered in the TDDFT calculation.

NMIX=nmix Number of transitions allowed to mix in a SOS calculation. Default is the number of transitions considered in the TDDFT calculation.

DCUTOFF=dcutoff MCD parameters will only be calculated for transitions with sufficient intensity. Each Cartesian component of each transition is considered separately. If the dipole strength D of that component is below dcutoff then the MCD is not calculated. The default is 1.0e-6.

MCDOUT=mcdout Number that determines the amount of output to be printed about the MCD calculation. Higher means more output. Possible values are 0, (orientationally averaged and Cartesian components of MCD parameters only) 1 (as for 0 but with the addition of a short analysis) or 2 (as for 1 but with the addition of a lengthy analysis). Theoretical analyses of MCD parameters are presented in several places including Refs. [*273-274* (page 923), *276-278* (page 923)]. The default for MCDOUT is 0.

CGOUT=cgout The perturbed transition densities used to evaluate the B and C term parameters can be obtained through an iterative conjugate-gradient procedure. Convergence information of the conjugate-gradient algorithm is printed every cgout iterations. Default is 10.

NANAL=nanal, NANAL2=nanal2 If MCDOUT is set to 2, a detailed analysis of the B and/or C term parameters in terms of which states mix and how much MCD each mixing causes, is presented. The parameters NANAL and NANAL2 determine how many contributions are included in the analyses. Defaults are 10 for NANAL and 5 for NANAL2.

FULLOMEGA A standard TDDFT calculation involves the solution of an eigenvalue equation to obtain the excitation energies and transition densities of interest. ADF can solve this eigenvalue equation two ways: through diagonalization of the full Omega matrix or through the Davidson procedure where Omega is never explicitly constructed. Construction of the complete Omega matrix is generally only feasible for smaller problems. The matrix Omega appears again in the equations solved to obtain MCD. Here again Omega can be built or only the products of Omega with a vector can be used as is the case in the Davidson procedure. The default is to not construct Omega. If the keyword FULLOMEGA is included then Omega is constructed. Note that the choice of FULLOMEGA is completely independent of whether EXACT or DAVIDSON is chosen in the earlier TDDFT calculation.

NOAB If this keyword is included then A and B terms are not calculated. NOAB only makes sense if SOMCD is included in the input otherwise no MCD will be calculated at all.

NODIRECT The perturbed transition density needed to evaluate B and C term parameters is obtained through the solution of a large system of equations. This system of equations is solved in two ways: through a sum-over-states (SOS) type approach where the solution is expanded in a known set of transition densities or through the direct solution of the system of equations by the conjugate gradient procedure. The SOS

method is much faster but also less accurate, particularly for larger systems. By default MCD parameters are evaluated through both approaches. If the NODIRECT keyword is included then only the SOS calculation is performed.

NOCG The conjugate gradient procedure is first used in combination with a preconditioner that generally speeds up convergence significantly. If no solution is found in a reasonable number of iterations then the procedure is restarted without the preconditioner. If the NOCG keyword in included then the preconditioner is never used.

CONVCG Convergence criterion for the CG iterative methods. The default value of 0.01 is probably good enough for most applications. This choice seems to produce B and C terms that are converged to 3 significant figures. Except for small systems, it is not recommended that CONVCG be set to a much smaller number as this will probably cause a large number of convergence failures.

ITERCG=itercg Number of iterations before failure in the first (preconditioned) CG solver. This solver either succeeds quickly or not at all so the default value is 30.

ITER2CG=iterc2g Number of iterations before failure in the B of C term parameter calculation of the unconditioned CG solver. This solver is often slow so the default value is 200.

BMIN=bmin, BMAX=bmax, NBFIELD=nbfield, TMIN=tmin, TMAX=tmax, NTEMP=ntemp
Temperature dependent MCD intensity often varies nonlinearly with T and B when T is small and/or B is large. It may therefore be of interest to evaluate the MCD intensity over a range of temperatures and/or magnetic fields. This can be achieved through the use of the BMIN, BMAX, NBFIELD, TMIN, TMAX and NTEMP keywords. The MIN and MAX keywords give the maximum values of B or T. NBFIELD and NTEMp indicate how many values are to be considered. Note that magnetic fields are assumed to be given in Tesla and temperatures in Kelvin. For example, BMIN=1, BMAX=5, NBFIELD=5 means that fields of 1,2,3,4 and 5 T will be considered. Defaults are BMIN=BMAX=1, TMIN=TMAX=5 and NBFIELD=NTEMP=1.

SELECT nselect1 nselect2 nselect3... Rather than selecting the first nmcdterm transitions for consideration individual transitions can be selected through the SELECT keyword. The transitions of interest are listed after the SELECT keyword. Note that the numbering follows that given in the summary table at the end of the TDDFT calculation. To consider a degenerate transition only the first component need be included. Note that it makes no sense to use both the SELECT and NMCDTERM keywords together.

DTENSOR Dxx Dxy Dyy Dxz Dyz DTENSOR D E/D As noted earlier, if the ZFS keyword is included with MCD and SOMCD then the influence of zero-field splitting on temperature-dependent MCD will be evaluated. As an alternative to the ZFS keyword the D-tensor parameters can be entered directly through the DTENSOR keyword in the EXCITATIONS block. Two input formats are possible. Five real numbers Dxx Dxy Dyy Dxz Dyz can be entered. These five numbers are sufficient to define the traceless tensor D. Alternatively, the two parameters D and E/D can be entered. In this case the coordinate system chosen to define the molecular geometry must be the the principle axis system of the D-tensor. D, Dxx, Dxy, Dyy, Dxz and Dyz should be given in wavenumbers (cm-1).

Notes

If an MCD calculation is run, the transition densities obtained in the TDDFT calculation are saved to TAPE21. For large molecules this can result in a very large TAPE21 file.

An MCD calculation relies on the excitation energies and, in particular, the transition densities that result from the preceding TDDFT calculation. If the results of the TDDFT calculation are poor then it is likely that the results of the MCD calculation will be poor. It therefore should be kept in mind that most TDDFT calculations will make use of the Davidson method for finding the eigenvalues and eigenvectors of the TDDFT equation. The Davidson approach involves some approximations that can lead to some variation in results with the applied parameters. The most important example of this is the fact that the results vary depending on how many eigenvalue/eigenvector pairs are calculated, ie how many transitions are selected through the LOWEST keyword. The variation is small for the eigenvalues (excitation energies) but can be significant for the eigenvectors (transition densities). A variation in the transition densities

leads to variation in the transition dipoles which can significantly impact calculated MCD parameters. The moral of this story is that when calculating MCD parameters it is best to choose one value of LOWEST and stick with it.

The most time-consuming part of an MCD calculation is the solution of the system of equations through the conjugate-gradient solver. The solver can fail so be aware of warnings concerning convergence in the output. A few hints to improve convergence are: a) choose a value of LOWEST that is at least double the number of transitions for which you desire MCD parameters. This helps to improve the SOS calculation which provides an initial guess for the conjugate gradient solver. The solver is sensitive to the initial guess so changing LOWEST by a small amount may help (or hinder) convergence significantly. Keep the previous note in mind when playing with LOWEST however. b) The preconditioned conjugate gradient solver is usually fast but does not converge monotonically to the correct answer. The unpreconditioned solver is much slower but tends to converge monotonically. If the preconditioned solver fails but leaves a fairly well converged result for the unpreconditioned solver the latter usually converges quickly. If the preconditioned solver does not leave a fairly well converged result it may be worth changing the number of iterations it uses since a few iterations earlier or later may provide a much better converged answer. c) The SELECT keyword can be used to work on the remaining transitions for which converged results have not been obtained.

All MCD parameters are presented in au. To convert A and C terms to the alternative unit D^2 (Debye squared) the value in au should be multiplied by 6.46044. To convert the B term to the alternative unit of D^2 /cm-1 the value in au should be multiplied by 2.94359e-05.

The A, B and C terms are defined through the equation suggested by Stephens (equation 1 in [*278* (page 923)] and also see [*273-274* (page 923), *107* (page 913)]). This equation assumes that MCD intensity varies linearly with applied magnetic field and that the temperature-dependent component varies linearly with temperature as 1/T. For the most part, these assumptions are reasonable. An exception is that the temperature-dependent part varies from linearity when T is very small. To allow for this situation a temperature and magnetic field dependent multiplicative constant (chi(B,T)) is evaluated whenever temperature-dependent MCD parameters are considered. This constant includes all magnetic field and temperature dependence of the temperature-dependent MCD. Thus chi(B,T)*C can be used in place of B*C/kT in equation 1 of [*278* (page 923)] when MCD spectra are to be simulated. Note that, since the g-factor for all states is here approximated by 2.0, chi applies to all transitions.

6.3.12 Analysis

By default the ADF output will show the excitation energies, oscillator strengths from ground state to excited state, and transition dipole moments from ground state to excited state. Also an analysis in terms of single orbital transitions (occupied to virtual) is given, and their contributions to the transition dipole moment. For analysis reasons one might be interested in pure single orbital transition. Such information can be obtained in case one calculates excitation energies as Kohn-Sham orbital energy differences, see subkey *SINGLEORBTRANS* (page 169) of the key EXCITATIONS.

NTO: Natural Transition Orbitals

Natural Transition Orbitals [*396* (page 930)] come from a transformation of the transition density matrix. They are the closest you can get to visualizing an excitation as a one-electron excitation from 1 orbital to another. They give insight in the localization of excitations. Calculation of Natural Transition Orbitals (NTO's) will be performed if the subkey NTO is included in the key EXCITATIONS.

```
EXCITATIONS
  ...
  NTO
End
```

For (meta-)hybrids and range-separated functionals NTO's will be calculated only if one uses the Tamm-Dancoff approximation (TDA).

NTOs can be visualized in ADFspectra as shown in the GUI tutorial .

SFO analysis

In ADF2018 an SFO analysis of the excitation is implemented. This includes an analysis of the transition dipole moment. Especially useful in a fragment calculation, where one only has a few fragments. Not implemented for spin-orbit coupled excitations.

```
EXCITATIONS
  ...
  SFOANALYSIS {NMAXPRINTED}
End
```

SFOANALYSIS {NMAXPRINTED} If SFOANALYSIS is included a the SFO analysis of the excitation will be calculated. NMAXPRINTED is the maximum number of printed contributions. Default value for NMAX-PRINTED is 40.

The SFO - SFO dipole matrix elements can be printed in the output with:

```
PRINT DIPOLEMAT
```

Charge-transfer descriptors

Two methods for calculating charge-transfer descriptors are implemented (also working for spin-orbit coupled excitations).

The first method is the charge-transfer diagnostic overlap quantity LAMBDA, developed by Peach, Tozer, et al. [*493* (page 935)] and a hole-electron distance R_EH, see Guido, Adamo, et al. [*494* (page 936)].

The second method is the charge transfer descriptor CT, hole-electron distance R_EH, and some other descriptors, developed by Plasser, Lischka, et al. [*495-498* (page 936)]. For this method a fragment calculation is required. Using a similar method a charge transfer descriptor CT_AT was defined that uses an atomic distance criterion, which will also be calculated in case of atomic fragments.

```
EXCITATIONS
  ...
  DESCRIPTORS
  {Descriptors_CT_AT_Rab Rab}
END
```

DESCRIPTORS If DESCRIPTORS is included the charge transfer descriptors are calculated.

Descriptors_CT_AT_Rab Rab Rab is the atomic distance criterion used for the calculation of CT_AT (in Angstrom). Default value for Rab is 2 Angstrom.

6.4 Excited state (geometry) optimizations

Starting from ADF2010 it is possible to do excited state geometry optimizations, see Ref. [*350* (page 927)]. Note that not all aspects of such calculations have been tested thoroughly.

With the keyword EXCITEDGO the gradients of the TDDFT excitation energy can be calculated. Naturally, the EXCITATIONS block must also be included in the input. The excitation energy gradients will only be calculated if the ground state gradients are calculated. Thus, the GEOMETRY keyword is also required.

The gradients of the excitation energy are combined with the ground state gradients to give the gradients of the excited state. These gradients can be used in much the same way as ground state gradients are used. The type of calculation is chosen in the same way as for a ground state calculation. Possible run types are:

- Geometry optimization

- Frequency analysis with numerical second derivatives: (analytical second derivatives (ANALYTICALFREQ) are not possible).

- Linear transit

- Transition state search

- IRC calculations may be possible but this possibility has not been tested yet.

In general, an option that applies to a ground state geometry optimization will also apply to an excited state geometry optimization. For example, convergence criteria can be set and constraints can be used. These options are set through the GEOMETRY block as usual. A TDDFT geometry optimization will proceed in very much the same way as a ground state geometry optimization. The major difference will be that a TDDFT calculation will take place after the SCF and before the ground state gradients are evaluated. TDDFT gradients are calculated after the ground state gradients.

Gradients for closed-shell singlet-singlet, closed shell singlet-triplet, conventional open shell and spin-flip open-shell TDDFT calculations can be evaluated. The FORCEALDA option and TDA options should be used with spin-flip calculations.

Not all functionals can be used in combination with TDDFT gradients. The following should work:

LDA: VWN, XALPHA

GGA: Any allowed combination of the Perdew86, LYP and PBEc correlation functionals and the Becke88, revPBEx, RPBEx, PBEx and OPTx exchange functionals.

Hybrid: B1LYP, B3LYP, B3LYP*, BHANDHLYP, BHANDH, O3LYP, X3LYP, B1PW91, MPW1PW, PBE0, OPBE0.

Most LibXC GGA and hybrid functionals.

RS Hybrid icw XCFUN: CAMY-B3LYP and more (starting form ADF2018).

QM/MM TDDFT gradients can be calculated.

COSMO TDDFT gradients can be calculated (starting from ADF2018).

Scalar relativistic effects can be included with the ZORA or mass-velocity-Darwin Hamiltonians.

At this time, gradients involving frozen cores and spin-orbit TDDFT can not be calculated.

TDDFT gradients can take advantage of symmetry but if the point group of interest includes degenerate irreducible representations then all grid points are needed in integration (equivalent to the ALLPOINTS keyword). This situation is detected automatically. This use of the full grid may make it more efficient to use a point group with only one-dimensional irreducible representations where only the symmetry-unique slice is utilized.

Degenerate excitations can be optimized. However, since in reality such degeneracies will be split by a Jahn-Teller distortion it is recommended that the symmetry of the chosen point group be lowered so that the transition of interest is no longer labeled by a degenerate representation. A Jahn-Teller distortion will not occur when the degeneracy cannot be broken by nuclear motion, e.g. for a diatomic molecule.

The EXCITEDGO block key has the following form:

```
EXCITEDGO
 {STATE Irreplab nstate}
 {SINGLET/TRIPLET}
 {OUTPUT=n}
 {CPKS EPS=err PRECONITER=precon NOPRECONITER=noprecon ITEROUT=iter}
 {EIGENFOLLOW}
END
```

STATE Irreplab nstate Choose the excitation for which the gradient is to be evaluated.

Irreplab Irreplab is the label from the TDDFT calculation. NOTE: the TDDFT module uses a different notation for some representation names, for example, A' is used instead of AA.

nstate This value indicates that the nstate-th transition of symmetry Irreplab is to be evaluated. Default is the first fully symmetric transition.

Note that in a numerical FREQUENCIES calculation symmetry is turned off except to reduce the number of points calculated so irrespective of the specified point group Irreplab is A in this case. Care should be taken to ensure that nstate is correct in a frequencies calculation as this number can change when the point group is changed.

SINGLET/TRIPLET SINGLET: A singlet-singlet excitation is considered. The default. TRIPLET: A singlet-triplet excitation is considered.

OUTPUT=n The amount of output printed. A higher value requests more detailed output. Default: Output=0

CPKS EPS=err PRECONITER=precon NOPRECONITER=noprecon ITEROUT=iter Some control parameters for the CPKS(Z-vector) part of the TDDFT gradients calculation.

EPS=err err is a real number that gives the convergence requirement of the CPKS. Default is 0.0001

PRECONITER=precon precon is the maximum number of iterations allowed for the preconditioned solver. Default = 30.

NOPRECONITER=noprecon noprecon is the maximum number of iterations allowed for the unpreconditioned solver. Default=200.

ITEROUT=iter Details of the CPKS calculation are printed every iter iterations. Default is 5.

EIGENFOLLOW This key tries to follow the eigenvector in excited state geometry optimizations. In the initial implementation the target state of an excited state geometry optimization was indicated by a number and a symmetry, e.g. A2g 3 or the 3rd state of A2g symmetry. This approach becomes problematic when states cross and the state you are interested in become the 4th A2g state for example. An eigenvector-following option has been added that attempts to alleviate this problem. This option is off by default. If the subkeyword EIGENFOLLOW in is included, the state of interest in the first iteration is the same as before. In the second and subsequent iterations the state for which gradients are determined is decided on the basis of the overlap between the transition density of the transition from the previous iteration and the transition densities available in the current iteration. The same symmetry is maintained. Note that this method is not full proof. It assumes that the transition density changes only because of the contributions from the various occupied-virtual orbital pairs change but that the orbitals remain unchanged. This is not necessarily the case. Secondly, the sign of the transition density components is not taken into account.

At each iteration of a TDDFT-gradients calculation the excited state electric dipole moment is also calculated. If the Output parameter is 1 or greater then the excited state dipole moment will be printed out.

6.5 Vibrationally resolved electronic spectra

To calculate vibrational effects on the electronic excitations (Uv/vis, X-ray), one needs to do a frequency calculation both at the ground state as well as the excited state of interest. Next Franck-Condon factors need to be calculated for the transition between the two electronic states, which can be done with the FCF program, described below. These Franck-Condon factor can then be used to predict the relative intensities of absorption or emission lines in the electronic spectra. Note that the Herzberg-Teller effect is not taken into account.

6.5.1 FCF program: Franck-Condon Factors

fcf is an auxiliary program which can be used to calculate Franck-Condon factors from two vibrational mode calculations [*418* (page 931)].

fcf requires an ASCII input file where the user specifies the TAPE21 files from two *adf* vibrational mode calculations, carried out for two different electronic, spin or charge states of the same molecule. These calculations can be either numerical or analytical. The number of vibrational quanta that have to be taken into account for both states in the evaluation of the Franck-Condon factors have to be specified.

fcf produces a (binary) KF file TAPE61, which can be inspected using the KF utilities. Furthermore, *fcf* writes the frequencies, vibrational displacements and electron-phonon couplings for both states too the standard output, including any error messages.

Introduction

Franck-Condon factors are the squares of the overlap integrals of vibrational wave functions. Given a transition between two electronic, spin or charge states, the Franck-Condon factors represent the probabilities for accompanying vibrational transitions. As such, they can be used to predict the relative intensities of absorption or emission lines in spectroscopy or excitation lines in transport measurements.

When a molecule makes a transition to another state, the equilibrium position of the nuclei changes, and this will give rise to vibrations. To determine which vibrational modes will be active, we first have to express the displacement of the nuclei in the normal modes:

$$\mathbf{k} = \mathbf{L'}^T \mathbf{m}^{1/2}(\mathbf{B}_0\mathbf{x}_0 - \mathbf{x'}_0)$$

Here, $\mathbf{k}$ is the displacement vector, $\mathbf{L}$ is the normal mode matrix, $\mathbf{m}$ is a matrix with the mass of the nuclei on the diagonal, $\mathbf{B}$ is the zero-order axis-switching matrix and $\mathbf{x}_0$ is the equilibrium position of the nuclei. For free molecules, depending on symmetry constraints, the geometries of both states may have been translated and/or rotated with respect to each other. To remove the six translational and rotational degrees of freedom, we can center the equilibrium positions around the center of mass and rotate one of the states to provide maximum overlap. The latter is included with the zero-order axis-switching matrix $\mathbf{B}$, implemented according to [419 (page 931)].

When we have obtained the displacement vector, it is trivial to calculate the dimensionless electron-phonon couplings. They are given by:

$$\lambda = (\mathbf{\Gamma}/2)^{1/2}\mathbf{k}$$

Here, $\mathbf{\Gamma} = 2\pi\omega/h$ is a vector containing the reduced frequencies. [420 (page 931)]. The Huang-Rhys factor $\mathbf{g}$ is related to λ as:

$$\mathbf{g} = \lambda^2$$

The reorganization energy per mode is

$$\mathbf{E} = (h/2\pi) * \omega\lambda^2$$

When the displacement vector $\mathbf{k}$, the reduced frequencies $\mathbf{\Gamma}$ and $\mathbf{\Gamma}'$, and the Duschinsky rotation matrix $\mathbf{J} = \mathbf{L'}^T\mathbf{B}_0\mathbf{L}$ have been obtained, the Franck-Condon factors can be calculated using the two-dimensional array method of Ruhoff and Ratner[420 (page 931)].

There is one Franck-Condon factor for every permutation of the vibrational quanta over both states. Since they represent transition probabilities, all Franck-Condon factors of one state which respect to one vibrational state of the other state must sum to one. Since the total number of possible vibrational quanta, and hence the total number of permutations, is infinite, in practice we will calculate the Franck-Condon factors until those sums are sufficiently close to one. Since the number of permutations rapidly increases with increasing number of vibrational quanta, it is generally possible to already stop after the sum is greater than about two thirds. The remaining one third will be distributed over so many Franck-Condon factors that their individual contributions are negligible.

In the limiting case of one vibrational mode, with the same frequency in both states, the expression for the Franck-Condon factors of transitions from the ground vibrational state to an excited vibrational state are given by the familiar expression:

$$|l_{0,n}|^2 = e^{-\lambda^2}\lambda^{2n}/n!$$

Input

The input for *fcf* is keyword oriented and is read from the standard input. *fcf* recognizes several keywords, but only two have to be specified to perform the calculation. All input therefore contains at least two lines of the following form:

```
$ADFBIN/fcf << eor
STATES state1 state2
QUANTA l1 l2
eor
```

STATES state1 state2 The filenames of two TAPE21 files resulting from a numerical or analytical frequency calculation. The calculations must have been performed on the same molecule, i.e. the type, mass and order of occurrence of all the atoms (or fragments) has to be the same in both files.

(optional) MODES first last The first and last mode to be taken into account in the calculation. If this option is omitted, all modes are taken into account. This option can be used to effectively specify and energy range for the Franck-Condon factors. When using this options, always check if the results (electron-phonon couplings, ground state to ground overlap integral, average sum of Franck-Condon factors, etc.) do not change too much.

(optional) LAMBDA lambda The minimum value of the electron-phonon coupling for a mode to be taken into account in the calculation. The default value is zero. Together with the MODES option, this provides a way to significantly reduce the total number of Franck-Condon factors. As with the MODES option, always check if the results do not change too much.

QUANTA l1 l2 The maximum number of vibrational quanta to be taken into account for both states. Franck-Condon factors will be calculated for every permutation of up to and including l1/l2 quanta over the vibrational modes.

(optional) TRANSLATE Move the center of mass of both geometries to the origin.

(optional) ROTATE Rotate the geometries to maximize the overlap of the nuclear coordinates.

(optional) SPECTRUM freqmin freqmax nfreq If SPECTRUM is included the vibrational spectrum is calculated. A histogram of the spectrum is calculated for the frequency range that is provided on input. The three parameters that define the frequency range are:

> **freqmin** minimum frequency for which the spectrum is calculated.

> **freqmax** maximum frequency for which the spectrum is calculated.

> **nfreq** number of frequencies for which the spectrum is calculated.

Only a few keys from the TAPE21 file are used for the calculation of the Franck-Condon factors. Disk space usage can be significantly reduced by extracting just these keys from the TAPE21 file before further analysis. The following shell script will extract the keys from the KF file specified by the first argument and store them in a new KF file specified by the second argument using the *cpkf* utility:

```
#!/bin/sh
cpkf $1 $2 "Geometry%nr of atoms" "Geometry%xyz" "Geometry%nr of atomtypes" \
"Geometry%fragment and atomtype index" "Geometry%atomtype" "Geometry%mass" \
"Freq%Frequencies" "Freq%Normalmodes"
```

Result: TAPE61

After a successful calculation, *fcf* produces a TAPE61 KF file. All results are stored in the Fcf section:

contents of TAPE61	comments
firstmode, lastmode	the first and last vibrational mode taken into account
lambda	the minimum value of the electron-phonon coupling
maxl1, maxl2	maximum level (or maximum number of vibrational quanta) in both states
translate, rotate	whether the TRANSLATE and ROTATE options were specified in the input
natoms	number of atoms in the molecule
mass	atomic mass vector ($\mathbf{m}$)
xyz1, xyz2	equilibrium geometries of both states ($\mathbf{x}_0$ and $\mathbf{x}_0$ ')
b0	zero-order axis-switching matrix matrix ($\mathbf{B}_0$)
nmodes	number of vibrational modes with a non-zero frequency
gamma1, gamma2	reduced frequencies of both states (Γ and Γ')
lmat1, lmat2	mass-weighted normal modes of both states ($\mathbf{L}$ and $\mathbf{L}$')
jmat	Duschinsky rotation matrix ($\mathbf{J}$)
kvec1, kvec2	displacement vectors for both states ($\mathbf{k}$ and $\mathbf{k}$', kvec1 is used for the calculation of the Franck-Condon factors)
lambda1, lambda2	electron-phonon couplings for both states (λ and λ')
maxp1, maxp2	maximum number of permutations of maxl1/maxl2 quanta over the vibrational modes
i0	ground state to ground state overlap integral ($I_{0,0}$)
freq1, freq2	frequencies of every permutation of the vibrational quanta for both states
fcf	maxp1 by maxp2 Franck-Condon factor matrix
fcfsum1, fcfsum2	average sum of the Franck-Condon factors for both states

In addition to producing a binary TAPE61 file, *fcf* also writes the frequencies, displacement vectors and electron-phonon couplings for both states to the standard output.

6.5.2 Example absorption and fluorescence

In this example it is assumed that the molecule has a singlet ground state S_0 , and the interesting excited state is the lowest singlet excited state S_1 . First one needs to a the ground state geometry optimization, followed by a frequency calculation. The TAPE21 of the ground state frequency calculation will be called s0.t21. Next one needs to do an excited state geometry optimization. Here it is assumed that the lowest singlet excited state S_1 is of interest:

```
EXCITATION
   Onlysing
   Lowest 1
END
EXCITEDGO
   State A 1
   Singlet
END
GEOMETRY
END
```

To get the frequencies for this excited state, numerical frequencies need to be calculated, at the optimized geometry of the first excited state:

```
EXCITATION
ONLYSING
lowest 1
END
```

```
EXCITEDGO
    State A 1
    Singlet
END
GEOMETRY
    frequencies
END
...
mv TAPE21 s1.t21
```

Next for the absorption spectrum, we look at excitations from the lowest vibrational state of the electronic ground state to the vibrational levels of the first singlet excited state S_1 (S_1 $leftarrow$ S_0), using the *FCF program* (page 184), which calculates the Franck-Condon factors between the vibrational modes of the two electronic states, with input

```
STATES s0.t21 s1.t21
QUANTA 0 5
SPECTRUM 0 10000 1001
TRANSLATE
ROTATE
```

The number of vibrational quanta for the excited state should be larger in case of small molecules. See *the description of FCF program* (page 184) for more details.

For the fluorescence spectrum, we look at excitations from the lowest vibrational state of the first singlet excited state S_1 to the vibrational levels of the singlet ground state state S_0 ($S_1 \rightarrow S_0$). Input for the *FCF program* (page 184) is in this case:

```
STATES s0.t21 s1.t21
QUANTA 5 0
SPECTRUM -10000 0 1001
TRANSLATE
ROTATE
```

The number of vibrational quanta for the ground state should be larger in case of small molecules.

Note that the FCF program calculates the spectrum relative to the 0-0 transition. Thus one should add to spectrum calculated with FCF the difference in energy of the lowest vibrational state of the ground state S_0 and the lowest vibrational state of the electronically singlet excited state S_1 .

6.5.3 Example phosphorescence

In this example it is assumed that the molecule has a singlet ground state S_0 , and the interesting excited state is the lowest triplet excited state T_1 . Emission from a triplet state to a singlet state is spin forbidden, however, due to spin-orbit coupling such transitions may occur. In the following we assume that the geometry of the triplet excited state is not influenced much by spin-orbit coupling.

First one needs to a the ground state geometry optimization, followed by a frequency calculation. The TAPE21 of the ground state frequency calculation will be called s0.t21. Next one needs to do an excited state geometry optimization of the lowest triplet excited state, followed by a frequency calculation.

```
CHARGE 0.0 2.0
UNRESTRICTED
GEOMETRY
END
AnalyticalFreq
End
```

```
...
mv TAPE21 t1.t21
```

For the phosphorescence spectrum, we look at excitations from the lowest vibrational state of the first triplet excited state T_1 to the vibrational levels of the singlet ground state state S_0 ($T_1 \rightarrow S_0$). Input for the *FCF program* (page 184) is in this case:

```
STATES s0.t21 t1.t21
QUANTA 5 0
SPECTRUM -10000 0 1001
TRANSLATE
ROTATE
```

The number of vibrational quanta for the ground state should be larger in case of small molecules.

Note that the FCF program calculates the spectrum relative to the 0-0 transition. Thus one should add to spectrum calculated with FCF the difference in energy of the lowest vibrational state of the ground state S_0 and the lowest vibrational state of the electronically triplet excited state T_1.

Zero field splitting (ZFS) and the radiative rate constants (i.e. radiative phosphorescence lifetimes) could be calculated with spin-orbit coupled ZORA time-dependent density functional theory (ZORA-TDDFT). In ADF spin-orbit coupling can be treated self-consistently (i.e. non perturbatively) during both the SCF and TDDFT parts of the computation, see *the section on excitation energies and spin-orbit coupling* (page 170).

An alternative to the use of the unrestricted formalism to calculate the lowest triplet excited state is to use the TDDFT formalism:

```
EXCITATION
    Onlytrip
    Lowest 1
END
EXCITEDGO
    State A 1
    Triplet
END
GEOMETRY
END
```

To get the frequencies for this excited state, numerical frequencies need to be calculated, at the optimized geometry of the first excited state.

```
EXCITATION
    Onlytrip
    Lowest 1
END
EXCITEDGO
    State A 1
    Triplet
END
GEOMETRY
    frequencies
END
...
mv TAPE21 t1.t21
```

6.6 (Hyper-)Polarizabilities, ORD, magnetizabilities, Verdet constants

A (frequency dependent) electric field induces a dipole moment in a molecule, which is proportional to the (frequency dependent) molecular polarizability. Van der Waals dispersion coefficients describe the long-range dispersion interaction between two molecules. Optical rotation or optical activity (ORD) is the rotation of linearly polarized light as it travels through certain materials. A (frequency dependent) magnetic field induces a magnetic moment in a molecule, which is proportional to the (frequency dependent) molecular magnetizability. The Faraday effects describes the rotation of the plane-polarized light due to a magnetic field, which is proportional to the intensity of the component of the magnetic field in the direction of the beam of light. The Verdet constant describes the strength of the Faraday effect for a particular molecule. All these properties are available in ADF as applications of time-dependent DFT (TDDFT).

Two keys can be used for calculating these properties:

- RESPONSE

- AORESPONSE

The RESPONSE key has many unique features cpmpared to the AORESPONSE key, like the use of symmetry during the calculation. The AORESPONSE key also has many unique features compared to the RESPONSE key, namely lifetime effects (polarizabilities at resonance), the calculation of (dynamic) magnetizabilities, Verdet constants, the Faraday B terms, and an alternative way to calculate (resonance) Raman scattering factors. Both RESPONSE and AORESPONSE can calculate polarizabilities in case of spin-orbit coupling.

6.6.1 RESPONSE: (Hyper-)Polarizabilities, ORD

RESPONSE: Polarizabilities

The calculation of frequency-dependent (hyper)polarizabilities and related properties (Raman, ORD) is activated with the block key RESPONSE

```
RESPONSE
END
```

In this example only the *zz* component of the dipole polarizability tensor is calculated, at zero frequency. The orientation of the molecule is therefore crucial. Be aware that the program may modify the orientation of the molecule if the input coordinates do not agree with the symmetry conventions in ADF! (This calculation could equivalently be done through a finite field method).

See also the alternative implementation with the AORESPONSE key that offers some unique features like magnetizability, and lifetime options.

The impact of various approximations on the quality of computed polarizabilities has been studied in, for instance, Refs. [*74* (page 911), *82* (page 911), *89* (page 912)]. If you are new to this application field, we strongly recommend that you study a few general references first, in particular when you consider hyperpolarizability calculations. These have many pitfalls, technically (which basis sets to use, application of the DEPENDENCY key) and theoretically (how do theoretical tensor components relate to experimental quantities, different conventions used). Please, take a good look both at ADF-specific references [*75-77* (page 911), *90* (page 912)] and at general references related to this subject: Refs. [*91-93* (page 912)], the entire issues of Chem.Rev.94, the ACS Symposium Series #628, and further references in the ADF-specific references.

```
RESPONSE
  ALLCOMPONENTS
  Nfreq Nfreq
  FrqBeg FirstFreq
  FrqEnd LastFreq
  ALLTENSOR
```

```
    Quadrupole
    Octupole
    Traceless
END
```

Entire tensor or only one component You specify the ALLCOMPONENTS subkey to get the entire polarizability tensor, instead of just the *zz* component.

Frequencies or wavelengths Instead of performing the calculation at zero frequency (which results in the *static* polarizability), one can specify an even-spaced sequence of frequencies, using the subkeys `Nfreq`, `FrqBeg`, and `FrqEnd` with obvious meaning. The unit of `FrqBeg`, and `FrqEnd` is eV.

Higher multipole polarizabilities Instead of just calculating the dipole-dipole polarizability, one may address the dipole-quadrupole, quadrupole-quadrupole, dipole-octupole, quadrupole-octupole, and octupole-octupole polarizability tensors. These can all be calculated in a single run, using the subkey ALLTENSOR. If only quadrupole-quadrupole or octupole-octupole tensors are needed, the subkey quadrupole or octupole should be used.

Traceless If the subkey TRACELESS is included this ensures the calculation of tensors using the traceless form of the quadrupole operator.

RESPONSE: Accuracy and convergence

```
RESPONSE
  erralf 1e-6
  erabsx 1e-6
  errtmx 1e-6
  ncycmx 30
END
```

erralf, erabsx, errtmx The subkeys erralf, erabsx, errtmx determine the convergence criteria for a polarizability calculation. The strict defaults are shown. It is rarely necessary to change the defaults, as these are rather strict but do not lead to many iterations.

ncycmx The maximum number of attempts within which the algorithm has to converge. The default appears to be adequate in most cases.

RESPONSE: Hyperpolarizabilities

Hyperpolarizabilities

```
RESPONSE
  HYPERPOL LaserFreq
END
```

The first hyperpolarizability tensor β is calculated (in atomic units in the 'theoreticians convention', i.e. convention T=AB in Ref. [92 (page 912)]) if the subkey HYPERPOL is present with a specification of the laser frequency (in Hartree units). If also the subkey ALLCOMPONENTS is specified, all components of the hyperpolarizability tensor will be obtained.

As mentioned before, by default only the static dipole hyperpolarizability tensor is computed. If one is interested in the frequency-dependent hyperpolarizability, the input could look like:

```
RESPONSE
  ALLCOMPONENTS
  HYPERPOL 0.01
```

```
  DYNAHYP
END
```

The subkey DYNAHYP has to be added and the main frequency ω has to be specified in Hartrees after the subkey hyperpol. In the output all nonzero components of the tensors governing the static first hyperpolarizability, second harmonic generation, electro-optic Pockels effect, and optical rectification are printed.

Note: Second hyperpolarizabilities are currently not available analytically in case of RESPONSE. Some can however be obtained by calculating the first hyperpolarizability in a finite field.

The effect of using different DFT functionals (LDA, LB94, BLYP) on hyperpolarizabilities in small molecules has been investigated in [*77* (page 911)].

RESPONSE: Optical rotation dispersion (ORD)

```
RESPONSE
  OPTICALROTATION
END
```

OPTICALROTATION With the subkey OPTICALROTATION the (frequency dependent) optical rotation [*80* (page 911), *94* (page 912)] will be calculated. For correct calculations one should calculate the entire tensor (see also the subkey ALLCOMPONENTS), which is done automatically.

An alternative implementation uses the AORESPONSE key, in which life time effects can be included.

6.6.2 AORESPONSE: Lifetime effects, (Hyper-)polarizabilities, ORD, magnetizabilities, Verdet constants

The AORESPONSE key offers some unique features compared to the RESPONSE key, namely lifetime effects (polarizabilities at resonance), the calculation of (dynamic) magnetizabilities, Verdet constants, the Faraday B terms, and an alternative way to calculate (resonance) Raman scattering factors. Note that the RESPONSE key also has many unique features, like the use of symmetry during the calculation.

AORESPONSE: Polarizabilities

If the block key AORESPONSE is used, by default, the polarizability is calculated.

```
AORESPONSE
  FREQUENCY      Nfreq freq1 freq2 ... freqN units
  FREQRANGE      freq1 freqN TotFreq units
  LIFETIME width
END
```

FREQUENCY Nfreq freq1 freq2 ... freqN units To calculate time-dependent properties, one needs to specify frequency of perturbation field. Here Nfreq specifies the number of frequencies that follow. The last item on the line specifies the units and is one of EV, HARTREE, ANGSTROM.

FREQRANGE freq1 freqN TotFreq units This key is useful when it is necessary to specify more than 20 equally spaced frequencies for the response calculations. The first frequency is freq1 and the last one is freqN. The total number of frequencies including the first and the last one is TotFreq. The last item specifies the units: EV, HARTREE or ANGSTROM.

LIFETIME width Specify the resonance peak width (damping) in Hartree units. Typically the lifetime of the excited states is approximated with a common phenomenological damping parameter. Values are best obtained

by fitting absorption data for the molecule, however, the values do not vary a lot between similar molecules, so it is not hard to estimate values. A value of 0.004 Hartree was used in Ref. [*266* (page 922)].

The spin-orbit ZORA polarizability code (Ref. [*311* (page 925)]) is automatically selected if the AORESPONSE keyword is given in a spin-orbit coupled calculation. In this case a spin-restricted calculation is required, but, unlike the rest of AORESPONSE, also SYMMETRY NOSYM. Spin-polarization terms in the XC response kernel are neglected. In Ref. [*311* (page 925)] the imaginary polarizability dispersion curves (spin-restricted) match well the broadened spin-orbit TDDFT data from Ref. [*182* (page 917)]. Thus the corrections from the spin-polarization terms appear to be rather minor. No picture change corrections were applied in the ZORA formalism. AORESPONSE icw hybrids icw spin-orbit is not implemented.

AORESPONSE: Technical parameters and expert options

```
AORESPONSE
 ...
  SCF        {NOCYC} {NOACCEL} {CONV=conv} {ITER=niter}
  GIAO
  FITAODERIV
  COMPONENTS {XX} {XY} {XZ} {YX} {YY} {YZ} {ZX} {ZY} {ZZ}
  ALDA|XALPHA
  ALPHA
END
```

SCF {NOCYC} {NOACCEL} {CONV=conv} {ITER=niter} Specify CPKS parameters such as the degree of convergence and the maximum number of iterations:

NOCYC - disable self-consistence altogether

NOACCEL - disable convergence acceleration

CONV - convergence criterion for CPKS. The default value is 10^{-6} . The value is relative to the uncoupled result (i.e. to the value without self-consistence).

ITER - maximum number of CPKS iterations, 50 by default.

Specifying ITER=0 has the same effect as specifying NOCYC.

GIAO Include the Gauge-Independent Atomic Orbitals (GIAO). This option should not be used with damping (LIFE-TIME keyword) and the VELOCITYORD option should be used instead.

FITAODERIV Use fitted AO Derivatives. This will improve the density fitting, can only be used in cae of STO fitting. In case of ZlmFit one can improve the fitting with the ZLMFIT block key.

COMPONENTS {XX} {XY} {XZ} {YX} {YY} {YZ} {ZX} {ZY} {ZZ} Limit the tensor components to the specified ones. Using this option may save the computation time.

ALDA|XALPHA If ALDA is specified the VWN kernel is used. This option is the default. If ALPHA is specified the $X\alpha$ kernel is used instead of the default VWN one.

ALPHA Writes perturbed density matrix to TAPE16.

AORESPONSE: Damped First Hyperpolarizabilities

In Ref.[*451* (page 933)] an implementation of finite lifetimes into TDDFT for the calculation of quadratic response properties in ADF is described. All β tensor components (27 in total) will be printed in the output.

```
AORESPONSE
  BETA|QUADRATIC
  FREQUENCY  2 freq1 freq2 units
```

```
  LIFETIME width
  STATIC|OPTICALR|EOPE|SHG
END
```

BETA Option to calculate the damped first hyperpolarizability (β) using quadratic response theory based on the 2n+1 rule. Two input frequencies are required for this calculation and the property $\beta(-(\omega_1 + \omega_2); \omega_1, \omega_2)$ will be the output. Note that one can choose certain values of the two frequencies to calculate different types of β, such as static case $\beta(0; 0, 0)$, optical rectification $\beta(0; \omega, -\omega)$, electro-optical Pockels effect $\beta(-\omega; \omega, 0)$, and second harmonic generation $\beta(-2\omega; \omega, \omega)$. Alternatively, these can be efficiently calculated using the (sub-)keywords STATIC, OPTICALR, EOPE, and SHG, respectively. Note that the needed input frequencies all rely on ω_1 (freq1) when using the (sub)keywords above.

QUADRATIC This option possess the same functionality as BETA, i.e., calculate the damped β, except not adapting the 2n+1 rule. The (sub)keywords STATIC, OPTICALR, EOPE,and SHG are also applicable here. Note that this approach facilitates the direct partitioning of the response into contributions from localized orbitals and is important for natural bond analysis.

Note: Please only use HARTREE or EV as the unit for the input frequencies. The unit option ANGSTROM does not work correctly due to the current implementation structure.

AORESPONSE: Damped Second Hyperpolarizabilities

In Ref.[*452* (page 933)] a general implementation for damped cubic response properties is presented using time-dependent density functional theory (TDDFT) and Slater-type orbital basis sets. To directly calculate two-photon absorption (TPA) cross sections, an implementation of a reduced damped cubic response approach is described in Ref.[*452* (page 933)]. All γ tensor components (81 in total) will be printed in the output.

```
AORESPONSE
  GAMMA|CUBIC
  FREQUENCY  3 freq1 freq2 freq3 units
  LIFETIME width
  STATIC|EFIOR|OKE|IDRI|EFISHG|THG|TPA
END
```

GAMMA Option to calculate the damped second hyperpolarizability (γ) using cubic response theory based on the 2n+1 rule. Three input frequencies are required for this calculation and the property $\gamma(-(\omega_1 + \omega_2 + \omega_3); \omega_1, \omega_2, \omega_3)$ will be the output. Note that one can choose certain values of the three frequencies to calculate different types of γ, such as static case $\gamma(0; 0, 0, 0)$, electric field induced optical rectification $\gamma(0; \omega, -\omega, 0)$, optical Kerr effect $\gamma(-\omega; \omega, 0, 0)$, intensity dependent refractive index $\gamma(-\omega; \omega, \omega, -\omega)$, electric field induced second harmonic generation $\gamma(-2\omega; \omega, \omega, 0)$, and third harmonic generation $\gamma(-3\omega; \omega, \omega, \omega)$. Alternatively, these can be efficiently calculated using the (sub)keywords STATIC, EFIOR, OKE, IDRI, EFISHG, and THG, respectively. The (sub)keyword TPA can be used to calculate the γ corresponding to the two photon absorption process (i.e., the reduced form of $\gamma(-\omega; \omega, \omega, -\omega)$), however, it can ONLY be used with keyword GAMMA. Note that the needed input frequencies all rely on ω_1 (freq1) when using the (sub)keywords above.

CUBIC This option possess the same functionality as GAMMA, i.e., calculate the damped γ, except not adapting the 2n+1 rule. The (sub)keywords STATIC, EFIOR, OKE, IDRI, EFISHG and THG are also applicable here. Note that this approach facilitates the direct partitioning of the response into contributions from localized orbitals and is important for natural bond analysis.

Note: Please only use HARTREE or EV as the unit for the input frequencies. The unit option ANGSTROM does not work correctly due to the current implementation structure.

AORESPONSE: ORD

```
AORESPONSE
  OPTICALROTATION
  VELOCITYORD
  FREQUENCY      Nfreq freq1 freq2 ... freqN units
  FREQRANGE      freq1 freqN TotFreq units
  LIFETIME width
END
```

OPTICALROTION Specify OPTICALROTION to calculate optical rotatory dispersion spectrum instead of polarizabilities.

VELOCITYORD This option should be used instead of OPTICALROT with GIAO if the finite lifetime effects need to be taken into account (LIFETIME option).

AORESPONSE: magnetizabilities, Verdet constants, Faraday B term

```
AORESPONSE
  MAGNETICPERT
  MAGOPTROT
  FREQUENCY      Nfreq freq1 freq2 ... freqN units
  FREQRANGE      freq1 freqN TotFreq units
  LIFETIME width
END
```

MAGNETICPERT Calculate static or time-dependent magnetizability, see also Ref. [*230* (page 920)].

MAGOPTROT Specify MAGOPTROT to calculate the Verdet constant instead of polarizability, see for the details of the implementation Ref. [*307* (page 925)]. When it is specified together with the LIFETIME key the real and imaginary part of the damped Verdet constant will be calculated. Combination of three keys MAGOPTROT, LIFETIME and FREQRANGE yields the magnetic optical rotatory dispersion and magnetic circular dichroism spectrum (Faraday A and B terms) calculated simultaneously in the range from freq1 to freqN. It is also possible to combine MAGOPTROT, LIFETIME and FREQUENCY. In order to obtain the Faraday B terms from the Verdet constant calculations it is necessary to perform several steps, involving a fit of the imaginary Verdet data to the MCD spectrum. You can request SCM for details on the fitting procedure. For details of the method, see Ref. [*272* (page 923)].

AORESPONSE: Raman

```
AORESPONSE
  RAMAN
  FREQUENCY  1  freq1 units
  LIFETIME width
END
```

RAMAN Calculates the Raman scattering factors. The AOREPONSE-Raman only works with one frequency. If one frequency is specified the Raman scattering factors are calculated at that frequency. The Raman option is compatible with the lifetime option so that resonance Raman scattering can be calculated. For details of this method, see Ref. [*266* (page 922)]. To get Raman intensities with AORESPONSE, numerical frequencies need to be calculated using a FREQUENCIES key in the GEOMETRY input block. Non-resonance Raman intensities can also be obtained using the RESPONSE key or, alternatively, using RAMANRANGE in combination with analytically or numerically pre-calculated frequencies.

Applications of AORESPONSE

It may be useful to consult the following applications of the AORESPONSE key in ADF:

- Calculation of static and dynamic linear magnetic response in approximate time-dependent density functional theory [*230* (page 920)]

- Calculation of CD spectra from optical rotatory dispersion, and vice versa, as complementary tools for theoretical studies of optical activity using time-dependent density functional theory [*231* (page 920)]

- Calculation of origin independent optical rotation tensor components for chiral oriented systems in approximate time-dependent density functional theory [*232* (page 920)]

- Time-dependent density functional calculations of optical rotatory dispersion including resonance wavelengths as a potentially useful tool for determining absolute configurations of chiral molecules [*233* (page 920)]

- Calculation of optical rotation with time-periodic magnetic field-dependent basis functions in approximate time-dependent density functional theory [*234* (page 920)]

- A Quantum Chemical Approach to the Design of Chiral Negative Index Materials [*235* (page 920)]

- Calculation of Verdet constants with time-dependent density functional theory. Implementation and results for small molecules [*236* (page 920)]

- Calculations of resonance Raman [*266,267* (page 922)]

- Calculations of surface-enhanced Raman scattering (SERS) [*268,269* (page 922)]

- Calculation of magnetic circular dichroism spectra from damped Verdet constants [*272* (page 923)]

- Calculation of the polarizability in case of spin-orbit coupling [*311* (page 925)]

See also:

- advanced tutorial on plasmon enhanced two photon absorption

6.6.3 POLTDDFT: Damped Complex Polarizabilities

A fast algorithm to solve the Time Dependent Density Functional Theory (TDDFT) equations in the space of the density fitting auxiliary basis set has been developed and implemented [*480* (page 935)]. The method, named POLT-DDFT, extracts the spectrum from the imaginary part of the polarizability at any given photon energy, avoiding the bottleneck of Davidson diagonalization. The original idea which made the present scheme very efficient consists in the simplification of the double sum over occupied-virtual pairs in the definition of the complex dynamical polarizability, allowing an easy calculation of such matrix as a linear combination of constant matrices with photon energy dependent coefficients. The method has been extended for the calculation of circular dichroism spectra [*481* (page 935)].

It is very important to use specially made auxiliary fit sets, which are available only for a very limited amount of elements. Symmetry and frozen cores can be used. Should not be used for hybrids, meta-hybrids, or range-separated functionals. Should not be used icw spin-orbit coupling. STOFIT can not be used.

UV/Vis spectra, CD spectra

If one includes the POLTDDFT keyword the (real and imaginary part of the) diagonal of the polarizability tensor and rotatory strengths will be calculated, which can be used to calculate the photoabsorption and circular dichroism (CD) spectra.

```
POLTDDFT
  IRREP
     Irrep1
     Irrep2
```

```
     ...
  SUBEND
  KGRID eVgrid ngrid
  FREQRANGE eVi eVf nfreq
  LIFETIME eVwidth
  CUTOFF eVcutoff
  LAMBDA lambda
  VELOCITY
END
```

IRREP Subblock key for selecting which symetry irreps of the excitations to calculate (all excitations by default). In the subkey data block list the symmetry irrep labels, like B1, for example.

KGRID eVgrid ngrid Keyword KGRID is used to discretize the energy scale for calculating the complex dynamical polarizability. Only pairs of an occupied and virtual orbital are included, for which the orbital energy difference is lower than eVgrid (9 eV by default). Ngrid is the number of points within the energy grid (180 by default).

FREQRANGE eVi eVf nfreq Keyword FREQRANGE is used to specify the equally spaced points in the spectrum for which one would like to calculate the complex dynamical polarizability. The first point is eVi (0 eV by default) and the last one is eVf (5 eV by default). The total number of points in the spectrum is nfreq (100 by default).

LIFETIME eVwidth Specify the resonance peak width (damping) in eV units. Typically the lifetime of the excited states is approximated with a common phenomenological damping parameter. Values are best obtained by fitting absorption data for the molecule, however, the values do not vary a lot between similar molecules, so it is not hard to estimate values. Default value is 0.1 eV.

CUTOFF eVcutoff For a given point in the spectrum, only include pairs of an occupied and virtual orbital, for which the orbital energy difference is lower than the energy of the point in the spectrum plus eVcutoff. The default value for eVcutoff is 4 eV.

LAMBDA lamba Jacob's scaling factor [*482* (page 935)] for the study of plasmonic resonances. This factor, 0<lambda<1 (default lambda=1), turns on the coupling matrix K.

VELOCITY If the subkey VELOCITY is included ADF calculates the dipole moment in velocity gauge. Default the dipole-length representation is used.

Reduced fit set

For POLTDDFT it very important is to use specially made auxiliary fit sets. These are available only for a very limited amount of elements, and for a limited amount of basis sets. These special basis sets can be found in the atomic database directories, with a directory name REDFIT, for example, the directory $ADFHOME/atomicdata/ZORA/TZP/REDFIT. An example:

```
Basis
  Au ZORA/DZ/REDFIT/Au.4f
End
```

	Available basis sets (July 2018)
ZORA/DZ/REDFIT	Ag.4p, Au.4f, B.1s, C.1s, F.1s, H, N.1s, S.2p
ZORA/TZP/REDFIT	Ag.4p, Au.4f, B.1s, Br.3d, C.1s, Cl.2p, F.1s, H, I.4d, Mo.4p, N.1s, O.1s, S.2p
ZORA/TZ2P/REDFIT	Ag.4p, Mo.4p, S.2p

6.6.4 Van der Waals dispersion coefficients

```
RESPONSE
  ALLCOMPONENTS
  VANDERWAALS NVanderWaals
  {ALLTENSOR}
END
```

Dispersion coefficients Simple dispersion coefficients (the dipole-dipole interaction between two identical molecules, the C_6 coefficient) are calculated in a single ADF calculation. General dispersion coefficients are obtained with the auxiliary program DISPER, which uses two output files (file named TENSOR) of two separate ADF runs as input. See the Properties and the Examples documents. To get the dispersion coefficients one has to calculate polarizabilities at imaginary frequencies between 0 and infinity. The ADF program chooses the frequencies itself. The user has to specify the number of frequencies, which in a sense defines the level of accuracy, as an argument to the subkey VanDerWaals.

NVanderWaals One can specify the number of frequencies with NVanderWaals. Ten frequencies is reasonable. Without the key ALLTENSOR only dipole-dipole interactions are considered. If ALLTENSOR is specified, higher dispersion coefficients are also calculated. This ADF calculation generates a file with name TENSOR, which contains the results of multipole polarizabilities at imaginary frequencies. This TENSOR file has to be saved. Similarly, the TENSOR file for the second monomer has to be saved. The files have to be renamed to files 'tensorA' and 'tensorB' (case sensitive) respectively. Then the program DISPER has to be called in the same directory where the 'tensorA' and 'tensorB' files are located. DISPER needs no further input.

DISPER program: Dispersion Coefficients

The DISPER program was originally written by V.Osinga [*79* (page 911)]. The original documentation was written by S.J.A. van Gisbergen.

Van der Waals dispersion coefficients

The program DISPER computes Van der Waals dispersion coefficients up to C_{10} for two arbitrary closed-shell molecules. ADF itself can already compute some C_6 and C_8 coefficients between two identical closed-shell molecules. These coefficients describe the long-range dispersion interaction between two molecules. It requires previous ADF-TDDFT calculations for the polarizability tensors at imaginary frequencies for the two interacting molecules. Each such ADF calculation produces a file TENSOR (if suitable input for ADF is given). The TENSOR files must be renamed tensorA and tensorB, respectively and must be present as local files for DISPER. The DISPER program takes no other input and prints a list of dispersion coefficients.

A schematic example, taken from the set of sample runs, for the usage of DISPER is the following:

Step1: run ADF for, say, the HF molecule. In the input file you specify the RESPONSE data block:

```
RESPONSE
  MaxWaals 8     ! Compute dispersion coefficients up to C8
  ALLTENSOR      ! This option must be specified in the ADF calc for a
                 ! subsequent DISPER run
  ALLCOMPONENTS  ! Must also be specified for DISPER
End
```

At the end of the run, copy the local file 'TENSOR' to a file 'tensorA'. For simplicity, we will now compute the dispersion coefficients between two HF molecules. Therefore, copy 'tensorA' to 'tensorB'.

Now run DISPER (without any other input). It will look for the local files 'tensorA' and 'tensorB' and compute corresponding dispersion coefficients to print them on standard output.

```
$ADFBIN/disper -n1 << eor
eor
```

The output might look something like this:

```
DISPER 2000.02 RunTime: Apr04-2001 14:14:13
  C-COEFFICIENTS
  n   LA  KA  LB  KB  L   coefficient(Y)      coefficient(P)
  6   0   0   0   0   0    28.29432373          28.29432373
  6   2   0   0   0   2     7.487547697          3.348533127
  8   0   0   0   0   0   416.1888455          416.1888455
  8   0   0   2   0   2     0.4323024202E-05     0.1933315197E-05
  8   2   0   0   0   2   402.3556946          179.9389368
  8   2   0   2   0   4     0.4238960180E-05
  8   4   0   0   0   4   -36.67895539         -12.22631846
  8   4   0   2   0   6    -0.2000286301E-05
```

The n-value in the first column refers to the long-range radial interaction. The case n=6 refers to the usual dipole-dipole type interaction related to a $1/R^6$ dependence in the dispersion energy. The n=7 case relates to a dipole-quadrupole polarizability on one system and a dipole-dipole polarizability on the other (this is not symmetric!). The n=8 term may contain contributions from a quadrupole-quadrupole polarizability on one system in combination with a dipole-dipole polarizability on the other as well as contributions from two dipole-quadrupole polarizabilities.

Terms which are zero by symmetry are not printed. In the example above, this is the case for all n=7 terms, because the systems (apparently) are too symmetric to have a nonzero dipole-quadrupole polarizability. The best known and most important coefficients are the isotropic ones, determining the purely radial dependence of the dispersion energy. They are characterized by the quantum numbers: 6 0 0 0 0 0 (or 8 0 0 0 0 0 etc.) Other combinations of quantum numbers refer to different types of angular dependence. The complete set determines the dispersion energy for arbitrary orientations between the two subsystems A and B.

The complete expressions are rather involved and lengthy. We refer the interested reader to the paper [79 (page 911)] which contains a complete description of the meaning of the various parts of the output, as well as references to the earlier literature which contain the mathematical derivations. In particular, a useful review, which was at the basis of the ADF implementation, is given in [414 (page 931)]. Of particular significance is Eq.(8) of the JCP paper mentioned above, as it defines the meaning of the calculated coefficients $C_n^{L_A, K_A, L_B, K_B, L}$ as printed above.

For highly symmetric systems, a different convention is sometimes employed. It is based on Legendre polynomials (hence the 'P' in the final column) instead of on the spherical harmonics (the 'Y' in the column before the last). The 'P' coefficients are defined only for those coefficients that are nonzero in highly symmetric systems and never contain additional information with respect to the 'Y' coefficients. They are defined [Eq. (14) in the mentioned J. Chem. Phys. paper] in terms of the 'Y' coefficients by:

$$C_n^L = (-1)^L C_n^{L,0,0,0,L} / \sqrt{2L+1}$$

Because the quality of the dispersion coefficients is determined by the quality of the polarizabilities that are the input for DISPER, it is important to get good polarizabilities from ADF. For that it is important, in the case of small systems, to use an asymptotically correct XC potential (several choices are available in ADF, such as SAOP or GRAC) and a basis set containing diffuse functions. We refer to the ADF User's Guide for details.

6.7 Ligand Field and Density Functional Theory (LFDFT)

6.7.1 Introduction

Hans Bethe and John Hasbrouck van Vleck have introduced the Ligand Field (LF) model simultaneously in the 1930s. This theory often plays a central role for molecules containing d- and/or f-elements with open shells. It is a semi-

empirical model with adjustable parameters. Twenty years later Christian Klixbüll Jorgensen and Klaus Erik Schäffer proposed the Angular Overlap Model (AOM) that is a transcription of LF theory using molecular orbitals, still using adjustable parameters. Starting from the very beginning of the twenty first century until nowadays, a new, non-empirical, Density Functional Theory (DFT) based Ligand Field model [*465* (page 934), *466* (page 934)] was developed, called LFDFT. LFDFT emerged through the collaboration of many European scientists during these years.

The key feature of the LFDFT approach is the explicit treatment of near degeneracy correlation using ad hoc Configuration Interaction (CI) within the active space of Kohn-Sham (KS) orbitals with dominant p-, d- or f-character. The calculation of the CI-matrices is based on a symmetry decomposition in the full rotation group and/or on a ligand field analysis of the energies of all single determinants (micro-states) calculated according to Density Functional Theory (DFT) for frozen KS-orbitals corresponding to the averaged configuration, possibly with fractional occupations, of the p-, d- or f-orbitals. In the literature one can find cases where this procedure yields multiplet energies with an accuracy of a few hundred wave numbers and fine structure splitting accurate to less than a tenth of this amount. This procedure was used to calculate different molecular properties, e.g. Zero Field Splitting (ZFS) [*467* (page 934)], Zeeman interaction [*468* (page 934)], Hyper-Fine Splitting (HFS) [*469* (page 934)], magnetic exchange coupling [*469* (page 934), *470* (page 934)], and shielding constants [*471* (page 934)].

Strictly speaking, LF theory is nothing but the consideration of active p-, d- and/or f-electrons moving in the potential of a passive chemical environment. A rigorous formulation of a passive chemical environment could be given using a frozen density embedding (FDE) scheme. However, in practice, it turns out that the method described in Ref. [*472* (page 934)] using an effective Hamiltonian is more advantageous and yields more insights to the experimentalists. The LFDFT model was extended to two-open-shell systems [*473* (page 934)]. This has relevance for inter-shell transitions in lanthanides, important for understanding the optical and magnetic properties of rare-earth materials. It could also be used, for example, to calculate multiplet effects in X-ray absorption spectroscopy. Theory and applications for f $\rightarrow$ d transitions can be found in Refs. [*473-475* (page 934)].

6.7.2 Input description

Average of Configuration run

From a defined atomic structure (for example, obtained from geometry optimization or crystal .cif structure file), run a DFT calculation representing the electron configuration system from which the metal ion belongs. This DFT run is achieved by the Average of Configuration (AOC) calculation with fractional occupations of s, p, d or f orbitals, i.e. for $3d^7$ electron configuration of Co^{2+}, 7 electrons are evenly distributed in the molecular orbitals having dominant cobalt character. Choosing which orbitals have to be fractionally occupied may not always be trivial, since there may be several close-lying levels with different character. This may lead to problematic SCF convergence. One should check that the final fractionally occupied molecular orbitals indeed have the expected metal character, since otherwise the subsequent LFDFT calculation will be meaningless. It is required that the metal ion is placed as first atom in the coordinate system. Besides, the calculation implies a single-point spin restricted SCF without symmetry constraint (C1 point group = SYMMETRY NOSYM). A non-relativistic or scalar relativistic Hamiltonian should be used.

For a Co^{2+} d^7 electron configuration the input for ADF could be something like:

```
$ADFBIN/adf << eor
...
Atoms
  Co 0.0 0.0 0.0
  ...
End
Occupations
  A 56 1.4 1.4 1.4 1.4 1.4
End
relativistic scalar ZORA
SYMMETRY NOSYM
eor
```

LFDFT atomic database

The LFDFT atomic database is *not* included in the ADF distribution.

The easiest way to install it is using the ADF-GUI and open ADFinput and go to the LFDFT section. Then it will be installed in $ADFHOME/atomicdata or in "$HOME/Library/Application Support/SCM" (on a mac). If you do not have an ADF-GUI download the LFDFT atomic database LDFDT from http://downloads.scm.com/Downloads/lfdft/LFDFT.zip, and save the unzipped file in the aforementioned location. If you save this LFDFT atomic database in a different location, make sure to point the environment variable SCM_LFDFT to this directory.

Note that at present available electron configurations for the LFDFT atomic database are:

- s^n, n=0,..,2

- p^n, n=0,..,6

- d^n, n=0,..,10

- f^n, n=0,..,14

- $p^5 d^n$, n=0,..,10

- $f^n d^1$, n=1,..,13

LFDFT run

The TAPE21 file of the average of configuration (AOC) run should be an input file for the program lfdft. Results of the LFDFT calculation will be put on TAPE21. The input for lfdft is keyword oriented and is read from the standard input. Spin-orbit coupling is calculated using the ZORA equation by default. In cases where this is an underestimation, the old approximate method (a simple core potential) and/or an extra multiplication factor can be included.

```
$ADFBIN/lfdft << eor
NSHELL nshell
NLVAL1 nval1 lval1
{NLVAL2 nval2 lval2}
MOIND1 MO#1 MO#2 ... MO#(2*lval1+1)
{MOIND2 MO#1 MO#2 ... MO#(2*lval2+1)}
SOC soc1 {soc2}
SOCTYPE
   shell1 type1
   {shell2 type2}
End
eor
```

NSHELL nshell nshell is the number of shells in the electron configuration system under consideration: for Co^{2+} in $3d^7$ electron configuration, nshell = 1; for Fe^{2+} in $2p^5 3d^7$, nshell = 2. nshell should be equal to 1 or 2.

NLVAL1 nval1 lval1 nval1, and lval1, are the main quantum number n and l for shell 1, in case of all-electron calculations on the metal atoms. For 5d this is '5 2'. For frozen core calculations the number n should be reduced with the number of core levels with angular momentum l that are in the core. For example for 5d, with 3d and 4d in the core, one should use '3 2'.

NLVAL2 nval2 lval2 If nshell=2, nval2, and lval2, are the main quantum number n and l for shell 2.

MOIND1 MO#1 MO#2 ... MO#(2*lval1+1) MO#1, ..., are the indices of the molecular orbitals which have the most dominant nval1 lval1 character, for example the most dominant 3d character. They should match the fractionally occupied orbitals that are used in the AOC run for shell 1.

MOIND2 MO#1 MO#2 ... MO#(2*lval1+1) If nshell=2, MO#1, ..., are the indices of the molecular orbitals which have the most dominant nval2 lval2 character. They should match the fractionally occupied orbitals that are used in the AOC run for shell 2.

SOC soc1 {soc2} soc1 indicates if spin-orbit coupling is considered (1) or not (0) for shell 1. If nshell=2, same applies for soc2 for shell 2. soc1 and soc2 can also be real numbers, in case one wants to scale the calculated approximate spin-orbit coupling(s) by LFDFT. By default, the spin-orbit coupling is included (soc1 = 1, soc2 = 1).

```
SOCTYPE
  shell1 type1
  {shell2 type2}
End
```

type1 and type2 can be either 'zora' or 'core' and indicate whether the spin-orbit coupling for the given shell is calculated using the corresponding term in the zora equation (type = zora) or using the old approximation with the core potential only (type = core). The default is set to zora.

For example for a $3d^7$ electron configuration the input for lfdft could be

```
$ADFBIN/lfdft <<eor
NSHELL 1
NLVAL1 3 2
MOIND1 29 30 31 32 33
SOC    1
SOCTYPE
   shell1 zora
End
eor
```

LFDFT intensities

One can calculate excitation energies and oscillator strengths between two atomic multiplet states which come from different electron configurations with the module lfdft_tdm. Note that electronic transitions between two multiplet states which come from the same electron configuration are not dipole allowed. lfdft_tdm can calculate excitation energies and oscillator strengths (in the dipole approximation) from the calculated ground state multiplet of one electron configuration to all multiplet states that can be made for the other electron configuration. The calculated transition dipole moments and oscillator strengths are in arbitrary units. They are averaged over the degeneracy of the ground state as well as over the degeneracy of the excited state multiplet. The oscillator strength has to be multiplied with the degeneracy of the excited state multiplet.

The input for lfdft_tdm is keyword oriented and is read from the standard input.

```
$ADFBIN/lfdft_tdm << eor
STATE1 file1
STATE2 file2
eor
```

STATE1 file1 Filename file1 should be a result TAPE21 of a lfdft calculation, that contains the ground state electron configuration.

STATE1 file2 Filename file2 should be a result TAPE21 of a lfdft calculation, that contains the excited state electron configuration. Results of the lfdft_tdm calculation will be put on file2.

For example for Pr $4f^2 \rightarrow$ Pr $4f^1 5d^1$ the input could be something like:

```
$ADFBIN/lfdft_tdm <<eor
STATE1 Pr_f2.t21
STATE2 Pr_f1d1.t21
eor
```

In this case lfdft_tdm will calculate the excitations from the ground state of Pr $4f^2$ to all multiplet states that can be made with the Pr $4f^1 5d^1$ electron configuration.

6.8 NMR

NMR chemical shifts and NMR spin-spin couplings can be calculated. Effects due to spin-orbit coupling can be included. All electron basis sets can be used.

The separate program EPR/NMR ($ADFBIN/epr) program is no longer documented, since most of its capabilities are implemented in newer modules. See the documentation of the EPR/NMR module in the old ADF 2010 Properties document (http://www.scm.com/Doc/Doc2010/ADF/Properties/Properties.pdf).

- Chemical shifts and nuclear spin-spin coupling constants w. NBO analysis: Analysis of NMR parameters with Localized Molecular Orbitals
- Chemical shifts and nuclear spin-spin coupling constants w. visualization of the spectrum in GUI H-NMR spectrum with spin-spin coupling

6.8.1 NMR Chemical Shifts

NMR Chemical shifts have been implemented [*113-117* (page 913)] in a separate property program NMR. It requires the TAPE21 result file from an ADF calculation. The NMR module can be combined with the ZORA treatment for relativistic effects and with Spin-Orbit effects, making it suitable for treatment of heavy elements. See also the general review on relativistic computations of NMR parameters [*331* (page 926)].

Two GUI tutorials on NMR Chemical shift calculations are currently available:

- Chemical shifts and nuclear spin-spin coupling constants w. NBO analysis: Analysis of NMR parameters with Localized Molecular Orbitals
- Chemical shifts and nuclear spin-spin coupling constants w. visualization of the spectrum in the GUI: H-NMR spectrum with spin-spin coupling

Important notes

TAPE21 and TAPE10

NMR requires an ASCII input file and TAPE21 and TAPE10 result files from an ADF calculation on the molecule to be analyzed. The ADF result files TAPE21 and TAPE10 must be present with names TAPE21 and TAPE10 in the directory where you execute NMR. Use the keywords SAVE TAPE10 in the adf calculation in order to obtain a TAPE10 result file.

Recalculation of TAPE10 by NMR

Warning: the NMR property program will not always give the correct result for every SCF potential in the ADF calculation, like for example the SAOP potential, or if one uses COSMO in the ADF calculation, if one lets the NMR recalculate TAPE10. This is due to the GIAO method used in this program, which requires the calculation of the SCF potential, which is not done correctly for potentials, other than the standard LDA and GGA potentials. To obtain correct results one should, in addition to the use of TAPE21, also use TAPE10 that ADF generates, using the keywords SAVE TAPE10, and use it as input for the NMR property program. On TAPE10 the SCF potential is written, which is read in by the NMR program.

Atomic calculation

NMR calculations on 1 atom must have symmetry NOSYM.

Spin-orbit coupling

NMR calculations on systems computed by ADF with Spin Orbit relativistic effects included must have used NOSYM symmetry in the ADF calculation. NMR can also be combined with ADF ZORA calculations. The NMR program

reads from TAPE21 the relativistic option that is used in the ADF calculation, and will use the same relativistic option in the NMR calculations.

Kernel used in spin-orbit coupling

One can include an improved exchange-correlation kernel, as was implemented by J. Autschbach [*427* (page 932)], but this is not used by default, see subkey USE. This is an important option that should be seriously considered and has been advocated in Ref. [*427* (page 932)].

Bug spin-orbit part ADF2008 - ADF2013

In the ADF2008.01 a bug was introduced in the spin-orbit part of the calculated chemical shielding, which caused the calculated chemical shielding to be gauge dependent. This bug is relevant for spin-orbit coupled calculations for ADF versions ADF2008-ADF2013. In ADF2014 this bug has been fixed.

Unscaled ZORA default ADF2014

There is gauge dependence if the scaled ZORA method is used in the calculation of NMR chemical shieldings. Therefore the default method for NMR chemical shielding calculations is changed in ADF2014 to use the unscaled ZORA method.

SAOP

The use of the model SAOP potential leads to isotropic chemical shifts which are substantially improved over both LDA and GGA functionals, and of similar accuracy as results with a self-interaction-corrected functional (SIC), see [*421* (page 931)]. SAOP is computationally expedient and routinely applicable to all systems, requiring virtually the same computational effort as LDA and GGA calculations.

NICS

The Nucleus-Independent Chemical Shift (NICS) can be calculated at any point in the molecule.

Hybrids

Starting from ADF2009.01 Hartree-Fock and the hybrid potentials can used in combination with NMR chemical shielding calculations. see Refs. [*422* (page 931), *423* (page 931)]. Use SAVE TAPE10 in the ADF calculation. The use of frozen cores and hybrids gives gauge dependent results for the NMR chemical shieldings, therefore the NMR program will stop in this case.

Meta-GGA's and meta-hybrids

Meta-GGA's and meta-hybrids should not be used in combination with NMR chemical shielding calculations. The results are wrong due to an incorrect inclusion of GIAO terms.

Input options

The input file for NMR uses the block key NMR, with several (optional) subkeys, each having a series of options. For analysis a separate block key can be used.

```
$ADFBIN/nmr << eor
ZSOAO2007
RECALCULATETAPE10
NMR
End
Analysis
End
eor
```

ZSOAO2007 keyword

In the ADF2008.01 a bug was introduced in the spin-orbit part of the calculated chemical shielding, which caused the calculated chemical shielding to be gauge dependent. This bug is relevant for spin-orbit coupled calculations for

ADF versions ADF2008-ADF2013. Workaround in ADF versions ADF2008-ADF2013 is to include the keyword ZSOAO2007 in the NMR part of the input, which causes a one-center approximation to be used. The bug has been fixed in ADF2014 by introducing an extra gauge-correction term for the spin-orbit coupled part. One can still get the (slightly) incorrect results in ADF2014 by using the keyword WRONGSOGAUGE, and not including ZSOAO2007.

REACALCULATETAPE10 keyword

If there is no TAPE10 present the NMR program will stop. One can use the key REACALCULATETAPE10 such that TAPE10 will be recalculated by the NMR module. Not recommended to be used. Better use 'SAVE TAPE10' in the ADF calculation, and use this TAPE10 as input for NMR.

NMR block key

```
NMR
  Out  OutOptions
  Calc CalcOptions
  Use UseOptions
  U1K  U1KOptions
  Nuc  NucOptions
  Atoms  AtomsOptions
  Ghosts  GhostsOptions
  Analysis  AnalysisOptions
End
```

Out OutOptions The subkey Out controls printed output. Its options specify the details by their (optional) presence. The following OutOptions are recognized (Default ISO TENS):

  All Implies all the other options

  ISO Isotropic shielding constants

  Tens Shielding tensors

  Eig Eigenvectors

  U1 The U1 matrix

  F1 The first order change in the Fock matrix

  S1 The first order change in the Overlap matrix

  AOP The paramagnetic AO matrix (= the matrix in the representation of elementary atomic basis functions)

  AOD The diamagnetic AO matrix

  AOF The Fermi-contact AO matrix

  REFS Literature references

  INFO General information

Calc CalcOptions The subkey Calc controls what is actually calculated. The following options are available (Default ALL):

  All Implies all of the other options to this key

  Para The paramagnetic part

  Dia The diamagnetic part

  FC The Fermi-contact part in case of the Pauli Hamiltonian

  SO The Fermi-contact part in case of the ZORA Hamiltonian

Use UseOptions The subkey Use controls some optional options (default none)

FXC Improves the exchange-correlation kernel used, as was implemented by J. Autschbach [*427* (page 932)]. Important only in case of spin-orbit coupled calculations. This may give some (small) gauge dependent results when using this. Important option that should be seriously considered and has been advocated in Ref. [*427* (page 932)].

SCALED Implies the scaled ZORA method, which gives (slightly) gauge dependent results, as was shown in Ref. [*427* (page 932)]. Note that in case of the ZORA Hamiltonian default the unscaled ZORA method is used. For chemical shifts, only compare results with the same options.

SO1C Before ADF2008.01 in the the spin-orbit term a 1-center approximation was used, which does not suffer from gauge dependence. This 1-center approximation can be used with USE SO1C.

U1K U1KOptions The subkey U1K determines which terms are included in the calculation of the U1 matrix (first order changes in MO coefficients). Options (Default Best):

Best The best (recommended) options for each relativistic option are included for this subkey. Implies 'None' for non-relativistic and scalar relativistic ZORA, 'SO' + 'SOFULL' for spin-orbit coupled ZORA, and 'MV + 'Dar' for the Pauli Hamiltonian.

None Implies none of the other options to this key.

All Implies all the other options to this key.

MV The mass-velocity term .

Dar The Darwin term.

ZMAN The Spin-Zeeman term. Can be included only in case of spin-orbit coupled Pauli Hamiltonian.

SO ZORA spin-orbit part.

SOFULL ZORA spin-orbit part.

Note: for chemical shifts, only compare results with the same options. If the subkey U1K is used with the option ALL in the ZORA calculation, then the scaled ZORA orbital energies are used in the making of the U1 matrix, which is not recommended. Recommended is to use 'U1K Best' in all cases, which uses plain ZORA orbital energies in the making of the U1 matrix.

NUC NucOptions This subkey NUC specifies for which nuclei the NMR shielding is calculated. Default all nuclei are calculated, i.e. as for omitting the subkeys ATOMS and NUC. Else you may use this options by simply typing Nuc in the NMR block (without any further data); this means: for no nuclei at all. Alternatively you may type the index of the atom(s) you want to see analyzed. Default all nuclei are calculated, i.e. as for omitting this subkey.

Example:

```
NUC 2 1
```

The numbers refer to the internal numbering of the nuclei as it appears somewhere early in the general ADF output. This internal numbering is also the internal NMR numbering, but it is not necessarily the same as the input ordering. Use the subkey ATOMS to specify the nuclei according to this input ordering in the ADF calculation.

Note that the number of nuclei has a significant consequence for the total CPU time.

Atoms AtomsOptions This subkey ATOMS specifies for which nuclei the NMR shielding is calculated. Default all nuclei are calculated, i.e. as for omitting the subkeys ATOMS and NUC.

Example:

```
ATOMS 2 1
```

The numbers refer to the input ordering in the ADF calculation. Use the subkey NUC to specify the nuclei according to the internal NMR numbers of the atoms.

GHOSTS The subkey GHOSTS is a block type subkey. The format is:

```
Ghosts
   xx1 yy1 zz1
   xx2 yy2 zz2
   ...
SubEnd
```

With this key, the user can specify ANY point(s) within the molecule at which the shielding is to be calculated (whatever the physical meaning of this shielding is). One can think of those points as neutrons within the molecule. There is a publication by P. Schleyer et al. using a similar feature (J. Am. Chem. Soc. **118**, 6317, 1996). They call it NICS, Nucleus-Independent Chemical Shift. Note that the NICS value is minus 1 times the isotropic part of the shielding tensor that is calculated at these points:

```
xx1 yy1 zz1
```

real numbers that specify the Cartesian coordinates of 'ghost' 1, etc.

The coordinates have to be specified in the same units as any other input (ADF subkey Units). That is, you use Angstrom for the ghosts if you did so for the atomic coordinates, or bohr otherwise. The same set of coordinates has to be specified as 'point charges with charge zero' using the key EFIELD. This is necessary in order to allow the appropriate distribution of integration points around the ghosts.

E.g., if you want to have two 'ghosts' with the coordinates xx1 yy1 zz1 and xx2 yy2 zz2 then you **must** also have in the input the key EFIELD as follows:

```
EFIELD
   xx1 yy1 zz1    0.0
   xx2 yy2 zz2    0.0
END
```

(the last number is the charge at these coordinates - zero).

Eventually, this step should be programmed internally but for now the procedure outlined above works. No check is done to verify whether those 'point charges' are taken care of or not, but their omission leads to unpredictable results.

Only Cartesian coordinates are possible for ghosts, even if the atoms were originally specified using internal coordinates. This shouldn't be a problem, though (e.g., one could start an ADF run of the molecule of interest, and get very soon the Cartesian coordinates of the atoms in the output. This run would then be aborted, and restarted with the ghosts specified as desired.) The ghosts are numbered in the output as NNUC+1, NNUC+2 ... where NNUC is the total number of nuclei in this molecule. Default: no ghosts.

Analysis AnalysisOptions The subkey Analysis controls the MO analysis. After the word (subkey) Analysis you type an integer, which then specifies that the first so many MOs are to be analyzed. Default no Analysis. The value of this analysis subkey in the block key NMR is somewhat limited. The separate ANALYSIS block key can give more analysis of the NMR chemical shielding.

Analysis block key

The NMR shielding tensor, can be analyzed in detail, see Refs. [*329-331* (page 926)]. For the analysis option with the ANALYSIS block key there are some restrictions. In the ADF calculation all electron basis sets should have been used, and SYMMETRY NOSYM. Can not be used in case of non-relativistic calculations. The ADF calculation should use relativistic scalar ZORA or relativistic spinorbit ZORA. In case of scalar relativistic ZORA the keyword FAKESO should be added to the NMR input (outside of the NMR or Analysis block keys). The analysis utilizes the ZORA spin-orbit branch of the NMR code. For scalar ZORA, the NMR analysis contributions will appear in equivalent pairs

for spin-orbitals even if the ADF calculation is closed-shell spin restricted. The MO numbering then also reflects this doubling of MOs. In the analysis, canonical MOs number 1 and 2 are the alpha and beta spin ADF MO 1, canonical MOs number 3 and 4 correspond to the alpha and beta spin ADF MO 2, and so on. In case of spinorbit relativistic ZORA the keyword FAKESO should not be included. For an NBO analysis of NMR, see the *section on NBO analysis* (page 243).

```
Analysis
  print threshold
  canonical
  {components}
End
{FakeSO}
```

print threshold The print keyword selects printout of contributions relative to the total diamagnetic, paramagnetic. For example in case of 'print 0.01' only contributions greater than 1% are printed. Set to zero to print ALL contributions.

canonical It enables an analysis of the shielding in terms of the canonical MOs.

components The components keyword is optional and enables an analysis not only of the isotropic shielding but also of the diagonal Cartesian components of the tensor XX, YY, and ZZ). In order to analyze the principal shielding tensor components with canonical MOs you can calculate the shielding tensor first with the NMR code, rotate the molecule such that the principal axes system aligns with the Cartesian coordinate system, and then repeat the NMR calculation with the analysis features switched on.

6.8.2 Paramagnetic NMR Chemical Shifts

Knowledge of the g-tensor and hyperfine A-tensor can be used in the prediction and analysis of paramagnetic NMR shifts, see Refs. [*340-342* (page 927)]. Because of the dependence on the g-tensor, prediction of paramagnetic NMR (pNMR) shifts is not straightforward, as the dependence of the pNMR shift on excess α or β electron spin density at a nucleus is to be combined with the sign and magnitude of the isotropic g-value. Ref. [*340* (page 927)] describes in detail how to calculate pNMR contact chemical shifts and pNMR pseudo-contact chemical shifts, using the ADF program.

Of course, like for NMR chemical shielding of closed shell molecules, one also needs to calculate of the so called orbital dependent part of the NMR chemical shielding of the open shell molecule. For open shell molecules in ADF this can only be calculated without taken into account spin-orbit coupling:

```
$ADFBIN/adf << eor
CHARGE charge spinpolarization
unrestricted
Relativistic scalar ZORA
Basis
   Core None
End
SAVE TAPE10
eor
...
$ADFBIN/nmr << eor
ALLINONE
nmr
   u1k best
   calc all
   out iso tens
end
end input
eor
```

Note that one can only do this at the scalar relativistic ZORA level, one needs to use all electron basis sets, and one needs to include the key ALLINONE in the input for NMR,

6.8.3 NMR spin-spin coupling constants

The NMR spin-spin coupling constants [*118, 119* (page 913)] have been implemented in a separate program CPL. It can be combined with ZORA and Spin-Orbit treatment of relativistic effects to study heavy elements. The original version of this part of the User's Guide was written by Jochen Autschbach, primary author of the CPL code.

Two GUI tutorials on spin-spin coupling constant calculations are currently available:

- Chemical shifts and nuclear spin-spin coupling constants w. NBO analysis: Analysis of NMR parameters with Localized Molecular Orbitals

- Chemical shifts and nuclear spin-spin coupling constants w. visualization of the spectrum in the GUI: H-NMR spectrum with spin-spin coupling

Introduction

The CPL code of the Amsterdam Density Functional program system allows the user to calculate *Nuclear Spin-spin Coupling Constants* (NSSCCs) [*118* (page 913), *119* (page 913)]. NSSCCs are usually observed in NMR (Nuclear Magnetic Resonance) spectroscopy and give rise to the splitting of the signals of the NMR spectrum in multiplets. They contain a wealth of information about the geometric and electronic structure of the compound being investigated.

The calculation needs a standard TAPE21 ADF output file. CPL reads also an input key and optional settings from stdin (usually from an input file). Technical parameters such as the maximum memory usage can be set here as well.

One of the key features of the program is its ability to treat heavy nuclei with the ZORA relativistic formalism. We refer the reader to the literature for details about our implementation [*118* (page 913), *119* (page 913)], and the general review on relativistic computations of NMR parameters [*331* (page 926)]. Please use the information printed in the output header of the CPL program in order to provide references of this work in scientific publications.

The development of the CPL program started in 2000. CPL provides the main functionality in order to evaluate NSSCCs based on DFT, as well as a number of additional features in order to provide an analysis of the results. Several analysis features for the coupling constant have been added, see the CONTRIBUTIONS sub key. Please report bugs or suggestions to SCM at support@scm.com.

Theoretical and technical aspects

Within the non-relativistic theory of nuclear spin-spin coupling, there are four terms contributing to the NSSCC between two nuclei A and B: the paramagnetic and diamagnetic orbital terms (OP and OD, respectively), and the electron-spin dependent Fermi-contact (FC) and spin-dipole term (SD). In the literature, the OP and OD terms are often named PSO and DSO (for paramagnetic and diamagnetic spin-orbital). In the more general ZORA formulation, very similar operators are responsible for the NSSCC, therefore we use the same terminology for the individual contributions. In general, the interpretation of the results for a heavy atom system is basically equivalent to a non-relativistic situation.

In most cases, the FC term yields the most important contribution to the NSSCC. However, many exceptions are known for which one or each of the other terms can be non-negligible or even dominant. We therefore suggest that you always check, at least for a smaller but similar model system, or by using a smaller basis set, which of the four terms are negligible and which are dominant.

By default, the CPL program computes the FC coupling between the first and all other nuclei of the molecule, respectively. Other couplings or the computation of the OP, OD and SD terms can be requested by input switches (see the 'Running CPL' section of this document for details).

All contributions to the NSSCC are evaluated with the help of the numerical integration scheme implemented into ADF. In general, the computation of the OD term is computationally very cheap, since only integrals involving the

electron density have to be evaluated. The next expensive term is the OP term. For this contribution, the first-order perturbed MOs have to be computed. With the available density functionals in ADF, the OP term does not cause a change in the Kohn-Sham potential, and the first-order MOs can be computed directly (i.e. without an iterative procedure). This is equivalent to the approach that has been implemented in the NMR code for ADF.

Both the FC and the SD terms induce electron spin-density to first-order as a perturbation. Equivalent to the iterative solution of the unperturbed Kohn-Sham equations, the first-order MOs depend on that first-order spin-density, which in turn depends on the first-order MOs. Therefore, in order to evaluate the FC and SD NSSCC contributions, the CPL program carries out a SCF cycle. In the scalar or non-relativistic case, the computational cost for the FC term is comparable to an ADF single point calculation with a local density functional. The evaluation of the SD term is more expensive. The current implementation utilizes the CPL spin-orbit code to compute the combined FC+SD contribution and therefore leaves some room for future speed-ups. In most cases, the SD term yields a negligible NSSCC and the much faster code for the scalar- or non-relativistic FC term can be used. However, it is very important to include the SD term in the computation if coupling anisotropies are to be evaluated.

In the case where the NSSCC computation is based on spin-orbit coupled relativistic two component ZORA MOs, the SD term causes only a marginal increase in computational time as compared to the FC term alone. Generally, in this case the computational cost for the FC term is already approximately one order of magnitude higher than in the scalar or non-relativistic case, since the 3 (x, y, z) components of the spin-density with respect to 3 components of the perturbation, respectively, have to be determined self-consistently. The additional presence of the SD term only shows up in a somewhat more costly evaluation of the matrix elements of the perturbation operator. However, CPL spends most of its computational time in the SCF cycle. Therefore, in spin-orbit computations the computation of the FC+SD terms is the default. The OP term has to be evaluated self-consistently, too, in this case and is added as a perturbation in the SCF cycle upon request.

We use the terminology 'perturbing' and 'responding nucleus' within the CPL output. The 'perturbing' nucleus is the one, for which the first-order MOs have to be computed (self-consistently), while the NSSCC is then determined by these first-order MOs and the FC, SD, and OP matrix elements of the second, 'responding' nucleus. For the OD term, this distinction makes no sense but is used in the output for reasons of consistency.

Experimental NSSCCs between two nuclei A and B are usually reported as J(A,B) in Hertz. From a computational point of view, the so-called reduced NSSCCs K(A,B) are more convenient for comparisons. CPL outputs both. The J's are set to zero in case the nuclear magneto-gyric ratio of one of the nuclei A or B is not available at run time.

Further technical aspects and current limitations

In order to facilitate the future computation of rather large molecules, all matrix elements of the perturbation operators FC, SD, and OP are evaluated in the Slater AO basis that is specified as input in the CREATE runs of ADF. The AO matrix elements are further transformed to the basis of MOs and the calculation proceeds within the MO basis. This allows for a convenient analysis of the results in terms of contributions from individual occupied and virtual MOs. Such an analysis can be requested by input.

The matrix elements themselves as well as the first-order contributions to the potential are evaluated by numerical integration. The CPL code, which is parallelized, can use multiple processors for these steps of the computation. The accuracy setting for the numerical integration is of high importance to obtain accurate matrix elements. Furthermore, the basis set being employed needs to be flexible enough to describe the perturbation correctly. This means usually that modified basis sets have to be used in particular for heavy element calculations.

The first-order potential is currently approximated by the VWN functional. The X:math:$alpha$ potential is available as an alternative but usually leads to less accurate results. In ADF2009.01 the first order potential of the PBE family of GGA functionals and the hybrid PBE0 functional can be used.

Currently, only spin-restricted computations for systems with an even number of electrons are supported. Further, the calculation does not make use of symmetry and must be based on an ADF run with input SYMMETRY=NOSYM. Non-Aufbau configurations are not supported. The atom input list must not contain dummy atoms.

With the present version of CPL, the SD term and the FC/SD cross term cannot be evaluated separately. Either, the sum of FC + SD + cross terms, or the FC term individually, are computed.

CPL is restartable after various time-consuming steps of the computation.

In ADF2009.01 the hybrid PBE0 functional can be used in combination with NMR spin-spin coupling calculations, see the documentation for the extra keys that are needed. However, other hybrid functionals and Hartree-Fock can not or should not be used in combination with NMR spin-spin coupling calculations.

In ADF2009.01 the effects of a finite size of a nucleus on the spin-spin couplings can be calculated. A finite size of the nucleus can be set with the NUCLEARMODEL key in the input for the ADF calculation.

Bug fix in case more than 1 perturbing atom and DSO or PSO

In the ADF2006.01b version a bug in the CPL module is fixed that gave problems in ADF2006.01 and older versions. The problem in ADF2006.01 and older versions is: In case there is more than 1 perturbing atom and the DSO or PSO term is calculated, only the results of the spin-spin couplings for the first perturbing atom are correct, but the results of the other spin-spin couplings may be incorrect.

Input file for CPL: TAPE21

In order to run the CPL code, you need the general ADF output file TAPE21 being present in the directory where CPL is running. Most of the computation's specific settings will be taken from TAPE21, such as the integration accuracy, the basis set, the density functional being employed, nuclear coordinates, and so on. *That also means that nearly all of the aspects that affect the quality of CPL's results are already determined in the input for the ADF run*. Five aspects are of particular importance here:

1. The numerical integration accuracy: the perturbation operators are large in the vicinity of the nuclei. Therefore, you have to make sure that the integration grid is fine enough in the atomic core regions.

2. The basis set: NSSCCs are sensitive chemical probes, and therefore flexible basis sets have to be employed in order to yield a valid description of the MOs that determine the NSSCCs. We have found that it is imperative to use at least a TZ2P basis set. Additional polarization functions in the valence shell may be necessary. Furthermore, the FC perturbation usually requires additional steep 1s functions (i.e. with exponents much higher than the nuclear charge) for a proper description. In the relativistic heavy element case, the use of additional steep basis functions as compared to the ZORA/TZ2P basis is mandatory. The use of steep functions is only of high importance for those nuclei, for which the NSSCC is to be evaluated. In ADF2017 specialized large all electron basis sets have been added in the directory $ADFHOME/atomicdata/ZORA/TZ2P-J and $ADFHOME/atomicdata/ZORA/QZ4P-J, suitable for NSSCCs, where steep basis functions have been added, which is especially important for the FC term. Previously in ADF2009.01 for a limited number of elements basis sets suitable for NSSCCs have been added to the ADF basis set directory, in the directory $ADFHOME/atomicdata/ZORA/jcpl. These specialized basis sets are especially important for the heavy NMR nuclei. For the nuclei for which NSSCCs are to be evaluated, it is necessary to use all-electron basis sets. This is not a restriction due to the implementation, but we have found that, with the available frozen core basis sets, the flexibility of the basis in the vicinity of the nuclei is not sufficient. It is possible to use frozen core basis sets if you add enough basis functions in the core region such that the basis approaches the flexibility of at least a double-zeta all-electron basis there [*118* (page 913)]. In that sense, the savings in computational time due to usage of a frozen core basis are not as pronounced as in standard ADF computations. Unless reliable frozen-core basis sets for the NSSCC computation are available we strongly discourage the use of frozen core basis sets with the CPL program!

3. The finite size of a nucleus: typically, the isotropic J-couplings are reduced in magnitude by about 10 to 15 % for couplings between one of the heaviest NMR nuclei and a light atomic ligand, and even more so for couplings between two heavy atoms, see Ref. [*270* (page 922)]. However, one should have really large basis sets with tight basis functions to observe this effect in calculations, see the previous point about basis sets. The TZ2P-J, QZ4P-J, and jcpl basis sets are suitable for finite nucleus calculations. A finite size of the molecule can be set in the ADF program with the key NUCLEARMODEL:

```
NuclearModel Gaussian
```

4. The density functional: the results of the CPL code depend mostly on the shape of the MOs that have been determined by ADF, and their orbital energies. Both, in turn, depend on the density functional or Kohn-Sham potential that has been chosen for the ADF run (and the basis set quality). It is difficult to give a general advice here concerning the NSSCCs. So far we have found that the use of GGAs improves the NSSCCs with respect to experiment in most cases in comparison to LDA. Different GGAs often yield very similar results. Further, in particular for those cases for which the OP term is large or even dominant, both standard LDAs and GGAs sometimes do not provide an accurate enough description of the orbitals, and deviation of the CPL results as compared to experiment can be substantial. Future developments of density functionals might be able to cure these problems. For the time being, we recommend that you base the CPL run on different choices of density functionals in the ADF run, and investigate the convergence of the result with respect to basis set and integration accuracy. Note that CPL itself uses the VWN functional by default to determine the first-order perturbed MOs. There are enough indications to believe that this is a reasonable approximation for NMR purposes. In ADF2009.01 the first order potential of the PBE family of GGA functionals and the first order potential of the hybrid PBE0 functional can be used. See Refs. [*425,426* (page 932)] for applications of such first order potentials. However, other hybrid functionals and Hartree-Fock can not or should not be used in combination with NMR spin-spin coupling calculations.

5. Modeling the experimental setup: computing such sensitive numbers as NMR chemical shifts and in particular NSSCCs can result in substantial deviations from experimental data. The simple reason might be that the isolated system that has been computed at zero temperature is not at all a good approximation to the system that has been studied experimentally. We [*403,404* (page 930)] and other authors have found that in particular solvent effects can contribute very substantially to the NSSCC. In case you are comparing CPL results to experimental data obtained in strongly coordinating solvents we suggest that you consider solvent effects as a major influence. We have found that even weakly coordinating solvents can cause sizable effects on the NSSCCs for coordinatively unsaturated metal complexes. Other sources of errors can be the neglect of vibrational corrections to the NSSCCs (usually in the range of a few percent).

If the parameters of the underlying ADF computation are carefully chosen and the density functional is able to provide an accurate description of the molecule under investigation, it is possible to compute NSSCCs by means of DFT with very satisfactory accuracy (please note that for properties as sensitive as NSSCCs, agreement with experimental results within about 10% error can be regarded as quite good). Further, chemical trends will be correctly reproduced for a related series of molecules in most cases. However, due to the inherent approximate character of the density functionals currently available with ADF, and necessary basis set limitations, great care should be taken that the results are reliable. *CPL assumes Aufbau configurations. Please make sure that there are no empty orbitals with energies below the highest occupied MO (HOMO). In addition, the SYMMETRY NOSYM key has to be used in the ADF computation. It is currently not possible to use dummy atoms in the ADF input if the*TAPE21 *is intended to be used for a subsequent CPL computation.*

Running CPL

Main input switches

With the ADF output TAPE21 present in the current working directory, the CPL code is invoked by:

```
$ADFBIN/cpl < input_file
```

where input_file contains the input for CPL. Best is supply also the TAPE10 of the ADF calculation in addition to TAPE21 (specify SAVE TAPE10 in the ADF input file).

input_file must contain at least one block-type input key in order to start the CPL run. The input key is

```
$ADFBIN/cpl << eor

NMRCOUPLING
END
eor
```

This represents a minimal input file for CPL. The NMRCOUPLING key hosts all optional keys that are relevant for the NSSCCs themselves. In addition to the mandatory NMRCOUPLING key, CPL recognizes the following input switches:

```
$ADFBIN/cpl << eor
GGA
...
eor
```

See the separate section for this key, which influences the first order potential that is used.

```
$ADFBIN/cpl << eor
RESTART /full/path/to/restart_file
...
eor
```

restart the computation from file restart_file (full pathname should be given). This is the TAPE13 produced during a CPL run. By default, TAPE13 is deleted after a successful completion of CPL. As with ADF restarts, you can not use the name TAPE13 for restart_file but you have to rename it, e.g. to tape13.restart.

```
$ADFBIN/cpl << eor
SAVE TAPE13
...
eor
```

keep the restart file even after a successful completion of CPL. TAPE13 is currently the only file that is meaningful as a parameter to SAVE.

NMRCOUPLING subkeys

The available switches within a NMRCOUPLING/END block control the computation of the NSSCCs. By default, the program will evaluate the FC coupling contribution for the first nucleus being the perturbing nucleus and all remaining nuclei responding.

Please **note** that the ordering of atoms in CPL is generally different from the ADF input. The ordering of atoms is the one being stored in TAPE21 and it is grouped by fragment types. In case you are in doubt about the ordering of atoms, you can run CPL for a few seconds. It will print a list of atoms with their coordinates. The ordering is currently the same as required the NMR program in the ADF program system. On the other hand, note that for the subkeys ATOMPERT and ATOMRESP the number of the atoms refer to the input ordering in the ADF calculation.

Available subkeys are:

```
$ADFBIN/cpl << eor
 NMRCOUPLING
   NUCLEI {npert nresp1 nresp2}
   ATOMPERT {npert1 npert2 npert3}
   ATOMRESP {nresp1 nresp2 nresp3}
   GAMMA {nnuc gamma}
   DSO
   PSO
   SD
   FC
   SCF {ITERATIONS=25 | NOCYCLE | CONVERGE=1e-4 }
   XALPHA
   CONTRIBUTIONS {1E19} {LMO, SFO, LMO2, SFO2}
END
...
eor
```

NUCLEI {npert nresp1 nresp2} Use nucleus no. npert as the perturbing nucleus, and nuclei nresp1, nresp2, etc as responding nuclei. You can supply more than one NUCLEI keys, in which case CPL evaluates the first-order MOs for each perturbing nucleus that is specified and computes the NSSCCs between all specified responding nuclei. For each NUCLEI line in the input, CPL has to perform an SCF cycle. Note: for the numbers of the atoms the internal CPL numbering should be used.

ATOMPERT {npert1 npert2 npert3} See next subkey ATOMRESP.

ATOMRESP {nresp1 nresp2 nresp3} ATOMPERT: use nucleus no. npert1, npert2, etc. as the perturbing nuclei. ATOMRESP: use nucleus no. nresp1, nresp2, etc. as the responding nuclei. You can supply more than one ATOMPERT and (or) ATOMRESP key. CPL computes the NSSCCs for all pairs of combinations of perturbing atoms and responding atoms. For each perturbing atom CPL has to perform an SCF cycle, which is the expensive part in the calculation. Note: the numbers refer to the input ordering in the ADF calculation. Use the subkey NUCLEI to specify the nuclei according to the internal CPL numbers of the atoms.

GAMMA {nnuc gamma} Input a non-default magneto-gyric ratio of g = gamma for nucleus no. nnuc, in units of rad/(T s). Note that one should include the the typical 10^7 factor. CPL normally uses the g value of the most abundant NMR active isotope for a nucleus of a given charge by default. With the GAMMA keyword you can override this value or supply a value if CPL does not know about it. A list of g's that is used in the computation is printed in the output. You have to provide the GAMMA key for each nucleus you want to specify.

DSO Compute the diamagnetic orbital term for each NSSCC that is requested (not default)

PSO Compute the paramagnetic orbital term for each NSSCC (not default)

SD Compute the SD term for each NSSCC. This is only default for spin-orbit ADF runs. The output will contain the sum of the FC and SD contributions. *Please note that requesting this option results in a greatly increased computational cost in scalar or non-relativistic runs.* The option NOSD will turn the SD computation off in spin-orbit runs and has no effect otherwise.

FC Compute the FC contribution to the NSSCCs. This is the default option. Please note that it is currently not possible to compute the SD term without the FC term. Consult the 'practical aspects' section for instructions how to estimate the FC/SD cross term. The option NOFC will disable both the FC and SD computation.

SCF {ITERATIONS=25 | NOCYCLE | CONVERGE=1e-4 } Settings related to the SCF cycle that is carried out by CPL. Valid options are (with default values if applicable):

ITERATIONS 25 maximum number of iterations

NOCYCLE perform no cycle, equivalent to ITERATIONS 0

CONVERGE 1e-4 convergence criterion, an input of e corresponds approximately to a convergence of log(-e) digits, i.e. the results will be converged to about four significant digits by default. The measurement for the convergence is based on the sum S of the magnitudes of all occupied-virtual matrix elements of the induced first-order exchange potential. Note that the actual convergence criterion being used in the computation is e times S of the first cycle, i.e. the convergence criterion is set relative to the initial value of S.

XALPHA Use first-order Xalpha potential instead of VWN potential (default). This will usually decrease the accuracy for couplings involving hydrogen, and does not have a large effect for couplings between heavier nuclei (not default). The key is mainly intended to ensure compatibility with our previously published results.

CONTRIBUTIONS {1e19} {LMO, SFO, LMO2, SFO2} Print contributions from individual orbitals to the FC and OP term of the NSSCCs that are larger in magnitude than a certain threshold. The threshold refers to the reduced coupling constant K in SI units (not default). Additionally, an analysis in terms of Boys localized MOs (see User's Guide and SFOs. At present, either each key LMO, SFO, LMO2, SFO2 can be used individually, or grouped as {LMO, SFO2} or {SFO2, LMO}. If you need all analyses or different combinations, it is recommended to restart the CPL calculation from TAPE13, and to specify 0 iterations in the SCF. This way, the only additional computational cost should be the analysis itself.

The equation and an application for the analyses due to the LMO and SFO keys is described in Refs. [*368,369* (page 928)]. The other analysis is based on the same equation as in Ref. [*370* (page 928)]. For an NBO analysis

of the spin-spin couplings, see the *section on NBO analysis* (page 243).

In order for the LMO-based analyses to work, the MO $\rightarrow$ LMO transformation matrix needs to be stored on TAPE21. In the ADF input, you can achieve this with the option "STORE" to the LOCORB key, i.e.

```
LOCORB STORE
    ... options
END
```

GGA key

```
$ADFBIN/cpl << eor

GGA
...
eor
```

GGA Use first-order GGA potential instead of the first-order VWN potential. Should only be used for the PBE family of GGA exchange-correlation functionals and for the hybrid functional PBE0. See Refs. [*425* (page 932), *426* (page 932)] for applications of calculating spin-spin couplings with PBE0. However, other hybrid functionals and Hartree-Fock can not or should not be used in combination with this key GGA. For consistency reasons of the first-order potential one should use the keyword USESPCODE in the ADF calculation. An example input for ADF for the hybrid PBE0 would then contain:

```
USESPCODE
XC
 hybrid PBE0
End
```

Practical Aspects

Minimal input

The default settings for CPL are invoked by the simple minimal content of the input file:

```
$ADFBIN/cpl << eor
NMRCOUPLING
END
eor
```

This is equivalent to

```
$ADFBIN/cpl << eor
NMRCOUPLING
 NUCLEI 1 2 3 4 5 6 7 8 ..(up to number of atoms)
 SCF CONVERGE 1e-4 ITERATIONS 25
 FC
END
eor
```

Restarts

CPL is restartable after the computation of each the complete set of FC or FC/SD and OP matrix elements, and after their transformation to the MO basis. Further, in spin-orbit runs or in scalar- or non-relativistic computations involving the SD term, CPL is restartable after each SCF cycle. As with ADF restarts, you need to supply a proper input file for a restarted computation, and the restart file TAPE13 (which needs to be renamed). Changing the input of a calculation for a restart is not supported. In restarted runs, the program will automatically continue at the latest possible point

before the execution stopped, and changing the input between restarts can cause inconsistencies that may lead to a crash.

Unless you are computing a very large molecule, the most likely need for a restart will probably occur during a computation of the FC/SD SCF cycle. We have already mentioned that this is a very time consuming part of the computation, and for this reason CPL can be restarted after each completed SCF cycle. The convergence of the results should not be affected by a restart. You can, e.g., use this in order to complete a lengthy CPL computation in case you have tight time limits in your queuing system, or after a power loss.

How to avoid the unnecessary computation of many SCF cycles

As already mentioned, once the first-order MOs with respect to the perturbation by one of the nuclear spins have been determined, the NSSCC between this and all other nuclei can be computed rather quickly. For each nucleus that participates in at least one of the coupling constants to be determined, the matrix elements of the FC, SD, and OP operators have to be evaluated once (unless the computation of the respective terms is disabled).

You can use this information in order to minimize the number of nuclei for which an SCF cycle has to be performed. This can lead to a great speedup of the computation. The final result, the NSSCC between A and B, does not depend on which nucleus has been chosen as the 'perturbing' one, and which as the 'responding' one (convergence has to be good enough, though). Suppose you want to compute the NSSCCs in the water molecule, with O being nucleus no. 1. In that case,

```
NUCLEI 1 2 3
NUCLEI 2 3
```

yields the same O-H and H-H coupling constants as the input

```
NUCLEI 2 1
NUCLEI 1 3
NUCLEI 3 2
```

but with less computational effort due to the fact that only 2 instead of 3 SCF cycles will be performed. The example chosen here is trivial, but in other cases it can be worthwhile to consider different sequences of computations.

Alternatively you can use the ATOMPERT and ATOMRESP subkeys:

```
ATOMPERT 1 2
ATOMRESP 2 3
```

which will calculate the spin-spin coupling of the nuclei 1-2, 1-3, and 2-3 (skips 2-2, since the nuclei are the same), which are the same O-H and H-H couplings as before.

Note: The numbers of the nuclei for the subkeys ATOMPERT and ATOMRESP refer to the input ordering in the ADF calculation, whereas the numbers of the nuclei for the subkey NUCLEI refer to the internal CPL numbers of the atoms.

Computing individual terms in the coupling tensor

As we have mentioned before, the FC, OP and OD terms can be calculated individually, but not the SD term. In case the SD input option is given, the FC+SD contribution is evaluated instead. This is NOT equal to the sum of the individual FC and SD contributions since there is a cross term between these two. Due to computational simplicity and efficiency, CPL evaluates either the matrix elements for the FC operator, or the combined ones for FC+SD. The final result therefore contains either FC only, or FC, SD plus the cross terms. Only the latter, in addition to the OP and OD contributions, should be compared to experimental results. We will implement the computation of the individual SD term in a future version of CPL in order to assist the analysis of the CPL results.

Likewise, in a spin-orbit based relativistic computation, there exists a cross term between the spin-dependent FC and SD terms, and the OP term. In the scalar- or non-relativistic limit, this contribution is always zero. With the PSO option present, CPL computes the FC, SD and OP terms including all cross contributions. Even though the output suggests that the individual OP and FC+SD terms are printed, they contain additional cross terms if spin-orbit coupling is large. You can run CPL with the options

```
$ADFBIN/cpl << eor
NMRCOUPLING
 NOFC
 NOSD
 PSO
END
...
eor
```

in order to evaluate the individual OP contribution(s). In a second run, you can then compute just the FC+SD contributions. The differences between these two CPL runs and a third one with all three terms present yields the relativistic (FC+SD)/OP cross term.

Two-bond and more-bond couplings

CPL does not discriminate between one-bond and two-bond couplings etc. in any technical sense. Even though we [*118,119* (page 913), *403,404* (page 930)] have validated the code mostly for one-bond NSSCCs, the coupling between any pair of nuclei in the molecule can be computed. See Ref. [*404* (page 930)] for an example.

Principal axis system, the whole coupling tensor

CPL evaluates the complete 3x3 coupling tensor with respect to the Cartesian input coordinate system. Depending on the orientation of the molecule, and the local symmetry, the coupling tensor has in fact often only a small number of independent components. CPL evaluates the 'principal components' by the following procedure: the 3x3 matrix is transformed into the basis of the eigenvectors of its symmetric part. This diagonalizes the symmetric part of the coupling tensor. A set of eigenvectors (= 'principal axis system') is also printed.

References

- [*118* (page 913)] J. Autschbach, T. Ziegler, J. Chem. Phys. 2000, **113**, 936.

- [*119* (page 913)] J. Autschbach, T. Ziegler, J. Chem. Phys. 2000, **113**, 9410.

- [*403* (page 930)] J. Autschbach, T. Ziegler, J. Am Chem. Soc. 2001, **123**, 3341.

- [*404* (page 930)] J. Autschbach, T. Ziegler, J. Am Chem. Soc. 2001, **123**, 5320.

- [*270* (page 922)] J. Autschbach, ChemPhysChem, 2009, **10**, 2274.

- [*425* (page 932)] J. Autschbach, J. Chem. Phys. 2008, **129**, 094105, J. Chem. Phys. 2009, **130**, 209901.

- [*426* (page 932)] D.L. Bryce and J. Autschbach, Can. J. Chem. 2009, **87**, 927.

- [*331* (page 926)] J. Autschbach and S. Zheng, Ann. Rep. NMR Spectr. 2009, **67**, 1.

See also:

- [*371* (page 928)] N. F. Ramsey, Phys. Rev. **91**, 303 (1953).

- [*372* (page 928)] Dickson, R.M.; Ziegler, T. J. Phys. Chem. 1996, **100**, 5286.

- [*370* (page 928)] Khandogin, J.; Ziegler, T. Spectr Acta A 1999, **55**, 607.

- [*373* (page 929)] D. L. Bryce, R. Wasylishen, J. Am. Chem. Soc. **122**, 3197 (2000). this ADF User's manual, SCM, Vrije Universiteit, Amsterdam, The Netherlands.

- [*374* (page 929)] G. Schreckenbach, S. K. Wolff, T. Ziegler, Modeling NMR chemical shifts, ACS Symposium Series, Washington DC (1999).

Worked out examples on chemical shift- and spin-spin-coupling calculations can be found in the **tutorials section**:

- Chemical shifts and nuclear spin-spin coupling constants w. NBO analysis: Analysis of NMR parameters with Localized Molecular Orbitals

- Chemical shifts and nuclear spin-spin coupling constants w. visualization of the spectrum in the GUI: H-NMR spectrum with spin-spin coupling

6.9 ESR/EPR

The EPR (ESR) g-tensor, hyperfine interaction (A-tensor), nuclear quadrupole interaction (Q-tensor), and zero-field splitting (ZFS, D-tensor) can be calculated. Effects due to spin-orbit coupling can be included. All electron basis sets can be used.

The separate program EPR/NMR ($ADFBIN/epr) program is no longer documented, since most of its capabilities are implemented in newer modules. See for the documentation of the EPR/NMR module in the old ADF 2010 Properties document (http://www.scm.com/Doc/Doc2010/ADF/Properties/Properties.pdf), and Refs. [*120,121* (page 914), *416,417* (page 931)].

6.9.1 ESR/EPR g-tensor and A-tensor

A-tensor, no spin-orbit coupling

```
$ADFBIN/adf << eor
  ESR
  END
  CHARGE charge spinpolarization
  unrestricted
  {Relativistic scalar ZORA}
  {NUCLEARMODEL gaussian}
eor
```

If spin-orbit coupling is neglected, the spin in the effective spin Hamiltonian, which is commonly used for the interpretation of ESR experiments, is the real electronic spin of the paramagnetic molecule. In the spin-unrestricted DFT calculations one then uses eigenfunctions of S_z. The A-tensor can then simply be calculated as expectation value of the corresponding operator, see [*96* (page 912)].

The A-tensor will be calculated for all nuclei. Terms due to the spin-polarization density at the nucleus are included in the evaluation of the A-tensor. For an accurate evaluation of the spin-polarization density at the nucleus it is important to use an all-electron basis set for the nuclei that one is interested in, avoiding the frozen core approximation. For heavy elements the incorporation of a Gaussian finite nucleus model can be important. However, one should have really large basis sets with tight basis functions to observe this effect in calculations. One possibility is to use the $ADFHOME/atomicdata/ZORA/QZ4P basis set, although even this large basis set is not large enough sometimes. The basis sets in the directories ZORA/TZ2P-J, ZORA/Z4P-J, and ZORA/jcpl are suitable for finite nucleus calculations.

In case one uses a finite nuclear model for the charge distribution, starting from ADF2013 ADF also uses a finite distribution of the nuclear magnetic dipole moment for the calculation of the A-tensor. **A-tensor, perturbative inclusion spin-orbit coupling**

```
$ADFBIN/adf << eor
CHARGE charge spinpolarization
unrestricted
Relativistic scalar ZORA
Symmetry NOSYM
...
eor

$ADFBIN/cpl << eor

hyperfine
```

```
 atoms 1 2 :: calculates A-tensor for atom 1 and 2, input order
 SCF Converge=1e-7 {Iterations=25}
end
...
eor
```

The calculation of A-tensors is implemented in the CPL program as a second derivative property (spin-orbit coupling and nuclear magnetic field as perturbation) within the two-component relativistic zeroth-order regular approximation (ZORA), see Ref. [*340* (page 927)]. This implementation allows for hybrid (only PBE0) DFT calculations, but not metaGGA's and not metahybrids.

Note that the CPKS convergence in CPL has to be set tightly (1e-7 or 1e-8) to get converged PSOSO terms for the A-tensor. For hyperfine calculations the default value is 1e-7.

g-tensor, perturbative inclusion spin-orbit coupling

```
$ADFBIN/adf << eor
CHARGE charge spinpolarization
unrestricted
Relativistic scalar ZORA
Symmetry NOSYM
...
eor

$ADFBIN/nmr << eor
nmr
 gfactors
 u1k best
 calc all
 out iso tens
end
end input
eor
```

The calculation of g-tensors is implemented in the NMR program as a second derivative property (spin-orbit coupling and external magnetic field as perturbation) within the two-component relativistic zeroth-order regular approximation (ZORA), see Ref. [*339* (page 926)]. This implementation allows for hybrid (B3LYP, PBE0, etc) DFT calculations, but not metaGGA's and not metahybrids. This implementation requires the use of all electron basis sets.

For an older implementation of this method, see the EPR/NMR module documentation in the old ADF 2010 Properties document (http://www.scm.com/Doc/Doc2010/ADF/Properties/Properties.pdf), and Refs. [*120,121* (page 914), *416,417* (page 931)].

g-tensor and A-tensor, self consistent spin-orbit coupling

```
$ADFBIN/adf << eor
ESR
END
CHARGE charge
unrestricted
Relativistic spinorbit ZORA
Collinear
Symmetry NOSYM
eor
```

In a spin-orbit coupled spin unrestricted relativistic ZORA calculation and the ESR block key, the g-tensor and the nuclear magnetic dipole hyperfine interaction (A-tensor) will be calculated, see also Refs. [*95,96* (page 912)]. In such a calculation degenerate perturbation theory is used with the external magnetic field or nuclear magnetic field as pertur-

bation. The calculation must use the collinear approximation, and symmetry must be NOSYM. This implementation does allow for metaGGA, and (meta-)hybrid DFT calculations, but then GIAO's are not used. There may be more than one unpaired electron. Terms due to the spin-polarization density at the nucleus are included in the evaluation of the A-tensor. However, one can not set the number of unpaired electrons, the 'spinpolarization' argument of the key CHARGE will be ignored.

Note: in a spin-orbit coupled spin restricted relativistic ZORA calculation and the ESR block key, ADF will also calculate and print the nuclear magnetic dipole hyperfine interaction, but the terms due to the spin-polarization density at the nucleus are absent. Furthermore, if there is more than one unpaired electron, the computed results will simply be incorrect, without any warning from the program. On the other hand, in case of one unpaired electron, and very large effects of spin-orbit coupling, the spin-restricted calculation may be of interest, since it uses Kramer's symmetry exact.

6.9.2 ESR/EPR Q-tensor

For the calculation of the ESR Q-tensor see the key *QTENS* (page 221).

6.9.3 ESR/EPR Zero-field splitting (D-tensor)

With the keyword ZFS the zero-field splitting (ZFS) of the ground state can be calculated.

```
ZFS
RELATIVISTIC SCALAR ZORA
{HARTREEFOCK}
```

Zero-field splitting is the breaking of degeneracies of the ground state that is not described by a standard non-relativistic Hamiltonian. ZFS as calculated by ADF is that exhibited by molecules whose ground state has spin S>1/2 and no spatial degeneracy. This type of ZFS has two contributions, second-order spin-orbit coupling and direct electron spin-spin coupling. The calculation of ZFS with DFT is described in [*312-315* (page 925)]. With the keyword ZFS the spin-orbit coupling term is included. If one also wants to calculate the direct electron spin-spin term, one has to include the key HARTREEFOCK, which calculates the Coulomb and (Hartree-Fock like) exchange contributions to the direct electron spin-spin term. In the spin-orbit coupling term no Hartree-Fock like exchange contributions are included.

```
HARTREEFOCK
```

ZFS can be calculated in combination with LDA and GGAs but not hybrid or meta-GGA functionals. In order to calculate ZFS the RELATIVISTIC SCALAR ZORA option must be included.

Just the simple keyword ZFS is needed in order to calculate zero-field splitting. Several optional additional keywords can also be included. The complete list is:

```
ZFS {PEDERSON|NEESE} {ANALYSIS|FULLANALYSIS}
```

PEDERSON|NEESE PEDERSON: The available approaches for calculating ZFS with DFT each differ subtly from the others. We believe that the method proposed by van Wüllen and coworkers [*314,315* (page 925)] is the most theoretically complete but it may be that for certain systems the other approaches are more accurate. The van Wüllen formulation is the default but if the PEDERSON keyword in included then the equation proposed by Pederson and Khanna [*312* (page 925)] is used. NEESE: If the NEESE keyword is included then the equation for ZFS proposed by Neese [*313* (page 925)] is used.

ANALYSIS|FULLANALYSIS ANALYSIS: Neese has presented some interesting analyses of ZFS [*313* (page 925)]. If the ANALYSIS keyword is invoked then the contributions to the ZFS is divided into terms from alpha-beta, alpha-alpha, beta-beta and beta-alpha one-electron excitations. FULLANALYSIS: The output requested by the ANALYSIS keyword is further extended to analyze each of the alpha-beta, alpha-alpha, beta-beta and beta-alpha contributions in terms of the individual one-electron excitations.

6.10 Nuclear Quadrupole Interaction (EFG)

```
QTENS
```

This key activates the computation of the Nuclear Electric Quadrupole Hyperfine interaction. It can be applied to open-shell and to closed-shell systems. QTENS gives you the Nuclear Electric Quadrupole Hyperfine interaction (Q-tensor) [*97* (page 912)]. The latter is directly related to the Electric Field Gradient (EFG) . The Q-tensor elements (in MHz) equal the the electric field gradient tensor elements (in a.u.) times 234.9647 times the nuclear quadrupole moment (NQM in barn units, 1 barn = 10-28m^2 = 10-24cm^2) and divided by 2I(2I-1), where I is the nuclear spin. The Nuclear Quadrupole Coupling Constant (NQCC) (in MHz) is the largest value of the principal values of the EFG (in a.u.) times 234.9647 times the nuclear quadrupole moment (in barn units). The electric field gradient tensor is printed next to the Q-tensor.

In the case of ZORA the program will calculate the EFG in the so called ZORA-4 approximation, which includes a small component density ("picture-change correction"), see [*97* (page 912)]. If you also want to calculate the EFG using the ZORA density, thus in the Foldy-Wouthuysen picture, include the keywords:

```
PRINT FWPICTURE
```

In case QTENS is used for ^{57}Fe, ^{119}Sn, ^{125}Te, ^{193}Ir, and ^{197}Au, quadrupole splittings are written in units of mm/s, used in Mössbauer spectroscopy.

Analysis of the EFG

With the EFG keyword in AOResponse a Mulliken type analysis of the EFG principal components, and an analysis in terms of canonical MOs, can be performed. Required is symmetry NOSYM. This not implemented in case of spin-orbit coupling. For an NBO analysis of the EFG, see the *section on NBO analysis* (page 243). For an explanation of the output and a general usage tutorial, see [*327* (page 926)]. Further references and recommended citations, see [*328* (page 926)].

```
Symmetry NOSYM
Aoresponse
 efg NUC
end
```

efg NUC Here NUC is the number of the nucleus at which the EFG is to be computed (ADF internal atom ordering). Availiable for one nucleus at the time.

6.11 Mössbauer spectroscopy

Isomer shifts

By default the electron density at the nuclei is calculated, no input key is required. In the implementation in ADF, the electron density is not calculated exactly at the center of the nucleus, however, at points on a small sphere around the center of a nucleus. The printed electron density in the output of ADF is the average electron density on these points. The radius of the sphere is an approximated finite nuclear radius. The electron density at the nuclei could be used for the interpretation of isomer shifts in Mössbauer spectroscopy. Typically one needs to perform a fit of the experimentally measured isomer shifts and calculated electron densities, like, for example, is done in Ref. [*281* (page 923)].

One should use all electron basis sets for the Mössbauer active elements. Important is to use the same basis set, same exchange correlation functional, same integration accuracy, and same nuclear model (see key NUCLEARMODEL), if electron densities at nuclei in different molecules are compared. Note that the absolute electron density at a nucleus heavily depends on the accuracy of the basis set in the core region of this nucleus, especially if relativistic effects are included.

In case of ZORA the electron density at the nuclei is calculated with the ZORA-4 method, which includes a small component density. Previously it was using the ZORA density, which does not include a small component density. If you also want to calculate this ZORA density at the nuclei, include the keywords:

```
PRINT FWPICTURE
```

Instead of SAPA (the Sum of neutral Atomical potential Approximation) MAPA is used by default for ZORA in ADF2017. The MAPA (the Minumium of neutral Atomical potential Approximation) at a point is the minimum of the neutral Atomical potentials at that point. Advantage of MAPA over SAPA is that the gauge dependence of ZORA is reduced. The ZORA gauge dependency is small for almost all properties, except for the electron density very close to a heavy nucleus.

A ZORA calculation gives a too large density at the nucleus compared to a relativistic Dirac calculation, mostly due to the 1s-orbital. If one performs a fit of the experimentally measured isomer shifts against calculated electron densities the fact that the ZORA electron density is too large is not so important. If an absolute value of the electron density at the nucleus is important one may use the relativistic X2C method, see key *RELATIVISTIC* (page 72). Note that one then needs large basis sets with tight STO 1s-functions to get accurate results (for example the basis sets in the TZ2P-J or QZ4P-J directory).

Quadrupole splittings

For the calculation of Mössbauer quadrupole splittings see the key *QTENS* (page 221).

Nuclear resonance vibrational spectroscopy (NRVS)

The nuclear resonance vibrational spectroscopy (NRVS) experiment can be thought of as Mösbauer spectroscopy with vibrational sidebands. The NRVS experiment provides the complete set of bands corresponding to modes that involve motion of the Mössbauer active atoms. In order to calculate this with the ADF program a partial vibrational Density-Of-States (PVDOS) has been implemented. A PVDOS factor for a given atom is the ratio of this atom nucleus kinetic (vibrational) energy to the total vibrational energy of all nuclei, for a given mode. PVDOS factors for every atom and every mode are written to TAPE21 if *IR Frequencies* (page 139) are calculated. To visualize the calculated PVDOS use the ADFspectrum program: select the PVDOS spectrum type. Next select one or more atoms to get the PVDOS spectrum generated by the selected atoms. This is useful for analysis of NRVS spectra in bioinorganic chemistry for NRVS-active nuclei.

TRANSPORT PROPERTIES

See also:
Examples: *transport properties* (page 754)

7.1 Charge transfer integrals (transport properties)

ADF can provide input parameters, such as charge transfer integrals, that are needed in approximate methods that model charge transport properties. ADF has the unique feature that it can (also) calculate such transfer integrals based on the direct method by the use of its unique fragment approach.

In theoretical models of charge transport in organic materials, see Refs. [*293-295* (page 924)], the whole system is divided into fragments, in which an electron or hole is localized on a fragment, and can hop from one fragment to another. In the tight-binding approximation that is used in these models the electron or hole is approximated with a single orbital, and it is assumed that only the nearest neighboring fragments can couple. The models require accurate values of electronic couplings for charge transfer (also referred to as charge transfer integrals or hopping matrix elements) and site energies (energy of a charge when it is localized at a particular molecule) as a function of the geometric conformation of adjacent molecules. Charge transfer integrals for hole transport can be calculated from the energetic splitting of the two highest-occupied molecular orbitals (HOMO and HOMO-1) in a system consisting of two adjacent molecules, also called "energy splitting in dimer" (ESID) method. For electron transport these can be calculated from the two lowest-unoccupied orbitals (LUMO and LUMO+1) in this ESID method. ADF can also calculate transfer integrals based on the direct method by the use of its unique fragment approach. see Refs. [*294,295* (page 924)]. ADF allows one to use molecular orbitals on individual molecules as a basis set in calculations on a system composed of two or more molecules. The charge transfer integrals obtained in this way differ significantly from values estimated from the energy splitting between the highest occupied molecular orbitals in a dimer. The difference is due to the nonzero spatial overlap between the molecular orbitals on adjacent molecules. Also, ADF's methods are applicable in cases where an orbital on one molecule couples with two or more orbitals on another molecule.

7.1.1 Charge transfer integrals with the TRANSFERINTEGRALS key

In this method the matrix elements of the molecular Kohn-Sham Hamiltonian in the basis of fragment orbitals is used to calculate site energies and charge transfer integrals. Likewise the overlap integrals between fragment orbitals are calculated. No explicit electrons are removed or added in this method. For electron mobility calculations the fragment LUMO's are considered. For hole mobility calculations the fragment HOMO's are considered.

To calculate the charge transfer integrals, spatial overlap integrals and site energies, include the key TRANSFERINTEGRALS in the input for ADF. Symmetry NOSYM should be used. The molecular system typically should be build from 2 fragments. In the fragment calculation full symmetry can be used.

```
TRANSFERINTEGRALS
Symmetry NOSYM
Fragments
 frag1 frag1.t21
 frag2 frag2.t21
End
```

By default, integrals are calculated only for the HOMO (LUMO) of the fragments, and possibly HOMO-1, HOMO-2 (LUMO+1, LUMO+2) if the energy of those fragment orbitals are close to the HOMO (LUMO) of that fragment. To calculate the matrix elements and overlap integrals based on all fragment orbitals one can use the key:

```
PRINT FMATSFO
```

The method described here to calculate charge transfer integrals is more approximate than the next method that uses FDE. The major difference is how effects of a localized charge are included.

If 2 fragments are used the electronic coupling V (also known as effective (generalized) transfer integrals J_eff) for hole transfer or electron transfer is calculated as V = (J-S(e1+e2)/2)/(1-S^2). Here e1, e2, are the site energies of fragment 1 and 2, respectively. J is the charge transfer integral, and S the overlap integral. Note that the effect of a possible degeneracy of HOMO or LUMO is not taken into account. The electronic coupling between the HOMO of the donor fragment and the LUMO of the acceptor fragment and vice-versa is also calculated, which represent the probability of a charge recombination process.

7.1.2 Charge transfer integrals with FDE

Overview The ELECTRONTRANSFER keyblock invokes the calculation of Hamiltonian (site energies and couplings) and overlap matrix elements with FDE-derived localized states. Two FDE calculations are (not strictly) needed before running the ELECTRONTRANSFER calculation. The calculated matrix elements are theoretically similar to the ones obtained with the TRANSFERINTEGRALS keyword. A recent review casts ELECTRONTRANSFER in the state-of-the-art of methods for computing charge transfer couplings, and provides a step-by-step guide for computing such couplings with ADF [*405* (page 930)].

Features

- Effects of orbitals relaxation due to localized charges, Refs.[*352,353* (page 927)].

- Effects of polarization due to molecules in the environment, Ref.[*353* (page 927)].

- ELECTRONTRANSFER is linear scaling in the number of fragments when the system is composed by more than one fragment.

- The code can tackle hole transfer, electron transfer, charge separation and charge recombination processes [*406* (page 931)].

- Can compute transfer integrals from diabats obtained with the Constrained DFT method (i.e., invoked by using the experimental CDFT keyword of ADF). This includes calculation of couplings between diabats made of a single fragment.

- It is possible to include the effect of the environment on CDFT diabats by coupling it with an FDE calculation [*407* (page 931)].

- The performance of ElectronTransfer in the evaluation of the hole transfer coupling, was benchmarked against wave functions methods, with an error below 7%. PBE, PW91, B3LYP and PBE0 functionals, with PW91k for the non-additive component of the kinetic energy and TZP basis function, are recommended to obtain the FDE-derived localized states [*408* (page 931)].

- The code was parallized in ADF2018.

Limitations

- Hybrid functionals are not yet supported

- TAPE21 files with charge or spin-localized states with specific names are needed

These two limitations do not apply to the method with the TRANSFERINTEGRALS key, but the TRANSFERINTE-GRALS has many other limitations. Like the method with the TRANSFERINTEGRALS key fragments are needed.

ELECTRONTRANSFER input

The minimum input for the ELECTRONTRANSFER key is:

```
FRAGMENTS
 frag1 FragFile1
 ...
 fragN FragFileN
END
ELECTRONTRANSFER
 NumFrag N
END
```

where frag1 ... fragN are the labels of the fragments in the calculation, and FragFile1 ... FragFileN are the TAPE21 files of spin RESTRICTED calculations of the isolated fragments, N is the total number of fragments employed in the calculation.

Files and file names

The fragment files to be used in the ELECTRONTRANSFER calculation are generally different from the TAPE21 files used in the FRAGMENTS key block. Two types of fragment TAPE21 files are needed by the calculation:

- The isolated closed-shell TAPE files for the FRAGMENTS keyblock

- The TAPE21 files of the charge or spin localized states (which can be obtained with an FDE calculation as done in the example below)

There are 2 charge localized states. They are labeled with A and B. The respective TAPE21 files must be names as follows:

```
fragA1.t21, fragA2.t21, ... , fragAN.t21 (for state A)
fragB1.t21, fragB2.t21, ... , fragBN.t21 (for state B)
```

The above files should be copied to the working folder of the ADF calculation prior to executing ADF.

Options

```
ELECTRONTRANSFER
 NumFrag N
 {Joint|Disjoint}
 {Debug}
 {Print EIGS|SAB}
 {FDE}
 {INVTHR threshold}
 {CDFT}
 {KNADD}
END
```

Joint|Disjoint The default is "Joint". Joint is always recommended. The "Disjoint" formalism is described in Ref.[*353* (page 927)] and is much faster than the "Joint" formalism when more than 2 fragments are considered. Joint and Disjoint are equivalent for systems composed of only 2 fragments. Disjoint should only be used if the fragment files are obtained in an FDE calculation (see FDE below).

Debug The code performs additional checks (determinants, diagonalizations, inversions, traces, etc.). Substantial increase in the output should be expected.

Print If EIGS, it will print the (unformatted) matrix of the MO coefficient in the AO representation. If SAB, it will print the (unformatted) matrix of the diagonal and transition overlap matrix in the MO representation.

FDE An FDE calculation including more than 2 fragments must include the following key block:

```
ELECTRONTRANSFER
  FDE
END
```

and the numerical integration precision in the *last* FDE calculation for every subsystem should be set to no less than:

```
BeckeGrid
   Quality Good
End
```

if in the subsequent ELECTRONTRANSFER calculation the DISJOINT subkey is used.

Invthr threshold Default 1.0e-3, is a threshold for the Penrose inversion of the transition overlap matrix. If warnings about density fitting are printed, invthr may be increased up to 1.0e-2. Larger invthr might affect the quality of the calculated couplings and excitation energies.

CDFT If disjoint is selected, this keyword must be selected if the evaluation of the electronic coupling is sought between diabats located on the same CDFT fragment. In this case, the CDFT fragment has to be always first under the ATOMS keyword.

KNADD If disjoint is selected, this allows the kinetic energy of each fragment to be obtained locally without using the full grid. This keyword is recommended when there are many tens of subsystems, such as systems with several solvent molecules.

Output

The output of the example in $ADFHOME/examples/adf/ElectronTransfer_FDE_H2O is discussed here. This example involves the calculation of electronic coupling, site energies and charge-transfer excitation energy for the hole transfer in a water dimer.

```
============   Electron Transfer RESULTS ==================

Electronic Coupling =          0.000000 eV
Electronic Coupling =         -0.003569 cm-1
H11-H22          =         -1.396546 eV
Excitation Energy  =          1.396546 eV
Overlap          =          0.000000
H11 H22 H12 =  -152.443000816341  -152.391678701092  -151.743979368040 Eh
S11 S22 S12 =     0.981795415192     0.981006454450    -0.000000023700


=========== END Electron Transfer RESULTS ===============
```

Due to symmetry, the overlap is almost diagonal (Overlap = 0.00), thus the transition density is evaluated with one less electron as explained in Ref. [*353* (page 927)].

The electronic coupling between the state with a positive charge localized on one water molecule and another with the charge localized on the other water molecule is given by "Electronic Coupling" and is reported in eV and cm^-1.

"H11-H22" is the difference of the site energies in eV. Values of the site energies are given by the first two values of "H11 H22 H12" in atomic units.

"Excitation Energy" reports the value of the transfer excitation energy as calculated by diagonalization of the 2X2 generalized eigenvalue problem in the basis of the charge-localized states, see Refs. [*352,353* (page 927)].

"S11 S22 S12" are the values of the non-normalized overlaps.

7.2 GREEN: Non-self-consistent Green's function calculation

green is an auxiliary program which can be used to calculate the density of states (DOS) and transmission of molecules connected to semi-infinite contacts. The transmission is the electron transmission through a molecule connected to semi-infinite contacts. The calculation is based on the non-self-consistent Green's function method, which is an approximation to the non-equilibrium Green's function (NEGF) method. The details of this method can be found in chapter 2 and appendix C of the PhD thesis of Jos Seldenthuis (2011) (http://www.scm.com/Doc/Seldenthuis2011.pdf). See Ref. [365 (page 928)] for more details on the applicability of the wide-band limit approximation. Suggestion is to consider to use NEGF implementation in the periodic program BAND program, see the BAND manual, since it might give better results.

7.2.1 Introduction

The utility program *green* calculates the density of states (DOS) and zero-bias transmission of molecule connected to two semi-infinite contacts. A typical calculation consists of two parts. The first is the calculation of the effect of the semi-infinite contacts, contained in the so-called self-energy matrix. The second is the calculation of the desired properties of the molecule with the self-energies.

Self-energy
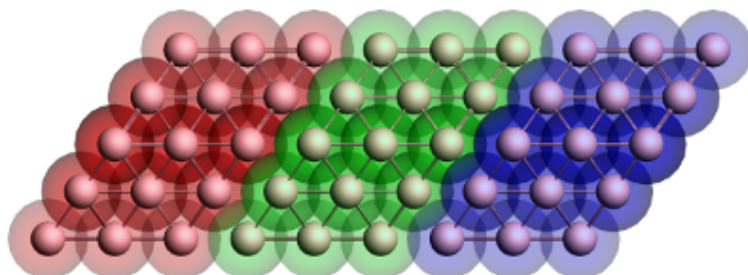
Figure 1: Geometry of the gold contact used in the calculation of the self-energy. The lead consists of two surface layers, left (red) and right (blue), and a bulk layer (green). Each principal layer in turn consists of three atomic layers. This should be sufficient to ensure that the Hamiltonian of the central (green) layer is a bulk Hamiltonian.

Since the contacts are semi-infinite, the calculation of their self-energy is effectively a bulk calculation. Since ADF only works with systems of finite size, approximations have to be made. Fig. 1 shows the typical geometry used in the calculation of a gold contact. The geometry consists of three parts, the so-called principal layers. These layers should be large enough that the atoms on one side are not influenced by whatever is attached to the other side. Three atomic layers usually suffice. The green region is the bulk layer. The red and blue regions are the surface layers. Note that the blue region corresponds to the left contact of a molecule and the red region to the right contact.

To calculate the self-energy, we first need to do a single-point calculation of a principal layer. This layer is then used as a fragment in the following calculations. Note that all ADF calculations have to be performed with SYMMETRY NOSYM. We then build up the contact geometry from three copies of the layer fragment as in Fig. 1 and perform another single-point calculation. This results in a Hamiltonian describing the three contact layers and the coupling between them.

From the TAPE21 file *green* can now calculate the self-energy matrices with the SURFACE key. This has to be done once for every energy for which we want to calculate the DOS or transmission. For the left contact of the molecule, *green* needs the blue and green fragments. The self-energy is calculated by taking the (blue) surface layer and iteratively adding more (green) bulk layers until matrices converge to the semi-infinite result. The self-energy of the right contact is similarly calculated from the red and green fragments. Since the self-energy described the effect of

---

an infinite chain of (green) bulk regions on a (red or blue) surface layer, this calculation does not depend on whatever is attached to the contacts. The self-energy matrices can therefore be reused for different molecules.

DOS and transmission
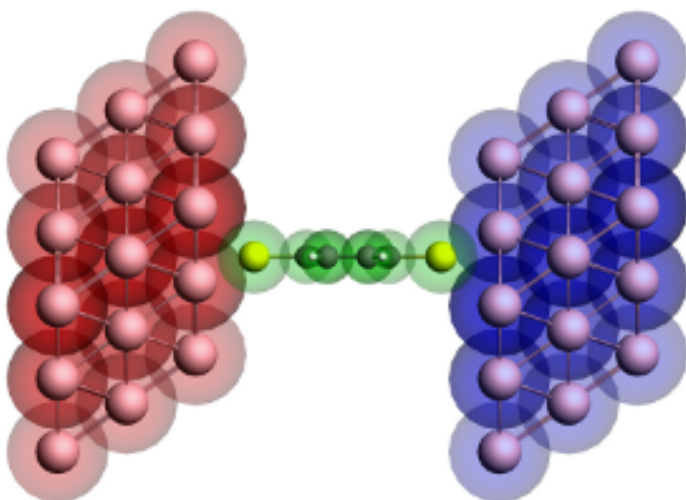
Figure 2: Geometry of the extended molecule used in the calculation of a benzenedithiol junction. The molecule is shown in green, while the left and right contact regions are shown in red and blue, respectively. Note that the red region corresponds to the blue surface layer in Figure 1 and vice versa.

Once the self-energy matrices have been calculated for the desired energies, we can compute the DOS and transmission of a molecule. However, since the self-energy matrices couple to the surface layers of the contacts, we need to include those surface layers in the calculation of the molecule (see Fig. 2). We therefore first perform a single-point calculation with ADF of the isolated molecule. The result is then used as a fragment and combined with the fragments of the surface layers to construct the so-called extended molecule. We then perform another single-point calculation of the final geometry.

From the self-energies of the contacts and the TAPE21 file of the extended molecule, *green* can now compute the DOS and transmission. This calculation is non-self-consistent since the ADF calculations are all performed on finite instead of semi-infinite systems. This will result in certain artifacts in the DOS and transmission spectra, but those can be made arbitrarily small by choosing the principal layers large enough.

7.2.2 Wide-band-limit

In the wide-band limit (WBL) the coupling to the leads is assumed to be independent of energy. Therefore one does not need to calculate any self-energies. This also means that the eigenspace of the Green's function is independent of energy. It can therefore be diagonalized in advance, greatly speeding up the calculation of the DOS and the transmission. See Ref. [*365* (page 928)] for more details on the applicability of the wide-band limit in DFT-based molecular transport calculations.

In the example $ADFHOME/examples/adf/green_Al/green_WBL.run of *green*, the transmission of *benzenedithiol junction* (page 804) in the wide-band limit (WBL) is calculated. In order to model the molecule-metal interface, we do need to include a few gold layers in the calculation. However, unlike *before* (page 798), only a single atomic layer as the principal layer is used. Because a single atomic layer is an unnatural configuration for gold, a minor amount of smearing is necessary to make the calculation converge. The molecule is sandwiched in between the electrodes just like before (see Fig. 2 in *the example for benzenedithiol* (page 804)). However, this time each atomic layer of gold gets its own fragment. The reason for this configuration is that if the WBL is used on the entire gold contact the result is an an unphysical coupling to the leads; even the gold atoms contacting the molecule would have a direct coupling to the environment. A much better result can be obtained by only using the WBL on the back-most atomic layer and letting

the electrons propagate naturally through the rest of the contact. Because the WBL is computationally so inexpensive, we can easily calculate the DOS and transmission for 10,000 points instead of 1000.

A comparison of the resulting transmission with the calculation with self-energies is shown in the following figure:

The WBL shows good agreement with the non-WBL transmission around the Fermi energy (-0.195 Hartree or -5.306 eV). Note that the quality of the WBL depends on the choice of the coupling (ETA). For this particular contact geometry we obtain good agreement for ETA = 0.02 Hartree, but a better value may be found for other electrodes. Finally, the WBL can be incrementally improved by adding more gold layers to the extended molecule. For many layers it converges to the calculation with full self-energies.

7.2.3 Input options

The input for *green* is keyword oriented and is read from the standard input. *green* is typically first used to calculate the self-energy matrices of the left and right contacts (with the SURFACE key), and then to calculate the density of states (DOS) and transmission (with the DOS and TRANS keys, respectively), using those self-energies. The only keyword required to be present in all calculations is the EPS keyword, which specifies the energy range.

```
$ADFBIN/green << eor
 EPS mineps maxeps numeps
 {ETA eta}
 {SO sh sl {moc}}
 {SURFACE filename
     FRAGMENTS f1 f2
  END}
 {DOS filename}
 {TRANS
  LEFT filename
     FRAGMENT fragment
     ETA eta
  END
  RIGHT filename
     FRAGMENT fragment
     ETA eta
  END}
eor
```

EPS mineps maxeps numeps The energy range for which either the self-energy matrices or the DOS and transmission have to be calculated. The range consists of numeps (≤ 1) points running from mineps to maxeps

---

inclusive.

(optional) `ETA eta` The imaginary energy, or the distance from the real axis, in the calculation of the Green's function. The value needs to be a small positive number to prevent singularities in the calculation. The default value (10^{-6} Hartree) is sufficient for most calculations.

(optional) `SO sh sl {moc}` The shifts for the scissors operator. All occupied orbitals (HOMO and below) are shifted by sh, while the unoccupied orbitals (LUMO and above) are shifted by sl. Orbitals are considered occupied if their (possibly fractional) occupation is larger than moc (default 0). The scissor operator can partially remedy the underestimation of the HOMO-LUMO gap in DFT. The sh and sl shifts generally have the same magnitude, but opposite sign (with sh usually being negative and sl positive). A good estimate for the magnitude is the sum of the ionization potential and the energy of the HOMO of the *free* molecule. This can be improved by including image charge effects. For more details, see Ref. [*351* (page 927)]. By default, sh=sl=0.

(optional) `SURFACE`

```
SURFACE filename
   FRAGMENTS f1 f2
END
```

The SURFACE block key enables the calculation of the self-energy matrices. The filename specifies the TAPE21 file resulting from an ADF calculation of the contacts. This calculation has to be performed with SYMMETRY NOSYM. The FRAGMENTS key is used to specify the two principal layers between which the surface is defined. The resulting self-energy matrices (one for every energy point given by EPS) is stored in a binary KF file named SURFACE.

(optional) `DOS filename` The DOS key enables the calculation of the density of states. The filename specifies the TAPE21 file containing the result of an ADF calculation of the extended molecule (performed with SYMMETRY NOSYM). Two text files will be generated: DOS_A and DOS_B, containing, respectively, the DOS of the spin-A and spin-B electrons. In the case of a spin-unrestricted calculation, DOS_A and DOS_B might differ. If only the DOS of the spin-A electrons is required, the calculation can be sped up by specifying NOSAVE DOS_B. The DOS key requires the presence of the LEFT and RIGHT keys.

(optional) `TRANS` The TRANS key enables the calculation of the transmission. The filename specifies the TAPE21 file containing the result of an ADF calculation of the extended molecule (performed with SYMMETRY NOSYM). Two text files will be generated: TRANS_A and TRANS_B, containing, respectively, the transmission of the spin-A and spin-B electrons. In the case of a spin-unrestricted calculation, TRANS_A and TRANS_B might differ. If only the transmission of the spin-A electrons is required, the calculation can be sped up by specifying NOSAVE TRANS_B. The TRANS key requires the presence of the LEFT and RIGHT keys.

`LEFT/RIGHT`

```
LEFT filename
   FRAGMENT fragment
   ETA eta
END
```

The LEFT and RIGHT block keys specify the left and right self-energies used in a calculation of the DOS and transmission. If a filename is specified, the self-energy matrices are read from that file. The energy range of the self-energies has to be consistent with the range specified by the EPS keyword. The FRAGMENT key is used to denote the fragment in the extended molecule (given by the argument to the DOS or TRANS key) to which the self-energy couples. If no filename is specified, the wide-band limit is used. The ETA key can then be used to specify the magnitude of the coupling (10^{-3} Hartree by default).

7.2.4 Output

After a successful calculation of the self-energy matrices, *green* produces a binary KF file named SURFACE containing two sections. The Surface section contains the energy range:

contents of Surface	comments
mineps	start of the energy range
maxeps	end of the energy range
numeps	number of points

The Sigma section contains the real and imaginary parts of the self-energy matrices:

contents of Sigma	comments
nfo	number of fragment orbitals (dimension of the self-energy matrices)
Re(Sigma_%d)	the real part of the %d self-energy matrix (numbered from 1 up to numeps)
Im(Sigma_%d)	the imaginary part of the %d self-energy matrix (numbered from 1 up to numeps)

A successful calculation of the density of states (DOS) or transmission results in the text files DOS_A and DOS_B, and TRANS_A and TRANS_B, respectively. The suffixes _A and _B denote the different spins. The text files the DOS and transmission for every energy point and can be plotted with, for example, gnuplot.

7.2.5 GREEN with ADF-GUI

In ADF2017 the ADF-GUI can be used to setup a GREEN calculation, and look at the results. Note, however, that not all GREEN options are supported with the ADF-GUI. For example, one can not use unrestricted or wide-band-limit calculations with the ADF-GUI. GUI tutorials exist for NEGF with DFTB or BAND. Suggestion is to consider to use NEGF implementation in the periodic program BAND program, see the BAND manual, since it might give better results.

ANALYSIS

See also:

ADF-GUI tutorial: all ADF tutorials, fragment analysis

Examples: *analysis* (page 808)

8.1 Molecules built from fragments

ADF analyzes the results in terms of user-specified subsystems from which the total system is built. The program tells you how the 'Fragment orbitals' (FO's) of the chemically meaningful sub-units mix with FO's on other fragments to combine to the final molecular orbitals.

ADF builds a molecule from user-defined fragments, which may be single atoms or larger moieties, for example, ligands, functional groups, or complete molecules in a donor-acceptor complex. In practice, this means that the results of the ADF calculation on a fragment are saved on a file and that the fragment files are then used in setting up the calculation on the overall system. The fragment orbitals (FOs), i.e., the MOs from the calculations on the fragments, are employed as basis functions in the new calculation. This does not imply a basis set truncation or contraction because the virtual FOs are included: the FOs constitute only a transformation of the basis set. If there are symmetry-equivalent fragments, for example, the six CO molecules in octahedral $Cr(CO)_6$, the program generates symmetry combinations of the FOs and uses the symmetrized fragment orbitals (SFOs) as basis functions. The SFOs transform as the irreducible representations (irreps) of the molecule, allowing a symmetry-driven analysis of the results. In absence of any symmetry the SFOs are identical to the FOs.

The fragment approach offers considerable advantages. It enhances the interpretative power of ADF as it leads to a more transparent picture of bonding, which reduces from a complicated mixing of many primitive basis functions (possessing little physical relevance) to a few key interactions between meaningful fragment (frontier) orbitals. The fragment approach also improves the numerical precision. In ADF, energies are calculated directly, with respect to the fragments, by one single numerical integral of the difference energy density $\epsilon[\rho, \mathbf{r}] - \sum_A \epsilon_A[\rho_A, \mathbf{r}]$ between the overall molecule and the constituting fragments.

$$\Delta E[\rho] = \int d\mathbf{r} \left(\epsilon[\rho, \mathbf{r}] - \sum_A \epsilon_A[\rho_A, \mathbf{r}] \right)$$

In other words, we evaluate the energy of the overall molecule, $E[\rho] = \int d\mathbf{r}\epsilon[\rho, \mathbf{r}]$, and the energies of each of the fragments, say the atoms that constitute the overall molecule, $E_A[\rho] = \int d\mathbf{r}\epsilon_A[\rho_A, \mathbf{r}]$, in the same numerical integration grid. This provides more accurate relative energies than subtracting total energies from separate calculations, because the same relative numerical integration error applies to a much smaller quantity, yielding, in turn, a much smaller absolute error.

Note that the user has the freedom to make his own choice of fragments. This is, however, not a matter of plain arbitrariness, and it does not make the analysis tools less meaningful. On the contrary, this freedom simply reflects the many perspectives from which a particular chemical phenomenon can be viewed.

In practice, many calculations are performed using as fragments the so-called basic atoms, which are the smallest possible building blocks in ADF. The basic atoms are not necessarily physically realistic objects - indeed, usually they are not, as they must be spin-restricted and spherically symmetric. The computed (bonding) energy w.r.t. basic atoms, then, does not yield quantities that can be compared to experimental data directly. Rather, one must correct for the true ground state of the isolated single atoms.

Text is mostly taken from: *Chemistry with ADF* (page 925), G. te Velde, F.M. Bickelhaupt, E.J. Baerends, C. Fonseca Guerra, S.J.A. van Gisbergen, J.G. Snijders, T. Ziegler J. Comp. Chem. **22** (2001) 931.

Link: *How to make molecular fragments* (page 39)

Tutorial: ADF fragment analysis

Examples: *analysis options* (page 808)

8.2 Bond energy analysis

No special input keys are required. However, if one has open shell fragments, see key FRAGOCCUPATIONS.

ADF calculates various chemically meaningful terms that add up to the bond energy, with an adaptation of Morokuma's bond energy decomposition to the Kohn-Sham MO method. The individual terms are chemically intuitive quantities such as electrostatic energy, Pauli repulsion, and orbital interactions. The latter are symmetry decomposed according to the Ziegler transition state method. For a discussion of bonding energy decompositions and applications see e.g. [*3* (page 907), *110* (page 913), *112* (page 913), *130-136* (page 914)]. For a discussion of forming electron pair bonding between open shell molecules see Ref. [*489* (page 935)]. For an ETS-NOCV analysis of the orbital interaction, see Ref. [*261* (page 922)].

In ADF2012 the calculation of the Pauli repulsion for metaGGA's and metahybrids is implemented. Note that for hybrids this was already implemented before in case of closed shell fragments. In ADF2012 for hybrids, metaGGA's, and metahybrids the calculation of the Pauli repulsion is also implemented if one is simulating an unrestricted fragment with the key FRAGOCCUPATIONS.

In ADF2012 for hybrids the exact exchange contribution to the Pauli term is isolated and the contributions to the orbital term are divided among orbital symmetries.

8.2.1 Bond energy details

In the framework of Kohn-Sham MO theory and in conjunction with the fragment approach, one can decompose the bond energy between the fragments of a molecular system - say, a base and a substrate for E2 elimination - into contributions associated with the various orbital and electrostatic interactions. In ADF, we follow a Morokuma-type energy decomposition method. The overall bond energy ΔE is divided into two major components. In the first place, the preparation energy ΔE_{prep} corresponding to the amount of energy required to deform the separated fragments, A and B say, from their equilibrium structure to the geometry they acquire in the overall molecule ($\Delta E_{\text{prep,geo}}$), and to excite them to their valence electronic configuration ($\Delta E_{\text{prep el}}$). In the second place, the interaction energy ΔE_{int} between the prepared fragments.

$$\Delta E = \Delta E_{\text{prep}} + \Delta E_{\text{int}} = \Delta E_{\text{prep,geo}} + \Delta E_{\text{prep,el}} + \Delta E_{\text{int}}$$

In the following step, the interaction energy ΔE_{int} is further decomposed into three physically meaningful terms, which are printed in the ADF output file.

$$\Delta E_{\text{int}} = \Delta V_{\text{elst}} + \Delta E_{\text{Pauli}} + \Delta E_{\text{oi}} = \Delta E^0 + \Delta E_{\text{oi}}$$

The term ΔV_{elst} corresponds to the classical electrostatic interaction between the unperturbed charge distributions of the prepared fragments as they are brought together at their final positions, giving rise to an overall density that is

simply a superposition of fragment densities $\rho_A + \rho_B$. (Note that we use the convention that energy terms containing potential energy only, kinetic energy only, or both kinetic and potential energy are indicated by V, T, and E, respectively.) For neutral fragments, ΔV_{elst} is usually attractive. The Pauli repulsion ΔE_{Pauli} arises as the energy change associated with going from $\rho_A + \rho_B$ the wave function $\Psi^0 = NA[\Psi_A\Psi_B]$ that properly obeys the Pauli principle through explicit antisymmetrization (A operator) and renormalization (N constant) of the product of fragment wave functions. It comprises the destabilizing interactions between occupied orbitals, and is responsible for any steric repulsion. In case of neutral fragments, it can be useful to combine ΔV_{elst} and ΔE_{Pauli} in a term ΔE_0 which, in the past, has been conceived as the steric interaction. However, we prefer to reserve the designation steric interaction or repulsion for ΔE_{Pauli} because that is, as already mentioned, the only source of net repulsive interactions between molecular fragments. Finally, the wavefunction is allowed to relax from Ψ^0 to the fully converged wave function Ψ. The associated orbital interaction energy ΔE_{oi} accounts for electron pair bonding, charge transfer (e.g., HOMO-LUMO interactions) and polarization (empty/occupied orbital mixing on one fragment due to the presence of another fragment). In a symmetric molecule this can be further decomposed into the contributions from the distinct irreducible representations Γ of the interacting system using the extended transition state method. In systems with a clear σ/π separation, this symmetry partitioning proves to be very informative.

$$\Delta E_{\text{oi}} = \sum_\Gamma \Delta E_{\text{oi},\Gamma}$$

An alternative, which can also be used in asymmetric molecules, is the ETS-NOCV analysis, which decomposes the bonding interactions in the context of natural orbitals for chemical valence, see Refs [*261* (page 922), *262* (page 922), *490* (page 935)].

The pure orbital interaction effect of forming electron pair bonding between open shell molecules can approximately be calculated with a bond energy analysis in which all virtual orbitals are removed from the fragments, see Ref. [*489* (page 935)].

An extensive discussion of the physical meaning of all the terms in the energy decomposition is given in F.M. Bickelhaupt and E.J. Baerends, *Kohn-Sham Density Functional Theory: Predicting and Understanding Chemistry*, In: Rev. Comput. Chem.; Lipkowitz, K. B. and Boyd, D. B., Eds.; Wiley-VCH: New York, 2000, Vol. 15, 1-86.

Text is mostly taken from: *Chemistry with ADF* (page 925), G. te Velde, F.M. Bickelhaupt, E.J. Baerends, C. Fonseca Guerra, S.J.A. van Gisbergen, J.G. Snijders, T. Ziegler J. Comp. Chem. **22** (2001) 931.

8.2.2 Total energy evaluation

ADF normally does not calculate the total energy of a system (the energy wrt bare nuclei and free electrons). However, ADF calculates the energy of the system with respect to fragment energies. By default, these fragments are the spherical spin-restricted neutral atoms, but one can also use larger fragments. For this reason total energies from other programs could not be compared to ADF directly. Note, however, that only energy difference comparisons are meaningful. These are the only energies that play a role in chemistry of course, and for this one does not need total energies.

If you really want to calculate the total energies, there are two options in ADF

Total energy by adding the binding energy of the atoms

There is a work-around to calculate the total energy of a system: calculate the total energies of the atomic fragments and add them to the bonding energy. Because total energy of an atom is, by definition, the energy difference between the atom and the (nucleus+free electrons) system one can calculate it by calculating a single atom with the charge equal to the number of electrons. 'Bonding energy' of such an 'atom' will then be equal to negative of the total energy of the atomic fragment. Care should be taken to apply this procedure to frozen-core fragments. In this case, it only makes sense to remove the valence electrons and leave the frozen core.

TOTALENERGY keyword

The total energies have not been tested extensively and should therefore be used with caution. In addition to bond energies it is now possible to compute total energies with ADF by including the keyword TOTALENERGY in the input. This work is in progress.

```
TOTALENERGY
```

The total energy will be computed for the chosen XC functional (LDA, GGA, hybrid functionals, or Hartree-Fock). MetaGGA functionals, (ZORA) scalar relativistic and relativistic spin-orbit calculations, electric fields and QM/MM are not supported yet.

In particular the requirements to the integration accuracy are somewhat higher than for bond energies. It is recommended to use an integration grid (BeckeGrid) of quality "Good". If in doubt, a convergence test with respect to the integration accuracy is recommended.

8.2.3 Interacting Quantum Atoms (IQA)

Note: This part of the code IS NOT parallelized. Please make sure that when using the IQA functionality you are performing your calculation on a **single core**. Moreover, **symmetry is not allowed**, so please use the NOSYM option. Besides, it can only be used in a single-point calculation.

As an alternative to the Bickelhaupt-Baerends' bond energy decomposition scheme, one can use the interacting quantum atoms approach (IQA) developed by Angel Martín Pendás and coworkers in the framework of real-space partitions of the molecular space [*499* (page 936), *500* (page 936)]. In ADF2018, this energy decomposition scheme within Bader's quantum theory of atoms in molecules has been introduced.

In a wavefunction context, the IQA QTAIM partition of the molecular energy leads exactly to:

$$E = \sum_A \left(T^A + E_{Ne}^{AA} + E_{eeCl}^{AA} + E_{eeXC}^{AA} \right) + \frac{1}{2} \sum_{A \neq B} \left(E_{NN}^{AB} + E_{eN}^{AB} + E_{Ne}^{AB} + E_{eeCl}^{AB} + E_{eeXC}^{AB} \right)$$

The first four terms of the right-hand side in the equation above correspond to intra-atomic (also coined 'self') terms for each atom 'A': the atomic kinetic energy (T^A), the interaction energy between the nucleus and the electron density inside basin A (E_{Ne}^{AA}), and the repulsion energy between electrons in A, decomposed into classical electrostatic (E_{eeCl}^{AA}) and exchange-correlation contributions (E_{eeXC}^{AA}).

The last sum in the equation corresponds to the interactions between each atom pair 'AB' (bonded or not bonded by a bond path): the repulsion energy between nuclei in A and B (E_{NN}^{AB}), the attraction of the electrons in A by the nucleus in B (E_{NN}^{AB}) the attraction of the electrons in B by the nucleus in A (E_{Ne}^{AB}), and the repulsion energy between electrons in A with those in B, which can be split into a classical electrostatic contribution (E_{eeCl}^{AB}) and an exchange-correlation one (E_{eeXC}^{AB}).

Within a Kohn-Sham DFT framework, the implementation of this partition is not straightforward (see ref. [*501* (page 936)] for more details). In ADF2018 a partial energy decomposition limited to **interatomic interactions** has been proposed. For any atom-atom pair 'AB', we evaluate:

$$E_{inter}^{AB} = E_{NN}^{AB} + E_{eN}^{AB} \left[\rho \right] + E_{Ne}^{AB} \left[\rho \right] + E_{eeCl}^{AB} \left[\rho \right] + E_{eeX}^{AB} \left[\left\{ \psi_i^{KS} \right\} \right]$$

where $E_{eeX}^{AB} \left[\left\{ \psi_i^{KS} \right\} \right]$ corresponds to the full exact exchange ('HF-like') interacting energy between the two atoms. Tognetti and Joubert have shown that despite underlying approximations, this expression can be safely used at least for semi-quantitative purposes [*502* (page 936)].

To calculate these atom-atom interactions, one must add the IQA block key with two optional subkeys.

```
IQA
  PRINT {normal} {verbose}
  ATOMSTODO N1 N2 {...} Nk
END
```

PRINT normal (default): three energetic terms are printed for each atom pair. The first one corresponds to the *total* interaction energy between the two atoms (E_{inter}^{AB}). This energy is then split into two contributions: the 'covalent part' corresponds to the exchange energy between the two atomic basins ($E_{eeX}^{AB}\left[\left\{\psi_i^{KS}\right\}\right]$), while the 'ionic one' is the sum of all remaining electrostatic contributions ($E_{NN}^{AB} + E_{eN}^{AB} + E_{Ne}^{AB}\left[\rho\right] + E_{eeCl}^{AB}\left[\rho\right]$).

PRINT verbose all detailed contributions are printed for each atom pair.

ATOMSTODO A list of integers designing a subset of atoms for which atom-atom interactions are calculated. By default, the program calculates interatomic interactions for each pair of atoms. We strongly recommend to use this option to calculate these interactions for a small subset of atoms.

8.3 Localized Molecular Orbitals

ADF provides the Boys-Foster method and Pipek-Mezey method for localization of Molecular Orbitals [*122-123* (page 914), *428* (page 932)]. This implies a unitary transformation of the occupied molecular orbitals as computed in the SCF procedure, with the objective to obtain a (transformed) set of orbitals that represent exactly the same charge density but with molecular orbitals that are more localized in space than the original MOs.

The goal of orbital-localization lies in analysis: the localized orbitals provide an easier-to-interpret picture. The localized molecular orbitals can be visualized with the ADF-GUI.

Orbital localization procedures require a measure of the localization of the orbitals which can then be optimized in the space of the allowed unitary transformations. Methods advocated in the literature differ in the definition of this measure. The Boys-Foster method minimizes the mean extension of the occupied orbitals around their center of gravity; see the literature for details. The Pipek-Mezey localization maximizes the sum of orbital-dependent partial charges on the nuclei, see literature for details. Both Boys-Foster localization method and Pipek-Mezey localization method have been implemented in ADF, see Ref. [*428* (page 932)].

Occasionally it is useful to apply the localization only to a subset of the MOs, with the objective to expose certain features better. This is accomplished by performing the localization in a number of distinct steps, where at each step the localization is restricted by keeping a subset of the MOs frozen. A case is worked out in the Examples document.

The computation of localized orbitals is controlled with the block-type key. By default (if the key is not supplied in input) no orbital localization is carried out.

```
LOCORB {nopop store}
  {Criterion <BOYS|PM>}
  Spintype FrozenMOs
  Spintype FrozenMOs
  ...
end
```

nopop Specifies that no SFO population analysis is to be carried out on the localized MOs. By default this population analysis will be printed in the output file.

store Specifies that the transformation from MOs to localized MOs is stored on TAPE21.

Criterion <BOYS|PM> By default the Boys-Foster localization method is used if Criterion is not specified and in case 'Criterion BOYS' is specified. If 'Criterion PM' is specified, the Pipek-Mezey localization method is used.

Spintype Must be either alfa or beta (not case sensitive) and refers to spin-A and spin-B orbitals respectively. In a spin-restricted run beta records are meaningless and must not be used.

FrozenMOs A list (possibly empty) of integers, referring to a list of MOs from the SCF, and/or labels of irreducible representations. The integers and/or labels may be given in any order.

Each record Spintype FrozenMOs in the data block defines a localization *cycle* in which the localization procedure is carried out on all orbitals (of the indicated spin), except those indicated by the FrozenMOs.

For either spin at least one localization cycle is carried out. If no data record for that spin is found in the data block, a full localization is performed, without any MOs excluded.

The data block may be completely empty (but the record end must be supplied since the key is block-type) and would be equivalent with specifying two records, one for either spin, without any FrozenMOs:

```
LOCORB {nopop}
end
```

is equivalent with

```
LOCORB {nopop}
  alfa
  beta
end
```

The integers in FrozenMOs refer to an overall list of SCF MOs consisting of all valence MOs in each symmetry representation up to and including the highest non-empty one. So, when for instance in the first irrep MO #4 is the highest non-empty one and in the second irrep mo #2 is the highest non-empty one, then in the overall list the first 4 are the orbitals of the first irrep, the no.s 5 and 6 are from the second irrep, et cetera.

Each symmetry label in FrozenMOs collectively denotes in one stroke all molecular orbitals of that representation up to and including the highest occupied one (in that symmetry). The label may be the name of an irreducible representation or of a subspecies. In the former case all partner representations are denoted collectively. In an atom symmetry for instance, specifying P would be equivalent to P:x P:y P:z.

Note that if the final SCF has in any symmetry representation empty orbitals *below* the highest non-empty orbital in that symmetry - violating the Aufbau principle - then these empty orbitals are included in the above-defined overall list and hence a FrozenMOs specification is necessary, namely to avoid mixing MOs with different occupation numbers in the localization.

Note: It is imperative that in a particular localization cycle only MOs from the SCF are combined that have identical occupation numbers. If this is violated the program will carry out the localization without error message, but the results are incorrect in the sense that the density defined by the localized orbitals is *not* the same anymore as the SCF density.

So, if any of the MOs in the overall list defined above is not fully occupied (open shell, excited state, ...) you need to define precisely the localization cycles - localizing in each cycle only MOs with identical occupations and freezing all others - in order to obtain sensible results.

In the output file the localized MOs are printed as expansions in SFOs and (optionally) a population analysis is given, again in terms of the SFOs. Furthermore, each localized MO has associated with it an energy value and an occupation number. The energy is the expectation value of the Fock operator for the orbital. The occupation number is obtained as a weighted sum from the SCF MOs that were combined into the localized orbital. As mentioned before one should combine only SCF MOs with identical occupations into a localized orbital, in which case its occupation number will be the same. The printout of the occupation number of the localized orbital allows therefore a verification that a correct localization procedure has been carried out.

8.3.1 Perturbed Localized Molecular Orbitals

Perturbed localized molecular orbitals, correct to first order in an applied field, can be calculated in case of AORESPONSE. Can be used if the applied field changes the density in first order. Ref. [*428* (page 932)] describes this

functionality and the implementation in ADF in detail, and gives some examples. The perturbed localized molecular orbitals can be visualized with the ADF-GUI. In addition to the AORESPONSE and the LOCORB key one has to specify the PERTLOC key in order to calculate these perturbed localized molecular orbitals. The optional subkeywords of the block key PERTLOC are described below.

```
LOCORB ..
  ..
End
AORESPONSE
  ..
End
PERTLOC
  efield
  bfield
  alfa
  gprime
  beta
  <static|dynamic>
  <diag|fulltens>
End
```

efield The perturbation is an electric field (default).

bfield The perturbation is a magnetic field. Should be consistent with AORESPONSE.

alfa Analyze the static or dynamic polarizability.

gprime Analyze the G' (gyration) tensor, for optical rotation dispersion. Requires a frequency dependent perturbation field, with a frequency (omega) unequal to zero.

beta Analyze the optical rotation parameter beta. The relation to G' is beta = -G'/omega. The optical rotation parameter beta is calculated directly [*89* (page 912)] and has a well-defined static limit, i.e. omega can be zero or non-zero.

<static|dynamic> The static or dynamic (frequency dependent) subkeyword should be consistent with what is used in AORESPONSE. Static is default, should be used for a static field. Dynamic should be used for a frequency dependent perturbation field.

<diag|fulltens> Diag is default, which will only analyze the diagonal of the response tensor. If fulltens is specified the full tensor is analyzed.

8.4 Advanced charge density and bond order analysis

In addition to Mulliken charge analysis, ADF calculates several atomic charges that do not share the flaws of Mulliken (strong basis set dependence). The multipole-derived charge analysis exactly reproduces dipole and higher multipole moments of the molecule. Other charge analysis methods ('Voronoi deformation density' and 'Hirshfeld' provide atomic charges that agree well with chemical intuition. Nalewajski bond orders can be calculated and show good agreement with experimental trends and chemical intuition, even for transition metal compounds.

Note that the amount of data can be regulated with the keys PRINT, NoPrint, EPrint and Debug.

8.4.1 Charges, Populations, Bond orders

Mulliken populations

See the input key *EPRINT* (page 257). See also the *section on Mulliken populations* (page 266).

Hirshfeld charges, Voronoi deformation density

No special input key required. See also the *section on Hirshfeld charges, Voronoi deformation density* (page 266).

Multipole derived charges

No special input key required. See also the *section on MDC* (page 267).

Charge model 5 (CM5)

Charges calculated with CM5 activated by keyword

```
CM5
```

See also the *section on CM5* (page 267).

Bond orders

See also the *section on bond order analysis* (page 267).

Nalewajski-Mrozek Bond orders

Bond order analysis in ADF is activated by keyword

```
BONDORDER {tol=xxx} {printall}
```

By default bond order indices calculated by the Nalewajski-Mrozek [*148-152* (page 915)] method are calculated. There exist three alternative definitions of the valence and bond order indices within the Nalewajski-Mrozek approach. By default the values obtained from partitioning of Tr(P:math:*Delta*P) are calculated and printed in the output. For more information on alternative Nalewajski-Mrozek bond order indices see Results/Properties section (4.1).

tol=xxx The tol=xxx option specifies the threshold value for bond orders to be printed in the output (default=0.2).

printall The values calculated from all three versions of the Nalewajski-Mrozek approach are printed when the option printall is present; in addition the Gopinathan-Jug [*153* (page 915)] and Mayer [*140* (page 915)] bond order indices are calculated for comparison.

Present bond order analysis is based on SFOs. Symmetry used in the calculation should be NOSYM. For this reason the analysis may be used only if the symmetry in the calculation is NOSYM. The analysis may be used also for multi-atomic fragments, the fragment-fragment bond orders are printed in such a case. Note that in the present implementation all fragment types should be different.

Mayer Bond orders The Mayer bond orders are calculated and printed if the keyword:

```
EXTENDEDPOPAN
```

is included in the input. Next to the Mayer bond orders Mulliken atom-atom populations per l-value will be calculated and printed if this keyword is included in the input. Note that this keyword is not a subkey.

8.4.2 ETS-NOCV: Natural Orbitals for Chemical Valence

With the ETS-NOCV charge and energy decomposition scheme the deformation density is partitioned into the different components (σ, π, δ) of the chemical bond. The energy contributions to the total bond energy is calculated for each specific orbital interactions between fragments, giving insight in the orbital interactions also for non-symmetric molecules. The ETS-NOCV analysis offers a compact quantitative picture of the chemical bond, which is also qualitatively attractive to chemists.

The Natural Orbitals for Chemical Valence (NOCV) approach has been derived from the Nalewajski-Mrozek valence theory [*148* (page 915), *150* (page 915)]. From the mathematical point of view, each NOCV ψ_i is defined as an

eigenvector of the deformation density matrix in the basis of fragment orbitals.

$$\Delta P \Psi_i = \nu_i \Psi_i$$

Thus, the deformation density $\Delta\rho$ can be expressed in the NOCV representation as a sum of pairs of complimentary eigenfunctions (ψ_{-k}, ψ_k) corresponding to eigenvalues $-\nu_k$ and ν_k with the same absolute value but opposite signs:

$$\Delta\rho(r) = \sum \Delta\rho_k(r) = \sum \nu_k \left(-\Psi^2_{-k}(r) + \Psi^2_k(r)\right)$$

here, k goes over the pairs of NOCV's.

In the combined ETS-NOCV scheme the orbital interaction term ΔE_{orb} is expressed in terms of NOCV's as [*261* (page 922), *262* (page 922)]:

$$\Delta E_{\mathrm{orb}} = \sum \Delta E_k^{\mathrm{orb}} = \sum \nu_k \left(-F_{-k}^{\mathrm{TS}} + F_k^{TS}\right)$$

here, $-F_{-k}^{\mathrm{TS}}$ and F_k^{TS} are diagonal transition-state Kohn-Sham matrix elements corresponding to NOCV's with eigenvalues $-\nu_k$ and ν_k , respectively. The advantage of this expression is that usually only a few complimentary NOCV pairs significantly contribute to the total ΔE_{orb}. Another advantage of this approach is that not only can each $\Delta\rho_k(r)$ be visualized but there is also a well defined bonding energy contribution $\Delta E_k^{\mathrm{orb}}$ corresponding to it.

Remarks

The ETS-NOCV analysis is often not very useful when atomic fragments are used. No symmetry must be used in the final calculation, thus, use a Symmetry NOSYM keyword if your molecule is symmetric. The analysis is not completely implemented for meta-GGA's and meta-hybrids.

Improvements in ADF2012 to both the ETS and NOCV analysis with hybrids. ETS: Now the exact exchange contribution to the Pauli term is isolated and the contributions to the orbital term are divided among orbital symmetries. NOCV: The exact exchange contribution to the Fock operator is included when calculating energy contributions. These changes do not apply to meta-hybrids.

Usage

In order to perform the ETS-NOCV analysis, the following two keywords must be specified at the same time:

```
ETSNOCV
    {RHOKMIN rhokmin}
    {EKMIN ekmin}
    {ENOCV enocv}
End
PRINT {ETSLOWDIN | ETSLOWDIN-Unrestricted}
```

ETSNOCV The ETSNOCV keyword specifies thresholds for printing of NOCV-related information. All three arguments are optional and when all three are omitted only the NOCV's corresponding to eigenvalues abs(ν_k) $\leq$ 0.05 are included in the analysis.

    RHOKMIN The threshold for population analysis of each deformation density contribution in terms of individual SFO's.

    EKMIN The threshold for orbital interaction energy contributions corresponding to deformation density components originating from each NOCV-pairs.

    ENOCV The threshold for NOCV-eigenvalues.

PRINT {ETSLOWDIN | ETSLOWDIN-Unrestricted} Only one of the two PRINT options is supposed to be used to activate printing of ETS-NOCV results. The choice depends on the bonding situation.

    ETSLOWDIN If one is interested in a description of bonding between closed-shell molecular fragments, then 'PRINT ETSLOWDIN' keyword must be used. In such a case one set of NOCV's originating from the total deformation density matrix $\Delta P = (\Delta P_\alpha + \Delta P_\beta)$ will be printed out. See the example of carbene bonding between closed shell CH2 and Cr(CO)5.

ETSLOWDIN-Unrestricted If, however, one is interested in a description of bonding between open-shell molecular fragments then the 'PRINT ETSLOWDIN-Unrestricted' keyword must be used. In this case two sets of NOCV's originating from ΔP_α and ΔP_β will be printed out. See the example of CH3-CH3 bonding between two CH3 radicals with opposite spins. This option must also be used when one wants to analyze bonding in a molecule with unpaired electrons.

PRINT NOCVHirshfeld The NOCVs can be integrated per fragment using the Hirshfeld partitioning scheme. Using these integrals, one can distinguish between inter- and intra-fragment NOCVs. An inter-fragment NOCV has non-zero integral Hirshfeld ΔQ value and corresponds to a charge transfer between fragments. An intra-fragment NOCV has a small ΔQ value and corresponds to polarization of the fragments. When 'PRINT NOCVHirshfeld' is specified, the $\int \Delta \rho = -\Delta Q$ values per NOCV are printed to the output file in the "Hirshfeld partitioning" table at the end of the ETS-NOCV section.

8.4.3 Adfnbo, gennbo: NBO analysis

- `(PDF) NBO manual`

Dr. Autschbach, SCM, and Prof. Weinhold have collaborated to prepare a simple in put file generator, called adfnbo, for the GENNBO program of Prof. Weinholds Natural Bond Orbital (NBO) package. In ADF2013 and later the NBO 6.0 version is supported http://nbo6.chem.wisc.edu.

The GENNBO executable is included in the ADF distribution and can be enabled via the license file for all those who buy an NBO manual from either the NBO authors or from SCM (info@scm.com). An extensive documentation of GENNBO is part of the NBO manual. The application of ADFNBO to frozen-core basis sets needs to be further tested. Usage can be found below and in the Examples Document.

Next a brief summary of the capabilities of GENNBO is given (by Prof. Weinhold). GENNBO implements most capabilities of the full NBO 6.0 program suite as described on the NBO website: http://nbo6.chem.wisc.edu These include determination of natural atomic orbitals (NAOs), bond orbitals (NBOs), and localized MOs (NLMOs), as well as the associated NPA (atomic charges and orbital populations) and NRT (resonance structures, weightings, bond orders) valence descriptors, for a wide variety of uncorrelated and correlated (variational, perturbative, or density functional) theoretical levels. GENNBO-supported options include all keywords except those explicitly requiring interactive communication with the host electronic structure system (viz., $DEL deletions, NEDA, NCS, NJC). The GENNBO program typically sits conveniently on the PC desktop, ready to analyze (or re-analyze at will, with altered options) the final results of a complex ADF calculation performed on a remote cluster.

GENNBO "communicates" with the original ADF calculation through an archive file (JOB.47 file, preserving all necessary details of the final density) that is initially generated by ADF and subsequently becomes the input file for GENNBO. The .47 file contains a standard $NBO ... $END keylist that can be edited with a standard word processor or text editor to include chosen NBO keyword options, just as though they might have appeared in the original input stream of an interactive ADFNBO run. The stand-alone GENNBO program therefore allows many alternative NBO analysis options to be explored at leisure, without costly re-calculation of the wave function.

In ADF2018 the NBO 6.0 version6 in the ADF distribution is updated to the March 2017 version, see also http://nbo6.chem.wisc.edu: "The 14-Mar-2017 distribution includes a number of other features of interest to general NBO users: The default NAO search algorithm has been modified to better preserve core-valence separation in rare cases where numerical near-degeneracies can lead to unphysical core-valence mixing. The numerical effects are generally negligible except where they become necessary. The "OLDNAO" keyword restores the legacy NAO algorithm for comparison purposes."

This new NAO method may change the NBO analysis of some properties (EFG, NMR) quite substantially, especially individual contributions.

Usage

ADF needs to write some data to file, which is done by including FULLFOCK, AOMAT2FILE, SAVE TAPE15, and SYMMETRY NOSYM in the input file for ADF. An all electron basis set is needed. Use the key 'spherical' in

the adfnbo input. A file named FILE47 is generated by adfnbo which is an input file for the general NBO program gennbo6. Thus the usage is like

```
$ADFBIN/adf <<eor
...
Basis
  ...
  Core None
End
FullFock
AOMat2File
Save TAPE15
Symmetry Nosym
eor

$ADFBIN/adfnbo <<eor
write
fock
spherical
end input
eor

$ADFBIN/gennbo6 FILE47
```

NBO analysis of EFG, NMR chemical shifts, NMR spin-spin coupling

For certain molecular properties it is possible to perform detailed analyses in terms of Natural Bond Orbitals (NBOs) and Natural Localized Molecular Orbitals (NLMOs). These features generally require a sequence of ADF and/or property code runs. An initial non-relativistic or scalar relativistic ADF run, followed by the generation of NBO and NLMO data, is required, and the resulting data files need to be present in subsequent property calculations, along with a keyword indicating that the NBO analysis is requested in the property module.

We have noted in the past some slight loss of numerical accuracy of the results after going through the various orbital transformations in the NBO - NLMO sequence. It is important that the user verifies in each case that the total contributions from the analysis are in agreement with the total calculated property, within the numerical integration accuracy limits. In order to assist the user with this, the analysis program always print the total analysis contributions, including small non-printed values.

Moreover, there appears to be a problem with the analysis of the Fock matrix in the NBO program in conjunction with ADF calculations. Therefore please do NOT use the Fock matrix second order perturbation theory analysis in NBO at this time. We will remove this disclaimer once the issue has been fixed. Applications of the NBO-NLMO property analysis codes have so far given no indication that the Fock matrix issue interferes with the analysis.

Important note: If properties are analyzed from within spin-orbit relativistic computations, the NBO/NLMO analysis is performed in terms of scalar (spin-free) relativistic orbitals, as detailed in the technical references. The results from these analyses are exact in the sense that they fully reproduce the final spin-orbit property result, and they allow to dissect the property in terms of more intuitive one component real scalar relativistic localized orbitals. Typically, the property analysis in a spin-orbit calculation involves contributions from unoccupied scalar NLMOs, whereas there are no such contributions if a non-relativistic or scalar relativistic property is analyzed.

Available properties for NBO analysis: EFG, NMR chemical shifts and NMR spin-spin coupling.

NBO analysis of EFG

EFGs: non-relativistic and scalar ZORA, in ADF/AORresponse. Requires initial ADF run with

```
AOresponse
 donothing
End
```

in order to generate orbitals that re equivalent to those generated in the subsequent ADF run where the EFG is calculated. Alternatively, simply calculate the EFG twice, once before the NBO generation step, and once afterward.

The next step (see below) is to create the NBOs and the required data files for the analysis. Afterward, in the second ADF run, use

```
Aoresponse
 efg NUC nbo
end
```

efg NUC nbo Here NUC is the number of the nucleus at which the EFG is to be computed (ADF internal atom ordering). Example: efg 1 nbo.

The threshold for printing the EFG-NBO contributions to output can be adjusted via the `Tresh` options. The default is 0.05, which means that only orbitals with absolute value contribution larger than 5% of the total EFG are printed. To increase the number of contributions printed, specify a smaller threshold. For example:

```
Aoresponse
   efg 1 nbo Tresh 0.01
end
```

In addition to the optional NBO analysis, the EFG program in AOResponse prints a Mulliken type analysis of the EFG principal components, and an analysis in terms of canonical MOs.

WARNING: The ordering of the principal components is lowest to highest including the sign. That is, we have $V11 \leq V22 \leq V33$. This does not conform to the usual convention of $|V11| \leq |V22| \leq |V33|$. Please make sure you select the right component for your analysis.

Example job: $ADFHOME/examples/adf/AlCl3_efgnbo. For an explanation of the output and a general usage tutorial, see [*327* (page 926)]. Further references and recommended citations, see [*328* (page 926)].

NBO analysis of NMR Chemical shift

An implementation is currently available for spin-orbit ZORA computations. If scalar ZORA calculations are to be analyzed, provide the input keyword

```
FAKESO
```

in the NMR input (outside of the 'nmr' keyword). If this feature is requested one should restrict the calculation to a single shielding tensor per NMR run. It would be good practice to check the results against regular NMR calculations where the analysis feature is not requested. No ZORA scaling is applied in the analysis results. The data should be equivalent to a regular computation in the NMR input with

```
NMR
 u1k best
 calc all
END
```

Depending on whether scalar or spin-orbit calculations are to be analyzed, the sequence of calculations is different:

scalar:

1. ADF, scalar ZORA, symmetry NOSYM, closed shell
2. generate NBOs and required data files for analysis

3. NMR with FAKESO and analysis keywords, use TAPE21, TAPE10 from step 1.

spin-orbit:

1. ADF, scalar ZORA, closed shell

2. generate NBOs and required data files for analysis

3. delete TAPE21, TAPE10, TAPE15

4. ADF, spin-orbit ZORA, symmetry NOSYM, closed shell

5. NMR with analysis keywords, using TAPE21, TAPE10 from step 4

In the NMR run, in addition to the NMR keyword, provide the following

```
analysis
 print 0.01
 canonical
 nbo
 components
end
```

The optional canonical keyword can be used independently from the NBO analysis features. It enables an analysis of the shielding in terms of the canonical MOs. The components keyword is optional and enables an analysis not only of the isotropic shielding but also of each principal component of the tensor. The print keyword selects printout of contributions relative to the total diamagnetic, paramagnetic. In the example, only contributions greater than 1% are printed. Set to zero to print ALL contributions.

Example job: $ADFHOME/examples/adf/CH4_nmrnbo. References [*329-331* (page 926)].

NBO analysis of NMR spin-spin coupling (J-coupling)

Non-relativistic, scalar ZORA, spin-orbit ZORA

The sequence of jobs is similar to those in the NMR section.

scalar or **non-relativistic**:

1. ADF, scalar ZORA or nonrel.

2. generate NBOs and required data files for analysis

3. CPL with analysis keyword, use TAPE21, TAPE10 from step 1.

spin-orbit:

1. ADF, scalar ZORA

2. generate NBOs and required data files for analysis

3. delete TAPE21, TAPE10, TAPE15

4. ADF, spin-orbit ZORA

5. CPL with analysis keyword, using TAPE21, TAPE10 from step 4

In the CPL run provide the following 'contributions' keyword to enable the analysis

```
nmrcoupling
   ... other options
   contributions 1E19 nbo
end
```

The numerical value selects a print threshold in SI units of T**2/J for the analysis. Increase the value to obtain less detail in the analysis. By default, 'contributions' triggers an analysis of the J-coupling in terms of canonical MOs. The nbo keyword enables in addition the NBO-NLMO analysis.

Please note that due to the history of how the program was developed the output from the scalar/nrel. analysis and from the spin-orbit calculations differs somewhat. The qualitative content is the same.

In scalar ZORA or non-relativistic CPL calculations without the SD term an orbital based analysis is only performed for the Fermi-contact mechanism. If you also need an analysis for the PSO and SD mechanisms but do not want to run a spin-orbit calculation with ADF please use the SD or NOSD keywords which will cause the spin-orbit branch of the CPL code to be used. In ZORA spin-orbit calculations the FC, SD, PSO, and cross terms are analyzed together by default. You can selectively switch them on or off in order to get individual mechanism analyses. The DSO mechanism is often negligible. An analysis tool for this mechanism has therefore not yet been developed.

Example job: $ADFHOME/examples/adf/CPL_CH3OH_NBO. References NMR spin-spin couplings with NBO analysis [*331-334* (page 926)]:

Generation of NBOs

How to generate the NBOs, NLMOs, and the data files needed for these calculations (step 2 below is step 2 in the examples above):

1. run ADF with scalar ZORA or non-relativistic options, and keep TAPE21 and TAPE15.

2.

```
# run adfnbo in WRITE  mode to create the gennbo input file FILE47
# and one of the required property analysis files, adfnbo.kf

$ADFBIN/adfnbo << eor
write
spherical
end input
eor


rm -f adfnbo.37 adfnbo.39 adfnbo.49 adfnbo.48
$ADFBIN/gennbo6 << FILE47

# run adfnbo in COPY mode to create the second property analysis
# file, adfnbo2.kf

$ADFBIN/adfnbo <<  eor
 spherical
 copy
end input
eor

# run adfnbo in READ mode: prepare locorb on TAPE21

$ADFBIN/adfnbo <<  eor
 spherical
 read
end input
eor

rm -f adfnbo.37 adfnbo.39 adfnbo.49 adfnbo.48

# keep the TAPE21 after this sequence in order to
# be able to plot the NBOs and NLMOs with adfview:
```

```
mv TAPE21 nbonlmo.t21

# clean up, keep adfnbo*.kf for any NBO property analyses.
```

8.4.4 QTAIM: Atoms in Molecules

One can calculate local and atomic properties using a real-space partition of the electronic density. It is based on the quantum theory of atoms in molecules (QTAIM) developed by Richard Bader [*455* (page 933)]. Another possibility for Bader analysis is to use the adf2aim utility with a third party program such as Xaim.

Local, atomic, and non-local properties

The QTAIM input block replaces the *Bader* keyword. The former *Bader Reactivity* option now corresponds to the new *ConceptualDFT* block (see below).

The QTAIM functionality encompasses two different but related features: topology analysis of the electron density gradient field [*363* (page 928)], and calculation of the condensed atomic properties by integrating various local functions over atomic basins [*228,229* (page 920)]. These calculation can also be used for relatively large systems (hundreds of atoms). The topology analysis is performed at all levels of the analysis and it produces a list of the molecule's electron density critical points and bond paths that can be visualized by the ADFview GUI module. Besides, the following properties at the critical points are calculated: electron density, its Laplacian and Hessian matrix, ellipticity, Jenkins' metallicity index [*456* (page 933)], reduced density gradient, DFT energy density variation rates [*457* (page 933)], and Abramov's local energy components [*458* (page 933)]. At the *Extended* analysis level the following condensed atomic properties are calculated: atomic charges, integral of the density Laplacian, dipole and quadrupole moments, Shannon and Renyi entropies, spatial distribution of the molecular orbitals over atomic basins. If the *Energy* keyword is specified, also the atomic energies [*453* (page 933)] are calculated. At the *Full* level, also the atom-atom matrix elements of the localization (LI) and delocalization (DI) indices [*462,463* (page 934)] are computed.

QTAIM

> **Type** Block
>
> **Description** This block is used to request a topological analysis of the gradient field of the electron density, also known as the Bader's analysis. If this block is specified without any sub-key, only local properties are calculated.

AnalysisLevel

> **Type** Multiple Choice
>
> **Default value** Normal
>
> **Options** [Normal, Extended, Full]
>
> **Description** Set the level of the QTAIM analysis: Normal - topology analysis and properties at the density critical points, Extended - same as Normal plus condensed atomic descriptors, Full - same as Extended plus non-local descriptors.

Energy

> **Type** Bool
>
> **Default value** False
>
> **Description** Calculate atomic energies. Requires an all-electron calculation (no frozen core), triggers the TotalEnergy and increases the [AnalysisLevel] to at least Extended.

AtomsToDo

> **Type** Integer List
>
> **Description** List of atoms for which condensed descriptors are to be calculated. By default all atoms are included.

Spacing

> **Type** Float
>
> **Default value** 0.5
>
> **Unit** Bohr
>
> **Description** Specifies spacing of the initial Cartesian grid when searching for critical points. It may be useful to specify a smaller value than the default if some critical points are missed. This will result in a more accurate but slower calculation.

Note that the accuracy of condensed atomic properties can be estimated by integrating the Laplacian of the electron density over a given basin, which, ideally, must vanish. The accuracy of the method can be improved by using a larger integration grid (NumericalQuality). Usually, the default grid is sufficient to achieve the accuracy of 10^{-3} a.u. (differences of milliHartree in the energies). Unfortunately, the convergence of the electron density Laplacian integral with the grid size is not monotonous. Therefore this type of Bader atomic property calculation should be considered for applications where computational efficiency is critical and moderate accuracy is sufficient.

Note: The *Energy* keyword requires calculation of the total energy so it has the same limitations as the *TotalEnergy* keyword. For example, it is not compatible with relativistic calculations. The electron density topology analysis, however, *is* possible for all-electron scalar-relativistic calculations as was shown in [*454* (page 933)].

Note: Atomic energies are correct only at equilibrium geometries (the virial theorem is used).

ADF2AIM

The ADF utility adf2aim (original name rdt21) developed by Xavi Lopez, Engelber Sans and Carles Bo converts an ADF TAPE21 to WFN format (for Bader analysis)

The program rdt21 is now called adf2aim and is part of the ADF package.

The WFN file is an input file for the third party program Xaim (see http://www.quimica.urv.es/XAIM for details), which is a graphical user interface to programs that can perform the Bader analysis. Usage of adf2aim can be found in the Examples Document.

Aromaticity index with QTAIM

The cyclic delocalisation of mobile electrons in two or three dimensions is probably one of the key aspects that characterise aromatic compounds. However, aromaticity cannot be measured directly by any physical or chemical experiment because it is not a well-defined magnitude. One can define indicators of aromaticity, some of which are implemented in ADF. The matrix of localisation/delocalisation indices (LI-DI) and the following from it Iring (ring index) and MCI (multi center index) aromaticity indices (see Ref. [*440* (page 932)]) can be computed for rings specified by the user. The rings are specified using the AROMATICITY input block, one line per ring. For example, for a benzene molecule where atoms 1-6 are carbon atoms the input would look like this:

```
AROMATICITY
  1 2 3 4 5 6
END
```

The program does not check whether the atoms in the AROMATICITY input block are connected so it is responsibility of the user to specify them correctly. Moreover, for the Iring index to be calculated correctly the atoms must be listed in the order they are connected in the ring.

Calculation of the aromaticity indices invokes the QTAIM analysis automatically and uses its results so all accuracy considerations for the QTAIM key above are also valid here. This feature requires that the calculation is performed without symmetry (Symmetry NOSYM).

Results of the calculation are printed to the output under the "Aromaticity Index" header. The first table presents the total number of electrons per atom and their breakdown in "localized" and "delocalized". Note: the total number of electrons per atom $N_{total} = N_{loc} + N_{deloc}/2$ since delocalized electrons are counted twice. The second table presents the Iring and MCI ring indices for each ring specified in the AROMATICITY block. These results (and the original LiDi matrix from which they are computed) are stored in the Properties section in the t21 file.

8.4.5 Conceptual DFT

Global, atomic, and non-local descriptors

The *ConceptualDFT* block replaces and extends the set of features behind the *Bader Reactivity* keyword found in the ADF versions before ADF2019. The frontier molecular orbital (FMO) approximation is used for all calculations. Orbital degeneracy is taken into account following Martinez [*491* (page 935)]. All these descriptors are described in Ref. [*503* (page 936)]. Which exactly descriptors are calculated depends on the *AnalysisLevel* key:

- *Normal:* global conceptual DFT descriptors, i.e. electronic chemical potential, electronegativity (its opposite), hardness and softness, hyperhardness [*509* (page 936)], electrophilicity index [*510* (page 936)], dissociation energies (nucleofuge and electrofuge, see Ref. [*511* (page 936)]), electrodonating and electroaccepting powers [*512* (page 937)] and the net electrophilicity [*513* (page 937)]. Also, new global descriptors called *global dual descriptors* $\Delta f+$ and $\Delta f-$ are calculated [*514* (page 937)], which correspond to the spatial integral of the positive (electrophilic) or negative (nucleophilic) regions of the original dual descriptor, respectively. These descriptors allow for comparing total reactivity of molecules, in contrast with Fukui functions or original dual descriptor that integrate over the whole space to 1 or 0 for all molecules.

- *Extended:* both global and atomic properties, including condensed (QTAIM) descriptors in both canonical and grand canonical ensembles. More precisely, the following properties are computed: condensed Fukui functions [*483* (page 935)], dual descriptor [*460* (page 934), *484* (page 935)] and some of their composite functions [*515-517* (page 937)], as well as condensed local electrophilicity and nucleophilicity [*518* (page 937)].

- *Full:* all properties are calculated, including non-local ones, such as the condensed linear response function [*464* (page 934)] and the softness kernel [*519* (page 937)].

Please note that atomic electronegativities [*461* (page 934)] are calculated separately with an optional *Electronegativity* keyword. In contrast with other properties, it requires all-electron basis sets and triggers the *TotalEnergy* keyword.

ConceptualDFT

> **Type** Block
>
> **Description** Conceptual DFT Properties
>
> **AnalysisLevel**
>
>> **Type** Multiple Choice
>>
>> **Default value** Normal
>>
>> **Options** [Normal, Extended, Full]
>>
>> **Description** Set the level of the ConceptualDFT analysis: Normal - global descriptors only, Extended - both global and condensed (QTAIM) local descriptors, Full - all descriptors including non local ones.

Electronegativity

> **Type** Bool
>
> **Default value** False
>
> **Description** Calculate atomic electronegativities. Requires an all-electron calculation (no frozen core), triggers the TotalEnergy and increases the [AnalysisLevel] to at least Extended.

AtomsToDo

> **Type** Integer List
>
> **Description** Define a subset of atoms for which properties are calculated. If the [Domains] block is present then this list specifies which atoms are used to define the domains bounding box.

For more information about *Conceptual DFT*, we suggest the reading of these excellent reviews: Refs. [*520* (page 937)] and [*459* (page 934)].

Domains of the dual descriptor

In a recent paper [*504* (page 936)], Tognetti et al. proposed a real-space partitioning scheme of the dual descriptor (DD) into chemically meaningful *reactivity domains*. Basically, a predominantly electrophilic (nucleophilic) DD domain constitutes a region of space where this function remains positive (negative). They can be visualized using the GUI and quantitative information can be obtained using the additional *Domains* key block.

Note: This analysis requires that the calculation is performed without symmetry (Symmetry NOSYM)

ConceptualDFT

> **Type** Block
>
> **Description** Conceptual DFT Properties

Domains

> **Type** Block
>
> **Description** Calculate integrated properties for the domains (same sign) of the dual descriptor.

Border

> **Type** Float
>
> **Default value** 7.0
>
> **Unit** Bohr
>
> **Description** Set the extent of the Cartesian grid. Extent is the distance between a face of the grid's bounding box and the most outlying atom in the corresponding direction. If the [AtomsToDo] key is present, the bounding box is created around the specified atoms.

Display

> **Type** Float
>
> **Default value** 0.005
>
> **Description** Domains for which the integrated DD value is smaller (in magnitude) than the specified value are omitted from the printed output.

Enabled

> **Type** Bool

Default value False

Description This key is used by ADFinput enable the feature. It does not have any other function.

Ensemble

Type Multiple Choice

Default value Canonical

Options [Canonical, GrandCanonical]

Description Statistical ensemble for DD domains. Canonical: DD values are calculated using the statistical canonical ensemble. GrandCanonical: DD values are calculated using the statistical grand canonical ensemble. The grand canonical DD corresponds to (S^2 f(2) - (gamma/eta^3) f^0), where f(2) is the canonical DD, gamma and eta - the hyper-hardness and hardness of the chemical system, respectively, and f^0 is the mean Fukui function. This statistical ensemble is a natural choice when comparing two chemical systems with a different number of electrons.

Radius

Type Float

Default value 0.0

Description This option adds a sphere around each nucleus, excluding all points inside it. This can help to separate domains around an atom or to exclude core electrons. Be careful when using this option. In particular, the radius of the sphere should exceed two or three times the [Spacing] value to be effective. By default, no spheres are added.

Spacing

Type Float

Default value 0.1

Unit Bohr

Description Specifies spacing (distance between neighboring points) of the rectangular Cartesian grid used when searching for DD domains. It may be useful to specify a smaller value (or increase the size of the grid, see [Border] key) if a substantial part of the electronic density is accounted for.

Threshold

Type Float

Default value 0.001

Description Arbitrary value of dual descriptor used to separate DD domains (values below this threshold are ignored).

8.4.6 adf2damqt: DAMQT interface

Interface (adf2damqt) to the 3rd party DAMQT 2.0 package [*397* (page 930)], which can analyze the electron density in molecules, and related quantities. DAMQT includes the method of deformed atoms in molecules (DAM). The DAMQT 2.0 package is described in Ref. [*397* (page 930)]. In the ADF calculation TAPE15 and TAPE21 need to be saved. The executable adf2damqt can be run with up to three optional arguments. If no argument is supplied, "ADF" is chosen as default root name (fname) for files generated by the interface, and files containing electron density matrix (fname.den) and molecular orbitals (fname.SLorba and, eventually, fname.SLorbb) will be created in a format suitable to be read by DAMQT.

```
$ADFBIN/adf << eor
...
SAVE TAPE21 TAPE15
eor
$ADFBIN/adf2damqt {fname {SPIN} {NOORBITALS}}
```

fname If a specific name is desired for the files (fname), it must be supplied as first optional argument, provided that the name does not coincide with any of the two additional options SPIN and NOORBITALS.

SPIN SPIN: for storing spin density matrix in fname.den file (instead of total electron density, which is the default).

NOORBITALS NOORBITALS: to prevent generation of files with molecular orbitals (by default orbitals are generated).

SPIN and NOORBITALS are case insensitive and can be given in any order (but always after optional fname when required).

8.4.7 FOD: fractional orbital density

Following the analysis method by Grimme and Hansen (Ref. [*398* (page 930)]) a fractional occupation number weighted electron density (FOD) can be plotted with ADFview. The scalar field is obtained by finite-temperature DFT calculations with pre-defined electronic temperature T_{el}, typically T_{el} =5000 K, and for hybrids T_{el} =20000*(amount of HF exchange)+5000, see Ref. [*398* (page 930)]. This analysis tool can be used to determine whether static electron-correlation effects are important.

```
$ADFBIN/adf <<eor
...
OCCUPATIONS ELECTRONICTEMPERATURE=5000
eor
```

The NFOD (integrated FOD) is written in the output of ADF. The FOD can be visualized with ADFview. The contour surface value in ADFview should be set to 0.005 $(e/(bohr)^3)$.

8.5 Controlling printed Output

The amount of printed output is regulated with the keys Print, NoPrint, EPrint and Debug. (No)Print and Debug are simple keys, EPrint is a block type key.

Many print options pertain to debugging situations and are included here only for completeness. This section is intended to give a survey of all possibilities. Some items may be mentioned again in other sections where the subject of a particular print switch is discussed.

8.5.1 Print / NoPrint

```
PRINT Argumentlist
Print Argumentlist
NoPrint Argumentlist
```

Argumentlist A sequence of names separated by blanks or commas. The keys Print and NoPrint may occur any number of times in the input file. The names in the argument list may refer to various items. For some of them printing is normally on, and you can turn them off with NoPrint. For others the default is not printing; use Print to override that.

Follows a list of the recognized items that are applicable in the argument lists, with a short explanation and defaults. Item names must be used exactly as given in the table - abbreviated or elongated forms will not be recognized - but they are not case sensitive.

Items

$ALL Default: No

> Turns on all print options. This will not be affected by any additional noprint instructions. Be careful: this generates a large amount of output. To be used only for debugging purposes.

Atdist Default: No

> Inter-atomic distance matrix at each new geometry (in an optimization)

Bas Default: Yes

> General control of output related to elementary basis functions (bas).

BlockCheck Default: No

> Intermediate data during the determination of the block length. (see Blocks)

Blocks Default: No

> Numerical integrals, consisting of loops over large numbers of points, are split up in loops over blocks of points. The block length is determined by the available amount of workspace. Given this amount, the maximum block lengths, according to memory usage in a few relevant routines, are computed (and printed with this print option) and used to impose upper bounds on the block length actually use.

Character-Table Default: No

> Table of characters for the irreducible representations of the point group symmetry.

Computation Default: Yes

> Reports progress of the computation, with (concise) info about each SCF cycle and each Geometry update in an optimization.

Core Default: No

> Description of the frozen core: frozen core expansion functions (corbas) and the expansion coefficients for the frozen orbitals. This printing can only be activated if Functions is also *on*, otherwise it is ignored.

CoreOrt Default: No

> The valence basis set contains auxiliary Core Functions. They are not degrees of freedom but are used solely to ensure orthogonalization of the valence set to the frozen Core Orbitals. The orthogonalization coefficients and some related overlap matrices are printed.

CoreTable Default: No

> Internally the charge density and potential of the atomic frozen cores are processed as tables with values for a sequence of radial distances. A few initial and a few final values from these tables are printed, along with the (radial) integral of the core density, which should yield the number of core electrons.

EKin Default: No

> At the end of SCF: Kinetic energy of each occupied MO.

EndOf Default: No

> Flags the exit from a few major routines, with cpu times used in these modules. Primarily a debug tool.

EPauli Default: Yes

The repulsive Pauli term in the bonding energy (also called exchange repulsion) with its decomposition in density functional (lda and nl) and Coulomb terms.

Fit Default: Yes

General control of output related to the density fitting.

Fmat Default: No

Fock matrix computed at each cycle of the SCF.

FmatSFO Default: No

Fock matrix (and overlap matrix) in the basis of symmetrized fragment orbitals (SFOs). This option requires the FULLFOCK and ALLPOINTS keyword to be present in the input. The matrix is printed only at the last SCF cycle. Use 1 iteration in the SCF for the Fock matrix at the first SCF cycle.

ForceConstants Default: Yes

Force constants matrix (Frequencies run only)

FreqHess Default: No

matrix of force constants (Frequencies run) after each applicable step in its processing: transformation from/to Cartesian and Z-matrix coordinates, symmetrizations, ...

Frag Default: No

General control of output related to build-molecule-from-fragments.

Functions Default: Yes

List of employed Slater-type exponential basis functions and fit functions.

Gradients Default: No

detailed info of computed energy gradients (in optimization runs)

Group-Operators,Default: No 3*3 matrices of point group symmetry operators, with the axis and angle of rotation

HessEig Default: No

Eigenvalues of the Hessian in each cycle of a Geometry Optimization. The print-out in the intermediate cycles is suppressed if output of updated coordinates etc. is turned off (see the EPRINT subkey Repeat (option GeoStep).

Idfree Default: No

List of free atomic coordinates with indication whether they are optimization coordinates (this info is also contained in the output of new atomic coordinates at each step of an optimization)

Inertia Default: No

Warning message in the log file in case of zero product of moments of inertia (this may correctly be the case for certain molecules)

Inputkeys Default: No

List of keys that were specified in input, together with some of the associated data. The list is printed directly after the echo of the Input File, before the header with ADF program information. A few special keys will not be echoed: (No)Print,(No)Skip, Allow.

Irrep-Matrices Default No

Irreducible representation matrices

Logfile Default: Yes

> At the end of the calculation a copy of the log file is appended to standard output

low Default: No

> Construction of the LOW basis from the elementary BAS functions and from the SFOs: combination coefficients

lowMO Default: No

> MOs are printed in the LOW (Lowdin) representation, in the RESULTS section

OvlBAS Default: No

> overlap matrices processed during the construction of the LOW basis. Only printed in case OLDORTHON is used in input.

Parser Default: No

> Most input records are echoed twice (at the very beginning of output). First the original version, then the parsed version in which expressions have been replaced, redundant blanks removed, etc. The parsed version is what the program really uses as input. Comment blocks, and function definitions (in define blocks) are not parsed, and are not affected by this switch. If the print switch is off only the original, non-parsed input record is echoed in output. This print switch affects only the part of input after its occurrence.

Pmat Default: No

> The density matrix (in Lowdin representation) in each cycle of the SCF.

QMpot Default: Yes

> At the end of the SCF for each atom the electrostatic potential at its nucleus (excluding its own contribution of course).

RedCrd Default: No

> Redundant Coordinates used in the construction of the initial - force field derived - Hessian

RedCrdBonds Default: No

> Atom-atom bonds determined for the construction of the initial Hessian

RedCrdH Default: No

> Hessian in the redundant coordinates representation

SCF Default: Yes

> Controls the information about progress of the SCF procedure. Applies only if the print switch computation is on.

sdiis Default: No

> Expansion coefficients applied by the DIIS procedure during the SCF.

sdiismat Default: No

> Turns on sdiis(see above) *and* prints the *error vector* constructed by the DIIS routine (this is the commutator of the Fock matrix and the Density matrix). This is used to determine the DIIS expansion coefficients and to assess convergence.

SFO Default: depends on system size

> General control of SFO-related output (if SFO subkey of key EPRINT is used). If turned off, (almost) all such output is suppressed. If on, such printing is controlled by the eprint subkey SFO. The default depends on the system size: if the number of primitive STOs < 1000, the default is Yes, else No.

Smat Default: No

> Overlap matrix of BAS functions.

Smearq Default: No

> Smear parameter - if and when applied - used in the determination of electronic occupation numbers for the MOs, with details of how it works out at every cycle of the SCF. For debugging purposes.

SpinOrbit Default: No

> detailed information about how double-group symmetry representations are related to the single group representations

Tails Default: No

> In each block of integration points (see Blocks) the evaluation of (Slater-type) exponential functions (basis, fit) is skipped when the function has become negligible for all points in that block due to the distance of those points from the atom where the function is centered. The relative savings due to this distance screening is printed at the first geometry cycle (use debug for printing at all cycles).

TechPar Default: Yes

> Technical parameters such as maximum vector length in vectorized numerical integration loops, SCF and Geometry Optimization strategy parameters.

Timing Default: No

> Print out of more timing info (in particular with respect to performance of the parallel version of ADF) than is provided by the standard Timing Statistics tables at the end of each output.

TimingDetail Default: No

> Similar, but more details.

TimingTooMuchDetail Default: No

> Similar, but even worse.

Workspace Default: No

> Statistics of calls to the Workspace Manager (memory management).

Arguments for the keys PRINT and NOPRINT.

For print switches Frag, Fit, Freq, Geostep, Numint, Repeat, SCF, SFO, TF, see the key EPRINT below.

8.5.2 Debug

The key DEBUG is used to generate extensive output that is usually only relevant for debugging purposes. It operates exactly like the PRINT key but there is no converse: nodebug is not recognized; it would be irrelevant anyway because by default all debug print switches are off.

A list of the possible items for the DEBUG key is given below.

All items of the print list can also be used with the debug key. If they are not mentioned in table III, the meaning is the same as for the print key, but the corresponding output may be generated more often, for instance at every SCF cycle rather than at the last one only.

Table 8.1: Table III. Arguments for the print key DEBUG. All debug switches are by default off.

Item	Explanation
Basis	Construction of the orthonormal LOW basis from elementary (BAS) and fragment (FO) basis.
Core	Core Orthogonalization procedure
Ekin	Kinetic energy matrices. (compare the print switches EKIN)
Fit	Construction of the symmetry adapted fit functions
Fitint	Construction of integrals used in the Fit procedure.
Freq	Force matrices processed in the computation of frequencies:
	Cartesian and internal representation, before and after symmetrization,
	etc. (as far as applicable).
GeoStep	Geometry optimization procedure. All relevant items.
Hess	Complete eigensystem of the Hessian during geometry optimizations.
NumInt	Numerical integration. Very extensive output
	(including the coordinates and weights of all generated points).
Pmat	P-matrix (density matrix) during SCF
	and in the ETS analysis program in the BAS representation.
Rhofih	Computation of fit coefficients during the SCF.
SCF	Extensive output during the SCF procedure about many different items.
	See also EPRINT, subkey SCF.
SDIIS	All data concerning the DIIS as used during the SCF.
	See ERPRINT, subkey SDIIS.
TransitionField	The Transition State procedure to compute and analyze certain terms in the bonding energy.
	The distinct components, the involved transition field Fock matrices, etc.

8.5.3 Eprint

The key EPRINT is an extended version of the (no)print key, employed for print switches that require more specification than just off or on.

Contrary to what is the case for the keys print and noprint, the key EPRINT must occur only once in the input file; any subsequent occurrences are incorrect and ignored or lead to abort.

```
EPRINT
  subkey
  subkey
  ...
end
```

subkey A subkey-type structure: it consists of a keyword followed by data, so that it functions as a simple (sub)key, *or* it is a keyword followed by a data *block* which must then end with the word subend.

The subkeys used in the EPRINT data block are called Eprint keys. A complete list of them is given below. All available EPRINT keys are discussed in the schemes below. The enclosing records EPRINT and end are omitted in these schemes.

Table 8.2: Table IV. List of EPRINT subkeys.

EPRINT subkeys	Subject
AtomPop (page 266)	Mulliken population analysis on a per-atom basis
BASPop (page 263)	Mulliken population analysis on a per-bas-function basis
Eigval (page 263)	One-electron orbital energies
Fit (page 258)	Fit functions and fit coefficients
Frag (page 259)	Building of the molecule from fragments.
FragPop (page 263)	Mulliken population analysis on a per fragment basis
Freq (page 259)	Intermediate results in the computation of frequencies (see debug: freq).
GeoStep (page 260)	Geometry updates (Optimization, Transition State, ...)
NumInt (page 260)	Numerical Integration
OrbPop (page 260)	(Mulliken type) population analysis for individual MOs
OrbPopEr (page 261)	Energy Range (ER) in hartree units for the OrbPop subkey
Repeat (page 261)	repetition of output in Geometry iterations (SCF, optimization, ...)
SCF (page 261)	Self Consistent Field procedure
SFO (page 262)	Information related to the Symmetrized Fragment Orbitals and the analysis
TF (page 262)	Transition Field method.

8.5.4 Eprint subkeys vs. Print switches

Several EPRINT subkeys are merely shortcuts for normal (no)print switches. All such simple subkeys are used in the following way:

```
EPRINT
  ESUBKEY argumentlist
END
```

Esubkey One of the following EPRINT subkeys: Fit, Frag, GeoStep, NumInt, Repeat, SCF, sdiis, SFO, TF, Time.

argumentlist A sequence of names, separated by delimiters. Each of these names will be concatenated with the esubkey and the combination will be stored as a normal print switch. Example: Frag rot, SFO will be concatenated to fragrot and fragsfo and both will be stored as print switches. All such combinations can also be specified directly with the key PRINT. The example is therefore exactly equivalent with the input specification: print FragRot, Fragsfo****

If any of the names starts with the two characters no, the *remainder* of the name will be concatenated with the EPRINT, but now the result will be stored and treated as a noprint switch. Items that are on by default can in this way be turned off. Example:

```
EPRINT
  FRAG noRot Eig
END
```

This turns Rot *off* and Eig *on* for the EPRINT subkey Frag. Equivalent would be:

```
NOPRINT FragRot
Print FragEig
```

Follows a description of all simple EPrint subkeys: **Fit**

The subkey fit controls output of how the elementary fit functions are combined into the symmetric (A1) fit functions. It controls also printing of the initial (start-up) and the final (SCF) fit coefficients.

```
EPRINT
  FIT list
END
```

list A list of items, separated by blanks or commas. The following items are recognized: Charge, Coef, Comb.

Charge The amount of electronic charge contained in the fit (start-up), total and per fragment.

Coef The fit coefficients that give the expansion of the charge density in the elementary fit functions.

Comb The construction of the totally symmetric (A1) fit function combinations from the elementary fit functions.

By default all options are off.

Frag

The subkey frag controls output of how the molecule is built up from its fragments.

```
EPRINT
  FRAG list
END
```

list A list of items, separated by blanks or commas. The following items are recognized: Eig, Fit, Rot, SFO.

Eig The expansion coefficients in elementary functions (bas) of the fragment Molecular Orbitals as they are on the fragment file.

Rot The rotation (and translation) required to map the master fragment (i.e. the geometrical data on the fragment file) onto the actual fragment which is part of the current molecule. N.B.: if eig and rot are both *on*, the rotated fragment orbitals are printed also.

Fit The fit coefficients that describe the fitted charge density of the fragments after the rotation from the *master* fragment on file to the actual fragment. These are the molecular fit coefficients that are used (by default) to construct the total molecular start-up (fitted) charge density and hence the initial Coulomb and XC potential derived from it.

SFO The Symmetry-adapted combinations of Fragment Orbitals that are used in the current calculation. This feature ensures that the definition of the SFOs is printed. This will happen anyway whenever the EPRINT subkey SFO itself is activated. By default all options are off. Remark: SFO analysis in a Spin-Orbit relativistic calculation is implemented only in the case there is one scalar relativistic fragment, which is the whole molecule.

Freq

Controls printing of Force matrices and a few more data that are intermediate results in the computation of frequencies after all coordinate displacements have been carried out.

```
EPRINT
  FREQ list
END
```

list contains any of the items SymCoord, DMuRot, Hess.

SymCoord print the Symmetry Coordinates, both as they are generated, and some related info in their processing later on. The Symmetry Coordinates are symmetry-adapted combinations of Cartesian displacements, with the pure translations and rotations projected out.

dmuRot info about generating Dipole-Derivative information in transformation between Cartesian and internal coordinate representation as regards the rotational aspects.

Hess processing of the completed matrix of force constants, symmetrization, transformation to other coordinates.

By default all options are off.

Controls output concerning the geometry update method, parameters, energy gradients, etc. It plays no role in a SinglePoint calculation. **GeoStep**

```
EPRINT

GEOSTEP list
END
```

list A list of items, separated by blanks or commas. The following items are recognized: Energy, GradientTerms, Gradients, Upd.

Energy summary of the (bonding) energy and its components as computed in the geometry update procedure.

Gradients Energy gradients on the free variables. These may be all or some of the Cartesian or the Z-matrix coordinates, depending on the case.

GradientTerms The decomposition of the gradients in computed terms, as described in the thesis of L.Versluis [*7* (page 907)].

Upd parameters used and adapted in the geometry update procedure.

By default Gradients, Upd are on, the other items off.

NumInt

Output related to the numerical integration procedure: parameters, generated points, tests on the accuracy of the generated scheme, etc.

```
EPRINT
  NUMINT list
END
```

list A list of items, separated by blanks or commas. The following items are recognized: All, Geo, Ovl, Par, Pnt, Res, Sym, Test.

All includes all other options and prints in addition the coordinates and weights of all generated points.* This can be a lot of output!*

Geo geometric data such as boundary planes around the molecule, as they are computed and used in the program section where the point grid is generated.

Ovl numerically integrated are the auto-overlaps of symmetry-adapted combinations of elementary basis functions SBAS. The deviations from the analytically computed values is printed. The test option, see below, yields a summary of these data: the maximum error and the root-mean-square error.

Par employed precision parameters, atomic spheres radii etc.

Pnt the generated *numbers* of points in each of the subregions processed in the point-generating procedure.

Res results as regards the total number of points, the sum-of-weights and the partitioning of the points in blocks (for segmented vectorization).

Sym the symmetry operators that are computed directly from the coordinates (irrespective of the input Schönfliess symbol) and that are used to construct thec numerical integration grid in a symmetric fashion.

Test a few external tests are performed after the grid has been generated, such as the numerical integration of the sum-of-fragment densities. See also the norms option.

By default Res and Test are on, the other options off.

OrbPop

Specifies that (Mulliken type) population analysis should be printed for individual MOs, both on a per-SFO basis and on a per-bas function basis. The format of the subkey is as follows:

```
EPRINT
  ORBPOP TOL=X Nocc Nunocc
  SUBEND
END
```

X is the threshold for the SFO coefficient value to include in the listing for the per-SFO analysis. Nocc is the number of the highest occupied and Nunocc is the number of the lowest unoccupied orbitals to analyze.

OrbPopEr Specifies the energy range for the MOs to which the OrbPop key applies. The default range is from -0.7 below the HOMO to 0.2 Hartree above the LUMO. Usage:

```
EPRINT
  OrbPopER minEn maxEn
END
```

where minEn and maxEn are both in Hartree, and have the defaults just specified. In order to get information on many more orbitals, simply specify a large negative value for minen and a large positive value to maxen. **Repeat**

Control the repetition of output in Geometry iterations: optimization, computation of frequencies, transition state search.

```
EPRINT
  Repeat list
END
```

list contains one or more of the following items: NumInt, SCF.

NumInt Output from the numerical integration procedure, like parameters, numbers of points generated, test data is controlled by the *numint* subkey (see below). The *repeat* subkey controls whether the output is repeated for all geometries (if the flag is on) or only for the first (if the flag is off). Some concise info is produced (repeatedly) anyway if the print switch computation is on.

SCF Controls similarly the SCF output, like population analysis and orbital eigenvalues. If the flag is on, these items are printed at the last SCF cycle in every geometry, otherwise only at the last (in case of an optimization, not in case of a Frequencies calculation).

By default both options are off.

SCF

Output during the SCF procedure.

```
EPRINT
  SCF list
END
```

list is a list of items, separated by blanks or commas. The following items are recognized: Eigval, Eigvec, Err, Fmat, Keeporb, MOPop, Occ, Pmat, Pop, Start.

Eigval Eigenvalues of the one-electron orbitals at the last SCF cycle. In a run with multiple SCF runs (Geometry Optimization,..) this printing occurs only for the last SCF procedure. See also the eigval subkey of EPRINT. (Use debug or the *repeat* subkey of EPRINT to get output on *all* cycles).

Eigvec MO eigenvector coefficients in the BAS representation. Only printed on the last SCF cycle.

Err SCF error data which are checked for convergence. By default this takes effect after cycle 25 of the SCF. If the key is set it takes effect at the first cycle. Optionally one may type ErrN,where n is an integer (written directly after Err without a blank in between), in which case the key takes effect at cycle n.

Fmat Fock matrix in the low representation.

Keeporb If the KeepOrbitals option is activated (see the key SCF), output is generated whenever this option actually results in a change of occupation numbers as regards the energy ordering.

Occ concise output of SCF occupation numbers on last SCF cycle if no eigenvalues are printed (see: Eigval).

moPop Mulliken populations in terms of the elementary basis functions (bas), per MO, for input-specified MOs (see the EPRINT subkey *orbpop*)

Pmat Density matrix

Pop General control of bas Mulliken populations. This supervises all printing (whether populations are printed or not) according to the EPRINT subkeys *atompop, fragpop, orbpop* (the latter only as regards the bas population analysis at the end of the SCF procedure).

Start Data pertaining to the *first* SCF cycle (of the *first* SCF procedure, in case of an optimization; use *repeat* to get this for *all* SCFs).

By default Eigval, Keeporb, Occ, and Pop are on, the others off. **SFO**

Information pertaining to the use of Symmetrized Fragment Orbitals (for analysis purposes).

```
EPRINT
  SFO list
END
```

list A list of items, separated by blanks or commas. The following items are recognized: eig, eigcf, orbpop, grosspop, fragpop, ovl.

Eig The MO coefficients in terms of the SFOs.

Eigcf idem, but now also containing the coefficients pertaining to the CoreFunctions.

OrbPop population analysis of individual orbitals. The orbitals analyzed are set with the EPRINT subkey *orbpop*.

GrossPop Gross populations of the SFOs, split out in symmetry representations. GrossPop is automatically turned on when OrbPop is activated.

FragPop Population analysis on a per-FragmentType basis. This analysis does in fact not depend on the SFOs (ie, the result does not depend on how the SFOs are defined), but the computation of these populations takes place in the SFO-analysis module, which is why it is controlled by the SFO print option. FragPop output is given per orbital when OrbPop is activated, per symmetry representation when GrossPop is activated, and as a sum-over-all-orbitals-in-all-irreps otherwise (if FragPop is active).

Ovl Overlap matrix of the SFO basis, separately for each symmetry representation.

By default orbpop is on, the other options off.

In a Spin-Orbit calculation the SFO analysis is not yet implemented completely.

Remark: the options eig and eigcf replace the previous (now disabled) simple print options eigsfo and eigsfo.

Note that the simple print key SFO controls whether or not the EPRINT subkey *sfo* is effective at all. **TransitionField**

Part of the bonding energy is computed and analyzed by the so-called Transition State procedure [*3* (page 907), *110* (page 913)]. This has nothing to do with physical transition states, but is related to the Fock operator defined by an average charge density, where the average is taken of the initial (sum-of-orthogonalized-fragments) and the final (SCF) charge density. There is also an analogous term where the average is taken of the sum-of-fragments and the sum-of-orthogonalized-fragments. Various terms, Fock operators and Density Matrices used in this approach may be printed. To avoid confusion with real Transition States (saddle points in the molecular Energy surface) the phrase TransitionField is used here.

```
EPRINT
  TF list
END
```

List A list of items, separated by blanks or commas. The following items are recognized: Energy, Fmat, DiagFmat, FragPmat, DiagFragPmat, F*dPmat, DiagF*dPmat, OrbE.

Energy Energy terms computed from the TransitionField.

Fmat TransitionField Fock matrices.

DiagFmat Idem, but only the diagonal elements.

FragPmat The molecular P-matrix constructed from the sum-of-fragments.

DiagFragPmat idem, but only the diagonal elements.

F*dPmat The TransitionField energy term can be expressed as a Fock operator times the difference between two P-matrices (initial and final density).

DiagF*dPmat only diagonal elements

OrbE Orbital energies in the TransitionField.

By default all options are off.

8.5.5 Other Eprint subkeys

We discuss now the remaining EPRINT sub keys that are not simple shortcuts for print switches.

```
EPRINT
  Eigval noccup {nvirtual}
END
```

This specifies the *number* of one-electron orbitals for which in the SCF procedure energies and occupation numbers are printed whenever such data is output: the highest noccup occupied orbitals and the lowest nvirtual empty orbitals. Default values are noccup=10, nvirtual=10. If only one integer is specified it is taken as the noccup value and nvirtual is assumed to retain its standard value (10). Printing can be turned off completely with the EPRINT sub key SCF, see above.

Mulliken Population Analysis All population subkeys of EPRINT refer to *Mulliken* type populations.

```
EPRINT
  ATOMPOP level
END
```

Populations accumulated per atom.

level must be none, gross or matrix. none completely suppresses printing of the populations; gross yields the gross populations; matrix produces the complete matrix of net and overlap populations. Default value: matrix.

```
EPRINT
  BASPop level
END
```

Populations are printed per elementary (bas) basis function. The level options are none, short, gross, matrix. none, gross and matrix are as for atompop.

short yields a summary of BAS gross populations accumulated per angular momentum (*l*) value and per atom.

Default value: gross.

```
EPRINT
  FragPop level
END
```

Completely similar to the atompop case, but now the populations per *fragment*. Of course in the case of single-atom fragments this is the same as atompop and only one of them is printed. Default: matrix.

For all three population keys atompop, fragpop and baspop, specification of a higher level implies that the lower-level data, which are in general summaries of the more detailed higher level options, are also printed.

Printing of any populations at the end of the SCF procedure is controlled with the EPRINT sub key *SCF* (pop).

Population Analysis per MO

A very detailed population analysis tool is available: the populations *per orbital* (MO). The printed values are independent of the occupation numbers of the MOs, so they are not populations in a strict sense. The actual populations are obtained by multiplying the results with the orbital occupations.

The analysis is given in terms of the SFOs and provides a very useful characterization of the MOs at the end of the calculation, after any geometry optimization has finished. This feature is now also available in a Spin-Orbit coupled relativistic calculation, in the case there is one scalar relativistic fragment, which is the whole molecule.

The same analysis is optionally (see EPRINT subkey *SCF*, option mopop also provided in terms of the elementary basis functions (bas).

```
EPRINT
  OrbPop {noccup {nvirtual}} {tol=tol}
    subspecies orbitals
    subspecies orbitals
    ...
  subend
END
```

noccup Determines how many of the highest occupied orbitals are analyzed in each irrep. Default noccup=10.

nvirtual Determines in similar fashion how many of the lowest virtual orbitals are analyzed in each irrep. Default nvirtual=4.

tol Tolerance parameter. Output of SFO contributions smaller than this tolerance may be suppressed. Default: 1e-2.

subspecies One of the subspecies of the molecular symmetry group. Can not be used (yet) in a Spin-Orbit coupled calculation.

orbitals A list of integers denoting the valence orbitals (in energy ordering) in this subspecies that you want to analyze. This overrules the noccup,nvirtual specification for that symmetry representation. In an unrestricted calculation two sequences of integers must be supplied, separated by a double slash (//).

Any subset of the subspecies can be specified; it is not necessary to use all of them. No subspecies must occur more than once in the data block. This can not be used in a Spin-Orbit coupled equation (yet).

A total SFO gross populations analysis (from a summation over the occupied MOs) and an SFO population analysis per fragment type are preformed unless *all* MO SFO-populations are suppressed.

8.5.6 Reduction of output

One of the strong points of ADF is the analysis in terms of fragments and fragment orbitals (SFOs) that the program provides. This aspect causes a lot of output to be produced, in particular as regards information that pertains to the SFOs.

Furthermore, during the SCF and, if applicable, geometry optimizations, quite a bit of output is produced that has relevance merely to check progress of the computation and to understand the causes for failure when such might happen.

If you dislike the standard amount of output you may benefit from the following suggestions:

If you are not interested in info about progress of the computation:

```
NOPRINT Computation
```

If you'd like to suppress only the SCF-related part of the computational report and make the GeometryUpdates related part more concise:

```
NOPRINT SCF, GEO
```

(Keep computation on, so you get at least some info about the GeometryUpdates)

If you don't want to see any SFO stuff:

```
NOPRINT SFO
```

To keep the SFO *definitions* (in an early part of output) but suppress the SFO-mo coefficients and the SFO overlap matrix:

```
EPRINT
 SFO noeig, noovl
END
```

Note: the SFO-overlap matrix is relevant only when you have the SFO-MO coefficients: the overlap info is needed then to interpret the bonding/anti-bonding nature of the various SFO components in an MO.

If you are not interested in the SFO *populations*:

```
EPRINT
 SFO noorbpop
END
```

8.6 Results on Output

The (standard) output file contains information of the main characteristics of the run, the SCF and geometry optimization results, bonding energy and population analyzes. Major parts of output can be regulated with print switches, see section *Controlling printed Output* (page 252).

By default the program produces quite a bit of output, for a large part related to (Mulliken-type) population analyzes of the molecule in total, as well as of individual orbitals, both in terms of the elementary basis functions and in terms of the SFOs, the symmetry-adapted Fragment Orbitals.

The fragment-oriented approach of ADF is very suitable for a thorough chemical analysis of molecular orbital properties and a conceptual representation of results. New users are advised to spend time and get familiar with the SFO-type analysis. It is an extremely more powerful tool to understand the electronic structure of the molecule than the classical atomic orbital populations.

The results for structure and reactivity, spectroscopic properties, transport properties that are printed are meant to be self-explanatory. See also the input options for each of these properties.

8.6.1 Electronic Configuration

The direct results from the SCF are the orbital energies and occupation numbers. This defines the electronic configuration: the occupation numbers and HOMO and LUMO energies for instance show whether or not the aufbau principle is satisfied in the final situation.

The energies of the Core Orbitals can be used to interpret for instance XPS (X-ray Photoelectron Spectroscopy) data: from Koopman's theorem these core orbital energies are an approximation to the core ionization energies. This neglects the effect of relaxation upon the ionization so that absolute energy values may not be very good; relative values, however, should be fair and can therefore be used to study (relative) chemical shifts.

8.6.2 Mulliken populations

Mulliken populations are based on the elementary atomic basis functions (bas). The individual BAS populations are printed together with summaries of the populations in all basis functions with the same angular moment quantum number on the same atom. A final summary is obtained by adding all functions on each atom, yielding the atom-atom populations. The atom-atom populations per l-value can be obtained if the key EXTENDEDPOPAN is included. The atomic gross charges are derived from the net and the overlap populations in the usual way. In addition, a population analysis may be given of individual MOs (by default this is suppressed). See the EPrint keys SCF (option mopop) and orbpop.

Mulliken-type populations are computed and printed at various levels of refinement (ranging from *per-basis function* to *per-fragment type*, data for the whole molecule as well as for individual MOs), and in two different representations, one based on the elementary basis functions (bas), the other on SFOs (Symmetrized Fragment Orbitals). This is potentially a very large amount of data. Precisely what is printed by default, and how this can be modified so as to suppress output or, alternatively, to get more information, is regulated by the print keys (print, eprint).

8.6.3 Hirshfeld charges, Voronoi deformation density

Mulliken populations can be summarized to yield atomic charges. Alternative methods exist to deduce atom charges from the self-consistent results of a molecular calculation. Several of those alternatives are provided by ADF: Hirshfeld analysis, Voronoi analysis, multipole derived charges, and charge model 5.

Of the methods applied in ADF to compute charges (Mulliken, Hirshfeld, Voronoi) we recommend the Hirshfeld analysis [*125* (page 914), *126* (page 914)] and the analysis based on Voronoi *deformation* density (VDD) charges [*109* (page 913), *127* (page 914)], see below. The fragments to which the Hirshfeld charges apply are enumerated in the early geometry part of the output file, where for each fragment the numbers of the atoms are given that belong to the fragment. The sum of the Hirshfeld charges may not add up to the analytical net total charge of the molecule. Any deviation from this is caused by numerical integration precision (small effect) and the neglect of long-distance terms that ADF uses to speed up the integral evaluations. This approximation does not affect very much the energy and molecular orbital properties, but it does show up in the sum-of-charges somewhat more. It does not indicate an error (unless the deviation is really large, say in the order of 1‰ of the total number of electrons).

The Hirshfeld analysis produces a charge value per fragment, computed as the integral of the SCF charge density over space, in each point weighted by the relative fraction of the (initial) density of that fragment in the total initial (sum-of-fragments) density:

$$Q^{\text{frag(i)}} = \int \rho^{\text{SCF}} \rho^{\text{initial frag(i)}} / \sum_j \rho^{\text{initial frag(j)}} \qquad (5.1.1)$$

The VDD method is based on the *deformation* density and a rigorous partitioning of space into non-overlapping atomic areas, the so-called Voronoi cells [*109* (page 913), *127* (page 914), *128* (page 914)]. The Voronoi cell of an atom *A* is the region in space closer to nucleus *A* than to any other nucleus (cf. Wigner-Seitz cells in crystals). The VDD charge of an atom *A* monitors the *flow* of charge into, or out of the atomic Voronoi cell as a result of 'turning on' the chemical

interactions between the atoms. The VDD method summarizes the three-dimensional deformation density on a per-atom basis. It is conceptually simple and affords a transparent interpretation based on the plausible notion of charge redistribution due to chemical bonding, i.e. the gain or loss of charge in well-defined geometrical compartments of space. For the use of VDD in analyzes involving molecular fragments, see Ref. [*129* (page 914)].

In the same fashion as for the Hirshfeld analysis, a summation over all atoms is given which should yield zero (for a neutral molecule). The deviation from zero is caused by numerical integration and by neglect-of-long-distance-terms; the same remarks apply as for the Hirshfeld analysis above.

The partitioning of space, using mid-way separation planes, is inappropriate to produce useful absolute numbers when neighboring atoms have very different sizes, for instance, Hydrogen and a heavy metal. However, *changes* in the density analyzed in this way do give a reasonable general insight in the effect of bonding on the location of charge densities, in particular because the Voronoi data per atom are split up in contributions within the atomic sphere and the rest of its Voronoi cell.

Hirshfeld and Voronoi charge analyzes are printed at the end of the SCF (of the last geometry, in case of an Optimization).

The Hirshfeld analysis in ADF produces charges *per fragment*, so that *atomic* charges are obtained only if single-atom fragments are used. This limitation does not apply to Voronoi charges (data per atom). Mulliken charges are given both per atom *and* per fragment.

In the printout of charges per fragment (as for the Hirshfeld analysis), you have to be aware of the *ordering* of fragments. A complete list of fragments is printed in the early GEOMETRY section of standard output, where you also find which atom(s) correspond(s) to which fragment. Note that even when you use single-atom fragments only, the order of fragments is usually quite different from the order of atoms in your input file. Typically (but not necessarily exactly in each case), when you use single-atom fragments: consider the first non-dummy atom in your ATOMS block. This defines the first atom *type*. Then browse the ATOMS list until you find an atom of a different type. This defines the second atom type, and so on. The single-atom fragment list will often be such that you first get *all* atoms of the first atom type, then all atoms of the second type, and so on. Check the printed list-of-fragments always, to avoid mistakes in assigning Hirshfeld charges to atoms (fragments).

8.6.4 Multipole derived charges

The multipole derived charges (MDC) analysis [*170* (page 916)] uses the atomic multipoles (obtained from the fitted density) up to some level X, and reconstructs these multipoles exactly (up to level X) by distributing charges over all atoms. This is achieved by using Lagrange multipliers and a weight function to keep the multipoles local. Since the atomic multipoles are reconstructed up to level X, the molecular multipoles are represented also up to level X. The recommended level is to reconstruct up to quadrupole: MDC-q charges. The SCF should have converged for a meaningful MDC analysis.

8.6.5 Charge model 5

The charge model 5 (CM5) [*378* (page 929), *439* (page 932)] uses the Hirshfeld analysis in combination with a parametrization to yield atomic charges that can accurately reproduce dipole moments obtained from experimental results. For input, use the keyword *CM5* (page 240).

8.6.6 Bond order analysis

The Mayer bond order between two atoms is calculated from the density and the overlap matrices (key EXTENDED-POPAN), see Ref. [*140* (page 915)].

The bond order analysis with the key *BONDORDER* (page 240). produces the output in which the bond order values are printed for each pair of atoms for which the Nalewajski-Mrozek bond order value is larger than the threshold that

can be specified with the keyword BONDORDER. For convenience the printed bond orders are accompanied by the corresponding inter-atomic distance. In the Nalewajski-Mrozek approach [*148-153* (page 915)] the bond order indices b_{AB} are calculated based on the one- and two-center valence indices

$$b_{AB} = V_{AB} + W_A^{AB} V_A + W_B^{AB} V_B$$

with the weighting factors for one-center indices given by

$$W_X^{XY} = \frac{V_{XY}^{\text{COV}}}{\sum_Z V_{XZ}^{\text{COV}}}$$

Unlike other definitions of covalent bond orders, the Nalewajski-Mrozek valence indices comprise both, covalent and ionic contributions. There exist three alternative sets of the Nalewajski-Mrozek valence indices, [*148-153* (page 915), *140* (page 915)]. The bond order indices calculated from each set of the valence indices differ slightly due to arbitrariness in the way of splitting the one-center terms between bonds. More detailed description of alternative valence indices and their physical meaning is summarized in [*148* (page 915)]; see also original papers [*149-153* (page 915)]

By default the bond order indices based on the valence indices obtained from partitioning of $Tr(P\Delta P)$ are printed in the ADF output. Note that in this version the covalent two-center part (also printed in the output) is equal to the Gopinathan-Jug [*153* (page 915)] bond order. The default values are:

$$V_A = V_A^{\text{ion}} + V_A^{\text{COV}}$$
$$V_A^{\text{ion}} = \sum_{a \in A} \left(P_{aa}^{\alpha} \Delta P_{aa}^{\alpha} + P_{aa}^{\beta} \Delta P_{aa}^{\beta} \right)$$
$$V_A^{\text{COV}} = 2 \sum_{a \in A} \sum_{a' \in A, a < a'} \left(P_{aa'}^{\alpha} \Delta P_{a'a}^{\alpha} + P_{aa'}^{\beta} \Delta P_{a'a}^{\beta} \right)$$
$$V_{AB}^{\text{COV}} = 2 \sum_{a \in A} \sum_{b \in B} \left(P_{ab}^{\alpha} \Delta P_{ba}^{\alpha} + P_{ab}^{\beta} \Delta P_{ba}^{\beta} \right)$$

To produce the values from all alternative versions of Nalewajski-Mrozek valence indices, accompanied by the Gopinathan-Jug [*153* (page 915)] and Mayer [*140* (page 915)] bond orders, see the keyword BONDORDER.

The Mayer [*140* (page 915)] bond orders can also be calculated using the keyword EXTENDEDPOPAN. The two implementations of calculating the Mayer bond orders differ slightly if one uses frozen cores. They should agree exactly in all electron calculations.

8.6.7 Dipole moment, Quadrupole moment, Electrostatic potential

Dipole moment. Note that in a ion the value of the dipole moment depends on the choice of the origin, as follows from elementary electrostatic theory.

Quadrupole moment. Note that the value of the quadrupole moment often depends on the choice of the origin, as follows from elementary electrostatic theory.

Electrostatic potential at the nuclei: the Coulomb potential of the molecule at the nuclear positions, where the contribution from the nucleus itself is omitted.

8.6.8 Fragments and Basis Functions

SFOs SFOs: the Symmetry combinations of Fragment Orbitals. The SFOs are the basic conceptual entities for the analysis of MOs and other results. Note: The FO *coefficients* that expand the SFOs are normalized in the sense that they add up (squared) to unity. The resulting SFO *function* is not necessarily a normalized function. The FOs are normalized, so it depends on the *overlap* between the FOs what the self-overlap and hence the norm of the SFO is. Also printed are, for each subspecies in each irrep separately, the indices of the elementary basis functions from which the FOs and hence the SFOs are built up. (The overlap matrix of SFOs is printed much later, in the SFO Populations section after everything (SCF, Geometry) has cycled to convergence).

Elementary basis functions (BAS) First the lists of function *sets*, defined by radial behavior and the angular quantum number, are printed for all atom types on which the functions are centered. Thereafter follows the complete BAS list where the function sets have been expanded over all atoms (the *sets* are printed only for the atom *types*) and also over all Cartesian harmonics (6, not 5 *d*-functions, et cetera). In this printout the numbering can be found to which the SFO survey above refers.

8.6.9 MO analysis

MOs expanded in SFOs This gives a useful characterization of the character of the self-consistent molecular orbitals. Additional information is supplied by the SFO population analysis, see below. The definition of the SFOs in terms of the Fragment MOs has been given in a earlier part of output (section build). The SFO occupation numbers that applied in the fragments are printed. This allows a determination of the orbital interactions represented in a MO. Be aware that the bonding/antibonding nature of a SFO combination in a mo is determined by the relative signs of the coefficients *and* by the overlap of the SFOs. This overlap *may be negative!* Note also that SFOs are generally *not* normalized functions. The SFO overlap matrix is printed later, in the SFO-populations part below.

SFO population analysis For each irrep:

- Overlap matrix of the SFOs. Diagonal elements are not equal to 1.0 if the SFO is a linear combination of two or more Fragment Orbitals. The Fragment Orbitals themselves are normalized so the diagonal elements of the SFO overlap matrix give information about the overlap of the Fragment Orbitals that were combined to build the SFO.

- Populations on a per-fragment basis for a selected set of MOs (see EPrint, subkey *OrbPop*). This part is by default *not* printed, see EPRINT subkey *SFO*.

- SFO contributions per MO: populations for each of the selected MOs. In these data the MO occupation numbers are not included, so that also useful information about the virtual MOs is obtained. The printout is in matrix form, with the MOs as columns. In each printed matrix a row (corresponding to a particular SFO) is omitted if all populations of that SFO are very small in all of the MOs that are represented in that matrix. See eprint, subkey *orbpop*. Note that this method to define SFO populations (for orbitals) is very similar to the classical Mulliken type analysis, in particular regarding the aspect that *gross* populations are obtained as the diagonal (*net*) populations plus half of the related off-diagonal (overlap) populations. Occasionally this may result in negative (!) values for the population of certain SFOs, or in percentages higher than 100%. If you have such results and wonder if they can be right, work out one of the offending cases by hand, using the printed SFO overlap matrix and the printed expansion of the MOs in SFOs to compute 'by hand' the population matrix of the pertaining MO. To avoid doing large calculations it is usually sufficient to take only the few largest MO expansion coefficients; this should at least qualitatively give the correct outcomes.

- Total SFO gross populations in a symmetry representation: from a summation over all MOs (not only those analyzed in the previous section of output) in the symmetry representation under consideration. In the gross populations the MO occupation numbers have been included.

- (Per spin): A full list of all MOs (combining all symmetry representations), ordered by energy, with their most significant SFO populations. Since there might be several significant SFO populations for a particular MO, and an SFO may actually be a linear combination of several (symmetry-related) Fragment Orbitals, this table could get quite extensive. In order to confine each SFO population specification to one line of output, the SFOs are indicated by the characteristics of the first term (Fragment Orbital) of its expansion in Fragment Orbitals. So, if you see the SFO given as the '2 P:x on the first Carbon fragment', it may actually refer to the symmetry combination of, for instance, 2P:x and 2P:y orbitals on the first, second and third Carbon fragments. A full definition of all SFOs in terms of the constituting Fragment Orbitals is given in an early part of the output.

8.6.10 Bond energy analysis

The bond energy and its decomposition in conceptually useful terms: Pauli (exchange) repulsion, total steric repulsion, orbital interactions (partitioned into the contributions from the distinct irreducible representations), and corrections for some approximations (fitting and Transition State analysis procedure). For a discussion of bonding energy decompositions and applications see e.g. [*3* (page 907), *110* (page 913), *112* (page 913), *130-136* (page 914)]

The program prints the bonding energy (not in a Create or Frequencies run) and its decomposition in terms that are useful for chemical interpretation. The *total* energy is not computed. The bonding energy is defined relative to the fragments. When *basic atoms* are employed as fragments one should realize that these do not represent the atomic ground state since they are computed as spin-restricted and spherically symmetric objects, with possibly fractional occupation numbers. The correct multiplet state is not computed. To obtain the bonding energy with respect to isolated atoms you should therefore add atomic correction terms to account for spin polarization and the multiplet state. See also the SLATERDETERMINANTS key and the discussion on multiplet states.

The spin polarization energy can be computed by running the single atom unrestricted, using as fragment the corresponding (restricted) basic atom. The true multiplet state is not necessarily obtained in this way.

For the comparison of computed bonding energies with experimental data one should furthermore be aware of any aspects that are not represented in the computational formalism, such as zero-point motions and environment (solvent) effects.

In a Geometry Optimization or Transition State search, the program may print a bonding energy evaluation at each geometry (depending on print switches). A test-energy value is written in the log file. This is *not* the bonding energy, although the difference is usually small. The test-energy printed in the log file is the energy expression from which the energy gradients are computed. The true bonding energy contains in addition a few (small) correction terms that are mostly related to the fit incompleteness. These correction terms are usually very small.

If Electric Fields are used in the computation (homogeneous and/or point charges), the printed Bonding Energy is the energy of the molecule in the field minus the energy of the fragments in the same field. The energy terms due to the field are also printed separately so that one can subtract them from the total bonding energy to obtain the energy-change without field-terms.

8.7 Densf: Volume Maps

densf is an auxiliary program to generate values of molecular orbitals, charge densities and potentials in a user-specified grid, to be used typically for plotting or graphical display. The TAPE41 result file can be used directly by the ADFview program to visualize these properties.

densf requires an ascii input file where the user specifies the grid and the items that he/she wishes to see calculated on the grid, plus the standard result file TAPE21 from an *adf* calculation. * densf* writes a summary of the items that have been requested to standard output, together with some general information.

densf produces a (binary) KF file TAPE41, see OUTPUTFILE keyword below. TAPE41 is a KF file and all KF utilities can be used to inspect and process its data.

densf can also read and write cube files. See the CUBINPUT and CUBOUTPUT input options for details.

Examples of using *densf* are contained in the set of sample runs; see the *Examples* document.

8.7.1 Input

The input for *densf* is keyword oriented. The keywords may be specified in any order with one exception: INPUTFILE, if present, must be specified before any other option. Reading input by *densf* ends when it encounters the record EndInput or the end-of-file, whichever comes first.

The current version of *densf* does have reasonable defaults for all input. That means that in many cases you probably will not need to specify any input at all.

Below follows a list of the allowed keywords with their description.

```
$ADFBIN/densf << eor
INPUTFILE {file}
OUTPUTFILE {file}
VTKFILE {file}
CUBINPUT {file}
CUBOUTPUT {file}
GRID ...
UNITS ...
Density ...
KinDens ...
Laplacian ...
DenGrad ...
DenHess ...
Potential ...
Orbitals ...
NOCV ...
NCI ...
SEDD
DualDescriptor
eor
```

Input/Output files

```
INPUTFILE {file}
```

INPUTFILE keyword specifies path to the TAPE21 file from which *densf* reads the input data. Absence of the keyword is treated as if **INPUTFILE TAPE21** has been specified.

```
OUTPUTFILE {file}
```

OUTPUTFILE keyword specifies path to the (possibly existing) TAPE41 file. If the file exists, *densf* will read grid specifications from it ignoring GRID keyword in the input. Computed quantities are saved in the file overwriting existing data with the same name, if any.

```
VTKFILE {file}
```

VTKFILE keyword specifies path to a file in the format readable by VTK directly. This option exists primarily for better integration with ADF-GUI and the user should not specify it.

```
CUBINPUT {file}
```

If the CUBINPUT keyword is present then the grid as specified in the **file** is used to calculate all requested quantities. Any volume data found in the cube file is also saved in the output file. NOTE: CUBINPUT option cannot be used with a pre-existing TAPE41 file because they both specify the grid, which may lead to a conflict.

```
CUBOUTPUT {file}
```

Presence of the CUBOUTPUT keyword tells densf to save all computed quantities as cube files using **file** as filename prefix. The prefix can also contain a complete path including directories. For example, specifying the following in the densf input

```
CUBOUTPUT /home/myhome/H2O
Density SCF
```

will result in a file /home/myhome/H2O%SCF%Density.cub being created containing volume data for the total SCF density. One file per requested quantity is created.

The OUTPUTFILE, CUBOUTPUT and VTKFILE options are mutually exclusive. Absence of any of these options is treated as if **OUTPUTFILE TAPE41** has been specified.

Grid

The Grid key is available either as simple key, or as block key.

The simple key options are as follows:

```
GRID {save} {coarse|medium|fine}
EXTEND grid_extent
```

If the word *save* is specified, the program will store all grid points on TAPE41 (in addition to the specification of the grid that is always stored). The default is NOT to store all grid points.

Either coarse, medium or fine may be specified. This instructs the program to generate the grid automatically within a box enclosing all atoms of the molecule. The distance between grid points is 0.5, 0.2 or 0.1 bohr for respectively a coarse, medium or fine grid. Obviously, the size of the result file TAPE41 depends strongly on this specification. The default value (used when the user does not specify the grid) is to generate a coarse grid. The size of the grid box is determined by the atomic coordinates and the grid extent. The latter is by default equal to 4.0 Bohr and can be changed using the EXTEND keyword, in which case it is affected by the unit_of_length value specified in the UNITS block (see below).

If GRID is used as a block key it must be followed by the word end in a later record. The records until the end are the data for the Grid keyword:

```
Grid {save}
 x0 y0 z0
 n1 n2 n3
 v1x v1y v1z length1
 v2x v2y v2z length2
 v3x v3y v3z length3
END
```

If the word save is specified, the program will store all grid points on TAPE41 (in addition to the specification of the grid that is always stored). The default is NOT to store all grid points.

The records in the data block must contain (*in the order specified below!*):

- 1 Three coordinates for the 'origin' (lower-left corner) of the grid.

- 2 Three integers: the numbers of points in three independent directions. If fewer integers are supplied the grid will accordingly be less-dimensional.

- 3 Three records each containing the coordinates for the direction of the independent vector (size irrelevant) and the total length of the grid in that direction. If a lower-dimensional grid is requested (see item #2), then fewer such direction-records are read and the redundant ones, if any, are ignored. The unit of length in which the grid size is input is by default Angstrom. The default can be overridden by using the input key UNITS, see below.

Notes:

- The second record ('three integers...') specifies the number of grid *points* in the different directions. The corresponding number of steps or intervals is one less!

- If the TAPE41 result file is to be used by ADFview, the grid used must be an three-dimensional orthogonal grid, with a single step size for all three dimensions.

- If the output TAPE41 file already exists and it contains valid grid data or if CUBINPUT is specified then the GRID input is ignored.

- The unit of length used in the input file has no relation to how the data are stored on the result file and how the program processes the data internally. Internal processing and storage on file is in bohr (atomic units).

Inline Grid

DENSF can now read grid as list of points. When specifying inline grid the GRID keyword should look as follows:

```
Grid Inline
  x1 y1 z1
  x2 y2 z2
  xN yN zN
End
```

Here, x#, y#, and z# are coordinates of points at which requested properties will be calculated. This feature may be used, for example, by external programs to calculate various properties at a number of points exactly and avoid interpolation with its inaccuracy. This feature should be used only when the output file has a TAPE41 format.

As in the ADF main program, the unit of length can be set with the block type key

Units

```
UNITS
 Length unit_of_length
END
```

In *densf* the only item that can be specified in the UNITS block is the length, so it seems a bit pointless to make UNITS a *block* type key rather than a simple key. However, to make its usage identical to the application in the *adf* main program the block form has been chosen to apply also here. The unit-of-length will apply to the grid specification in the input file. Default is angstrom.

Density

Generates the charge density in the grid. It is a simple keyword (not block-type).

```
density {fit} {frag} {ortho} {scf} {trans}
```

Occurrence of the word fit specifies that all densities specified in this record will be computed from the fit functions (an approximation to the exact density), rather than from the occupied molecular orbitals.

frag, ortho, scf, and trans causes each of the corresponding densities to be computed. frag stands for the sum-of-fragments (i.e. the initial) density, scf for the final result of the *adf* calculation, ortho for the orthogonalized fragments (orthogonalization to account for the Pauli repulsion, see the ADF User's Guide), and trans for excitation transition density.

Transition density is a product of initial and final states of an excitation. In the simplest case when initial and final states consist of one molecular orbital each, in this case the corresponding transition density is a product of the two MOs. To obtain transition densities one needs to perform an excitations calculation with ADF, see EXCITATIONS keyword in ADF User's Guide. Transition densities for all excitations found in the input TAPE21 file will be calculated. The transition densities are always fit-densities.

If both the exact and the fit-densities are required the density keyword must be repeated, once with and once without the fit option specified.

The default (when the DENSITY key does not occur in the input file) is to calculate the final SCF density and the sum-of-fragments density.

The frozen core density is calculated with:

```
density core
```

Kinetic Energy Density and Electron Localization Function (ELF)

```
KinDens {frag} {orth} {scf}
```

Generates the Kinetic energy density and electron localization function on the grid.

Occurrence of any of the words requests calculation of the two quantities (KinDens and ELF) based on the corresponding density: sum-of-fragments, orthogonalized fragments, or SCF, respectively. If none of the options is present, scf is assumed.

Laplacian of the Density

The Laplacian of the exact SCF density is calculated with:

```
Laplacian
```

The Laplacian of the fitted SCF density is calculated with:

```
Laplacian fit
```

The LAPLACIAN key can occur multiple times. The LAPLACIAN feature is also supported by ADFview.

Gradient of the Density

The gradient of the exact SCF density is calculated with:

```
DenGrad
```

The gradient of the fitted SCF density is calculated with:

```
DenGrad fit
```

The gradient of the frozen core density is calculated with:

```
DenGrad core
```

Hessian of the Density

The DENGRAD key can occur multiple times. This feature should be used only when the output file has a TAPE41 format.

The Hessian of the exact SCF density is calculated with:

```
DenHess
```

The Hessian of the fitted SCF density is calculated with:

```
DenHess fit
```

The Hessian of the frozen core density is calculated with:

```
DenHess core
```

The DENHESS key can occur multiple times. This feature should be used only when the output file has a TAPE41 format.

Potential

Generates the coulomb and/or exchange-correlation potential in the grid.

```
potential {coul / XC} {frag} {ortho} {scf}
```

frag, ortho, and scf are as for the density: at least one must be specified.

coul and XC specify that the Coulomb potential, respectively the exchange-correlation potential must be computed. Precisely one of these options must be specified in the record. If both potential types are required, another input record with the potential key must be used.

In the present release the xc option is not yet operational.

The default (when the POTENTIAL key does not occur in the input) is to calculate the SCF Coulomb potential.

Orbitals

A block type key in which the required molecular orbitals are specified. The key can be repeated in input any number of times; all occurrences are read and applied.

```
Orbitals type
  (data)
END
```

The argument of the orbitals key (type) must be scf (for the scf orbitals) or frag (for the fragment orbitals) or loc (for the localized molecular orbitals, see the ADF User's Guide) or generic (see separate section).

The frag option is not operational in the present release.

In many data records in the ORBITALS block, as noted in the description of these data records, you may specify a HOMOLUMO range.

A HOMOLUMO range is the following:

```
{HOMO{{-}n}} {LUMO{{+}n}}
```

HOMO: the highest occupied orbital HOMO-n, with n an integer: the highest (n+1) occupied orbitals LUMO: the lowest virtual orbital LUMO+n, with n an integer: the lowest (n+1) virtual orbitals.

The HOMO part, or the LUMO part, or both must be specified. The integer n with sign is always optional, and the sign is always optional (and has no meaning, it is intended to enhance readability).

Thus, as an example,

```
HOMO-1 LUMO+1
```

means a range of 4 orbitals: the two highest occupied ones, and the two lowest virtuals.

Each data record in the orbitals block must have either of the following formats:

the word alpha or beta. This specifies that subsequent records refer to spin-alpha or spin-beta orbitals respectively. In a restricted calculation this has no meaning and beta must not be specified. alpha and/or beta may occur any number of times in the orbitals block. All records until the first occurrence of alpha or beta are assumed to refer to spin-alpha orbitals.

label n1, n2, n3, ... label is one of the subspecies of the point group symmetry used in the adf calculation and n1 etc. are indices of the molecular orbitals (in that subspecies) that are to be computed. This format is meaningless and must not be used for the loc orbitals type, because localized orbitals do not (necessarily) belong anymore to a particular symmetry

label HOMOLUMO label is one of the subspecies of the point group symmetry used in the calculation, the orbitals follow from the HOMOLUMO range.

label occ or label virt occ specifies all orbitals (in that symmetry representation) up to and including the highest occupied one. virt specifies all orbitals above the highest occupied one. In this context partially occupied orbitals are considered occupied. Note carefully that if in a particular symmetry representation an empty orbital is computed below the highest occupied one in that same representation (excited state), that particular empty

one is included in the list of occ. Again, this format is meaningless and must therefore not be used for the loc type of orbitals.

all occ or all virt or all HOMOLUMO Specifies for each symmetry representation:

- all orbitals up to and including the highest occupied one (in that symmetry), or

- all orbitals above the highest occupied one, or

- all orbitals defined by the HOMOLUMO range.

This form is not to be used for the LOC type of orbitals. However, using this for LOC will not result in an error but will be interpreted as identical to the following format.

all This format must be used only for the LOC type of orbitals and simply means: all computed localized orbitals (irrespective of occupation numbers).

n1, n2, ... a simple list of integer indices. This format must be used only for the loc type of orbitals since no reference is made to any symmetry representation. The indices refer of course to the list of localized orbitals as computed by ADF.

The default value used when the ORBITALS key is not present is:

```
Orbitals SCF
 All HOMO-1 LUMO+1
End
```

NOCV

In ADF2009.01 it is possible to use DENSF to calculate $\epsilon^*\phi^2$ values of Natural Orbitals for Chemical Valence (NOCVs). Additional information on NOCVs is available in Ref. [*335* (page 926)].

The relevant part of the DENSF input is as follows:

For spin-unrestricted:

```
NOCV
  Alpha
   N1:math:`\alpha`
   N2:math:`\alpha`
   ...
  Beta
   N1:math:`\beta`
   N2:math:`\beta`
   ...
END
```

For spin-restricted:

```
NOCV
    N1
    N2
    ...
END
```

N1, N2, etc. specify sequential numbers of the orbitals for which $\epsilon^*\phi^2$ is to be calculated.

Alpha and *Beta* specify that the numbers that follow refer to spin α and β, respectively. Both *Alpha* and *Beta* are optional, *Alpha* being assumed if omitted. The NOCV input block must be closed with "END".

Alternatively, one can specify to calculate all (alpha- or beta-) NOCV's:

For spin-unrestricted:

```
NOCV
  Alpha
    ALL
  Beta
    ALL
END
```

For both spin-restricted and spin-unrestricted:

```
NOCV
    ALL
END
```

The last and probably the most convenient form of the NOCV input blocks lets one to specify an NOCV eigenvalue threshold as a criterion for selecting orbitals:

For spin-unrestricted:

```
NOCV
  Alpha
    THRESH threshold
  Beta
    THRESH threshold
END
```

For both spin-restricted and spin-unrestricted:

```
NOCV
    THRESH threshold
END
```

When this form of the input is used, only those NOCVs will be included whose absolute eigenvalue is equal to or larger than the given *threshold*.

NOCV profile

It is possible to compute an integral profile of the NOCV $\epsilon^* \phi^2$ function along a given line. To calculate a single point of the profile the $\epsilon^* \phi^2$ is integrated in the plane orthogonal to the profile axis and passing through the point. In addition to the input above used for selecting NOCVs, the following input is required to calculate the profile:

```
NOCV
    PROFILE
END
LINE
    x1 y1 z1
    x2 y2 z2
    delta
END
```

The LINE input block is used to define the axis along which the profile will be generated. The (*x1,y1,z1*) point defines the origin of the profile axis and (*x2,y2,z2*) defines the second point of the line and the positive direction. The distance between the point does not matter. The *delta* value is the distance between two point in the profile. All values are given in the units specified in the UNITS input block. The positions of the left- and rightmost profile points depend on coordinates of the atoms and the grid extent value (specified by the EXTEND keyword, see above).

Generic orbitals

There is also a possibility to calculate any orbital as long as it is present in the t21 file in the BAS representation. The input syntax is as follows:

```
Orbitals GenBas
 section1%variable1
 section2%variable2
End
```

In the example above, each line contains the section and variable name of the orbital in the input t21 file. The length of the variable should be equal to the number of atomic functions (naos) and it is supposed to contain expansion coefficients of the orbital on the basis of atomic (primitive) functions.

The calculation results are stored in the output file in sections and variables with exactly the same names as specified in the input. The section and variable names may contain spaces although the leading and training spaces are discarded.

NCI The areas of non-covalent interactions (NCI), see Refs.[*356,357* (page 928)], can be recognized by the a low value of the electron density coupled with a low value of RDG (reduced density gradient $s = 1/2\ (3\ \pi^2\ )^{-1/3}\ |\nabla\rho|\ \rho^{-4/3}$) and a negative (or a small positive) value of the second eigenvalue of the Hessian of the electron density (λ_2). The regions of significant hydrogen bonding are recognized by strictly negative λ_2 while in the regions of VdW interactions it may be slightly positive. The relevant DENSF input keyword is:

```
NCI {BOTH|FIT} {RHOVDW=*RhoVdW*} {RDG=*Rdg*}
```

All arguments are optional

By default, the exact density is used to calculate the NCI properties. If FIT is specified then the fitted density is used to calculate the fields and their names are prepended with "Fit". If BOTH is specified then the NCI properties are calculated using both exact and the fittend density. Again names of the fields calculated from the fitted density start with "Fit"

The remaining arguments set relevant thresholds (all in atomic units): *RhoVdW*: density threshold for detection of weak interaction regions (default 0.02); *Rdg*: threshold on the reduced density gradient value *s* (default 0.5). A point is considered for NCI only if *s* value is smaller than Rdg.

DENSF creates three variables per density type (exact or fitted) when NCI is present in the input: *SCF%RDG* (or *SCF%FitRDG*): the reduced density gradient value *s*; *SCF%DenSigned* (or *SCF%FitDenSigned*): the sign(λ_2) ρ value for regions where *s < Rdg*; *SCF%NCI* (or *SCF%FitNCI*): the NCI flag value, see below;

If the point is considered for NCI (that is if *s < Rdg*), the sign(λ_2) ρ value (or ρ) is tested against *RhoVdW*. If $\rho <$ *RhoVdW* then the NCI value is set to 1 to flag a VdW interaction region. If sign(λ_2) $\rho <$ -*RhoVdW* then the NCI value is set to -1 to flag a hydrogen bonding region. In all other cases the NCI value is zero.

SEDD The single exponential decay detector (SEDD), see Ref. [*358* (page 928)], extracts information about bonding and localization in atoms, molecules, or molecular assemblies. The practical evaluation of SEDD does not require any explicit information about the orbitals. The only quantity needed is the electron density (calculated or experimental) and its derivatives up to the second order. For the exact equation to be used, and pictures, see Ref. [*358* (page 928)].

Specifying SEDD in the input also invokes calculation of the Density Overlap Regions Indicator (DORI) [*441* (page 933)]. The DORI is best visualized as a 0.9 isosurface colored with the sign(λ_2) ρ value (see the DenSigned variable from the NCI keyword above).

DualDescriptor

Calculate the dual descriptor (DD) in the frontier molecular orbitals approximations. The DD is essentially a difference between (an average of) the square of the lowest unoccupied and the highest occupied molecular orbitals: $DD = \frac{1}{N_{LUMO}}(\sum_i^{N_{LUMO}} \psi_i^2) - \frac{1}{N_{HOMO}}(\sum_j^{N_{HOMO}} \psi_j^2)$, where *i* and *j* are indices of orbitals within 10^{-3} Hartree of the LUMO and HOMO, respectively. Please note that this method will produce incorrect results for molecules with fractional occupation numbers.

8.7.2 Result: TAPE41

Follows a description of the contents of TAPE41. We start with a brief discussion of the sections. At the end you can find an uncommented list of all variables and sections. Note that some data are only generated when certain keywords are provided.

Sections on TAPE41

Grid

This is a general section. It contains the grid data and some more general info.

The grid characteristics are stored as:

- The 'origin' of the grid.

- The numbers of points in three independent directions.

- Three vectors, called 'x-vector', 'y-vector' and 'z-vector'. They are the *steps* in the three independent directions that define the grid.

If the save option was used in input (key grid) also all grid coordinates are stored: for each point three coordinates (xyz), also if only a 2-dimensional or 1-dimensional grid has been generated (a 2D grid does not necessarily lie in the xy-plane).

Note that the grid values are now stored in a simpler manner than in previous (prior to 2004) versions of densf, because the 'x values', 'y values', and 'z values' now each have their own, separate sections.

The remaining (general) data in this section comprises:

- The number of subspecies ('symmetries') for which data such as Molecular Orbitals may be present.

- The names of the subspecies.

- A logical with the name 'unrestricted', which flags whether the data pertain to an unrestricted calculation.

- The total number. of grid points.

SumFrag

Contains grid data of the Sum-of-fragments (charge density, coulomb potential, kinetic energy density, ELF, etc.).

Ortho

Contains similar data for the orthogonalized-fragments.

SCF

Contains the (spin) density, potential, etc. of the final (scf) solution.

Core

Contains grid data of the frozen core (charge density, gradients, Hessian).

TransDens_L1_L2

Contains grid data for electron transition densities. L1 is either SS or ST, and L2 is a symmetry label for all transitions in the section. Here SS and ST stand for Singlet-Singlet and Singlet-Triplet, respectively. Variables in each section are Fitdensity_N and Coulpot_N for the density and Coulomb potential for excitation N within this spin and symmetry.

SCF_label

'Label' is one of the symmetry subspecies.

Each such section contains the total number of orbitals in that subspecies (as used in the *adf* calculation), with their occupation numbers and energy eigenvalues.

In addition it contains the grid-values of the (user-specified subset of) MOs in that subspecies. The variable name corresponding to an orbital is simply its index in the energy-ordered list of all orbitals (in that subspecies): '1', '2', etc.

In case of spin-orbit coupling spinors are calculated. The label is one of the double group symmetry subspecies, and the square root of the density of the spinor is calculated with an approximate phase factor. The full description of the spinor can be found in the section SO_label.

SO_label

'Label' is one of the double group symmetry subspecies.

The grid-values of the (user-specified subset of) spinors in that subspecies. A spinor is a two-component complex wave function, which can be described with four real functions: real part alpha, real part beta, imaginary part alpha, and imaginary part beta. The variable name corresponding to a spinor is its index in the energy-ordered list of all orbitals (in that subspecies), combined with the real alpha part, the real beta part, the imaginary alpha part, or the imaginary beta part: '1_alpha_R', '1_beta_R', '1_alpha_I', '1_beta_I', '2_alpha_R', etc.

LocOrb

Values of the localized orbitals.

NOCV

Values related to the NOCVs.

Profile

Values related to the NOCV profile. The variables whose name start with "Dif" contain the in-plane integral (in e/Bohr) for the given point on the profile axis. A variable without "Dif" in the name contains the integral (in electrons) of the corresponding "Dif" variable on the (-infinity,x) interval.

Geometry

Some general geometric information: the number of atoms (not counting any dummy atoms that may have been used in the *adf* calculation), their Cartesian coordinates (in bohr) and nuclear charges.

Note: the *order* of the atoms here is not necessarily identical to the input list of atoms: they are grouped by atom type.

- In an unrestricted calculation the section SCF_label is replaced by SCF_label_A and SCF_label_B for the spin-alpha and spin-beta data, respectively, and similarly for LocOrb: LocOrb_A and LocOrb_B.

- One or more subspecies may not have been used in the *adf* calculation. This happens when the basis set used in that calculation does not contain the necessary functions to span symmetry-adapted combinations of basis functions for that subspecies. In such a case the corresponding section on TAPE41 will not be created by *densf*.

- If you want to verify the contents of TAPE41, use the *pkf* utility to obtain a survey or *dmpkf* to get a complete ASCII printout.

The information is presented in three columns. In the left-most column, section and variable names are printed, variable names being indented. In the middle column, variable's type and size is given. If the type is omitted, double precision floating point is assumed. The right-most column contains comments, if any.

Note that the name of a section of variable may consist of more than one word and that blanks in such names are significant. Furthermore, they are case-sensitive. Each line below contains the name of only one section or variable.

```
    NAME            length              Comment
Grid
  Start-point         (3)
  nr of points x      (one integer)
  nr of points y      (idem)
  nr of points z      (idem)
  total nr of points  (idem)
  x-vector            (3)
```

```
  y-vector            (3)
  z-vector            (3)
  nr of symmetries    (one integer)
  labels              (nr of symmetries160 characters)
  unrestricted        (one logical)
SumFrag
  CoulPot             (total nr of points)
  XCPot_A             (idem)   spin-restricted: XCPot
  XCPOt_B             (idem)
  Density_A           (idem)   spin-restricted: Density
  Density_B           (idem)
  Fitdensity_A        (idem)   spin-restricted: Fitdensity
  Fitdensity_B        (idem)
  Kinetic Energy Density_A (idem)   spin-restricted:
                                    Kinetic Energy Density
  Kinetic Energy Density_B (idem)
  ELF_A               (idem)   spin-restricted: ELF
  ELF_B               (idem)
Ortho
  Same variables as in SumFrag
SCF
  Same variables as in SumFrag and Ortho, and:
  DensityLap_A        (idem)   spin-restricted: DensityLap
  DensityLap_B        (idem)
  DensityGradX_A      (idem)   spin-restricted: DensityGradX
  DensityGradX_B      (idem)
  DensityGradY_A      (idem)   spin-restricted: DensityGradY
  DensityGradY_B      (idem)
  DensityGradZ_A      (idem)   spin-restricted: DensityGradZ
  DensityGradZ_B      (idem)
  DensityHessXX_A     (idem)   spin-restricted: DensityHessXX
  DensityHessXX_B     (idem)
  DensityHessXY_A     (idem)   spin-restricted: DensityHessXY
  DensityHessXY_B     (idem)
  DensityHessXZ_A     (idem)   spin-restricted: DensityHessXZ
  DensityHessXZ_B     (idem)
  DensityHessYY_A     (idem)   spin-restricted: DensityHessYY
  DensityHessYY_B     (idem)
  DensityHessYZ_A     (idem)   spin-restricted: DensityHessYZ
  DensityHessYZ_B     (idem)
  DensityHessZZ_A     (idem)   spin-restricted: DensityHessZZ
  DensityHessZZ_B     (idem)
  DenSigned           (idem)   sign( :math:`\lambda`\ :sub:`2` ) :math:`\rho`
  FitDenSigned        (idem)
  RDG                 (idem)   reduced density gradient
  FitRDG              (idem)
  NCI                 (idem)   +/-1 flag for non-covalent interaction regions
  FitNCI              (idem)
  SEDD                (idem)   single exponential decay detector
  DORI                (idem)   density overlap regions indicator
Core
  Density             (total nr. of points)
  DensityGradX        (idem)
  DensityGradY        (idem)
  DensityGradZ        (idem)
  DensityHessXX       (idem)
  DensityHessXY       (idem)
  DensityHessXZ       (idem)
```

```
  DensityHessYY        (idem)
  DensityHessYZ        (idem)
  DensityHessZZ        (idem)
TransDens_L1_L2                      L1: SS or ST; L2 is excitation's symmetry
  Fitdensity_1         (total nr. of points)
  Fitdensity_2         (idem)
  Fitdensity_3         (idem)
  Coulpot_1            (idem)
  Coulpot_2            (idem)
  Coulpot_3            (idem)
SCF_label_A
     (label is a symmetry subspecies.
      Spin-restricted: SCF_label)
  nr of orbitals       (one integer)
  Occupations          (nr of orbitals)
  Eigenvalues          (idem)
  1                    (total nr of points)
  2                    (idem)
  3                    (idem)
     (as many as there are Molecular Orbitals in that
      symmetry representation for the indicated spin)
SCF_label_B
     (only if spin-unrestricted same variable as
      in SCF_label_A)
SO_label
     (label is a double group symmetry subspecies.)
  1_alpha_R            (total nr of points) real alpha part of the 1st spinor
  1_beta_R             (idem) real beta part of the 1st spinor
  1_alpha_I            (idem) imaginary alpha part of the 1st spinor
  1_beta_I             (idem) imaginary beta part of the 1st spinor
  2_alpha_R            (idem) real alpha part of the 2nd spinor
  2_beta_R             (idem) real beta part of the 2nd spinor
  2_alpha_I            (idem) imaginay alpha part of the 2nd spinor
  2_beta_I             (idem) imaginay beta part of the 2nd spinor
     (as many as there are spinors in that
      double group symmetry representation)
LocOrb_A                          if unrestricted, otherwise
                                  LocOrb
  nr of orbitals       (one integer)
  1                    (total nr. of points)
  2                    (idem)
  (etc)
NOCV
  Dens_A number(occupation number)     (total nr. of points)
  Dens_B number(occupation number)     (idem)
  (etc)
Profile
  DifDens_A number*(NOCV eigenvalue)   (number of profile points)
  DifDens_B number*(NOCV eigenvalue)   (idem)
  DifSumDens_A number                  (idem)
  DifSumDens_B number                  (idem)
  DifRestSumDensities_A                (idem)
  DifRestSumDensities_B                (idem)
  DifSumBelow_A                        (idem)
  DifSumBelow_B                        (idem)
  Dens_A number*(NOCV eigenvalue)      (idem)
  Dens_B number*(NOCV eigenvalue)      (idem)
  SumDens_A number                     (idem)
```

```
  SumDens_B number                     (idem)
  RestSumDensities_A                   (idem)
  RestSumDensities_B                   (idem)
  SumBelow_A                           (idem)
  SumBelow_B                           (idem)
Geometry
  nnuc                   (one integer)
     (nr of nuclei, omitting dummy atoms)
  xyznuc                 (nnuc times 3)
     (the atoms are not in the same order as in the adf input
      file. Rather they are grouped by atomtype.)
  qtch                   (nnuc)        Atomic charges
x values
       x values          (total nr. of points)
y values
       y values          (idem)
z values
       z values          (idem)
```

8.8 Dos: Density of States

The auxiliary program *dos* computes various types of densities-of-states (DOS) for a user-specified energy interval.

dos requires an ASCII input file where the user specifies the items to be calculated and computational details, plus the standard result file TAPE21 from an *adf* calculation. The latter file must be present as a local file with name TAPE21 in the directory where *dos* is executed.

dos produces as result one or more ASCII files with the density-of-states values. Error messages and computational info (if any) are written to standard output.

8.8.1 Introduction

The program *dos* gives information on the number and character of one-electron levels (molecular orbitals) as a function of the (orbital) energy. The total density of states $N*(*E)$ is a well known concept in electronic structure theory of infinite systems (crystals). $N*(*E)*dE*$ denotes the number of one-electron levels (orbitals) in the infinitesimal energy interval dE. The total density of states (TDOS) at energy E is usually written as

$$N(E) = \sum_i \delta(E - \epsilon_i) \qquad (4.5.1)$$

where the ϵ_i denote the one-electron energies. So the integral of $N(E)$ over an energy interval E_1 to E_2 gives the number of one-electron states in that interval. Usually the δ-functions are broadened to make a graphical representation possible.

When the δ-functions are multiplied by a weight factor that describes some property of the one-electron state ϕ_i at energy ϵ_i various types of densities-of-states are obtained that provide a graphical representation of the state character (orbital character) as a function of one-electron energy.

In calculations on finite molecules the total density of states as a function of (orbital) energy may also be useful, but the main use of various types of densities-of-states is to provide a pictorial representation of Mulliken populations. The weight factors employed are related to the orbital character determined by means of a Mulliken population analysis *per orbital* (see below). The program *dos,* therefore, provides the same information as can be generated by the ADF program (a population analysis per orbital) but *dos* enables an easy graphical representation and is particularly useful when there are many one-electron levels, for instance in calculations on clusters. You can obtain a simple view of the

character of the orbitals in a certain energy range. You can also find out in which orbitals (at which energies) certain basis functions or fragment orbitals give a large contribution, and whether such contributions are bonding, nonbonding or antibonding with respect to particular bonds. Such information is provided by *dos* in the form of (weighted) density of states values over a user-specified energy range, which can for instance be plotted by *gnuplot*.

The following options are available for computations by *dos*:

- TDOS: Total Density of States

- GPDOS: Gross Population Density of States

- OPDOS: Overlap Population Density of States

- PDOS: Projected Density of States

The total density of states (TDOS) has large values at energies where there are many states per energy interval.

The GPDOS (Gross Population based Density Of States) of a function χ_μ (or a sum of such functions) has large values at energies where this function (these functions) occur(s) in the molecular orbitals.

The PDOS of a function χ_μ provides similar information, but with the projection of χ_μ onto the orbital ϕ_i as weight factor for the importance of χ_μ in the orbital ϕ_i .

The OPDOS (Overlap Population based Density Of States) between χ_μ and χ_ν has large positive values at energies where the interaction between them is bonding, and negative values where the interaction is anti-bonding. An example of the use of these plots is provided in [*326* (page 926)].

We review below the Mulliken population analysis, and then describe the forms of density of states analysis performed by DOS. Finally an input description of DOS is given.

8.8.2 Mulliken population analysis

The orbitals ϕ_i with energies ϵ_i are expanded in basis functions χ_μ , which leads to the definition of density matrices P_i describing orbital densities, from which the total density matrix can be constructed:

$$\phi_i(r) = \sum_\mu \chi_\mu(r) C_{\mu i}$$

$$\rho_i(r) = \int |\phi_i(r)|^2 = \sum_{\mu\nu} P_{i,\mu\nu} \chi_\mu(r) \chi_\nu(r)$$

$$P_{i,\mu\nu} = C_{\mu i} C_{\nu i}$$

$$\rho(r) = \sum_i n_i \rho_i(r) = \sum_{\mu\nu} P_{\mu\nu} \chi_\mu(r) \chi_\nu(r)$$

$$P_{i,\mu\nu} = \sum_i n_i C_{\mu i} C_{\nu i} \qquad (4.5.2)$$

Here μ and μ run over the basis functions, which may be either primitive functions, or combinations of primitive functions, for instance the SCF orbitals of atoms or larger fragments.

The Mulliken population analysis provides a partitioning of either the total charge density or an orbital density. The total density is written as

$$\rho(r) = \sum_{\mu\nu} P_{\mu\nu} \chi_\mu(r) \chi_\nu(r)$$

$$= \sum_{A \leq B} \sum_{\mu \in A} \sum_{\nu \in B} P_{\mu\nu} \chi_\mu(r) \chi_\nu(r)$$

$$= \sum_{A \leq B} \rho_{AB} \qquad (4.5.3a)$$

$$\rho_{AB} = \sum_{\mu \in A} \sum_{\nu \in B} P_{\mu\nu} \chi_\mu(r) \chi_\nu(r) \qquad (4.5.3b)$$

The total number of electrons, $N = \int \rho(r)dr$, is now partitioned over the atoms by assigning an overlap population $P_{\mu\nu}S_{\mu\nu} + P_{\nu\mu}S_{\nu\mu}$ for one half to the atom A of χ_μ and one half to atom B of χ_ν,

$$N = \int \rho(r)dr = \sum_{\mu\nu} P_{\mu\nu}S_{\mu\nu} = \sum_\mu GP_\mu \qquad (4.5.4a)$$

$$GP_\mu = \sum_\nu P_{\mu\nu}S_{\mu\nu} \qquad (4.5.4b)$$

GP_μ is the gross population of χ_μ. It contains the net population $P_{\mu\mu}$ and half of each total overlap population $P_{\mu\nu}S_{\mu\nu} + P_{\nu\mu}S_{\nu\mu}$ between χ_μ and χ_ν. Summing the gross population over the functions $\mu \in A$ yields the total number of electrons assigned to atom A, or the gross population of atom A, GP_A, and hence the gross charge Q_A of atom A,

$$GP_A = \sum_{\mu \in A} GP_\mu \qquad (4.5.5a)$$

$$Q_A = Z_A - GP_A \qquad (4.5.5b)$$

The overlap population $OP_{\mu\nu}$ between two functions and the overlap population Q_{AB} between two atoms are defined in an analogous manner,

$$OP_{\mu\nu} = P_{\mu\nu}S_{\mu\nu} + P_{\nu\mu}S_{\nu\mu} \qquad (4.5.6a)$$

$$Q_{AB} = \sum_{\mu \in A} \sum_{\nu \in B} OP_{\mu\nu} \qquad (4.5.6b)$$

These quantities can be evaluated for a single orbital density, $N = 1 = \int |\phi_i(r)|^2 dr$. The gross population $GP_{i,\mu}$ of a function in a specific orbital density $|\phi_i(r)|^2$ is then associated with the fraction of the orbital density belonging to that function (or the percentage χ_μ character of orbital ϕ_i, and the overlap population $OP_{i,\mu\nu}$ gives an indication of the strength of bonding or anti-bonding between χ_μ and χ_ν in orbital ϕ_i,

$$GP_{i\mu} = \sum_\nu P_{i,\mu\nu}S_{\mu\nu} = \sum_\nu C_{\mu i}C_{\nu i}S_{\mu\nu} \qquad (4.5.7a)$$

$$OP_{i,\mu\nu} = P_{i,\mu\nu}S_{\mu\nu} + P_{i,\nu\mu}S_{\nu\mu} = 2C_{\mu i}C_{\nu i}S_{\mu\nu} \qquad (4.5.7b)$$

8.8.3 Density of states analyses based on Mulliken population analysis

Total density of states

The total density of states TDOS at energy E is written as

$$TDOS : N(E) = \sum_i \delta(E - \epsilon_i)$$

so the integral of $N(E)$ over an energy interval E_1 to E_2 gives the number of one-electron states in that interval. In practice the delta functions are approximated by Lorentzians,

$$TDOS : N(E) = \sum_i L(E - \epsilon_i) = \sum_i \left(\frac{\sigma}{\pi} \frac{1}{(E - \epsilon_i)^2 + \sigma^2} \right)$$

A plot of $N(E)$ versus E reveals energetic regions where many levels are located. The width parameter s determines of course the appearance of the plot. A typical value is 0.25 eV (used as default in *dos*).

Partial (gross population and projected) density of states

In order to find out if a given function χ_μ contributes strongly to one-electron levels at certain energies, one may weigh a one-electron level with the percentage χ_μ character. We usually determine the χ_μ character by the gross populations, obtaining the GPDOS form of the partial density of states,

$$GPDOS : N_\mu(E) = \sum_i GP_{i,\mu}L(E - \epsilon_i)$$

If the weight factor is determined by projection of ϕ_i against χ_μ, we obtain the projected density of states PDOS,

$$PDOS : N_\mu(E) = \sum_i |\langle \chi_\mu | \phi_i \rangle|^2 L(E - \epsilon_i)$$

One should not use the PDOS for d-type or f-type primitive basis functions ('BAS'). A d-type function consists of 6 Cartesian functions, while there can of course be only 5 true d-type functions among them: one (linear combination) of them is in fact an s-type function (x2+y2+z2). Similarly, there are 10 f-type Cartesian functions, 3 of which are in fact p-functions. The PDOS is calculated for the 6 d-type and 10 f-type Cartesian functions, which leads to undesired results. An PDOS for SFOs does not suffer from this problem.

Overlap population density of states (OPDOS)

If the delta function representing orbital ϕ_i is weighed with the overlap population between χ_μ and χ_ν in ϕ_i, the overlap population density of states OPDOS is obtained,

$$OPDOS : N_{\mu\nu}(E) = \sum_i OP_{i,\mu\nu}L(E - \epsilon_i)$$

If an orbital ϕ_i at energy ϵ_i is strongly bonding between χ_μ and χ_ν the overlap population is strongly positive and OPDOS(e) will be large and positive around $E = \epsilon_i$. Similarly, OPDOS(E) will be negative around energy ϵ_i when there is antibonding between χ_μ and χ_ν in ϕ_i.

The OPDOS(E) has been used under the name coop (crystal orbital overlap population) in Extended-Hückel solid state calculations by Hoffmann and coworkers [2].

[2] R. Hoffmann, *A chemist's view of bonding in extended structures* (VCH Publishers, New York, 1988).

8.8.4 Generalizations of OPDOS, GPDOS, PDOS

As observed above, the basis functions in the above expressions may be primitive basis functions ('Slater type orbitals'), but of course the formulas are equally applicable for other types of MO expansions. In *dos* the user may select either the expansion in primitive basis functions ('BAS') or the expansion in SFOs (Symmetrized Fragment Orbitals) for the DOS analyses.

It is also possible in DOS to treat a *set* of basis functions simultaneously. For instance, the GPDOS for a set of basis functions μ_1, μ_2, ... is simply defined as the summation of the corresponding single-function GPDOS(E) values

$$N_{\mu-set}(E) = \sum_{\mu \in \mu-set} \sum_i GP_{i,\mu}L(E - \epsilon_i)$$

In a similar fashion the OPDOS can be defined for *two sets* of basis functions μ_1, μ_2, ... and ν_1, ν_2, ... as

$$N_{\mu-set,\nu-set}(E) = \sum_{\nu \in \nu-set} \sum_{\mu \in \mu-set} \sum_i OP_{i,\mu\nu}L(E - \epsilon_i)$$

and finally for the PDOS we get in similar fashion

$$N_{\mu-set}(E) = \sum_{\mu \in \mu-set} \sum_i |\langle \chi_\mu | \phi_i \rangle|^2 L(E - \epsilon_i)$$

8.8.5 Input

The (ASCII) input for *dos* is keyword oriented. Reading input by *dos* terminates whenever it finds a line END INPUT or the end-of-file, whichever comes first.

Follows a list of keywords with their meaning. Generally keys may occur more than once and *the order in which they appear is relevant in some cases*. For instance the key energyrange (which defines for what energy values to compute densities-of-states, see below) applies to all items that come after it in input until the next occurrence of energyrange.

```
$ADFBIN/dos << eor
ENERGYRANGE {Npoint=nr} {E-start=e1} {E-end=e2 / E-step=de}
LORENTZIAN width=width
FILE file
TDOS { title }
OPDOS ...
GPDOS ...
PDOS ...
eor
```

Energy scan values

```
ENERGYRANGE {Npoint=nr} {E-start=e1} {E-end=e2 / E-step=de}
```

This specifies for which energy values the densities-of-states are computed that are specified *after* it in the input file and *until* the next occurrence of ENERGYRANGE.

ENERGYRANGE specifies the lower bound, upper bound and number of equidistant energy values (including endpoints). All items are optional with defaults applying for those omitted.

The E end and E-step values determine one another and must therefore not be specified both (or be consistent).

The initial defaults are:

```
nr=301
e1=-20
de=0.1
```

All energy data are in eV.

When values have been changed with the key ENERGYVALUE, the so-modified values are the defaults for the next occurrence of ENERGYVALUE.

Peak widening

The peaks in the DOS curves corresponding to the energies of the molecular orbitals are widened by a Lorentzian curve, the width of which can be adjusted.

```
LORENTZIAN width=width
```

Initial default width is 0.25 (eV).

As for ENERGYRANGE, the key LORENTZIAN may occur more than once and each occurrence sets the width for all items after it.

Result files

The computed densities-of-states are stored on one or more ASCII files, which have to be specified in input.

```
FILE file
```

The key FILE may occur any number of times in input. Each time it occurs the specified file is opened by *dos*. The file must not yet exist and the new file will accumulate (ASCII) the densities-of-states data of all DOS items subsequently specified, until the next occurrence of FILE. The first occurrence of the key FILE must be given before any DOS specification (by the keys TDOS, OPDOS, GPDOS, PDOS, see below).

The format of the result file is such that it can be fed directly into *gnuplot*.

Densities of States

Total density of states.

```
TDOS { title }
```

TDOS instructs the program to compute the total density of states.

title (optional) will appear as title to the section of corresponding Density-of-States data in the result file.

The other types of densities-of-states require block-type keyword input.

```
OPDOS { title }
Ftype numbers
Ftype numbers
...
*SUBEND*
Ftype numbers
Ftype numbers
...
END
```

Ftype Specifies the type of basis functions to use in the MO expansions. If the primitive basis functions are to be used Ftype must be bas. For the SFO representation Ftype must be one of the irreducible representations of the point group symmetry. All Ftype values in the data block must be consistent: either all are bas or all are irrep labels. The scope of this consistency requirement is the data block of the current key: in a next OPDOS data block, for instance, a different choice may be made.

numbers Must be a sequence of integers referring to the basis functions to be selected, i.e. the 'μ-set' and 'ν-set' in (4.5.13) etc. If bas-type basis functions are selected the numbers refer to the overall list of all basis functions as printed in the output file of the adf run. If SFOs are selected the numbers refer to the SFO list of the pertaining symmetry representation without the core functions, see the adf output file.

SUBEND Must be typed as such and separates the 'μ-set' and the 'ν-set': all records before subend specify together the 'μ-set' and all records below subend comprise the 'ν-set'. Each of these two sections may consist of any number of records.

The input for GPDOS and PDOS are similar, but simpler because only one set of functions ('μ-set') has to be specified, so there is no subend in the data blocks for these keys.

```
GPDOS { title }
Ftype numbers
Ftype numbers
...
END
```

```
PDOS { title }
Ftype numbers
Ftype numbers
...
END
```

The keys GPDOS, OPDOS, PDOS and (TDOS) may occur any number of times in input and in any order. Each time the DOS key occurs the current energyrange and lorentzian settings apply and the results are written to the current file.

8.9 VCD Analysis: VCDtools

The auxiliary program *VCDtools* provides insight regarding the origin of the VCD intensity of a given normal mode. This is done by analysing and visualising the various contributions to the total Electric (E) and Magnetic (M) Dipole Transition Moments (DTM). *e.g.*, the contributions from atoms, electrons, nuclei and molecular fragments. The *VCD-tools* program is accessible through the ADF-GUI and allows one to:

1. Scale the size of the atoms with the magnitude of their atomic EDTM or MDTM contributions.

2. Visualize the various contributions to the EDTMs and MDTMs by arrows.

3. Perform a General Coupled Oscillator (GCO) analysis to evaluate the interactions between the different parts of a molecule.

4. Decompose the total EDTM and MDTM into nuclear and electronic contributions to gain insight into effects induced in the VCD spectra by charge transfer.

5. Perform a VCD robustness analysis.

VCDtools requires only the standard result file TAPE21 from an VCD calculation in *ADF*. Once this file is opened with *ADFspectra* the *VCDtools* program will be accessible through the standard user interface of *ADFspectra*.

VCDtools produces one ASCII file, VCDtools.out where all output from the program is printed. This file will be located in the *.results* folder in the directory of the opened TAPE21 result file. Additionally, VCDtools will run on the background every time a new normal mode is selected. It then computes the atomic EDTM and MDTM which can be displayed inside *ADFspectra* in various ways.

The following references should be cited when publishing results obtained with VCDtools Refs. [*476-478* (page 934)].

8.9.1 General Theory

The VCD intensity associated with the fundamental vibrational transition for a given normal mode j is given by the rotational strength (RS):

$$R_{01}(j) = -i\vec{E}_{01}^{tot}(j) \cdot \vec{M}_{10}^{tot}(j)$$

where $E_{01}^{tot}(j)$ and $M_{10}^{tot}(j)$ are the total EDTM and MDTM of normal mode j , i is the unit imaginary number.

Whiten the harmonic approximation the total EDTM and MDTM can be written as sums of atomic contributions:

$$\vec{E}_{01}^{tot}(j) = \sum_{\lambda=1}^{N} \vec{E}_{01}^{\lambda}(j)$$

$$\vec{M}_{10}^{tot}(j) = \sum_{\lambda=1}^{N} \vec{M}_{10}^{\lambda}(j)$$

where N is the total number of atoms in the molecule and λ runs over all atoms.

8.9.2 General Coupled Oscillator Analysis

The GCO analysis computes the contribution to total rotational strength from the interaction of two molecular fragments. This information provides important insight regarding the source of the VCD intensity and the robustness of

the computed VCD sign. In the following, a brief summary of the GCO theory originally published in Ref. [*478* (page 935)] is given.

Following the original coupled oscillator VCD mechanism, the atoms of a molecule are grouped into three fragments: **A**, **B**, and **R**. The fragments **A** and **B** represent the important Coupling Oscillator (CO) fragments, while fragment **R** is the ancillary fragment containing the rest of the atoms. As a result, the total RS can be decomposed in three components:

$$R_{01}(j) = R_{01}^{\mathbf{GCO}}(j) + R_{01}^{\mathbf{IF}}(j) + R_{01}^{\mathbf{R}}(j)$$

The first contribution $R_{01}^{\mathbf{GCO}}(j)$, is the GCO contribution to the RS and is given by the interaction between the EDTM and MDTM of fragments **A** and **B**:

$$R_{01}^{\mathbf{GCO}}(j) = -i \cdot \left[\vec{E}_{01}^{\mathbf{A}}(j) \cdot \vec{M}_{10}^{\mathbf{B}}(j) + \vec{E}_{01}^{\mathbf{B}}(j) \cdot \vec{M}_{10}^{\mathbf{A}}(j) \right]$$

The $R_{01}^{\mathbf{IF}}(j)$ term represents the contribution from the individual fragments (IF) **A** and **B** without any interaction with other fragments:

$$R_{01}^{\mathbf{IF}}(j) = -i \cdot \left[\vec{E}_{01}^{\mathbf{A}}(j) \cdot \vec{M}_{10}^{\mathbf{A}}(j) + \vec{E}_{01}^{\mathbf{B}}(j) \cdot \vec{M}_{10}^{\mathbf{B}}(j) \right]$$

and the $R_{01}^{\mathbf{R}}(j)$ contribution contains all contributions from fragment **R**:

$$R_{01}^{\mathbf{R}}(j) = -\, i \cdot \left[\vec{E}_{01}^{\mathbf{A}}(j) \cdot \vec{M}_{10}^{\mathbf{R}}(j) + \vec{E}_{01}^{\mathbf{R}}(j) \cdot \vec{M}_{10}^{\mathbf{A}}(j) \right]$$
$$-\, i \cdot \left[\vec{E}_{01}^{\mathbf{B}}(j) \cdot \vec{M}_{10}^{\mathbf{R}}(j) + \vec{E}_{01}^{\mathbf{R}}(j) \cdot \vec{M}_{10}^{\mathbf{B}}(j) + \vec{E}_{01}^{\mathbf{R}}(j) \cdot \vec{M}_{10}^{\mathbf{R}}(j) \right]$$

In the above equations $\vec{E}_{01}^{\mathbf{X}}(j)$ and $\vec{M}_{01}^{\mathbf{X}}(j)$, with $\mathbf{X} = (\mathbf{A}, \mathbf{B}, \mathbf{R})$, are the EDTM and MDTM associated with the three fragments.

Using the origin dependency of the MDTM, the GCO contribution can be rewritten in a form similar to the original CO term:

$$R_{01}^{\mathbf{GCO}}(j) = \frac{\pi \nu_j}{c} \cdot \vec{Y}^{\mathbf{GCO}}(j) \cdot \left[\vec{E}_{01}^{\mathbf{A}}(j) \times \vec{E}_{01}^{\mathbf{B}}(j) \right]$$

where $\vec{Y}^{\mathbf{GCO}}(j)$ is the general coupled oscillator vector.

As shown in ref. [*478* (page 935)] most normal modes of a molecule (i.e. not just the carbonyl stretching modes) can be interpreted in terms of the GCO mechanism. That is, for most normal modes one can define the GCO fragments **A** and **B** fragments in such a way that the $R_{01}^{\mathbf{GCO}}(j)$ term represent the dominant contributions to the rotational strengths. This second expression for $R_{01}^{\mathbf{GCO}}(j)$ does not depend on the MDTM (which is origin dependent). As such, it allows one to interpret the VCD intensity of a mode in terms of interacting EDTMs that are associated with the various moieties of a molecule and their relative orientation.

The identification of the GCO fragments (which are normal mode dependent) is not always trivial, especially, in molecules without symmetry. Consequently, *VCDtools* offers several options for dividing the molecule into fragments. In *ADFspectra* these fragments are referred as *'Regions'* in analogy to the regions in the *ADFinput* program. Under the **Regions**-menu there are many options to set, alter and save the fragments. Additionally, *VCDtools* is able to make a guess for fragments **A** and **B** as will be discussed in more detail below.

8.9.3 Available options

Beside the standard calculation and visualization of the atomic contributions to the normal mode (NM) motion, EDTM and MDTM, *VCDtools* offers three more advanced tools that can be utilized when analysing VCD spectra inside *ADFspectra:*

- GCO Analysis

- Guess GCO Fragments

- NM Localization on Regions

The first option, *"GCO Analysis"*, uses the above equations to decompose the RS in its different contributions. Printing both the values and important angles between the vectors. Before this option can be run two regions inside the molecule should be defined as fragments **A** and **B**. It is important that these regions do not contain the same atoms. Also since the decomposition is different for each NM, a specific NM should be selected.

The second option, *"Guess Fragments"*, guesses which atoms belong the fragments **A** and **B** for a selected normal mode. In doing so the $R_{01}^{\mathbf{GCO}}(j)$ is maximized while keeping $R_{01}^{\mathbf{R}}(j)$ low. Additionally, it ensures that the fragments are localized on a part of the molecule.

The third option, *"NM Localization on Regions"*, computes the percentage of the mass-weighed normal mode motion that is located on the atoms in the selected regions. One or multiple regions can be computed at the same time and the localizations are determined for all modes within a selected frequency window.

8.10 PyFrag: Activation Strain Model Analysis

The standalone script PyFrag allows one to analyse various quantities along a trajectory with ADF. Usually such a trajectory results from either an internal reaction coordinate (IRC) or a linear transit (LT) calculation and is analyzed in the context of extended activation strain model (ASM).

See also:

Example: Activation Strain Model Analysis using PyFrag (page 855)

8.10.1 Requirements

The trajectory can thereby either be a file containing concatenated molecular geometries in the xyz-format (see example below) or a T21 binary output file from a previous LT or IRC calculation with ADF. ASM is based on the concept molecular fragments, which have to be defined by specifying the corresponding atomic indices for each fragment. Furthermore, frequently used ADF input options can be imported from a separate input file (see *example* (page 855)).

8.10.2 Running PyFrag

Running PyFrag is done by starting the corresponding Python script as follows:

```
$ADFBIN/startpython $ADFHOME/scripting/standalone/pyfrag/PyFrag.py [options]
```

whereas the details of possible `[options]` are explained in the following

8.10.3 Specifying the Trajectory

The trajectory is specifed by one of the following:

```
--xyzpath <PATH>/trajectory.xyz
--irct21 <PATH>/IRCtrajectory.T21
--lt <PATH>/LTtrajectory.T21
```

8.10.4 Molecular Fragments

The system can be split into an arbitrary number of disjoint subsystems, fragments. Each fragment is defined by its own argument `--fragment` followed by the indices of atoms in that fragment:

```
--fragment 1 3 4 5 6 --fragment 2
```

8.10.5 ADF Options

The settings for the ADF calculations are retrieved from either a pyfrag-specific settings file `adfsettings` (see *example* (page 855)).

```
--adfinputfile adfsettings
```

or by individual command line arguments for each option, e.g.:

```
--adfinput basis.type=TZ2P
```

8.10.6 Analysis Options

The following options control the quantities reported by PyFrag in the final results table:

Offsets for the strain energies in kcal/mol of the fragments (one argument for each fragment, corresponding to the order of the fragment definition above):

```
--strain <StrainEnergy Fragment1> --strain <StrainEnergy Fragment2>
```

Bond length between selected pairs of atoms and equilibrium offset value in Angstrom

```
--bondlength <AtomIndex1> <AtomIndex2> <offset value>
```

Angle length between selected triples of atoms and equilibrium offset value in degrees

```
--angle <AtomIndex1> <AtomIndex2> <AtomIndex3> <offset value>
```

Hirshfeld charges of fragments

```
--hirshfeld frag<FragmentNumber>
```

Atomic charges (Voronoi deformation density charges, VDD)

```
--VDD  <AtomIndex1> [<AtomIndex2> [...]]
```

Orbital interaction energies for orbitals belonging to a point group symmetry irrep (specified by its symbol) of the whole system

```
--irrepOI <IrrepSymbol>
```

Orbital energies within fragments can be either specified for a certain orbital in a given irrep specifically for HOMO and LUMO levels in each fragment

```
--orbitalenergy <IrrepSymbol> <FragmentNumber> <OrbitalIndexWithinIrrep>
--orbitalenergy frag<FragmentNumber> HOMO
--orbitalenergy frag<FragmentNumber> LUMO
```

Orbital population numbers within fragments are again be either specified for a certain orbital in an irrep or for the HOMO or LUMO in each fragment

```
--population <IrrepSymbol> frag<FragmentNumber> <OrbitalIndexWithinIrrep>
--population frag<FragmentNumber> HOMO
--population frag<FragmentNumber> LUMO
```

Orbital overlaps between pairs of orbitals, e.g.

```
--overlap frag<FragmentNumber> HOMO  frag<FragmentNumber> LUMO
```

or

```
--overlap <IrrepSymbol> frag<FragmentNumber> <OrbitalIndexWithinIrrep> <IrrepSymbol>␣
↪frag<FragmentNumber> <OrbitalIndexWithinIrrep>
```

ACCURACY AND EFFICIENCY

See also:
Examples: *accuracy* (page 870)

9.1 Precision and Self-Consistency

The precision of a calculation is determined by

- The function sets (basis sets, levels of frozen core approximation, and fit sets for the computation of the Coulomb potential)

- Numerical integration settings in real space

- The accuracy of the density fitting procedure

- Convergence criteria (for the SCF procedure and the geometry optimization)

- A few more items that are rather technical and usually irrelevant (these are not discussed here).

The fragments you attach determine, through the fragment files, the function sets. Since each fragment traces back to one or more Create runs, the employed data base files in the Create runs determine the finally employed function sets.

For convergence of the geometry optimization see the key GEOMETRY.

In this part we examine numerical integration, density fitting and the SCF procedure. **Numerical Quality**

With the key NUMERICALQUALITY one can set the density fitting quality (ZlmFit) and the numerical integration quality (BeckeGrid) simultaneously

```
NUMERICALQUALITY [basic|normal|good|verygood|excellent]
```

9.1.1 SCF

The SCF procedure is regulated with keys that set the maximum number of iterations, the convergence criterion, and various items that control the iterative update method. Molecules may display wildly different SCF-iteration behavior, ranging from easy and rapid convergence to troublesome oscillations. We expect that the default settings take care of most cases, but one should realize that this is a difficult and tricky subject. The user has a few (main) options to adapt the procedure to the situation at hand: simple damping or one of the SCF acceleration schemes.

See also:

In case of SCF convergence problems, see *SCF troubleshooting* (page 333)

At each cycle the electron density is computed as a sum of occupied orbitals squared; the new density defines the potential from which the orbitals are re-computed. The cycle is repeated until convergence is reached. To speed-up convergence and to avoid non-convergent oscillatory behavior the values at the next iteration are constructed as a mixture of the computed new data and those used at the cycles before. This may involve only the previous cycle and is then called *damping*. Alternatively some sort of DIIS (direct inversion in iterative space) or LIST (LInear-expansion Shooting Technique) procedure can be invoked, which is a generalization of damping to include more previous iterations.

Starting from ADF2016, the SCF module has been rewritten from scratch. The old SCF is still available and it can be switched on using the `OldSCF` keyword. The description below refers to the new SCF code only. User documentation for the OldSCF module is available in the ADF2014 User's Guide.

There are various SCF acceleration techniques implemented in ADF. By default the mixed ADIIS+SDIIS method by Hu and Wang [*291* (page 924)] is used, unless the number of DIIS expansion vectors is set to zero (see the `DIIS N` key below) or `NoADIIS` is specified. Here, SDIIS stands for the original Pulay DIIS scheme. Other available methods mostly belong to the `LIST` family developed in the group of Y.A. Wang. The ARH and Energy-DIIS methods are available with the `OldSCF` method only.

Main options

Subkeys in the data block of the master key SCF control the aspects mentioned above. Each subkey is optional. Omission means the application of default values. Omission of the SCF block altogether implies defaults for all subkeys.

```
SCF
 Iterations Niter
 Converge SCFcnv { sconv2 }
 AccelerationMethod { ADIIS | LISTi | LISTb | fDIIS | LISTf | MESA | SDIIS }
 MESA [NoFDIIS] [NoLISTb] [NoLISTf] [NoLISTi] [NoSDIIS] [NoADIIS]
 ADIIS
   {THRESH1 a1}
   {THRESH2 a2}
 End
 NoADIIS
 DIIS
   {N n}
   {OK ok}
   {cyc cyc}
 End
 Mixing mix
 OldSCF
End
```

Iterations Niter

> **Niter** The maximum number of SCF cycles allowed. In case of Geometry Optimizations it applies separately to each of the SCF procedures that are executed. Default is 300. The program executes at least one cycle, even if you specify a non-positive number.

Converge SCFcnv { sconv2 }

> **SCFcnv** The criterion to stop the SCF updates. The tested error is the commutator of the Fock matrix and the P-matrix (=density matrix in the representation of the basis functions) from which the F-matrix was obtained. This commutator is zero when absolute self-consistency is reached. Convergence is considered reached when the maximum element falls below SCFcnv and the norm of the matrix below 10*SCFcnv. The default is 1e-6 (in Create mode: 1e-8).

sconv2 A second criterion which plays a role when the SCF procedure has difficulty converging. When in any SCF procedure the currently applicable criterion does not seem to be achievable, the program stops the SCF. When the secondary criterion (sconv2) has been met, only a warning is issued and the program continues normally. If the secondary criterion was not met either, the program terminates any further geometry optimizations, frequency steps, etc. You can prevent the program from terminating in such a case with the key ALLOW. The default for sconv2 is 1e-3.

AccelerationMethod Specify which SCF acceleration method must be invoked. By default the ADIIS+SDIIS method (denoted as ADIIS) is used and normally it results in optimal performance. In some cases it may be useful to use one of the methods from the LIST family or fDIIS. The LIST family methods have been implemented following publications by the group of Y.A. Wang [*432* (page 932)]. The number of vectors to store specified by `DIIS N` also applies to the LIST methods. This number is a very important parameter and it is worthwhile increasing or decreasing it in case of SCF convergence problems. A word of caution: do not just blindly increase the number for every system. Testing showed that a large number breaks convergence for some, mainly small, systems.

MESA The MESA method has also been developed in the group of Y.A. Wang. It combines several other acceleration methods (currently ADIIS, fDIIS, LISTb, LISTf, LISTi, and SDIIS). In order to improve the method's performance, some of the components can be disabled by specifying the corresponding "No" argument. For example, to remove SDIIS from MESA specify `MESA NoSDIIS`. Specifying this key invokes the MESA method regardless of the AccelerationMethod's value.

ADIIS This key has effect only if ADIIS acceleration method is used. The THRESH1 and THRESH2 arguments correspond to values of the maximum element of the [F,P] commutator matrix, ErrMax. If ErrMax $\geq$ *a1* then only A-DIIS coefficients are used to determine the next Fock matrix. If ErrMax $\leq$ *a2* then only SDIIS coefficients are used. For ErrMax between *a2* and *a1* the total DIIS coefficients are calculated from SDIIS and A-DIIS values weighted proportionally according to the ErrMax value. Thus, the weight of A-DIIS coefficients decreases as the ErrMax value becomes smaller. The default values for a1 and a2 are 0.01 and 0.0001, respectively. In difficult SCF cases, especially when the Pulay DIIS appears to be unstable, it may be worth trying to decrease the thresholds and let A-DIIS approach the final solution on its own.

NoADIIS If OldSCF is not set, specifying NoADIIS has the same effect as setting AccelerationMethod to SDIIS. it disables A-DIIS and switches SCF to a damping+SDIIS scheme. In this scheme, SCF starts with simple damping (a.k.a. mixing) and continues until the maximum element value of the [F,P] commutator drops below the `DIIS OK` parameter but for no more than `DIIS Cyc` iterations. After this, the pure SDIIS scheme becomes effective till the end of the SCF procedure. If OldSCF is set then specifying NoADIIS disables the automatic switching to ADIIS+SDIIS in case of a difficult SCF convergence.

DIIS The DIIS sub-block specifications can be given to control the SCF acceleration procedure. All these specifications are optional. Normally, ADIIS+SDIIS is activated starting from the 2nd iteration regardless of the `OK` and `Cyc` settings below. The two settings become effective only when `NoADIIS` is specified.

  N The number of expansion vectors used for accelerating the SCF. The number of previous cycles taken into the linear combination is then n-1 (the new computed potential is also involved in the linear combination). By default n=10. An input value smaller than 2 disables the DIIS. Note that this number applies not only to Pulay DIIS scheme but also to other DIIS-like methods, such as A-DIIS and LIST methods. It should be noted that methods from the LIST family are quite sensitive to the number of expansion vectors used. For this reason, these methods have a built-in limit on the number depending on the iteration number and the degree of convergence. The number of vectors in LIST increases when approaching convergence but it is always limited by `DIIS N`, which is a hard limit. It may be useful to set `DIIS N` to a value larger than 10 when using a LIST method with a difficult to converge system. A value between 12 and 20 can sometimes get the job done.

  OK The SDIIS starting criterion, when A-DIIS is disabled, by default 0.5 a.u..

  Cyc When A-DIIS is disabled, the SDIIS will start at this iteration (by default 5) irrespective of the `DIIS OK` value above.

Mixing mix

> **mix** When none of the SCF acceleration methods is active, the next Fock matrix is determined as $F = mixF_n + (1 - mix)F_{n-1}$. The default value is 0.2.

OldSCF Enforce the old SCF procedure. It will also be used automatically when one of the following is true: `Occupations Steep` is specified, level-shifting is enabled (see the `Lshift` key below), the augmented Roothaan-Hall method is enabled (see the `ARH` key below), the Energy-DIIS method is enabled (see the `EDIIS` key below), the SIC-OEP method is used, the `RESTOCC` key is specified.

Lshift vshift {Err=shift_err} {Cyc=shift_cyc} Level shifting is currently not implemented in the new SCF, therefore specifying `Lshift` enables `OldSCF` automatically.

> **VShift** The level shifting parameter. The diagonal elements of the Fock matrix, in the representation of the orbitals of the previous iteration, are raised by vshift hartree energy units for the virtual orbitals. This may help to solve convergence problems when during the SCF iterations charge is sloshing back and forth between different orbitals that are close in energy and all located around the Fermi level. Level shifting is not supported in the case of Spin-Orbit coupling. **At the moment properties that use virtuals, like excitation energies, response properties, NMR calculations, will give incorrect results if level shifting is applied.**

> **Shift_err** Specifies that level shifting will be turned off by the program as soon as the SCF error drops below a threshold; default value: 0.

> **Shift_cyc** Specifies that level shifting is not turned on before the given SCF cycle number (for the start-up geometry); default value: 1.

Note: *Electron smearing*, may be used to overcome convergence difficulties. The idea is to distribute electron occupations fractionally over the states around the Fermi level using the Fermi distribution function. This aspect is controlled by the Smear or the ElectronTemperature option of the Occupations key. See the Occupations key for more details.

Note: A-DIIS is not compatible with enforced non-aufbau electronic configurations and it is disabled in such a case automatically. A non-aufbau electronic configuration may be enforced using a block form of the Occupations key, but it may also result from the KeepOrbitals (a.k.a. orbital tracking) feature.

Energy-DIIS

```
SCF
 EDIIS
 ...
End
```

Energy-DIIS is implemented following the paper by Kudin, Scuseria, and Cances [*289* (page 924)]. The method is invoked by specifying an EDIIS keyword in the SCF block. Please note that similar to ARH and unlike the standard SCF procedure in ADF this method requires energy evaluation at each SCF cycle, which makes it significantly slower compared to energy-free SCF. You might need a higher integration accuracy to get an accurate total energy. The same restrictions apply as for the key *TOTALENERGY* (page 236). The EDIIS method will start at the 2nd SCF cycle, and the size of the DIIS space will be the same as for the normal DIIS. This subkey EDIIS can be used in addition to the other subkeys of the block key SCF. This method has been superceeded by A-DIIS and the LIST family methods.

Augmented Roothaan-Hall (ARH)

The Augmented Roothaan-Hall method has been developed by T. Helgaker and coworkers and is extensively discussed in Ref. [*271* (page 922)]. The basic idea of the method is that the density matrix is optimized directly to minimize total energy. At each step, the new density matrix is parametrized in terms of matrix exponent:

$P_{new} = \exp(-X) \, P_{old} \, \exp(X)$,

here, X is an anti-symmetric step matrix subject to the following conditions:

X = argmin{E(P(X))} - X minimizes the energy

|X| < h - length of X is smaller than or equal to some trust radius

The optimal X is found using a Conjugate Gradient method, possibly with pre-conditioning. The trust radius is updated based on how well the energy change is predicted. **ARH Input**

The ARH procedure is invoked by including the ARH block in the SCF input block. This subkey ARH can be used in addition to the other subkeys of the block key SCF.

```
SCF
  ARH
    {CONV conv}
    {ITER iter}
    {NSAVED nsaved}
    {START start}
    {FINAL} ...
  End
  ...
End
SYMMETRY NOSYM
```

All parameters in the ARH keyword are optional. The following arguments determine the main parameters of the ARH procedure.

CONV conv ARH convergence criterion. When the RMS gradient and its maximum components are both lower than the criterion, the ARH procedure will be considered converged. The default value is 10^{-4} .

ITER iter Maximum number of ARH iteration to perform. Please note that in difficult cases a huge number of iterations may be required for complete SCF convergence. The default value is 500.

FINAL Determines whether SCF is continued after ARH has completed. If this option is set, one Fock matrix diagonalization will be performed to get orbitals and the SCF procedure will be halted. By default this option is OFF.

START start Sets the SCF cycle number on which the ARH method is invoked. The default value is 2. Using a larger value may provide a better starting guess for the ARH minimization.

NSAVED nsaved Sets the number of saved density and Fock matrices used for augmentation of the electronic Hessian. The default value is 8. A larger nsaved value should be used in difficult cases when the number of orbitals very close to the Fermi level is large.

The default minimization method is Untransformed Pre-conditioned Conjugate Gradient. The following two parameters may be used to change this.

NOPRECOND Disables pre-conditioning during the CG minimization. This option should not be used if atoms heavier than the second-row elements are present.

TRANSPCG Specifying this option will enable the use of the Transformed Pre-conditioned CG method, which may result in better SCF convergence in some cases.

At each SCF step, the procedures begins by performing usual CG minimization keeping track of the total step length. If at some micro-iteration the step length exceeds the trust radius, the procedure switches to trust-radius optimization in the reduced space, which, in turn, is halted as soon as the level-shift parameter *mu* has converged. The final step is then calculated as a Newton step in the reduced space of all the trial vectors generated during CG minimization. The following options may be used to modify this behavior.

NOSWITCHING Setting this option turns OFF the switching from the normal CG to a trust-radius minimization in reduced space. Using this option helps to reduce the total number of SCF cycles is some cases.

SHIFTED Setting this option will turn ON the trust-radius optimization from the first micro- iteration.

CGITER=cgiter Sets the maximum number of micro-iterations.

The next two options determine the trust radius.

TRUSTR=trustr Initial value for the trust radius. Default: 0.5

MAXTRUSTR=maxtrustr The maximum trust radius value. This is set to 0.5 by default and should never be changed.

ARH Notes and Recommendations

Restriction: The method currently works for symmetry NOSYM calculations only. The NOSYM requirement comes from the fact that during direct optimization of the density matrix it may have a symmetry lower than that of the molecule.

The method requires the total energy to be calculated at each step, which makes it much more expensive compared to the standard SCF procedure that does not need the energy. Therefore, the method should only be used when the standard SCF procedure fails. Another complication caused by the use of the total energy is that somewhat higher integration accuracy may be required to get stable SCF convergence, and that the method may not be applicable in all cases. It is also recommended to use the *ADDDIFFUSEFIT* (page 307) keyword or a higher ZlmFit quality setting to increase accuracy of the total energy and, thus, improve convergence. Please refer to the *TOTALENERGY* (page 236) keyword for more information.

9.1.2 Numerical Integration

Many integrals in ADF are calculated via numerical integration. The integration grid used for computing these integrals is therefore an important technical aspect of an ADF calculation, affecting both the accuracy of the results and the computation time.

Becke Grid

The default numerical integration grid is a refined version of the fuzzy cells integration scheme developed by Becke [*361* (page 928)]. The implementation in ADF is described in Ref. [*375* (page 929)].

The quality of the Becke integration grid can be changed within the BECKEGRID block key:

```
BECKEGRID
   Quality [basic|normal|good|verygood|excellent]
End
```

Quality Default: **Normal**. For a description of the various "qualities" and the associated numerical accuracy, see Ref. [*375* (page 929)]. The integration grid quality defined in the BECKEGRID block key overrules the *NumericalQuality* (page 295).

Advanced options:

```
BECKEGRID
  {AtomDepQuality
      Ia1 {basic|normal|good|verygood|excellent}
      Ia2 {basic|normal|good|verygood|excellent}
      ...
  SubEnd}
  {qpnear qpnear}
  {RadialGridBoost boost}
End
```

AtomDepQuality One can define a different grid quality for each atom, with input numbers Ia1, Ia2, etc. If an atom is not present in the AtomDepQuality section, the quality defined in the `Quality` key will be used. *Example: Multiresolution* (page 884) illustrates how to use this option.

qpnear Default: 4.0 Angstrom. Only relevant if you have specified point charges in the input file. ADF generates grids only about those point charges that are close to any real atoms. The criterion, input with the qpnear subkey, is the closest distance between the point charge at hand and any real atom. Any input value is interpreted in the unit-of-length specified with the Units key.

RadialGridBoost Default: 1.0 (or 3 if a numerically sensitive functional is used). The number of radial integration points will be boosted by this factor. Some XC functionals require very accurate radial integration grids, so ADF will automatically *boost* the radial grid for the following numerically sensitive functionals: `LibXC M05`, `LibXC M05-2X`, `LibXC M06-2X`, `LibXC M06-HF`, `LibXC M06-L`, `LibXC M08-HX`, `LibXC M08-SO`, `LibXC M11-L`, `LibXC MS0`, `LibXC MS1`, `LibXC MS2`, `LibXC MS2H`, `LibXC MVS`, `LibXC MVSH`, `LibXC N12`, `LibXC N12-SX`, `LibXC SOGGA11`, `LibXC SOGGA11-X`, `LibXC TH1`, `LibXC TH2`, `LibXC WB97`, `LibXC WB97X`, `MetaGGA M06L`, `MetaHybrid M06-2X`, `MetaHybrid M06-HF`, `MetaGGA MVS`

Notes:

- The space-partition function used in ADF differs from the one described in Ref. [*375* (page 929)]. The unnormalized partition function used in the program is defined as (Ω_I is an element-dependent parameter: 0.1 Bohr for H, 0.3 Bohr for He-Xe and 0.6 Bohr for Cs-Og):

$$\mathcal{P}_{i,U} = \begin{cases} 1 & \text{if } r_{i,U} < \Omega_I \\ 0 & \text{if } \exists j : r_{j,U} < \Omega_J \\ \eta_i \frac{e^{-2(r_{i,U}-\Omega_I)/a_0}}{(r_{i,U}-\Omega_I)^2} & \text{elsewhere} \end{cases}$$

- If either the (block) key `INTEGRATION` or the key `NOBECKEGRID` are used in the input, the Voronoi grid is used:

```
NOBECKEGRID
```

- A Becke grid of normal quality is roughly equivalent (in both absolute accuracy and computation time) to INTEGRATION 4 (Voronoi scheme), and a Becke grid of good quality is roughly equivalent to INTEGRATION 6 (Voronoi scheme).

- The Becke grid is not very well suited to calculate Voronoi deformation density (VDD) charges. For accurate calculation of VDD charges the Voronoi integration scheme is recommended.

Voronoi grid (deprecated)

In ADF2012 and previous versions the default integration scheme was the cellular Voronoi quadrature scheme, implemented by te Velde and Baerends [*104* (page 913), *105* (page 913)]. Starting from ADF2013 this method is no longer the default sheme for integration. The so called Becke grid is the default, see the key BECKEGRID. If the key INTEGRATION is used, the Voronoi scheme will be used. Thanks to a smoother behavior of the relative integration

error as a function of the nuclear coordinates, the Becke grid is better suited for geometry optimization and TS search compared to the Voronoi scheme.

The Voronoi integration method requires only one input parameter which determines the precision of numerical integrals and derives from that the number of integration points.

```
INTEGRATION
  accint value
End
```

accint A positive real number: the numerical integration scheme generates points and weights such that a large number of representative test integrals are evaluated with an accuracy of accint significant digits. The default for accint depends on the runtype: 4.0 for Single Point runs and simple Geometry Optimizations, including Linear Transits; 5.0 for Transition State searches; 6.0 for the computation of Frequencies; 10.0 in Create runs.

The *number* of integration points varies strongly with accint, and this determines to a large extent the computational effort. Decreasing accint from 4.0 to 3.0 for instance roughly halves the number of points (this depends somewhat on the molecule).

The defaults should yield good precision for the very large majority of applications. Lower values (3.0 or even 2.0) can be used if precision is not crucial and the purpose is to get an impression. We recommend that you experiment for yourself to get a feel for how results may vary in quality and computing time.

The default in Create mode is very large: 10.0. This is computationally no problem thanks to the simplicity of the single atom case, in particular due to the high symmetry. There is no reason to override the default integration settings when creating basic atoms.

We've now only explained the normal, simple application of the Integration key, which we hope and expect is adequate for all your computations. Next additional details will be discussed. The distribution of points over space is internally regulated by quite a few parameters. Each of these parameters can be controlled in input. By default they depend on one another, and all of them depend on the main parameter accint. Advanced users may wish to experiment and override the default relations between the parameters.

You may also have rather non-standard applications where the default relations are less adequate. A thorough understanding of the underlying method is required to make a sensible choice for all parameters [*105* (page 913), *109* (page 913)].

More options can be defined in the `Integration` block. Consult the literature for detailed information about the various schemes.

```
INTEGRATION
  data
  data
  ...
end
```

The block form is used to override default relations between various parameters that are applied in the generation of the integration grid in the polyhedron method [*105* (page 913)]. All these parameters are accessible with subkeys in the data block of Integration. Most of the subkeys are simple keys with one single value as argument; a few subkeys are block-type (sub) keys themselves and hence require the usual format of a data block closed by subend.

accint The main precision parameter Its value defines the number of significant digits by which an internal set of standard integrals must be evaluated. The number and distribution of integration points is tuned accordingly. For normal applications this should yield a nearly optimal (given the underlying method) generation of points and weights. The default depends on the run type.

accsph The polyhedron method of generating integration points partitions space in atomic polyhedrons, partitioned in pyramids with their tops at the atom in the center of the polyhedron. A core like atomic sphere is constructed around the atom; this truncates the tops of the pyramids. accsph specifies the test precision for the generation of points within the spheres. By default accsph=accint.

accpyr Similarly this subkey sets the test level for the parts of the pyramids outside the atomic sphere. Default: accpyr=accint.

accpyu, accpyv, accpyw The truncated pyramids are mathematically transformed into unit cubes. A product Gauss integration formula is applied to the cubes, with three (test precision) parameters for the three dimensions. Accpyw controls the direction that is essentially the radial integration from the surface of the atomic sphere to the base of the pyramid. The other two control the orthogonal directions (angular). By default all three equal accpyr.

accout The region of space further away from the atoms, outside the polyhedrons, has its own precision parameter. By default accout=accint.

nouter This outer region is treated by a product formula: outwards times parallel. The latter involves two dimensions: the surface of the molecule say. The outward integration is performed with Gauss-Legendre quadrature, in a few separate steps. The lengths of the steps are not equal, they increase by constant factors. The total length is fixed. The number of steps is controlled with this subkey; default: 2.

outrad The parameter that defines the number of Gauss-Legendre integration points for each outward step. The precise relation between the actual number of points and this subkey, and the default relation between outrad and accout can be found in the implementation.

outpar Similarly the integration in the directions parallel to the surface of the atomic system is controlled by a parameter. See the implementation for details.

dishul Sets the distance between the outermost nuclei of the molecule and the boundary planes that define the boundary between the polyhedrons and the outer region. By default dishul=2.3*R, where Ris the radius of the largest atomic sphere in the molecule.

frange The outward range of the *outer region*: integration is not performed to infinity but to a distance frange from the outermost atoms, where all functions can be assumed to be essentially zero. By default frange is derived both from accint, the general precision parameter, and from the present chemical elements: heavier atoms have longer-range functions than hydrogen say. The precise relations can be found in the implementation.

linrot This parameter is significant only for symmetries with an axis of infinite rotational symmetry: Cand D It is the highest rotational quantum number around this axis that occurs among the integrands. This depends on the employed basis functions and fit functions. By default the program finds this out for itself.

qpnear If you specify point charges in the input file, there are two considerations implied for the numerical integration grid. First, since the point charges create a Coulomb singularity. The integrands (of for instance the basis function products against the Coulomb potential) can only be evaluated with high precision if the grid around the point charges has spherical symmetry and uses local spherical coordinates, exactly as is done for the atomic nuclei. Second, the point charges do not carry fit or basis functions, hence they play only a role in the more diffuse tails of the actual functions involved in integrals. Therefore, a relative low precision of the integral part close to the point charge may have little effect on the total integration accuracy. Since additional 'spherical centers' with their own surrounding grids increase the total number of points significantly, typically a few thousands *per Coulomb center*, this may result in high computational effort. Therefore, the program generates spherical grids only about those point charges that are close to the other atoms. The criterion, input with the qpnear subkey, is the closest distance between the point charge at hand and any real atom. Default 4.0 Angstrom. Any input value is interpreted in the unit-of-length specified with the Units key.

Next come the subkeys that require a list of data. The subkey must be placed on one line, the data on the next. This somewhat peculiar structure suggests that the subkeys are block keys; however their data blocks have no end code (subend) as for normal block type subkeys.

The list of data for such a subkey contains one value for each atom type. The data must be in the order in which the atom types were defined under atoms, implicitly or explicitly: remember that atoms belonging to different fragment types automatically have different atom types, even if their atom type *names* have been specified as identical under atoms.

rspher gives the radii of the atomic spheres, one value for each atom type. By default, the radii are derived from the chemical element (heavier atoms get larger spheres) and from the environment: the sphere must not be too large for the atomic cell (polyhedron).

linteg The maximum angular momentum quantum number of integrands centered on an atom of that type (one value for each atom type). This depends on the basis functions and on the fit functions. By default the program checks the function sets and sets the linteg values accordingly. This subkey is applied for the generation of grid points in the atomic spheres.

Items that relate to geometric lengths (dishul, frange, rspher) must be given in bohr (=atomic units), irrespective of the unit of length defined with units.

Atomic radial grid

For each atom the charge densities and the coulomb potentials of frozen core and valence electrons are computed in a radial grid and stored on TAPE21. The values in the points of the molecular numerical integration grid are then evaluated by interpolation from the table of radial values.

The radial grid consists of a sequence of r-values, defined by a smallest value, a constant multiplication factor to obtain each successive r-value, and the total number of points. Equivalently it can be characterized by the smallest r-value, the largest r-value, and the number of points; from these data the program computes then the constant multiplication factor. The characteristics are set with

```
RADIALCOREGRID {nrad=points} {rmin=rmin} {rmax=rmax}
```

points The number of radial grid points; default: 5000.

rmin The shortest distance used in the radial grid; default 1e-6 Angstrom

rmax The largest distance in the radial grid; default: 100 Angstrom.

rmin and rmax, when specified, are interpreted as specified in units of length defined by units.

The keyword name radialcoregrid has historical reasons: in earlier releases the radial grid was used only for the frozen core density and potential.

9.1.3 Density fitting

Zlm Fit: density fitting with radial spline functions and real spherical harmonics Note: In ADF2013 and previous versions, a different density-fitting scheme (pair-fit) was used. Include the key STOFIT if you want to use the old fitting scheme.

The basic ideas behind the so-called Zlm Fit can be described as follows. The total electron-density is split into atomic densities (in a similar way as the volume is partitioned for the Becke grid). These atomic densities are then approximated by a combination of radial spline functions and real spherical harmonics (Zlm). The implementation in ADF is described in Ref. [*379* (page 929)]. The algorithm that is used in ADF is related to the procedures proposed by Becke [*380* (page 929)] and Delley [*360* (page 928)]).

The Zlm Fit scheme (which is the default fitting scheme in ADF2016) offers certain advantages compared to the old pair-fit method, especially the possibility to calculate the Coulomb potential to very high precision.

```
ZLMFIT
 Quality {basic|normal|good|verygood|excellent}
 {AtomDepQuality
    Ia1 {basic|normal|good|verygood|excellent}
    Ia2 {basic|normal|good|verygood|excellent}
    ...
```

```
  SubEnd}
End
```

Quality The default quality of the Zlm Fit is normal. It can be changed with the subkey Quality.

AtomDepQuality One can define a different Zlm Fit quality for atoms, with input numbers Ia1, Ia2, etc.

The Zlm Fit method can be used for most features of the ADF program. For the calculation of Hartree-Fock exchange integrals, ADF uses a different fitting method, see *the section on Hartree-Fock exchange* (page 306).

Pair fit: symmetric density fit The non-default density fitting procedure in ADF, called pair fit method, is carried out separately for each pair of atoms. To use it one needs to include the keyword STOFIT.

```
STOFIT
```

The implemented approach has several advantages in efficiency but it has a drawback in that it necessitates the use of all available fit functions rather than only the symmetric combinations although the final result of course needs only a symmetric fit because the total density is a symmetric (A1) function. For atoms far apart the density fitting is performed with only symmetric functions. Given the implemented algorithm this entails an approximation which can be tuned:

```
A1FIT atomicseparation
```

atomicseparation is the threshold distance between atoms, *in Angstrom*. The symmetric fit approximation is applied only for atoms farther apart. Default is 10.0 Angstrom

Pair fit: fit integrals

```
STOFIT
```

For the computation of the Coulomb potential with the pair fit method the program uses a large number of so-called *fit integrals*: the overlap integrals of a fit function with a *product* of two basis functions, where at least two of the involved three functions are centered on the same atom. In fact these are ordinary overlap integrals of STOs because the fit and basis functions are all STOs and a product of STOs on a center is itself also an STO. To use this STO fitting method, which was previously the default, use the key STOFIT in the input of adf (also include it in the create mode of an atom, if that is explicitly used). For the bond energy a first order fit correction term is included, which makes the bond energy accurate to first order in the fit.

Obviously, when the two involved atoms are far enough apart, such overlap integrals become negligibly small. All fit integrals are ignored (and not computed) that are smaller - according to a rough but reasonable estimate - than a preset threshold.

The value of this threshold can be set via input, using the subkey CUTOFF_FIT of the *LINEARSCALING* (page 315) block key word.

True density in XC potential

For the computation of the exchange-correlation potential (XC-potential) the program uses as default the fitted density. This is an approximation. For the XC potential the true density can be used if one includes the keyword EXACTDENSITY:

```
EXACTDENSITY
```

Using the EXACTDENSITY keyword makes the calculation more time-consuming but more accurate in the following cases:

- calculations that require accurate description of virtual orbitals, such as most of the TDDFT;

- when studying systems where weak interaction, such Van der Waals forces and hydrogen bonds, are important. For example, EXACTDENSITY should be switched on when performing geometry optimization of DNA pairs.

Precision of density fitting on standard Output

In the output file of the ADF calculation one can find at the end of the SCF concise information about the density-fit precision: the error integral for the SCF density. The error integral is the integral of the difference between the exact density and the fit density, squared. Such values have very little to do with numerical integration, rather they show whether or not the employed set of fit functions are adequate to describe the SCF density. Error integral values that significantly exceed 1e-4 times the number of atoms are suspicious and may indicate some deficiency in the fit set for the actual calculation. On the last geometry (in an optimization) the fit-error integrals are also printed for the initial (sum-of-fragments) density and the orthogonalized fragments. In the bond energy analysis a 1st-order fit-correction term is given.

9.1.4 Hartree-Fock RI scheme

The implementation in ADF of the calculation of exact exchange (**Hartree Fock exchange**), which is needed for the hybrid functionals, is based on work by Watson et al., Ref. [*138* (page 915)]. This procedure is generally referred to as **resolution of the identity** (abbreviated as **RI**). It employs an auxiliary set of functions (**fit functions**) to efficiently approximate the Hartree Fock exchange matrix.

Note: In ADF2017 and previous versions, a different implementation of the Hartree-Fock RI scheme was used by default. See *Old Hartree-Fock RI scheme* (page 307).

Technical aspects of the RI scheme can be tweaked in the `RIHartreeFock` key block:

```
RIHartreeFock
  UseMe                 {True|False}
  Quality               {Auto|Basic|Normal|Good|Verygood}
  FitSetQuality         {VeryBasic|Basic|Normal|Good|VeryGood}
  DependencyThreshold rVal

  AtomDepQuality
    Ia1 {VeryBasic|Basic|Normal|Good|VeryGood}
    Ia2 {VeryBasic|Basic|Normal|Good|VeryGood}
    ...
  SubEnd
End
```

`UseMe` Default: True. Set to False if you want to use the old RI scheme.

`Quality` Default: Auto. Set the quality of the **FitSet** and of other RI specific technical procedures, including numerical integration and linear scaling parameters. If `Auto`, the quality defined in *NumericalQuality* (page 295) will be used.

`FitSetQuality` The quality of auxiliary fit set employed in the RI scheme (If both `RIHartreeFock%Quality` and `RIHartreeFock%FitSetQuality` are specified, the value specified in `RIHartreeFock%FitSetQuality` will be used). `Normal` quality is generally sufficient for basis sets up to and including TZ2P. For larger basis sets (or for benchmarking purposes) a `VeryGood` fit set is recommended. **Note that the FitSetQuality heavily influences the computational cost of the calculation.**

`DependencyThreshold` Default: 1.0E-3. To improve numerical stability, almost linearly-dependent combination of basis functions are removed from the Hartree-Fock exchange matrix. If you obtain unphysically large bond energy in an Hybrid calculation, you might try setting the DependencyThreshold to a larger value (*e.g.* 3.0E-3).

`AtomDepQuality` One can define different fit sets qualities (`FitSetQuality`) for each atom, with input numbers Ia1, Ia2, etc. If an atom is not present in the AtomDepQuality section, the quality defined in the `FitSetQuality` key will be used. *Example: Multiresolution* (page 884) illustrates how to use this option.

Old Hartree-Fock RI scheme

To use the old Hartree-Fock RI scheme one must specify:

```
RIHartreeFock
  UseMe False
End
```

In ADF2017 and previous versions, a different implementation of the Hartree-Fock RI scheme was used by default.

The old Hartree-Fock RI uses a smaller fit set than the default scheme, and does not include *H* and *I* fit functions; the new scheme should give to more accurate results, especially for f-block elements.

Memory usage

Calculation of the Hartree-Fock exchange may require a lot of memory. Shared memory is used to buffer the necessary data used by all processes on a multi-processor node. By default, ADF will use 30% of the total physical memory of the computer for this buffer, which may be more than is desirable or possible. This amount, in megabytes, can be set using a HFMAXMEMORY input keyword.

```
HFMaxMemory mbytes
```

The amount of memory used is related how many atoms can be done in a single pass. Thus, another way to limit the amount of memory used by ADF is to limit the number of atoms done per pass. The latter can be done using the HFATOMSPERPASS keyword. The safest, but also the slowest, is to set HFATOMSPERPASS to 1.

```
HFAtomsPerPass AtomsPerPass
```

If both HFMAXMEMORY and HFATOMSPERPASS are present, the value specified in HFATOMSPERPASS takes precedence. To debug memory usage in the Hartree-Fock routine, one can use a PRINT HARTREEFOCK keyword.

Numerical issues

Numerical problems have been found with the present implementation of Hartree-Fock or (meta-)hybrids during the SCF, especially if the molecule has symmetry NOSYM and a basis set TZP or larger is used. Workaround is to use always the DEPENDENCY key with rather strict criteria for the basis set dependence, namely bas=4e-3. In ADF2010 these numerical problems have been reduced. The DEPENDENCY key is automatically switched on in the case of a Hartree-Fock or a (meta-)hybrid potential. The result of the DEPENDENCY key is that linear dependence of the basis set is reduced by removing linear combinations that correspond with eigenvalues in the virtual SFOs overlap matrix, which are smaller than, in this case, 4e-3. Note that this is a rather large value, such that it will have an effect on the bonding energy. For DZ and DZP basis sets this value will normally not result in reduction of the virtual space. However, for TZP, TZ2P, QZ4P and larger this will often result in reduction of the basis set, which will have an effect on the accuracy of the bonding energy. In these cases one could try a smaller value than 4e-3, but be aware that numerical problems may occur. If the molecule has symmetry the numerical problems are reduced.

The origin of this problem is that for an accurate description of Hartree-Fock exchange one needs more (diffuse) fit functions in the fit procedure which is used in ADF, which uses only fit functions on the two centers of the two STOs. One can get more diffuse fit functions if one adds in the Create run of an atom the key:

```
AddDiffuseFit
```

If the BASIS key is used one can also add this key in the molecular calculation (the scripts in ADF will then automatically add this in the Create runs of the atoms). If one adds this key preliminary results indicate that one can lower the value for the dependency key to bas=1e-4. Such a low value for the dependency key normally means that the basis set is not reduced for basis sets of TZP or TZ2P quality.

A different way to add fit functions that is useful for standard basis sets DZ, DZP, TZP, and TZ2P is to add the subkey 'Fittype QZ4P' in the BASIS key, thus:

```
BASIS
  Type ...
  Core ...
  FitType QZ4P
End
```

In ADF2013 use is made of distance cut-offs for the calculation of the HF exchange integrals. This can reduce computation time and memory usage, especially for large molecules, however, this can also reduce the precision, which can lead to numerical problems. It is possible to set the distance cut-off threshold (starting from ADF2013.01b) for the calculation of the HF exchange integrals, A value of 99 for hffit virtually excludes the possibility that something will be neglected:

```
LINEARSCALING
  HF_FIT hffit
END
```

SCF problems

However, for larger molecules and in case the molecule contains heavy elements (Z>36) one still should use rather strict criteria for the basis set dependence, such as bas=4e-3.

In case of SCF problems that might be related to numerical issues one can try one or more of the following, that were discussed before

```
LINEARSCALING
  HF_FIT 99
END
Basis
 FitType QZ4P
End
AddDiffuseFit
Dependency bas=5e-3
```

9.1.5 Dependency (basis set, fit set)

Conceivably the sizes of basis and/or fit sets may be so large that the function sets become almost linearly dependent. Numerical problems arise when this happens and results get seriously affected (a strong indication that something is wrong is if the core orbital energies are shifted significantly from their values in normal basis sets). Although for the fit set a few (incomplete) tests are carried out, the program will generally not check such aspects and carry on without noticing that results may be unreliable.

A new feature has been implemented to take care of this. For reasons of compatibility with previous versions and also because our experience with it is limited so far, we have chosen to make application of it not the default.

You have to activate it explicitly. Our experience so far suggests that real problems only arise in case of large basis sets with very diffuse functions (i.e.: not with the normal basis sets provided in the standard package).

Use of the block DEPENDENCY turns internal checks on and invokes countermeasures by the program when the situation is suspect. A few technical (threshold-type) parameters can be set as well, but this is not necessary, assuming that the defaults are adequate.

```
DEPENDENCY
   {bas tolbas}
   {eig BigEig}
   {fit tolfit}
End
```

tolbas A criterion applied to the overlap matrix of unoccupied normalized SFOs. Eigenvectors corresponding to smaller eigenvalues are eliminated from the valence space. Default value: 1e-4. Note: if you choose a very coarse value, you'll remove too many degrees of freedom in the basis set, while if you choose it too strict, the numerical problems may not be countered adequately.

BigEig Merely a technical parameter. When the DEPENDENCY key is activated, any rejected basis functions (i.e.: linear combinations that correspond with small eigenvalues in the virtual SFOs overlap matrix) are normally processed until diagonalization of the Fock matrix takes place. At that point, all matrix elements corresponding to rejected functions are set to zero (off-diagonal) and BigEig (diagonal). Default: 1e8.

tolfit Similar to tolbas. The criterion is now applied to the overlap matrix of fit functions. The fit *coefficients*, which give the approximate expansion of the charge density in terms of the fit functions (for the evaluation of the coulomb potential) are set to zero for fit functions (i.e.: combinations of) corresponding to small-eigenvalue eigenvectors of the fit overlap matrix. Default 1e-10.

Notes:

- Application / adjustment of tolfit is not recommended: it will seriously increase the cpu usage while the dependency problems with the fit set are usually not so serious anyway.

- Application of the dependency/tolbas feature should not be done in an automatic way: one should test and compare results obtained with different values: some systems look much more sensitive than others. We have, so far, not been able to understand an unambiguous pattern in these experiences. Of course, when things become clearer in this respect, we will implement the corresponding intelligence into the program.

- When the dependency key is used, the numbers of functions that are effectively deleted is printed in the output file, in the SCF part (cycle 1) of the computation section.

- The TAPE21 result file of a calculation that used the DEPENDENCY key contains information about the omitted functions and these will also be omitted from the fragment basis when the TAPE21 is used as a fragment file.

9.2 Basis Set Superposition Error (BSSE)

The Ghost Atom feature enables the calculation of Basis Set Superposition Errors (BSSE). The idea is as follows. In a normal calculation of the bonding energy of a molecule c, composed of fragments a and b, one compares the total energies of c vs. those of isolated a and isolated b added together. In ADF this can be done in one stroke by running c from fragments a and b.

The BSSE is determined as the bonding energies of a pseudo-molecule d composed of (1) a plus a ghost b and (2) b plus a ghost a. The ghost atoms in the calculations are at their normal positions in the true molecule c, and they have their normal basis (and fit) functions. However, they do not have a nuclear charge and no electrons to contribute to the molecule. To set such a calculation up one needs first to make the appropriate ghost basis set files: for each involved atom, copy the basis set file that was used for its creation and modify it so as to remove the frozen core. Next, Create the ghosts with zero mass and zero nuclear charge. Apply these ghost fragments in the BSSE runs.

See also: *Example: Basis Set Superposition Error (BSSE): Cr(CO)5 +CO* (page 870).

9.3 Control of Program Flow

9.3.1 Limited execution

```
STOPAFTER programpart
```

programpart Must be a predefined name associated with a (major) part of the program With this key you tell ADF to terminate the job after the named program part has been executed.

A survey of the recognized names with a brief explanation follows below. The program parts are listed in order of execution: by taking a name further down the list you execute a larger part of the program.

init initialization procedure, input reading and printing of the output header with the job identification.

input input-reading module.

geomet geometry section: organization of atoms in types of atoms and fragments, checks of the actual fragments against information on the attached fragment files.

config electronic configuration (if not determined only by the SCF procedure), printout of symmetry subspecies.

mainsy generation of symmetry information, representation matrices, etc.

symfit construction of symmetry adapted fit functions.

cblock generation of integration points and the distribution of them in the blocks that control the internally used segmented vectorization loops.

engrad Relevant only in an optimization calculation. Engrad calculates energy gradients. The geometry is not yet updated and no printing of convergence tests and new coordinates is carried out.

geopt This routine evaluates energy gradients and updates the geometry accordingly; it also prints the convergence tests and the computed new coordinates. Compare 'stopafter engrad'.

forcematrix in a Frequencies run, terminate the calculation when all displacements have been done and before any further processing of the computed Hessian, such as the determination of normal modes, takes place.

9.3.2 Skipping

With the following key you can restrict which parts of the program are actually executed:

```
SKIP argumentlist
```

argumentlist A sequence of names, separated by blanks or commas. skip may occur any number of times in input. The names in the argument list refer to various items that are associated with parts of the program. With this key you tell ADF to skip the named program part(s) and to continue execution thereafter. The program does not check any consequences and may even crash when variables have not been initialized or have attained incorrect values due to the skipping.

Use of this key should be contemplated only in debugging and testing sessions, in which you may skip the computation of certain data when before that data will be needed you'll halt the program to inspect something.

Recognized and operational arguments are for instance (possibly not complete due to frequent extensions in this respect): atpair, ets, fitint, orthon, qmpot

9.3.3 Ignore checks

ADF performs several checks during a calculation, and stops with an error message when intermediate results are suspicious, when input-specified instructions are incompatible, etc. These controlled aborts can in some cases be overruled. Of course, the checks have been inserted for good reasons and one should realize that ignoring them probably produces incorrect results and/or may lead to a program-crash.

```
ALLOW argumentlist
```

argumentlist A sequence of names, separated by blanks or commas. allow may occur any number of times in input, see the list below for the names that can be used.

BadCoreInt Numerical integration of the frozen core density should closely approximate the analytical value. If the deviation is large compared to the user-specified numerical integration precision the program aborts with an error message like 'BAD CORE INTEGRAL'. This control is overruled by using this ALLOW option.

BadIntegrals Only applicable when the direct-SCF option is turned off for the basis functions. (This happens automatically for ZORA full-potential calculations). In that case, a sequence of elementary overlap integrals are evaluated with the numerical integration grid and the outcomes tested against the analytical value. If the deviation is too large a warning is issued. Above a certain threshold the program will abort, unless you override the exit with this Allow option.

BadSCF If the SCF procedure hasn't converged, any geometry manipulations (optimization, linear transit ...) will be aborted because the energy gradients are not reliably computed in a non-self-consistent field.

CloseAtoms Atom-atom distances should not be less than 0.2 Bohr. This is checked in the program section where the numerical integration grid is generated.

RelGeo Geometry manipulation (optimization, linear transit...) is not supported for all of the relativistic options. See Relativistic

SmallBlocks The list of numerical integration points is partitioned in blocks, so as to fit data arrays (for instance values of all basis functions in the points of a block) in available memory. The program computes the maximum block length from available memory and size parameters such as numbers of basis functions. A small block size implies a severe reduction in CPU efficiency. Therefore, the program aborts (by default, to override by this ALLOW option) if the block length turns out to be very small (less than 10).

xc Certain combinations of the Density Functional options or application of them with some other features are not allowed. See XC.

9.3.4 Parallel Communication Timings

With the key

```
COMMTIMING
```

in the input you instruct ADF to skip normal execution and perform only a test on the gather, broadcast and combine routines. Used mainly for ADF developers to test parallel schemes.

9.4 Technical Settings

9.4.1 GPU Acceleration

ADF2017 can use NVidia GPUs for accelerating ADF calculations that use GGA functionals.

To benefit from GPU acceleration in ADF, you will need:

- A 64 bit Linux OS

- The IntelMPI+CUDA version of ADF2017

- An NVidia GPU with good double precision performance (we also support multi-GPU)

- The official NVidia drivers version 367 or higher installed

What parts of ADF are accelerated?

Single Point, Geometry Optimization and Frequency calculations can be accelerated. Only GGA functionals are supported at the moment. The GPU acceleration in ADF uses a hybrid scheme in which the GPU works in tandem with the CPU, so both need to be beefy for fast calculations.

What GPU do I need for accelerating ADF?

You will need an NVidia GPU that has good double precision performance, such as the Tesla K20/K40/K80, Tesla P100, GTX Titan, Titan Black, Titan Z, or the Quaddro GP100. See the Wikipedia list of NVidia GPUs (https://en.wikipedia.org/wiki/List_of_Nvidia_graphics_processing_units) for more details on NVidia GPUs and their performance numbers. As a rough estimate your GPU will need to have ~40-50 GFLOPS of double precision performance per CPU core in your machine.

How do I get it to work?

- Make sure your Linux OS is 64bit: run the following command in a terminal and check that it says x86_64 or amd64:

```
uname -m
```

- Make sure your NVidia device is recognized and your driver version is higher than 367: run the following command in a terminal and check the output for driver version and your GPU:

```
nvidia-smi
```

- The user running the ADF calculation must have read and write access to the device. To make sure you can access the GPU, run the following command in a terminal and check the permissions:

```
ls -l /dev/nvidia*
```

- Make sure the GPU has "Compute Mode" set to "Default". To check this run the following command in a terminal:

```
nvidia-smi -a
```

- Download and install the IntelMPI+CUDA version of ADF from the "Less frequently used platforms" section on the download page (https://www.scm.com/support/downloads/)
- Make sure your calculation uses a GGA functional

What input options are available?

The IntelMPI+CUDA version of ADF2017 enables GPU acceleration by default for any ADF calculation, no changes to the input are needed for this. To disable GPU acceleration (while using the IntelMPI+CUDA version), set the SCM_GPUENABLED environment variable to "FALSE". You can do this by inserting the following two lines on the second line in the .run script of your ADF job:

```
SCM_GPUENABLED="FALSE"
export SCM_GPUENABLED
```

The ADF input recognizes the following input options in the GPUENABLED block:

```
GPUENABLED
  SETVALIDDEVS [0 1 2 ...]
  NOFOCKY
  NOGRADS
END
```

The SETVALIDDEVS keyword should be followed by a list of integers that correspondent to the CUDA id numbers of your GPUs (note: these numbers are not the same as the ones reported by nvidia-smi). This keyword can also be (ab)used for manual load balancing if you have multiple GPUs. For example if you want to distribute load 2:1 between device 0 and device 1, specify SETVALIDDEVS 0 0 1. If you simply want to make sure that only your Tesla card

is used and not some other NVidia GPU that is in the system, have a look at the CUDA_VISIBLE_DEVICES environment value in the NVidia documentation (http://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html#env-vars).

The NOFOCKY keyword disables GPU acceleration in the SCF cycles that calculate the Fock matrix.

The NOGRADS keyword disables GPU acceleration in the Frequency calculation part.

Some technical details about the GPU acceleration in ADF:

ADF is parallelized using the MPI standard, and CUDA 8.0 is used on top of that.

The code decides per CPU process which GPU device to use from the SETVALIDDEVS list, or from all devices that are available in the system with CUDA compute capability 2.0 or higher. This usually means that multiple CPU cores will be sharing a GPU board for the acceleration, and the ratio of CPU cores per GPU should thus not be too high. See the Wikipedia list of NVidia GPUs (https://en.wikipedia.org/wiki/List_of_Nvidia_graphics_processing_units) for more details on NVidia GPUs and their performance numbers. As a rough estimate your GPU will need to have ~40-50 GFLOPS of double precision performance per CPU core in your machine.

Only parts of the ADF calculations are GPU accelerated, so you will still need a fast CPU. The accelerated parts work in a hybrid scheme: the GPU performs a numerical integration while the CPU calculates the needed input values. The speed-up that can be obtained this way depends on your system, the basis set size and the grid quality settings. We have seen speed-ups of up to 1.9x for single-point calculations, and up to 2.2x for frequency calculations.

ADF can use multiple GPUs when running on a single machine and also when running on multiple nodes. Each CPU process uses one GPU that is local to the CPU it is running on. This means that ADF should run on at least as many cores as GPUs it is supposed to use. When running in a cluster environment, ADF will only detect and use GPUs that are inside the node the job is running on.

What should I do if I have problems?

Send an email to support@scm.com that mentions GPU acceleration and explains your problems. Attach the input and output of your job to this email, as well as the outputs of "nvidia-smi", "ls -l /dev/nvidia*", "uname -a" and "cat /proc/cpuinfo"

9.4.2 Memory usage

The amount of memory used by the program during a calculation is determined by three quantities:

- The size of the program itself (executable statements, static arrays). This quantity depends on the program version and is currently around 20 MB.

- Buffer space used by ADF for more efficient I/O handling. This quantity is set at installation. See the Installation Manual.

- Dynamically allocated arrays. The program allocates memory dynamically during the run conform the requirements of the actual calculation.

Starting from ADF2010 in case of parallel calculations some of the data arrays that are used within ADF will be shared by processes on the same node, provided the operating system allows shared memory. This will reduce the total amount of memory used by all ADF processes on each node because only one copy of certain large arrays per node will be present. Note that shared arrays is not the same as distributed arrays. To disable the use of shared memory one can specify the following keyword:

```
NoSharedArrays
```

9.4.3 Direct SCF: recalculation of data

The program's performance can be defined in terms of the amounts of time (CPU and I/O seconds) and disk space used in a calculation. Also important for the human user is the turn-around time. On multi-user machines CPU-cheap jobs may take a lot of real time to execute due to I/O scheduling. Therefore ADF recomputes some items, like basis functions and fit functions in the integration points, rather than store them on disk. This will increase the amount of CPU time but reduce disk access and it may also improve the turn-around.

The direct SCF method is default and can not be changed. The keys DISK or DIRECTSCF are therefore obsolete and can not be used.

9.4.4 Vector length

Numerical integration is applied in ADF to evaluate Fock matrix elements and many other quantities that are defined as integrals over basis functions, the charge density, the potential, etc. As a consequence a large part of the CPU time is spent in simple do-loops over the integration points. The total number of points depends on the required precision and on the number of atoms, the geometry and symmetry. All such numerical integration loops are segmented into loops over *blocks* of points, each block consisting of a certain number of points. This latter defines the most inner do-loop and hence determines vectorization aspects.

Depending on the computer, c.f. the compiler, vector operations may be executed more efficiently using longer vectors. Long vectors increase the demand on Central Memory however because the program may sometimes have to access large numbers of such vectors in combination (for instance all basis functions) so that they must be available in memory simultaneously. The optimum vector length depends therefore on the balance between vectorization efficiency and memory usage. The maximum vector length that you allow the program to use can be set via input.

```
VECTORLENGTH vectorlength
```

The default is set at the installation of ADF on your platform, see the Installation manual. For organizational reasons the true vector length actually used in the computation may be smaller than the value defined with this key, but will not exceed it (except in a Create run, but in that case performance and memory usage are no hot topics).

9.4.5 Tails and old gradients

The key TAILS is currently obsolescent because of the introduction of the LINEARSCALING block and may be removed in future versions.

Each block of points (see above) covers (more or less) a certain region in space and can hence be assigned a distance value with respect to a particular atom. These distances are used to control whether or not to evaluate functions centered on that atom in that particular block of points.

```
TAILS {bas=tailbas} {fit=tailfit}
```

tailbas, tailfit Accuracy levels, similar to the integration parameter: a higher value implies higher precision: in this case, basis functions and fit functions respectively are assumed zero in blocks of points that are at a sufficiently large distance from the atom at which the function is centered. Sufficiently large is defined by comparing the integral of the (radial part of the) function beyond that distance with the total integral. By default tailbas and tailfit both depend on the numerical integration parameter

Note: in contrast with some of the older versions, supplying only the keyword without parameters does not switch off the use of function cutoffs. To effectively switch off the distance effects in gradients evaluation one should specify large values for the BAS and FIT parameters. The value of 100 should be more than enough, thus, for example:

```
TAILS bas=100 fit=100
```

Improved performance in geometry optimizations and frequency runs is achieved by a new implementation of the calculation of the gradients that now uses linear scaling techniques.

The key TAILS is not used in geometry optimizations anymore. For controlling the use of distance effects in normal SCF calculations, and for calculations with the RESPONSE or EXCITATIONS keywords, please check the LINEARSCALING block.

9.4.6 Linearscaling

The LINEARSCALING block has a very similar function to the TAILS keyword described above. In addition to defining the precision of operations related to operations in the numerical integration grid, it also defines the precision for the calculation of the overlap matrix, the fit integrals, and the density fit procedure. Default values have been chosen which result in negligible differences in the results for our test calculations, so that these defaults can be considered safe. They have been chosen similarly to the defaults for the TAILS keyword.

However, it may be advisable to modify the settings for the linear scaling parameters in two cases. First, if a very accurate result is needed, and numerical noise is to be completely eliminated, strict values can be specified. Especially for small molecules, where timings are not so large anyway, this may be of interest. Second, for large molecules, in which the calculations are very time-consuming, one can experiment with less strict values for the LINEARSCALING block keyword. In such a case one should be aware of the reduced accuracy and preferably test the influence of the changes on the results.

In the simplest application of the LINEARSCALING block, only one parameter is provided. All the subkeys described below will then be given this value. A very large value implies a calculation where no distance cut-offs are used. A normal value (almost default situation) would be 8 for linscal, 6 gives a faster but somewhat sloppier result. Whether this is acceptable is strongly case-dependent. A value of 10 or 12 is already quite strict and, unless there are some sort of numerical problems, there should not be much influence on the results by choosing a stricter value than that. A value of 99 for linscal virtually excludes the possibility that something will be neglected.

```
LINEARSCALING linscal
End
```

More refined control is possible by using the full block key

```
LINEARSCALING
  CUTOFF_FIT epsfit
  OVERLAP_INT ovint
  PROGCONV progconv
  CUTOFF_COULOMB epsvc
  CUTOFF_MULTIPOLES epsmp
END
```

CUTOFF_FIT determines how many atom pairs are taken into account in the calculation of the fit integrals and the density fit procedure. If the value is too low, charge will not be conserved and the density fitting procedure will become unreliable. This parameter is relevant for the timings of the FITINT and RHOFIH routines of ADF.

OVERLAP_INT determines the overlap criterion for pairs of AO's in the calculation of the Fock-matrix in a block of points. Indirectly it determines what the cut-off radii for AO's should be. The value of ovint has a strong influence on the timing for the evaluation of the Fock matrix, which is very important for the overall timings. The default value for ovint is accint + 2 (typically 6). Again, a higher value implies a safer but slower calculation.

PROGCONV determines how the overall accuracy changes during the SCF procedure ('progressive convergence'). The idea is that one might get away with a lower accuracy during the initial SCF cycles, as long as the last cycle(s) is/are sufficiently accurate. The current default is that progconv has the value 0, which means that the accuracy in the beginning of the SCF is the same as in the rest of the SCF. This keyword is currently still in the testing phase, so we do not recommend changing its default value. The value of progconv determines how much lower the other parameters in the LINEARSCALING input block are at the beginning of the SCF than at the end.

CUTOFF_COULOMB determines the radii for the fit functions in the evaluation of the (short-range part of) the Coulomb potential. As the Coulomb potential may take a sizable amount of time, the value chosen for epsvc may influence the total ADF timing significantly as well. The default value for epsvc is accint + 4 (typically 8).

CUTOFF_MULTIPOLES determines the cut-offs in the multipole (long-range) part of the Coulomb potential. This term scales quadratically with system size, but has a small prefactor. In most cases, change in the epsmp value will not affect the CPU time significantly. The default value for epsmp is accint + 4 (typically 8).

9.4.7 All Points

ADF makes use of symmetry in the numerical integrations. Points are generated for the *irreducible wedge,* a symmetry unique sub region of space. Optionally the symmetry equivalent points are also used. This is achieved by setting the key

```
ALLPOINTS
```

The key has no argument. The CPU time increases roughly by a factor equal to the number of symmetry operators, and the results should be the same. This key is available only as a debugging feature, to check the correctness of certain symmetry related algorithms.

9.4.8 Full Fock

At every cycle in the SCF procedure the Fock operator is computed in all integration points. By default the *difference* with the values of the previous cycle are used to compute *changes* in the Fock matrix elements. This leads in general to better computational efficiency in two ways: 1) when all such difference values in a block of integration points are very small such a block is skipped in the calculation. 2) if the values are not negligible but still rather small, the contribution from such a block to matrix elements between basis functions with small overlaps are neglected.

With the key

```
FULLFOCK
```

this is turned off, so that the complete matrix elements are computed, no blocks are skipped and the neglect of matrix elements between functions with small overlaps (see also the key TAILS) is controlled solely by the function characteristics and precision requirements, not by the development of the SCF.

9.4.9 Electrostatic interactions from Fit density

By default the program tries to evaluate the electrostatic Coulomb interaction energy between the fragments in a molecule using the exact fragment charge densities. The implemented algorithm requires that all fragments are spherically symmetric. This is checked by the program by verifying that all fragments have been computed in atom symmetry. It that is not the case, an alternative method is applied, using the fitted charge densities of the atoms; this is an approximation with a small, but not insignificant error. The following key forces the program to apply the fit density approach even in the case of spherically symmetric fragments. This aspect applies only to the final bonding energy analysis, not to energy computations and their gradients within the automatic geometry optimizer. The purpose of this option is to simulate a previously existing situation where the electrostatic term in the bonding energy was computed from the fit density regardless of the fragments and their internal symmetries.

```
FITELSTAT
```

presence of this key in the input file triggers using the fit density.

9.4.10 Save info

Several types of information, gathered during the run, are lost on exit. The SAVE key allows you to prevent the removal of such information.

```
SAVE info
```

info A sequence of names separated by blanks or commas. save may occur any number of times in the input file. save turns save-info options on. Possible `info`:

TAPE10 File with numerical integration data: points and weights, values of functions (depends on direct-SCF options) and core densities and potentials.

TAPE11 File with fit integrals (STOFIT).

TAPE13 Check point file. This file is lost (by default) only upon normal program exit, i.e. a program-controlled termination (including a program-detected error condition leading to controlled exit). In all such cases all info on TAPE13 is also present on TAPE21. tape13 exists when the program crashes into a core dump for instance, in which case it is uncertain what the contents of TAPE21 will be. The save feature allows you to specify that TAPE13 is kept *also* upon normal exit."

TAPE14 Scratch file with numerical integration data, mainly pertaining to individual fragments.

Timing During an ADF calculation the program gathers a large amount of timing information about the performance of different program parts. It can be printed, at various levels of detail, on standard output (key PRINT). It can also be stored on TAPE21, for later inspection, in a section Timing."

RESTARTS

10.1 Restart files

When an ADF calculation terminates abnormally - not controlled by the program itself, for instance after a core dump due to some bug - there will usually be a file TAPE13, which serves as a checkpoint file. TAPE13 can be used to restart the calculation at a point not too far before the fatal condition occurred. It contains only data for the restart, but none of the special analysis data on TAPE21 that would be useful for analysis, to serve as fragment file, etc.

TAPE13 is upgraded during the calculation but discarded upon normal termination, namely when all relevant information has been saved on TAPE21. At that point all info that would have been on TAPE13 is present on TAPE21. If you wish to keep TAPE13 anyway - for instance because you plan a restart after normal termination and don't intend to keep the substantially bigger TAPE21 - you must use the SAVE key.

Upon normal (i.e. program-controlled) termination of a calculation, the TAPE21 result file can be used for restart purposes. When a crash occurs, however, chances are that TAPE21 has not correctly been closed and that its data structure is inconsistent: during the calculation large portions of TAPE21 are kept in memory rather than on file, and only at the point of final termination, all data is flushed to file.

General remarks

In all restart calculations a normal input file must be supplied (you can, for instance, simply take the original one), with a specification of the restart file added: the restart file does *not replace* the input file. From the program's point of view, it first reads the 'normal' input file and then inspects whether a restart file is present to replace some of the information read from input.

The concept of restarts in ADF is rather simple and primarily directed at increasing computational efficiency by providing cost-expensive data. The continuation run is to a large extent independent from the one that generated the restart file. The runtype, the choice of density-functional and other features in the Hamiltonian, precision of numerical integration, thresholds on convergence, et cetera are all determined solely from the input file for the new run: no such data is read from the restart file. Most input items should, therefore, be supplied in the restart run again, even if it is a direct continuation of a previous calculation: omission implies using the standard defaults, which are not necessarily the settings of the calculation that generated the restart file.

Even the key ATOMS with the list of atomic coordinates must be supplied again: the program needs the information herein to deduce what fragments are used, which coordinates are free or frozen respectively in an optimization, etc. The coordinate *values* may be supplied with the restart file and these will then overwrite those specified in the input file.

Obviously, the two runs cannot be completely unrelated. To let the restart data make sense the runs should correspond to the same molecule (i.e. its general definition in terms of fragment building blocks). The program does not check all aspects related to this and certain abuses will therefore survive the internal tests, but will surely lead to some error later on: it is the user's responsibility to ensure that the restart data match the calculation one has in mind.

Interdependencies between data read from the restart file (rather than from input or fragment files) and other items imply that some input keys and some options to specific keys may be inaccessible when restart data are provided. In

most cases supplying such inaccessible input options will simply be ignored; in some cases a warning is issued or an error abort occurs.

A restart file supplies data from a previous run that might be useful in the current one. The applications are (combinations are possible):

- Get a better start in the (first) SCF procedure by providing the electronic charge density (in the form of fit coefficients) from the preceding run,

- Continue an optimization by supplying the latest geometry (coordinates) from a previous run via the restart file (rather than typing them in),

- Get faster geometry convergence by supplying a Hessian,

- Breaking large jobs (Linear Transits, Frequencies) in smaller ones, each time doing a part and passing this on to the continuation run.

WARNING: The SCF and optimization procedures use *history* to improve convergence behavior. Most of such history information is not stored on a restart file. As a consequence, a restart may not continue exactly as the original run would have done if it hadn't terminated. In a SCF restart, for instance, the DIIS procedure has to rebuild the information. The same holds for geometry optimizations, although history plays usually not a very big role there.

10.2 The restart key

The name of the restart file must be provided with the key RESTART (see below). A list of data items is read from the file (if present on the file and only as far as significant for the new run) and used unless their usage is explicitly suppressed by the user.

Simple usage:

```
RESTART
  File restartfile
End
```

restartfile The name of a file with restart data. The path (absolute or relative) to the file must be included if the file is not local to the directory where the calculation executes. In most cases it will be a TAPE21 file from an ADF calculation, but this is not necessary. It may be any file - constructed by the user for instance - provided it has the right structure. It must be a kf file and the data to be used must be stored in sections and under variable names as defined below, which is exactly how such data are generated by a normal ADF run on TAPE21 or on the checkpoint file TAPE13. Note: the filename must not be one of the standard filenames used internally by the program, such as TAPE21, TAPE13 etc. Generally: don't use a name like TAPEnn where nn is a two-digit number.

One can also specify a list of restart options:

```
RESTART
  File restartfile
  optionlist
  optionlist
  ...
end
```

This is the list of possible options:

noSCF Do not use any fit coefficients from the restart file as a first approximation to the (fitted) SCF density for the new calculation. Instead, the sum-of-fragments density will be used, as in a non-restart run. Note, typically noSCF should be used in combination with noORB.

noORB Do not use orbitals from the restart file.

nogeo Do not use the geometry - Cartesian, Z-matrix, etc. coordinates - from the restart file.

nohes Do not use any Hessian from the restart file.

SPINFLIP atomnumbers See the separate *section about the spin-flip method* (page 50) for converging broken-symmetry systems.

Note: in the continuation of a Linear Transit, IRC or Frequencies run, geometric data are read from the restart file and will be used: the option nogeo is ignored. In a continued Frequencies run the input coordinates (key ATOMS) must be correct (i.e. the equilibrium geometry). In a continued LT or IRC run, the input coordinate values from atoms are ignored (but they must be supplied to give the program a preliminary count of atoms and fragments involved).

10.3 Structure of the restart file

All data that may be retrieved from the restart file must be stored in a specific location on the restart file. If you're simply using a TAPE21 result file or a TAPE13 checkpoint file you don't need to bother about this: ADF has put all data in the right place; the following discussion is primarily for those who want to manipulate the restart file or even construct one themselves.

Since the restart file must be a kf file, the location of the data is of the form Section%Variable, specifying the section and the variable name. The section and variable names are case sensitive. See the utilities document for general information about kf files.

If the specified variable is not present in the specified section on the restart file - or if there is no such section at all - the data is not used, usually without an error message. In some cases a few global tests are carried out on the retrieved data; if they fail the tests the data are not used and a warning - in some cases an error abort - may be issued by the program.

KF files are binary files and so are the TAPE21 result file, the TAPE13 checkpoint file and generally any restart files. If you wish to edit and modify the contents, or just inspect them, the standard KF utilities can be used. Apply pkf to get a survey of the sections and variables on the file, dmpkf to get a complete ASCII version of the file and udmpkf to transform an ASCII version - presumably edited and modified - back into binary format. Please consult the Scripting Section for further information about the standard KF utilities.

Data on the restart file

Follows a survey of all data items that the program may search for on the restart file.

SCF data

Fit%coef_SCF The fit-expansion of the charge density to be used as start-up for the next SCF. Without these restart fit data the first SCF will start from the (fitted) sum-of-fragments charge density.

Fit%coef_FreqCenter Only in a Frequencies run: the fit-expansion of the SCF-converged equilibrium geometry. It usually helps to get a somewhat better start-up of the SCF in displaced geometries.

If the noSCF option is used to the restart key, any Fit%coef_? data on the restart file are ignored.

Coordinates

Geometry%xyz Cartesian atomic coordinates. The option nogeo suppresses using such data. In a Frequencies or continued Linear Transit run, they may be read but will be ignored (i.e. replaced by other coordinates data from the restart file, see below).

In most applications, when coordinates are read (and used) from the restart file, only *Cartesian* coordinates are retrieved and the corresponding Z-matrix values are computed from them, using the Z-matrix *structure* defined in the atoms data block. This is one of the reasons why the ATOMS key must be used even when the atomic coordinates are supplied on the restart file.

Hessian

`GeoOpt%Hessian_CART`

`GeoOpt%Hessian inverted_CART`

`GeoOpt%Hessian_ZMAT`

GeoOpt%Hessian inverted_ZMAT All these four varieties are searched for if the new run searches for a restart Hessian matrix at all, that is: in an optimization, Linear Transit or Transition State search. As the names should suggest these variables stand for the Hessian, respectively the inverse of the Hessian in Cartesian or z-matrix coordinates. In all cases the full square matrix must be present, with dimension the number of atomic coordinates, 3 times the number of atoms. This holds also for z-matrix coordinates. The 6 dummy coordinates play no role, the corresponding matrix elements in the Hessian should be zero. The order of atoms is the same as in the input.

If a Hessian is searched for on the restart file, all four possibilities above are tried and the first one found is used, the other ones being ignored. The order in which they are tried is:

If the current run uses *Cartesian* coordinates as optimization variables, then first the two cart varieties are tried, and vice versa for *z*-matrix* optimization.

In a minimization (simple optimization or Linear Transit) first the *inverted* variety is tried; in a Transition State search the normal (not inverted) Hessian is looked for first.

Note: If a z-matrix Hessian is retrieved from the restart file the program will use the underlying z-matrix *structure* to derive a Cartesian Hessian from it. In such case the restart file must also contain:

GeoOpt%kmatrix The z-matrix structure (references to the atoms in this matrix assume the ordering of atoms as used internally by the program).

Note: the kmatrix on the file need not be identical to the kmatrix used in the current calculation. In fact, the current calculation may not even have a z-matrix structure.

Transition State

In a continued TS run the program retrieves, apart from general geometry *optimization* data such as the Hessian - see above - only the latest TS search vector: the eigenvector of the (approximate) Hessian that points to the Transition State. All other TS-specific data are input-determined with corresponding defaults.

The TS search vector is stored in:

TS%mode to follow A list of atomic coordinates (Cartesian or Z-matrix, depending on the type of optimization variables used. The underlying list of atoms has the atoms not necessarily in the order in which they have been given in input: rather they are grouped together by atom type.

Linear Transit

In a continued Linear Transit (LT) calculation the continuation run proceeds from where the previous run stopped. The total number of points by which the transit is scanned, the current point (its index and the Cartesian coordinates), the accumulated results of completed points on the transit etc. are copied from the restart file. If the restart file contains a section LT, then all relevant data must be present on it and correct (i.e. matching those of the current run: same number of LT *parameters*, and of course the same molecule.

LT%nr of points The number of points by which the LT is scanned; this is identical to the Fortran variable ltimax in the code. The value on the restart file applies in the calculations and overwrites any input/default value (see the subkey *lineartransit* of the geometry block)

lt%current point Index of the current LT scan point. This is where the program will continue. In a non-restart LT run, this index initializes at 1.

LT%Energies An array with energy values, one for each LT point. When the LT run is completed, this array allows you to map out the energy along the LT path. The values for the completed LT points are stored on the restart file. This size of the array on the restart file must (at least) be the total number of points on the complete path.

LT%Parameters Initial and final values for the LT parameters, which describe roughly the path (all other coordinates may be optimized at each point, depending on other input keys). The values from the restart file overwrite input values. The input values should be supplied, however, as if it were a non-restart run.

LT%atmcrd zmat if a z-matrix structure is available for the molecule, cart otherwise. This is used to control printing of results. It does not define the type of optimization variables: see the next item.

LT%geocrd zmat or cart: the type of optimization variables. This defines in which type of coordinates the LT parameters are defined and any optimization of other coordinates takes place.

LT%xyz Cartesian coordinates for *all* LT points: 3*atoms*ltpoints. The size of the array must conform to this. Only the values of the completed LT points *and* those of the current point are relevant. Those of the current LT point are used as initial coordinates to start the current run.

LT%zmatrix Same for the Z-matrix coordinates. They should match the Cartesian coordinates for the completed LT points (this is not checked). Those for the current LT point will be recomputed from the current *Cartesian* coordinates.

IRC

In a continued Intrinsic Reaction Coordinate (IRC) calculation, the continuation run processes the path(s) as specified in input. Any info for such path(s) on the restart file will then be used to continue from there. If the restart file contains the relevant IRC sections, see below, then all relevant data must be present on it and correct (i.e. matching those of the current run.

The sections on file pertaining to the IRC are:

IRC: this section contains information about the central (TS) point, which variables are optimized in each of the IRC points, the connection matrix defining the z-matrix structure, etc.

IRC_Forward and IRC_Backward: these sections contain the data of the two paths from the Transition State down to the two adjacent local energy minima: for each point the distance from the previous point and the local curvature and molecular properties such as energy, atomic charges and dipole moment.

LT%nr of points The number of points by which the LT is scanned; this is identical to the Fortran variable ltimax in the code. The value on the restart file applies in the calculations and overwrites any input/default value (see the subkey *lineartransit* of the geometry block)

LT%current point Index of the current LT scan point. This is where the program will continue. In a non-restart LT run, this index initializes at 1.

lt%Energies An array with energy values, one for each LT point. When the LT run is completed, this array allows you to map out the energy along the LT path. The values for the completed LT points are stored on the restart file. This size of the array on the restart file must (at least) be the total nr of points on the complete path.

lt%Parameters Initial and final values for the LT parameters, which describe roughly the path (all other coordinates may be optimized at each point, depending on other input keys). The values from the restart file overwrite input values. The input values should be supplied, however, as if it were a non-restart run.

lt%atmcrd zmat if a z-matrix structure is available for the molecule, cart otherwise. This is used to control printing of results. It does not define the type of optimization variables: see the next item.

lt%geocrd zmat or cart: the type of optimization variables. This defines in which type of coordinates the LT parameters are defined and any optimization of other coordinates takes place.

lt%xyz Cartesian coordinates for *all* LT points: 3*atoms*ltpoints. The size of the array must conform to this. Only the values of the completed LT points *and* those of the current point are relevant. Those of the current LT point are used as initial coordinates to start the current run.

LT%zmatrix Same for the Z-matrix coordinates. They should match the Cartesian coordinates for the completed LT points (this is not checked). Those for the current LT point will be recomputed from the current *Cartesian* coordinates.

Frequencies

In the continuation of a Frequencies calculation all Frequencies-related data are retrieved from the section Freq on the restart file. (SCF fit data are, as always, retrieved from the section Fit). A fairly large number of items will be read and must all be present (*if* a section Freq is present in a restart file supplied to a Frequencies run). Technical parameters such as the type of numerical differentiation, size of displacements etc. are read from the restart file. Any input specifications are ignored.

Freq%kountf Counter of number of geometries completed. In a non-restart run this is initialized at zero; in a restart it is read from the file.

Freq%nraman Flag for RAMAN calculations

Freq%numdif 1 or 2: defines numerical differentiation used to compute the force constants from the gradients in slightly displaced geometries (by 1-point or 2-point differentiation).

Freq%disrad Size of displacement for Cartesian or bond-length displacements.

Freq%disang Size of angular (bond angle, dihedral angle) displacements.

Freq%atmcrd zmat or cart: specifies whether a z-matrix structure is present. This does not define the type of displacement coordinates, see the next item.

Freq%geocrd Type of coordinates in which the displacements are carried out: zmat or cart

Freq%nfree Number of free and independent displacement variables.

Freq%idfree References from the atomic coordinates (in internal order) to the independent displacement variables.

Freq%all freedoms (logical) flags whether or not the complete energy surface is scanned around the equilibrium or only part of the internal degrees of freedom are used.

Freq%xyz equilibrium coordinates (internal order of atoms).

Freq%kmatrix Z-matrix structure. Pointers are indexed by and refer to atoms in the internally used order.

Freq%zmatrix Z-matrix coordinates of the equilibrium geometry (internal ordering of atoms).

Freq%rigids 6 rigid motion vectors (one may be zero, in case of a linear molecule). Each vector has as many components as there are atomic coordinates. The values correspond to the internal ordering of atoms.

Freq%xyz displaced Cartesian coordinates of displaced geometry to carry out now. In a non-restart run this would be the equilibrium geometry.

Freq%zmatrix displaced Similar for the Z-matrix coordinates.

Freq%Dipole previous dipole vector (3 components) for the last geometry handled.

Freq%Dipole dipole at the equilibrium geometry.

Freq%Dipole derivatives Derivatives of the dipole wrt atomic coordinate displacements.

Freq%Gradients previous Energy gradients (derivatives wrt atomic coordinate displacements) in the last handled geometry.

Freq%Force constants Matrix of force constants. This is, together with the *Dipole derivatives* the final quantity to compute. At each cycle of the Frequencies data are added to it. Upon completion of the Frequencies cycles the frequencies and normal modes are computed from it. Together with the dipole derivatives it then also yields the InfraRed intensities.

RECOMMENDATIONS AND TROUBLESHOOTING

11.1 Recommendations

11.1.1 Precision

The quality of the calculation, *given* the selected model Hamiltonian - density-functional, relativistic features, spin-restricted/unrestricted... - is determined to a large extent by several technical precision parameters.

The most significant ones are:

Basis set Obviously, the quality of the basis set may have a large impact on the results. As a general rule, minimum and almost-minimum basis sets (types SZ and DZ) may be used for pilot calculations, but polarization functions should be included (DZP, TZP) for more reliable results.

SCF convergence The self-consistent-field (SCF) and geometry optimization procedures terminate when convergence criteria are satisfied. If these are set sloppy the results may carry large *error bars*. The default SCF convergence tolerance is tight enough to trust the results from that aspect. However, when the SCF procedure encounters severe problems an earlier abort may occur, namely if a secondary (less stringent) criterion has been satisfied (see the key SCF). Although this still implies a reasonable convergence, one should be aware that for instance the energy may be off by a few milli-Hartree (order of magnitude, may depend quite a bit on the molecule). It is recommended that in such cases you try to overcome the SCF problems in a secondary calculation, by whatever methods and tricks you can come up with, rather than simply accept the first outcomes. Note: in a geometry optimization the SCF convergence criteria are relaxed as long as the geometry optimization has not yet converged. This should generally not affect the final results: the SCF density and hence the energy gradients may be somewhat inaccurate at the intermediate geometries, but since these are not a goal in themselves the only concern is whether this might inhibit convergence to the correct final geometry. Our experiences so far indicate that the implemented procedure is reliable in this aspect.

Geometry convergence This is a far more troublesome issue. Three different types of convergence criteria are monitored: energy, gradients and coordinates. The energy does not play a critical role. Usually the energy has converged well in advance of the other items. The coordinates are usually what one is interested in. However, the program-estimated uncertainty in the coordinates depends on the Hessian, which is not computed exactly but estimated from the gradients that are computed in the various trial geometries. Although this estimated Hessian is usually good enough to guide the optimization to the minimum - or transition state, as the case may be - it is by far not accurate enough to give a reasonable estimate of force constants, frequencies, and as a consequence, neither of the uncertainties in the coordinates. An aspect adding to the discrepancy between the Hessian-derived coordinate-errors and the true deviations of the coordinates from the minimum-energy geometry is that the true energy surface is not purely quadratic and using the Hessian neglects all higher order terms. The gradients provide a better criterion for convergence of the minimizer and therefore it is recommended to tighten the criterion on the gradients, rather than anything else, when stricter convergence than the default is required. The default convergence criteria, in particular for the gradients, are usually more than adequate to get a fair estimate of the minimum energy. Tighter convergence should only be demanded to get more reliable coordinate values

(and in particular, when the equilibrium geometry needs to be determined as a preliminary for a Frequencies run).

Numerical integration accuracy The key BeckeGrid key block (or alternatively the old INTEGRATION key block) determines the numerical precision of integrals that are evaluated in ADF by numerical integration, primarily the Fock matrix elements and most of the terms in the gradients. In addition the integration settings also determine several other computational parameters. The demands on numerical integration precision depend quite a bit on the type of application. The SCF convergence seems to suffer hardly from limited integration precision, but geometry convergence does, especially when tight convergence is required and also in transition state searches, which are generally more sensitive to the quality of the computed energy gradients. An extreme case is the computation of frequencies, since they depend on differences in gradients of almost-equal geometries. Frequency calculations on molecules with sloppy modes suggest that a BeckeGrid of "good" quality may be required. Note: a large integration value implies that a lot more points will be used in the numerical integrals, thereby increasing the computational effort (roughly linear in the number of points).

11.1.2 Electronic Configuration

Not specifying occupation numbers in input will *not* automatically result in the computational of the ground state. It may even lead to non-convergence in the SCF and/or in the determination of minimum-energy geometries or transition states. Therefore: whenever possible, specify occupation numbers explicitly in input (key OCCUPATIONS)!

Misunderstanding results of a calculation may easily result from a lack of awareness of how ADF treats the electronic configuration, which orbitals are occupied and which are empty. Unless you specify occupation numbers in input they will be determined from the aufbau principle but only during the first few SCF cycles. Thereafter the distribution of electrons over the different symmetry representations is frozen (see the key OCCUPATIONS, options AUFBAU and aufbau2). If at that point the potential has not yet sufficiently relaxed to self-consistency the *final* situation may be non-aufbau.

A related aspect is that the *ground state* does not necessarily *have* an aufbau occupation scheme. In principle, different competing electronic states have to be evaluated to determine which has the lowest total (strongest bonding) energy.

Check output always carefully as to which orbitals are occupied. In general, whenever possible, supply occupation numbers in input. Be aware that the automatic choice by the program may in a Geometry Optimization result in different configurations in successive geometries: the automatic assessment by the program will be carried out anew in each SCF procedure. If competing configurations with comparable energies have different equilibrium geometries, the geometry optimization has a high failure probability. The gradients computed from the SCF solution of a particular configuration drive the atoms in a certain direction, but in the next geometry, when the program re-determines the occupations and finds a different configuration, the resulting gradients may drive the atoms in another direction.

See the keys CHARGE and OCCUPATIONS for user-control of occupation numbers.

Spin-unrestricted versus spin-restricted, Spin states

If your molecule has unpaired electrons, you should run an unrestricted calculation, in principle. However, if this exhibits convergence problems (or if you simply want to save time: an unrestricted calculation takes a factor 2 more CPU time and data storage), you may consider to do it in two steps. First, run a spin-*restricted* calculation. Then perform a spin-unrestricted calculation using the restricted TAPE21 as a restart file. In the follow-up calculation you should specify the precise occupation numbers for the state you're interested in, *and* use the SCF input key to specify *only one* SCF cycle (iterations=1). This prohibits convergence (so you keep the converged *restricted* orbitals) and gives you a fairly adequate approximation to a converged unrestricted result. See also the H2 example run for a discussion in the Examples document.

An unrestricted calculation does not necessarily yield the multiplet configuration (triple, doublet ...). This is a rather complicated matter, see the discussion on multiplet states, key SLATERDETERMINANTS.

11.1.3 What basis set should I use in ADF?

This question is hard to answer in general, but a few general suggestions can be made.

ZORA or non-relativistic calculation?

The first question to ask is: am I going to do a ZORA calculation to include scalar relativistic effects? This is recommendable for systems containing heavy nuclei, but there is no objection to doing a ZORA calculation for a system with light atoms only, as the required CPU time does not increase very much if ZORA is used.

If you are doing a ZORA calculation, you will need the ZORA basis sets which can be found in $ADFHOME/atomicdata/ZORA. You may also use the all electron basis sets from the ET or AUG directory, but be aware that these were optimized to non-relativistic calculations. Currently the ZORA basis sets cover the entire periodic table and besides all electron basis sets offer a choice of frozen cores. At present the ZORA directory does not contain basis sets with very diffuse functions, which may be required in calculations for hyperpolarizabilities or high-lying excitation energies, but for the lighter elements (H-Kr) you can certainly use the all-electron basis sets from the ET or AUG directory. Warning: in a ZORA calculation use only the frozen core basis sets coming from the $ADFHOME/atomicdata/ZORA directory, or use all electron basis sets.

If you do not use ZORA, your basis sets should come from the directories SZ, DZ, DZP, TZP, TZ2P, or one of the ET or AUG basis sets. For many of the heavy elements only ZORA basis sets are available, but for such elements it would be inadvisable to do non-relativistic calculations anyway. For light elements the ZORA and normal basis sets should be identical except for the description of the frozen core. Usually the ZORA basis sets contain much steeper basis and fit functions to accurately describe the core region.

Large or small molecule?

For standard calculations (energies, geometries, etc.) we recommend the following hierarchy of basis sets: SZ < DZ < DZP < TZP < TZ2P < TZ2P+ < ET/ET-pVQZ < ZORA/QZ4P where the largest and most accurate basis is on the right. Not all basis sets are available for all elements. For small negatively charged atoms or molecules, like F^- or OH^-, basis sets with extra diffuse functions are needed, like they are available in the AUG or ET/QZ3P-nDIFFUSE directories. For example, the standard basis sets, or even the large ZORA/QZ4P basis set will often not be large enough for the accurate calculation of such anions.

In general it is advisable to use the best basis set that you can afford to use in terms of CPU time and memory. If you want to optimize the geometry or calculate the atomization energy of a diatomic molecule there is little reason not to use the very large ZORA/QZ4P basis, or (for light elements) a similarly large ET basis (we recommend the ET-pVQZ basis). If you are studying a molecule with 100 atoms or more, the use of such large basis sets does not only become prohibitive because of the required CPU time and memory, but it also is much less needed than for smaller systems. In medium-sized or large molecules even the moderately large basis sets will prove to be quite adequate because of the effect of basis set sharing. Each atom profits from the basis functions on its many neighbors. Additionally, if a large basis contains diffuse functions, linear dependency problems may occur. See also the input key DEPENDENCY. In many cases basis DZ or DZP will give acceptable accuracy for calculations on large systems. If you are inexperienced it may be prudent to test a few different basis sets to get a feel for the size of basis set effects. To get a rough idea for the size of various basis sets, we mention here the number of functions for all-electron basis sets from the directories ZORA/SZ up to ZORA/QZ4P. For carbon, the number of functions is 5 (basis ZORA/SZ), 10 (DZ), 15 (DZP), 19 (TZP), 26 (TZ2P), 43 (QZ4P). The same numbers for hydrogen are: 1 (SZ), 2 (DZ), 5 (DZP), 6 (TZP), 11(TZ2P, 21(QZ4P). These numbers arise because ADF uses 'pure' d and f functions. In other words, 5 instead of 6 d functions are used and 7 instead of 10 f functions. Note that especially the jump from TZ2P to QZ4P is quite steep.

In an overgeneralizing fashion we can state that the single zeta basis SZ is hardly ever sufficient to get more than a qualitative picture and should be used only when larger basis sets are not affordable. The double zeta basis DZ performs already much better and may give quite reasonable results, for example in geometry optimizations on large molecules. However, in more subtle situations, for example if hydrogen bonds are important, it is advisable to use

at least one set of polarization functions. This is the double zeta polarized DZP basis set. Basis set TZP extends the valence space (but not the core space which remains double zeta) to triple zeta. In basis TZ2P an additional polarization function is added. For example, hydrogen gets a d polarization function in addition to its p polarization function and carbon gets an f polarization function on top of a d polarization function. Several tests have shown that often the second polarization function is of more use when it has an l value one higher than the first polarization function. This is reflected in the choice just described. If another polarization function is needed it is usually best to add another one of the lowest l-value (2p+1d for hydrogen, 2d+1f for carbon). This choice has been made in the ET basis ET-QZ3P. There, sometimes even three d polarization functions were added, for example 3 p functions for Be, and 3 d functions for S. The reason for this is that S can occur in hypervalent species such as SF_6 , which put special demands on the basis set. In the case of Be, the unoccupied p level is so close in energy to the occupied ones that it is sometimes called a valence level. Semantics aside, it is clear that a proper description of the p level of Be is very important and it is therefore not strange to add a third p function. In general, the unoccupied levels for the atoms on the left side of the periodic table are close to the occupied ones. This makes it necessary to add a few extra functions for the lowest unoccupied levels in order to get a description which corresponds to the general level of accuracy one expects from the hierarchical basis set names SZ-TZ2P. The basis set quality for a particular subdirectory is now rather uniform throughout the periodic system. At the same time we have attempted to increase the number of functions in a systematic fashion so that each element is described by at least as many functions of a particular l value as its predecessor.

The ZORA/QZ4P basis sets might be roughly described as core triple zeta, valence quadruple zeta, with 4 polarization functions (2 d and 2 f functions for C, 2 p and 2 d for H). The fit sets corresponding to these basis sets are also much larger than the fit sets found in basis sets SZ-TZ2P. If one has doubts about the adequacy of a fit set for a certain element, this can be tested by replacing its fit set by the large one from the QZ4P directory, see the subkey FitType of the key BASIS. In the ZORA/QZ4P basis sets, the choice for the exponents of the polarization functions was done in a systematic, but somewhat hand-waving manner. For this reason the exponents were always rounded to half integers. Also the geometric mean of the exponents usually does not coincide with the choices made in directories SZ-TZ2P and the ET basis sets. However, the fact that two polarization functions (with reasonable exponents) are present instead of a single one is far more important. A reasonable intermediate basis set, in size between TZ2P and QZ4P might be envisaged in which a single polarization function is added, as described above. This is roughly the choice for the polarization functions made in the ET directory ET-QZ3P.

Frozen core or all-electron?

In general we recommend the use of frozen core basis sets if available. Especially for the heavier atoms the number of functions is much smaller than for their all-electron counterparts. Our tests indicate that the error made by invoking the frozen core approximation is usually clearly smaller than the difference with respect to slightly higher quality basis sets. For the ZORA/QZ4P basis sets, only all-electron basis sets are available as these are intended for near basis set limit calculations only in which the CPU time is not a major concern.

Geometry optimizations involving atoms with a too large frozen core may give rise to numerical problems. In such cases it is recommendable to use a smaller frozen core. In previous occurrences we have removed such basis sets.

For accurate results on properties like nuclear magnetic dipole hyperfine interactions (ESR), nuclear quadrupole coupling constants, and chemical shifts (NMR), all electron basis sets are needed on the interesting atoms. For such properties tight functions might be necessary for high accuracy, especially in a ZORA calculation.

Diffuse functions needed?

For small negatively charged atoms or molecules, like F^- or OH^- , basis sets with extra diffuse functions are needed, like they are available in the AUG or ET/QZ3P-nDIFFUSE directories. For example, the standard basis sets, or even the large ZORA/QZ4P basis set will often not be large enough for the accurate calculation of such anions.

For accurate results on properties like polarizabilities, hyperpolarizabilities, and high-lying excitation energies, also diffuse functions are needed. This is especially true for calculations on small molecules. In larger molecules the

nature of the relevant virtuals is much more 'molecular', much less Rydberg-like, so that the normal basis sets may be sufficient. Basically all properties calculated through the RESPONSE keyword may require diffuse functions. If you use the EXCITATIONS keyword, the necessity of diffuse functions depends on the type of excitation you are interested in. The lowest excitations do not require diffuse functions, but Rydberg excitations do.

In case of diffuse basis functions the risk of linear dependency in the basis increases. This can be checked, and corrected for with the DEPENDENCY keyword. It is recommended to use this keyword for all calculations involving diffuse functions. A good default setting is

DEPENDENCY bas=1d-4

However, it may be advisable to experiment a bit with the parameter, especially if many linear dependent combinations of AOs are removed. Using too many diffuse functions on a large molecule will lead to insurmountable numerical problems. In such a case it is not only useless, but even harmful, to add many diffuse functions.

In the previous release only some basis sets were provided which contained diffuse functions. These were gathered in the directory Vdiff. Now several ET basis sets have been developed for the elements up to Ar containing some or many diffuse functions. We recommend to use these instead of the Vdiff directory. Most of these basis sets are quite large and not very suitable for large molecules.

In ADF2005.01 augmented basis sets were added in the AUG directory, especially devised for use in in TDDFT calculations, such that one can do a reasonable accurate calculation of excitation energies, with a relatively small basis set, see D.P. Chong [*322* (page 925)].

Normal or even-tempered basis?

For normal calculations (these form the vast majority) we recommend the use of the optimized basis sets in the directories SZ-TZ2P and, for ZORA calculations, ZORA/SZ-QZ4P. These should be sufficient in accuracy for even very demanding users and are available for the entire periodic system (in the case of the ZORA basis sets). They are also available with a frozen core variety, which saves much CPU time.

The ET basis sets on the other hand are available only in all-electron form at the moment. Furthermore, most are pretty large (larger, but also better than TZ2P). Additionally, relatively large basis set superposition errors were detected for molecules containing atoms in the row K-Kr. For this reason we only recommend ET basis sets for the elements H-Ar at the moment. There they have yielded quite nice, near basis set limit, results for the G2 test set. For these light elements the ET basis sets can be comparable in quality to the ZORA/QZ4P basis, even though it is smaller. The ET basis sets are considered to be especially useful when diffuse functions are required. In that case it is very easy to adapt the original ET basis and fit set. The utilities provided for this in ADF will be described below in an Appendix. The ET basis set utilities will also prove useful for users who want to experiment with making their own basis sets, or who have very special demands on the basis and fit. The provided utilities automate much of the work needed to make new atomicdata files.

What accuracy do the basis sets give?

Tests on many diatomics were performed to test the various basis sets. We now document the results of some of these tests, in order to give a feeling for the quality that can be obtained from the various basis sets. See also van Lenthe and Baerends [*320* (page 925)].

Tests for non-relativistic calculations on 36 diatomics containing oxygen, namely the oxides of the first 36 elements (H-Kr). All-electron basis sets were used. The ZORA/QZ4P basis set was used to define the basis set limit result. The numbers in the table refer to bonding energies in eV. Differences were taken between the QZ4P results and the results in smaller basis sets. By construction, the errors in the QZ4P column are zero.

	QZ4P	DZ	DZP	TZP	TZ2P
Average error	0.0	1.33	0.39	0.18	0.06
Average absolute error	0.0	1.33	0.39	0.18	0.06
Maximum error	0.0	2.84	1.07	0.41	0.13
Worst case	all	SO	BeO	FO	O_2

A few comments are in order to explain this table.

The oxides were used as a small test set because their equilibrium bond lengths are known in many cases. Also, they have a large influence on the electronic structure of the molecule, so that they also test the adequacy of the polarization functions.

The errors in the small basis sets are systematic, because the isolated atoms are described reasonably well, but the molecular energy is not deep enough. For this reason the average errors and average absolute errors are (nearly) always equal.

Test calculations on 100 diatomics containing oxygen, using all-electron ZORA basis sets. Many basis sets for (very) heavy elements are included here, which could not be included in the table above. The numbers have the same interpretation as above and are again in eV.

	QZ4P	DZ	DZ	TZP	TZP	TZ2P	TZ2P
	ae	fc	ae	fc	ae	fc	ae
Average error	0.00	0.95	1.07	0.20	0.20	0.05	0.05
Average absolute error	0.00	0.98	1.07	0.20	0.21	0.05	0.05
Maximum error	0.00	2.86	2.83	0.74	0.74	0.19	0.17
Worst case	all	SO	SO	OgO	OgO	ThO	OgO

Again we place a few comments on these frozen core and all-electron results.

The trends are very similar to those in the previous table for the lighter elements.

The frozen core results are very satisfactory, as they are very close to the results with the corresponding all-electron basis sets. The error introduced by the frozen core approximation is typically much smaller than the one introduced by basis set incompleteness.

The average errors are quite comparable to those from the previous table. The heavier elements do not seem to be much more difficult than the lighter ones.

For heavy elements no reliable ET basis set is yet available for comparison.

More results, all-electron, non-relativistic on roughly 140 different diatomics at experimental or 'reasonable' equilibrium geometries.

	QZ4P	DZ	TZP
Average error	0.00	0.89	0.11
Average absolute error	0.00	0.89	0.11
Maximum error	0.00	2.84	0.32
Worst case	all	SO	O_2

Only the non-relativistic basis sets DZ and TZP are fairly complete for heavier elements.

Also for these general diatomics (not just oxides) the average and maximum errors have decreased substantially, especially for basis TZP.

Same table, but now for frozen core basis sets. In all these tests the smallest frozen core files were employed (i.e. the largest basis).

	QZ4P	DZ	TZP
Average error	0.00	0.73	0.13
Average absolute error	0.00	0.75	0.16
Maximum error	0.00	2.87	1.80
Worst case	all	SO	ThO

The frozen core approximation has little influence on the accuracy for the new basis DZ, but a somewhat larger effect on the new basis TZP. This is especially due to certain worst cases, such as ThO.

ZORA, all electron, over 240 diatomics

	QZ4P	DZ	TZP	TZ2P
Average error	0.00	0.70	0.11	0.02
Average absolute error	0.00	0.70	0.11	0.03
Maximum error	0.00	2.83	0.44	-0.16
Worst case	all	SO	I_2	Cr_2

The average error goes down very nicely from 0.70 to 0.11 to 0.03 eV when going from DZ to TZP to TZ2P. The average error in basis TZ2P is clearly below 1kcal/mol (the famous chemical accuracy). Errors due to deficiencies in current xc functionals are still much larger than this. As a consequence, the ZORA/TZ2P basis will be more than adequate for all standard calculations.

It is to be expected that these conclusions will not dramatically change if larger test molecules are used. Also for geometry optimizations the improved basis sets SZ-TZ2P and ZORA/SZ-TZ2P should be more than sufficient for all standard cases. The ZORA/QZ4P can be considered a very safe (though expensive) option for basis set limit calculations.

11.1.4 Relativistic methods

The ZORA relativistic approach is often superior and in other cases at least similar to the older Pauli method. In particular for all-electron calculations generally, and for very heavy elements even within the frozen core approach, the Pauli method may exhibit significant shortcomings. This is mostly due to the variational instability of the Pauli formalism in the deep-core region near the nucleus. The bigger the basis set and the smaller the frozen core, the more likely this will show up, while generally speaking you might be tempted to use smaller cores and bigger basis sets to *improve* your results. The ZORA approach does not suffer from these problems and is, therefore, highly recommended over the Pauli formalism.

11.2 Troubleshooting

This chapter contains hints to help you solve some problems and comments on frequently asked questions.

11.2.1 License file corrupt

You may find that, after having installed the license file, the program still doesn't run and prints a message like 'your license file is corrupt'. To explain how this may come about, and how you overcome this, a few words on license files.

Each license file consists of pairs of lines. The first of each pair is text that states, in more or less readable format typical aspects such as an expiration date, the version number of the software and so on. The second line contains the same information in encrypted format: a (long) string of characters that seem to make little sense. The program reads the license file and checks, with its internal encrypting formulas, that the two lines match. If not, it stops and prints the 'corrupt' message. So, there are two common reasons why it may happen to you:

- You are using a license file for another version of the software than your executables correspond to. Newer (major) releases may contain a different encrypting formula, so that the match in old license files is not recognized anymore. So, please verify that your license file and executable belong to the same major release.

- More likely: the license file as it has been created has been modified in some way. Sometimes, people inspect it and 'clean it up' a little bit, for instance by removing 'redundant' spaces, or by making some other 'improvements'. Unfortunately, every such modification will destroy the encryption match and lead to the 'corrupt' error. Most of the times, however, the reason lies in the mailing system, by which the license file has been sent to you.

If the encrypted line is rather long, the mailer may have cut it in two shorter lines. To verify (and correct) this: edit the license file and see if it consists of pairs of lines as described above. If not, re-unify the broken lines and try again.

- Finally, the problem may lie in your O/S, which may have inserted additional hidden CR characters (Carriage-Return) into the license file. You can remove them with our fix_license utility (in $ADHFOME/Install), see the Installation manual.

11.2.2 Recover from Crash

A calculation may terminate in two ways: controlled or uncontrolled. Controlled termination includes cases where the program itself detects an error and decides that continuation of the calculation is impossible or pointless. In all such cases the standard exit routine is executed, resulting in an output section with some final information. This also ensures that the general result file TAPE21 is closed properly and all relevant information flushed to it.

Uncontrolled termination may occur, for instance when some bug causes the program to divide by zero, violate memory access restrictions, etc. Usually this leads to an immediate abort of the program by the Operating System and hence loss of control by the program. In such situations the information on TAPE21 may be incomplete because some of the data are kept in memory until the final termination of the program is carried out. It would be a terrible nuisance to see all time spent so far being lost. To remedy this ADF supports a check point file, named TAPE13, to help you recover at least some, if not most, of the results: not for analysis, but for continuation from a point not too long before the fatal condition occurred. TAPE13 can be used, just like TAPE21, as a normal restart file. See the restart key.

11.2.3 Memory Management

Problem: The program aborts with an error message "MEMORY ALLOCATION ERROR". This message is issued both in the logfile and in the output file.

Cause: Memory allocation may fail due to:

- Insufficient virtual (i.e. total RAM + swap) memory

- On Unix: too low values for per-process memory limits

- Restrictions of the 32-bit architecture

Cure: Problem 1: add more physical RAM or increase the size of the swap space (page file). Problem 2: add one or more ulimit commands to your run script setting relevant limits to "unlimited". Problem 3: Perform your calculations on a 64-bit system. ADF version for the most common 64-bit operating systems are available so use them!

All the three problems above can be avoided by reducing the size of the calculation. The most important parameter defining the amount of used memory is the size of the basis set or, more precisely, the total number of Cartesian Slater functions, naos. Current value can always be found in the out file of the calculation, just search for the "naos" string. The amount of memory used by a particular calculation depends on the naos value and of the type of the calculation and, for large naos, it scales as $naos^2$. For example, a non-relativistic calculation during SCF can use up to 40 $naos^2$ bytes of memory. Using spin-orbit coupling may double this amount and using a hybrid or a meta-GGA XC functional will add extra on top of it. Also TDDFT calculations require additional memory.

What can be done to reduce memory usage? First of all, reducing the basis set size for non-critical parts of the molecule will reduce the memory requirement without reducing the quality of the results. Secondly, performing a calculation with a pure GGA instead of B3LYP will not only reduce the amount of memory used but also make the calculation faster. The latter especially applies to geometry optimizations because there B3LYP does not perform any better than some of the GGAs.

Note: If workspace problems occur for relatively small calculations, there might be a bug. Notify your ADF contact: send us the output file so that we can have a look and check things out.

11.2.4 SCF troubleshooting

SCF convergence problems can have various reasons. Thus, finding the reasons for a particular SCF behavior is a half-way to solving the problem. You'll be surprised but the majority of SCF convergence problems are caused by a **non-physical calculation setup**, such as mistakes in the geometry or a too large negative charge. Thus, the first thing to do is to check if the geometry is really what it is meant to be. Check for too short inter-atomic distances, make sure the coordinates are specified in the right units. By default ADF expects atomic coordinates in Angstrom so check that the coordinates are provided in these units. Also check that no atoms got "lost" when importing coordinates.

So, your calculation is set up correctly, but the SCF still does not converge. Before discussing other options, let's look at how the SCF process in ADF is organized. In a nutshell, it consists of the following steps:

- A Fock matrix is constructed from the current density and the potential.

- The Fock matrix is used in a DIIS procedure where it is mixed with some previous Fock matrices to construct a new one. By default ADF uses a mixed ADIIS+DIIS scheme.

- This new Fock matrix is diagonalized to obtain molecular orbitals (MOs).

- MOs are populated by electrons following the *aufbau* principle or, if the KeepOrbitals feature is ON, by overlap with a previous density matrix.

- A new density matrix is constructed from occupied MOs and the steps 1-5 are repeated.

Problems on any of the steps 1, 2, and 4 above can cause problems in the whole SCF process. Usually one can identify which step causes the problems by looking in the logfile and in the output file. In the logfile, two values are printed for each SCF cycle: ErrMat and MaxEl. Both values are related to commutator of the current Fock and density matrices, [F,P]. ErrMax is a sum of squares of the commutator matrix elements while MaxEl is its largest (by absolute value) element. Below, different SCF patterns will be discussed with suggestions on how to solve them.

By far the most common reason for non-converging SCF is a very small or absent HOMO-LUMO gap. This problem is most frequently observed for compounds containing d- and f-elements (transition and rare-earth metals). This causes different MOs to be populated in subsequent cycles at the step 4 above, which, in turn, leads to large changes in the density and Fock matrices between cycles. In the logfile, the problem manifests itself by the ErrMat and MaxEl values remaining rather large (in the order of 0.1 to a few tens) sometimes going down to smaller values but then jumping back up. By looking at the MO population numbers in the output file one can sometimes see that the HOMO changes from cycle to cycle. There are a few ways to get the SCF converged in such a situation.

For open-shell electronic configurations, it is possible that a spin-unrestricted calculation will converge better than a spin-restricted one. Thus, if the molecule is not going to be used as a fragment (in which case it *must be* spin-restricted) then it is recommended to perform a spin-unrestricted calculation in a high-spin configuration. This is particularly useful for molecules with multiple radical centers, such bi- or multi-nuclear transition metal complexes. After a high-spin calculation of the complex has converged one can perform a broken-symmetry low-spin calculation using the high-spin results as a restart and a *SpinFlip* (page 50) feature. See also a tutorial on spin coupling of an iron compound for an example of this approach.

If a spin-unrestricted calculation is not desirable or if it also has SCF convergence problems one may consider trying different DIIS methods in the step 2 above or try a completely different SCF method, preferably in the order listed below:

- A-DIIS is used by default but it is incompatible with non-aufbau electronic configurations so it may be disabled in this case and SDIIS is used. A non-aufbau electronic configuration may be enforced using a block form of the Occupations key, but it may also result from the KeepOrbitals (a.k.a. orbital tracking) feature, or by the Fermi orbital population scheme. Thus, check if A-DIIS has been disabled (i.e. you do not see any A-DIIS coefficients in the SCF part of the text output) and if it was then you can try using one of the options below.

- The *LIST family of methods* (page 297) are developed by Y.A. Wang and coworkers. The convergence when using fDIIS, LISTb, LISTf, or LISTi is often comparable to that of A-DIIS, but sometimes it may be better or worse. Like A-DIIS, these methods also replace the DIIS step only leaving other steps unchanged. In the LIST

methods, the number of the expansion vectors (also known as the size of the DIIS space) is an important factor which may have to be adjusted to achieve convergence.

- The *MESA method* (page 297) is developed by the authors of the LIST family. This method combines up to five individual acceleration methods. From our experience, using this method in combination with the *Fermi orbital population* (page 45) one can converge transition metal clusters in the shortest time even though each SCF cycle is somewhat more expensive when using MESA than when using any of the acceleration methods mentioned above.

- *Energy-DIIS* (page 298) by Scuseria and Kudin is a good method but in general not better than the LIST family methods. It requires evaluation of the total energy, which is its strong point and a weakness at the same time. It is a strong point because it lets the algorithm converge the SCF to a configuration with the lowest energy and it is a weakness because energy evaluation is a computationally expensive procedure in ADF. To use Energy-DIIS just add an EDIIS keyword to the SCF input block. Energy-DIIS can only be used in combination with OldSCF.

- The *Augmented Roothaan-Hall (ARH)* (page 299) method is an alternative SCF method that combines steps 2-4 above into a single step. Essentially, ARH performs a direct minimization of the energy as a function of the density matrix combining a preconditioned conjugate-gradient method with a trust-radius approach. This is probably the most powerful SCF method to date because it can converge even the most difficult cases. However, it also has its limitations and drawbacks discussed in the corresponding section of the ADF User's Guide. The most important drawback is that, like Energy-DIIS, it also requires evaluation of the total energy. Besides, for the method to work reliably, the energy must be accurate, which means that an accurate fit and high integration accuracy should be used. ARH can only be used in combination with OldSCF.

As mentioned above, A-DIIS, LIST and Energy-DIIS affect only the DIIS step of the whole SCF process. Thus, it is possible that the SCF still has trouble converging even with the best DIIS method, because different MOs are occupied in different SCF cycles, which induces large changes in the density. ADF has a built-in feature called KeepOrbitals that assigns electrons to MOs based on their overlap with occupied MOs from the previous SCF cycle. KeepOrbitals is usually switched on after half of the allowed SCF cycles has passed (150 by default). However, if there are SCF problems, switching on KeepOrbitals may not have the desired effect. For example, the system may be trapped in an excited state due to it. Changing the SCF cycle at which KeepOrbitals kicks in may affect the final electronic configuration. Thus, playing with KeepOrbitals and trying different values for its parameter (the SCF cycle number) is encouraged.

For the DIIS-like methods listed above, the number of expansion vectors is an important parameter. We found that in some cases, for example, transition metal clusters, increasing this number to 20 or 25 solves most of the problems. Thus, the first thing to try when there are SCF convergence problems is to use the following input parameters:

```
SCF
  AccelerationMethod ADIIS
  DIIS N=20
END
Occupations KeepOrbitals=300
```

The input above uses the default A-DIIS but increases the number of expansion vectors to 20 and disables KeepOrbitals during the whole SCF process. You can also try replacing ADIIS with any method from the LIST family.

If none of the individual acceleration methods work one can try using MESA with the increased number of DIIS vectors:

```
SCF
  AccelerationMethod MESA
  DIIS N=20
END
```

Numerical noise in the exchange-correlation potential may also contribute to SCF convergence problems. If this is the case, the SCF process starts off converging quite well until some point after which the ErrMat and MaxEl values remain relatively small but do not decrease further. This behavior is typically observed for systems where weak (Van

der Waals or hydrogen bonding) interactions are present. It is caused by the density fit not being good enough in the chemically relevant region between weakly bonded fragments.

This problem can be resolved by adding the *EXACTDENSITY* (page 305) keyword to the input file. When using ADFinput, the corresponding option called "Density used in XC-potential" found on the Accuracy tab should be set to "Exact" or "Exact MO-based", "Exact" being preferred.

11.2.5 Geometry Optimization troubleshooting

No convergence

First of all one should look how the energy changed during the latest ten or so iterations. If the energy is changing more or less in one direction (increasing or decreasing), possibly with occasional jumps, then there is probably nothing wrong with the optimization. This behavior is typical in the cases when the starting geometry was far away from the minimum and the optimization has a long way to go. Just increase the allowed number of iterations, restart from the latest geometry and see if the optimization converges.

If the **energy oscillates** around some value and the energy gradient hardly changes then you may need to look at the calculation setup. We give some recommendations below.

The success of geometry optimization depends on the accuracy of the calculated forces. The default accuracy settings are sufficient in most cases. There are, however, cases when one has to increase the accuracy in order to get geometry optimization converged. First of all, this may be necessary if you tighten the optimization convergence criteria. In some cases it may be necessary to increase the accuracy also for the default criteria. Here's what you can do to increase the accuracy of gradients:

- Increase the numerical quality to "good"
- Add an ExactDensity keyword or select "Exact" in the "Density used in XC-potential" list in the Details:Accuracy panel. This will make the calculation 2 or 3 times slower.
- Tighten the SCF convergence criteria, for example, to 1e-8.

Example input with some of these stricter settings using a TZ2P basis set. ExactDensity is not included here since it will make the calculation much slower.

```
NumericalQuality Good
Basis
   Type TZ2P
   Core None
End
SCF
 converge 1e-8
End
```

Small HOMO-LUMO gap: check the HOMO-LUMO gap at the last SCF cycle at a recent geometry. Is it comparable with the changes in the MO energies between geometries? If yes, then it is possible that the electronic structure changes between optimization steps, which may lead to non-convergence. This may indicate a fundamental problem with the calculation setup. First of all, check that you obtain a ground state in a single-point calculation. Is the spin-polarization value correct? Try calculating some of the high-spin states if possible and see if they have lower energy. If the MO repopulation that is taking place is between MOs of different symmetry, then you can try freezing the number of electrons per symmetry using an OCCUPATIONS block.

In the next two sections we've collected some troubleshooting tips that apply only to the new or only to the old branch. In the last section, we also address a very rare issue with very short bonds that may appear independently of the optimization branch.

New Branch

Are there **constraints** imposed on the optimization? If yes, then check that the constraints do not break the symmetry. ADF automatically preserves symmetry when the starting structure is symmetric and no *Symmetry NOSYM* has been specified. This symmetry preservation may conflict with constraints if the constraints are not specified according to symmetry. For example, if two inter-atomic distances are symmetry-equivalent and you freeze one of them, then you should also freeze the other explicitly.

Optimization in Cartesian coordinates usually needs more steps to converge compared to delocalized. If you haven't tried delocalized optimization for the system then you should do it.

Near 180-degree angles with optimization in delocalized coordinates. ADF normally does not have a problem with a near-180-degree valence angle if the initial value of the angle was larger than 175 degrees or if it is a terminal bond angle. If the initial angle was larger than 175 degrees then it gets a special treatment. For example, any torsion angle that contains the three atoms is removed or replaced with a torsion angle involving only the two end atoms of the near-180-degrees angle. If the angle defines a terminal bond then two angles in two coordinate planes are used to define the bond instead of a valence and a torsion angle. However, if the initial value of the angle was far from 180 degrees and has become close to it during optimization then optimization may become unstable, especially if this is an angle connecting large fragments. In this case, it may be useful to restart geometry optimization from the latest geometry. As a last resort, you may want to constrain the angle to a value close, but not equal, to 180 degrees.

Old Branch

Spurious jumps

Problem: During geometry optimization in Z-Matrix coordinates, the atomic configuration makes a large unrealistic jump.

Possible cause 1: the triplet of atoms to which the current atom is related by the Z-matrix is (almost) co-linear. When, in a geometry step, the triplet passes through co-linearity, the dihedral angle for the current atom should make a discontinuous jump of 180 degrees. This is not checked in the program and the dihedral angle may not get corrected, resulting in a geometric jump of the atom (and hence of all atoms related to it by the Z-matrix).

Cure: Check the triplets of atoms, used in your Z-matrix to define the dihedral angles. If one of them is almost colinear, then redefine the Z-matrix or use *Cartesian* optimization.

Possible cause 2: the connectivity of the Z-matrix does not reflect some important bonds. Especially when the molecule contains (a large number of) rings, this badly affects the stability of the geometry update step. The reason is basically that computed *Cartesian* forces are transformed into changes of the curvilinear *internal* coordinates. The transformation between the two systems of coordinates is non-linear, but mathematically assumed to be linear. This is only a good approximation for small steps.

Cure: Redefine the Z-matrix and/or (if the geometry steps are very large) set a smaller upper bound on the maximum step (key GEOMETRY, subkey *step*).

Constraints are violated

Problem: constraints are violated: coordinates that were specified as frozen turn out to change during the optimization or coordinates that should remain the same start to differ after a few geometry update cycles.

Possible cause: there is an internal conflict between different demands, usually: symmetry versus constraints. The problem arises easily when a constrained optimization is requested for a molecule with some symmetry while the coordinates were defined with a Z-matrix structure that does not properly reflect the symmetry. Usually the deviations from the requested constraints are small. If they are really large, there might be a bug and you should contact ADF support.

Cure: redefine the Z-matrix and/or use Cartesian optimization (if the constraints are expressible in Cartesians).

Very short bonds

If the computed equilibrium geometry appears to exhibit unlikely values, typically significantly *too short* bond lengths, you may have run into a basis set problem, in particular (but not only) if the Pauli relativistic method is applied.

Problem: Optimized bond lengths are clearly too short. The energy may also look suspicious.

Possible cause 1: Basis set trouble: onset of Pauli variational collapse, if you have applied the Pauli relativistic option. Caused by small (or absent) frozen cores and/or relatively large basis sets, applied to heavy elements.

Possible cause 2: Basis set trouble also, but quite different from the previous potential cause: you have used relatively *large* frozen cores. When the atoms approach each other during the optimization and the frozen cores start to overlap, the energy computation and the computed energy gradients become more and more incorrect. This is a result of the inappropriateness of the frozen core approximation, which indeed assumes that frozen cores of neighboring atoms do not significantly overlap. Without going into a detailed explanation here, the net effect is that certain repulsive terms in the energy computation are missing and hence a spurious tendency to a 'core collapse' arises, yielding too short bond lengths.

Cure: Best is to abandon the Pauli method and use the ZORA approach instead for any relativistic calculation. If for whatever reason you insist on using the Pauli formalism, apply bigger frozen cores and, if that doesn't help, reduce the basis set (not by deleting polarization functions, but by reducing the flexibility of the occupied-atomic-orbitals space, in particular *s*- and *p*-functions). Note, however, that large frozen cores can be a cause for trouble by themselves, irrespective of any relativistic feature. If you have reason to believe that your frozen cores might be too *large*, given the resulting bond lengths in your calculation, you have to pick smaller cores (and hence be very wary of using the Pauli formalism for any relativity).

11.2.6 Frequencies

Imaginary Frequencies

Problem: totally unexpected significant imaginary frequencies are obtained (in a Frequencies run) where you are pretty convinced that all frequencies should be real.

Possible cause 1: problems with the electronic configuration. If there are competing configurations, the electronic *states* in the different displaced geometries may be different, resulting in energies and gradients belonging to different potential energy surfaces to be compared and combined into force constants (frequencies).

Check: orbital occupations and SCF convergence behavior: if the SCFs in the displaced geometries start with large errors and/or converge very slowly you are likely to have stumbled into different configurations, so that the results from the displaced geometries are incompatible.

Cure: This is a difficult situation that may require some experimenting and judicious manipulation of the various SCF options. The bottom line is that you should try anything you can to ensure that all involved geometries have the same electronic configuration. As long as you fail to achieve this, the results are meaningless.

Possible cause 2: flat potential energy surface (think about almost free rotation modes) coupled with relatively high noise level in gradients caused by numerical integration errors or not sufficiently converged geometry optimization.

Check: visualize the imaginary frequencies in ADFspectra and check that their respective normal modes correspond to movements that are expected to be have (nearly) flat energy profile.

Cure :

- restart geometry optimization with more strict convergence criteria. The default criterion on gradients 0.001 Hartree/Angstrom may be not strict enough for some systems. In such cases a value of 0.0001 is recommended, and for accuracy reasons use "good" numerical quality, and EXACTDENSITY (important for GGA's).

Example input with strict settings using analytical frequencies, and a TZ2P basis set. Unlike the numerical frequencies, the analytical frequencies can be computed immediately after a geometry optimization by including both block keywords in the same input file.

```
Geometry
  converge grad=1e-4
End
AnalyticalFreq
End
NumericalQuality Good
Basis
  Type TZ2P
  Core None
End
```

Geometry-displacement numbers in the logfile are not contiguous

Problem: successive displaced geometries in the logfile are numbered, but in your case these numbers make sudden jumps, like '0, 1, 2, 5, 6, 13...'

Cause: you're using Cartesian displacements in a system that has some symmetry in its equilibrium geometry. The program skips the displacements of symmetry-equivalent atomic coordinates to save time. The displacement counts in the logfile do not run over the actually performed displacements but over all atomic coordinates that could be displaced if no use were made of symmetry properties.

Cure: there is no error, don't worry.

11.2.7 Input ignored

Problem: the program doesn't get past input and aborts with a message eof while reading (....). Or the program seems to ignore some parts of input and as a consequence goes wrong somewhere. Or it seems that part of the input has not been read correctly or not at all.

Cause 1. You have used tab characters in your input file. These are not normally visible when you edit your file, but they will affect the program's scanning of the input. When you use tab characters in the input, it is very likely that the program will do something wrong somewhere. Tabs may be ignored by the program, so that items that you believed were separate (by a tab!) are in fact read as contiguous.

Check: the input file on tab characters.

Cause 2: misusage of one of the block-type keys or general keys.

A case that relatively often shows up is typing a title as first line of the input file, *without preceding it by the keyword title*. The program does not understand this as the title, but rather tries to interpret the first word as a keyword. This leads to an error if the first word is recognized as one of the pre-defined block-type keys (possibly abbreviated).

Check: the input file on usage of block-type keys and on proper usage of a title.

Cause 3: incorrect processing of expressions or unintended replacement of names by numerical values. Various kinds of mis-typing or incorrect usage of variables may cause this.

Check: how the program sees input, *after parsing*. This can be done by rerunning the job, with as first line in input : print parser.

This will cause the program to copy each input line *twice* to output, the second time after having parsed it. You may use StopAfter Input or StopAfter Init to let the program quit early so you can inspect what is going on with the input reading.

11.2.8 SFO Populations

In the section that prints the SFO populations of (selected) MOs you may occasionally find, for some SFOs in some MOs, *negative* SFO contributions. This may seem unphysical and hence suspicious, but it is 'only' a result of the Mulliken-type analysis method that underlies the computation of the SFO contributions. See the section below that discusses the output file. Likewise for larger-than-100% contributions: don't worry too much, these numbers may be correct (mathematically, given the Mulliken population formulas).

11.2.9 Error Aborts

The program performs a large number of checks during the calculation and may stop when it detects and error. It is close to impossible to show here a complete list of all possible error messages. In a large number of cases, additional information is printed in the output file to provide a clue as to the cause of the error. It is always useful to carefully inspect the printed info and to try to understand the meaning of any error- or warning messages. If you can't find your way out, try to get help from your ADF provider. If that fails, contact us directly at support@scm.com

11.2.10 Warnings

The program attempts to detect bugs, instabilities, convergence problems, et cetera and may issue warnings when something looks suspicious. This is not necessarily fatal to your results, but you should be cautious and try to understand what the messages are about. Most warnings are printed in the logfile. Usually there is corresponding and more extensive information in the standard output file.

ADF AS AN AMS ENGINE

With the 2018 release of the Amsterdam Modeling Suite, we have started to turn what used to be separate programs (e.g. DFTB and BAND) into libraries used by the new AMS driver program. In this more modular setup these libaries, which we refer to as engines, calculate energies and gradients, while the AMS driver steers the calculation over the potential energy surface, in e.g. a geometry optimization or molecular dynamics calculation. A good introduction to this new setup can be found in the beginning of the new AMS manual.

We also plan to convert ADF into an AMS engine, which means that ADF will cease to exist as a separate program and will instead be used through the AMS driver. In the 2019 release this has not yet happened and ADF is still the standalone program that it has always been. Nevertheless, the AMS driver already has a few features that the standalone ADF program is missing, e.g. a PES scan task, which is basically the multidimensional generalization of a linear transit calculation. In order to make this functionality also available for ADF, there is an **ADF engine** in the AMS driver, which simply wraps the standalone **ADF program** and thereby makes it usable through AMS. In this way the AMS driver can run the ADF program under the hood to calculate energy and gradients, as if it was a library.

It is relatively straightforward to run AMS with the ADF engine. Basically one just makes an `Engine ADF` block that contains the normal ADF keywords as they are documented in this manual. In order to perform a calculation with ADF running as an AMS engine, using a TZP basis and the PBE exchange-correlation functional, we would, for example, put the following engine block into the AMS input:

```
Engine ADF
   Basis
      Type TZP
   End
   XC
      GGA PBE
   End
EndEngine
```

One should keep the following points in mind when using ADF as an AMS engine in the 2019 release:

1. One should only use keywords that are relevant for an ADF single point calculation. It obviously does not make sense to have ADF perform a geometry optimization, when the AMS driver just wants to pick up the energy and gradients for a specific geometry. There are some checks for this in AMS, e.g. it will not let users put the `Geometry` block in the ADF engine input, but probably not all of these cases are checked automatically, so users should be extra careful and keep the ADF engine input to a minimum. The following keywords should definitely not be put into the ADF engine input:

   - `Geometry`

   - `AnalyticalFreq`

   - `Fragments`

   - `Symmetry`

   - `Title`

2. The overhead of running ADF through AMS is relatively large. Furthermore ADF is not restarted from step to step when running through AMS, i.e. it's SCF cycle will always start from scratch, even if the geometry barely from the last step, e.g. during the last steps a geometry optimization. It is also run with `Symmetry NOSYM`, which depending on the system geometry might result in another performance penalty.

3. The AMS driver has the be run in serial (e.g. with `NSCM=1`) when using ADF as an engine. This is necessary because not all MPI implementations support nested MPI runs, and it is generally more useful to have the AMS driver run serially and then run ADF in parallel using MPI. By default the spawned ADF process will use all available (physical) cores, but this behaviour can be changed with the `NSCM_AMSEXTERNAL` environment variable.

As of the 2019 release of AMS, we recommend using the ADF engine only for tasks which are not supported by the standalone ADF program, e.g. multidimensional PES scans.

THIRTEEN

APPENDICES

13.1 Basis set file format

In `$ADFHOME/atomicdata` you can find standard basis sets (including fit sets and frozen core orbitals) for all chemical elements of the periodic table at different levels of accuracy. The database is partitioned in subdirectories. Some of these are special: for example, the subdirectory Dirac contains input files for the program *dirac* (computation of relativistic potentials and charge densities). Most subdirectories contain files for the create runs: for example, the subdirectories SZ through TZ2P. *The section about the STO basis sets* (page 29) describes all subdirectories in more details.

The names of the basis set files consist of two parts: the standard symbol for the chemical element and the level of frozen core approximation; e.g. Mn.2p is a basis set file for Manganese with a frozen core up to and including the 2p shell.

Many all-electron basis can be found in the data base, especially for the elements H-Kr. All electron basis sets for the heavier elements can be found in the ZORA subdirectory. Fit functions for the all-electron basis sets must include more, in particular more contracted functions than the standard fit sets that are provided in the frozen core basis set files. If you would combine a basis set with an inadequate fit set the results are unreliable and absolutely inadequate, in the same fashion as when you would have used a highly inadequate basis set.

13.1.1 Sections

The basis set file supplied to ADF in Create mode contains the following sections:

```
Title
Basis Functions
Core Expansion Functions
Core Description
Fit Functions
```

Each of these items is discussed below. The basis set file does *not* define the applied density functional, the electronic configuration, precision parameters (numerical integration, SCF convergence criterion...), etc etc. These items can be set in the normal input file if the default is not satisfactory.

Title The title is the first record of the file. It may contain any text. Only the first 60 characters are actually used. This title is (by default) printed in the output; it is also used to stamp an identification on the result file (TAPE21). The file stamp will be printed whenever you use it as a fragment file in another calculation.

Basis functions A list of Slater type basis function characteristics. This part has the following format (example):

```
BASIS
  1s 5.4
  2s 1.24
```

```
  ...
  (etc.)
  ...
end
```

The words basis and end signal the beginning and the end of this section in the data file. The records in-between list the basis functions; each record contains the main quantum number, the angular quantum number, and the exponential decay factor for a *set* of Slater type basis functions.

The order of specification of the basis functions is not free. First must come the Core Functions used for core-orthogonalization. The CFs must be in order: s-functions first, then p-functions, then d-functions, and finally f-functions (as far as applicable). In the valence basis set there must be exactly one core-orthogonalization function for each frozen core shell (1s, 2s, 2p, ...).

Here as well as in all other function definitions below, the unit of length, implicit in the exponential decay factor, is bohr (atomic units), irrespective of the unit of length used in input for geometric items such as atomic positions (see units).

Core expansion functions This part has the form

```
CORE ns, np, nd, nf
  1s 7.68
  ...
  (etc.)
  ...
end
```

It looks very much like the *basis functions*: a list of Slater type function descriptions, closed by end. The header record however (core...) contains in addition four integers ns, np, nd, nf. They are the numbers respectively of s-, p-, d-, and f- frozen core shells in the atom. If you create for instance a Ruthenium atom with a frozen core up to the 4p shell, these numbers would be 4 3 1 0 : four frozen s-shells (1s,2s,3s,4s), three frozen p-shells (2p,3p,4p), one frozen d-shell (3d), and no frozen f-shells.

The core expansion sets defined in this section are used to describe the frozen core orbitals; they are not included in the valence basis set. In the list of core expansion sets all s-type functions must come first, then the p-type functions, then the d-functions, and then the f-functions (as far as applicable).

Core description Describes the frozen core shells as linear combinations of the core expansion functions. This section has the form

```
COREDESCRIPTION
  coefficients for the first frozen s-shell
  for the second s-shell
  for the n-th shell
  coefficients for the first frozen p-shell
  for the second p-shell
  for the d-shells
  for the f-shells
  pseudopotential parameters
end
```

For each of the angular momentum quantum numbers *l=s, p, d, f* all n *l* frozen shells are described by giving expansion coefficients. There are as many coefficients as there are function *sets* with the pertaining *l* value in the list of expansion functions. There are no separate coefficients for all *m* values: all *m* values are equivalent in a spherically symmetric model atom. See the Ca example below.

At the end of the (core) description section there is a record with pseudopotential parameters. The pseudopotential option, as an alternative to the frozen core approximation, is currently not supported, all values in this record must be zero, one for each frozen core shell. Equivalently you can put one zero, followed by a slash (/).

Fit functions is again a list of Slater type functions. These are used for an expansion of the density. The Coulomb potential due to the electronic charge distribution is computed from this expansion.

The format of this section is similar to the *basis functions*:

```
FIT
   1s 10.8
   ...
   (etc.)
   ...
end
```

The program cannot handle fit functions with *l*-value higher than 4, i.e. not higher than *g*-type functions. Bear this in mind if you construct alternative fit sets.

13.1.2 Example of a basis set file: Calcium

An example may serve to illustrate the format of a Create data file for DZ/Ca.2p (empty records inside and between the various sections are meaningless and ignored):

```
Calcium (DZ, 2p frozen)

BASIS
 1S   15.8
 2S    6.9
 2P    8.1

 3S    2.6
 3S    3.9
 3P    2.1
 3P    3.4
 4S    0.8
 4S    1.35
 4P    1.06

 3D    2.000
END

CORE     2  1  0  0
 1S   24.40
 1S   18.25
 2S    7.40
 2S    4.85
 3S    4.00
 3S    2.55
 4S    0.70
 4S    1.05
 4S    1.65
 2P   10.85
 2P    6.45
 3P    1.85
 3P    2.70
 3P    4.00
END

DESCRIPTION
  0.2076143E+00   0.7975138E+00  -0.7426673E-04   0.1302616E-03  -0.6095738E-04
```

```
   0.1508446E-04   0.1549420E-06  -0.2503155E-07  -0.1843317E-05
   0.8487466E-01  -0.4505954E+00   0.1009184E+01   0.9627952E-01  -0.3093986E-01
   0.1678301E-01  -0.2381843E-02   0.6270439E-02  -0.8899688E-02
   0.3454503E+00   0.6922138E+00  -0.1610756E-02   0.5640782E-02  -0.5674517E-02

0/
END

FIT
 1S   31.80
 2S   29.37
 3S   25.15
 4S   21.06
 4S   13.99
 5S   11.64
 5S    8.05
 6S    6.69
 6S    4.76
 6S    3.39
 7S    2.82
 7S    2.06
 7S    1.50
 2P   24.10
 3P   14.78
 4P    9.29
 5P    5.98
 6P    3.94
 6P    2.24
 7P    1.50
 3D   16.20
 4D   10.47
 5D    6.91
 6D    4.65
 6D    2.70
 7D    1.85
 4F    7.00
 5F    4.00
 5G    3.50
END
```

13.1.3 Extending a basis set

Polarization functions are provided for most elements. We strongly recommend to use one of the default basis sets described *here* (page 29).

If you are considering making your own basis set by including one or more polarization functions, a good rule of thumb to choose the function characteristics is the following: take the next higher *l*-value that does not yet occur in the function set (however, do not go beyond *f*-functions: the program cannot (yet) handle *g*-type basis functions), select the minimum value for the main quantum number n that is compatible with the *l*-value (i.e.: 2p, 3d, 4f), and determine the exponential decay factor ζ, such that the function attains its maximum value at somewhere between 1/3 and 1/2 times the bond length. The functional maximum for a Slater-type function is at $R = (n-1)/\zeta$. The maximum for r^2 times the square of a Slater-type function is at $R = n/\zeta$.

13.2 Elements of the Periodic Table

A few characteristics are predefined in ADF for all elements of the periodic table, as shown below.

The electronic configuration defines the default occupation numbers in Create mode. Basis sets for the elements Rf-Og (Z=104-118) are only available in `$ADFHOME/atomicdata/ZORA/`.

	Nuclear Charge Z	mass number of default isotope used for mass	electronic configuration
H	1	1	$1s^1$
He	2	4	$1s^2$
Li	3	7	$2s^1$
Be	4	9	$2s^2$
B	5	11	$2s^2\ 2p^1$
C	6	12	$2s^2\ 2p^2$
N	7	14	$2s^2\ 2p^3$
O	8	16	$2s^2\ 2p^4$
F	9	19	$2s^2\ 2p^5$
Ne	10	20	$2s^2\ 2p^6$
Na	11	23	$3s^1$
Mg	12	24	$3s^2$
Al	13	27	$3s^2\ 3p^1$
Si	14	28	$3s^2\ 3p^2$
P	15	31	$3s^2\ 3p^3$
S	16	32	$3s^2\ 3p^4$
Cl	17	35	$3s^2\ 3p^5$
Ar	18	40	$3s^2\ 3p^6$
K	19	39	$4s^1$
Ca	20	40	$4s^2$
Sc	21	45	$3d^1\ 4s^2$
Ti	22	48	$3d^2\ 4s^2$
V	23	51	$3d^3\ 4s^2$
Cr	24	52	$3d^5\ 4s^1$
Mn	25	55	$3d^5\ 4s^2$
Fe	26	56	$3d^6\ 4s^2$
Co	27	59	$3d^7\ 4s^2$
Ni	28	58	$3d^9\ 4s^1$, $3d^8\ 4s^2$
Cu	29	63	$3d^{10}\ 4s^1$
Zn	30	64	$3d^{10}\ 4s^2$
Ga	31	69	$3d^{10}\ 4s^2\ 4p^1$
Ge	32	74	$3d^{10}\ 4s^2\ 4p^2$
As	33	75	$3d^{10}\ 4s^2\ 4p^3$
Se	34	80	$3d^{10}\ 4s^2\ 4p^4$
Br	35	79	$3d^{10}\ 4s^2\ 4p^5$
Kr	36	84	$3d^{10}\ 4s^2\ 4p^6$
Rb	37	85	$5s^1$
Sr	38	88	$5s^2$
Y	39	89	$4d^1\ 5s^2$
Zr	40	90	$4d^2\ 5s^2$
Nb	41	93	$4d^4\ 5s^1$
Continued on next page			

Table 13.1 – continued from previous page

	Nuclear Charge Z	mass number of default isotope used for mass	electronic configuration
Mo	42	98	$4d^5 5s^1$
Tc	43	[98]	$4d^5 5s^2$
Ru	44	102	$4d^7 5s^1$
Rh	45	103	$4d^8 5s^1$
Pd	46	106	$4d^{10}$
Ag	47	107	$4d^{10} 5s^1$
Cd	48	114	$4d^{10} 5s^2$
In	49	115	$4d^{10} 5s^2 5p^1$
Sn	50	120	$4d^{10} 5s^2 5p^2$
Sb	51	121	$4d^{10} 5s^2 5p^3$
Te	52	130	$4d^{10} 5s^2 5p^4$
I	53	127	$4d^{10} 5s^2 5p^5$
Xe	54	132	$4d^{10} 5s^2 5p^6$
Cs	55	133	$6s^1$
Ba	56	138	$6s^2$
La	57	139	$5d^1 6s^2$
Ce	58	140	$4f^1 5d^1 6s^2$
Pr	59	141	$4f^3 6s^2$
Nd	60	142	$4f^4 6s^2$
Pm	61	[145]	$4f^5 6s^2$
Sm	62	152	$4f^6 6s^2$
Eu	63	153	$4f^7 6s^2$
Gd	64	158	$4f^7 5d^1 6s^2$
Tb	65	159	$4f^9 6s^2$
Dy	66	164	$4f^{10} 6s^2$
Ho	67	165	$4f^{11} 6s^2$
Er	68	166	$4f^{12} 6s^2$
Tm	69	169	$4f^{13} 6s^2$
Yb	70	174	$4f^{14} 6s^2$
Lu	71	175	$4f^{14} 5d^1 6s^2$
Hf	72	180	$4f^{14} 5d^2 6s^2$
Ta	73	181	$4f^{14} 5d^3 6s^2$
W	74	184	$4f^{14} 5d^4 6s^2$
Re	75	187	$4f^{14} 5d^5 6s^2$
Os	76	192	$4f^{14} 5d^6 6s^2$
Ir	77	193	$4f^{14} 5d^7 6s^2$
Pt	78	195	$4f^{14} 5d^9 6s^1$
Au	79	197	$4f^{14} 5d^{10} 6s^1$
Hg	80	202	$4f^{14} 5d^{10} 6s^2$
Tl	81	205	$4f^{14} 5d^{10} 6s^2 6p^1$
Pb	82	208	$4f^{14} 5d^{10} 6s^2 6p^2$
Bi	83	209	$4f^{14} 5d^{10} 6s^2 6p^3$
Po	84	[209]	$4f^{14} 5d^{10} 6s^2 6p^4$
At	85	[210]	$4f^{14} 5d^{10} 6s^2 6p^5$
Rn	86	[222]	$4f^{14} 5d^{10} 6s^2 6p^6$
Fr	87	[223]	$7s^1$
Ra	88	[226]	$7s^2$
Continued on next page			

Table 13.1 – continued from previous page

	Nuclear Charge Z	mass number of default isotope used for mass	electronic configuration
Ac	89	[227]	$6d^1\ 7s^2$
Th	90	232	$6d^2\ 7s^2$
Pa	91	231	$5f^2\ 6d^1\ 7s^2$
U	92	238	$5f^3\ 6d^1\ 7s^2$
Np	93	[237]	$5f^4\ 6d^1\ 7s^2$
Pu	94	[244]	$5f^6\ 7s^2$
Am	95	[243]	$5f^7\ 7s^2$
Cm	96	[247]	$5f^7\ 6d^1\ 7s^2$
Bk	97	[247]	$5f^9\ 7s^2$
Cf	98	[251]	$5f^{10}\ 7s^2$
Es	99	[252]	$5f^{11}\ 7s^2$
Fm	100	[257]	$5f^{12}\ 7s^2$
Md	101	[258]	$5f^{13}\ 7s^2$
No	102	[259]	$5f^{14}\ 7s^2$
Lr	103	[260]	$5f^{14}\ 6d^1\ 7s^2$
Rf	104	[261]	$5f^{14}\ 6d^2\ 7s^2$
Db	105	[262]	$5f^{14}\ 6d^3\ 7s^2$
Sg	106	[263]	$5f^{14}\ 6d^4\ 7s^2$
Bh	107	[264]	$5f^{14}\ 6d^5\ 7s^2$
Hs	108	[265]	$5f^{14}\ 6d^6\ 7s^2$
Mt	109	[268]	$5f^{14}\ 6d^7\ 7s^2$
Ds	110	[269]	$5f^{14}\ 6d^8\ 7s^2$
Rg	111	[272]	$5f^{14}\ 6d^9\ 7s^2$
Cn	112	[277]	$5f^{14}\ 6d^{10}\ 7s^2$
Nh	113	[280]	$5f^{14}\ 6d^{10}\ 7s^2\ 7p^1$
Fl	114	[280]	$5f^{14}\ 6d^{10}\ 7s^2\ 7p^2$
Mc	115	[280]	$5f^{14}\ 6d^{10}\ 7s^2\ 7p^3$
Lv	116	[280]	$5f^{14}\ 6d^{10}\ 7s^2\ 7p^4$
Ts	117	[280]	$5f^{14}\ 6d^{10}\ 7s^2\ 7p^5$
Og	118	[280]	$5f^{14}\ 6d^{10}\ 7s^2\ 7p^6$

Default (most abundant) isotope, used to set atomic mass (nr. of brackets gives mass directly). Default electronic configurations used in Create mode.

13.3 Multiplet States

Calculations with ADF yield results for one-determinant electronic states, which are not always the 'true' states of the molecule. The evaluation of the correct multiplet energies is not trivial in this approach, see further below the section on multiplet energies. The point is to evaluate a specific multiplet state as a linear combination of selected one-determinant functions, each computed in the field of the so-called Average-of-Configuration (AOC). Typically, in an open shell system, the AOC is the spin-restricted system in which all orbitals in the open shell are degenerate and equally occupied. The AOC serves then as a fragment for the subsequent calculations, in which the different open shell orbitals are occupied differently by specifying the appropriate occupation numbers as explained below.

Important: in these follow-up calculations it is imperative that the results are obtained in the AOC field: no SCF convergence must be carried out, because we only want to assign the electrons differently, while keeping exactly the

AOC orbitals. To achieve this, the follow-up calculations must use the keyword SCF, and the subkey iterations must be set to 0.

Since ADF requires that the point-group symmetry matches not only to the nuclear frame but also to the electronic charge density and MO occupations, these calculations must run in a lower point group symmetry. Often you will also want to run the modified calculations spin-*un*restricted. For an example, see the set of sample runs that come with the package and the discussion in the *Examples* (page 397).

The calculation of the one-determinant states based on the AOC reference state is controlled with the block SLATERDETERMINANTS,

```
SLATERDETERMINANTS file
End
```

were the argument must be a file (including the path). The file must be an ASCII file containing data in the same format as you would supply in the data block, see below. All information on the file until the *eof* must be suitable for the data block, but no record 'end' on the file must be specified: only the *contents* of the data block.

Alternatively, you can use the SLATERDETERMINANTS block as follows

```
SLATERDETERMINANTS
 title1
    irrep occups
    irrep occups
    ...
 subend
 title2
    irrep occups
    ...
 subend
 title3
    ...
 subend
 ...
end
```

Each 'title' functions as a subkey, but is otherwise an arbitrary string to label the resulting one-determinant calculation. Each such subkey block contains the occupation numbers for a single one-determinant calculation. It is necessary that the calculation uses the reference AOC run as its only fragment file. The occupations in the subkey blocks must be re-arrangements of the AOC open-shell electrons. In the Slaterdeterminants calculation you must explicitly specify the point group symmetry in which you want to run; this must be a lower symmetry than the AOC one, otherwise you couldn't rearrange the open shell electrons. See the sections below on multiplet energies. An sample run is included in *Examples* (page 397).

Each 'irrep occups' record specifies the occupations for the indicated irrep in the usual way (see for instance the occupations key). The irrep labels must correspond to the (lower) point group symmetry used in the slaterdeterminants calculation. Note that in an unrestricted calculations, occupations numbers must be given for both spins, using the double slash (//) to separate the occupations for spin-α and spin-β.

In this setup, the program will for each of the subkey blocks under the slaterdeterminants key execute an SCF calculation with only one cycle, i.e. no convergence, where the start-up field is the fragment field, i.e. the AOC field. So all one-determinant states in this calculation are evaluated in the AOC field. The resulting energies for the distinctly computed one-determinant states can then be combined to the desired multiplet values, corresponding to how the multiplet states are combinations of the one-determinant states.

13.3.1 Multiplet energies

The energies of atomic and molecular multiplet states that correspond to a given electron configuration can be calculated approximately with the method suggested in ref. [*323* (page 926)]. There it is indicated that it would not be justified to take an arbitrary configuration-state function (CSF), defined in general as a linear combination of determinants that has specific spin and space symmetry properties, and use the corresponding alpha and beta spin densities in a DFT energy expression. The same holds true for the densities corresponding to the average-of-configuration (see section 'DFT energy of a one-determinantal wave function', the 'procedure' notes).

Therefore, we restrict ourselves to just computing the DFT energies of single-determinant wavefunctions. Usually (but not always) this is sufficient information to obtain the multiplet energies. The procedure, which is explained in [*323* (page 926)], requires knowledge of the CSFs belonging to a given configuration. This means that a multiplet state with specific L, ML and S, MS values has to be written as a linear combination of the determinant wavefunctions that belong to the given configuration. With the auxiliary program ASF (Adf Single-determinants Fribourg, developed by Claude Daul in Fribourg, Switzerland) all the CSFs can be obtained, printed as linear combinations of the determinants [*324* (page 926)]. The inverse transformation yields the determinants written as linear combinations of the CSFs.

It is often advantageous to search for CSFs that consist of one determinant only, since the energy of this determinant should correspond directly to the multiplet energy. Sometimes there is redundancy in the information and there may even be some inconsistency: two determinants may exist that both are CSFs belonging to the same multiplet state but yield somewhat different energies. We will illustrate this for the Carbon atom example treated below.

The discrepancies are a measure of 'error bars' associated with the theoretical multiplet energies. As a matter of fact, there are certain symmetry relations between the energies of the determinants of a configuration, calculated as the expectation value of the determinant for the full many-electron Hamiltonian. An example is the equal energy for the determinants of a p^1 configuration, whether the electron is placed in the p_0 ($= p_z$) orbital or in the $p_{+1}(= (p_x + ip_y)/\sqrt{2})$ orbital. This equality is not obtained with present-day density functionals, leaving an ambiguity ('error bar') in the determination of the energy. A more complete treatment of the symmetry relations between determinant energies is given in [*324* (page 926)].

The auxiliary program ASF, that for finite point groups finds the CSFs as linear combinations of determinants, performs also a symmetry analysis of all the two-electron integrals for a configuration, reducing them to a minimum number of non-redundant ones. ASF expresses the energies of the multiplets in the non-redundant two-electron integrals.

Warning: However, there have occasionally been found inconsistent results. A comparison to the results obtained by the procedure outlined in [*323* (page 926)] may show significant differences and the latter seem more accurate and consistent.

DFT energy of a one-determinantal wave function

The determinant corresponds to a well defined ρ^α and ρ^β. Suppose we are dealing with a d^2 configuration and we wish to know the energy of

$$D_1 = |d_2^\alpha(1)d_1^\alpha(2)|$$

where d_m has the Y_{2m} angular part. This determinant is a CSF of the ^3F multiplet

$$D_1 = |^3F; M_L = 3; M_S = 1\rangle$$

We can easily transform to the real spherical harmonics that are used in ADF:

$$Z_{lm}^C = \frac{1}{\sqrt{2}}(Y_l^{-m} + Y_l^{-m*}) = \frac{1}{\sqrt{2}}(Y_l^{-m} + (-1^m)Y_l^m)$$

$$Z_{lm}^S = \frac{i}{\sqrt{2}}(Y_l^{-m} + Y_l^{-m*}) = \frac{i}{\sqrt{2}}(Y_l^{-m} + (-1^m)Y_l^m)$$

with back transformations:

$$Y_l^m = \frac{1}{\sqrt{2}}(-1)^m(Z_{lm}^C + iZ_{lm}^S)$$

$$Y_l^{-m} = \frac{1}{\sqrt{2}}(Z_{lm}^C - iZ_{lm}^S)$$

Here the superscripts c and s stand for the cosine, respectively sine type of combinations of $exp(-im\phi)$ and $exp(im\phi)$. This yields explicitly:

$$d_{z^2} = d_0$$

$$d_{xz} = 1/\sqrt{2}(d_{-1} - d_1)$$

$$d_{yz} = i/\sqrt{2}(d_{-1} + d_1)$$

$$d_{x^2-y^2} = 1/\sqrt{2}(d_{-2} + d_2)$$

$$d_{xy} = i/\sqrt{2}(d_{-2} - d_2)$$

$$d_0 = d_{z^2}$$

$$d_1 = -1/\sqrt{2}(d_{xz} + id_{yz})$$

$$d_{-1} = 1/\sqrt{2}(d_{xz} - id_{yz})$$

$$d_2 = 1/\sqrt{2}(d_{x^2-y^2} + id_{xy})$$

$$d_{-2} = 1/\sqrt{2}(d_{x^2-y^2} - id_{xy})$$

For D_1 we obtain:

$$\rho_\alpha = |d_2|^2 + |d_1|^2 = 1/2|d_{x^2-y^2}|^2 + 1/2|d_{xy}|^2 + 1/2|d_{xz}|^2 + 1/2|d_{yz}|^2$$

$$\rho_\beta = 0$$

The fractional occupations have to be used in order to generate the densities $\rho\alpha$ and $\rho\beta$ and the corresponding density matrices ρ_α and ρ_α. The density matrices can be used to calculate the energy of D1 (and ^3F) with respect to the energy of the 'master fragment', which should be the restricted atom with d^2 occupation. Other determinants of this configuration can be treated similarly to obtain more multiplet energies of the d^2 configuration.

Below is an example of how you can obtain determinant energies 'by hand', i.e. by carrying out a specific sequence of ADF calculations. ADF supports an automatic procedure to do this, using the input key SLATERDETERMINANTS, see the ADF User's Guide, the *Examples* (page 397), and below.

Procedure

1. Determine a set of orbitals belonging to the given configuration. These orbitals are generated in what we call the average-of-configuration (AOC) calculation. This is a spin-restricted SCF calculation where the electrons of the configuration are distributed equally over the subspecies of the open shell irreps in order to retain the A1 symmetry of the total density in the symmetry group of the molecule. For instance, in case of an atomic d^2 configuration, the AOC calculation can be done in symmetry atom with occupation 2 in the d irrep. In case of an t_{2g} ^{5} e_g ^{1} configuration of an octahedral complex, the AOC calculation requires an occupation of 5 electrons in the t_{2g} , and 1 electron in the e_g . The result file TAPE21 of the AOC calculation has to be saved, to be used as a fragment file in the subsequent calculations.

2. The AOC is used as a fragment in all subsequent calculations that are performed to obtain single determinant energies. This means that those single determinant energies are always with respect to the AOC energy. This is a case where there is only one "fragment", which is actually the complete system, but in a different electronic configuration and in possibly a different symmetry group.

Suppose that a single determinant corresponds to spin-up and spin-down densities ρ^α and ρ^β , i.e. to specific spin-unrestricted occupations of the AOC orbitals. These densities ρ^α and ρ^β correspond to a symmetry group that will in

general be a subgroup of the symmetry group of the molecule. For instance, the occupation $(p_{+1}\ \alpha)^1$ in the case of an atomic p^1 configuration corresponds to

$$\rho^\alpha = 1/2p_x^2 + 1/2p_y^2$$

with $D_{\infty h}$ symmetry.

To obtain the energy of the determinant wave function we must now perform *one* cycle (iterations= 0 in the block key SCF) of a spin-unrestricted calculation, with AOC as (the only) fragment with alpha and beta occupation numbers (using the input key occupations) such that ρ^α and ρ^β result. Note that the appropriate (lower) symmetry point group must be specified in the input file.

Occasionally, the single determinant corresponds to a closed shell configuration in the appropriate lower symmetry, for instance determinant $D_{10} = |0^+\ 0^-\ |$ of the p^2 configuration of Carbon, with density r=p_z ^{2} . In that case the one-cycle calculation can of course be spin-restricted.

N.B.1. One cycle will regenerate the SCF orbitals of AOC, if the same field is used as the converged AOC field. This will actually be the case because the starting potential is taken from the fragment TAPE21 file. The key modifystart-potential must not be used (the density should be distributed equally over the spins).

N.B.2. After diagonalization in the one-cycle run, the AOC orbitals have been obtained again and are occupied as specified. The ('bonding') energy is calculated from the resulting charge density.

Remarks:

- If one does not perform just one cycle, but instead converges the unrestricted calculation, the energy will be lowered by 'polarization' of the orbitals. It is theoretically not so clear what the status of such converged energies is. Usually the energy lowering is in the order of 0.1 eV, but it may be quite a bit larger.

- It is not necessary to use AOC as fragment in the single-determinant runs. It is also perfectly allowed to run all calculations (ground state, AOC, determinants) from one set of fragments, for instance the standard atomic fragments. Since we must arrange that the one-cycle determinant calculations use the AOC field, so as to reproduce the AOC orbitals, we must then supply the result file TAPE21 of the AOC run as a restart file, using the key restart; see the adf User's Guide. Of course, in such an approach the computed energies are with respect to another reference, for instance the restricted atoms.

Results for first period atoms

In one of the next sections tables are given for the energy lowering in going from the converged spherically symmetric spin-restricted atom (the 'master' fragment) to specific one-determinant wavefunctions with the orbital occupations as specified. Note that the p_x and p_y populations are always equal; only their sum is given. In many cases the determinant corresponds to a specific state, which is then given in the last column. For each atom, the first calculation is for the spherically symmetric spin-*un*restricted atom. These tables are now obsolete, all information needed to obtain the atomic reference energies, i.e. the ground state multiplet energy with respect to the AOC, can be found in ref. [325 (page 926)].

Examples worked out for all first period atoms:

H: Configuration $(1s)^1$. Only one determinant: $|\ 1s\ \alpha\ (1)\ |$

He: Configuration $(1s)^2$. Closed shell.

Li: Configuration $(2s)^1$. Only one determinant: $|\ 2s\ \alpha\ (1)\ |$

Be: Configuration $(2s)^2$. Closed shell.

B: Configuration $(2p)^1$. Ground multiplet ^{2} P.

$$D_1 = |p_1\alpha(1)| = |^2P; M_L = 1; M_S = 1/2\rangle$$
$$\rho^\alpha = 1/2(p_x - ip_y)(p_x + ip_y) = 1/2(p_x^2 + p_y^2)$$

The occupation numbers for D_1 are

$$p_x^\alpha = p_y^\alpha = 1/2$$
$$p_z^\alpha = p_x^\beta = p_y^\beta = p_z^\beta = 0$$

Another determinant belonging to ^{2}P is $D_2 = |..p_0\ \alpha\ (1)|$ with occupations $p_z^\alpha = 1$ and all other p-occupations zero. This determinant is 0.04 eV lower in energy than D_1 for LDA, but 0.15 eV for BP.

C: Configuration p^2 . Multiplet states are ^{3}P, ^{1}S and ^{1}D. We use this atom as an example of the general procedure. First write down all determinants belonging to p^2 and group them according to M_S and M_L ($1^+ \equiv p_1\ \alpha$, ...)

Determinant	M_S	M_L
$D_1 = |1^+\ 1^-\ |$	0	2
$D_2 = |1^+\ 0^+\ |$	1	1
$D_3 = |1^+\ 0^-\ |$	0	1
$D_4 = |1^+\ -1^+\ |$	1	0
$D_5 = |1^+\ -1^-\ |$	0	0
$D_6 = |1^-\ 0^+\ |$	0	1
$D_7 = |1^-\ 0^-\ |$	-1	1
$D_8 = |1^-\ -1^+\ |$	0	0
$D_9 = |1^-\ -1^-\ |$	-1	0
$D_{10} = |0^+\ 0^-\ |$	0	0
$D_{11} = |0^+\ -1^+\ |$	1	-1
$D_{12} = |0^+\ -1^-\ |$	0	-1
$D_{13} = |0^-\ -1^+\ |$	0	-1
$D_{14} = |0^-\ -1^-\ |$	-1	-1
$D_{15} = |-1^+\ -1^-\ |$	0	-2
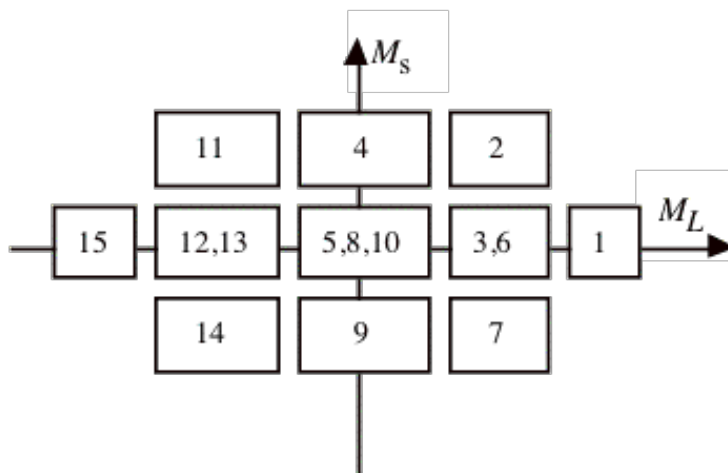
The presence of a determinant with $M_L = 2$, $M_S = 0$ and no $M_L = 2$, M_S 0 determinant indicates the presence of a ^{1}D multiplet, and E(^{1}D) = E(D_1). There is also a ^{3}P, the determinant with $M_S = 1$, $M_L = 1$. We should have E(^{3}P) = E(D_2) = E(D_4). The two determinants D_3 and D_6 in the $M_S = 0$, $M_L = 1$ box will mix, and the solutions of the 2 by 2 secular problem will be E(^{1}D) and E(^{3}P). Since the sum of the eigenvalues is equal to the sum of the initial diagonal elements, we have E(^{1}D) + E(^{3}P) = E(D_3) + E(D_6). We should also have E(D_3) + E(D_6) = E(D_1) + E(D_2). Such a relation provides a test on the consistency of the results. Finally we have the $M_S = 0$, $M_L = 0$ block. The sum of the energies of D_5 , D_8 and D_{10} should be the sum of the energies of ^{1}S, ^{3}P and ^{1}D. Since E(^{3}P) and E(^{1}D) are already known, E(^{1}S) can be calculated.

In the appendix we first locate for C the spherical unrestricted atom. Next we have E(D_4), yielding E(^{3}P) = -1.345 eV (LDA + Becke). Next E(D_2) = E(^{3}P) = -1.189 (always LDA + Becke). The difference is substantial:

~ 0.15 Next we have E(D$_3$) = - 0.812. Since E(D$_6$) = E(D$_3$), because ρ^α (D$_6$) = ρ^β (D$_3$) and ρ^β (D$_6$) = ρ^α (D$_3$), we should have 2E(D$_3$) = -1.624 = E(^{1} D) + E(^{3} P). Therefore E(^{1} D) = -1.624 - (-1.345) = - 0.279 or E(^{1} D) = - 1.624 - (-1.189) = - 0.435. These numbers can be checked against E(D$_1$) which also should be E(^{1} D): E(D$_1$) =+ 0.044. The discrepancy is large! Finally, ^{1} S can be obtained: E(D$_{10}$) = + 0.319 (D$_8$) = E(D$_1$) = + 0.044 E(D$_5$) = E(D$_1$) = + 0.044. So 0.407 = E(^{1} S) + E(^{3} P) +E(^{1} D). Different results for E(^{1} S) are obtained depending on the E(^{3} P) and E(^{1} D) we choose: e.g. E(^{1} S) = 0.407 -(-1.345) - (- 0.279) = 2.031 or E(^{1} S) = 0.407 - (-1.189) - (0.044) = 1.552. Comparing to experiment we might calculate the excitation energies w.r.t. E(^{3} P):

	calculated	experimental	HF
^{3} P $\rightarrow$ ^{1} D:	1.066 to 1.389	1.26	1.55
^{3} P $\rightarrow$ ^{1} S:	2.741 to 3.376	2.684	3.78

N: Configuration p^3 . Ground multiplet ^{4} S, corresponds to the spherical unrestricted atom, energy -2.943 eV. Other multiplets: ^{2} P, ^{2} D. According to the printed output for configuration p^3 we have |^2 D;M$_L$ =2;M$_S$ =1/2 = |p$_1$ $^\alpha$ p$_1$ $^\beta$ p$_0$ $^\alpha$ | = D$_2$ ρ^α = 1/2 p$_x$ ^{2} + 1/2 p$_y$ ^{2} + p$_z$ ^{2} ρ^β = 1/2 p$_x$ ^{2} + 1/2 p$_y$ ^{2} E(D$_2$) = - 0.745 according to the table in the Appendix (LDA + Becke). The energy of D$_1$, with ρ^α = p$_x$ ^{2} + p$_y$ ^{2} , ρ^β = p$_z$ ^{2} , is E(|1A 2B 3A|) = -1.9702. The energy of D$_3$, with ρ^α = 1/2 p$_x$ ^{2} + 1/2 p$_y$ ^{2} + p$_z$ ^{2} , ρ^β = p$_z$ ^{2} corresponding to |1A 2A 2B| or |2A 2B 3A|, is E(D$_3$) = - 0.158. Finally, D$_4$ has ρ^α = p$_x$ ^{2} + p$_y$ ^{2} , ρ^β = 1/2 p$_x$ ^{2} + 1/2 p$_y$ ^{2} , corresponding to |1A 1B 3A| and |1A 3A 3B|, and E(D$_4$) = - 0.109. The M$_L$ = 1, M$_S$ =1/2 determinants are |1A 1B 3A| and |1A 2A 2B|. Therefore E(^{2} D) + E(^{2} P) = E(D$_4$) + E(D$_3$), so E(^{2} P) = - 0.109 - 0.158 - (- 0.745) = + 0.478. We can use D$_1$ in the M$_L$ =0, M$_S$ =1/2 block, from which we find E(^{4} S) + E(^{2} D) + E(^{2} P) = 2E(D$_2$) =+ E(D$_1$). Hence E(^{2} P) = -1.490 - 1.9702 - (- 0.745) - (- 2.943) = + 0.2278.

O: Configuration p^4 . Multiplet states ^{3} P, ^{1} S, ^{1} D. D$_1$, with ρ^α = p$_x$ ^{2} + p$_y$ ^{2} + p$_z$ ^{2} , ρ^β = p$_z$ ^{2} corresponds to |1A 2A 2B 3A| , the M$_L$ =1, M$_S$ =1 determinant of ^{3} P: E(^{3} P) = -1.836 D$_2$ with ρ^α = p$_x$ ^{2} + p$_y$ ^{2} + p$_z$ ^{2} , ρ^β = 1/2 p$_x$ ^{2} + 1/2 p$_y$ ^{2} , corresponds to |1A 1B 2A 3A|, the determinant of ^{3} P: E(^{3} P) = -1.568 D$_3$, with ρ^α = 1/2 p$_x$ ^{2} + 1/2 p$_y$ ^{2} + p$_z$ ^{2} , ρ^β = p$_x$ ^{2} + p$_y$ ^{2} , corresponds to |1A 1B 2A 3B|, and M$_L$ =1, M$_S$ =0 belonging to ^{3} P as well as ^{1} D.

F: Configuration p^5 . Ground multiplet ^{2} P. As in B, we have two determinants with different energies belonging to this state: D$_1$ = |...(p$_0$ $^\alpha$)^1 (p$_0$ $^\beta$)^0 | $\rightarrow$ E(D$_1$) = - 0.715. D$_2$ = |(p$_{-1}$)^2 (p$_0$)^2 (p$_1$ $^\alpha$)^1 (p$_1$ $^\beta$)^0 | $\rightarrow$ E(D$_2$) = - 0.467.

Ne: Configuration p^6 . Closed shell.

Ground and Excited State Multiplets

The computation of multiplets can be carried out with adf, using the input block `SLATERDETERMINANTS`.

The method described in [*323* (page 926)] for the calculation of the energies of proper spin and spatial symmetry adapted Configuration State Functions is based on the calculation of the energies of single determinantal wavefunctions. The densities corresponding to those Slater determinants are inserted in the approximation used for the exchange-correlation energy.

The procedure as detailed above is somewhat involved. Moreover, one would like to have an easy procedure to calculate many determinants. This is particularly desirable if one wishes to calculate the energies of all Slater determinants of a given configuration, for instance if one wishes to calculate certain averages in view of the inconsistencies of the method.

We have implemented a semi-automatic procedure, using the block `SLATERDETERMINANTS`.

The general idea of this method is to first perform a restricted calculation in the symmetry that is appropriate for the molecule. This is called the average-of-configuration (AOC) calculation. This AOC calculation generates the orbitals which will be used in all the Slater determinants.

The AOC is the fragment that *must* be used in all subsequent calculations. The subsequent calculations are characterized by having the AOC as the (only) fragment, and by specifying the keyword `SLATERDETERMINANTS`. If an argument is given this must be a file name. The named file should contain the occupations for the determinants (see below). If no file name is specified, the occupations should be specified in the data block.

The required information, on file or in the data block, is the specification of the determinant or determinants that are to be calculated in the form of orbital occupation numbers for the AOC orbitals, using the irrep labels of the point group of the AOC calculation, see below for format. All specified determinants will be calculated, and the obtained energy will always be the energy difference with respect to the AOC. Default occupations for all subspecies of the AOC fragment are the occupations of the AOC itself. Therefore, only the open (modified) subspecies have to be specified.

One has to be careful with respect to the point group symmetry to use in the SLATERDETERMINANTS calculation. The density belonging to a specific determinant is usually lower than the AOC symmetry (which is the full symmetry group of the system). In that case this lower point group symmetry has to be specified in the SLATERDETERMINANTS calculation. Everything will always work fine if one just does not use any symmetry at all (nosym). However, if for reasons of computational efficiency one does want to use a subgroup of the system that corresponds to the determinant density or densities, this is perfectly possible. However, all the densities of the determinants specified *must* then have this (or a higher) symmetry. The program does not check on this, it is the user's responsibility to make sure that this condition is satisfied for all the determinants. The only check that is performed is that occupations of equivalent representations (subspecies of one irrep) in the lower point group of the SLATERDETERMINANTS run, that result from the specified occupations of the subspecies of the AOC symmetry, are equal.

Format of the input.

Important: in the SlaterDeterminants calculations it is imperative that the results are obtained in the AOC field: no SCF convergence must be carried out, because we only want to assign the electrons differently, while keeping the AOC orbitals exactly as they are. To achieve this, the calculations should use the keyword SCF, and the subkey iterations has to be set to 0 in the SCF data block.

Since adf requires that the point-group symmetry conforms not only to the nuclear frame but also to the electronic charge density and mo occupations, these calculations must run in a lower point group symmetry. Often you will also want to run the modified calculations spin-unrestricted.

For an example, see the set of sample runs that come with the package and the comments in the Examples document.

See: *SLATERDETERMINANTS block* (page 350). Each 'title' functions as a subkey, but is otherwise an arbitrary string to label the resulting one-determinant calculation. Each such subkey block contains the occupation numbers for a single one-determinant calculation. It is necessary that the calculation uses the reference AOC run as its only fragment file. The occupations in the subkey blocks must be re-arrangements of the AOC open-shell electrons. In the SLATERDETERMINANTS calculation you must explicitly specify the point group symmetry in which you want to run. The ρ^α and ρ^β densities of all determinants to be calculated must have this point group symmetry, or a higher symmetry.

Each 'irrep occups' record specifies the occupations for the indicated irrep in the usual way (see for instance the occupations key). The irrep labels must correspond to the AOC point group symmetry used in the AOC calculation, *not the point group symmetry used in the ''SLATERDETERMINANTS'' calculation!*. Note that in an unrestricted calculations, occupations numbers must be given for both spins, using the double slash (//) to separate the occupations for spin-alpha and spin-beta.

In this setup, the program will for each of the subkey blocks under the SLATERDETERMINANTS key execute an SCF calculation with only one cycle, i.e. no convergence, where the start-up field is the fragment field, i.e. the AOC field. So all one-determinant states in this calculation are evaluated in the AOC field. The resulting energies for the distinctly computed one-determinant states can then be combined to the desired multiplet values.

Example: Carbon p^2

```
SlaterDeterminants
C(p2) ALFA: s=1, px=py=2/3, pz=2/3; BETA: s=1, p=0 ! title
   S 1 // 1 ! irrep name and occupations
   P:x 0.666666666666666666 // 0 ! another irrep, et cetera
   P:y 0.666666666666666666 // 0
   P:z 0.666666666666666666 // 0
   D:z2 0 // 0
   D:x2-y2 0 // 0
```

```
   D:xy 0 // 0
   D:xz 0 // 0
   D:yz 0 // 0
SUBEND
C(p2) ALFA: S=1, px=py=1, pz=0; BETA: s=1 !next (Sl.Determinant) title
   S 1 // 1
   P:x 1 // 0
   P:y 1 // 0
   P:z 0 // 0
   D:z2 0 // 0
   D:x2-y2 0 // 0
   D:xy 0 // 0
   D:xz 0 // 0
   D:yz 0 // 0
SUBEND
C(p2) ALFA: s=1, px=py=0.5, pz=1; BETA: s=1
   S 1 // 1
   P:x 0.5 // 0
   P:y 0.5 // 0
   P:z 1 // 0
   D:z2 0 // 0
   D:x2-y2 0 // 0
   D:xy 0 // 0
   D:xz 0 // 0
   D:yz 0 // 0
SUBEND
C(p2) ALFA: s=1, px=py=0.5, pz=0; BETA: s=1, px=py=0, pz=1
   S 1 // 1
   P:x 0.5 // 0
   P:y 0.5 // 0
   P:z 0 // 1
   D:z2 0 // 0
   D:x2-y2 0 // 0
   D:xy 0 // 0
   D:xz 0 // 0
   D:yz 0 // 0
SUBEND
C(p2) ALFA: s=1, px=py=0.5, pz=0; BETA: s=1, px=py=0.5, pz=0
   S 1 // 1
   P:x 0.5 // 0.5
   P:y 0.5 // 0.5
   P:z 0 // 0
   D:z2 0 // 0
   D:x2-y2 0 // 0
   D:xy 0 // 0
   D:xz 0 // 0
   D:yz 0 // 0
SUBEND
C(p2) ALFA: s= 1, px=py=0, pz=1; BETA: s=1, px=py=0, pz=1
   S 1 // 1
   P:x 0 // 0
   P:y 0 // 0
   P:z 1 // 1
   D:z2 0 // 0
   D:x2-y2 0 // 0
   D:xy 0 // 0
   D:xz 0 // 0
   D:yz 0 // 0
```

```
SUBEND
```

In the example the AOC calculation is the Carbon atom in spherical symmetry (symmetry name atom).

Several spin states can be generated from this AOC set of orbitals, but they all have a lower symmetry than the AOC. In the example the point group $D_{\infty h}$ (DLIN) could be used in the SLATERDETERMINANTS calculation. In $D_{\infty h}$ the p orbitals split into two sets, p^*_x * and p^*_y * occur in π_x and π_y respectively, so their occupations must be identical, and p^*_z * is a Σ_u orbital.

In the block SLATERDETERMINANTS (or in the file) we now specify the occupations for the subspecies of the atom irreps of a specific Slater determinant and the program will sort out the corresponding occupations in the d(lin) symmetry.

In all cases the orbitals used for the energy calculation(s) will be the self-consistent AOC orbitals.

In the given example, the first set of occupations does not correspond to a Slater determinant, but is the spin-polarized spherical case with the p electrons evenly distributed over all components.

LDA results, with and without GGA (Becke-Perdew)

Energy changes (eV) for atoms going from restricted to (one-cycle) unrestricted. Results between parentheses are for *converged* unrestricted calculations)

All calculations have been performed in $D_{\infty h}$ symmetry, since p_x and p_y always had equal occupations and therefore could occur as π_u -x and π_u -y partners of the Π_u irrep.

El.	Occupations						LDA	LDA+Becke	BP
	alpha-spin			beta-spin					
	s	px+py	pz	s	px+py	pz			
H	1	0	0	0	0	0	-0.868 (-0.898)	-0.758 (-0.837)	-0.889 (-0.948)
Li	1	0	0	0	0	0	-0.231 (-0.235)	-0.195 (-0.207)	-0.249 (-0.256)
Be	1	0	0	1	0	0	0.000 (0.000)	0.000 (0.000)	0.000 (0.000)
B	1	2/3	1/3	1	0	0	-0.247 (-0.255)	-0.231 (-0.242)	-0.276 (-0.281)
	1	0	1	1	0	0	-0.295 (-0.321)	-0.436 (-0.474)	-0.448 (-0.485)
	1	1	0	1	0	0	-0.266 (-0.279)	-0.296 (-0.316)	-0.333 (-0.348)
C	1	4/3	2/3	1	0	0	-1.163 (-1.203)	-1.109 (-1.158)	-1.252 (-1.285)
	1	2	0	1	0	0	-1.152 (-1.211)	-1.271 (-1.345)	-1.372 (-1.436)
	1	1	1	1	0	0	-1.152 (-1.197)	-1.134 (-1.189)	-1.267 (-1.307)
	1	1	0	1	0	1	-0.462 (-0.506)	-0.726 (-0.812)	-0.778 (-0.868)
	1	1	0	1	1	0	0.159 (0.150)	0.039 (0.044)	0.087 (0.086)
	1	0	1	1	0	1	0.730 (0.668)	0.322 (0.319)	0.480 (0.450)
N	1	2	1	1	0	0	-2.936 (-3.032)	-2.827 (-2.943)	-3.101 (-3.190)
	1	2	0	1	0	1	-1.362 (-1.454)	-1.811 (-1.972)	-1.943 (-2.108)
	1	1	1	1	1	0	-0.581 (-0.618)	-0.688 (-0.745)	-0.746 (-0.801)
	1	1	1	1	0	1	0.178 (0.088)	-0.104 (-0.158)	-0.069 (-0.140)
	1	2	0	1	1	0	0.197 (0.135)	-0.077 (-0.109)	-0.011 (-0.053)
O	1	2	1	1	2/3	1/3	-1.400 (-1.477)	-1.361 (-1.447)	-1.480 (-1.552)
	1	2	1	1	0	1	-1.442 (-1.583)	-1.698 (-1.836)	-1.816 (-1.957)
	1	2	1	1	1	0	-1.422 (-1.515)	-1.470 (-1.568)	-1.590 (-1.678)
	1	1	1	1	2	0	-0.564 (-0.623)	-0.866 (-0.960)	-0.913 (-1.013)
	1	1	1	1	1	1	0.358 (0.321)	0.255 (0.237)	0.292 (0.266)
	1	2	0	1	2	0	1.323 (1.220)	0.825 (0.789)	0.992 (0.932)
F	1	2	1	1	4/3	2/3	-0.374 (-0.398)	-0.366 (-0.391)	-0.394 (-0.416)
	1	2	1	1	2	0	-0.323 (-0.409)	-0.605 (-0.686)	-0.627 (-0.715)
	1	2	1	1	1	1	-0.349 (-0.389)	-0.401 (-0.441)	-0.427 (-0.467)

13.4 Dirac program: relativistic core potentials

In all relativistic calculations - scalar as well as spin-orbit - the relativistic atomic core densities and the relativistic atomic potentials *must* be made available to ADF on a file specified with the key COREPOTENTIALS. This is necessary only in the 'create' run of the atoms. In the molecular calculation this key is not required anymore. If supplied then the file must contain data for *all* atom types in the molecule, even for those atoms where relativistic aspects are expected to be negligible or that may not have a frozen core at all (such as Hydrogen). Excepted are any Ghost atoms (for instance for a BSSE calculation): these can not have any core potentials. This is tested by the program, internally, by looking at the nuclear charge and at the number of electrons belonging to an atom: if both numbers are zero, no (relativistic or other) core potential is allowed. Also the potential used in the ZORA kinetic energy operator in the SAPA (sum of neutral atomic potential approximation) method should be present on this file (which will be the case if the program DIRAC is used to generate this file).

Relativistic potentials can and should be generated with the auxiliary program dirac, see the next section, and the examples (in particular *Example: Gold electrodes* (page 798)).

The recommended way to generate atomic fragments and relativistic potentials is by using the *BASIS* (page 33) keyword.

The auxiliary program DIRAC, which is installed together with ADF, serves to compute relativistic frozen core potentials (and densities), necessary to apply the (scalar) relativistic option in ADF. `$ADFHOME/atomicdata` has a subdirectory `Dirac`, which contains input files for DIRAC for all atoms of the periodic table of elements. The *names* of the input files indicate the frozen core level: *Ag.3d* for instance is the input file for a calculation on a Silver atom with a frozen core up to and including the 3d shell (i.e.: 1s, 2s, 2p, 3s, 3p, and 3d). The frozen core level used in the DIRAC calculation defines the core data computed and should therefore match the frozen core level in the ADF Create run for the atom that it will be used for.

A DIRAC run with the inputs provided in the `$ADFHOME/atomicdata/Dirac/` involves a fully relativistic calculation on the atom (spin-orbit coupling, double group symmetries). It generates a file TAPE12 with the corresponding core potential and density (a table of values for a sequence of radial distance values). Other files produced by DIRAC should be removed after the DIRAC run; they are needed for other applications of the program but play no role here.

If you run DIRAC while a file TAPE12 already exists the computed core data will be written at the end of it, after the existing data. The program will assume, however, that the existing data on the file are also core-data from DIRAC runs, and may abort otherwise.

Starting from ADF2006.01 it is not necessary anymore to make one big TAPE12 which contains data for all atoms involved in the molecular calculation. Instead only in the ADF Create run for each atom one needs a TAPE12 which contains data for the atom that is created. The corresponding core data is written to the TAPE21 of this atomic fragment. In the molecular ADF run one then should not include the CorePotentials key, such that ADF will read core data on the TAPE21's of the atomic fragments. One can still use the CorePotentials key, but then one should proceed as in previous releases.

In previous releases (ADF2005 and older), if a CorePotentials file was needed for an *adf* calculation with the (scalar) relativistic option, the simplest approach was to subsequently run DIRAC for each of the involved atoms types. This builds up the TAPE12 file for this particular molecule. Then, specify in the ADF input which sections correspond to the distinct atom types. Alternatively, which we do not recommend, if you frequently perform relativistic ADF runs, with many different types of atoms, you might, once and for all, construct one big TAPE12 file, containing the core potentials of all atoms that you may ever need, and use that file again and again. Of course you need then to remember which section numbers correspond to which atoms.

Implied options

The DIRAC calculations imply the *local* Density Functional in its simple X-alpha approximation without any gradient corrections. Not the *scalar* relativistic but the *fully* relativistic Hamiltonian is used, including spin-orbit coupling. In ADF you may use the *scalar* relativistic Hamiltonian and most users will employ a more sophisticated lda than X-alpha, such as the default vwn (Vosko, Wilk, Nusair) formulas, and may in addition routinely apply gradient cor-

rections. The core potential may not exactly match the Fock operator applied in the molecular calculation. The effect is very small and one can neglect the discrepancy.

Input

The ASCII input files for DIRAC, as available in the directory `$ADFHOME/atomicdata/Dirac` (point nucleus) and `$ADFHOME/atomicdata/Dirac.gaussian` (finite nucleus), have a structure as described below. With this information you should be able to construct alternative input files, with other frozen cores for instance.

1. Title (60 characters at most). Plays no role

2. Ngrid, Nshell, rmin, rmax, Z, Xion, Anuc Ngrid=number of radial grid points, in which the core potentials are computed. Nshell=number of atomic orbital shells rmin, rmax=minimum and maximum radial grid values Z=nuclear charge Xion=net charge of the 'atom' Anuc=atomic weight

3. Pinit, Pfinal, eps, del, delrv Pinit, Pfinal= initial and final density iteration averaging factors. Each iteration cycle changes the actual averaging factor by taking the average of the previous and the final one, starting with the 'initial' one. eps=Exp(-sqrt(eps)) is set to zero, so eps determines the exponential underflow control. del=absolute convergence criterion for orbital eigenenergies. delrv=convergence criterion on the potential (multiplied by the radial distance *r*).

4. Idirc, Nmax, Ndebu, Nprin, Ipun, Ircor, Iwcor Idirc=zero for non-relativistic, otherwise one. Nmax=maximum number of iterations allowed to reach convergence. Ndebu=non-zero for additional output (for debugging purposes mainly) Nprin=print parameter. Use 2 or larger to get the orbitals printed. Ipun=punched output is produced if Ipun is non-zero. (out-of-date) Ircor=number of core orbitals from the fully relativistic run, to be kept frozen in the subsequent (if any) first-order perturbation calculation. Iwcor=number of core orbitals used to construct the core density and core potential, that are output on TAPE12. So, here you specify the relativistic core.

5. Xalph, Xlatt, Rnuc Xalph= Exchange parameter in the Xalpha formalism. Xlatt=Coulomb tail parameter Rnuc=size of nuclear radius, in bohr. If set to 1.0 or larger, it is recomputed as 0.0000208*Anuc**(1/3)

6. For each orbital shell: N, L, J, E, Z, D N, L, J = The usual orbital quantum numbers. J is used only for relativistic runs. E = Initial estimate of orbital energy, in atomic units. Z = Number of electrons on the shell D = Initial estimate of the error in the orbital energy

7. Icorp, Npcl, Demp, Peps Icorp = If 1 (one): Do a first order perturbation calculation after the fully relativistic run. This option plays no role in the current application for ADF. Npcl = Maximum number of cycles in the perturbation calculation. Demp = Damping factor in the perturbation iterations Peps = Convergence criterion in the perturbation iterations.

13.5 Symmetry

13.5.1 Schönfliess symbols and symmetry labels

A survey of all point groups that are recognized by ADF is given below. The table contains the Schönfliess symbols together with the names of the subspecies of the irreducible representations as they are used internally by ADF. The subspecies names depend on whether single-group or double-group symmetry is used. Double-group symmetry is used only in relativistic spin-orbit calculations.

Note that for some input of TDDFT (Response) calculations, other conventions apply for the subspecies. This is explicitly mentioned in the discussion of that application.

Table 13.2: Schönfliess symbols and the labels of the irreducible representations.

Point Group	Schönfliess Symbol in ADF	Irreducible representations in single-group symmetry	Irreducible representations in double-group symmetry
C_1	NOSYM	A	A1/2
R^3	ATOM	s p d f	s1/2 p1/2 p3/2 d3/2 d5/2 f5/2 f7/2
T_d	T(D)	A1 A2 E T1 T2	E1/2 U3/2 E5/2
O_h	O(H)	A1.g A2.g E.g T1.g T2.g	E1/2.g U3/2.g E5/2.g
		A1.u A2.u E.u T1.u T2.u	E1/2.u U3/2.u E5/2.u
$C_{\infty v}$	C(LIN)	Sigma Pi Delta Phi	J1/2 J3/2 J5/2 J7/2
$D_{\infty h}$	D(LIN)	Sigma.g Sigma.u Pi.g Pi.u	J1/2.g J1/2.u J3/2.g J3/2.u
		Delta.g Delta.u Phi.g Phi.u	J5/2.g J5/2.u J7/2.g J7/2.u
C_i	C(I)	A.g A.u	A1/2.g A1/2.u
C_s	C(S)	AA AAA	A1/2 A1/2*
C_n	C(N), n must be 2	A B	A1/2 A1/2*
C_{nh}	C(NH), n must be 2	A.g B.g A.u B.u	A1/2.g A1/2.g* A1/2.u A1/2.u*
C_{nv}	C(NV), n<9	A1 A2 B1 B2 E1 E2 E3 ...	E1/2 E3/2 E5/2 ...
		odd n: without B1 and B2	odd n also: An/2 An/2*
D_n	D(N), n<9	n=2: A B1 B2 B3	E1/2 E3/2 ...
		other: A1 A2 B1 B2 E1 E2 E3 ...	
		odd n: without B1 B2	odd n also: An/2 An/2*
D_{nh}	D(NH), n<9	n=2: A.g B1.g B2.g B3.g A.u B1.u B2.u B3.u	even n: E1/2.g E1/2.u
		other: A1.g A2.g B1.g B2.g E1.g	E3/2.g E3/2.u ...
		E2.g E3.g ... A1.u A2.u B1.u ...	
		odd n: AA1 AA2 EE1 EE2 ...	odd n: E1/2 E3/2 E5/2 ...
		AAA1 AAA2 EEE1 EEE2	
D_{nd}	D(ND), n<9	even n: A1 A2 B1 B2 E1 ...	even n: E1/2 E3/2 ...
		odd n: A1.g A2.g E1.g E2.g ... E(n-1)/2.g	odd n: E1/2.g E1/2.u E3/2.g E3/2.u ...
		A1.u A2.u E1.u E2.u ... E(n-1)/2.u	An/2.g An/2.u An/2.g* An/2.u*

Most labels are easily associated with the notation usually encountered in literature. Exceptions are AA, AAA, EE1, EEE1, EE2, EEE2, etcetera. They stand for A', A'', E1', E1'', and so on. The AA, etc. notation is used in ADF to avoid using quotes in input files in case the subspecies names must be referred to.

The symmetry labeling of orbitals may depend on the choice of coordinate system. For instance, B1 and B2 representations in C_{nv} are interchanged when you rotate the system by 90 degrees around the z-axis so that x-axis becomes y-axis and vice-versa (apart from sign).

Labels of the symmetry subspecies are easily derived from those for the irreps. For one-dimensional representations they are identical, for more-dimensional representations a suffix is added, separated by a colon: For the two- and three-dimensional E and T representations the subspecies labels are obtained by adding simply a counting index (1, 2, 3) to the name, with a colon in between; for instance, the EE1 irrep in the Dnh pointgroup has EE1:1 and EE1:2 subspecies. The same holds for the two-dimensional representations of $C_{\infty v}$ and $D_{\infty h}$. For the R3 (atom) point group symmetry the subspecies are p:x, p:y, p:z, d:z2, d:x2-y2, etc.

All subspecies labels are listed in the Symmetry section, very early in the ADF output. To get this, perform a quick run of the molecule using the STOPAFTER key (for instance: stopafter config).

13.5.2 Molecular orientation requirements

ADF requires that the molecule has a specific orientation in space, as follows:

- The origin is a fixed point of the symmetry group.

- The z-axis is the main rotation axis, xy is the σ_h -plane (axial groups, C(s)).

- The x-axis is a C_2 axis (D symmetries).

- The xz-plane is a σ_v -plane (C_{nv} symmetries).

- In T_d and O_h the z-axis is a fourfold axis (S_4 and C_4 , respectively) and the (111)-direction is a threefold axis.

If the user-specified symmetry equals the true symmetry of the nuclear frame (including electric field and point charges) the program will adapt the input coordinates to the above requirements, if necessary. If no symmetry has been specified at all ADF assumes you have specified the symmetry of the nuclear frame, accounting for any fields. If a subgroup has been specified for the molecular symmetry the input coordinates will be used as specified by the user. If a Z-matrix input is given this implies for the Cartesian coordinates: first atom in the origin, second atom on the positive x-axis, third atom in the xy-plane with positive y value.

13.6 Binary result files, KF browser

13.6.1 TAPE21

TAPE21 is the general result file of an ADF calculation. It is a KF file: Direct-Access, binary, and keyword driven. It contains information about the calculation. You can use it as a fragment file in a subsequent calculation on a bigger molecule, where the current one may be a part, or in an analysis program.

The contents of TAPE21 is keyword-accessible and you may use the KF utilities (see Scripting Section) for conversion of TAPE21 from binary to ASCII format and vice versa. This facility is also useful when you intend to use a TAPE21 result file produced on one type of hardware, for a continuation run on a quite different computer: Transform the binary file to ASCII format with the KF utilities on the first machine. Then transport the ASCII file to the other machine, and make a binary out of it again.

Another utility (*pkf*) can be used to obtain a summary of the contents of TAPE21. The output should be more or less self-documenting: all variables are listed by name, type (integer, real, ..) and size (number of array elements) and grouped in named sections.

The data on TAPE21 is organized in Sections which group together related data. Each section contains a number of variables. Each variable may be an array or a scalar and may be integer, real, logical or character type.

A complete dump of the contents of TAPE21 is obtained with *dmpkf*. The resulting ASCII file contains for all variables on the file:

- The name of the section it belongs to;

- The name of the variable itself;

- Three integers coding for the data of the variable:

- The number of data elements reserved on the file for the variable;

- The number of data elements actually used for the variable. In virtually all cases the number of *used* elements is equal to the number of *reserved* elements. The number of *used* elements is relevant for interpreting the data, the number of *reserved* elements has only relevance for the management of data on the file by kf-specific modules and utilities;

- An integer code for the data type: 1=integer, 2=real, 3=character, 4=logical;

- The variable value(s).

A typical case of the contents of TAPE21 obtained by *dmpkf* operating on the binary TAPE21 file from an optimization run on H2O would be:

contents of TAPE21	comment
General	name of (first) section
file-ident	name of (first) variable in the current section (General)
6 6 3	characteristics of the data: 6 elements reserved on file for the variable, 6 data elements actually used, 3=integer code for the data type: character
TAPE21	Value of the variable fileident in the section General.
General	again: name of the section
title	name of the (second) variable
80 80 3	reserved and used number of data elements (both 80), and the data type code (3: character)
Water Geometry Optimization with Internal Coordinates	value
(etc.)	(etc.)

A description of the various utilities that can be used to process TAPE21 can be found in other parts of this ADF manual.

Contents of TAPE21

Follows a survey of the sections and variables on TAPE21. Details may differ between different run types (SinglePoint, Frequencies...). Most items should be self-explanatory. Some are only significant for internal proceedings of the program and will not be explained in detail. The sections are described in an order that corresponds to the order in which they are generated and hence printed by the KF utility programs. However, the order of generation depends somewhat on the type of application, so some difference may be found when comparing to your own TAPE21 printout.

Note that variable and section names may contain spaces: these are significant.

A special section is the 'SUPERINDEX' section, which is in fact a table-of-contents that lists all the sections in the file, with technical information regarding their position on the file, the amount of data corresponding to that section and similar items. The SUPERINDEX section is not discussed further here. See the KF documentation for more details.

Section General

General information about the calculation and the file

fileident Name of the file. Here: TAPE21

jobid ADF release number with date and time of the calculation

title Title of the calculation. This may have been set in the input file, or be internally generated. In a create run it is picked up from the Create basis set file (if no input value for the title key has been given).

runtype The type of calculation, for instance SinglePoint or Frequencies

nspin 1 for a spin-restricted calculation, 2 for spin-unrestricted

nspinf Similar for the fragment occupation numbers as they are used in the calculation, See the key FRAGOCCU-PATIONS

ldapot An integer code for the applied LDA part of the XC potential functional used in the SCF. 1 for VWN, 2 for VWN+Stoll ...

xcparv X-alpha parameter value. Only relevant for the X-alpha LDA potential, meaningless if another LDA potential functional has been selected.

ldaen As for ldapot: integer code for the LDA part of the Density Functional, now however pertaining to the (post-SCF) energy evaluation. Usually ldaen and ldapot are identical. See the key XC for details.

xcpare As xcparv, but now for the energy evaluation.

ggapot Specification (string) of the GGA part of the XC potential used in the SCF, for instance 'Becke Perdew'. If no GGA potential is applied, the string ggapot is empty.

ggaen Similar for the GGA part of the XC energy evaluation

iopcor Code for usage of frozen core: 1=use frozen cores, 0=pseudopotentials. Pseudopotentials are not supported anymore in ADF, so this variable must always be 1.

electrons The number of valence electrons Note that this is not necessarily the same as what may consider, chemically, as the valence space. Rather, it equals the total number of electrons in the calculation minus the electrons in the frozen core orbitals.

unit of length Transformation factor between input-used geometrical units (for distances) and atomic units (bohr). If input of, say, the atomic coordinates is in Angstrom, the unit of length is approximately 1.89

unit of angle Similar for angles. Internal units in the program are radians. Input (bond and dihedral angles) may be in degrees, in which case the unit of angle equals approximately 0.017

Section Geometry

Geometrical data such as number of atoms, coordinates, etc: Most variable names should be self-explanatory

grouplabel Point group symmetry (string) used in the calculation, for instance O(H). This may be set in the input file.

Geometric Symmetry Auto-determined ('true') symmetry (considering the nuclear frame and any external fields, but not taking into account any user-defined MO occupation numbers and hence the electronic charge distribution.

symmetry tolerance Threshold for allowed deviation of input atomic coordinates from symmetry to be detected or verified.

orient Affine transformation (3,4 matrix: rotation and translation) between the input coordinates and the frame in which the program processes the atoms. ADF has certain orientation requirements for all supported point group symmetries and may rotate and translate the input coordinates accordingly.

oinver The inverse transformation of orient

lrotat A logical flag to signal whether or not a rotation has been applied between the input frame and the internally used frame.

nr of fragmenttypes The number of distinct types of fragments

nr of dummy fragmenttypes Idem, but counting only dummy atom fragments. A dummy fragment, if it exists, must consist of one single (dummy) atom.

fragmenttype Names (string) of the fragment types.

fragment mass Sum of atomic masses in the fragment.

fragment charge An array with 3 values per fragment type (nftypes,3): 1=sum of nuclear charges, 2=sum of effective nuclear charges (discounting for the frozen core shells), 3=nr of valence electrons

fframe Signals whether or not special local coordinate frames are used for the atoms. Usually this is not so, in which case the variable has the value DEFAULT. fframe is an array that runs over the atoms. See the 'z=' option to the data in the ATOMS input key block.

cum nr of fragments An array (0:nftyps) that gives the total number of fragments for the fragment types up to and including the indexed one. The ordering of fragments and fragment types is printed in the standard output file.

nr of fragments The total number of fragments in the calculation This equals the last element of the previous variable 'cum nr of fragments'

nr of dummy fragments The total number of fragments that each consist of a single dummy atom.

fragment mapping Affine transformation matrices (3,4: rotation and translation), one for each fragment in the molecule, that transform the fragment coordinates as they are on the fragment file(s), to the actual position of the fragments in the molecule.

cum nr of atomtypes An array (0:fragmenttypes) that counts the number of atom types up to and including the indexed fragment type.

nr of atomtypes Total number of atom types in the molecule. Must equal the last element of the 'cum nr of atomtypes' array

nr of dummy atomtypes Similar, now counting only the atom types consisting of a dummy atom.

atomtype Names (strings) of the atom types

mass Atomic masses: array running over the atom types. Compare 'fragment mass'.

charge Similar as for 'fragment charge', but now the values per atom type.

cum nr of atoms An array (0:atomtypes) that counts the number of atoms up to and including the indexed atom type.

nr of atoms Total number of atoms. Must equal the last element of the array 'cum nr of atoms'.

nr of dummy atoms Total number of dummy atoms

atmcrd Type of atomic coordinates in input: CART (Cartesian) or ZMAT (Internal).

kmatrix InputOrder The connection matrix listing (and referencing) the atoms in the order as they were in the input file. This ordering aspect is significant because internally the program reorders the atoms and groups them together by atom type and fragment type. Hence it is relevant to know what ordering (input- or internal-) is assumed in data arrays.

zaxis For each atom the direction of the local z-axis. Normally this is identical to the standard (0,0,1), but it may be different for analysis purposes. See the 'z=' option to the data records in the ATOMS block.

fragment and atomtype index An integer array (natoms,2) that specifies for each atom the fragment and the atom type it belongs to.

atom order index An integer array (natoms,2) that defines the re-ordering of atoms between the list in the input file and the internally used list (which is driven by fragment types, fragments, atom types; dummies come last). The internally used list can be derived from the printout of the fragments, early in the standard output.

kmatrix The connection matrix using the internally applied ordering of atoms

xyz Cartesian coordinates of the atoms, in the internally used ordering of atoms

xyz Inputorder Similar, but now for the ordering of atoms as in the input file.

zmatrix Internal (Z-matrix) atomic coordinates

zmatrix Inputorder Internal coordinates in the input-order of atoms

Atomic Distances Inter atomic distance matrix

ntyp Number of atom types, not counting dummy atoms,

nqptr A cumulative counting array, very similar to 'cum nr of atoms' Differences: it runs only over 'ntyp' atom types (not including dummy atoms) and its indexing as well as its values are shifted by one: nqptr(k) is the total number of atoms plus one, counting the atom types up to and including #(k-1)

nnuc Total number of non-dummy atoms

qtch Nuclear charges of the non-dummy atoms

qeff Effective nuclear charges (subtracting charge for the frozen core shells) of the non-dummy atoms

nfragm Total number of non-dummy fragments

---

nofrag_1 Integer array specifying for each non-dummy atom the fragment it belongs to.

nofrag_2 Integer array specifying for each non-dummy atom the fragment type it belongs to

nuclab Names of the non-dummy atom types.

Section Fragments

(To be completed)

FragmentFile Names of all used fragment files

FragRun Ident,Title Job identification and title of each fragment run that is used in the current molecule

Section AtomTypes

(To be completed)

Section Properties

AtomCharge Mulliken Atomic charges derived from Mulliken population analysis.

Dipole Dipole moment in atomic units.

FragmentCharge Hirshfeld Fragment charges derived from Hirshfeld analysis

AtomCharge_initial Voronoi Atomic charges derived from Voronoi analysis for the initial (sum-of-fragments) charge density

AtomCharge_SCF Voronoi Similar as the previous item, but now for the SCF density

Electrostatic Pot. at Nuclei Coulomb potentials at the positions of the atoms, not including the contribution from the nucleus itself

Section Basis

Information about the (valence) basis set

nbset The total number of basis 'sets', where a 'set' here means a Cartesian function set (3 for a *p*-type function, 6 for a *d*-type function, and so on), given by an entry in the 'list-of-basis-functions' in the data base file.

nbaspt Cumulative number of basis sets (see previous variable, for 'set'), on a per atom type basis. Only non-dummy atoms (type) are considered. nbaspt(k) is 1+nr-of-basis sets up to, but not including atom type #k

nqbas Main quantum number of each basis set. A 1s function has nqbas()=1

lqbas Angular momentum quantum number of each basis set. The current implementation of ADF supports only *s, p, d,*and* f* basis functions, so the allowed lqbas values are 0, 1, 2, and 3

alfbas The exponential decay parameters of the STO functions in the basis set

basnrm Normalization coefficients for the basis sets

naos The total number of basis functions, counting all Cartesian polynomials and all copies of the functions on the atoms of the pertaining atom type

nbos The total number of Cartesian basis functions, *not* counting the copies of the functions on the different atoms of the atom type: the functions are defined per atom type and are (for nbos) counted only once. The next few variables relate to lists of basis functions that run from 1 to nbos: all the Cartesian polynomials, but counting the function only once per atom type. Essentially, this means counting all functions with distinct characteristics (apart from their geometrical center).

nbptr Index array of the nbos functions, where the entries are the cumulative numbers of functions (+1) up to, but not including the atom type. The size of the array is (ntyp+1): one plus the number of (non-dummy) atom types.

kx Powers of x of the nbos Cartesian STO basis functions

ky Powers of y of the nbos Cartesian STO basis functions

kz Powers of z of the nbos Cartesian STO basis functions

kr Powers of r of the nbos Cartesian STO basis functions

alf Exponential decay factors of the nbos Cartesian STO basis functions

bnorm Normalization factors for the nbos Cartesian STO basis functions

nprta Consider a list of all (naos) Cartesian STO basis functions, including copies of the functions on all atoms of the same atom type. Build that list by first taking all true valence functions on all atoms (loop over atom types, inner loops over atoms, inner loop over basis sets of the atom type, inner loop over Cartesian polynomials for the function set), then all auxiliary core-orthogonalization functions (similar loop structure). nprta(i) gives the index in that list of function #*i*, where *i* corresponds to a similar list of all naos functions in which the core and valence subsets are not separated.

norde An array that runs over the non-dummy atom types. Each element gives the maximum of the main quantum number for all STO basis and fit functions corresponding to that atom type.

lorde As norde, but lorde applies to the angular momentum quantum numbers.

Section Core

Information about frozen core orbitals and the Slater-type exponential functions used to describe them.

nrcset The number of STO function sets to describe the frozen core orbitals in the calculation. The array is sized (0:llqcor,1:ntyp). llqcor is the maximum l-value in core orbitals (3), ntyp is the number of non-dummy atom types.

nrcorb An array (0:llqcor, 1:ntyp) specifying the number of frozen core orbitals per *l*-value and per non-dummy atom type.

ncset The total number of core expansion STO function sets, not counting copies on all atoms, and not counting the Cartesian polynomials (1 value per *p*-set, et cetera)

ncorpt Index array: 1 + cumulative number of core expansion sets up to, but not including, the indexed atom type. The array runs from 1 to ntyp+1

nqcor Main quantum numbers for the core expansion sets

lqcor Angular momentum quantum numbers for the core expansion sets.

alfcor Exponential decay factors for the core expansion sets.

cornrm Normalization factors for the core expansion sets.

ncos Total number of core expansion functions, counting all copies on different atoms of each atom type, and counting all Cartesian polynomials.

nccpt Index array: 1 + cumulative number of core orbitals, counting all copies on different atoms and all Cartesian (sub) functions.

ncptr Similar, but applying to the STO core expansion functions.

ccor All core expansion coefficients, which express the core orbitals in the core expansion functions. The array stores the expansion coefficient sequence for each core orbital shell (not for each Cartesian sub function) and only one sequence per orbital per atom type (no duplication for the different atoms of the atom type).

npos An index array. For each atom type: the index where its data are stored on the TAPE12 core data file. npos(k) may be zero if no data for atom type #k are available on TAPE12.

kcos The total number of core expansion functions, like ncos, but now counting only the truly independent functions. For instance: 5 functions per *d*-set, while in ncos there are 6 functions per *d*-set. The *s*-type combination in the 6-membered *d*-set is in the calculation projected out and does not represent a degree of freedom.

s The (kcos,kcos) overlap matrix of the core expansion functions. Note that, since the dimension is (kcos,kcos), the *s*-type combination has been eliminated, and likewise for the 3 *p*-type functions in each *f*-set.

idfcor Integer that indicates whether or not the core set contains *d*- and/or *f*-type functions. 1=yes, 0=no

nd Total number of *d*-type core orbital sets (not counting the Cartesian sub functions)

nf Total number of *f*-type core orbital sets (not counting the Cartesian sub functions)

ndorb An array running over the *d*-type core orbital sets (loop over atom types, loop over atoms, loop over core orbitals with *l*=2). *It gives for each the index of the orbital (the first of the Cartesian subset) in the overall list of all core orbitals in the molecule (including the spurious *s*-type functions in the d-sets, and so on)*

nforb Similar as ndorb, but now for the *f*-type core orbitals.

cmat Overlap matrix between core-orbitals (ncos, counting all Cartesian functions including the *s*-type function in each *d*-set, et cetera), and the basis functions. In the list of basis functions, all core functions (the auxiliary orthogonalization functions) come before all true valence basis functions, see array NPRTA.

Section Fit

This section stores information about the fit functions, which are used for the Coulomb potential evaluation.

Unrestr.SumFrag A logical that flags whether or not the fit coefficients have been set and stored for the sum-of-fragments, but adjusted for the unrestricted fragments option (see the keys UnrestrictedFragments, ModifyStart-Potential).

coef_SumFrag Fit coefficients pertaining to the sum-of-fragments charge density.

coef_SCF SCF fit coefficients.

nfset Total number of fit function sets (not counting the Cartesian sub functions, not counting the copies of the functions on the atoms of an atom type)

nfitpt Index array: 1+the total number of fit function sets up to, but not including, the indicated atom type.

nqfit Main quantum numbers of the fit sets

lqfit Angular momentum quantum numbers of the fit sets

alffit Exponential decay factors of the STO fit sets.

fitnmr Normalization factors for the STO fit sets.

nfos Total number of Cartesian fit functions, not counting copies on all atoms of an atom type, but including all (for instance, 6 for a *d*-set) Cartesian sub functions.

nfptr Index array: 1+ total number of Cartesian (see variable nfos) fit functions, up to but not including the indicated atom type.

nprimf Total number of Cartesian ('primitive') fit functions, counting also the copies on all atoms of each atom type.

nsfos The total number of fully symmetric (A1 symmetry) fit function combinations that represent the true dimension (variational freedom) of the space of fit functions in the calculation.

na1ptr Index array, like nfptr, but applying to the nsfos symmetric function combinations.

niskf This refers to an atom-limited symmetry combination of primitive fit functions, in the code and some documentation indicated as a 'g'. A 'g' is the specific part of a molecule-wide A1 fit function combination (see nsfos) that consists of all the terms that are centered on one particular atom. The number niskf gives the total number of such 'g' function combinations. To clarify this, consider an A1 fit function combination in the molecule. Assume, that it consists of a specific linear combination the following functions: a p-x function on atom A, its partner p-y function, and the corresponding p-x and p-y functions on atom B. (Atoms A and B must

be symmetry equivalent). In this example we have one A1 function (in the list of nsfos such functions) and two 'g''s. Each 'g' consists of a p-x and a p-y function combination on a specific atom.

iskf Compound index array. It runs over the niskf 'g' fit function combinations and has 4 entries for each function (1:4,1:niskf). The meaning of the entries is as follows. #1=number of the fit set (not counting the copies of fit functions on different atoms of an atom type, and not counting the Cartesian sub functions) this 'g' belongs to. #2=index where the combination coefficients for this 'g' start in the arrays cofcom and numcom (see next). #3=number of terms in the expansion of this 'g'. #4=number of the molecular fit A1 function combination this 'g' belongs to.

na1cof Length of the arrays numcom and cofcom, see next

numcom Numcom (and cofcom) consists of a sequence of smaller sub arrays. Each sub array gives the expansion of a 'g' function in terms of the Cartesian functions in the pertaining fit function set. The elements of numcom specify the particular Cartesian sub functions that participate in the expansion. Its values are therefore limited to lie between 1 and (L+1)(L+2)/2, where L is the maximum *l*-value occurring in the fit function sets.

cofcom Compare numcom: cofcom gives the actual expansion coefficients for the expression of a 'g' function in primitive Cartesian fit functions.

Section Num Int Params

Numerical integration parameters: the general precision parameter, but also more technical parameters used by the grid-generating modules.

method Label of the method used to generate the grid. Usually: 'polyhedra'

accint min Minimum integration precision parameter. It is the lower bound of the range in which the value of the actual numerical integration precision parameter may vary.

accint max Maximum value of the precision general parameter

accint Actual value of the precision parameter. This variable governs by default almost all other integration parameters.

ldim In fact, this a geometric parameter: the number of dimensions in which the system is periodic. For molecules this is zero.

PointChargeTypes The number of point charges types used in the calculation. Point charges belong to a different point charge type if, and only if, their strengths are not equal.

accsph The precision parameter that determines the (radial) integration grid in the atomic spheres

accpyr The precision parameter that determines the general precision level of the grid in the atomic polyhedra

accout The precision parameter that determines the general precision level of the grid in the outer region

accpyu The precision parameter that determines the 1D grid along the first direction in the quadrangles and triangles of the bases of the atomic pyramids

accpyv The precision parameter that determines the 1D grid along the second direction in the quadrangles and triangles of the bases of the atomic pyramids

accpyw The precision parameter that determines the 1D radial integration in the atomic pyramids, between the atomic sphere surfaces and the pyramid basis

frange Estimated maximum range of functions, to determine how far the integration grid has to extend outwards, away from the molecule

rspher An array with the radii of the atomic sphere (a value per atom type)

rsph0 The smallest sphere radius

rsphx The largest sphere radius

dishul The distance between the innermost boundary planes, which separate the atomic pyramids from the outer region, and the surfaces of the outermost atoms

nouter The number of intervals in which the outward (radial) integration in the outer region is broken up

outrad The precision parameter that determines the outward radial integration in the outer region

outpar The precision parameter that determines the 2D integrals in the outer region parallel to the boundary planes

linteg An array with maximum angular momentum quantum numbers (one value per atom type), to determine the angular integration grid in the atomic spheres

lintgx Maximum of linteg()

linrot Angular momentum quantum number to determine the rotational integration parameter around the molecular axis (in linear molecules only)

ntyps The number of atom types as seen by the numerical integration grid generator. This means in practice: the number of non-dummy atom types plus the number of point charge types.

nnucs The number of atoms as seen by the numerical integration grid generator. This means in practice: the number of non-dummy atoms plus the number of point charges.

qatm Nuclear charges for all ntyps atom types

nratst1 The numerical integration grid generator automatically determines the symmetry of the nuclear (nnucs atoms!) frame and then puts the atoms in sets of symmetry equivalent ones. nratst1() is an array (0:ntyps) that contains the cumulative number of atoms in the symmetry sets. nratst1(k) is the total number of atoms in the sets up to and including set #k

xyzatm Cartesian coordinates of the atoms.

linteg all Similar to array linteg(), extended to include also the point charge types

npowx Maximum power of the radial variable r, in the set of test functions that the grid generator uses to tune the grid

alfas An array that stores the exponential decay factors of all test functions, ordered by atom type and by the power of the radial variable r.

Section Symmetry

Symmetry related data.

nogr The number of symmetry operators in the point group used in the calculation. NB, for the special cases of infinite symmetries, only the operators corresponding to finite elements are counted. Therefore, ATOM has nogr=1 (only the unit operator); C(LIN) has nogr=1, D(LIN) has nogr=2.

faith An array that stores all the (3,3) symmetry operator matrices in the real space representation

nsetat The number of sets of symmetry equivalent atoms under the used symmetry

napp An array that stores for each atom the number of the symmetry set it belongs to

notyps An array that stores for each set of symmetry equivalent atoms, the atom type to which the set belongs

noat Map between the normal list of atoms and the symmetry sets. When you loop over the symmetry sets and, inside, loop over the atoms in each set, you thereby run over the index of noat(). The value points to the position of that atom in the original (not set-ordered) list.

ntr An array (nogr,nnuc) that stores for the each atom A and each symmetry operator R, the atom onto with A is mapped by R. The row index runs over all symmetry operators, the column index over the atoms.

npeq The number of symmetry unique pairs of atoms

jjsym An array that runs over the npeq sets of symmetry equivalent atom pairs. Its value gives for the indicated set the index of a (c.f. the first) atom pair in that set.

jasym An array that runs over the npeq sets of equivalent atom pairs. Its value gives for the indicated the set the number of pairs in that set.

ja1ok An array (1:npeq), with values 0 or 1. 1=the pair density can be fitted using A1 fit functions only. 0=all fit functions (on the involved atoms) are to be used. The value 1 may arise because of symmetry properties, or because the distance between the atoms is so large that the inaccuracy from using only A1 fit functions can be neglected.

ntr_setat A condensed variety of array ntr: the columns are not the atoms, but the nsetat sets of symmetry equivalent atoms. The value is the index of the atom, onto which a representative (c.f. the first) atom of the indicated symmetry set is mapped by the given symmetry operator.

igr A code that fixes, together with nogr and ngr, the point group symmetry. See the header of routine adf/maisya for a list

ngr One of the code components that fix the symmetry group. See routine adf/maisya

grouplabel Schönfliess symbol as used in ADF

nsym The number of symmetry representation (including subspecies) used in the calculation.

norb For each of the nsym representations the number of basis function combinations (SFOs) that belong to it.

nfcn For each of the nsym representations the number of primitive atom centered basis functions that participate in the representation.

ncbs For each of the nsym representations the number of core orthogonalization functions that participate in the representation.

jsyml For each of the nsym representations: if it belongs to a one-dimensional irrep, the value is 1, otherwise: for the first subspecies in the irrep the value is the dimension of the irrep, for the other subspecies in the same irrep the value is 0

symlab For each of the nsym representations the label (string) of the representation

norboc An array (-2:2,nsym). The column runs over the symmetry representations. The positive row indices (1,2) specify for spin-A and spin-B (the latter only if the calculation is spin-unrestricted), the highest non-empty orbital. The negative indices (-1,-2) specify for spin-A and spin-B (if the unrestricted fragment option is used) the total number of non-empty SFOs. The zero row index specifies the number of non-empty SFOs, before applying any fragment occupation changes.

Section Spin_orbit

(To be completed)

Section Energy

XC energies 16 elements of an array enxc(2,2,4): exchange-correlation energies of various charge densities: first index: 1=exchange term, 2=correlation term second index: 1=lda tern, 2=gga term third index: 1=energy of fragments (summed over fragments), 2=energy of sum-of-fragments density, 3=energy of orthogonalized fragments, 4=SCF.

Pauli TS Correction (LDA) Correction to the 'Transition State' method to compute terms in the bonding energy, in this case the Pauli exchange energy term. The Pauli TS Correction is not separately printed in the standard output file, but included in the Pauli interaction term.

Pauli FitCorrection The first-order correction to the Pauli exchange interaction term, for the error in the Coulomb energy due to the fit incompleteness. This correction term is not printed in the output file but included in the Pauli interaction term

Elstat Core terms An obsolete variable, not used in the energy computation

---

Elstat Fitcorrection The first-order correction to the electrostatic interaction term (putting the fragments together, without any relaxation of Pauli orthogonalization), for the error in the Coulomb energy due to the fit incompleteness

Orb.Int. FitCorrection The first-order correction to the electrostatic interaction term in the SCF relaxation energy (Orbital Interactions), for the error in the Coulomb energy due to the fit incompleteness. This term is not printed (anymore) separately, but incorporated in the symmetry-specific interaction terms.

Orb.Int. TSCorrection (LDA) The difference between the representation-specific orbital interaction terms added, and a straightforward computation of the SCF relaxation energy is the result of the neglect of higher order terms in the Taylor expansion that underlies the 'Transition State' method. This difference, therefore, corrects exactly this neglect. It is not printed separately anymore in the output, but incorporated in (distributed over) the representation-specific orbital interaction terms.

Ebond due to Efield Bond energy term due to any homogeneous electric field

Corr. due to Orthogonalization For analysis purposes, the concept of 'orthogonalized fragments' has been introduced and the bonding energy is split in a part that describes the difference between the sum-of-fragments situation and the orthogonalized-fragments density at the one hand, and the SCF relaxation (from the orthogonalized fragments density) at the other. Both terms contain a first order fit correction term. The result of adding the two parts is not identical to computing the total bonding energy directly and applying the first order correction to that approach. The difference is given by this term, which therefore corrects for the additional second order fit errors caused by using the orthogonalized fragments split-up

SumFragmentsSCF FitCorrection The 'true' first order fit correction for the complete bonding energy, resulting from a direct calculation that takes the sum-of-fragments as starting point and the SCF as final situation, without the intermediate step of orthogonalized fragments.

Pauli Efield The contribution to the Pauli interaction energy due to any electric field

Orb.Int. Efield The contribution to the SCF relaxation energy (orbital interactions) due to any electric field

Electrostatic Interaction The electrostatic interaction energy including any first order fit correction (if computed from the fit density)

Pauli Total The Pauli exchange (orbital orthogonalization) interaction energy

Steric Electrostatic INCORRECT. Do not use. The electrostatic interaction energy including any first order fit correction (if computed from the fit density)

Steric Total The total steric interaction energy, consisting of the electrostatic and the Pauli interactions

Orb.Int. *Irrep* *Irrep* stands for one of the irreps of the point group symmetry. The value gives the orbital interaction (SCF relaxation) term for that symmetry representation

Orb.Int. Total The total orbital interaction energy

SCF Bond Energy Total bonding energy

elstat INCORRECT. Do not use. Electrostatic interaction energy. Same as the 'Electrostatic Interaction' variable in this section

Bond Energy Total bonding energy, same as the 'SCF Bond Energy' variable

Pauli Kinetic Kinetic energy term in the Pauli exchange interaction energy

Pauli Coulomb Coulomb energy term in the Pauli exchange interaction energy

Pauli Kinetic+Coulomb Sum of the kinetic and Coulomb terms in the Pauli exchange interaction energy

Section Point_Charges

NumberofPointCharges The total number of point charges used

PointCharges The array with point charge values: (4,np), where np is the number of point charges and the 4 components are, respectively, the x y z components and the strength.

Section GeoOpt

Optimization data.

Where references are made to the list of atoms, the atoms are assumed to be in internal order. This may be different from the input-list of atoms.

nfree number of independent optimization variables

idfree indices (3,nr-of-atoms) for all atomic coordinates referring to the optimization variables (values 1..nfree) and/or LinearTransit parameters (values nfree+k, k being the k-th LT parameter). A zero value means that the coordinate is frozen.

all freedoms A logical the flags whether or not all fundamental degrees of freedom in the system are allowed to vary. This is not the case when constraints are applied.

Gradients The most recent values for the derivatives of the energy with respect to the atomic coordinates (cartesian or z-matrix, depending on the type of optimization variables).

Displacements The most recent step executed for the atomic coordinates (optimization variables)

kmatrix The connection matrix.

Hessian_CART The Hessian matrix (second derivatives) as a n*n matrix, in the Cartesian coordinates representation. Note that the reduced storage mode (typically, Fortran upper-triangular) is not applied.

Hessian_ZMAT Same, but now in the internal coordinates representation

Hessian inverted_CART The *inverted* Hessian, in Cartesian coordinates

Hessian inverted_ZMAT Likewise, in internal coordinates

Note: in most cases only one, or maybe two of the Hessian cases are present on TAPE13. They can be transformed into each other quite easily. The order of atoms is the same as in the input.

xyz old cartesian coordinates at previous geometry cycle

zmatrix old idem for internal coordinates

Section TS

Information about the Transition State search

modtrc Defines the initial search direction. Positive value *n*: the n-th Hessian eigenvector (default:1). Negative value *n*: the Hessian eigenvector with the largest (absolute value) component in the n-th optimization variable

itrace Index of the Hessian eigenvector that is being followed

neghes The assumed number of negative eigenvalues of the Hessian at the Transition State. Should be 1: searches for higher-order transition states are not supported.

mode to follow Direction vector in atomic coordinates (Cartesian or Z-matrix, depending on the variable geocrd) that corresponds to the current estimate of the unique Hessian eigenvector with negative eigenvalue

Section LT

Information about the Linear Transit calculation

nr of points The total number of LT points to be computed.

current point Index of point that is currently computed

Energies Energy values in the LT points

Dipole Dipole moments in the LT points

Parameters LT parameters, initial and final values (along the path, the values are obtained by even-spaced linear interpolation)

atmcrd ZMAT if a Z-matrix structure (connection matrix) is available. CART otherwise. Used for printing

geocrd Type of coordinates to optimize and scan along the path (CART or ZMAT)

xyz Cartesian coordinates in the LT points (3,natoms,nlt)

zmatrix Internal coordinates in the LT points

AtomCharge Mulliken Mulliken atomic charges in the LT points

FragmentCharge Hirshfeld Hirshfeld fragment charges in the LT points

AtomCharge_initial Voronoi Voronoi atomic charges corresponding to the sum-of-fragment densities in the LT points

AtomCharge_SCF Voronoi Voronoi atomic charges corresponding to the SCF densities in the LT points

Section IRC

This section contains general information about the IRC (Intrinsic Reaction Coordinate) calculation. Details of the computed reaction path are in sections IRC_Forward and IRC_Backward.

atmcrd ZMAT is a Z-matrix structure (connection matrix) is available. CART otherwise

geocrd CART or ZMAT: the type of coordinates to change, optimize and trace

PointStatus A string status variable of the current IRC point. Value can be 'DONE' (if the point has been computed), 'EXEC' if its computation has not yet finished.

nfree Number of optimization coordinates that can be varied. See section GeoOpt

idfree (3,natoms) pointers to the optimization variables for each of the atomic coordinates. A zero means: frozen by constraint

xyz Cartesian coordinates

kmatrix Connection matrix, if a Z-matrix structure is available

zmatrix Internal coordinates

Energies Energy at the Transition State

Dipole Dipole moment at the Transition State

Gradients Computed energy gradients at the (assumed) Transition State (should be very small)

AtomCharge Mulliken Mulliken atomic charges, for the TS geometry

AtomSpinDen Mulliken Atomic spin densities (Mulliken) at the TS

AtomCharge_initial Voronoi Voronoi atomic charges at the TS, from the sum-of-fragments density

AtomCharge_SCF Voronoi Similar, for the SCF density

modtrc Defines the start direction for the IRC path. A positive value n selects the n-th eigenvector of the Hessian. A value -1 selects the gradient vector (which must then, of course, not be exactly zero). A value -2 specifies that the start direction is specified in the input file.

step Step length (in mass-weighted metric) between successive points of the IRC path.

stepMin The minimum value for the step

stepMax The maximum value for the step

Hessian inverted_ZMAT Inverse Hessian in internal coordinates.

lfree The number of independent optimization step directions (for the restricted optimization orthogonal to the IRC path.

vfree Direction vectors (3,natoms,lfree) for the independent optimization directions

GradientVector The current gradient vector (during optimization)

Section IRC_Forward

Information about the 'forward' IRC path. The choice, which direction down from the Transition State is forward or backward is arbitrary. By definition, in ADF the forward direction is in the positive direction along the first Hessian eigenvector, for which the sign convention is that the largest coefficient is positive.

PathStatus Status (string) variable for the 'forward' half of the IRC path. May be 'EXEC', or 'DONE', 'UNKNOWN', 'WAIT', 'OFF'

PointStatus Status variable for the current point at the 'forward' path. May be 'DONE', 'EXEC'

nset Size of arrays to store data in the IRC points along the path. Will be increased when too small

pivot Coordinates of the current pivot point

xyz Cartesian coordinates of the IRC points (3,natoms,nset)

zmatrix Internal coordinates of the IRC points (3,natoms,nset)

Path Lengths measures in mass-weighted metric along the path to the IRC points

Curvature Local curvature values of the path at the IRC points

Energies Energy values at the IRC points

Gradients Energy gradients at the IRC points (one value: the gradient along the path. The orthogonal components are presumably zero)

Dipole Dipole moments at the IRC points

AtomCharge Mulliken Mulliken atom charges at the IRC points

FragmentCharge Hishfeld Hirshfeld fragment charges at the IRC points

AtomCharge_initial Voronoi Voronoi atomic charges at the IRC points, corresponding to sum-of-fragments densities

AtomCharge_SCF Voronoi Voronoi atomic charges at the IRC points, corresponding to the SCF densities

CurrentPoint Integer index of the current IRC point (in the set of nset points)

step Current step length

Section IRC_Backward

All entries in this section match those in the section IRC_Forward. Of course, here they refer to the other half of the IRC path.

Section Freq

This section contains information about (progress) of the Frequencies calculation and results.

kountf An integer counter that keeps track of how many geometric displacements have been carried out to scan the potential energy surface around the equilibrium

nraman Integer to flag whether or not Raman intensities are computed

numdif Integer to determine the type of numerical differentiation (of gradients, to get the second derivative): 1=one-sided, 2=two-sided displacements.

disrad Size of displacements of Cartesian coordinates or bond lengths (in case of displacements in internal coordinates)

disang Size of displacements of angular coordinates

geocrd Type (string) of coordinates to displace: CART or ZMAT

atmcrd ZMAT if a z-matrix structure is available. This determines printed output but does not affect the computation. Compare the variable geocrd

nfree The number of degrees of freedom

idfree An array (3,natoms) that stores for each atomic coordinates (Cartesian or internal, depending on geocrd) the number of the (1..nfree) variational freedom it corresponds to. If zero, the coordinate is frozen by constraint.

xyz Cartesian coordinates of the equilibrium geometry

kmatrix Connection matrix that defines a Z-matrix

zmatrix The Z-matrix variable values of the equilibrium geometry

all freedoms Logical: true if all atomic coordinates are allowed to be displaced, not restricted by constraints.

nr of atoms The total number of atoms, including dummy's

rigids Vectors of rigid motion directions, expressed in the atomic coordinates (3,natoms,6)

Dipole previous The dipole moment of the previous geometry. This is used to compute dipole derivatives by numerical differentiation. The 'previous' geometry is the equilibrium geometry in case of one-sided displacements.

Dipole The dipole moment corresponding to the current geometry

Dipole derivatives The matrix of dipole derivatives with respect to atomic displacements

Polbty previous The polarizability tensor (6 elements, triangular representation) of the 'previous' geometry. See the remarks about the dipole moment

Polbty The polarizability tensor corresponding to the current geometry

Polbty derivatives The matrix of derivatives (w.r.t. the atomic coordinates) of the polarizability tensor

Gradients The energy gradients corresponding to the current geometry

Gradients previous The energy gradients of the 'previous' geometry. See the remarks about 'previous' dipole moment

Force constants The matrix of force constants (second derivatives), built up during the frequencies calculation.

xyz displaced The Cartesian coordinates of the current (displaced) geometry

zmatrix displaced Internal coordinates of the current (displaced) geometry

Dipole derivatives_CART Dipole derivatives with respect to Cartesian coordinate changes

Hessian_CART The Hessian matrix in Cartesian coordinates, computed at the end, when the construction of the 'Force constants' has been completed.

Frequencies An array with harmonic frequencies.

Sections Ftyp n

n is an integer. All such sections give general information about fragment type #n, and more specifically about the ADF calculation that produced the corresponding fragment file.

jobid Job identification of the fragment run

title Title of that calculation

---

nsym Number of symmetry representations (subspecies) used

norb For each representation the size of the Fock matrix (variational degrees of freedom)

bb Labels of the subspecies

igr (Partial) code for the point group symmetry

ngr (Partial) code for the point group symmetry

grouplabel Schönfliess symbol of the point group symmetry (of the fragment calculation)

nfcn An array over the representation: for each subspecies the number of primitive STO basis functions that partic-
ipate in that subspecies

jsyml An array (1:nsym). Value 1 means that the corresponding subspecies belongs to a 1D irrep. A value larger
than 1 means a correspondingly higher dimensionality of the irrep *and* indicates that that subspecies is the first
in that irrep. A value 0, finally, means that it is not the first subspecies in its irrep.

nfrag Number of fragments used in that fragment calculation

natom Number of atoms in the fragment

naos Number of primitive atomic basis functions

nrat 1 Maps the atoms of this fragment (the '1' signals the first fragment of this type) onto the list of all atoms

rotfrg Rotation matrix to map the fragment coordinates as they are on the fragment file onto their actual orientation
in the molecule

nsot Total number of MO degrees of freedom, summation over all subspecies

nmis The number of symmetry representations that could not be spanned by the basis set

mis Indices of the missing symmetry representations

Sections Ftyp n?

n stands for the n-th fragment type. The ? stands for one of the symmetry representations (of the point group symmetry
used in the fragment calculation)

froc MO occupation numbers for the MOs in this subspecies

eps Orbital energies When they result from a ZORA calculation, the non-scaled values are stored on file (the scaled
values are printed in the standard output file).

eigvf Fragment MO eigenvectors, expressed in all the primitive atomic orbitals of the fragment.

nsos 1 Total number of MOs in this subspecies: size of variational problem

nbas 1 Number of primitive atomic basis functions that participate in this subspecies

npart 1 Indices that give for each of the nbas functions, the number of the basis function in the list of all basis
functions

FO 1 The fragment MOs (nbas*nsos coefficients)

nocc 1 Number of non-empty orbitals

Section Freq Symmetry

Information about the true (possibly input-specified) symmetry of the equilibrium geometry (in a frequencies calcula-
tion). The displaced geometries may loose symmetry. Therefore, the program uses NOSYM symmetry, internally, for
a frequencies calculation. The 'true' symmetry of the system is used for analysis purposes.

nr of operators Number of symmetry operators used

operators (3,3) matrices of the operators

nr of symmetries Number of subspecies

symmetry labels Names of the subspecies

atom indices List of indices to map the symmetry-ordered atoms (loop over symmetry sets, loop over atoms in each set) to the 'normal' list of all atoms

nr of atomsets Number of sets of symmetry equivalent atoms

atom mappings Integer array that provides mapping (back and forth) between the atom list in the input file and the internally used list, which is atom type driven

atomset indices The number of atoms in each of the sets of symmetry equivalent atoms

nr of displacements_X (X must be one of the symmetry representations.) The number of symmetry-combined atomic displacements that transform as X

degeneracy_X Degeneracy of X

displace_X The actual displacement direction vectors (3,natoms,N). N is the number of symmetry displacements for X.

nr of rigids_X The number of rigid motion direction vectors that transform as symmetry representation X

displ_InputOrder_X The displacement vectors, but now expressed in the atomic coordinates using the ordering of atoms in the input file

NormalModes_X Harmonic frequency normal modes in representation X

Frequencies_X The harmonic frequency values

IR intensities_X The infrared intensities

Sections X

X stands here for the label of a subspecies of the point group symmetry, for instance A1. Depending on the point group symmetry, there may be many such sections, each corresponding to one of the subspecies. All such sections have an identical structure.

nmo_A The number of MOs with spin-A, for which the coefficient vectors are calculated. During the SCF this may be severely reduced, at the end it is usually the complete basis in the pertaining symmetry representation.

nmo_B Similar for spin b. This variable is not present in a restricted calculation.

SFO The definition of the SFOs in the representation, consisting of expansion coefficients in terms of the primitive atomic STO basis functions

frocf The occupation numbers of the SFOs in this representation

npart A list of indices of the bas functions that are used in this symmetry representation

froc_A The occupation numbers of the MOs in the representation, for spin-A

froc_B Similar for spin-B, if a spin-unrestricted calculation is performed

smx Overlap matrix between core functions and SFOs

frocor SFO occupation numbers

Orth-Bas The orthogonalized fragment orbitals in the BAS representation.

Low-Bas The Lowdin orbitals in the BAS representation: the matrix to transform the MOs from Lowdin representation (orthonormalized SFOs) to the BAS representation

Eigen_Bas_A mo expansion coefficients in the bas representation for all nmo_A orbitals. The coefficients run over all bas functions indicated by npart

Eigen_Bas_B Similar for spin-B, if present

eps_A The orbital energies for the nmo_A orbitals of spin-A When they result from a ZORA relativistic calculations, the non-scaled values are stored on file. (The scaled energies are printed in standard output.

eps_B Similar for spin-B, if present

Eig-CoreSFO_A MOs expressed in SFOs, for spin-A MOs

Eig-CoreSFO_B Similar for spin-B

Sections Atyp n X

Each such section contains the (core- and possibly also valence-) radial density and potential of one particular atom type. *X* is the atom type label and *n* is an index running over all atom types in the calculation. The list of all atom types is printed on standard output in the early geometry section.

The radial densities and potentials may be represented as simple tables - a sequence of values for *r*, the distance to the nucleus, and the corresponding density or potential - or as a piecewise expansion in Chebyshev polynomials over a sequence of intervals (r1,r2).

The core density and potential have been constructed from the Frozen Core orbitals, which are defined in the section Core. If a TAPE12 (corepotentials) file has been attached to the calculation the core data is read off from that TAPE12 and stored also.

rx val Maximum r-value for which the valence density is non-negligible

nrint val Number of intervals for piecewise expansion of the valence density in Chebyshev polynomials

rup val Arrays (1..nrint) of upper bounds of the intervals. The lower bound of the first interval is zero

ncheb val Array (1..nrint) with the number of expansion coefficients for each interval

ccheb val Coefficients of the expansion. All coefficients, for all intervals, are stored contiguously in one linear array. The parts pertaining to a particular interval are determined by using the arrays ncheb()

nrad Number of points used in the direct tabular representation of the atomic densities and potentials

rmin The first r-value of the table: the radial grid is defined by a first value (rmin), a constant multiplication factor defining rk+1 w.r.t. rk (rfac, see next), and the total nr of points (nrad).

rfac The multiplication factor of the radial grid

valence den The valence density, in a table of nrad values.

valence pot Similar for the Coulomb potential of the density, including a nuclear term Q/r, such that the long-range monopole term in the potential is zero

qval The number of electrons contained in the valence density

rx core Maximum r-value for which the core density is non-negligible

nrint core Number of intervals for piecewise expansion of the core density in Chebyshev polynomials

rup core Arrays (1..nrint) of upper bounds of the intervals. The lower bound of the first interval is zero

ncheb core Array (1..nrint) with the number of expansion coefficients for each interval

ccheb core Coefficients of the expansion. All coefficients, for all intervals, are stored contiguously in one linear array. The parts pertaining to a particular interval are determined by using the arrays ncheb()

qcore The number of electrons contained in the core density

core den The core density, in a table of nrad values.

core pot Similar for the Coulomb potential of the density, including a nuclear term Q/r, such that the long-range monopole term in the potential is zero

Section LqbasxLqfitx_xyznuc

This section will be removed again in the future. Temporarily it serves to transfer data from the calling program to the grid generator.

lqbasx An array with for each atom type the maximum angular moment quantum number in the basis functions for that type

lqfitx An array with for each atom type the maximum angular moment quantum number in the fit functions for that type

xyznuc Cartesian coordinates of the non-dummy atoms

Section GenptData

This section will be removed in the future. It serves, temporarily, to transfer data from the calling program to the numerical integration grid generator. Most of the entries here occur also in other sections but are packed together as replacement for previous common block structure.

numint Integer code for the type of integration grid. Usual value: 2 (polyhedra method)

iexcit Integer flag for excitations (response) calculation

lpolar Integer flag for polarizability (response) calculation

ldim Number of dimensions of periodicity

mdim Dimensionality of the molecule, for instance a linear molecule has mdim=1

r0mult A technical parameter that sets the radius outside which the multipole part of the fit coulomb potential functions is separated (from the exponentially decaying part), for separate treatment in the evaluation of the molecular coulomb potential.

avec (3,3) matrix with lattice vectors. Only the (ldim,ldim) sub matrix is significant.

bvec Inverse of avec (apart from a factor of 2 pi): lattice vectors in reciprocal space.

ngimax Maximum number of geometry optimization iterations

llbloc Block length determination parameter (maximum)

ipnbl Number of integration blocks processed by the current process

nbleqv The number of symmetry equivalent blocks to each symmetry unique block of points. This value is 1 if any equivalent blocks are not constructed and used.

ngmax The number of integration points, accumulated over all parallel processes

nblock The number of integration blocks

lblock The block length

lblx An upper bound of the block length applied during the computation of the block length

nmax The number of integration points generated by this process

twopi Value of the constant $2\,\pi$

fourpi Value of the constant $4\,\pi$

Section Multipole matrix elements

Information in a response calculation

dipole elements The matrix elements of the 3 dipole operator components between occupied and virtual orbitals: outer loop over the operators (in order: y, z, x), loop over virtual MOs, inner loop over occupied MOs

quadrupole elements Similar as for dipole. Order of operators: $\sqrt{3}$ *xy $\sqrt{3}$ *yz z2-(x2+y2)/2 $\sqrt{3}$ *xz $\sqrt{3}$ *(x2-y2)/2

octupole elements Similar as for dipole and quadrupole. Order of operators: $\sqrt{10}$ *y*(3*x2-y2)/4 $\sqrt{15}$ *xyz $\sqrt{6}$ *y*(4*z2-x2-y2)/4 z*(z2-3(x2-y2)/2) $\sqrt{6}$ *x*(4*z2-x2-y2)/4 $\sqrt{15}$ *z*(x2-y2)/2 $\sqrt{10}$ *x*(x2-3y2)/4

hexadecapole elements Similar as for dipole and quadrupole. Order of operators: $\sqrt{35}$ *xy*(x2-y2)/2 $\sqrt{70}$ *z*(3x2y-y3)/4 $\sqrt{5}$ *xy*(6z2-x2-y2)/2 $\sqrt{10}$ *(4yz3-3yz*(x2+y2))/4 (8z4-24*z2*(x2+y2)+3(x4+2x2y2+y4))/8 $\sqrt{10}$ *(4xz3-3xz*(x2+y2))/4 $\sqrt{5}$ *(x2-y2)*(6z2-x2-y2)/4 $\sqrt{70}$ *z*(x3-3xy2)/4 $\sqrt{35}$ (x4-6x2y2+y4)/8

Section Irreducible matrix elements

Information in a response calculation

irreducible dipole elements The dipole matrix elements between occupied and virtual MOs, as in the section Multipole matrix elements. Here, however, the matrix elements are ordered by symmetry representations and 'symmetry zeros' are omitted. The stored arrays, however, have the same size as in the previous section. See the implementation for details about the storage of this data. (Directory $ADFHOME/adf/response/)

irreducible quadrupole elements Similar as for the dipole elements

irreducible octupole elements Similar as for the dipole elements

irreducible hexadecapole elements Similar as for the dipole elements

Section ETS

Technical data used in the ets procedure.

nff Size of array ncspt (next)

ncspt Pointer array to find, for each atom type, the first element corresponding to that atom type's section in the arrays ncsett, alfcst, and cfcset, see below

ncs Size of the matrices ncsett, alfcst, and cfcset, see below

ncsett Build a list of products of core orbital expansion functions, taking only the one-center products and looping over the atom types (not the atoms). ncsett stores the powers of the radial variable r for the products (from the main quantum numbers, one subtracted). A product of a 1s and a 2p yields ncsett()=1

alfcst Similar as ncsett: the sum of the exponential decay factors of the factor functions

cfcset The density matrix corresponding build from the frozen core orbitals (all atom types, but no copies for the distinct atoms of a type), in the representation of the core orbital expansion functions. Stored are, per atom and per l-value (0..3) the upper-triangles of the corresponding density matrices, one after the other, all in cfcset

nnuc The number of (non-dummy) nuclei

qcore For each atom the number of electrons summed over its core orbitals, resulting from analytical integration of the core orbital expansions in STO core expansion functions.

Using Data from TAPE21

An ASCII dump of TAPE21 (complete or partial) can be obtained with the kf utility dmpkf, see the utilities document. Alternatively you may build your own small program to extract any required information, using the KF library routines in the ADF package. Consult the KFS documentation for a description of this software.

Representation of functions and frozen cores

adf uses the cartesian representation for the spherical harmonics part in functions:

f(x,y,z)=xaybzcrde-ar

The angular momentum quantum number l is then given by $l=a+b+c$, and the main quantum number $n = l +d+1$.

There are $(l + 1)(l + 2)/2$ different combinations of (a, b, c) for a given l - value, rather than $(2 l +1)$. The excess is caused by the presence of spurious non-l Functions in the set; a Cartesian d-set for instance consists of six functions, five of which are true d-functions while one linear combination is in fact an s-type function (x2+y2+z2). Only the five true d-combinations are actually used as degrees of freedom in the basis set, but lists of primitive basis functions (bas) for instance run over all Cartesian functions including the improper ones.

A function *set* in ADF is characterized by the quantum numbers l and n, and by the exponential decay factor a. A set thus represents $(l +1)(l +2)/2$ Cartesian functions and $(2 l+1)$ degrees of freedom.

The atomic frozen core orbitals are described as expansions in Slater-type functions; these are not the functions of the normal basis set but another set of functions, defined on the data files you use in Create mode.

Orthogonality of the valence space to the frozen core states is enforced as follows: for each frozen core shell (characterized by the quantum numbers l and n: all orbitals with $m = - l ... + l$ are identical apart from rotation in space) the set of valence basis functions is augmented with a so-called core orthogonalization function set. You may conceptually interpret the core orthogonalization functions as single zeta expansions of the true frozen core states. Each of the normal valence basis functions is now transformed into a linear combination of that valence function with all core orthogonalization functions, where the coefficients are uniquely defined by the requirement that the resulting function is orthogonal to all true core functions.

So the list of all Cartesian basis functions is much larger than the degree of freedom of the basis: it contains the spurious *non-l* combinations and it contains also the core orthogonalization functions.

Evaluation of the charge density and molecular orbitals

TAPE21 contains all the information you need to evaluate the charge density or a Molecular Orbital (MO) in any point in space. Most of the information is located in section Basis:

A list of function characteristics (kx,ky,kz,kr,alf), including the core orthogonalization functions. This list does *not* run over *all* bas functions used in the molecule: if a particular function is used on the atoms of a given atomtype, the function occurs only once in the list, but in the molecule it occurs as many times as there are atoms of that type.

With array nbptr you locate the subsections in the function list that correspond to the different types of atoms: for atom type i the functions nbptr(i)...nbptr(i+1)-1. The distinct atom types are listed in an early section of the standard output file.

Array nqptr gives the number of atoms for type i: nqptr(i+1)-nqptr(i). With this information you construct the complete list of all functions. Repeat the subsection of type i as many times as there are atoms of that type: the complete list can be considered to be constructed as a double loop, the outer being over the atom types, the inner over the atoms that belong to that type. The total 'overall' list of functions you obtain in this way contains naos functions. Note that in this way we have implicitly also defined a list of all atoms, where all atoms that belong to a particular atom type are contiguous. This list is the so-called 'internal' atom ordering, which may not be identical to the order in which atoms were specified in input, under atoms.

For a given symmetry representation (Sections S) the array npart gives the indices of the basis functions in the overall list that are used to describe orbitals in this representation. In case of an unrestricted run the array npart applies for either spin: the same basis functions are used; the expansion coefficients for the molecular orbitals are different of course.

In the symmetry-representation sections Eigen_bas gives the expansion coefficients that describe the MOs. The expansion refer to the functions indicated by npart, and the function characteristics are given by the arrays kx,ky,kz,kr,alf, *and* bnorm, i.e. the expansion is in* normalized*functions.

The value of an MO is now obtained as a summation of values of primitive basis functions. For the evaluation of any such basis function you have to take into account that its characteristics are defined in the local coordinate system of its atom.

To obtain the charge density you sum all MOs, squared and multiplied by the respective occupation numbers (array froc in the appropriate irrep section).

Note that the auxiliary program densf, which is provided with the ADF package, generates orbital and density values on a user-specified grid. See the utilities document.

13.6.2 TAPE13

TAPE13 is the checkpoint file for restarts after a crash. It is a concise version of TAPE21, containing only the data the program uses for restarting the calculation. See the restart keyword.

Like TAPE21, TAPE13 is a binary, keyword driven KF file. You can manipulate it with the KF utilities, to get a print-out of its 'table of contents', or a complete ASCII dump of its full contents.

The calculation that produces TAPE13 determines which section are written on it. The following sections may occur (and if they occur, the listed variables are stored in them). The actual values of the variables should be identical to the corresponding variables on TAPE21. Also they should have the same names and be located in the same sections. In some cases, TAPE13 contains the complete corresponding section of TAPE21.

Section Fit

coef_SCF SCF fit coefficients. Total number of them is nprimf, the number of primitive fit functions (counting all *Cartesian* spherical polynomials: 3 for a *p*-set, 6 for a *d*-set, and so on). If the calculation is spin-unrestricted, each spin has its own set of fit coefficients: first all coefficients of spin-A, then those of spin-B

coef_FreqCenter Only in a Frequencies calculation: the fit coefficients that correspond to the equilibrium geometry. The variable coef_SCF corresponds always to the current geometry, or the previous one if the geometry has just been changed and the new SCF has yet to start.

13.6.3 KF browser

With the GUI module kfbrowser one can browse through the raw data on KF files (like the .t21 result files).

```
$ADFBIN/kfbrowser file.t21
```

13.7 Error messages

ERROR: STOP GEOMETRY ITERATIONS
The SCF procedure did not converge, which caused the geometry optimization to be aborted. See WARNING: SCF NOT COMPLETELY CONVERGED below.

ERROR: imo is not occupied PT1W
ERROR: imo is occupied PT1W
The electronic structure is not auf-bau, that is the LUMO is lower in energy than the HOMO. Disable the KeepOrbitals option by setting it to a large number and try different SCF version until you get the SCF properly converged. See the SCF Troubleshooting section for details.

ERROR: More cycles needed. Geometry NOT CONVERGED
ERROR: GEOMETRY DID NOT CONVERGE
ERROR: GEOMETRY NOT CONVERGED

ERROR: TRANSIT FAILED

The geometry optimization did not converge in the allowed number of steps. See the Geometry Optimization Troubleshooting section for recommendations on how to proceed.

ERROR: TAPE12 NOT SUPPLIED with TAPE10

ERROR: Relativistic option is used but no file with core potentials was found

Before running a relativistic calculation you need to run the dirac program to create a TAPE12 file. This can be done automatically by adding a Basis block to the input and executing ADF as $ADFBIN/adf instead of adf.exe.

ERROR: calculating FDE interactions for the same fragment

It's a program bug. Send your input file to SCM

ERROR: Abort because one processor has no integration points

Repeat the calculation with fewer processors in the MPI job.

ERROR: Davidson method not converged

The Davidson method did not converge that is used in the calculation of excitation energies. Increase the number of ITERATIONS in the EXCITATIONS block key, or use a larger TOLERANCE.

ERROR: EMPTY FIELD FOUND AT END OF ...

ERROR: Cannot read ... even as a string

Check the QMMM block in the input. Every simple key in the block (as opposed to a block key) must be followed by a value. See the QMMM Manual for the correct syntax.

ERROR: LenRecUnits: reclen units not determined

This error is most likely caused by compiler problems or by incorrect compilation flags. See https://www.scm.com/support/downloads/platform-specific-information/ for supported compiler versions and contact SCM for an advice on compilation flags. Do not forget to include the $ADFHOME/buildinfo file in your request.

13.8 Warnings

WARNING: (slightly) inaccurate ETA integral(s)

WARNING: SUSPICIOUS ETA INTEGRALS

Eta's are symmetry adapted STO's. This error is usually caused by numerical stability problems in the symmetric matrix eigensolver. On the x86(-64) platforms you can try using a different version of the Intel MKL library (newer is not always better).

WARNING: 1-character identifier(s) in DEFINE

The Define input block does not allow single-character identifiers.

WARNING: activating keyword SDFTENERGY for the calculation of subsystem DFT energy.

For more than two frozen subsystems the sDFT energy needs the corresponding logical switch, which is normally activated via the SDFTENERGY key in the FDE input block.

WARNING: AMAT not far from singularity
WARNING: Bmatrix not far from singular
Amat and Bmatrix are used in COSMO solvation model. This warning could be printed when the distance between two COSMO surface points is shorter than 0.01 Bohr. You may want to try diferent COSMO accuracy parameters.

WARNING: Atom (...) exceeded the max allowed bonds in DimQMHybridization
The number of bonds per atom is exceeded in DimQM. Check your input geometry for correctness (coordinate units, etc.). The bond limit per atom type is currently hardcoded in the program and cannot be increased with recompiling the program.

WARNING: inaccurate integration of Core Density
WARNING: BAD CORE INTEGRAL ...
The core density must integrate to number of core electrons very accurately. If this is not the case then something is wrong either with the integration grid, with the input geometry or with the core orbitals. In any case, check your input geometry (Angstrom vs. Bohr vs. the Units input block, check bond lengths). Try using a smaller core and/or better grid.

WARNING: BAD FIT
WARNING: CANNOT NORMALIZE THE FIT
WARNING: LOSS OF CHARGE
These warnings are printed when the accuracy of the STO fit is not high enough. This might, for example, happen for negatively charged molecules for which the electron density cannot be adequately represented by the fit functions. This can be solved by using a larger fit type (e.g. FitType QZ4P in the Basis input block) or using the AddDiffuseFit keyword, or by using the ZlmFit instead of STOfit.

WARNING: The use of the SOLVATION option with non-basic fragments is not supported
WARNING: Bond energies are incorrect
Do not use the energy decomposition analysis if the fragments are calculated with COSMO solvation.

WARNING: C(lin) or D(lin) not supported in Freq runs
When performing numerical frequency calculation for a linear molecule you should not set the symmetry explicitly to C(lin) or D(lin). Use something like C(8v) or D(8h) instead.

WARNING: CD spectrum for triplex excit. NOT SUPPORTED
WARNING: CD spectrum for triplet excitations is NOT SUPPORTED
A CD spectrum calculation has been requested for triplet-only excitations, which is not supported by the program. The excitation rotary strengths will not be calculated.

WARNING: In ReadEpsilons: HOMO energy is positive
WARNING: Check if basis or fit sets are dependent

The HOMO energy is positive where it should usually be negative. The positive HOMO energy means the solution is not stable and the following TDDFT calculation may produce incorrect results.

WARNING: Combination of unrestricted and MODEL potential not extensively tested

Using the model XC functionals for spin-unrestricted calculations is not officially supported. Read the literature on the functional used to make sure you do not obtain nonsense results.

WARNING: Virtuals almost lin. dependent
WARNING: Consider using keyword DEPENDENCY

This warning is printed is when the smallest eigenvalue of the virtual SFO overlap matrix is smaller than 1e-5. You may want to add a Dependency keyword to your input.

WARNING: convMCD very small. Achieving convergence is unlikely

This warning is printed when the MCD convergence criterion is smaller than 1e-4. You may want to set the CONVMCD parameter in the MCD input block to a larger value if the CG solver does not converge.

WARNING: Core orb. en. with MODEL xc pot not implemented

Calculation of the core orbital energies in combination with model XC potentials (e.g. LB94) is not implemented.

WARNING: Could not find vecdimension excitations on SUBTAPE
WARNING: Trying to find vecdimension in ResponsePmats section ...

When performing a Subsystem DFT excitations calculation the exitations must be calculated for each sub-system (fragment).

WARNING: Density fitting may not be accurate enough!
WARNING: H12 and Electronic Coupling possibly INACCURATE
WARNING: Diagonal energies and overlap(s) should be accurate

These messages are printed together and already contain a hint. To avoid the problem you may want to increase the quality of the fit using one or more of the following options: FitType QZ4P, AddDiffuseFit, NumericalQuality {Good|VeryGood|Excellent}.

WARNING: diagonalization error in dvdrvrhf. Results will be incomplete.
WARNING: diagonalization error in dvdrvrhfso. Results will be incomplete.

The LAPACK subroutine dsygv (dzhegv in case of dvdrvrhfso) returned a non-zero status for the (A-B)(A+B) matrix. Analysis of the obtained solution will not be done.

WARNING: dihedral jumps 180 degrees

This warning is printed by the old optimizer branch. Check the optimization results and swtich to the new branch if you encounter any problems.

WARNING: DIM dipoles not converged upon SCF convergence

Use more tight SCF convergence criteria.

WARNING: No default parameters for classical Hirshfeld partioning
WARNING: Dispersion added with factor 1.0

You may want to set the steepness factor ATT0 and the scaling factor BTT0 for the SCBR empirical dispersion.

WARNING: doing final FDE cycle - FDE NOT converged

The FDE freeze-and-thaw procedure has not converged. You may want to increase the number of the FDE freeze-and-thaw cycles.

WARNING: DPERTURB calculation, no CI-type couplings calculated

Since the CI-coupling does not make much sense in combination with DPERTURB, the former is disabled.

WARNING: ERROR OPENING ONE OF THE NBO DATA FILES. SEE OUTPUT

Check the output to see which NBO file is missing and look for clues in the NBO output preceeding the calculation.

WARNING: Eucledian distance between NEB images is too short. NEB may have problems

The total Eucledian distance between two NEB images should be large compared to the Cartesian optimization step size. Otherwise the NEB optimization may result in a zig-zag path and may eventually fail. One way to increase the distance is by reducing the number of images.

WARNING: Extra DFT term related to g_xc suspiciously large

This warning is printed when the abs(beta(diff)) value is larger than 1e-3 and is larger than abs(beta(total)) in a hyperpolarizability calculation. This may be an indication of a numerical instability.

WARNING: FDE in combination with ZlmFit not thoroughly tested. Consider using STOFIT

[[[Mirko???]]]

WARNING: FILE POSSIBLY TOO BIG IN AORUN

This warning is printed when the AORUN input keyword is specified in certain types of TDDFT calculations (e.g. polarizability). In such a case the writing of the orbital values on the grid is disabled because otherwise it would result in a huge TAPE10 file. Note: the AORUN keyword serves no other purpose but to disable writing of the orbitals to TAPE10, so it should be used with care.

WARNING: Geometry optimization did not converge - frequencies are not calculated

This warning is printed when ADF is instructed to compute the vibrational spectrum of a moleule after geometry optimization but the optimization did not converge. In this case the frequencies are not computed.

WARNING: GRADIENT ILL-DEFINED FOR BOND ANGLE 180 OR 0

This warning is generated by the old geometry optimization branch when a valence angle gets close to 0 or 180 degrees. Such angles cannot be treated properly by the optimizer so one should avoid them, for example by adding dummy atoms.

WARNING: Hessian is incomplete

This warning just informs the user that only a fraction of the Hessian matrix has been done, as specified by the NUC keyword of the AnalyticalFreq input block. The missing matrix elements are calculated from an empirical MM forcefield and the combined Hessian matrix is analyzed.

WARNING: inaccurate elimination of XX motions, recover

This warning is generated by the old geometry optimization branch. Add PRINT FREEMV to your input to get more information.

WARNING: Inconsistent relativistic options used

This warning is printed when inconsistent relativistic options are used for the fragments and the final calculation and the RELOPTINCONS keyword is used to prevent ADF from stopping the calculation. If you add RELOPTINCONS then you probably know what you're doing.

WARNING: Infrared Intensities are wrong
WARNING: Infrared Intensities wrong

These warnings are issued when the computed dipole moment derivative is larger than 0.1 a.u. for a normal modes for which it must be zero by symmetry.

WARNING: Inter-node communication is slow! You may want to run on fewer nodes (or even one).
WARNING: Intra-node communication is VERY slow! Check CPU affinity, hyperthreading and over-subscription

The program tests the latency of the MPI communication at the beginning of each parallel calculation. The overall (i.e. the whole MPI "world" including all processes from all nodes) and the intra-node (i.e. including only the processes within one node) latency are measured separately. The first warning is printed if the overall latency is higher than 1 ms. This usually occurs when running on multiple nodes with a slow interconnect (e.g. 1 Gbps Ethernet), or when running on too many nodes. In this case it is recommended to run on fewer nodes. It is **not** recommended to run multi-node jobs at all if the cluster does not have a fast interconnect. The second warning is printed when the intra-node latency is higher than 1 ms, which should **never** happen. If the intra-node communication is slow the user is advised to stop running further MPI calculations until the issue is resolved. Possible reasons for this are: incorrect CPU affinity settings enforced by the batch scheduler (e.g. more than one process is bound to the same core), host is oversubscribed (i.e. other jobs are running on the same machines and ADF has to compete with them for resources). When two or more MPI jobs have to compete for the same resources both jobs will a low performance because the MPI synchronization calls also consume CPU cycles.

WARNING: INVALID SPRING PARAMETER. DEFAULTS ARE ENFORCED.

The valid values for the first NEBSPRING parameter are 1 through 6. If a value outside this range is specified it is set to 1 and this warning is printed.

WARNING: IRC START REQUEST IMPOSSIBLE: ADJUSTED

This warning is printed when the IRC eigenvector index specified by the IRC HESS parameter is larger than the number of free coordinates.

WARNING: KeepOrbitals on

This warning is printed when the MO populations become non-aufbau due to the KeepOrbitals feature. The KeepOrbitals usually kicks in after N/2 SCF iterations, where N is the max number of SCF iterations allowed. See the Occupations keyword for details.

WARNING: LB94 XC FUNCTIONAL USED FOR ENERGY GRADIENTS

The LB94 XC functional is supposed to be used to calculate spectroscopic properties such as optical excitation spectra and is not intended for the bonding energy (and gradients thereof) calcualtions.

WARNING: LMO beta analysis has nonzero B-field frequency. Check output.

This warning is printed when the magnetic perturbation field in the dynamic beta tensor (optical rotation) calculation has a finite frequency. Use the DEBUG keyword of the LOCORB input block to get more information.

WARNING: localized MOs: unknown symmetry key ...

This warning is printed when there is an unrecognized keyword is found in the LOCORB input block. Check that your input conforms the documentation of the LOCORB input block.

WARNING: Low value for "numvec" detected in AllocDavidson

WARNING: Using numvec = [number of excitations] + 5 instead

WARNING: Specify larger value for VECTORS in EXCITATION input block

These warnings are printed together and are self-explanatory.

WARNING: GIAOS ARE NOT IMPLEMENTED WITH LIFETIME. SWITCHING OFF GIAOS

WARNING: MAGOPTROT AND QUADRUPOLE OPTIONS ARE MUTUALLY EXCLUSIVE. SWITCHING OFF QUADRUPOLE

WARNING: MAGPERT AND QUADRUPOLE OPTIONS ARE MUTUALLY EXCLUSIVE. SWITCHING OFF QUADRUPOLE

WARNING: MAGPERT AND VROA OPTIONS ARE MUTUALLY EXCLUSIVE. SWITCHING OFF MAGPERT

WARNING: OPTICALROT AND VROA OPTIONS ARE MUTUALLY EXCLUSIVE. SWITCHING OFF OPTICALROT

WARNING: OPTROT AND GIAO AND NBO OPTIONS WITHOUT MAGPERT ARE NOT IMPLEMENTED. SWITCHING OFF NBO

WARNING: OPTROT AND QUADPERT OPTIONS ARE MUTUALLY EXCLUSIVE. SWITCHING OFF QUADPERT

WARNING: VROA OPTION MUST BE USED WITH QUADRUPOLE. SWITCHING ON QUADRUPOLE

WARNING: VELOCITYORD DOES NOT WORK WITH GIAO. SWITCHING OFF GIAO

WARNING: ZLMFIT AND FITAODERIV ARE NOT IMPLEMENTED. SWITCHING OFF FITAODERIV

These warnings are results of the AOResponse internal checks.

WARNING: max no. of iterations reached

This warning is printed when the CPKS procedure did not converge in the given number of steps. This may indicate a numerical stability issue or some other problem. In any case, you should check if the SCF has converged, if the electronic configuration is correct. Increasing the numerical accuracy might also help. Finally, you can try increasing the number of CPKS cycles for the feature in question (TD-DFT, analytical frequencies, etc.).

WARNING: maxBlockSize does not divide (maxBasisSize - minRestartSize). Suboptimal performance expected!

This warning is printed by the Davidson method using the PRIMME library. The fact that (maxBasisSize - minRestartSize) is not divisible by maxBlockSize does not affect the results so there is no reason to worry. [[[Robert

is there any way to influence this?]]]

WARNING: MIXED OCCUPATIONS in locmo (localized MO)

This warning is printed when occupation numbers of the localized MOs after 2x2 rotations do not match the original ones, which essentially means that the occupied and virtual orbitals have been mixed together.

WARNING: mo levels nearly degenerate

This warning is usually issued by the CPKS driver routine when the energy difference between an occupied and a virtual orbital is less than 1e-3 a.u.. This usually means that the HOMO-LUMO gap is very small which may lead to problems in the CPKS procedure.

WARNING: MODTRC refers to non-existing free variable
WARNING: ... selecting the first mode instead

This warning is printed by the old optimization branch in response to an invalid setting for the transition state search vector. In this case the TS procedure will use the eigenvector corresponding to the lowest eigenvalue of the Hessian matrix.

WARNING: More iterations needed. MODERATELY CONVERGED
WARNING: More iterations needed. NOT CONVERGED
WARNING: SCF did not converge - frequencies are not calculated
WARNING: SCF NOT COMPLETELY CONVERGED

One of these warnings is issued when the SCF procedure did not (completely) converge. See the SCF Troubleshooting section for recommendation on how to improve the SCF convergence.

WARNING: More nodes than energies on the contour: this is inefficient

This warning is printed by the NEGF code when the number of energies on the contour is smaller than the number of MPI tasks. This means that you can probably perform the calculation in the same time using fewer processors.

WARNING: multiple small orbital energy differences found

This warning is printed when the undocumented ASYMPCOR keyword is specified and more than one occupied-virtual orbital pair is found for which the energy difference is less than 1e-3 a.u..

WARNING: Negative root found. AllocDavidson
WARNING: Negative root found. AllocDavidsonso

These warnings are issued when the Davidson procedure finds a negative eigenvalue. This usually indicates a triplet instability.

WARNING: negative sqrt in the CI-type coupling parameters
WARNING: the results might be not trustworthy!!!!!

This warning is printed by the subsystem TDDFT module when checking requirements for the CI coupling calculation. The latter will not be done if any element of the CI_omega matrix is imaginary.

WARNING: no convergence in svdcmp

This is a generic warning from a singular value decomposition routine. When this occurs the SVD procedure will be aborted, which may lead to incorrect results in the module that called the routine.

WARNING: No DIM points found for projection matrix.

This warning is printed when projecting rigid rotations/translations from the DIM/QM energy gradients in the case there are no DIM points found within the cutoff radius specified by the PROJECTIONMATRIXPOINTS CUTOFF keyword.

WARNING: no optimization done in DVDSON: using guess vector results

This warning is printed to remind the user that he/she has specified SFGUESS or TRUSTSFGUESS in the input.

WARNING: root must be bracketed in rtsafe

The rtsafe function finds a root of the given callback-function in the given interval. The requirement is that the function has the opposite signs at the ends of the interval. This warning is printed when the requirement is not fulfilled.

WARNING: rtsafe exceeding maximum iterations

This warning is printed when the rtsafe function fails to find a root within the allowed number of iterations, currently 100.

WARNING: Non Aufbau energies used

This warning is printed by the Response code (excitations, polarizabilities, etc.) when the HOMO lies higher than LUMO in energy or when these are degenerate.

WARNING: Non-standard use of LAP functional
WARNING: Non-standard use of LYP functional

The LAP and LYP functionals already contain a term for correlation at the LDA level. This warning is printed when an LDA functional is specified in addition to LAP or LYP.

WARNING: NONRELATIVISTIC RUN WITH RELATIVISTIC COREPOTENTIALS
WARNING: RELATIVISTIC RUN WITHOUT RELATIVISTIC COREPOTENTIALS

This warning is printed when the CorePotentials keyword is used (or not used) inconsistently with the Relativistic option. Before running a relativistic calculation you need to run the dirac program to create a TAPE12 file, which then needs to be specified in the CorePotentials keyword. This can be done automatically by adding a Basis block to the input and executing ADF as $ADFBIN/adf instead of adf.exe.

WARNING: not all AIM bond paths converged

This warning is printed by the AIM bond path search module when some of the paths failed to converge to an attractor (a nucleus).

WARNING: not all scratch files were closed

This warning probably indicates a programmer error. Send the output to support@scm.com.

WARNING: not enough initial estimates
WARNING: Too many initial estimates.?
One of these warnings is issued when the number of initial vectors in the Davidson procedure is either too high or too low.

WARNING: Number of states in energy window changed
This warning is printed when the number of molecular orbitals in the energy window given with the ELTHRES/EUTHRES sub-keys of the VIBRON input block changes during calculation.

WARNING: occupations may be NOT AUFBAU
This warning is printed during a single-point calculation or the last iteration of a geometry optimization if the HOMO has a higher energy than the LUMO. For fractionally occupied orbitals, the HOMO is the highest MO that is populated by at least 1e-6 electrons and the LUMO is the lowest MO that misses at least 1e-6 electron to be fully occupied.

WARNING: omega and d(omega) incompatible in Verdet run
This warning is printed during calculation of the Verdet's constant when the value of the frequency ω is lower than the half-width of the interval used for numerical differentiation $\Delta\omega$.

WARNING: Out of parameterized Coulomb radii limits
The Coulomb radius parameters used to determine the COSMO sphere radii are currently available for elements up to 105. For heavier elements this warning is printed and the calculation stops.

WARNING: partially occupied orbitals
This warning is printed at the end of the SCF process if it results in partially occupied orbitals. The user should examine the resulting electronic structure carefully to see if this is the result they expected. One way to avoid fractional occupations is to repeat the calculation as spin-unrestricted with a non-zero total spin.

WARNING: Patchkovskii routines for PBE used. Check output
This warning is printed when USESPCODE is specified in the input together with the PBE functional. The bonding energy and other fragment analysis data may be incorrect.

WARNING: potential problem with dependency in AOResponse. Check output
WARNING: potential problem with dependency in LOCMO. Check output
When using the Dependency keyword, the linearly-dependent orbitals are not removed completely but are effectively excluded from variation by setting the corresponding diagonal matrix element to a large positive value. One of these warnings is issued when the any of the molecular orbitals included in the variation have the energy close to that large value, which probably means there is some bug in the SCF bookkeeping.

WARNING: Pressure must be positive, adjusted
This warning is printed when the pressure specified in the Thermo keyword is smaller than 1e-4 atmosphere.

WARNING: PRIMITIVE FIT COEFFICIENTS A BIT LARGE

WARNING: PRIMITIVE FIT COEFFICIENTS LARGE
WARNING: PRIMITIVE FIT COEFFICIENTS TOO LARGE
One or more of these warnings are printed when the number of electrons due to a single STO fit-function exceeds certain limits (20, 200, and 2000 times the largest number of valence electrons on a single atom in the molecule, respectively). This may indicate a linear dependence (over-fitting) problem in the fit or the basis set.

WARNING: problem with NEB optimisation. Re-initializing Hessian
This warning is printed during an NEB calculation if the Eucledian angle between three images becomes less than 30 degrees. This may indicate that there are too many NEB images for such a short reaction path.

WARNING: ReadExSym: found large energy differences
This warning is printed in a Vibron calculation when the excitation energy difference between two geometries is larger than 0.2 eV. You should check the results carefully.

WARNING: Restricted Open-Shell calculation, Unrestricted setup may be needed
This warning is printed when a molecule with an odd number of electrons (a radical) is calculated spin-restricted. This often indicates an error in the input (either wrong geometry or a missing Charge keyword). This warning may safely be ignored if the result of this calculation is to be used as a fragment.

WARNING: SchmidtOrth keeps finding noise
This warning is printed by the Schmidt orthogonalization procedure when it fails to orthogonalize the input vector to the reference set within 4 attempts. Two normalized vectors are considered orthogonal when their dot product is smaller than 1e-8 in the absolute value.

WARNING: sDFT energies should be used together with the FULLGRID option! Results might not be reliable!
Enable the FULLGRID option.

WARNING: SLOW SCF
WARNING: ... apply ElectronSmearing
In case of really problematic SCF a finite electron temperature (smearing of electrons around the Fermi level) is applied automatically if the user a) has not specified smearing via input and b) has not defined the occupation numbers explicitly. Furthermore, smearing is not automatically turned on in single-point calculations and in the final geometry. That is, it is only allowed during geometry optimization in the intermediate geometries. To disable automatic smearing specify Occupations Smearq=0 in the input.

WARNING: small delta epsilon found
This warning is printed when the energy difference in an occupied-virtual MO pair is smaller than 0.001 a.u., which means the HOMO and LUMO are nearly degenerate.

WARNING: SOLVATION block: DISC is not compatible with C-Mat POT
This warning is printed when the DISC option in the Solvation input block is set for the C-Matrix POT (which is the default), where it does not make sense. The user should check their input and either remove the DISC option or set C-Matrix to EXACT or FIT.

WARNING: SOMCD indicated but closed-shell calculation

The SOMCD feature (used to calculate the C terms of the magnetic circular dichroism spectra) is available only in spin-unrestricted calculation. If SOMCD is specified in a spin-restricted calculation then it is ignored and this warning is printed.

WARNING: SSB-D XC FUNCTIONAL USED FOR ENERGY GRADIENTS

The use of "GGA SSB-D" in a geometry optimization may lead to numerical issues for some systems. Check carefully that the energy goes down smoothly and continuously. Also compare with a calculation performed with the "METAGGA SSB-D" specified in the XC input block.

WARNING: Strange column number EXCITATIONS

The second (undocumented) argument of the irrep sub-keys of the Davidson and Exact sub-blocks can optionally specify that the 2nd or 3rd projection of a multi-dimensional irrep should be used to calculate intensities. This warning is printed when the number specified is larger that the dimensionality of the irrep.

WARNING: Suspicious delocalized coordinates. Change GEigenvalueThreshold or optimize in Cartesian coords

This warning is printed when the number of generated delocalized coordinates is not equal to 3N-6 for a system of 5 atoms and more. This may indicate some problems in generation of internal coordinates for the input geometry, for example because of isolated molecules (which should normally not be a problem). This warning may also be caused by the presense of a block constraint, in which case it may be safely ignored. Otherwise, one should check the number of the generated delocalized coordinates in the output file. If it is larger than 3N-6 then it is advised to look at the "G-matrix eigenvalues" table in the output and set the GEigenvalueThreshold parameter in the Geometry block to an average of the eigenvalues number 3N-6 and 3N-5.

WARNING: Symmetry is not yet supported in VCD

The vibrational circular dichroism (VCD) intensities are not calculated if the syemmetry used in the calculation is other than C:sub:1 (i.e. NOSYM)

WARNING: TDA in FDEc also requires TDA excitations as input

Use TDA consistently in coupled frozen density embedding (FDEc).

WARNING: Temperature must be positive, adjusted

The temperature value specified in the Thermo keyword must be larger than 0.1 K.

WARNING: The full TDDFT kernel icw symmetry may give incorrect results.
WARNING: Results can be checked with symmetry NOSYM

As this warning already suggest, you can check the results with SYMMETRY NOSYM.

WARNING: thin plane error. HULL33
WARNING: thin planes inconsist 1. HULL33

These warnings can appear when the Te Velde integration grid is used. Use the Becke grid in this case.

WARNING: Unknown GGA funtional for UFF dispersion.'

The UFF dispersion parameters depend on the XC functional used. Currently the parameters are defined for the PBE, B3LYP, PW91 and B3PW91 functionals. For any other functional the PBE parameters are used and this warning is printed.

WARNING: Unknown GGA functional for MBD dispersion.'

The MBD dispersion parameters depend on the XC functional used. Currently the parameters are defined for the PBE, PBE0, HSE03 and HSE06 functionals. For any other functional no short-range damping is applied and this warning is printed.

WARNING: Unknown GGA used. Dispersion added with factor 1.0

The Grimme dispersion corrections are defined per XC functional. For any unknown functional the dispersion is, by default, added with a factor 1.0, which can be changed from input.

WARNING: Unknown global optimization type requested. Using Quasi-Newton
WARNING: Unknown NEB optimization type specified. Using default
WARNING: Unknown optimization type requested. Using Quasi-Newton

One of these warnings is printed when the NEBOPT sub-key of the Geometry block contains an invalid argument.

WARNING: Unknown XC functional for DFT-D3

This warning is printed when an unknown XC functional is specified together with "Dispersion Grimme3" keyword in the XC input block. In such a case the PBE parameters are used for the dispersion correction.

WARNING: Untested MBD dispersion parameters are used. Check results.
WARNING: Untested dispersion parameters are used. Check results.

This warning is printed when the system under stady contains atoms with an element index greater than the maximum defined for the given correction. Currently, the limits are set as follows: 54 for Grimme, 86 for MBD, 94 for Grimme3, 102 for sCBR, and 103 for the UFF dispersion type.

WARNING: Unrecognized key for Irrep

This warning is printed when an unrecognized irreducible representation name is given in the Davidson or Exact sub-blocks of the Excitations block.

WARNING: use of multiple fragments not yet allowed !!!
WARNING: Use results from parallel runs with caution!

These self-explanatory informative warnings are printed by the (relatively new) SubsystemResponse module.

WARNING: USING >ADIABS< scheme in vibron.
WARNING: DO YOU REALLY KNOW WHAT YOU ARE DOING?

The ADIABS scheme in the Vibron module can only be used for totally symmetric modes. Add ONLYSYM to the Vibron block key.

WARNING: using LDA-only response. Please check your output

This warning is printed by the AOResponse code in cases when the use of the GGA is disabled automatically, usually in open-shell calculations.

WARNING: VERY HIGH VALUE FOR FRANGE IN RANG2
WARNING: VERY HIGH VALUE FOR QPNEAR IN RANGQP

This warning is printed by the grid generation routine when a basis or an STO fit function has a very long range (>10,000 a.u.). This may occur for an extremely diffuse function when a very high integration accuracy is requested.

WARNING: VERY SMALL A(0) IN LOCORB1. Check output!

This warning is printed by the routines performing the Pipek-Mezey (PM) and Boys localization on first-order perturbed molecular orbitals when the absolute value of the A(0) denominator becomes smaller than 1e-10.

WARNING: VROA OPTION MUST BE USED WITH QUADRUPOLE

The AOResponse Quadrupole options is switched on and this warning is printed when the vibrational Raman optical activity (VROA) is to be calculated.

WARNING: WRONG COORDINATES

This warning is printed by QUILD in the same situations where ADF exits with the same "WRONG COORDINATES" message: the input geometry does not satisfy the point group specified by the Symmetry keyword. This usually happens when the molecule's orientation does not match the symmetry operations. Note that when the point group is set from input, the program does not re-orient the input geometry to match the symmetry.

WARNING: wrong nr of functions deleted. ORTNRM

Internal ADF error message. Related to STO fit. Send your input file to SCM. Alternative is to use ZlmFit.

WARNING: zero number of filtered transitions found, try larger demax

This warning is printed by the approximate TD-DFT module (TD-DFTB, sTDDFT, sTDA, KSSPECTRUM) when there are no transitions found for the given energy range. This should not happen if there is at least one occupied and one virtual orbital present because in ADF the range is by default very large (~1e308 a.u.)

EXAMPLES

14.1 Introduction

The ADF package contains a series of sample runs. They consists of UNIX scripts (to run the calculations) and the resulting output files.

The examples serve:

- To demonstrate how to do calculations. The number of options available in ADF is substantial and the sample runs do not cover all of them. They should be sufficient, however, to get a feeling for how to explore the possibilities.

- To work out special applications that do not fit well in the User's Guide.

Note: Most of the provided samples have been devised to be short and simple, at the expense of physical or chemical relevance and precision or general quality of results. They serve primarily to illustrate the use of input, necessary files, and type of results. The descriptions have been kept brief. Extensive information about using keywords in input and their implications is given in the User's Manual.

Where references are made to the operating system (OS) and to the file system on your computer, the terminology of a UNIX type OS is used.

All sample files are stored in subdirectories under `$ADFHOME/examples/`, where `$ADFHOME` is the main directory of the ADF package. There are many subdirectories in `$ADFHOME/examples/`: the examples presented in this section are located in `$ADFHOME/examples/adf/`. Each sample run has its own directory. For instance, `$ADFHOME/examples/adf/HCN/` contains an ADF calculation on the HCN molecule.

Each sample subdirectory contains:

- A file TestName.run: the UNIX script to execute the calculation(s). A sample may involve several calculations, for instance a few CREATE runs (with ADF), then a molecular calculation (also ADF), and finally a NMR calculation (with the NMR program) to compute chemical shifts.

- A file TestName_orig.out.gz: the resulting output(s) against which you can compare the outcome of your own calculation. Note: the files are compressed using gzip (https://en.wikipedia.org/wiki/Gzip).

- Zero or more files with a .adf extension. These files, if present, are intended for ADFinput and demonstrate the same functionality as the two files above. However, there are also differences between the .adf and the Test-Name.run files so the results obtained with the .adf files **cannot be compared directly** with TestName_orig.out. Also, the TestName.run file usually contains more than one calculation, for which more than one .adf file is required. That's why in some directories you may find more than one .adf file.

Technical notes:

- Running the examples on Windows: You can run an example calculation by double-clicking on the appropriate .run file. After the calculation has finished, you can compare the TestName.out file with the reference TestName_orig.out file. See remarks about comparing output files below.

- The UNIX scripts make use of the *rm* (remove) command. Some UNIX users may have aliased the *rm* command. They should accordingly adapt these commands in the sample scripts so as to make sure that the scripts will remove the files. New users may get stuck initially because of files that are lingering around after an earlier attempt to run one of the examples. In a subsequent run, when the program tries to open a similar (temporary or result) file again, an error may occur if such a file already exists. Always make sure that no files are left in the run-directory except those that are required specifically.

- It is a good idea to run each example in a separate directory that contains no other important files.

- The run-scripts use the environment variables `ADFBIN` and `ADFRESOURCES`. They stand respectively for the directory that contains the program executables and the main directory of the basis set files. To use the scripts as they are you must have defined the variables `ADFBIN` and `ADFRESOURCES` in your environment. If a parallel (PVM or MPI) version has been installed, it is preferable to have also the environment variable `NSCM`. This defines the default number of parallel processes that the program will try to use. Consult the Installation Manual for details.

- As you will note the sample run scripts refer to the programs by names like 'adf', 'nmr', and so on. When you inspect your `$ADFBIN` directory, however, you may find that the program executables have names 'adf.exe', 'nmr.exe'. There are also files in `$ADFBIN` with names 'adf', 'nmr', but these are in fact scripts to execute the binaries. We strongly recommend that you use these scripts in your calculations, in particular when running parallel jobs: the scripts take care of some aspects that you have to do otherwise yourself in each calculation.

- You need a license file to run any calculations successfully. If you have troubles with your license file, consult the Installation manual. If that doesn't help contact us at support@scm.com

When you compare your own results with the sample outputs, you should check in particular (as far as applicable):

- Occupation numbers and energies of the one-electron orbitals;

- The optimized geometry;

- Vibrational frequencies;

- The bonding energy and the various terms in which it has been decomposed;

- The dipole moment;

- The logfile. At the end of a calculation the logfile is automatically appended (by the program itself) to the standard output.

General remarks about comparisons:

- For technical reasons, the discussion of which is beyond the scope of this document, differences between results obtained on different machines (or with different numbers of parallel processes) may be much larger than you would expect. They may significantly exceed the machine precision. What you should check is that they fall well (by at least an order of magnitude) within the *numerical integration* precision used in the calculation.

- For similar reasons the orientation of the molecule used by the program may be different on different machines, even when the same input is supplied. In such cases the different orientations should be related and only differ in some trivial way, such as by a simple rotation of all coordinates by 90 degrees around the z-axis. When in doubt, contact an ADF representative.

14.2 Model Hamiltonians

14.2.1 Special exchange-correlation functionals

Example: Asymptotically correct XC potentials: CO

```
Download CO_model.run
```

```sh
#!/bin/sh

# For property calculations, XC potentials with asymptotically correct (-1/r)
# behavior outside the molecule, the results tend to be superior to regular LDA
# or GGA calculations. This is especially true for small molecules and for
# properties that depend heavily on the proper description of the outer region
# of the molecule. In the example, all-electron basis sets are used. This is
# mandatory for the SAOP potential.

# In the first example, excitation energies are calculated with the GRACLB
# potential. This potential requires one number as argument: the experimental
# ionization potential in atomic units. This number can be either based on an
# experimental value, or on previous GGA total energy calculations.


$ADFBIN/adf <<EOR
title CO excitations grac potential

BeckeGrid
 Quality good
End

XC
 Model GRACLB 0.515
End

Atoms
O  0              0   0
C  1.128205364   0   0
end

Excitations
 Lowest 10
 Onlysing
End

Basis
 Type TZ2P
 Core None
End

EOR

rm TAPE21 logfile



# The same calculation with the SAOP XC potential would differ in the XC block
# only:


$ADFBIN/adf <<EOR
title CO excitations SAOP

BeckeGrid
 Quality good
End

XC
```

```
 Model SAOP
End

Atoms
O  0              0   0
C  1.128205364  0   0
end

Excitations
 Lowest 10
 Onlysing
End

Basis
 Type TZ2P
 Core None
End

EOR

mv TAPE21 CO.t21


# SAOP depends on the orbitals which makes it more expensive to evaluate than
# GRAC for large molecules, but is much easier to use, since it does not require
# an ionization potential parameter as input.
```

Example: Meta-GGA energy functionals: OH

Download OH_MetaGGA.run

```
#! /bin/sh

$ADFBIN/adf <<eor
TITLE  OH radical, C*v symm. MP2(full)/6-31G* Geom. MP2=-75.5232063            ␣
↪

ATOMS
O    0.000000    0.000000    0.108786
H    0.000000    0.000000   -0.870284
END

SYMMETRY C(2V)

IrrepOCCUPATIONS
A1 3 // 3
A2 0 // 0
B1 1 // 1
B2 1 // 0
END

UNRESTRICTED

DEPENDENCY bas=1.0e-4
NOPRINT BAS FIT FUNCTIONS SFO
METAGGA
```

```
XC
 MetaHybrid TPSSh
END

CHARGE 0 1

NumericalQuality VeryGood

Basis
 Type ET/ET-QZ3P
END

eor

rm TAPE21 logfile

$ADFBIN/adf <<eor
TITLE  OH radical, C*v symm. MP2(full)/6-31G* Geom. MP2=-75.5232063                ␣
↪

ATOMS
O    0.000000    0.000000    0.108786
H    0.000000    0.000000   -0.870284
END

SYMMETRY C(2V)

IrrepOCCUPATIONS
A1 3 // 3
A2 0 // 0
B1 1 // 1
B2 1 // 0
END

UNRESTRICTED

DEPENDENCY bas=1.0e-4
NOPRINT BAS FIT FUNCTIONS SFO
METAGGA

XC
 METAGGA SCF TPSS
END
HARTREEFOCK

CHARGE 0 1

NumericalQuality VeryGood

Basis
 Type ET/ET-QZ3P
END

eor

rm TAPE21 logfile

$ADFBIN/adf -n1 <<eor
```

```
create H file=$ADFRESOURCES/ET/ET-QZ3P/H
XC
GGA PBE
END
HARTREEFOCK
eor

mv TAPE21 t21.H

$ADFBIN/adf -n1 <<eor
create O file=$ADFRESOURCES/ET/ET-QZ3P/O
XC
GGA PBE
END
HARTREEFOCK
eor

mv TAPE21 t21.O

$ADFBIN/adf -n1 <<eor
title   H unrestricted correction

ATOMS
H     0.0    0.0    0.0
end

fragments
H        t21.H
end

unrestricted
charge 0 1
NumericalQuality VeryGood
METAGGA
xc
GGA PBE
end
HARTREEFOCK

eor

mv TAPE21 t21.unr.H

$ADFBIN/adf -n1 <<eor
title   O unrestricted correction

ATOMS
O     0.0    0.0    0.0
end

fragments
O        t21.O
end

unrestricted
charge 0 2
NumericalQuality VeryGood
METAGGA
```

```
symmetry C(lin)
Irrepoccupations
sigma  3 // 3
pi     2 // 0
end
xc
GGA PBE
end
HARTREEFOCK


eor


mv TAPE21 t21.unr.O




$ADFBIN/adf <<eor
TITLE  OH radical, C*v symm. MP2(full)/6-31G* Geom. MP2=-75.5232063              ␣
↪

ATOMS
O    0.000000    0.000000    0.108786
H    0.000000    0.000000   -0.870284
END

SYMMETRY C(2V)

IrrepOCCUPATIONS
A1 3 // 3
A2 0 // 0
B1 1 // 1
B2 1 // 0
END

UNRESTRICTED

DEPENDENCY bas=1.0e-4
NOPRINT BAS FIT FUNCTIONS SFO
METAGGA

XC
GGA PBE
END
HARTREEFOCK

CHARGE 0 1

NumericalQuality VeryGood

FRAGMENTS
O  t21.O
H  t21.H
END

ENERGYFRAG
O  t21.unr.O
H  t21.unr.H
```

```
END

eor

rm TAPE21 logfile

$ADFBIN/adf -n1 <<eor
create H file=$ADFRESOURCES/ET/ET-QZ3P/H
XC
GGA OLYP
END
HARTREEFOCK
eor

mv TAPE21 t21.H

$ADFBIN/adf -n1 <<eor
create O file=$ADFRESOURCES/ET/ET-QZ3P/O
XC
GGA OLYP
END
HARTREEFOCK
eor

mv TAPE21 t21.O

$ADFBIN/adf -n1 <<eor
title   H unrestricted correction

ATOMS
H    0.0    0.0    0.0
end

fragments
H        t21.H
end

unrestricted
charge 0 1
NumericalQuality VeryGood
METAGGA
xc
GGA OLYP
end
HARTREEFOCK

eor

mv TAPE21 t21.unr.H

$ADFBIN/adf -n1 <<eor
title   O unrestricted correction

ATOMS
O    0.0    0.0    0.0
end

fragments
```

```
O          t21.O
end

unrestricted
charge 0 2
NumericalQuality VeryGood
METAGGA
symmetry C(lin)
Irrepoccupations
sigma  3 // 3
pi     2 // 0
end
xc
GGA OLYP
end
HARTREEFOCK

eor

mv TAPE21 t21.unr.O




$ADFBIN/adf <<eor
TITLE  OH radical, C*v symm. MP2(full)/6-31G* Geom. MP2=-75.5232063          ␣
  ↪

ATOMS
O    0.000000   0.000000   0.108786
H    0.000000   0.000000  -0.870284
END

SYMMETRY C(2V)

IrrepOCCUPATIONS
A1 3 // 3
A2 0 // 0
B1 1 // 1
B2 1 // 0
END

UNRESTRICTED

DEPENDENCY bas=1.0e-4
NOPRINT BAS FIT FUNCTIONS SFO
METAGGA

XC
GGA OLYP
END
HARTREEFOCK

CHARGE 0 1

NumericalQuality VeryGood

FRAGMENTS
```

```
O  t21.O
H  t21.H
END

ENERGYFRAG
O  t21.unr.O
H  t21.unr.H
END

eor
```

Example: Hartree-Fock: HI

Download HI_EFG.run

```sh
#! /bin/sh


# Example shows a Hartree-Fock calculation with a non-relativistic, scalar␣
↪relativistic ZORA,
# and a spin-orbit coupled ZORA Hamiltonian. In this case ADF also calculates
# the electric field gradient (EFG) at the H and I nuclei (keyword QTENS).

# First the non-relativistic calculation.
# Note that in this case the all-electron basis sets are obtained from the
↪$ADFRESOURCES/ZORA directory

# == non-relativistic ==


$ADFBIN/adf  <<eor
Atoms
 H 0 0 0
 I 0 0 1.609
End

qtens

xc
 hartreefock
end

BeckeGrid
 Quality good
End

Basis
 Type ZORA/TZ2P
 Core None
End
eor

rm TAPE21 logfile

# == scalar relativistic ZORA ==

# Next the scalar relativistic ZORA calculation. Note that in this case the all-
↪electron basis sets are
```

```
# also obtained from the $ADFRESOURCES/ZORA directory, but this is default place␣
↪where the key BASIS will
# search for basis sets in case of ZORA. ADF will also calculate the EFG including␣
↪the small component density,
# also called SR ZORA-4.

$ADFBIN/adf  <<eor
Atoms
 H 0 0 0
 I 0 0 1.609
End

qtens

xc
 hartreefock
end

Relativistic  Scalar ZORA

BeckeGrid
 Quality good
End

Basis
 Type TZ2P
 Core None
End
eor

rm TAPE21 logfile


# == Relativistic Spinorbit ZORA ==

# Next the spin-orbit coupled relativistic ZORA calculation. Note that in this case␣
↪the all-electron basis sets are
# also obtained from the $ADFRESOURCES/ZORA directory, but again this is default␣
↪place where the key BASIS will search
# for basis sets in case of ZORA. ADF will calculate the EFG including the small␣
↪component density, also called ZORA-4.

$ADFBIN/adf  <<eor
Atoms
 H 0 0 0
 I 0 0 1.609
End

qtens

xc
 hartreefock
end

Relativistic Spinorbit ZORA

BeckeGrid
 Quality good
```

```
End

Basis
 Type TZ2P
 Core None
End
eor

rm TAPE21 logfile


# == Relativistic Spinorbit ZORA with no symmetry ==

# Should give the same results as with symmetry.

$ADFBIN/adf  <<eor
Atoms
 H 0 0 0
 I 0 0 1.609
End

qtens

xc
 hartreefock
end

Relativistic Spinorbit ZORA
symmetry nosym

BeckeGrid
 Quality good
End

Basis
 Type TZ2P
 Core None
End
eor

rm TAPE21 logfile
```

Example: B3LYP: H2PO

```
Download H2PO_B3LYP.run
```

```
#! /bin/sh

# Example shows an unrestricted B3LYP calculation. In this case ADF also
# calculates the hyperfine interactions at H, P, and O nuclei (keyword ESR).

# The 'RIHartreeFock%DependencyThreshold' key is set to 1E-4.
# One should check that the results remain reliable if one uses a smaller value
# for the DependencyThreshold key.


$ADFBIN/adf  <<eor
```

```
Title hfs H2PO B3LYP TZ2P
Atoms
    O   1.492  0.000  0.000
    P   0.000  0.000  0.000
    H  -0.600 -0.650  1.100
    H  -0.600 -0.650 -1.100
End

xc
 hybrid B3LYP
end

Basis
 Type TZ2P
 Core None
End

NumericalQuality good

RIHartreeFock
   Quality Normal
   DependencyThreshold 1E-4
End

esr
end
unrestricted
charge 0 1

eor

rm TAPE21 logfile


# For the hyperfine interactions it is important to use all-electron basis sets
# on the interesting nuclei. One can get more accurate results if one uses a
# larger basis set, like the QZ4P basis set, which is present in the
# $ADFRESOURCES/ZORA directory.


$ADFBIN/adf  <<eor
Title hfs H2PO B3LYP QZ4P
Atoms
    O   1.492  0.000  0.000
    P   0.000  0.000  0.000
    H  -0.600 -0.650  1.100
    H  -0.600 -0.650 -1.100
End

xc
 hybrid B3LYP
end

Basis
 Type ZORA/QZ4P
 Core None
End
```

```
RIHartreeFock
   Quality Normal
   DependencyThreshold 1E-4
End

NumericalQuality good

esr
end
unrestricted
charge 0 1

eor

rm TAPE21 logfile

# The QZ4P results for the isotropic value of the A-tensor are approximately:
# -24.61 MHz for 17O, 957.33 MHz for 31P, and 110.83 MHz for 1H.

# You may want to compare the results with previous B3LYP results by N. R.
# Brinkmann and I. Carmichael, J. Phys. Chem. A (2004), 108, 9390-9399, which
# give for the Isotropic Fermi Contact Couplings (MHz) for the 2 A' State of H2
# PO using B3LYP, with an aug-cc-pCVQZ basis set: -24.24 MHz for 17O, 963.33
# MHz for 31P, and 111.51 MHz for 1H.
```

Example: Long-range corrected GGA functional LCY-BP: H2O

Download LCY-BP_H2O.run

```
#! /bin/sh

# Example for the range separated functional LCY-BP for a geometry optimization
# of water.

$ADFBIN/adf <<eor
TITLE  H2O geometry with LCY-BP functional

ATOMS
O     0.000000    0.000000    0.000000
H     0.000000    0.500000   -0.6
H     0.000000   -0.500000   -0.6
END

NOPRINT BAS FIT FUNCTIONS SFO

XC
 GGA Becke Perdew
 xcfun
 RANGESEP GAMMA=0.75
END

NumericalQuality Good

GEOMETRY
  Converge grad=0.0001
END
```

```
Basis
 Type DZP
END

eor

rm TAPE21 logfile
```

Example: Range-separated functional CAMY-B3LYP: H2O

Download CAMY-B3LYP_H2O.run

```sh
#! /bin/sh



# Example for the range separated functional CAMY-B3LYP for a geometry
# optimization of water. Note that CAMY-B3LYP is not the same as CAM-B3LYP.


$ADFBIN/adf <<eor
   TITLE  H2O geometry with LC-BP functional

   ATOMS
      O    0.000000    0.000000    0.000000
      H    0.000000    0.500000   -0.6
      H    0.000000   -0.500000   -0.6
   END

   NOPRINT BAS FIT FUNCTIONS SFO

   XC
      HYBRID CAMYB3LYP
      xcfun
      RANGESEP GAMMA=0.34
   END

   BeckeGrid
      Quality good
   End

   GEOMETRY
      Converge grad=0.0001
   END

   RIHartreeFock
      UseMe False
   End

   Basis
      Type DZP
   END

eor
```

Example: Grimme Molecular Mechanics dispersion-corrected functionals (DFT-D3-BJ)

Download MM_Dispersion.run

```sh
#! /bin/sh

# In this example a structure with 2 benzene molecules and a hydrogen molecule
# is optimized with the Grimme dispersion corrected PBE, in this case the (so
# far) latest variant the DFT-D family, which is DFT-D3-BJ. Needed is the subkey
# DISPERSION in the key XC, and the arguments Grime3 and BJDAMP. If one starts
# with atomic fragments the part of the bond energy that is due to the Grimme
# dispersion corrected functional is both inter-molecular as well as intra-
# molecular.


$ADFBIN/adf <<eor
Title Geometry optimization with Grimme3-BJ dispersion correction for GGA
basis
  type TZP
  core small
End

XC
  GGA PBE
  DISPERSION Grimme3 BJDAMP
End

geometry
  converge grad=0.0001 rad=0.001
  iterations 50
end

SCF
  Iterations  60
  Converge  1.0E-06  1.0E-6
End

noprint sfo

Atoms    cartesians
C  0.000000000000     3.050000000000     1.391500000000
H  0.000000000000     3.050000000000     2.471500000000
C  1.205074349366     3.050000000000     0.695750000000
H  2.140381785453     3.050000000000     1.235750000000
C  1.205074349366     3.050000000000    -0.695750000000
H  2.140381785453     3.050000000000    -1.235750000000
C -0.000000000000     3.050000000000    -1.391500000000
H -0.000000000000     3.050000000000    -2.471500000000
C -1.205074349366     3.050000000000    -0.695750000000
H -2.140381785453     3.050000000000    -1.235750000000
C -1.205074349366     3.050000000000     0.695750000000
H -2.140381785453     3.050000000000     1.235750000000
C -1.205074349366    -3.050000000000    -0.695750000000
H -2.140381785453    -3.050000000000    -1.235750000000
C -0.000000000000    -3.050000000000    -1.391500000000
H -0.000000000000    -3.050000000000    -2.471500000000
C  1.205074349366    -3.050000000000    -0.695750000000
H  2.140381785453    -3.050000000000    -1.235750000000
```

```
C  1.205074349366    -3.050000000000     0.695750000000
H  2.140381785453    -3.050000000000     1.235750000000
C -0.000000000000    -3.050000000000     1.391500000000
H -0.000000000000    -3.050000000000     2.471500000000
C -1.205074349366    -3.050000000000     0.695750000000
H -2.140381785453    -3.050000000000     1.235750000000
H  0.0                0.35               0.0
H  0.0               -0.35               0.0
End

eor

mv TAPE21 Grimme-dispersion.t21


# The next calculations in this example demonstrate dispersion corrections when
# using non-atomic fragments. First three molecules (2 benzene molecules and a
# hydrogen molecule) are calculated . Needed again is the subkey DISPERSION in
# the key XC.


$ADFBIN/adf <<eor
Title Grimme3-BJ dispersion-corrected GGA
basis
  type TZP
  core small
End

XC
  GGA PBE
  DISPERSION Grimme3 BJDAMP
End

SCF
  Iterations  60
  Converge  1.0E-06  1.0E-6
End

noprint sfo

Atoms
C        0.000000   1.398973   -3.054539
H        0.000000   2.490908   -3.049828
C        1.211546   0.699486   -3.054539
H        2.157190   1.245454   -3.049828
C        1.211546  -0.699486   -3.054539
H        2.157190  -1.245454   -3.049828
C        0.000000  -1.398973   -3.054539
H        0.000000  -2.490908   -3.049828
C       -1.211546  -0.699486   -3.054539
H       -2.157190  -1.245454   -3.049828
C       -1.211546   0.699486   -3.054539
H       -2.157190   1.245454   -3.049828
End

eor

mv TAPE21 benzene1.t21
```

```
$ADFBIN/adf <<eor
Title Grimme3-BJ dispersion-corrected GGA
basis
  type TZP
  core small
End

XC
  GGA PBE
  DISPERSION Grimme3 BJDAMP
End

SCF
  Iterations  60
  Converge  1.0E-06  1.0E-6
End

noprint sfo

Atoms
C      -1.211546   -0.699486    3.054539
H      -2.157190   -1.245454    3.049828
C       0.000000   -1.398973    3.054539
H       0.000000   -2.490908    3.049828
C       1.211546   -0.699486    3.054539
H       2.157190   -1.245454    3.049828
C       1.211546    0.699486    3.054539
H       2.157190    1.245454    3.049828
C       0.000000    1.398973    3.054539
H       0.000000    2.490908    3.049828
C      -1.211546    0.699486    3.054539
H      -2.157190    1.245454    3.049828
End

eor

mv TAPE21 benzene2.t21

$ADFBIN/adf <<eor
Title Grimme3-BJ dispersion-corrected GGA
basis
  type TZP
  core small
End

XC
  GGA PBE
  DISPERSION Grimme3 BJDAMP
End

SCF
  Iterations  60
  Converge  1.0E-06  1.0E-6
End

noprint sfo
```

```
Atoms
H          0.000000    0.000000   -0.377906
H          0.000000    0.000000    0.377906
End

eor

mv TAPE21 h2.t21

$ADFBIN/adf <<eor
Title Grimme3-BJ dispersion-corrected GGA

Fragments
  b1 benzene1.t21
  b2 benzene2.t21
  h2 h2.t21
End

XC
  GGA PBE
  DISPERSION Grimme3 BJDAMP
End

SCF
  Iterations  60
  Converge  1.0E-06  1.0E-6
End

noprint sfo

Atoms
C          0.000000    1.398973   -3.054539  f=b1
H          0.000000    2.490908   -3.049828  f=b1
C          1.211546    0.699486   -3.054539  f=b1
H          2.157190    1.245454   -3.049828  f=b1
C          1.211546   -0.699486   -3.054539  f=b1
H          2.157190   -1.245454   -3.049828  f=b1
C          0.000000   -1.398973   -3.054539  f=b1
H          0.000000   -2.490908   -3.049828  f=b1
C         -1.211546   -0.699486   -3.054539  f=b1
H         -2.157190   -1.245454   -3.049828  f=b1
C         -1.211546    0.699486   -3.054539  f=b1
H         -2.157190    1.245454   -3.049828  f=b1
C         -1.211546   -0.699486    3.054539  f=b2
H         -2.157190   -1.245454    3.049828  f=b2
C          0.000000   -1.398973    3.054539  f=b2
H          0.000000   -2.490908    3.049828  f=b2
C          1.211546   -0.699486    3.054539  f=b2
H          2.157190   -1.245454    3.049828  f=b2
C          1.211546    0.699486    3.054539  f=b2
H          2.157190    1.245454    3.049828  f=b2
C          0.000000    1.398973    3.054539  f=b2
H          0.000000    2.490908    3.049828  f=b2
C         -1.211546    0.699486    3.054539  f=b2
H         -2.157190    1.245454    3.049828  f=b2
H          0.000000    0.000000   -0.377906  f=h2
H          0.000000    0.000000    0.377906  f=h2
End
```

```
eor

mv TAPE21 fragments.t21
```

Example: Density-Dependent Dispersion Correction (dDsC): CH4-dimer

Download methane_dimer_dispersion.run

```
#! /bin/sh

# The density-dependent dispersion energy correction, dDsC, by S.N. Steinmann
# and C. Corminboeuf, is used to calculate interactions between non-overlapping
# densities, which standard density functional approximations cannot accurately
# describe. The example is for the methane dimer.

# The 'debug dispersion' is included such that in the output one can find more
# details on the exact parameters that are used.


$ADFBIN/adf <<eor
TITLE methane-dimer non-relativistic

debug dispersion

ATOMS
C  -0.000959    0.000775    1.853082
H  -0.747186    0.712608    1.489389
H   0.987865    0.294742    1.490258
H  -0.241511   -0.998876    1.480724
H  -0.002970   -0.005330    2.946903
C   0.000962   -0.000776   -1.853082
H   0.004264    0.004684   -2.946903
H  -0.989749   -0.289559   -1.491241
H   0.743184   -0.716138   -1.488107
H   0.246099    0.997870   -1.481022
End

Basis
 Type TZP
End

NumericalQuality good

XC
 GGA Becke LYP
 Dispersion DDsC
End

Geometry
End

eprint
geostep Energy GradientTerms Gradients Upd
end

symmetry nosym
```

```
eor


mv TAPE21 Methane2_nonrel.t21
rm logfile
```

Example: DFT-ulg Dispersion Correction: Benzene dimer T-shaped

Download Benzene_dimer_tshaped.run

```
#! /bin/sh

# The universal correction of density functional theory to include London
# dispersion (DFT-ulg) is used, which is related to UFF. This is yet another MM
# dispersion correction. The example is for the T-shaped benzene dimer.

$ADFBIN/adf <<eor
Title Calculate the optimum structure of a dibenzene molecule and dispersion

Atoms
   C 0.000000 0.000000 1.059035
   C 0.000000 -1.206008 1.757674
   C 0.000000 -1.207177 3.151591
   C 0.000000 0.000000 3.848575
   C 0.000000 1.207177 3.151591
   C 0.000000 1.206008 1.757674
   H 0.000000 0.000000 -0.021580
   H 0.000000 -2.141639 1.214422
   H 0.000000 -2.143566 3.692995
   H 0.000000 0.000000 4.930150
   H 0.000000 2.143566 3.692995
   H 0.000000 2.141639 1.214422
   C -1.394063 0.000000 -2.454152
   C -0.697047 1.207238 -2.454628
   C 0.697047 1.207238 -2.454628
   C 1.394063 0.000000 -2.454152
   C 0.697047 -1.207238 -2.454628
   C -0.697047 -1.207238 -2.454628
   H -2.475399 0.000000 -2.450322
   H -1.238232 2.143565 -2.453676
   H 1.238232 2.143565 -2.453676
   H 2.475399 0.000000 -2.450322
   H 1.238232 -2.143565 -2.453676
   H -1.238232 -2.143565 -2.453676
end

Basis
   type TZP
End

XC
   GGA PBE
   dispersion uff
END

eor
```

```
$ADFBIN/densf <<eor
Grid Medium
End

NCI BOTH
eor
```

14.2.2 ZORA and spin-orbit Relativistic Effects

Example: ZORA Relativistic Effects: Au2

Download Au2_ZORA.run

```
#! /bin/sh


# A relativistic geometry optimization with the ZORA formalism. Both a scalar
# relativistic geometry optimization, as well as a spin-orbit coupled
# relativistic geometry optimization are performed. Spin-orbit effects on the
# geometry are often not so large for closed shell molecules, but takes
# typically a factor 4 times as much as a scalar relativistic calculation.


$ADFBIN/adf <<eor
Title  Au2  relativistic optimization: scalar ZORA

Atoms  Zmat
  Au   0 0 0
  Au   1 0 0  2.5
End

Basis
 Type TZ2P
 Core Small
End

XC
 GGA Becke Perdew
End

Relativistic scalar ZORA

Geometry
  converge grad=1e-4
End

eor

rm TAPE21 logfile


$ADFBIN/adf <<eor
Title  Au2  relativistic optimization: spinorbit ZORA

Atoms  Zmat
```

```
  Au   0 0 0
  Au   1 0 0  2.5
End

Basis
 Type TZ2P
 Core Small
End

Relativistic SpinOrbit ZORA

Geometry
  converge grad=1e-4
End

eor

rm TAPE21 logfile
```

Example: Spin-Orbit coupling: Bi and Bi2

Download SO_ZORA_Bi2.run

```
#! /bin/sh

# Application of the Spin-Orbit relativistic option (using double-group
# symmetry) to Bismuth (atom and dimer).

# For comparison with the full double-group calculation, the 'standard'
# unrestricted calculation on Bismuth is carried out, using the scalar
# relativistic option.

# A net spin polarization of 3 electrons is applied (key charge).

$ADFBIN/adf <<eor
title Bi unrestricted
BeckeGrid
 quality good
End
relativistic scalar ZORA
ATOMS
Bi  0.000000     0.000000     0.00000000
end
Basis
 Type TZ2P
 Core None
end
unrestricted
charge 0 3
xc
  GGA  becke perdew
end
eor

rm TAPE21 logfile

# The CHARGE key, in conjunction with the UNRESTRICTED key is used to specify
```

```
# that 3 electrons must be unpaired (second value of the CHARGE key), while the
# system is neutral (first value of the CHARGE key).

# Next we do a Spin-Orbit calculation on the Bismuth atom.

# Note that it is a 'unrestricted' run using the noncollinear approximation, and
# SYMMETRY NSOYM. The electronic charge density is spin-polarized.

$ADFBIN/adf <<eor
title Bi spinorbit unrestricted noncollinear
BeckeGrid
 quality good
End
relativistic spinorbit ZORA
ATOMS
Bi   0.000000      0.000000      0.00000000
end
symmetry nosym
unrestricted
noncollinear
Basis
 Type TZ2P
 Core None
end
xc
  GGA  becke perdew
end
eor

rm TAPE21 logfile

# Because an all electron basis set is used, the bond energy is huge, due to the
# very large higher order spin-orbit effect on the 2p orbitals.

# == Bi2 dimer ==

# Now we turn to the dimer Bi2: a series of Single Point calculations, all with
# the same inter atomic distance.

# First the scalar relativistic run.

$ADFBIN/adf <<eor
title   Bi2, scalar relativistic
BeckeGrid
 quality good
End
relativistic scalar ZORA
ATOMS
Bi       0.0             0.0             1.33
Bi       0.0             0.0            -1.33
end
Basis
 Type TZ2P
 Core None
end
xc
  GGA  becke perdew
end
```

```
eor

rm logfile
mv TAPE21 t21Bi2

# The calculated scalar relativistic atomization energy will be close to 2.74
# eV. This is the bond energy of the dimer minus 2 times the bond energy of the
# unrestricted scalar relativistic atom.

# The result file tape21 is used as reference in subsequent calculations: run
# the spin-orbit case starting from the just completed dimer calculation as a
# fragment. The resulting 'bonding energy', ie the energy w.r.t. the scalar
# relativistic ZORA dimer, gives directly the effect of the full-relativistic
# ZORA versus the scalar relativistic ZORA option: the energy is lowered by huge
# amount, again mainly due to the large spin-orbit effect on the 2p orbitals.

$ADFBIN/adf <<eor
title   Bi2 from fragment Bi2,  with SpinOrbit coupling
PRINT SpinOrbit
BeckeGrid
 quality good
End
relativistic spinorbit ZORA
ATOMS
Bi    0.0    0.0    1.33  f=Bi2
Bi    0.0    0.0   -1.33  f=Bi2
end
fragments
Bi2     t21Bi2
end
xc
  GGA  becke perdew
end
eor

rm TAPE21 logfile

# If one looks at the SFO analysis in the output of this calculation, one can
# observe that a first-order spin-orbit splitting of the scalar relativistic
# fragment orbitals is a good approximation to many of the calculated valence
# spinors.

# A final consistency check: run the spin-orbit dimer from single-atom
# fragments. The bonding energy should equal the sum of the bonding energies of
# the previous two runs: scalar relativistic dimer w.r.t. single atom fragments
# plus spin-orbit dimer w.r.t. the scalar relativistic dimer.

$ADFBIN/adf <<eor
title   Bi2 from atomic fragments, SpinOrbit coupling
PRINT SpinOrbit
BeckeGrid
 quality good
End
relativistic spinorbit ZORA
ATOMS
Bi      0.0            0.0            1.33
Bi      0.0            0.0           -1.33
end
```

```
Basis
 Type TZ2P
 Core None
end
xc
  GGA  becke perdew
end
eor

rm logfile
mv TAPE21 Bi2.t21


# The calculated spin-orbit coupled relativistic atomization energy will be
# close to 2.18 eV. This is the bond energy of the dimer minus 2 times the bond
# energy of the unrestricted non-collinear spin-orbit coupled relativistic atom.
# Note that one has to subtract huge numbers.
```

Example: Spin-Orbit unrestricted non-collinear: Tl

Download Tl_noncollinear.run

```
#! /bin/sh

# Application of the Spin-Orbit relativistic option (using double-group
# symmetry, in this case NOSYM) to Tl using the collinear and non-collinear
# approximation for unrestricted Spin-Orbit calculations

# Note: For the collinear and the non-collinear approximation one should use
# symmetry NOSYM and use the key UNRESTRICTED.

# == Title Tl scalar restricted ==

$ADFBIN/adf -n1 <<eor
Title Tl scalar restricted
Atoms
 Tl 0 0 0
End
XC
 GGA Becke Perdew
End
Relativistic Scalar ZORA
Basis
 Tl ZORA/TZ2P/Tl.4f
End
BeckeGrid
 quality good
End
eor
mv TAPE21 Tl.t21

# == Tl spinorbit restricted ==

$ADFBIN/adf  <<eor
Title Tl spinorbit restricted
Atoms
 Tl 0 0 0
End
```

```
Relativistic  Spinorbit ZORA
XC
 GGA becke perdew
end
EPRINT
 SFO eig
END
Fragments
 Tl Tl.t21
End
BeckeGrid
 quality good
End
eor

rm TAPE21 logfile

# == Tl spinorbit collinear z-axis ==

$ADFBIN/adf  <<eor
Title Tl spinorbit collinear z-axis
Atoms
 Tl 0 0 0
End
Relativistic  Spinorbit ZORA
XC
 GGA becke perdew
end
symmetry nosym
unrestricted
collinear
EPRINT
 SFO eig
END
Fragments
 Tl Tl.t21
End
BeckeGrid
 quality good
End
eor

rm TAPE21 logfile

# == Tl spinorbit noncollinear ==

$ADFBIN/adf  <<eor
Title Tl spinorbit noncollinear
Atoms
 Tl 0 0 0
End
Relativistic  Spinorbit ZORA
XC
 GGA becke perdew
end
symmetry nosym
unrestricted
noncollinear
```

```
EPRINT
 SFO eig
END
Fragments
 Tl Tl.t21
End
BeckeGrid
 quality good
End
eor

rm TAPE21 logfile

# == Tl spinorbit noncollinear ==

$ADFBIN/adf  <<eor
Title Tl spinorbit noncollinear
Atoms
 Tl 0 0 0
End
Relativistic  Spinorbit ZORA
XC
 GGA becke perdew
end
symmetry nosym
unrestricted
noncollinear
SOUEXACT
EPRINT
 SFO eig
END
Fragments
 Tl Tl.t21
End
BeckeGrid
 quality good
End
eor

rm TAPE21 logfile
```

Example: Excitation energies including spin-orbit coupling: AuH

Download AuH_analyse_exciso.run

```
#! /bin/sh


# Calculation of the excitation energies of AuH including spin-orbit coupling.

# ADF can not handle ATOM and linear symmetries in excitation calculations.
# Therefore a subsymmetry is used, in this case symmetry C(7v).

# A relatively small TZ2P basis set is used, which is not sufficient for
# excitations to Rydberg-like orbitals, one needs more diffuse functions.

# The key STCONTRIB is used, which will give a composition of the spin-orbit
```

```
# coupled excitation in terms of singlet-singlet and singlet-triplet scalar
# relativistic excitations. In order to use the key STCONTRIB the scalar
# relativistic fragment should be the complete molecule.

# For precision reasons the Beckegrid quality is set to good. One might also
# increase the ZlmFit quality for better accuracy

# One needs to include the subkey SFO of the key EPRINT with arguments eig and
# ovl in order to get the SFO MO coefficients and SFO overlap matrix printed on
# standard output.


$ADFBIN/adf  <<eor
Title [AuH]

Atoms
 Au .0000  .0000  1.5238
  H .0000  .0000  0.0000
End

relativistic scalar zora

Basis
 Type TZ2P
 Core None
End

symmetry C(7v)

EPRINT
 SFO eig ovl
END

BeckeGrid
 quality good
End

Excitations
 lowest 40
 nto
 sfoanalysis
 descriptors
End

eor

mv TAPE21 t21.frag
rm logfile

$ADFBIN/adf <<eor
Title [AuH]

Atoms
 Au .0000  .0000  1.5238  f=Frag
  H .0000  .0000  0.0000  f=Frag
End

relativistic spinorbit zora
```

```
symmetry C(7v)

EPRINT
SFO eig ovl
END

BeckeGrid
 quality good
End

Excitations
 lowest 40
 nto
 sfoanalysis
 descriptors
End

Fragments
 Frag t21.frag
End

STCONTRIB

eor

rm logfile TAPE21
```

Example: ZORA, X2C and RA-X2C: HgI2 = Hg + I2

Download HgI2.run

```
#! /bin/sh

# The scalar relativistic ZORA, scalar relativistic X2C, and scalar relativistic
# RA-X2C are compared in this example for the bond dissociation energy of HgI2
# in Hg and I2. The division between scalar and spin-orbit relativistic effects
# is somewhat arbitrary and depends on the method of separation. Thus the scalar
# relativistic results of X2C and RA-X2C will be different even in the basis set
# limit. However, the differences for bond energies are not so large. The
# results for scalar relativistic ZORA are often closer to scalar relativistic
# RA-X2C, than to scalar relativistic X2C.

for f in "ZORA" "X2C" "RA-X2C"
do

$ADFBIN/adf <<eor
ATOMS
 Hg 0 0  0
 I  0 0  2.554
 I  0 0 -2.554
END
Basis
Type ZORA/DZ
Core None
End
numericalquality good
```

```
xc
 gga bp86
end
relativistic scalar $f
eor
rm TAPE21 logfile

$ADFBIN/adf <<eor
ATOMS
 I  0 0  1.333
 I  0 0 -1.333
END
Basis
Type ZORA/DZ
Core None
End
numericalquality good
xc
 gga bp86
end
relativistic scalar $f
eor
rm TAPE21 logfile

done
```

14.2.3 Solvents, other environments

Example: COSMO: HCl

```
Download Solv_HCl.run
```

```
#! /bin/sh


# Computing solvent effects, with the COSMO model, is illustrated in the HCl
# example.

# After a non-solvent (reference) calculation, which is omitted here, two
# solvent runs are presented, with somewhat different settings for a few input
# parameters. The block key Solvation controls all solvent-related input.

# All subkeys in the SOLVATION block are discussed in the User's Guide. Most of
# them are rather technical and should not severely affect the outcome.
# Physically relevant is the specification of the solute properties, by the
# SOLVENT subkey: the dielectric constant and the effective radius of the
# solvent molecule.

# Note that a non-electrostatic terms as a function of surface area is included
# in the COSMO calculation, by setting the values for CAV0 and CAV1 in the
# subkey SOLVENT of the key SOLVATION. In ADF2010 one should explicitly include
# such values for CAV0 and CAV1, otherwise this non-electrostatic term will be
# taken to be zero, since the defaults have changed in ADF2010.

# A rather strong impact on the computation times has the method of treating the
# 'C-matrix'. There are 3 options (see the User's Guide): EXACT is the most
```

```
# expensive, but presumably most accurate. POTENTIAL is the cheapest alternative
# and is usually quite adequate. EXACT uses the exact charge density for the
# Coulomb interaction between the molecular charge distribution and the point
# charges (on the Van der Waals type molecular surface) which model the effects
# of the solvent. The alternatives, notably 'POTENTIAL', use the fitted charge
# density instead. Assuming that the fit is a fairly accurate approximation to
# the exact charge density, the difference in outcome should be marginal.


$ADFBIN/adf  <<eor
TITLE  HCl(0) reference run (gas phase)

ATOMS
  H    0.000000     0.000000     0.000000
  Cl   1.304188     0.000000     0.000000
END

Basis
 Type DZP
End

NOPRINT Bas EigSFO EKin SFO, frag, functions
EPRINT
SCF NoEigvec
END
eor

rm TAPE21 logfile


$ADFBIN/adf <<eor
TITLE  HCl(1) Solv-excl surfac; Gauss-Seidel (old std options)

SYMMETRY  NOSYM

ATOMS
  H    0.000000     0.000000     0.000000     R=1.18
  Cl   1.304188     0.000000     0.000000     R=1.75
END

Fragments
 H  t21.H
 Cl t21.Cl
End

SOLVATION
   Solv      epsilon=78.8 radius=1.4 cav0=1.321 cav1=0.0067639
   Surf      delley
   Div       ND=4  min=0.5  Ofac=0.8
   Charged   Method=Gauss-Seidel
   Disc      SCale=0.01  LEGendre=10 TOLerance=1.0e-2
   SCF       Variational
   C-Mat      Exact
END

NOPRINT Bas EigSFO EKin SFO, frag, functions
EPRINT
SCF NoEigvec
```

```
END
eor

rm TAPE21 logfile

# In the second solvent run, another (technical) method is used for determining
# the charge distribution on the cavity surface (conjugate-gradient versus
# Gauss-Seidel in the previous calculation), and the POTENTIAL variety is used
# for the C-matrix handling. The results show that it makes little difference in
# outcome, but quite a bit in computation times.

$ADFBIN/adf  <<eor
TITLE  HCl(9) NoDisk and Cmatrix potential

FRAGMENTS
   H    t21.H
   Cl   t21.Cl
END

ATOMS
  H    0.000000     0.000000    0.000000    R=1.18
  Cl   1.304188     0.000000    0.000000    R=1.75
END

SOLVATION
   Solv      epsilon=78.8 radius=1.4 cav0=1.321 cav1=0.0067639
   Surf      delley
   Div       ND=4  min=0.5  Ofac=0.8
   Charged   Method=conjugate-gradient
   SCF       Variational
   C-Mat      POTENTIAL
END

NOPRINT Bas EigSFO EKin SFO, frag, functions
EPRINT
SCF NoEigvec
END
eor
```

Example: solvation model SM12: Acetamide

Download Acetamide_SM12.run

```
#! /bin/sh

# In this example solvent effects are calculated with the solvation model SM12
# for Acetamide.

# After a non-solvent (reference gas phase) calculation, the same geometry is
# used in the solvated case. At the moment it is not possible to do a geometry
# optimization with the SM12 model in ADF. The default SM12 parameters are used,
# which means a solvation in Water is simulated.


$ADFBIN/adf <<eor
   Atoms
      O      -0.08557000     1.83373000     0.00836000
```

```
      C        0.09782000        0.61787000       -0.01303000
      C       -1.04418000       -0.38854000       -0.01026000
      N        1.35837000        0.07376000       -0.01671000
      H       -0.79705000       -1.31834000       -0.54107000
      H       -1.92168000        0.07841000       -0.46849000
      H       -1.29425000       -0.64220000        1.03061000
      H        1.51930000       -0.92568000       -0.01800000
      H        2.15296000        0.70225000        0.03378000
   END

   Basis
      Type TZ2P
      Core None
   End

   XC
      hybrid B3LYP
   End

eor

mv TAPE21 Acetamide.t21
rm logfile


$ADFBIN/adf <<eor
   Atoms
      O       -0.08557000        1.83373000        0.00836000 f=Acetamide
      C        0.09782000        0.61787000       -0.01303000 f=Acetamide
      C       -1.04418000       -0.38854000       -0.01026000 f=Acetamide
      N        1.35837000        0.07376000       -0.01671000 f=Acetamide
      H       -0.79705000       -1.31834000       -0.54107000 f=Acetamide
      H       -1.92168000        0.07841000       -0.46849000 f=Acetamide
      H       -1.29425000       -0.64220000        1.03061000 f=Acetamide
      H        1.51930000       -0.92568000       -0.01800000 f=Acetamide
      H        2.15296000        0.70225000        0.03378000 f=Acetamide
   END

   Fragments
      Acetamide Acetamide.t21
   End

   XC
      hybrid B3LYP
   End

   Solvation SM12
   End
eor
```

Example: Electric Field, Point Charge(s): N2 and PtCO

```
Download Efield.PntQ_N2.run
```

```sh
#! /bin/sh

# Two illustrations of applying the very useful BASIS keyword and of application
```

```
# of an Electric Field.

# For N2, three calculations are provided: 1) a normal N2 run as a reference
# with the BASIS keyword, 2) with a homogeneous electric field, 3) with a point
# charge.

# In this example, no Create run is needed in the input file, because the first
# molecular calculation uses the BASIS keyword. If the $ADFBIN/adf script finds
# this keyword, it will first generate a new input file which will then be
# executed. The new input file will contain the required Create run for the N
# atom in this case. The proper xc functional and relativistic options will
# automatically be selected by the BASIS keyword. This includes Dirac
# calculations in case of relativistic runs. The output files is identical to
# what would have appeared if one would provide the Create runs explicitly in
# the input file. It also copies the atomic input, so that everything can be
# checked.


$ADFBIN/adf -n1 <<eor
title N2  reference for comparison with E-Field runs

atoms
 N   0 0 -.55
 N   0 0 +.55
end

Basis
 Type DZP
 Core Small
End

eor

mv TAPE21 N2.t21
rm logfile


$ADFBIN/adf <<eor
title N2 in a homogeneous electric field
scf
 converge 1e-8
end

atoms
 N   0 0 -.55
 N   0 0 +.55
end
fragments
 N   t21.N
end

EField 0 0 0.01

eor

mv TAPE21 N2_field1.t21
rm logfile
```

```
$ADFBIN/adf <<eor
title  N2 polarized by a point charge on the axis

PointCharges
   0 0 3.0 1.0
end

atoms
 N 0 0 -.55
 N 0 0 .55
end

Fragments
 N  t21.N
end

eor

mv TAPE21 N2_field2.t21


# In the second n2 run the homogeneous field is supplied with the key efield,
# used as simple key: one record, data on the same line as the keyword. The
# field strength is specified in atomic units.

# Homogeneous electric fields can be used to study the polarizability: for
# sufficiently small fields the dipole moment should respond linearly.

# For point charges, the third calculation, the block form of the key efield
# must be used. The program first tries to find data on the same line as the
# keyword (defining a homogeneous field). If this is absent, a data block is
# expected with point-charge specifications: x, y, z and q.

# The coordinates are in the same units as in the atoms block (angstrom by
# default) (but always Cartesian). Q is the charge in elementary units (+1 for a
# proton).

# Point charges can be used for instance to simulate crystal fields (Madelung
# potential).

# Note: the symmetry will be determined automatically by the program as C(lin),
# rather than D(lin), in the two runs that involve an electric field: the fields
# break the symmetry.

# For PtCO, a fairly large electric field is applied in combination with a tight
# SCF convergence criterion.

# The BASIS keyword in this example illustrates how different choices can be
# made for different atoms (in this case a frozen core for Pt):
#    Basis
#     Type DZ
#     Core None
#     Pt Pt.4d
#    END
```

```
Download Field_PtCO.run
```

```
#! /bin/sh


$ADFBIN/adf << eor
TITLE Pt-CO in electric field

ATOMS
Pt 0 0 0
C  0 0 1.70
O  0 0 2.85
END

Basis
 Type DZ
 Core None
 Pt Pt.4d
End

EFIELD 0 0 0.01

SCF
CONVERGE 1.0E-10
END

BeckeGrid
 Quality good
End

eor

mv TAPE21 PtCO.t21
```

Example: 3D-RISM: Glycine

Download 3DRISM-Glycine.run

```
#! /bin/sh


# == Expert option ==

# Computing solvent effects with the 3D-RISM model is illustrated on the glycine
# example.

# All subkeys in the RISM block are discussed in the User's Guide. The things to
# pay attention to here are SigU and EpsU parameters for each atom in the ATOMS
# block, the solvent parameters in the SOLVENT sub-block and the FFT box
# parameters in the SOLUTE sub-block. Both SigU and EpsU values as well as the
# solvent parameters may be obtained from force field parameter lists.
# Parameters for some common solvents are available in the ADF User's Guide.

# One should take into account the following when choosing FFT box parameters in
# the SOLUTE block:

# - the box should be at least twice as large as your model in each dimension,
# - the number of grid points in each dimension must be a power of 2, and
# - accuracy of the results and the memory usage depend on the number of grid-
```

```
#   points


# Note: the 3D-RISM part in ADF has not been parallelized, thus calculating on
# more than 1 processor will not reduce the elapsed time. In this example only
# the gradients are calculated, no geometry optimization is performed.


# == Remarks for different solvent than water ==

# The default values for the RISM1D subblock key are only applicable if the
# solvent is water. You must change the FLUIDPARAM subkey of the subblock key
# RISM1D when modeling a different solvent, at least the dielectric constant and
# the density. In that case you may also have to change other subkeys of the
# subblock key RISM1D. For a different solvent than water, of course, you also
# have to change the description of the subblock SOLVENT. You may have to change
# the boxsize.


$ADFBIN/adf << eor
Title 3D-RISM test

SYMMETRY C(s)

NOBECKEGRID

ATOMS   internal
  C   0  0  0   0.0        0.0              0.0          SigU=3.50   EpsU=0.066
  O   1  0  0   1.208031   0.0              0.0          SigU=2.96   EpsU=0.200
  O   1  2  0   1.341959   123.553475       0.0          SigU=2.96   EpsU=0.200
  C   1  2  3   1.495685   124.769221    180.0           SigU=3.50   EpsU=0.066
  N   4  1  2   1.427005   115.495309       0.0          SigU=3.25   EpsU=0.170
  H   3  1  2   0.992780   105.645766       0.0          SigU=1.00   EpsU=0.046
  H   4  1  2   1.107716   107.591718    123.973836      SigU=1.00   EpsU=0.046
  H   4  1  2   1.107716   107.591718   -123.973836      SigU=1.00   EpsU=0.046
  H   5  4  1   1.028574   109.800726     57.697485      SigU=1.00   EpsU=0.046
  H   5  4  1   1.028574   109.800726    -57.697485      SigU=1.00   EpsU=0.046
End


Basis
  Type DZP
  Core small
End

XC
  LDA
End
RISM glycine 1N
  RISM1D
  subend

  SOLVENT1 water
    UNITS      uWeight=g/mol  ULJsize=A  ULJenergy=kcal/mol Ucoord=A Udens=1/A3
    Parameters Weight=18.015    nAtoms=2
    1    -0.8476   3.166    0.1554      0.000000  0.00000  0.000000
    2     0.4238   1.000    0.0460     -0.816490  0.00000  0.577359
                                        0.816490  0.00000  0.577359
      DenSpe=0.03333
  SUBEND
```

```
  SOLUTE   CO
     BOXSIZE 32.0 32.0 32.0
     BOXGRID    64    64    64
  SUBEND
END

Gradient

eor

# The densf utility can be used to convert the 3D-RISM grid data stored in the TAPE21␣
→file to
# the TAPE41 format suitable for visualization by ADFview. The following fields are␣
→created
# for each solvent site: Huv (solvent total correlation function), Uuv (solvent␣
→potential in RT),
# Guv (solvent pair distribution function), Cuv (solvent direct correlation),
# and PMF (potentials of mean force in kcal/mol)


$ADFBIN/densf << eor
RISM
eor

mv TAPE41 glycine.t41
mv TAPE21 glycine.t21
rm -f RISMDATA
```

Example: ReaxFF: ADF geometry optimization using ReaxFF forces

Download reaxff_optimization.run

```
#! /bin/sh

# == Expert Option ==

# An example of ADF optimization using ReaxFF forces.

cp "$ADFHOME/atomicdata/ForceFields/ReaxFF/CuCl-H2O.ff" ffield

cat > control <<eor
# General parameters
     0 itrans
     1 icentr     0: off, 1: put center of masss at center of cube, 2: put com at␣
→origin
     0 imetho     Normal MD-run 1: Energy minimisation
     5 igeofo     5: read from adf-provided file, calc forces and stop
     0 timing     0: no tining, 2: timing details
 250.0 axis1      a cell axis
 250.0 axis2      b cell axis
 250.0 axis3      c cell axis
 90.000 angle1    cell angles
 90.000 angle2    cell angles
 90.000 angle3    cell angles
    25 irecon     Frequency of reading control-file
     5 ixmolo     xmolout 0: xyz only, 1: xyz + vels + molnr, 2: xyz + mol.nr, 5:␣
→xyz + bonds
```

```
      1 ichupd     Charge update frequency
      4 icharg     always 4: Full system EEM
 298.00 mdtemp     MD-temperature (K), unless tregime file is present
  100.0 tdamp1     1st Berendsen/Anderson temperature damping constant (fs)
# MD-parameters
      1 imdmet     MD-method. 1:Velocity Verlet+Berendsen 2:Hoover-Nose (again NVT);␣
→3:NVE 4:NPT
  0.250 tstep      MD-time step (fs)
   0.00 mdpres     MD-pressure (GPa)
  500.0 pdamp1     Berendsen pressure damping constant (fs)
      0 inpt       0: Change all cell parameters in NPT-run  1: fixed x 2: fixed y 3:␣
→fixed z
      1 nmdit      Number of MD-iterations
      5 iout1      Output frequency to unit 71 and unit 73
   5000 iout2      Save coordinates (xmolout, moldyn.vel, Molfra)
      1 iout3     Create moldyn.xxx files (0: yes, 1: no)
      0 ivels      0: Use velocities from vels restart-file; 1: Zero initial␣
→velocities
   2000 iout6      Frequency of molsav.xxxx restart file creation (xyz, vels and␣
→accel)
     50 iout7      Frequency of reaxout.kf writing
     25 irten      Frequency of removal of rotational and translational motions
      0 npreit     Nr. of iterations in previous runs
   0.00 range      range for back-translation of atoms outside periodic box
# MM-parameters
1.00000 endmm      End point criterium for MM energy minimisation (force)
      0 imaxmo     0: conjugate gradient, -1: temperature dependent MD minimization␣
→(with imethod 1)
      1 imaxit     Maximum number of iterations
      1 iout4      Frequency of structure output during minimisation
      0 icelop     0 : no cell opt, 1: numerical cell opt
1.00010 celopt     Cell parameter change factor
      0 icelo2     0: Cubic cell optimization; 1/2/3: only a/b/c; 4: c/a ratio
eor


NSCM=1
export NSCM

"$ADFBIN/adf" <<eor
ATOMS
O                  9.21547  18.20279   5.17255
H                  9.82360  18.92182   4.93781
H                  9.54331  17.44166   4.66744
O                  4.62358  16.65456  21.14691
H                  3.99888  17.39042  21.24792
H                  5.47889  17.03353  21.40531
O                 21.12941   6.02920   5.89396
H                 21.47696   6.74566   6.44878
H                 20.69255   5.43353   6.52347
END

XC
LDA SCF VWN
END

GEOMETRY
   branch New
```

```
    optim cartesian
    iterations 1000
    ExternProgram "sh $ADFBIN/reax2adf"
    ExternProgram coord=adf_reaxff.coords energy=adf_reaxff.energy grad=adf_reaxff.
↪grad
END

SAVE  TAPE21 TAPE13

NOPRINT LOGFILE

Basis
End
eor
```

14.2.4 FDE: Frozen Density Embedding

Example: FDE: H2O in water

Download FDE_H2O_128.run

```sh
#! /bin/sh

# This example demonstrates how to use FDE in combination with a large
# environment, that is modeled as a superposition of the densities of isolated
# molecules. Here, the excitation energies of a water molecule surrounded by an
# environment of 127 water molecules. For details, see also: C.R. Jacob, J.
# Neugebauer, L. Jensen and L. Visscher, Comparison of frozen-density embedding
# and discrete reaction field solvent models for molecular properties, Physical
# Chemistry Chemical Physics 8, 2349 (2006)

# This calculation consists of two steps:
# - First a prototype water molecule is calculated.
# - Next the embedding calculation of water in water is performed.

# To reduce the amount of output the next lines are included in the adf
# calculations:
#   EPRINT
#     SFO NOEIG NOOVL NOORBPOP
#     SCF NOPOP
#   END
#   NOPRINT BAS FUNCTIONS


# create atomic fragment files


$ADFBIN/adf -n 1 <<eor
```

```
create O file=$ADFRESOURCES/DZP/O
eor
mv TAPE21 t21.DZP.O
rm logfile

$ADFBIN/adf -n 1 <<eor
create H file=$ADFRESOURCES/DZP/H
eor
mv TAPE21 t21.DZP.H
rm logfile



# First, a prototype water molecule is calculated. The density of this isolated
# water molecules will afterwards be used to model the environment.


$ADFBIN/adf <<eor
Title Input generated by modco

EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END

NOPRINT BAS FUNCTIONS

UNITS
  length bohr
  angle degree
END

XC
LDA
END

SYMMETRY NOSYM

SCF
  iterations 50
  converge 1.0e-6 1.0e-6
  mixing 0.2
  lshift 0.0
  diis n=10 ok=0.5 cyc=5 cx=5.0 cxx=10.0
END
NumericalQuality good

FRAGMENTS
  O  t21.DZP.O
  H  t21.DZP.H
END

ATOMS
 O      -11.38048700000000   -11.81055300000000    -4.51522600000000
 H      -13.10476265095705   -11.83766918322447    -3.96954531282721
 H      -10.51089289290947   -12.85330720999229    -3.32020577897331
END

eor
```

```
mv TAPE21 t21.mol_1
rm logfile


# Afterwards, the FDE calculation is performed. In this FDE calculation, there
# is one nonfrozen water molecule and the previously prepared water molecule is
# included as a frozen fragment that is duplicated 127 times. For this frozen
# fragment, the more efficient fitted density is used.



$ADFBIN/adf <<eor
Title Input generated by modco

EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END

NOPRINT BAS FUNCTIONS

UNITS
  length bohr
  angle degree
END

XC
MODEL SAOP
END

SYMMETRY NOSYM

NumericalQuality Good

SCF
  iterations 50
  converge 1.0e-6 1.0e-6
  mixing 0.2
  lshift 0.0
  diis n=10 ok=0.5 cyc=5 cx=5.0 cxx=10.0
END

EXCITATIONS
  ONLYSING
  LOWEST  5
END

FRAGMENTS
  O      t21.DZP.O
  H      t21.DZP.H
  frag1  t21.mol_1 type=fde &
    fdedenstype SCFfitted
  SubEnd
END

ATOMS
 O       0.00000000000000      0.00000000000000      0.00000000000000
```

```
H         -1.43014300000000      0.00000000000000      1.10739300000000
H          1.43014300000000      0.00000000000000      1.10739300000000
O        -11.38048700000000    -11.81055300000000     -4.51522600000000    f=frag1/1
H        -13.10476265095705    -11.83766918322447     -3.96954531282721    f=frag1/1
H        -10.51089289290947    -12.85330720999229     -3.32020577897331    f=frag1/1
O         -1.11635000000000      9.11918600000000     -3.23094800000000    f=frag1/2
H         -2.82271357869859      9.71703285239153     -3.18063201242303    f=frag1/2
H         -0.12378551814273     10.53819303003839     -2.70860866559857    f=frag1/2
O        -16.96901200000000     -3.65835300000000    -10.78571300000000    f=frag1/3
H        -15.70823656958949     -4.23996689198245     -9.62648920152293    f=frag1/3
H        -16.47814038897754     -4.36080868684451    -12.37857352487904    f=frag1/3
O        -10.76560800000000     -4.16357700000000      0.51803200000000    f=frag1/4
H         -9.16115845895954     -3.98623150326079     -0.29800688710696    f=frag1/4
H        -11.97483852095230     -3.79636551545084     -0.77600078745886    f=frag1/4
O         -6.55423300000000     -9.20297500000000    -11.23125400000000    f=frag1/5
H         -8.15301944239612     -9.65354271558258    -10.51536304076499    f=frag1/5
H         -6.90715220963067     -7.74998343249339    -12.24902957370683    f=frag1/5
O        -14.16713600000000     -6.55813200000000     -6.62957400000000    f=frag1/6
H        -13.74689800192609     -8.31204468643143     -6.76675830514850    f=frag1/6
H        -15.43294077464585     -6.49537593456625     -5.33906885435316    f=frag1/6
O         -1.01233800000000     -6.53737500000000    -20.63228300000000    f=frag1/7
H          0.51358895613388     -7.01674081354516    -19.78766204266023    f=frag1/7
H         -2.25846754637793     -6.48721751306148    -19.32222614537351    f=frag1/7
O         -7.97372300000000      4.33381600000000      8.85869700000000    f=frag1/8
H         -7.39209897744333      2.71703281820261      9.42381190837837    f=frag1/8
H         -7.34133495205587      4.47772485325760      7.17021271491547    f=frag1/8
O          1.68768000000000    -11.29316300000000     -2.87636700000000    f=frag1/9
H          1.52000307205134    -11.65999321781241     -4.63958827351129    f=frag1/9
H          0.92563970293820     -9.66478590395236     -2.67815953059656    f=frag1/9
O          1.09184600000000     -1.69811700000000    -18.81717100000000    f=frag1/10↵
↪
H          2.07215988260319     -2.34326255514507    -20.19354345122146    f=frag1/10↵
↪
H          0.13398919688371     -3.11903886392137    -18.23829710737747    f=frag1/10↵
↪
O        -13.85258700000000     -1.51317200000000     -3.42847700000000    f=frag1/11↵
↪
H        -13.34561409429037     -1.90947670800034     -5.11890601995987    f=frag1/11↵
↪
H        -15.31887324674441     -2.53345289474331     -3.14451376182312    f=frag1/11↵
↪
O          5.28960400000000     -6.81266800000000    -13.10585800000000    f=frag1/12↵
↪
H          7.01709448187639     -6.62178372023976    -12.60488989024026    f=frag1/12↵
↪
H          4.34671789396407     -5.97791594280375    -11.80749041866683    f=frag1/12↵
↪
O        -11.47407200000000    -10.90708300000000     -9.44085000000000    f=frag1/13↵
↪
H        -11.21611173248622    -12.45441460630369    -10.34132244308043    f=frag1/13↵
↪
H        -11.17634020521638    -11.33593027595366     -7.70907363612158    f=frag1/13
O         -1.64592100000000    -11.43523300000000      1.48098000000000    f=frag1/14↵
↪
H         -0.30557034252542    -10.47362907146570      0.73908565451392    f=frag1/14↵
↪
H         -1.72993170289473    -12.94068698496448      0.48190338473989    f=frag1/14↵
↪
```

```
O        -7.32468400000000      -5.01818100000000       4.38774900000000    f=frag1/15
↪
H        -8.81838077448469      -4.73028985072788       3.40916949260461    f=frag1/15
↪
H        -7.75923926090726      -6.38911120628163       5.48470646359115    f=frag1/15
↪
O         5.20879700000000      -6.83850600000000       3.69146100000000    f=frag1/16
↪
H         5.26579130261858      -6.68511818556767       1.89011343551333    f=frag1/16
↪
H         6.39162739624924      -5.60718291987245       4.28841938067655    f=frag1/16
↪
O         6.73749100000000      -0.26065600000000      -6.58850100000000    f=frag1/17
↪
H         6.56301678911955      -2.04681439624047      -6.81394756463717    f=frag1/17
↪
H         8.29925639010887       0.14828805200118      -7.40413401232653    f=frag1/17
↪
O         5.30624300000000      -5.54309500000000      -6.43599300000000    f=frag1/18
↪
H         3.81279050995479      -6.09072594663625      -7.29699777727678    f=frag1/18
↪
H         6.65189674819999      -6.43527789528092      -7.25136362341575    f=frag1/18
↪
O        -5.72214300000000      -7.59376400000000      -5.60989200000000    f=frag1/19
↪
H        -5.63766272358672      -7.49959469188493      -7.41422690365166    f=frag1/19
↪
H        -3.99734274429106      -7.51425600722341      -5.07105423295232    f=frag1/19
↪
O        -4.96412100000000       4.32334900000000      -3.25899700000000    f=frag1/20
↪
H        -6.26920382212011       5.55951483151335      -3.05829302735551    f=frag1/20
↪
H        -3.43114054713299       5.22630232648312      -2.93303718111639    f=frag1/20
↪
O       -10.94232600000000       0.55437900000000     -13.18244900000000    f=frag1/21
↪
H        -9.99364526446256       2.08335246724114     -13.36649850073860    f=frag1/21
↪
H       -12.25798450895047       0.68645755108753     -14.41662950801929    f=frag1/21
↪
O        -9.29636700000000       5.39923600000000     -12.62463400000000    f=frag1/22
↪
H        -7.78716246923940       5.39969899801535     -11.62767327349474    f=frag1/22
↪
H        -8.71366631604628       5.57073538127912     -14.32835095341294    f=frag1/22
↪
O         2.97641300000000     -12.44506100000000       3.62605000000000    f=frag1/23
↪
H         3.66110125710447     -10.77090992621746       3.63307801290360    f=frag1/23
↪
H         1.22148596207982     -12.21500271544869       3.25334655288432    f=frag1/23
↪
O        -7.52033800000000      -2.70960600000000     -16.43185500000000    f=frag1/24
↪
H        -8.54144292463491      -2.01657808726233     -15.10947598274278    f=frag1/24
↪
```

```
 H         -8.12619297191493         -1.91113986970532        -17.93750940057940      f=frag1/24
↪
 O         -1.69701700000000         -2.04382200000000         -8.20675700000000      f=frag1/25
↪
 H         -1.16935910005967         -0.43265303835196         -8.83710685355905      f=frag1/25
↪
 H         -0.63663168422293         -2.33781858979711         -6.77122484449879      f=frag1/25
 O          5.30132200000000         -9.56047300000000          8.45223600000000      f=frag1/26
↪
 H          5.27386189469879         -8.61762538446407          6.90889108173823      f=frag1/26
↪
 H          4.55017557362690         -8.45544294990206          9.67137143822390      f=frag1/26
↪
 O          5.99489300000000         -5.84398700000000         -1.27866800000000      f=frag1/27
↪
 H          6.10499825242707         -5.94881824033588         -3.08103213209623      f=frag1/27
↪
 H          7.02298278120587         -4.42064928461433         -0.84422038296580      f=frag1/27
↪
 O         -8.49933400000000          6.30730300000000          1.26206900000000      f=frag1/28
↪
 H         -9.26414421853983          5.21634012956926          0.03875605345364      f=frag1/28
↪
 H         -9.84131776823032          6.72028388514297          2.40230140460829      f=frag1/28
↪
 O          1.42265800000000          7.15100500000000         -6.70245700000000      f=frag1/29
↪
 H          0.49123529007929          7.74531228135872         -5.27036768330312      f=frag1/29
↪
 H          2.87243479479048          6.33486591183760         -5.99276142000362      f=frag1/29
↪
 O          1.16572300000000        -15.86327300000000         -8.03011700000000      f=frag1/30
↪
 H         -0.43980652899404        -16.56646285956896         -8.47669081602252      f=frag1/30
↪
 H          1.27023658519214        -14.33069688850776         -8.98504681689730      f=frag1/30
↪
 O         -1.89194800000000        -10.63552100000000         -8.58469800000000      f=frag1/31
↪
 H         -2.78816305171149        -10.71751973024788        -10.15367983850320      f=frag1/31
↪
 H         -2.90779524204234        -11.59056143861734         -7.43250195607063      f=frag1/31
↪
 O          1.30322100000000         -6.54666700000000         -9.29066800000000      f=frag1/32
↪
 H          0.04144584195376         -7.80383112457906         -8.97588071525340      f=frag1/32
↪
 H          0.43997569188912         -4.97291783823648         -9.06767765025571      f=frag1/32
↪
 O         -4.10800400000000         -6.59387200000000        -16.43397700000000      f=frag1/33
↪
 H         -4.90090013771901         -8.11414568563407        -17.00992218470793      f=frag1/33
↪
 H         -5.44796070593591         -5.38061594011036        -16.36971704696261      f=frag1/33
↪
 O         -5.19983400000000        -12.82687600000000         -4.96878500000000      f=frag1/34
↪
 H         -6.70471369416354        -13.43327229448220         -5.76831602526536      f=frag1/34
↪
```

```
H        -5.42955949756766    -11.03545071340780     -4.87067679172327    f=frag1/34␣
↪
O        -4.89786700000000      5.50178800000000    -16.51771100000000    f=frag1/35␣
↪
H        -4.96675000682788      7.30587317882606    -16.40743398906903    f=frag1/35␣
↪
H        -3.22762417250174      5.07795982200470    -15.96792726530632    f=frag1/35␣
↪
O       -17.57898700000000     -4.99346700000000     -2.97681500000000    f=frag1/36␣
↪
H       -18.76387465651032     -3.84193458717524     -2.24085062321750    f=frag1/36␣
↪
H       -17.39866900954461     -6.30611916234463     -1.74554195748830    f=frag1/36␣
↪
O        -3.50776800000000    -17.32380200000000    -10.10442500000000    f=frag1/37␣
↪
H        -4.40193812130934    -18.63309464188228    -10.97496609167801    f=frag1/37␣
↪
H        -4.53952158473348    -16.94953791159138     -8.66671344522861    f=frag1/37␣
↪
O       -10.81911600000000     -4.66640800000000    -11.14487100000000    f=frag1/38␣
↪
H       -11.62185710527301     -4.88296610749300     -9.53852778505075    f=frag1/38␣
↪
H       -10.60915967583231     -2.87735761828481    -11.30864113107593    f=frag1/38␣
↪
O        -6.39205500000000      0.01793500000000     -6.73248500000000    f=frag1/39␣
↪
H        -5.82125006722311      1.22625909145182     -5.51356587267281    f=frag1/39␣
↪
H        -4.87489255123650     -0.72801711445480     -7.37542911048570    f=frag1/39␣
↪
O       -11.30505500000000      0.46379700000000     -8.25065700000000    f=frag1/40␣
↪
H        -9.64226007733910      0.80674572776125     -7.62685540050746    f=frag1/40␣
↪
H       -11.19896085378185      0.79109534466454    -10.02638879504226    f=frag1/40␣
↪
O         9.00392400000000      1.26046800000000    -13.97805200000000    f=frag1/41␣
↪
H         8.87294935217592      1.02947962110145    -12.18888481086687    f=frag1/41␣
↪
H         7.29424845398649      1.45338312957695    -14.53608144577292    f=frag1/41␣
↪
O       -10.89041700000000      3.43422700000000     -2.58901500000000    f=frag1/42␣
↪
H       -11.04828427051772      1.96350979597573     -3.63002318964871    f=frag1/42␣
↪
H       -12.48348293054753      3.56267021936991     -1.74211071819919    f=frag1/42␣
↪
O        -5.77532000000000     11.85364600000000     -3.10184700000000    f=frag1/43␣
↪
H        -7.42098370181530     11.26896732951829     -2.63113365512790    f=frag1/43␣
↪
H        -5.59494422717391     13.43644886209393     -2.24522382582680    f=frag1/43␣
↪
O        -1.56797700000000      3.41013100000000     -9.46461500000000    f=frag1/44␣
↪
```

```
H        -0.62647981010343     4.69229425860768    -8.60364482634544    f=frag1/44
↪
H        -1.22975840621731     3.70013224540235   -11.21764410491830    f=frag1/44
↪
O        -5.85114500000000     5.94789900000000    -8.75297600000000    f=frag1/45
↪
H        -4.36435430864595     4.93739949116572    -8.95290949638818    f=frag1/45
↪
H        -5.23356351746774     7.64390869587085    -8.63572496098009    f=frag1/45
↪
O        -5.55413300000000    -7.92360000000000     0.25554800000000    f=frag1/46
↪
H        -4.38326585687764    -9.28844448621082     0.45022435153192    f=frag1/46
↪
H        -6.01769107687214    -7.47995835111906     1.94667176216252    f=frag1/46
↪
O       -10.80147900000000     4.92460700000000   -18.42825600000000    f=frag1/47
↪
H        -9.46772203676955     4.11444445205056   -19.34278223208070    f=frag1/47
↪
H       -12.08285494971162     3.65684979597704   -18.27835293347214    f=frag1/47
↪
O        11.84765600000000     6.04652000000000    -5.60335700000000    f=frag1/48
↪
H        12.23822731048506     7.54282902057004    -4.66519785219186    f=frag1/48
↪
H        10.14515882225660     6.29374114330687    -6.16191644024222    f=frag1/48
↪
O         9.40177200000000    -2.16451100000000     0.06593400000000    f=frag1/49
↪
H         9.99192042697073    -0.60334687050214    -0.63128950443925    f=frag1/49
↪
H        10.90850263662718    -3.09972302317630     0.42194641911128    f=frag1/49
↪
O        -3.83766900000000    11.00009100000000    -8.55948700000000    f=frag1/50
↪
H        -3.74373964993300    10.83305937748681    -6.76090229682198    f=frag1/50
↪
H        -4.87440105609424    12.45821519867243    -8.82531410042130    f=frag1/50
↪
O        -8.92277200000000     7.51515500000000    -5.13005000000000    f=frag1/51
↪
H        -7.64240242456374     6.79630036125453    -6.18623699995818    f=frag1/51
↪
H        -9.98750814060132     6.11915279273326    -4.69517258001857    f=frag1/51
↪
O        -0.32879900000000     4.85643700000000   -14.28493900000000    f=frag1/52
↪
H        -0.20121901190276     6.61816864447670   -13.89551098390984    f=frag1/52
↪
H         1.10559652418602     4.51955612567933   -15.33404544784882    f=frag1/52
↪
O        -2.88119200000000   -21.16143900000000   -13.73773700000000    f=frag1/53
↪
H        -1.09646919425263   -20.88154858029820   -13.82748709656689    f=frag1/53
↪
H        -3.12787937393977   -22.76872743498338   -14.52980568926457    f=frag1/53
↪
```

```
O         4.08430100000000    -17.32350200000000     4.72137100000000    f=frag1/54
↪
H         4.02413151632988    -15.51826748869058     4.62578223161330    f=frag1/54
↪
H         2.36935484299032    -17.81891859379031     5.01313401917970    f=frag1/54
↪
O       -12.71856600000000      7.20558900000000     4.46079500000000    f=frag1/55
↪
H       -14.45999505747860      7.69416069947355     4.47955006523861    f=frag1/55
↪
H       -11.90975612966972      8.36888327939108     5.58515108034031    f=frag1/55
↪
O         3.64630900000000     -5.77729500000000    11.93616900000000    f=frag1/56
↪
H         4.69770672445712     -5.28069451760816    13.32165808553043    f=frag1/56
↪
H         1.98018929821852     -5.85215747794930    12.63620066423870    f=frag1/56
↪
O       -11.58290200000000     -8.00977000000000   -14.71744100000000    f=frag1/57
↪
H       -11.15452934215944     -6.96177583707747   -13.30682612671804    f=frag1/57
↪
H       -13.31014912674778     -7.60719902330076   -15.07262764680259    f=frag1/57
↪
O       -22.69131900000000     -4.60206700000000   -11.33851100000000    f=frag1/58
↪
H       -22.22272486043639     -6.12263420792208   -12.19869942051499    f=frag1/58
↪
H       -21.13544211851963     -3.70468130859725   -11.12508740068061    f=frag1/58
↪
O         4.56872600000000     15.36672100000000    -5.27290200000000    f=frag1/59
↪
H         6.23209824653603     14.72493681821289    -5.57775451137379    f=frag1/59
↪
H         4.56888064677390     17.00621625499974    -6.03687769799284    f=frag1/59
↪
O       -19.42024500000000      0.10101200000000   -13.92810800000000    f=frag1/60
↪
H       -18.31405236374814     -1.07169409540655   -13.10790833801040    f=frag1/60
↪
H       -20.71321177209865      0.43336205711680   -12.70770480925147    f=frag1/60
↪
O         4.02837000000000     -8.18071600000000   -24.03669300000000    f=frag1/61
↪
H         5.65512609076238     -8.96585900909256   -24.13073712086871    f=frag1/61
↪
H         3.08016870759219     -9.23647651818391   -22.91513710771192    f=frag1/61
↪
O       -16.59841600000000    -12.07983100000000    -3.02725000000000    f=frag1/62
↪
H       -16.86792190502099    -10.47849662259938    -2.23055432463633    f=frag1/62
↪
H       -17.75928521418718    -12.09663441135308    -4.41422653012776    f=frag1/62
↪
O        -7.26966600000000     11.01317800000000   -13.83432500000000    f=frag1/63
↪
H        -8.50533384743054     11.78916627619676   -12.76540199703817    f=frag1/63
↪
```

```
H          -5.99921818801193        12.28193782381027       -14.05299015201462        f=frag1/63
↪
O          14.92607800000000         0.72989200000000        -0.88069200000000        f=frag1/64
↪
H          14.68468379195669         0.44423157042895         0.88898485624517        f=frag1/64
↪
H          13.43262120167169         1.60465761841764        -1.40602569068362        f=frag1/64
↪
O         -10.24993000000000       -15.51193700000000       -12.56703500000000        f=frag1/65
↪
H         -10.81569785539303       -16.80205979146874       -11.43252226892721        f=frag1/65
↪
H          -8.81118264459048       -16.23360868276287       -13.39214081855071        f=frag1/65
↪
O           6.15377400000000        -6.81034900000000       -18.37876400000000        f=frag1/66
↪
H           5.66882347580422        -7.60131711703521       -16.82608191819916        f=frag1/66
↪
H           7.80239299356986        -6.14232206320171       -18.05104612139325        f=frag1/66
↪
O           3.76793000000000       -11.87319400000000       -12.00003600000000        f=frag1/67
↪
H           5.53017998300823       -12.15041382890993       -12.29878904864723        f=frag1/67
↪
H           3.52066686277755       -10.11552847274610       -12.34799636768586        f=frag1/67
↪
O           4.23865500000000         2.16969600000000       -16.37654900000000        f=frag1/68
↪
H           3.03592238765764         0.92692640056214       -16.90625382926868        f=frag1/68
↪
H           5.02860305993563         2.71732123822040       -17.90876707954152        f=frag1/68
↪
O         -14.68271000000000         0.72381400000000       -16.71344500000000        f=frag1/69
↪
H         -15.14615918038738        -0.55817879758533       -17.90228415832504        f=frag1/69
↪
H         -16.24034166467332         1.19193697588290       -15.92208663689043        f=frag1/69
↪
O          -8.56171900000000        -1.14046300000000       -21.11237400000000        f=frag1/70
↪
H          -7.24726382704255         0.10164744794682       -21.14397231666602        f=frag1/70
↪
H          -7.96722618972135        -2.45668030593841       -22.20129726831652        f=frag1/70
↪
O           3.09803900000000       -19.30437100000000        -4.72950400000000        f=frag1/71
↪
H           4.47592822929893       -18.53926720940844        -3.84199812304267        f=frag1/71
↪
H           2.31606384558652       -17.93639511851180        -5.61762585411266        f=frag1/71
↪
O         -11.48197100000000        -2.62773000000000         8.98577000000000        f=frag1/72
↪
H         -12.10181120734499        -1.96357292082485         7.42169752349214        f=frag1/72
↪
H         -12.53715834883977        -4.05824316030399         9.32015800703900        f=frag1/72
↪
O         -15.32683200000000         4.99378800000000         0.71520300000000        f=frag1/73
↪
```

```
H       -16.28231903578986      3.62492951609735      1.41155075487152    f=frag1/73
↪
H       -14.24707811603547      5.52532871717421      2.06546208560370    f=frag1/73
↪
O         4.69924600000000     -3.09331300000000    -22.21218100000000    f=frag1/74
↪
H         4.44004838777616     -3.02721933390031    -24.00105702850179    f=frag1/74
↪
H         4.90782404353437     -4.85303119199437    -21.84956458183150    f=frag1/74
↪
O         5.48125400000000    -14.87156900000000     -2.30008800000000    f=frag1/75
↪
H         6.04793062228805    -15.53700673074120     -0.71651626102270    f=frag1/75
↪
H         4.08123415431168    -13.81006617390298     -1.87025109596218    f=frag1/75
↪
O         7.71692700000000      7.49170500000000     -0.28146900000000    f=frag1/76
↪
H         7.15009959892487      7.47463483734514      1.43610036604224    f=frag1/76
↪
H         6.70446411622425      6.23931957052779     -1.10490887468917    f=frag1/76
↪
O        -0.09971700000000     10.17858400000000    -12.50337600000000    f=frag1/77
↪
H         1.43320771620038     10.52645688573944    -11.60852657687974    f=frag1/77
↪
H        -1.38648439654237     10.22688417289634    -11.23313966659599    f=frag1/77
↪
O         0.17403100000000    -13.35840700000000    -17.21967600000000    f=frag1/78
↪
H        -1.53885665094696    -13.32910704018195    -16.63934279586859    f=frag1/78
↪
H         1.16036533522916    -13.51667655796879    -15.71179705645900    f=frag1/78
↪
O       -12.87620000000000     -6.35854400000000      5.05140600000000    f=frag1/79
↪
H       -11.96560105641672     -7.83983990822539      5.54960396913384    f=frag1/79
↪
H       -12.07232354985510     -5.82614364999607      3.52106744189462    f=frag1/79
↪
O        10.37496800000000      3.01364200000000     -1.65565900000000    f=frag1/80
↪
H        10.56060903951036      4.59152033506358     -0.79109750165770    f=frag1/80
↪
H        10.50573068374942      3.43968959450338     -3.40865305303111    f=frag1/80
↪
O       -10.54672300000000     11.48736100000000     -1.14189100000000    f=frag1/81
↪
H       -10.87795175377343      9.85135127695166     -1.83858958546889    f=frag1/81
↪
H       -11.17355773718848     12.64365945215504     -2.38352527948959    f=frag1/81
↪
O        16.77188300000000      5.46660900000000     -7.08505700000000    f=frag1/82
↪
H        15.05944999981585      5.87188178668200     -6.66677909180415    f=frag1/82
↪
H        17.62236540790539      5.39342341222859     -5.49040119833247    f=frag1/82
↪
```

```
O         3.75457600000000    11.31207900000000    -9.30406600000000    f=frag1/83
↪
H         3.86003631554596    12.75207030857270    -8.21460174023312    f=frag1/83
↪
H         3.06160535139532    10.00588129699877    -8.26231611492555    f=frag1/83
↪
O       -12.29225900000000     3.56799200000000    15.55877400000000    f=frag1/84
↪
H       -11.55132238527440     2.03988079459175    14.93626432369441    f=frag1/84
↪
H       -13.88471345817920     3.05255666747083    16.24435317034552    f=frag1/84
↪
O        -6.10108600000000   -22.91748700000000    -8.30037600000000    f=frag1/85
↪
H        -4.39703335260680   -23.51961699551631    -8.22775875933953    f=frag1/85
↪
H        -6.81851364400339   -23.76013175020010    -9.73105799918286    f=frag1/85
↪
O       -16.47411100000000   -10.06716200000000   -10.99026800000000    f=frag1/86
↪
H       -16.57824909007337    -8.41061556718373   -11.70904362169910    f=frag1/86
↪
H       -14.81816261652035   -10.64777696598199   -11.42881650898706    f=frag1/86
↪
O       -19.79746500000000    -9.66763600000000    -6.29115500000000    f=frag1/87
↪
H       -19.30039040153150    -9.92412834081941    -8.01125899188181    f=frag1/87
↪
H       -20.69151106695596    -8.09537391433191    -6.30780271266956    f=frag1/87
↪
O        -5.17342500000000   -12.71280800000000   -15.39516900000000    f=frag1/88
↪
H        -6.29838943798551   -11.96053470711536   -16.59523946722675    f=frag1/88
↪
H        -5.88688186704289   -12.30005902553624   -13.78513137150979    f=frag1/88
↪
O        -0.98586800000000     2.69855900000000   -21.38983900000000    f=frag1/89
↪
H        -2.77204881462932     2.96779943896921   -21.48309619116062    f=frag1/89
↪
H        -0.80499110215822     1.20344697675802   -20.38807696838254    f=frag1/89
↪
O        -6.03694100000000     3.69212200000000   -20.94210000000000    f=frag1/90
↪
H        -6.14071267530565     5.16732160002454   -21.98356377744071    f=frag1/90
↪
H        -5.38953086150776     4.29598484602752   -19.36481963241442    f=frag1/90
↪
O         8.56921900000000    -3.79464900000000     6.33985500000000    f=frag1/91
↪
H         9.50635468663971    -2.25984004884893     6.53419338664101    f=frag1/91
↪
H         9.73243046417235    -5.09792887162821     6.80890110437856    f=frag1/91
↪
O         1.62911600000000    13.17414900000000    -1.60348400000000    f=frag1/92
↪
H         2.66594110342472    13.03446639023091    -0.12798012211089    f=frag1/92
↪
```

```
H          2.66565709768483      14.04258475743428      -2.80473562652369      f=frag1/92
↪
O         -1.79383000000000      17.03966900000000      -2.19674400000000      f=frag1/93
↪
H         -0.53200588641356      15.75727680306069      -2.00993024293071      f=frag1/93
↪
H         -1.42725302708876      17.80645974200545      -3.79338324012655      f=frag1/93
↪
O         -7.66877400000000     -18.37494100000000      -6.40129600000000      f=frag1/94
↪
H         -8.48426657117146     -18.61293045852318      -4.80443592340425      f=frag1/94
↪
H         -7.09966518295817     -20.02669116913405      -6.86974035671357      f=frag1/94
↪
O         10.40397100000000      -3.09718200000000     -22.16627000000000      f=frag1/95
↪
H          8.72972380889992      -2.49326115967796     -21.84408188422351      f=frag1/95
↪
H         11.11812043738098      -1.87363862919898     -23.29078579938355      f=frag1/95
↪
O        -15.30392700000000       1.38087600000000       7.30094900000000      f=frag1/96
↪
H        -14.62180203945480       2.68484739436449       8.35261118963587      f=frag1/96
↪
H        -16.43220546537447       0.47060667386509       8.38261486369176      f=frag1/96
↪
O          8.58612500000000      -8.60056200000000      -9.05727500000000      f=frag1/97
↪
H          8.70489512259771      -9.81319946167069     -10.39407463523348      f=frag1/97
↪
H          8.62856020038021      -9.58970781613463      -7.54354008582316      f=frag1/97
↪
O         16.03912900000000       1.51573000000000     -10.24266300000000      f=frag1/98
↪
H         15.15629413827772       2.19658252062845     -11.66697745696204      f=frag1/98
↪
H         16.44244740848228       2.95789688018386      -9.22822101876796      f=frag1/98
↪
O        -15.02490500000000       4.61696500000000      -7.22941300000000      f=frag1/99
↪
H        -13.96280647503120       5.31092864978721      -8.51859506482347      f=frag1/99
↪
H        -14.28853929511416       3.00824288473906      -6.85340464006069      f=frag1/99
↪
O         -3.38051600000000     -15.91202400000000      -1.00061000000000      f=frag1/
↪100
H         -4.22410184376431     -15.04185487754888      -2.34329307606914      f=frag1/
↪100
H         -1.90382516434679     -16.61447063748981      -1.77362777442889      f=frag1/
↪100
O        -15.26048000000000       6.49459500000000     -20.09531600000000      f=frag1/
↪101
H        -13.52897204979343       6.24438887420643     -19.63607060457812      f=frag1/
↪101
H        -15.47768447407662       5.50940621247909     -21.59659174135803      f=frag1/
↪101
O        -18.03907100000000      -8.43103500000000       0.81066000000000      f=frag1/
↪102
```

```
H      -19.83364526460744    -8.64920488317246    0.75120975933977    f=frag1/
↪102
H      -17.60426054198021    -8.72510837516712    2.54157429534231    f=frag1/
↪102
O       -1.66133700000000   -24.85433900000000   -7.59562300000000    f=frag1/
↪103
H       -0.93912195987639   -26.25608107787563   -6.70953395077215    f=frag1/
↪103
H       -2.16378824805376   -25.53854670810930   -9.19280994513164    f=frag1/
↪103
O      -13.11411600000000    4.62406100000000    10.65633500000000    f=frag1/
↪104
H      -12.77074621481150    4.11089532103550    12.35644893641187    f=frag1/
↪104
H      -11.48603903104276    4.71705763077764     9.87382583582304    f=frag1/
↪104
O      -12.41558900000000   11.84804700000000    -6.99447300000000    f=frag1/
↪105
H      -11.30168153595745   10.44978337527290    -6.71933990939709    f=frag1/
↪105
H      -12.47123642287374   12.03565896451106    -8.79261334416275    f=frag1/
↪105
O       -6.99029300000000   15.09813200000000    -9.14840700000000    f=frag1/
↪106
H       -8.45117315114006   15.70172502049625   -10.02768731357616    f=frag1/
↪106
H       -7.14889401470228   15.78429001622505    -7.48238284738053    f=frag1/
↪106
O      -12.34625400000000   10.56150500000000   -12.23873500000000    f=frag1/
↪107
H      -11.44958979728050    8.99509889893791   -12.35703315867646    f=frag1/
↪107
H      -12.49821882155289   11.12665028131793   -13.95020252415008    f=frag1/
↪107
O       11.40803400000000   -1.03651800000000    -9.30311700000000    f=frag1/
↪108
H       13.12420580954423   -0.46728539979694    -9.35167405119891    f=frag1/
↪108
H       11.31625950576437   -2.30813269496276   -10.58615133000387    f=frag1/
↪108
O       -5.11896700000000   -0.12253200000000    11.02394500000000    f=frag1/
↪109
H       -3.92681627947473    1.22947829217064    10.87403152613883    f=frag1/
↪109
H       -4.37657440580777   -1.48587544764282    10.09563930570322    f=frag1/
↪109
O       -9.99475100000000   -0.33650000000000    13.26488800000000    f=frag1/
↪110
H       -8.34486345702615    0.24573595608403    12.80608345859702    f=frag1/
↪110
H      -10.59609456116402   -1.19896273277279    11.79310420533769    f=frag1/
↪110
O       -2.30216800000000   10.69499300000000     2.99722800000000    f=frag1/
↪111
H       -1.93606116560115   10.01830345774807     1.36025326914961    f=frag1/
↪111
H       -4.04923519757226   11.15628108452666     2.91615338962990    f=frag1/
↪111
```

```
O          13.02295800000000     -5.27441800000000      1.23847500000000     f=frag1/
↪112
H          14.31033554297163     -4.63482922705865      0.14065238973499     f=frag1/
↪112
H          13.91152105495786     -5.78567177035616      2.72866923628817     f=frag1/
↪112
O           7.56659100000000      6.94211200000000     -9.52506900000000     f=frag1/
↪113
H           8.22168391057836      5.87923316520027    -10.83379780356249     f=frag1/
↪113
H           6.67225115956347      8.24497386381330    -10.40501831435555     f=frag1/
↪113
O          -2.17755000000000      7.03555800000000      6.40129900000000     f=frag1/
↪114
H          -2.96551413318437      5.65850851485069      5.53269295552737     f=frag1/
↪114
H          -2.21627359256819      8.40062348976667      5.21525504034563     f=frag1/
↪114
O           9.00788300000000     12.60897400000000     -5.62029800000000     f=frag1/
↪115
H          10.26246081511818     11.86068291737544     -4.55365645678727     f=frag1/
↪115
H           7.97453521284851     11.22086747860001     -6.14656429973985     f=frag1/
↪115
O          -1.44785500000000     -3.72480700000000    -24.99109500000000     f=frag1/
↪116
H           0.23683814213660     -4.37145767071870    -25.11471193705296     f=frag1/
↪116
H          -1.92063360111663     -4.01890345598422    -23.27016756076567     f=frag1/
↪116
O          -7.64825200000000     12.61552300000000      3.06517700000000     f=frag1/
↪117
H          -7.81167293245439     14.38063407144895      3.42476805177055     f=frag1/
↪117
H          -8.83689943470846     12.31223844466806      1.73598952764555     f=frag1/
↪117
O           4.46933200000000     -0.50378800000000      2.21087700000000     f=frag1/
↪118
H           4.27357465941304     -2.08510082915893      3.06688986215400     f=frag1/
↪118
H           5.91907845739645     -0.73702803715971      1.15473407570514     f=frag1/
↪118
O          -1.24895400000000     -6.97750900000000     -2.93109400000000     f=frag1/
↪119
H          -2.37833578618925     -6.59579569242172     -1.57079221286929     f=frag1/
↪119
H          -0.21288275291882     -5.50308282339477     -3.08674554325879     f=frag1/
↪119
O          -5.92082900000000     -3.03107100000000     -2.51072800000000     f=frag1/
↪120
H          -5.78071219226800     -4.56089081895937     -3.46553050415439     f=frag1/
↪120
H          -6.21267678923423     -1.74791100910179     -3.75166671237991     f=frag1/
↪120
O           1.14317500000000     -2.76826300000000     -3.93086600000000     f=frag1/
↪121
H           2.94971664531870     -2.78102101161944     -3.84213191918490     f=frag1/
↪121
```

```
 H          0.62850650734009      -1.73939210305163      -2.53510649267729      f=frag1/
↪121
 O         -5.21546600000000      -0.71679400000000       1.97558900000000      f=frag1/
↪122
 H         -5.88193806317475      -1.35755002215181       0.42095852001025      f=frag1/
↪122
 H         -5.82594782496319      -1.86151827602300       3.23595476257813      f=frag1/
↪122
 O         -5.42031600000000       3.66296900000000       4.47881900000000      f=frag1/
↪123
 H         -6.47734991529745       4.75371897638942       3.49669222043088      f=frag1/
↪123
 H         -5.28387721038046       2.15431400285173       3.49041213110813      f=frag1/
↪123
 O          4.64978600000000       3.67542600000000      -3.78351700000000      f=frag1/
↪124
 H          5.30464821970122       2.16136749204504      -4.52541424890080      f=frag1/
↪124
 H          3.40016650933066       3.10039840282737      -2.60903925511290      f=frag1/
↪124
 O          1.38687600000000      -3.56100200000000       5.14710300000000      f=frag1/
↪125
 H          2.35586583709350      -2.84555947523344       6.49648430622687      f=frag1/
↪125
 H          2.25995224007779      -5.08706726956497       4.72228980076397      f=frag1/
↪125
 O         -3.30635200000000      -3.51075400000000       7.52143300000000      f=frag1/
↪126
 H         -1.82144215646564      -3.75776796826252       6.51860894948698      f=frag1/
↪126
 H         -4.64977283415043      -4.18421132786886       6.51481526117547      f=frag1/
↪126
 O          5.96480100000000       4.51370300000000       3.70332800000000      f=frag1/
↪127
 H          5.24291272273548       3.06620845434369       2.89384293177905      f=frag1/
↪127
 H          4.73614594944492       5.00201400735317       4.93765482424434      f=frag1/
↪127
END

FDE
  PW91K
END

eor
```

Example: FDE freeze-and-thaw: HeCO2

Download FDE_HeCO2_freezeandthaw.run

```
#!/bin/sh

# This example demonstrates how a freeze-and-thaw FDE calculation can be
# performed. As test system, a He-CO2 van der Waals complex is used. It will
# further be shown how different exchange-correlation potential can be used for
# different subsystems, and how different basis set expansions can be employed.
# For details, see C.R. Jacob, T.A. Wesolowski, L. Visscher, J. Chem. Phys. 123
```

```
# (2005), 174104. It should be stressed that the basis set and integration grid
# used in this example are too small to obtain good results.

# Summary:
# - PW91 everywhere
# - SAOP for He; PW91 for CO2
# - FDE(s) calculation with PW91 everywhere

# Important, this kind of FDE geometry optimization only works with the non-
# default STO pair fitting method. This means that one has to include the key
# STOFIT in the input for ADF. If one does not use the BASIS key, like in this
# example, one should also include the key STOFIT in all fragment calculations
# (also in the create runs).


# Create atom


$ADFBIN/adf  -n1  <<eor
Create C   $ADFRESOURCES/TZ2P/C
STOFIT
eor
mv TAPE21 t21.C
rm logfile

$ADFBIN/adf  -n1  <<eor
Create O   $ADFRESOURCES/TZ2P/O
STOFIT
eor
mv TAPE21 t21.O
rm logfile

$ADFBIN/adf  -n1  <<eor
Create He   $ADFRESOURCES/TZ2P/He
STOFIT
eor
mv TAPE21 t21.He
rm logfile


# == PW91 everywhere ==

# In the first part, the PW91 functional will be used for both the He and the
# CO2 subsystems. In this part, the FDE(m) basis set expansion is used, i.e.,
# basis functions of the frozen subsystem are not included in the calculation of
# the nonfrozen subsystem.

# First, the CO2 molecule is prepared. In this calculation, the C2v symmetry of
# the final complex is used, and the NOSYMFIT option has to be included because
# this molecule will be rotated as a frozen fragment.


############################
# Preparation of frozen CO2
############################

$ADFBIN/adf   <<eor
Title TEST 1 -- Preparation of frozen CO2
```

```
STOFIT

EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END

NOPRINT BAS FUNCTIONS

Units
 Length Bohr
end

Atoms
C         0.000000  0.000000  0.000000
O        -2.192000  0.000000  0.000000
O         2.192000  0.000000  0.000000
end

Symmetry C(2V)
NOSYMFIT

Fragments
 C  t21.C
 O  t21.O
End

NumericalQuality Good

xc
 GGA pw91
end

eor

mv TAPE21 t21.co2.0
rm logfile


# Afterwards, the FDE calculation is performed. In this calculation, the He atom
# is the nonfrozen system, and the previously prepared CO2 molecule is used as
# frozen fragment. For this frozen fragment the RELAX option is specified, so
# that the density of this fragment is updated in freeze-and-thaw iteration (a
# maximum number of three iteration is specified).

########################
# Embedding calculation
########################


$ADFBIN/adf   <<eor
Title TEST 1 -- Embedding calulation: He with frozen CO2 density -- freeze-and-thaw
STOFIT

EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END
```

```
NOPRINT BAS FUNCTIONS

Units
 Length Bohr
end

Atoms
 He    0.000000   0.000000   6.019000 f=He
 C     0.000000   0.000000   0.000000 f=co2
 O    -2.192000   0.000000   0.000000 f=co2
 O     2.192000   0.000000   0.000000 f=co2
end

Fragments
 He    t21.He
 co2   t21.co2.0  type=fde   &
    fdeoptions RELAX
 SubEnd
End

NOSYMFIT

NumericalQuality Good

xc
 GGA pw91
end

FDE
 PW91K
 FULLGRID
 RELAXCYCLES 3
end

eor

rm TAPE21
rm logfile


# == SAOP for He; PW91 for CO2 ==

# In this second part, the above example is modified such that PW91 is employed
# for the CO2 subsystem, while the SAOP potential is used for He. This can be
# achieved by choosing SAOP in the XC key (this sets the functional that will be
# used for the nonfrozen subsystem). Additionally, for the frozen fragment the
# XC option is used to chose the PW91 functional for relaxing this fragment.
# Furthermore, the PW91 functional is chosen for the nonadditive exchange-
# correlation functional that is used in the embedding potential with the
# GGAPOTXFD and GGAPOTCFD options in the FDE key.


########################
# Embedding calculation
########################

$ADFBIN/adf    <<eor
```

```
Title TEST 2 -- Embedding calulation: He with frozen CO2 density -- freeze-and-thaw
STOFIT

EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END

NOPRINT BAS FUNCTIONS

Units
 Length Bohr
end

Atoms
 He   0.000000  0.000000  6.019000 f=He
 C    0.000000  0.000000  0.000000 f=co2
 O   -2.192000  0.000000  0.000000 f=co2
 O    2.192000  0.000000  0.000000 f=co2
end

Fragments
 He   t21.He
 co2  t21.co2.0  type=fde   &
   fdeoptions RELAX
   XC          GGA PW91
 SubEnd
End

NOSYMFIT

NumericalQuality Good

xc
 MODEL SAOP
end

FDE
 PW91K
 FULLGRID
 XCNADD PW91
 RELAXCYCLES 3
end

eor

rm TAPE21 t21.co2.0
rm logfile


# == FDE(s) calculation with PW91 everywhere ==

# In this third part, the PW91 functional is applied for both subsystems again,
# but in contrast to part 1, now the FDE(s) basis set expansion is used, i.e.,
# the basis functions of the frozen subsystem are included in the calculation of
# the nonfrozen subsystem. This can be achieved by employing the USEBASIS
# option. This option can be combined with the RELAX option.
```

```
###########################
# Preparation of frozen CO2
###########################


$ADFBIN/adf   <<eor
Title TEST 3 -- Preparation of frozen CO2
STOFIT

EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END

NOPRINT BAS FUNCTIONS

Units
 Length Bohr
end

Atoms
C         0.000000  0.000000  0.000000
O        -2.192000  0.000000  0.000000
O         2.192000  0.000000  0.000000
end

Symmetry C(2V)
NOSYMFIT

Fragments
 C  t21.C
 O  t21.O
End

NumericalQuality Good

xc
 GGA pw91
end

eor

mv TAPE21 t21.co2.0
rm logfile

#######################
# Embedding calculation
#######################

$ADFBIN/adf   <<eor
Title TEST 3 -- Embedding calulation: He with frozen CO2 density -- freeze-and-thaw
STOFIT

EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END
```

```
NOPRINT BAS FUNCTIONS

Units
 Length Bohr
end

Atoms
 He   0.000000  0.000000  6.019000 f=He
 C    0.000000  0.000000  0.000000 f=co2
 O   -2.192000  0.000000  0.000000 f=co2
 O    2.192000  0.000000  0.000000 f=co2
end

Fragments
 He   t21.He
 co2  t21.co2.0  type=fde   &
   fdeoptions RELAX USEBASIS
 SubEnd
End

NOSYMFIT

NumericalQuality Good

xc
 GGA pw91
end

FDE
 PW91K
 FULLGRID
 RELAXCYCLES 3
end

eor

rm TAPE21
rm logfile
```

Example: FDE energy: NH3-H2O

Download FDE_Energy_NH3-H2O.run

```
#! /bin/sh

# This is example for a calculation of FDE interaction energies in ADF in case
# of closed shell fragments#

# It performs single point runs for H2O and NH3
# with LDA/DZ (all-electron) and uses these fragments in
# A) an FDE energy embedding calculation calculation in which the energy of
#    water in presence of a frozen ammonia is computed
#    This requires a supermolecular integration grid
# B) a fully variational FDE energy calculation (with freeze-and-thaw)
#
# ZlmFit quality needs to be at least Good and
```

```
# Becke Integration quality is good which should give total energies for the
# fragments accurate at least up to 10**(-4) atomic units


# NH3 LDA/DZ


"$ADFBIN/adf" <<EOF
Title H2O LDA/DZ single point

 ATOMS
        N      -1.51248      -0.03714      -0.00081
        H      -1.71021       0.95994      -0.11003
        H      -1.96356      -0.53831      -0.76844
        H      -1.92899      -0.35123       0.87792
 END

 UNITS
   Length Angstrom
   Angle Degree
 END

 CHARGE  0

 SYMMETRY tol=1e-2

 BASIS
   Type DZ
   Core None
 END

 XC
   LDA
 END

 ZlmFit
   Quality good
 END

 BeckeGrid
   Quality good
 END

 SCF
   iterations  100
   converge 1.0e-06 1.0e-06
 END

 EXACTDENSITY

 EPRINT
   SFO NOEIG NOOVL NOORBPOP
   SCF NOPOP
 END

 NOPRINT BAS FUNCTIONS

 NOSYMFIT
```

```
EOF

rm logfile
mv TAPE21 t21.ammonia


# H2O LDA/DZ


"$ADFBIN/adf" <<EOF
Title H2O LDA/DZ single point

 ATOMS
        O        1.45838        0.10183        0.00276
        H        0.48989       -0.04206        0.00012
        H        1.84938       -0.78409       -0.00279
 END

 UNITS
   Length Angstrom
   Angle Degree
 END

 CHARGE  0

 SYMMETRY tol=1e-2

 BASIS
   Type DZ
   Core None
 END

 XC
   LDA
 END

 ZlmFit
   Quality good
 END

 BeckeGrid
   Quality good
 END

 SCF
   iterations  100
   converge 1.0e-06 1.0e-06
 END

 EXACTDENSITY

 EPRINT
   SFO NOEIG NOOVL NOORBPOP
   SCF NOPOP
 END

 NOPRINT BAS FUNCTIONS
```

```
 NOSYMFIT

EOF
rm logfile
mv TAPE21 t21.water


# NH3-H2O: FDE energy of H2O in presence of frozen NH3


"$ADFBIN/adf" <<EOF
Title NH3-H2O LDA/Thomas-Fermi/DZ FDE single point with interaction energy

 ATOMS
        O        1.45838       0.10183        0.00276    f=frag1
        H        0.48989      -0.04206        0.00012    f=frag1
        H        1.84938      -0.78409       -0.00279    f=frag1
        N       -1.51248      -0.03714       -0.00081    f=frag2
        H       -1.71021       0.95994       -0.11003    f=frag2
        H       -1.96356      -0.53831       -0.76844    f=frag2
        H       -1.92899      -0.35123        0.87792    f=frag2
 END

 UNITS
   Length Angstrom
   Angle Degree
 END

 CHARGE  0

 SYMMETRY tol=1e-2

 FRAGMENTS
  frag1  t21.water
  frag2  t21.ammonia type=FDE
 END

 XC
   LDA
 END

 ZlmFit
   Quality good
 END

 BeckeGrid
   Quality good
 END

 SCF
   iterations  100
   converge 1.0e-06 1.0e-06
 END

 EXACTDENSITY

 EPRINT
```

```
   SFO NOEIG NOOVL NOORBPOP
   SCF NOPOP
 END

 NOPRINT BAS FUNCTIONS

 FDE
   THOMASFERMI
   FULLGRID
   ENERGY
 END

EOF
rm TAPE21 logfile


# NH3-H2O FDE interaction energy


"$ADFBIN/adf" <<EOF
Title NH3-H2O LDA/Thomas-Fermi/DZ FDE single point with interaction energy

 ATOMS
        O         1.45838          0.10183          0.00276     f=frag1
        H         0.48989         -0.04206          0.00012     f=frag1
        H         1.84938         -0.78409         -0.00279     f=frag1
        N        -1.51248         -0.03714         -0.00081     f=frag2
        H        -1.71021          0.95994         -0.11003     f=frag2
        H        -1.96356         -0.53831         -0.76844     f=frag2
        H        -1.92899         -0.35123          0.87792     f=frag2
 END

 UNITS
   Length Angstrom
   Angle Degree
 END

 CHARGE   0

 SYMMETRY tol=1e-2

 FRAGMENTS
  frag1  t21.water
  frag2  t21.ammonia type=FDE  &
     fdeoptions RELAX
  SubEnd
 END

 XC
   LDA
 END

 ZlmFit
   Quality good
 END

 BeckeGrid
   Quality good
```

```
 END

 SCF
   iterations  100
   converge 1.0e-06 1.0e-06
 END

 EXACTDENSITY

 EPRINT
   SFO NOEIG NOOVL NOORBPOP
   SCF NOPOP
 END

 NOPRINT BAS FUNCTIONS

 FDE
   THOMASFERMI
   RELAXCYCLES 3
   ENERGY
 END

EOF
rm TAPE21 logfile t21.water t21.ammonia
```

Example: FDE energy: unrestricted fragments: Ne-H2O

Download FDE_Energy_H2O-Ne_unrestricted.run

```
#! /bin/sh

# This is example for a calculation of FDE interaction energies in ADF for an
# open-shell frozen fragment.

# It performs single point runs for H2O and Ne, the latter unrestricted
# with LDA/DZ (all-electron) and uses these fragments in an FDE energy
# embedding calculation in which the energy of water in presence of a frozen
# (open-shell) neon atom is computed
# This is a bit of an artificial test but it serves its purpose
#
# No freeze-thaw is done, this is at present not possible with unrestricted
# (open shell) fragments, but has to be done manually, see the example in
# $ADFHOME/examples/Test/FDE_Spin.run
#
# ZlmFit quality needs to be at least Good and
# Becke Integration quality is good which should give total energies for the
# fragments accurate at least up to 10**(-4) atomic units
#
# This test has been checked to yield the same energy as a run with a closed-
# shell (restricted) Ne atom (just comment UNRESTRICTED in the input below)
#

# Ne LDA/DZ (unrestricted)

"$ADFBIN/adf" <<EOF
Title Ne LDA/DZ single point, unrestricted
```

```
 ATOMS
       Ne     -1.51248     -0.03714     -0.00081
 END

 UNITS
   Length Angstrom
   Angle Degree
 END

 CHARGE  0

 UNRESTRICTED

 SYMMETRY tol=1e-2

 BASIS
   Type DZ
   Core None
 END

 XC
   LDA
 END

 ZlmFit
   Quality good
 END

 BeckeGrid
   Quality good
 END

 SCF
   iterations  100
   converge 1.0e-06 1.0e-06
 END

 EXACTDENSITY

 EPRINT
   SFO NOEIG NOOVL NOORBPOP
   SCF NOPOP
 END

 NOPRINT BAS FUNCTIONS

 NOSYMFIT

EOF

rm logfile
mv TAPE21 t21.ne


# H2O LDA/DZ


"$ADFBIN/adf" <<EOF
```

```
Title H2O LDA/DZ single point

 ATOMS
        O        1.45838       0.10183       0.00276
        H        0.48989      -0.04206       0.00012
        H        1.84938      -0.78409      -0.00279
 END

 UNITS
   Length Angstrom
   Angle Degree
 END

 CHARGE  0

 SYMMETRY tol=1e-2

 BASIS
   Type DZ
   Core None
 END

 XC
   LDA
 END

 ZlmFit
   Quality good
 END

 BeckeGrid
   Quality good
 END

 SCF
   iterations  100
   converge 1.0e-06 1.0e-06
 END

 EXACTDENSITY

 EPRINT
   SFO NOEIG NOOVL NOORBPOP
   SCF NOPOP
 END

 NOPRINT BAS FUNCTIONS

 NOSYMFIT

EOF
rm logfile
mv TAPE21 t21.water


# NH3-H2O: FDE energy of H2O in presence of frozen (unrestricted) Ne
```

```
"$ADFBIN/adf" <<EOF
Title Ne-H2O LDA/Thomas-Fermi/DZ FDE single point with interaction energy

 ATOMS
        O       1.45838        0.10183        0.00276    f=frag1
        H       0.48989       -0.04206        0.00012    f=frag1
        H       1.84938       -0.78409       -0.00279    f=frag1
        Ne     -1.51248       -0.03714       -0.00081    f=frag2
 END

 UNITS
   Length Angstrom
   Angle Degree
 END

 CHARGE  0

 SYMMETRY tol=1e-2

 FRAGMENTS
  frag1  t21.water
  frag2  t21.ne type=FDE
 END

 XC
   LDA
 END

 ZlmFit
   Quality good
 END

 BeckeGrid
   Quality good
 END

 SCF
   iterations  100
   converge 1.0e-06 1.0e-06
 END

 EXACTDENSITY

 EPRINT
   SFO NOEIG NOOVL NOORBPOP
   SCF NOPOP
 END

 NOPRINT BAS FUNCTIONS

 FDE
   THOMASFERMI
   FULLGRID
   ENERGY
 END

EOF
rm TAPE21 logfile
```

Example: FDE geometry optimization: H2O-Li(+)

Download GO_FDE_H2O-Li.run

```sh
#!/bin/sh

# This example demonstrates how to perform a geometry optimization of an active
# subsystem (a lithium atom) in the presence of a frozen FDE fragment (the water
# molecule).

# Notes:
#
# a) Only the geometry of the active fragment will be relaxed (in this case, the
#    Li atom). The geometry of the FDE fragment (the water molecule) will not
#    change during the optimization. If you want to relax the geometry of the FDE
#    fragment as well, see the keyword 'FDEOption Optimize'
#
# b) When performing geometry optimization ICW FDE one must:
#    - Disable symmetry ('Symmetry None')
#    - Use the old optimizer branch ('Geometry => Branch Old')
#    - Use Cartesian coordinates in the optimization procedure
#      ('Geometry => Optim Cartesian')
#    - It is advised to use the FullGrid option ('FDE => FullGrid')




# Prepare the water molecule fragment:
# ====================================

$ADFBIN/adf <<eor
  Atoms
    O         0.0000000    0.0000000   -1.53403200
    H        -0.7782160    0.0000000   -2.13596600
    H         0.7782160    0.0000000   -2.13596600
  End

  Symmetry NOSYM

  Basis
    type DZP
    core None
  End

  XC
    LDA
  End

  NumericalQuality Good
  TotalEnergy
eor

test -f TAPE21 && mv TAPE21 t21.iso.rho2


# H2O-Li(+) Geometry optimization
# (Li optimization in the presence of a 'frozen' water molecule)
# ==============================================================
```

```
$ADFBIN/adf <<eor
  Atoms
    LI         0.0000000    0.0000000     0.4529560
    O          0.0000000    0.0000000    -1.5340320   f=rho2
    H         -0.7782160    0.0000000    -2.1359660   f=rho2
    H          0.7782160    0.0000000    -2.1359660   f=rho2
  End

  Symmetry NOSYM

  Basis
    type DZP
    core None
  End

  XC
    LDA
  End

  Geometry
    Optim Cartesian
    Branch Old
    Converge E=1.0E-04 grad=1.0E-05
  End

  Charge 1 0
  NumericalQuality Good
  TotalEnergy

  Fragments
    rho2 t21.iso.rho2 type=fde &
    SubEnd
  End

  FDE
    ThomasFermi
    SDFTEnergy
    FullGrid
  End
eor

test -f TAPE21 && mv TAPE21 GO_FDE_H2O-Li.t21

echo "Final Geometry"
echo ""
echo "Final Geometry after FDE optimization"

$ADFBIN/adfreport GO_FDE_H2O-Li.t21 'distances#labels'
```

Example: Geometry optimization ICW FDE/sSDFT

Download GO_SDFT_H2O_2.run

```
#!/bin/sh

# This example demonstrates how to perform a FDE geometry optimization (both
# active fragment and FDE fragments are optimized)
```

```
# Notes:
#
# a) The key 'FDEOptions Optimize' triggers the optimization of the FDE fragment
#
# b) When performing geometry optimization ICW FDE one must:
#    - Disable symmetry ('Symmetry None')
#    - Use the old optimizer branch ('Geometry => Branch Old')
#    - Use Cartesian coordinates in the optimization procedure
#      ('Geometry => Optim Cartesian')
#    - It is advised to use the FullGrid option ('FDE => FullGrid')

# Prepare the water molecule fragment:
# =================================

$ADFBIN/adf <<eor
  Atoms
    O        2.06777224800       0.02791802500       0.00097906200
    H        2.44439386300      -0.41046204100       0.76710344400
    H        2.44003656500      -0.44829495000      -0.74442344000
  End

  Symmetry NOSYM

  BASIS
    type DZP
    core NONE
  End

  XC
    LDA
  End

  TotalEnergy
eor

test -f TAPE21 && mv TAPE21 t21.iso.rho2

# H2O-H2O Geometry optimization (both active and FDE fragments will be optimized)
# ===============================================================================

$ADFBIN/adf <<eor
  Atoms
    O       -0.70219605400      -0.05606025600       0.00994226200
    H       -1.02219322400       0.84677578200      -0.01148871400
    H        0.25752106200       0.04212149600       0.00521899900
    O        2.06777224800       0.02791802500       0.00097906200    f=rho2
    H        2.44439386300      -0.41046204100       0.76710344400    f=rho2
    H        2.44003656500      -0.44829495000      -0.74442344000    f=rho2
  End

  Symmetry NOSYM

  BASIS
    type DZP
    core NONE
  End
```

```
  XC
    LDA
  End

  Geometry
    Branch Old
    Optim Cartesian
    converge E=1.0E-04 grad=1.0E-04
  End

  NumericalQuality Normal
  TotalEnergy

  FRAGMENTS
    rho2 t21.iso.rho2 type=fde &
      FDEOptions Optimize
    SubEnd
  End

  FDE
    ThomasFermi
    SDFTEnergy
    FullGrid
    RelaxCycles 20
  End
eor

test -f TAPE21 && mv TAPE21 GO_SDFT_H2O_2.t21

echo "Final Geometry"
echo ""
echo "Final geometry after FDE optimization"

$ADFBIN/adfreport GO_SDFT_H2O_2.t21 'distances#labels'
```

Example: FDE NMR shielding: Acetonitrile in water

Download FDE_NMR_relax.run

```
#! /bin/sh



# This examples demonstrates both the calculation of NMR shieldings using FDE,
# and how the approximate environment density can be improved by partial
# relaxation of individual solvent molecules. The test system is a cluster of
# acetonitrile and 12 solvent water molecules, of which for two the densities
# are relaxed, while for the remaining 10 the frozen density of the isolated
# water is used. For details, see Refs. C. R. Jacob, J. Neugebauer, and L.
# Visscher, A flexible implementation of frozendensity embedding for use in
# multilevel simulation, submitted, 2007. R. E. Bulo, Ch. R. Jacob, and L.
# Visscher, NMR Solvent Shifts of Acetonitrile from Frozen-Density Embedding
# Calculation


# First, the isolated solvent water molecule is prepared. Again, because this
# will be rotated and translated afterwards, the option NOSYMFIT has to be
# included.
```

```
# create atomic fragment files

$ADFBIN/adf -n 1 <<eor
create O file=$ADFRESOURCES/DZP/O
eor
mv TAPE21 t21.O.DZP
rm logfile

$ADFBIN/adf -n 1 <<eor
create H file=$ADFRESOURCES/DZP/H
eor
mv TAPE21 t21.H.DZP
rm logfile

$ADFBIN/adf -n 1 <<eor
create C file=$ADFRESOURCES/DZP/C
eor
mv TAPE21 t21.C.DZP
rm logfile

$ADFBIN/adf -n 1 <<eor
create N file=$ADFRESOURCES/DZP/N
eor
mv TAPE21 t21.N.DZP
rm logfile

############################
# prepare H2O
############################

$ADFBIN/adf <<eor
Title Input generated by PyADF and later modified

 ATOMS
     O       -1.46800        2.60500        1.37700
     H       -0.95200        3.29800        0.96500
     H       -1.16100        1.79900        0.96100
 END

 SYMMETRY NOSYM tol=1e-2

 FRAGMENTS
   H       t21.H.DZP
   O       t21.O.DZP
 END

 XC
   LDA
 END

 SCF
   iterations  100
   converge 1.0e-06 1.0e-06
 END

 EPRINT
   SFO NOEIG NOOVL NOORBPOP
```

```
   SCF NOPOP
 END

 NOPRINT BAS FUNCTIONS
 NUMERICALQUALITY GOOD

eor

mv TAPE21 t21.h2o


# Afterwards, the FDE calculation is performed. In addition to the nonfrozen
# acetonitrile molecule, three different fragments are used for the solvent
# water molecules. The first two fragments frag1 and frag2 are relaxed (in up to
# two freeze-and-thaw cycles), while the third fragment is used for the
# remaining 10 solvent molecules. Since a calculation of the shielding is
# performed afterwards, the option has to be included.

####################################
# and the embedding calculation
####################################

$ADFBIN/adf <<eor
Title Input generated by PyADF and later modified

 ATOMS
     C        0.83000        0.66100       -0.44400
     N        0.00000        0.00000        0.00000
     C        1.87800        1.55900       -0.81900
     H        1.78500        2.40300       -0.13500
     H        1.76200        1.94900       -1.83000
     H        2.82900        1.12200       -0.51300
     O       -1.46800        2.60500        1.37700    f=frag1/1
     H       -0.95200        3.29800        0.96500    f=frag1/1
     H       -1.16100        1.79900        0.96100    f=frag1/1
     O        2.40400       -2.51000       -0.36200    f=frag2/1
     H        2.70000       -3.41900       -0.40900    f=frag2/1
     H        1.77500       -2.50000        0.35900    f=frag2/1
     O       -3.22800       -1.61500        1.18500    f=frag3/1
     H       -3.33300       -2.55300        1.03000    f=frag3/1
     H       -3.14200       -1.23600        0.31000    f=frag3/1
     O        0.84000       -2.61200        2.89000    f=frag3/2
     H        0.58800       -3.43700        3.30500    f=frag3/2
     H        0.02500       -2.11500        2.82900    f=frag3/2
     O        2.95400       -0.85100        2.99700    f=frag3/3
     H        2.12000       -1.22400        2.71200    f=frag3/3
     H        2.71800       -0.24100        3.69600    f=frag3/3
     O        3.62200       -0.74000       -2.19300    f=frag3/4
     H        3.05100       -1.25200       -1.62100    f=frag3/4
     H        4.08100       -0.14200       -1.60200    f=frag3/4
     O       -3.80000       -1.13100       -1.71100    f=frag3/5
     H       -3.02600       -0.80900       -2.17400    f=frag3/5
     H       -4.31600       -0.34500       -1.53300    f=frag3/5
     O       -1.77100       -3.79600       -2.15500    f=frag3/6
     H       -2.71500       -3.79000       -2.31700    f=frag3/6
     H       -1.65100       -3.19400       -1.42100    f=frag3/6
     O        1.60000       -0.17800       -3.98800    f=frag3/7
     H        2.40800       -0.18200       -3.47500    f=frag3/7
```

```
      H        1.13900       -0.97100       -3.71300     f=frag3/7
      O       -1.63900       -1.73400        3.28100     f=frag3/8
      H       -1.97000       -1.69700        4.17900     f=frag3/8
      H       -2.38200       -2.04200        2.76400     f=frag3/8
      O        1.57900        2.85500        2.45800     f=frag3/9
      H        0.92600        2.71500        3.14400     f=frag3/9
      H        1.85200        3.76600        2.57000     f=frag3/9
      O       -3.44400        2.36700        3.13700     f=frag3/10
      H       -2.70200        2.29200        2.53700     f=frag3/10
      H       -3.47300        3.29500        3.36800     f=frag3/10
END

SYMMETRY tol=1e-2

FRAGMENTS
  H        t21.H.DZP
  C        t21.C.DZP
  N        t21.N.DZP
  frag1  t21.h2o   type=FDE  &
    fdeoptions RELAX
    RELAXCYCLES 2
  SubEnd
  frag2  t21.h2o   type=FDE  &
    fdeoptions RELAX
    RELAXCYCLES 2
  SubEnd
  frag3  t21.h2o   type=FDE  &
    FDEDENSTYPE SCFexact
  SubEnd
END

XC
  GGA BP86
END

SCF
  iterations  100
  converge 1.0e-07 1.0e-07
END

NUMERICALQUALITY GOOD

SAVE TAPE10

EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END

NOPRINT BAS FUNCTIONS

FDE
  PW91k
END

eor
```

```
# Finally, the calculation of the NMR shielding of the nitrogen atom is
# performed using the NMR program.


$ADFBIN/nmr <<eor
 NMR
  out tens iso
  nuc 3
 END
eor

rm TAPE10 TAPE21
rm t21.h2o
```

Example: FDE NMR spin-spin coupling: NH3-H2O

Download FDE_CPL_NH3-H2O.run

```
#! /bin/sh

# This example demonstrates calculation of NMR spin-spin couplings (J coupling)
# in case of frozen density embedding (FDE). It performs single point runs for
# H2O and NH3 with PBE/(ZORA/jcpl basis) (all-electron) and uses these fragments
# in:
# - an FDE embedding calculation calculation in which water in presence of a
#   frozen ammonia is computed (first the electron density / KS orbitals with
#   ADF then the NMR J couplings with CPL (scalar code branch))
# - a corresponding FDE freeze-thaw calculation followed by calculation of NMR J
#   couplings with CPL scalar, then spin-orbit code branch

# First NH3 with PBE and an all electron TZP basis set. No cpl calculation for
# NH3, thus regular basis is OK. This is not shown. Next H2O with PBE and an all
# electron ZORA/jcpl basis set:

# NH3 PBE/TZP (no cpl calculation for NH3, thus regular basis is OK)

"$ADFBIN/adf" <<eor
 Title NH3 PBE/TZP single point
 ATOMS
  N       -1.39559       -0.02156        0.00004
  H       -1.62981        0.96110       -0.10622
  H       -1.86277       -0.51254       -0.75597
  H       -1.83355       -0.33077        0.86231
 END
 SYMMETRY NOSYM
 BASIS
  Type TZP
  Core None
 END
 XC
  GGA PBE
 END
 NumericalQuality Good
 EXACTDENSITY
 EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
```

```
   END
 NOPRINT BAS FUNCTIONS
eor

rm logfile
mv TAPE21 t21.ammonia


# H2O PBE/(ZORA/jcpl)


"$ADFBIN/adf" <<eor
 Title H2O PBE/(ZORA/jcpl) single point
 ATOMS
   O        1.56850         0.10589          0.00001
   H        0.60674        -0.03396         -0.00063
   H        1.94052        -0.78000          0.00022
 END
 SYMMETRY NOSYM
 BASIS
   Type ZORA/jcpl
   Core None
 END
 XC
   GGA PBE
 END
 NumericalQuality Good
 EXACTDENSITY
 EPRINT
   SFO NOEIG NOOVL NOORBPOP
   SCF NOPOP
 END
 NOPRINT BAS FUNCTIONS
eor
rm logfile
mv TAPE21 t21.water


# NH3-H2O: FDE calculation of H2O in presence of frozen NH3


"$ADFBIN/adf" <<eor
 Title NH3-H2O PBE/PW91k/(ZORA/jcpl) FDE single point calculation (no freeze-thaw)
 ATOMS
   O        1.56850         0.10589          0.00001    f=frag1
   H        0.60674        -0.03396         -0.00063    f=frag1
   H        1.94052        -0.78000          0.00022    f=frag1
   N       -1.39559        -0.02156          0.00004    f=frag2
   H       -1.62981         0.96110         -0.10622    f=frag2
   H       -1.86277        -0.51254         -0.75597    f=frag2
   H       -1.83355        -0.33077          0.86231    f=frag2
 END
 SYMMETRY NOSYM
 FRAGMENTS
   frag1  t21.water
   frag2  t21.ammonia type=FDE
 END
 XC
```

```
  GGA PBE
 END
 NumericalQuality Good
 EXACTDENSITY
 SAVE TAPE10
 EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
 END
 NOPRINT BAS FUNCTIONS
 FDE
  PW91k
 END
eor
rm logfile



# NH3-H2O CPL calculation J(O-H) in water H-bond donor



"$ADFBIN/cpl" <<eor
 GGA
 NMRCoupling
  fc
  dso
  pso
  AtomPert 1
  AtomResp 2
  scf iterations=15 converge=1.0e-05
 End
eor
rm TAPE21 TAPE10 logfile



# NH3-H2O FDE calculation (1 freeze-thaw cycle)



"$ADFBIN/adf" <<eor
 Title NH3-H2O PBE/PW91k/(ZORA/jcpl) FDE single point calculation (1 freeze-thaw␣
→cycle)
 ATOMS
  O       1.56850        0.10589        0.00001    f=frag1
  H       0.60674       -0.03396       -0.00063    f=frag1
  H       1.94052       -0.78000        0.00022    f=frag1
  N      -1.39559       -0.02156        0.00004    f=frag2
  H      -1.62981        0.96110       -0.10622    f=frag2
  H      -1.86277       -0.51254       -0.75597    f=frag2
  H      -1.83355       -0.33077        0.86231    f=frag2
 END
 SYMMETRY NOSYM
 FRAGMENTS
  frag1  t21.water
  frag2  t21.ammonia type=FDE  &
     fdeoptions RELAX
  SubEnd
 END
 XC
  GGA PBE
```

```
 END
 NumericalQuality Good
 EXACTDENSITY
 SAVE TAPE10
 EPRINT
   SFO NOEIG NOOVL NOORBPOP
   SCF NOPOP
 END
 NOPRINT BAS FUNCTIONS
 FDE
   PW91k
   RELAXCYCLES 1
 END
eor
rm logfile


# NH3-H2O CPL calculation J(O-H) in water H-bond donor
# tests scalar code branch (no sd term)


"$ADFBIN/cpl" <<eor
 GGA
 NMRCoupling
   fc
   dso
   pso
   AtomPert 1
   AtomResp 2
   scf iterations=15 converge=1.0e-05
 End
eor
rm logfile


# NH3-H2O CPL calculation J(O-H) in water H-bond donor
# tests spin-orbit code branch (sd term)


"$ADFBIN/cpl" <<eor
 GGA
 NMRCoupling
   fc
   sd
   dso
   pso
   AtomPert 1
   AtomResp 2
   scf iterations=15 converge=1.0e-05
 End
eor
rm TAPE21 TAPE10 logfile
rm t21.H t21.N t21.O t21.water t21.ammonia
```

Example: Subsystem TDDFT, coupled FDE excitation energies

```
Download SUBEXCI_dimer.run
```

```
#! /bin/sh

# This is example for coupled FDE calculation of excitation energies. The
# subsystem TDDFT code couples the monomer excitations to obtain the excited
# states of the total system (often denoted as coupled frozen density embedding,
# FDEc).

# First the isolated fragments are calculated, the TAPe21's of these fragments
# are t21.iso.rho1 and t21.iso.rho2. Next uncoupled FDE excitation energies are
# calculated in which one fragment is frozen and the other active. The key ALLOW
# PARTIALSUPERFRAGS is necessary to be able to use subsystem information for
# only one subsystem from a TAPE21 file of a previous FDE calculation.


####################################################################
# PREPARATION OF ISOLATED FRAGMENTS
####################################################################


$ADFBIN/adf <<eor
Title Fragment no.        1; (isolated)
SYMMETRY NOSYM
XC
  GGA BECKE88 PERDEW86
END
BASIS
 type DZP
 core none
 createoutput None
END
ATOMS
     1 C         1.05754858422573     -1.70701086799077     -3.50000000000000
     2 O         2.28164544472573     -1.70701086799077     -3.50000000000000
     3 C         0.20221626882573     -0.49944933059077     -3.50000000000000
     4 H         0.49106545372573     -2.67285236319077     -3.50000000000000
     5 C        -1.19491351307427     -0.64892031589077     -3.50000000000000
     6 C         0.76942690052573      0.78743686120923     -3.50000000000000
     7 C        -2.02186702237427      0.47538393990923     -3.50000000000000
     8 H        -1.62606655117427     -1.65281003349077     -3.50000000000000
     9 C        -0.05719256647427      1.90851291410923     -3.50000000000000
    10 H         1.85606600152573      0.87752625020923     -3.50000000000000
    11 C        -1.45186603427427      1.75322106580923     -3.50000000000000
    12 H        -3.10608163477427      0.35931991730923     -3.50000000000000
    13 H         0.37787441672573      2.90894982040923     -3.50000000000000
    14 H        -2.09570824397427      2.63406412680923     -3.50000000000000
END
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
SCF
  Converge 1.0e-8
END
NOPRINT BAS FUNCTIONS
eor

mv TAPE21 t21.iso.rho1

$ADFBIN/adf <<eor
Title Fragment no.        2; (isolated)
SYMMETRY NOSYM
```

```
XC
  GGA BECKE88 PERDEW86
END
BASIS
 type DZP
 core none
 createoutput None
END
ATOMS
     1 C         2.00708906832899     0.06235850568037     3.50000000000000
     2 O         2.61913749857899     1.12245748356614     3.50000000000000
     3 C         0.53364394260760    -0.07460023943380     3.50000000000000
     4 H         2.56029077395134    -0.91115102374797     3.50000000000000
     5 C        -0.03547527794391    -1.35928561559301     3.50000000000000
     6 C        -0.29722687542061     1.06006167281502     3.50000000000000
     7 C        -1.42262807969967    -1.51329623449550     3.50000000000000
     8 H         0.61834220104568    -2.23461995830645     3.50000000000000
     9 C        -1.68141695030640     0.90472624158027     3.50000000000000
    10 H         0.16807297559397     2.04616343352651     3.50000000000000
    11 C        -2.24426699857796    -0.38074233566867     3.50000000000000
    12 H        -1.86422099386266    -2.51028564328820     3.50000000000000
    13 H        -2.33028723444571     1.78172375452933     3.50000000000000
    14 H        -3.32902057100121    -0.49790451479758     3.50000000000000
END
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
SCF
  Converge 1.0e-8
END
NOPRINT BAS FUNCTIONS
eor

mv TAPE21 t21.iso.rho2


##################################################################
# FROZEN-DENSITY EMBEDDING CALCULATIONS
# CK: first TDDFT then TDA
##################################################################

$ADFBIN/adf <<eor
Title Fragment no.        1; relaxed; TDDFT
SYMMETRY NOSYM
XC
  GGA BECKE88 PERDEW86
END
EXCITATIONS
  ONLYSING
  LOWEST  20
  CDSPECTRUM
  ANALYTICAL
  VELOCITY
END
ALLOW PARTIALSUPERFRAGS
FRAGMENTS
  rho1 t21.iso.rho1 subfrag=active
  rho2 t21.iso.rho2 subfrag=active type=fde
END
```

14.2. Model Hamiltonians

```
ATOMS
     1 C         1.05754858422573      -1.70701086799077      -3.50000000000000    ␣
→f=rho1
     2 O         2.28164544472573      -1.70701086799077      -3.50000000000000    ␣
→f=rho1
     3 C         0.20221626882573      -0.49944933059077      -3.50000000000000    ␣
→f=rho1
     4 H         0.49106545372573      -2.67285236319077      -3.50000000000000    ␣
→f=rho1
     5 C        -1.19491351307427      -0.64892031589077      -3.50000000000000    ␣
→f=rho1
     6 C         0.76942690052573       0.78743686120923      -3.50000000000000    ␣
→f=rho1
     7 C        -2.02186702237427       0.47538393990923      -3.50000000000000    ␣
→f=rho1
     8 H        -1.62606655117427      -1.65281003349077      -3.50000000000000    ␣
→f=rho1
     9 C        -0.05719256647427       1.90851291410923      -3.50000000000000    ␣
→f=rho1
    10 H         1.85606600152573       0.87752625020923      -3.50000000000000    ␣
→f=rho1
    11 C        -1.45186603427427       1.75322106580923      -3.50000000000000    ␣
→f=rho1
    12 H        -3.10608163477427       0.35931991730923      -3.50000000000000    ␣
→f=rho1
    13 H         0.37787441672573       2.90894982040923      -3.50000000000000    ␣
→f=rho1
    14 H        -2.09570824397427       2.63406412680923      -3.50000000000000    ␣
→f=rho1
    15 C         2.00708906832899       0.06235850568037       3.50000000000000    ␣
→f=rho2
    16 O         2.61913749857899       1.12245748356614       3.50000000000000    ␣
→f=rho2
    17 C         0.53364394260760      -0.07460023943380       3.50000000000000    ␣
→f=rho2
    18 H         2.56029077395134      -0.91115102374797       3.50000000000000    ␣
→f=rho2
    19 C        -0.03547527794391      -1.35928561559301       3.50000000000000    ␣
→f=rho2
    20 C        -0.29722687542061       1.06006167281502       3.50000000000000    ␣
→f=rho2
    21 C        -1.42262807969967      -1.51329623449550       3.50000000000000    ␣
→f=rho2
    22 H         0.61834220104568      -2.23461995830645       3.50000000000000    ␣
→f=rho2
    23 C        -1.68141695030640       0.90472624158027       3.50000000000000    ␣
→f=rho2
    24 H         0.16807297559397       2.04616343352651       3.50000000000000    ␣
→f=rho2
    25 C        -2.24426699857796      -0.38074233566867       3.50000000000000    ␣
→f=rho2
    26 H        -1.86422099386266      -2.51028564328820       3.50000000000000    ␣
→f=rho2
    27 H        -2.33028723444571       1.78172375452933       3.50000000000000    ␣
→f=rho2
    28 H        -3.32902057100121      -0.49790451479758       3.50000000000000    ␣
→f=rho2
END
```

```
FDE
  PW91K
END
SCF
  Converge 1.0e-8
END
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
eor

mv TAPE21 t21.emb.rho1

$ADFBIN/adf <<eor
Title Fragment no.          1; relaxed; TDA
SYMMETRY NOSYM
XC
  GGA BECKE88 PERDEW86
END
TDA
EXCITATIONS
  ONLYSING
  LOWEST  20
  CDSPECTRUM
  ANALYTICAL
  VELOCITY
END
ALLOW PARTIALSUPERFRAGS
FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active
  rho2 t21.emb.rho1 subfrag=rho2 type=fde
END
ATOMS
    1 C       1.05754858422573    -1.70701086799077    -3.50000000000000   ␣
↪f=rho1
    2 O       2.28164544472573    -1.70701086799077    -3.50000000000000   ␣
↪f=rho1
    3 C       0.20221626882573    -0.49944933059077    -3.50000000000000   ␣
↪f=rho1
    4 H       0.49106545372573    -2.67285236319077    -3.50000000000000   ␣
↪f=rho1
    5 C      -1.19491351307427    -0.64892031589077    -3.50000000000000   ␣
↪f=rho1
    6 C       0.76942690052573     0.78743686120923    -3.50000000000000   ␣
↪f=rho1
    7 C      -2.02186702237427     0.47538393990923    -3.50000000000000   ␣
↪f=rho1
    8 H      -1.62606655117427    -1.65281003349077    -3.50000000000000   ␣
↪f=rho1
    9 C      -0.05719256647427     1.90851291410923    -3.50000000000000   ␣
↪f=rho1
   10 H       1.85606600152573     0.87752625020923    -3.50000000000000   ␣
↪f=rho1
   11 C      -1.45186603427427     1.75322106580923    -3.50000000000000   ␣
↪f=rho1
   12 H      -3.10608163477427     0.35931991730923    -3.50000000000000   ␣
↪f=rho1
```

```
    13 H        0.37787441672573        2.90894982040923      -3.50000000000000     ␣
→f=rho1
    14 H       -2.09570824397427        2.63406412680923      -3.50000000000000     ␣
→f=rho1
    15 C        2.00708906832899        0.06235850568037       3.50000000000000     ␣
→f=rho2
    16 O        2.61913749857899        1.12245748356614       3.50000000000000     ␣
→f=rho2
    17 C        0.53364394260760       -0.07460023943380       3.50000000000000     ␣
→f=rho2
    18 H        2.56029077395134       -0.91115102374797       3.50000000000000     ␣
→f=rho2
    19 C       -0.03547527794391       -1.35928561559301       3.50000000000000     ␣
→f=rho2
    20 C       -0.29722687542061        1.06006167281502       3.50000000000000     ␣
→f=rho2
    21 C       -1.42262807969967       -1.51329623449550       3.50000000000000     ␣
→f=rho2
    22 H        0.61834220104568       -2.23461995830645       3.50000000000000     ␣
→f=rho2
    23 C       -1.68141695030640        0.90472624158027       3.50000000000000     ␣
→f=rho2
    24 H        0.16807297559397        2.04616343352651       3.50000000000000     ␣
→f=rho2
    25 C       -2.24426699857796       -0.38074233566867       3.50000000000000     ␣
→f=rho2
    26 H       -1.86422099386266       -2.51028564328820       3.50000000000000     ␣
→f=rho2
    27 H       -2.33028723444571        1.78172375452933       3.50000000000000     ␣
→f=rho2
    28 H       -3.32902057100121       -0.49790451479758       3.50000000000000     ␣
→f=rho2
END
FDE
  PW91K
END
SCF
  Converge 1.0e-8
END
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
eor

mv TAPE21 t21.emb.rho1_TDA

$ADFBIN/adf <<eor
Title Fragment no.        2; relaxed; TDDFT
SYMMETRY NOSYM
XC
  GGA BECKE88 PERDEW86
END
EXCITATIONS
  ONLYSING
  LOWEST  20
  CDSPECTRUM
  ANALYTICAL
```

```
  VELOCITY
END
ALLOW PARTIALSUPERFRAGS
FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active type=fde
  rho2 t21.iso.rho2 subfrag=active
END
ATOMS
     1 C        1.05754858422573     -1.70701086799077     -3.50000000000000     ␣
↪f=rho1
     2 O        2.28164544472573     -1.70701086799077     -3.50000000000000     ␣
↪f=rho1
     3 C        0.20221626882573     -0.49944933059077     -3.50000000000000     ␣
↪f=rho1
     4 H        0.49106545372573     -2.67285236319077     -3.50000000000000     ␣
↪f=rho1
     5 C       -1.19491351307427     -0.64892031589077     -3.50000000000000     ␣
↪f=rho1
     6 C        0.76942690052573      0.78743686120923     -3.50000000000000     ␣
↪f=rho1
     7 C       -2.02186702237427      0.47538393990923     -3.50000000000000     ␣
↪f=rho1
     8 H       -1.62606655117427     -1.65281003349077     -3.50000000000000     ␣
↪f=rho1
     9 C       -0.05719256647427      1.90851291410923     -3.50000000000000     ␣
↪f=rho1
    10 H        1.85606600152573      0.87752625020923     -3.50000000000000     ␣
↪f=rho1
    11 C       -1.45186603427427      1.75322106580923     -3.50000000000000     ␣
↪f=rho1
    12 H       -3.10608163477427      0.35931991730923     -3.50000000000000     ␣
↪f=rho1
    13 H        0.37787441672573      2.90894982040923     -3.50000000000000     ␣
↪f=rho1
    14 H       -2.09570824397427      2.63406412680923     -3.50000000000000     ␣
↪f=rho1
    15 C        2.00708906832899      0.06235850568037      3.50000000000000     ␣
↪f=rho2
    16 O        2.61913749857899      1.12245748356614      3.50000000000000     ␣
↪f=rho2
    17 C        0.53364394260760     -0.07460023943380      3.50000000000000     ␣
↪f=rho2
    18 H        2.56029077395134     -0.91115102374797      3.50000000000000     ␣
↪f=rho2
    19 C       -0.03547527794391     -1.35928561559301      3.50000000000000     ␣
↪f=rho2
    20 C       -0.29722687542061      1.06006167281502      3.50000000000000     ␣
↪f=rho2
    21 C       -1.42262807969967     -1.51329623449550      3.50000000000000     ␣
↪f=rho2
    22 H        0.61834220104568     -2.23461995830645      3.50000000000000     ␣
↪f=rho2
    23 C       -1.68141695030640      0.90472624158027      3.50000000000000     ␣
↪f=rho2
    24 H        0.16807297559397      2.04616343352651      3.50000000000000     ␣
↪f=rho2
    25 C       -2.24426699857796     -0.38074233566867      3.50000000000000     ␣
↪f=rho2
```

```
    26 H        -1.86422099386266     -2.51028564328820      3.50000000000000    ␣
→f=rho2
    27 H        -2.33028723444571      1.78172375452933      3.50000000000000    ␣
→f=rho2
    28 H        -3.32902057100121     -0.49790451479758      3.50000000000000    ␣
→f=rho2
END
FDE
  PW91K
END
SCF
  Converge 1.0e-8
END
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
eor

mv TAPE21 t21.emb.rho2

$ADFBIN/adf <<eor
Title Fragment no.          2; relaxed; TDA
SYMMETRY NOSYM
XC
  GGA BECKE88 PERDEW86
END
TDA
EXCITATIONS
  ONLYSING
  LOWEST  20
  CDSPECTRUM
  ANALYTICAL
  VELOCITY
END
ALLOW PARTIALSUPERFRAGS
FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active type=fde
  rho2 t21.emb.rho2 subfrag=active
END
ATOMS
     1 C         1.05754858422573     -1.70701086799077     -3.50000000000000    ␣
→f=rho1
     2 O         2.28164544472573     -1.70701086799077     -3.50000000000000    ␣
→f=rho1
     3 C         0.20221626882573     -0.49944933059077     -3.50000000000000    ␣
→f=rho1
     4 H         0.49106545372573     -2.67285236319077     -3.50000000000000    ␣
→f=rho1
     5 C        -1.19491351307427     -0.64892031589077     -3.50000000000000    ␣
→f=rho1
     6 C         0.76942690052573      0.78743686120923     -3.50000000000000    ␣
→f=rho1
     7 C        -2.02186702237427      0.47538393990923     -3.50000000000000    ␣
→f=rho1
     8 H        -1.62606655117427     -1.65281003349077     -3.50000000000000    ␣
→f=rho1
     9 C        -0.05719256647427      1.90851291410923     -3.50000000000000    ␣
→f=rho1
```

```
    10 H         1.85606600152573      0.87752625020923     -3.50000000000000    ⎵
→f=rho1
    11 C        -1.45186603427427      1.75322106580923     -3.50000000000000    ⎵
→f=rho1
    12 H        -3.10608163477427      0.35931991730923     -3.50000000000000    ⎵
→f=rho1
    13 H         0.37787441672573      2.90894982040923     -3.50000000000000    ⎵
→f=rho1
    14 H        -2.09570824397427      2.63406412680923     -3.50000000000000    ⎵
→f=rho1
    15 C         2.00708906832899      0.06235850568037      3.50000000000000    ⎵
→f=rho2
    16 O         2.61913749857899      1.12245748356614      3.50000000000000    ⎵
→f=rho2
    17 C         0.53364394260760     -0.07460023943380      3.50000000000000    ⎵
→f=rho2
    18 H         2.56029077395134     -0.91115102374797      3.50000000000000    ⎵
→f=rho2
    19 C        -0.03547527794391     -1.35928561559301      3.50000000000000    ⎵
→f=rho2
    20 C        -0.29722687542061      1.06006167281502      3.50000000000000    ⎵
→f=rho2
    21 C        -1.42262807969967     -1.51329623449550      3.50000000000000    ⎵
→f=rho2
    22 H         0.61834220104568     -2.23461995830645      3.50000000000000    ⎵
→f=rho2
    23 C        -1.68141695030640      0.90472624158027      3.50000000000000    ⎵
→f=rho2
    24 H         0.16807297559397      2.04616343352651      3.50000000000000    ⎵
→f=rho2
    25 C        -2.24426699857796     -0.38074233566867      3.50000000000000    ⎵
→f=rho2
    26 H        -1.86422099386266     -2.51028564328820      3.50000000000000    ⎵
→f=rho2
    27 H        -2.33028723444571      1.78172375452933      3.50000000000000    ⎵
→f=rho2
    28 H        -3.32902057100121     -0.49790451479758      3.50000000000000    ⎵
→f=rho2
END
FDE
  PW91K
END
SCF
  Converge 1.0e-8
END
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
eor

mv TAPE21 t21.emb.rho2_TDA

################################################################
# SUBSYS EXCITATIONS CALCULATION
################################################################

$ADFBIN/adf <<eor
```

```
Title COUPLED SUBSYSTEM EXCITATIONS; TDDFT
SYMMETRY NOSYM
XC
  GGA BECKE88 PERDEW86
END
DIFFUSE
ALLOW PARTIALSUPERFRAGS
FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active
  rho2 t21.emb.rho2 subfrag=active type=fde
END
ATOMS
    1 C        1.05754858422573    -1.70701086799077    -3.50000000000000    ↵
↪f=rho1
    2 O        2.28164544472573    -1.70701086799077    -3.50000000000000    ↵
↪f=rho1
    3 C        0.20221626882573    -0.49944933059077    -3.50000000000000    ↵
↪f=rho1
    4 H        0.49106545372573    -2.67285236319077    -3.50000000000000    ↵
↪f=rho1
    5 C       -1.19491351307427    -0.64892031589077    -3.50000000000000    ↵
↪f=rho1
    6 C        0.76942690052573     0.78743686120923    -3.50000000000000    ↵
↪f=rho1
    7 C       -2.02186702237427     0.47538393990923    -3.50000000000000    ↵
↪f=rho1
    8 H       -1.62606655117427    -1.65281003349077    -3.50000000000000    ↵
↪f=rho1
    9 C       -0.05719256647427     1.90851291410923    -3.50000000000000    ↵
↪f=rho1
   10 H        1.85606600152573     0.87752625020923    -3.50000000000000    ↵
↪f=rho1
   11 C       -1.45186603427427     1.75322106580923    -3.50000000000000    ↵
↪f=rho1
   12 H       -3.10608163477427     0.35931991730923    -3.50000000000000    ↵
↪f=rho1
   13 H        0.37787441672573     2.90894982040923    -3.50000000000000    ↵
↪f=rho1
   14 H       -2.09570824397427     2.63406412680923    -3.50000000000000    ↵
↪f=rho1
   15 C        2.00708906832899     0.06235850568037     3.50000000000000    ↵
↪f=rho2
   16 O        2.61913749857899     1.12245748356614     3.50000000000000    ↵
↪f=rho2
   17 C        0.53364394260760    -0.07460023943380     3.50000000000000    ↵
↪f=rho2
   18 H        2.56029077395134    -0.91115102374797     3.50000000000000    ↵
↪f=rho2
   19 C       -0.03547527794391    -1.35928561559301     3.50000000000000    ↵
↪f=rho2
   20 C       -0.29722687542061     1.06006167281502     3.50000000000000    ↵
↪f=rho2
   21 C       -1.42262807969967    -1.51329623449550     3.50000000000000    ↵
↪f=rho2
   22 H        0.61834220104568    -2.23461995830645     3.50000000000000    ↵
↪f=rho2
   23 C       -1.68141695030640     0.90472624158027     3.50000000000000    ↵
↪f=rho2
```

```
    24 H         0.16807297559397      2.04616343352651      3.50000000000000     ⌴
→f=rho2
    25 C        -2.24426699857796     -0.38074233566867      3.50000000000000     ⌴
→f=rho2
    26 H        -1.86422099386266     -2.51028564328820      3.50000000000000     ⌴
→f=rho2
    27 H        -2.33028723444571      1.78172375452933      3.50000000000000     ⌴
→f=rho2
    28 H        -3.32902057100121     -0.49790451479758      3.50000000000000     ⌴
→f=rho2
END
FDE
  PW91K
END
SUBEXCI
  CTHRES  10000.00
  SFTHRES 0.00010000
  COUPLBLOCK
END
SCF
  Converge 1.0e-8
END
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
eor

$ADFBIN/adf <<eor
Title COUPLED SUBSYSTEM EXCITATIONS; TDA
SYMMETRY NOSYM
XC
  GGA BECKE88 PERDEW86
END
DIFFUSE
ALLOW PARTIALSUPERFRAGS
FRAGMENTS
  rho1 t21.emb.rho1_TDA subfrag=active
  rho2 t21.emb.rho2_TDA subfrag=active type=fde
END
ATOMS
    1 C         1.05754858422573     -1.70701086799077     -3.50000000000000     ⌴
→f=rho1
    2 O         2.28164544472573     -1.70701086799077     -3.50000000000000     ⌴
→f=rho1
    3 C         0.20221626882573     -0.49944933059077     -3.50000000000000     ⌴
→f=rho1
    4 H         0.49106545372573     -2.67285236319077     -3.50000000000000     ⌴
→f=rho1
    5 C        -1.19491351307427     -0.64892031589077     -3.50000000000000     ⌴
→f=rho1
    6 C         0.76942690052573      0.78743686120923     -3.50000000000000     ⌴
→f=rho1
    7 C        -2.02186702237427      0.47538393990923     -3.50000000000000     ⌴
→f=rho1
    8 H        -1.62606655117427     -1.65281003349077     -3.50000000000000     ⌴
→f=rho1
    9 C        -0.05719256647427      1.90851291410923     -3.50000000000000     ⌴
→f=rho1
```

```
    10 H        1.85606600152573        0.87752625020923       -3.50000000000000    ␣
↪f=rho1
    11 C       -1.45186603427427        1.75322106580923       -3.50000000000000    ␣
↪f=rho1
    12 H       -3.10608163477427        0.35931991730923       -3.50000000000000    ␣
↪f=rho1
    13 H        0.37787441672573        2.90894982040923       -3.50000000000000    ␣
↪f=rho1
    14 H       -2.09570824397427        2.63406412680923       -3.50000000000000    ␣
↪f=rho1
    15 C        2.00708906832899        0.06235850568037        3.50000000000000    ␣
↪f=rho2
    16 O        2.61913749857899        1.12245748356614        3.50000000000000    ␣
↪f=rho2
    17 C        0.53364394260760       -0.07460023943380        3.50000000000000    ␣
↪f=rho2
    18 H        2.56029077395134       -0.91115102374797        3.50000000000000    ␣
↪f=rho2
    19 C       -0.03547527794391       -1.35928561559301        3.50000000000000    ␣
↪f=rho2
    20 C       -0.29722687542061        1.06006167281502        3.50000000000000    ␣
↪f=rho2
    21 C       -1.42262807969967       -1.51329623449550        3.50000000000000    ␣
↪f=rho2
    22 H        0.61834220104568       -2.23461995830645        3.50000000000000    ␣
↪f=rho2
    23 C       -1.68141695030640        0.90472624158027        3.50000000000000    ␣
↪f=rho2
    24 H        0.16807297559397        2.04616343352651        3.50000000000000    ␣
↪f=rho2
    25 C       -2.24426699857796       -0.38074233566867        3.50000000000000    ␣
↪f=rho2
    26 H       -1.86422099386266       -2.51028564328820        3.50000000000000    ␣
↪f=rho2
    27 H       -2.33028723444571        1.78172375452933        3.50000000000000    ␣
↪f=rho2
    28 H       -3.32902057100121       -0.49790451479758        3.50000000000000    ␣
↪f=rho2
END
FDE
  PW91K
END
SUBEXCI
  CTHRES  10000.00
  SFTHRES 0.00010000
  COUPLBLOCK
  TDA
END
SCF
  Converge 1.0e-8
END
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
eor
```

Example: FDE and COSMO: H2O-NH3

Download LoCOSMO_H2O-NH3.run

```
#!/bin/sh

# This is example for a calculation of FDE and (localized) COSMO in ADF in case
# of closed shell fragments.

# It performs single point runs for H2 O and NH3 in COSMO (water) and uses these
# fragments in a freeze-and-thaw calculations. Recommended is to limit the
# number of freeze-and-thaw cycles in such calculations where FDE and COSMO are
# combined, because the FT convergence criterion may not be met in a finite
# number of cycles, although the density is practically converged.

# In the freeze-and-thaw cycles LoCosmo is included in the SOLVATION key, thus
# the local COSMO will be used in the calculation. A cutoff radius (Angstrom)
# must be specified after the LoCOSMO subargument, in this case it is 3.0
# Angstrom. All surface charges within this radius from any atom of the active
# fragment are included in the active charge space.


$ADFBIN/adf <<EOR
Title H2O isolated
ATOMS
   H        0.0201  0.6299  0.0000
   O       -0.0370  1.6130  0.0000
   H        0.8936  1.9007  0.0000
END
SYMMETRY NOSYM
XC
  GGA BP86
END
BASIS
  Type DZP
  CreateOutput None
END
NumericalQuality Good
EXACTDENSITY
EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END
NOPRINT BAS FUNCTIONS
SOLVATION
END
EOR
mv TAPE21 t21.iso.rho1


$ADFBIN/adf <<EOR
Title NH3 isolated
ATOMS
   N       -0.0370 -1.4327  0.0000
   H        0.8110 -2.0086  0.0000
   H       -0.5850 -1.6987  0.8244
   H       -0.5850 -1.6987 -0.8244
END
```

```
SYMMETRY NOSYM
XC
  GGA BP86
END
BASIS
  Type DZP
  CreateOutput None
END
NumericalQuality Good
EXACTDENSITY
EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END
NOPRINT BAS FUNCTIONS
SOLVATION
END
EOR
mv TAPE21 t21.iso.rho2


$ADFBIN/adf <<EOR
Title FT cycles for H2O-NH3
ATOMS
   H       0.0201   0.6299   0.0000   f=frag1
   O      -0.0370   1.6130   0.0000   f=frag1
   H       0.8936   1.9007   0.0000   f=frag1
   N      -0.0370  -1.4327   0.0000   f=frag2
   H       0.8110  -2.0086   0.0000   f=frag2
   H      -0.5850  -1.6987   0.8244   f=frag2
   H      -0.5850  -1.6987  -0.8244   f=frag2
END
SYMMETRY NOSYM
FRAGMENTS
  frag1  t21.iso.rho1
  frag2  t21.iso.rho2 type=FDE &
     FDEOPTIONS RELAX
  SubEnd
END
XC
  GGA BP86
END
NumericalQuality Good
EXACTDENSITY
EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END
NOPRINT BAS FUNCTIONS
SOLVATION
  CHARGED LoCOSMO 3.0
END
FDE
  PW91k
  RELAXCYCLES 3
END
EOR
```

Example: FDE and COSMO: H2O-NH3

Download LoCOSMO_H2O-NH3_TDDFT.run

```sh
#!/bin/sh

# This is example for a calculation of excitation energies using FDE and
# localized COSMO, which is based on the LoCOSMO_H2O-NH3 example. The difference
# is that it includes the calculation of excitation energies (EXCITATION block
# key), and performs only 1 freeze-and-thaw cycle for H2 O and NH3 in COSMO
# (water). The input for the fragment calculations are not repeated here.

$ADFBIN/adf <<EOR
Title H2O isolated
ATOMS
    H        0.0201  0.6299  0.0000
    O       -0.0370  1.6130  0.0000
    H        0.8936  1.9007  0.0000
END
SYMMETRY NOSYM
XC
  GGA BP86
END
BASIS
  Type DZP
  CreateOutput None
END
NumericalQuality Good
EXACTDENSITY
EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END
NOPRINT BAS FUNCTIONS
SOLVATION
END
EOR
mv TAPE21 t21.iso.rho1

$ADFBIN/adf <<EOR
Title NH3 isolated
ATOMS
    N       -0.0370 -1.4327  0.0000
    H        0.8110 -2.0086  0.0000
    H       -0.5850 -1.6987  0.8244
    H       -0.5850 -1.6987 -0.8244
END
SYMMETRY NOSYM
XC
  GGA BP86
END
BASIS
  Type DZP
  CreateOutput None
END
NumericalQuality Good
EXACTDENSITY
EPRINT
```

```
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END
NOPRINT BAS FUNCTIONS
SOLVATION
END
EOR
mv TAPE21 t21.iso.rho2

$ADFBIN/adf <<EOR
Title FT cycles for H2O-NH3
ATOMS
   H       0.0201   0.6299   0.0000   f=frag1
   O      -0.0370   1.6130   0.0000   f=frag1
   H       0.8936   1.9007   0.0000   f=frag1
   N      -0.0370  -1.4327   0.0000   f=frag2
   H       0.8110  -2.0086   0.0000   f=frag2
   H      -0.5850  -1.6987   0.8244   f=frag2
   H      -0.5850  -1.6987  -0.8244   f=frag2
END
SYMMETRY NOSYM
FRAGMENTS
  frag1  t21.iso.rho1
  frag2  t21.iso.rho2 type=FDE &
     FDEOPTIONS RELAX
  SubEnd
END
XC
  GGA BP86
END
NumericalQuality Good
EXACTDENSITY
EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END
NOPRINT BAS FUNCTIONS
SOLVATION
  CHARGED LoCOSMO 3.0
END
EXCITATIONS
END
FDE
  PW91k
  RELAXCYCLES 1
END
EOR
```

14.2.5 QM/MM calculations

- QM/MM manual: QM/MM Examples
- QM/MM manual: pdb2adf example

14.2.6 Quild: Quantum-regions Interconnected by Local Descriptions

- Quild manual: Quild examples

14.2.7 DIM/QM: Discrete Interaction Model/Quantum Mechanics

Example: DRF: H2O and H2O

Download DIMQM_DRF_BE.run

```
#! /bin/sh

# Example of a DRF bond energy calculation.

"$ADFBIN/adf" <<eor
ATOMS
  O      2.698990442000      0.000000000000     -0.000314732000
  H      1.927785302000      0.000000000000     -0.591074192000
  H      3.470195581000      0.000000000000     -0.591074192000
END

SYMMETRY NOSYM

BASIS
  type TZP
  core Large
  createoutput None
END

NOPRINT LOGFILE
DIMQM
  drf
END

DIMPAR
  GROUP mol2
    3
    O     -0.801009558000      0.000000000000     -0.000314732000 pol=5.7494 char=-
→0.7134114435520156
    H     -1.572214698000     -0.000000000000     -0.591074192000 pol=2.7927␣
→char=0.3567057217727492
    H     -0.029804419000     -0.000000000000     -0.591074192000 pol=2.7927␣
→char=0.3567057217727492
  SUBEND
END
eor
```

Example: DRF: hyperpolarizability H2O in water

Download DIMQM_DRF_Hyperpol.run

```
#!/bin/sh

# == Expert option ==
```

```
# This example demonstrates how to use DRF. Here, the hyperpolarizability is
# calculated of a water molecule surrounded by an environment of 127 water
# molecules. See also: C.R. Jacob, J. Neugebauer, L. Jensen and L. Visscher,
# Comparison of frozen-density embedding and discrete reaction field solvent
# models for molecular properties, Physical Chemistry Chemical Physics 8, 2349
# (2006)


$ADFBIN/adf <<eor
title DRF-TEST H2O in H2O

Basis
H DZ/H
O DZ/O
END

Symmetry NOSYM

Units
 LENGTH Bohr
End

Atoms
O   0.000000   0.000000   0.000000
H  -1.430143   0.000000   1.107393
H   1.430143   0.000000   1.107393
End

DIMQM
 DRF
End

Response
 ALLCOMPONENTS
 HYPERPOL 0.0428
 DYNAHYP
 HARTREE
End

EXTERNALS
O   4 water    2, -0.6690, -11.380487, -11.810553,  -4.515226, 9.3005, 3.0236
H   5 water    2,  0.3345, -13.104751, -11.837669,  -3.969549, 0.0690, 1.5118
H   6 water    2,  0.3345, -10.510898, -12.853311,  -3.320199, 0.0690, 1.5118
        ANAL 2
O   7 water    3, -0.6690,  -1.116350,   9.119186,  -3.230948, 9.3005, 3.0236
H   8 water    3,  0.3345,  -2.822714,   9.717033,  -3.180632, 0.0690, 1.5118
H   9 water    3,  0.3345,  -0.123788,  10.538199,  -2.708607, 0.0690, 1.5118
        ANAL 2
O  10 water    4, -0.6690,   4.469332,  -0.503788,   2.210877, 9.3005, 3.0236
H  11 water    4,  0.3345,   4.273576,  -2.085090,   3.066884, 0.0690, 1.5118
H  12 water    4,  0.3345,   5.919073,  -0.737018,   1.154734, 0.0690, 1.5118
        ANAL 2
O  13 water    5, -0.6690, -16.969012,  -3.658353, -10.785713, 9.3005, 3.0236
H  14 water    5,  0.3345, -15.708232,  -4.239969,  -9.626485, 0.0690, 1.5118
H  15 water    5,  0.3345, -16.478125,  -4.360815, -12.378556, 0.0690, 1.5118
        ANAL 2
O  16 water    6, -0.6690, -10.765608,  -4.163577,   0.518032, 9.3005, 3.0236
H  17 water    6,  0.3345,  -9.161172,  -3.986233,  -0.298000, 0.0690, 1.5118
```

```
H  18 water    6,  0.3345, -11.974840,  -3.796367,  -0.775995,  0.0690,  1.5118
          ANAL 2
O  19 water    7, -0.6690,  -6.554233,  -9.202975, -11.231254,  9.3005,  3.0236
H  20 water    7,  0.3345,  -8.153024,  -9.653544, -10.515361,  0.0690,  1.5118
H  21 water    7,  0.3345,  -6.907173,  -7.749987, -12.249022,  0.0690,  1.5118
          ANAL 2
O  22 water    8, -0.6690, -14.167136,  -6.558132,  -6.629574,  9.3005,  3.0236
H  23 water    8,  0.3345, -13.746902,  -8.312028,  -6.766757,  0.0690,  1.5118
H  24 water    8,  0.3345, -15.432950,  -6.495396,  -5.339056,  0.0690,  1.5118
          ANAL 2
O  25 water    9, -0.6690,  -1.012338,  -6.537375, -20.632283,  9.3005,  3.0236
H  26 water    9,  0.3345,   0.513580,  -7.016738, -19.787667,  0.0690,  1.5118
H  27 water    9,  0.3345,  -2.258461,  -6.487220, -19.322220,  0.0690,  1.5118
          ANAL 2
O  28 water   10, -0.6690,  -7.973723,   4.333816,   8.858697,  9.3005,  3.0236
H  29 water   10,  0.3345,  -7.392103,   2.717044,   9.423808,  0.0690,  1.5118
H  30 water   10,  0.3345,  -7.341328,   4.477715,   7.170208,  0.0690,  1.5118
          ANAL 2
O  31 water   11, -0.6690,   1.687680, -11.293163,  -2.876367,  9.3005,  3.0236
H  32 water   11,  0.3345,   1.520005, -11.659989,  -4.639568,  0.0690,  1.5118
H  33 water   11,  0.3345,   0.925637,  -9.664781,  -2.678161,  0.0690,  1.5118
          ANAL 2
O  34 water   12, -0.6690,   1.091846,  -1.698117, -18.817171,  9.3005,  3.0236
H  35 water   12,  0.3345,   2.072156,  -2.343260, -20.193538,  0.0690,  1.5118
H  36 water   12,  0.3345,   0.133995,  -3.119042, -18.238305,  0.0690,  1.5118
          ANAL 2
O  37 water   13, -0.6690, -13.852587,  -1.513172,  -3.428477,  9.3005,  3.0236
H  38 water   13,  0.3345, -13.345615,  -1.909476,  -5.118903,  0.0690,  1.5118
H  39 water   13,  0.3345, -15.318863,  -2.533452,  -3.144529,  0.0690,  1.5118
          ANAL 2
O  40 water   14, -0.6690,   5.289604,  -6.812668, -13.105858,  9.3005,  3.0236
H  41 water   14,  0.3345,   7.017101,  -6.621783, -12.604888,  0.0690,  1.5118
H  42 water   14,  0.3345,   4.346729,  -5.977914, -11.807486,  0.0690,  1.5118
          ANAL 2
O  43 water   15, -0.6690, -11.474072, -10.907083,  -9.440850,  9.3005,  3.0236
H  44 water   15,  0.3345, -11.216111, -12.454419, -10.341325,  0.0690,  1.5118
H  45 water   15,  0.3345, -11.176336, -11.335948,  -7.709073,  0.0690,  1.5118
          ANAL 2
O  46 water   16, -0.6690,  -1.645921, -11.435233,   1.480980,  9.3005,  3.0236
H  47 water   16,  0.3345,  -0.305580, -10.473636,   0.739091,  0.0690,  1.5118
H  48 water   16,  0.3345,  -1.729933, -12.940677,   0.481912,  0.0690,  1.5118
          ANAL 2
O  49 water   17, -0.6690,  -7.324684,  -5.018181,   4.387749,  9.3005,  3.0236
H  50 water   17,  0.3345,  -8.818380,  -4.730290,   3.409170,  0.0690,  1.5118
H  51 water   17,  0.3345,  -7.759277,  -6.389120,   5.484697,  0.0690,  1.5118
          ANAL 2
O  52 water   18, -0.6690,   5.208797,  -6.838506,   3.691461,  9.3005,  3.0236
H  53 water   18,  0.3345,   5.265791,  -6.685119,   1.890123,  0.0690,  1.5118
H  54 water   18,  0.3345,   6.391615,  -5.607197,   4.288436,  0.0690,  1.5118
          ANAL 2
O  55 water   19, -0.6690,  -1.248954,  -6.977509,  -2.931094,  9.3005,  3.0236
H  56 water   19,  0.3345,  -2.378326,  -6.595799,  -1.570804,  0.0690,  1.5118
H  57 water   19,  0.3345,  -0.212883,  -5.503084,  -3.086746,  0.0690,  1.5118
          ANAL 2
O  58 water   20, -0.6690,   6.737491,  -0.260656,  -6.588501,  9.3005,  3.0236
H  59 water   20,  0.3345,   6.563018,  -2.046802,  -6.813946,  0.0690,  1.5118
H  60 water   20,  0.3345,   8.299253,   0.148272,  -7.404135,  0.0690,  1.5118
          ANAL 2
```

```
O  61 water   21, −0.6690,   5.306243,  −5.543095,  −6.435993,  9.3005,  3.0236
H  62 water   21,  0.3345,   3.812804,  −6.090721,  −7.296990,  0.0690,  1.5118
H  63 water   21,  0.3345,   6.651877,  −6.435286,  −7.251376,  0.0690,  1.5118
          ANAL 2
O  64 water   22, −0.6690,  −5.722143,  −7.593764,  −5.609892,  9.3005,  3.0236
H  65 water   22,  0.3345,  −5.637663,  −7.499595,  −7.414221,  0.0690,  1.5118
H  66 water   22,  0.3345,  −3.997335,  −7.514255,  −5.071065,  0.0690,  1.5118
          ANAL 2
O  67 water   23, −0.6690,  −4.964121,   4.323349,  −3.258997,  9.3005,  3.0236
H  68 water   23,  0.3345,  −6.269204,   5.559515,  −3.058293,  0.0690,  1.5118
H  69 water   23,  0.3345,  −3.431147,   5.226326,  −2.933032,  0.0690,  1.5118
          ANAL 2
O  70 water   24, −0.6690, −10.942326,   0.554379, −13.182449,  9.3005,  3.0236
H  71 water   24,  0.3345,  −9.993653,   2.083340, −13.366497,  0.0690,  1.5118
H  72 water   24,  0.3345, −12.257976,   0.686468, −14.416629,  0.0690,  1.5118
          ANAL 2
O  73 water   25, −0.6690,  −9.296367,   5.399236, −12.624634,  9.3005,  3.0236
H  74 water   25,  0.3345,  −7.787156,   5.399699, −11.627669,  0.0690,  1.5118
H  75 water   25,  0.3345,  −8.713646,   5.570736, −14.328345,  0.0690,  1.5118
          ANAL 2
O  76 water   26, −0.6690,   2.976413, −12.445061,   3.626050,  9.3005,  3.0236
H  77 water   26,  0.3345,   3.661100, −10.770913,   3.633078,  0.0690,  1.5118
H  78 water   26,  0.3345,   1.221500, −12.214985,   3.253348,  0.0690,  1.5118
          ANAL 2
O  79 water   27, −0.6690,  −7.520338,  −2.709606, −16.431855,  9.3005,  3.0236
H  80 water   27,  0.3345,  −8.541446,  −2.016576, −15.109472,  0.0690,  1.5118
H  81 water   27,  0.3345,  −8.126205,  −1.911131, −17.937498,  0.0690,  1.5118
          ANAL 2
O  82 water   28, −0.6690,  −1.697017,  −2.043822,  −8.206757,  9.3005,  3.0236
H  83 water   28,  0.3345,  −1.169364,  −0.432668,  −8.837101,  0.0690,  1.5118
H  84 water   28,  0.3345,  −0.636624,  −2.337803,  −6.771228,  0.0690,  1.5118
          ANAL 2
O  85 water   29, −0.6690,   5.301322,  −9.560473,   8.452236,  9.3005,  3.0236
H  86 water   29,  0.3345,   5.273862,  −8.617629,   6.908897,  0.0690,  1.5118
H  87 water   29,  0.3345,   4.550177,  −8.455441,   9.671362,  0.0690,  1.5118
          ANAL 2
O  88 water   30, −0.6690,   5.994893,  −5.843987,  −1.278668,  9.3005,  3.0236
H  89 water   30,  0.3345,   6.104998,  −5.948818,  −3.081028,  0.0690,  1.5118
H  90 water   30,  0.3345,   7.022997,  −4.420634,  −0.844246,  0.0690,  1.5118
          ANAL 2
O  91 water   31, −0.6690,  −8.499334,   6.307303,   1.262069,  9.3005,  3.0236
H  92 water   31,  0.3345,  −9.264138,   5.216349,   0.038766,  0.0690,  1.5118
H  93 water   31,  0.3345,  −9.841332,   6.720278,   2.402299,  0.0690,  1.5118
          ANAL 2
O  94 water   32, −0.6690,   1.422658,   7.151005,  −6.702457,  9.3005,  3.0236
H  95 water   32,  0.3345,   0.491242,   7.745308,  −5.270378,  0.0690,  1.5118
H  96 water   32,  0.3345,   2.872442,   6.334862,  −5.992754,  0.0690,  1.5118
          ANAL 2
O  97 water   33, −0.6690,   1.165723, −15.863273,  −8.030117,  9.3005,  3.0236
H  98 water   33,  0.3345,  −0.439800, −16.566460,  −8.476689,  0.0690,  1.5118
H  99 water   33,  0.3345,   1.270225, −14.330699,  −8.985052,  0.0690,  1.5118
          ANAL 2
O 100 water   34, −0.6690,  −1.891948, −10.635521,  −8.584698,  9.3005,  3.0236
H 101 water   34,  0.3345,  −2.788166, −10.717520, −10.153685,  0.0690,  1.5118
H 102 water   34,  0.3345,  −2.907812, −11.590566,  −7.432521,  0.0690,  1.5118
          ANAL 2
O 103 water   35, −0.6690,   1.303221,  −6.546667,  −9.290668,  9.3005,  3.0236
H 104 water   35,  0.3345,   0.041455,  −7.803822,  −8.975883,  0.0690,  1.5118
```

```
H 105 water   35,  0.3345,   0.439977,  -4.972924,  -9.067678,  0.0690,  1.5118
         ANAL 2
O 106 water   36, -0.6690,  -4.108004,  -6.593872, -16.433977,  9.3005,  3.0236
H 107 water   36,  0.3345,  -4.900893,  -8.114132, -17.009917,  0.0690,  1.5118
H 108 water   36,  0.3345,  -5.447969,  -5.380617, -16.369719,  0.0690,  1.5118
         ANAL 2
O 109 water   37, -0.6690,  -5.920829,  -3.031071,  -2.510728,  9.3005,  3.0236
H 110 water   37,  0.3345,  -5.780713,  -4.560882,  -3.465525,  0.0690,  1.5118
H 111 water   37,  0.3345,  -6.212678,  -1.747904,  -3.751669,  0.0690,  1.5118
         ANAL 2
O 112 water   38, -0.6690,  -5.199834, -12.826876,  -4.968785,  9.3005,  3.0236
H 113 water   38,  0.3345,  -6.704708, -13.433270,  -5.768313,  0.0690,  1.5118
H 114 water   38,  0.3345,  -5.429559, -11.035446,  -4.870676,  0.0690,  1.5118
         ANAL 2
O 115 water   39, -0.6690,  -4.897867,   5.501788, -16.517711,  9.3005,  3.0236
H 116 water   39,  0.3345,  -4.966750,   7.305873, -16.407434,  0.0690,  1.5118
H 117 water   39,  0.3345,  -3.227614,   5.077983, -15.967922,  0.0690,  1.5118
         ANAL 2
O 118 water   40, -0.6690, -17.578987,  -4.993467,  -2.976815,  9.3005,  3.0236
H 119 water   40,  0.3345, -18.763866,  -3.841943,  -2.240856,  0.0690,  1.5118
H 120 water   40,  0.3345, -17.398660,  -6.306125,  -1.745551,  0.0690,  1.5118
         ANAL 2
O 121 water   41, -0.6690,  -3.507768, -17.323802, -10.104425,  9.3005,  3.0236
H 122 water   41,  0.3345,  -4.401937, -18.633093, -10.974965,  0.0690,  1.5118
H 123 water   41,  0.3345,  -4.539528, -16.949546,  -8.666718,  0.0690,  1.5118
         ANAL 2
O 124 water   42, -0.6690, -10.819116,  -4.666408, -11.144871,  9.3005,  3.0236
H 125 water   42,  0.3345, -11.621853,  -4.882965,  -9.538536,  0.0690,  1.5118
H 126 water   42,  0.3345, -10.609162,  -2.877355, -11.308636,  0.0690,  1.5118
         ANAL 2
O 127 water   43, -0.6690,  -6.392055,   0.017935,  -6.732485,  9.3005,  3.0236
H 128 water   43,  0.3345,  -5.821252,   1.226255,  -5.513570,  0.0690,  1.5118
H 129 water   43,  0.3345,  -4.874890,  -0.728009,  -7.375421,  0.0690,  1.5118
         ANAL 2
O 130 water   44, -0.6690, -11.305055,   0.463797,  -8.250657,  9.3005,  3.0236
H 131 water   44,  0.3345,  -9.642283,   0.806741,  -7.626864,  0.0690,  1.5118
H 132 water   44,  0.3345, -11.198987,   0.791088, -10.026387,  0.0690,  1.5118
         ANAL 2
O 133 water   45, -0.6690,   9.003924,   1.260468, -13.978052,  9.3005,  3.0236
H 134 water   45,  0.3345,   8.872949,   1.029479, -12.188880,  0.0690,  1.5118
H 135 water   45,  0.3345,   7.294259,   1.453381, -14.536071,  0.0690,  1.5118
         ANAL 2
O 136 water   46, -0.6690,   1.143175,  -2.768263,  -3.930866,  9.3005,  3.0236
H 137 water   46,  0.3345,   2.949715,  -2.781021,  -3.842132,  0.0690,  1.5118
H 138 water   46,  0.3345,   0.628519,  -1.739393,  -2.535107,  0.0690,  1.5118
         ANAL 2
O 139 water   47, -0.6690,  -5.215466,  -0.716794,   1.975589,  9.3005,  3.0236
H 140 water   47,  0.3345,  -5.881937,  -1.357549,   0.420961,  0.0690,  1.5118
H 141 water   47,  0.3345,  -5.825955,  -1.861526,   3.235942,  0.0690,  1.5118
         ANAL 2
O 142 water   48, -0.6690,  -5.420316,   3.662969,   4.478819,  9.3005,  3.0236
H 143 water   48,  0.3345,  -6.477348,   4.753717,   3.496694,  0.0690,  1.5118
H 144 water   48,  0.3345,  -5.283884,   2.154320,   3.490405,  0.0690,  1.5118
         ANAL 2
O 145 water   49, -0.6690, -10.890417,   3.434227,  -2.589015,  9.3005,  3.0236
H 146 water   49,  0.3345, -11.048285,   1.963503,  -3.630028,  0.0690,  1.5118
H 147 water   49,  0.3345, -12.483490,   3.562647,  -1.742125,  0.0690,  1.5118
         ANAL 2
```

```
O 148 water    50, -0.6690,   -5.775320,  11.853646,  -3.101847,  9.3005,  3.0236
H 149 water    50,  0.3345,   -7.420979,  11.268969,  -2.631135,  0.0690,  1.5118
H 150 water    50,  0.3345,   -5.594961,  13.436458,  -2.245210,  0.0690,  1.5118
        ANAL 2
O 151 water    51, -0.6690,    4.649786,   3.675426,  -3.783517,  9.3005,  3.0236
H 152 water    51,  0.3345,    5.304648,   2.161368,  -4.525414,  0.0690,  1.5118
H 153 water    51,  0.3345,    3.400165,   3.100381,  -2.609039,  0.0690,  1.5118
        ANAL 2
O 154 water    52, -0.6690,   -1.567977,   3.410131,  -9.464615,  9.3005,  3.0236
H 155 water    52,  0.3345,   -0.626480,   4.692294,  -8.603645,  0.0690,  1.5118
H 156 water    52,  0.3345,   -1.229747,   3.700148, -11.217631,  0.0690,  1.5118
        ANAL 2
O 157 water    53, -0.6690,   -5.851145,   5.947899,  -8.752976,  9.3005,  3.0236
H 158 water    53,  0.3345,   -4.364358,   4.937402,  -8.952909,  0.0690,  1.5118
H 159 water    53,  0.3345,   -5.233572,   7.643902,  -8.635725,  0.0690,  1.5118
        ANAL 2
O 160 water    54, -0.6690,   -5.554133,  -7.923600,   0.255548,  9.3005,  3.0236
H 161 water    54,  0.3345,   -4.383280,  -9.288428,   0.450222,  0.0690,  1.5118
H 162 water    54,  0.3345,   -6.017691,  -7.479959,   1.946682,  0.0690,  1.5118
        ANAL 2
O 163 water    55, -0.6690,  -10.801479,   4.924607, -18.428256,  9.3005,  3.0236
H 164 water    55,  0.3345,   -9.467718,   4.114442, -19.342785,  0.0690,  1.5118
H 165 water    55,  0.3345,  -12.082844,   3.656850, -18.278358,  0.0690,  1.5118
        ANAL 2
O 166 water    56, -0.6690,   11.847656,   6.046520,  -5.603357,  9.3005,  3.0236
H 167 water    56,  0.3345,   12.238226,   7.542824,  -4.665201,  0.0690,  1.5118
H 168 water    56,  0.3345,   10.145160,   6.293766,  -6.161903,  0.0690,  1.5118
        ANAL 2
O 169 water    57, -0.6690,    1.386876,  -3.561002,   5.147103,  9.3005,  3.0236
H 170 water    57,  0.3345,    2.355857,  -2.845566,   6.496472,  0.0690,  1.5118
H 171 water    57,  0.3345,    2.259957,  -5.087063,   4.722297,  0.0690,  1.5118
        ANAL 2
O 172 water    58, -0.6690,    9.401772,  -2.164511,   0.065934,  9.3005,  3.0236
H 173 water    58,  0.3345,    9.991920,  -0.603348,  -0.631289,  0.0690,  1.5118
H 174 water    58,  0.3345,   10.908511,  -3.099725,   0.421947,  0.0690,  1.5118
        ANAL 2
O 175 water    59, -0.6690,   -3.306352,  -3.510754,   7.521433,  9.3005,  3.0236
H 176 water    59,  0.3345,   -1.821460,  -3.757765,   6.518621,  0.0690,  1.5118
H 177 water    59,  0.3345,   -4.649769,  -4.184209,   6.514819,  0.0690,  1.5118
        ANAL 2
O 178 water    60, -0.6690,   -3.837669,  11.000091,  -8.559487,  9.3005,  3.0236
H 179 water    60,  0.3345,   -3.743740,  10.833060,  -6.760909,  0.0690,  1.5118
H 180 water    60,  0.3345,   -4.874401,  12.458215,  -8.825308,  0.0690,  1.5118
        ANAL 2
O 181 water    61, -0.6690,   -8.922772,   7.515155,  -5.130050,  9.3005,  3.0236
H 182 water    61,  0.3345,   -7.642400,   6.796299,  -6.186239,  0.0690,  1.5118
H 183 water    61,  0.3345,   -9.987497,   6.119150,  -4.695181,  0.0690,  1.5118
        ANAL 2
O 184 water    62, -0.6690,   -0.328799,   4.856437, -14.284939,  9.3005,  3.0236
H 185 water    62,  0.3345,   -0.201220,   6.618155, -13.895514,  0.0690,  1.5118
H 186 water    62,  0.3345,    1.105603,   4.519559, -15.334049,  0.0690,  1.5118
        ANAL 2
O 187 water    63, -0.6690,   -2.881192, -21.161439, -13.737737,  9.3005,  3.0236
H 188 water    63,  0.3345,   -1.096491, -20.881552, -13.827486,  0.0690,  1.5118
H 189 water    63,  0.3345,   -3.127890, -22.768721, -14.529801,  0.0690,  1.5118
        ANAL 2
O 190 water    64, -0.6690,    4.084301, -17.323502,   4.721371,  9.3005,  3.0236
H 191 water    64,  0.3345,    4.024132, -15.518282,   4.625783,  0.0690,  1.5118
```

```
H 192 water   64,  0.3345,   2.369338, -17.818891,   5.013135,  0.0690,  1.5118
         ANAL 2
O 193 water   65, -0.6690, -12.718566,   7.205589,   4.460795,  9.3005,  3.0236
H 194 water   65,  0.3345, -14.459989,   7.694159,   4.479550,  0.0690,  1.5118
H 195 water   65,  0.3345, -11.909758,   8.368880,   5.585148,  0.0690,  1.5118
         ANAL 2
O 196 water   66, -0.6690,   3.646309,  -5.777295,  11.936169,  9.3005,  3.0236
H 197 water   66,  0.3345,   4.697693,  -5.280701,  13.321640,  0.0690,  1.5118
H 198 water   66,  0.3345,   1.980185,  -5.852160,  12.636193,  0.0690,  1.5118
         ANAL 2
O 199 water   67, -0.6690, -11.582902,  -8.009770, -14.717441,  9.3005,  3.0236
H 200 water   67,  0.3345, -11.154529,  -6.961775, -13.306825,  0.0690,  1.5118
H 201 water   67,  0.3345, -13.310153,  -7.607196, -15.072626,  0.0690,  1.5118
         ANAL 2
O 202 water   68, -0.6690, -22.691319,  -4.602067, -11.338511,  9.3005,  3.0236
H 203 water   68,  0.3345, -22.222724,  -6.122637, -12.198701,  0.0690,  1.5118
H 204 water   68,  0.3345, -21.135438,  -3.704687, -11.125091,  0.0690,  1.5118
         ANAL 2
O 205 water   69, -0.6690,   5.964801,   4.513703,   3.703328,  9.3005,  3.0236
H 206 water   69,  0.3345,   5.242911,   3.066205,   2.893841,  0.0690,  1.5118
H 207 water   69,  0.3345,   4.736133,   5.001999,   4.937650,  0.0690,  1.5118
         ANAL 2
O 208 water   70, -0.6690,   4.568726,  15.366721,  -5.272902,  9.3005,  3.0236
H 209 water   70,  0.3345,   6.232090,  14.724940,  -5.577753,  0.0690,  1.5118
H 210 water   70,  0.3345,   4.568866,  17.006209,  -6.036869,  0.0690,  1.5118
         ANAL 2
O 211 water   71, -0.6690, -19.420245,   0.101012, -13.928108,  9.3005,  3.0236
H 212 water   71,  0.3345, -18.314060,  -1.071686, -13.107914,  0.0690,  1.5118
H 213 water   71,  0.3345, -20.713215,   0.433363, -12.707702,  0.0690,  1.5118
         ANAL 2
O 214 water   72, -0.6690,   4.028370,  -8.180716, -24.036693,  9.3005,  3.0236
H 215 water   72,  0.3345,   5.655124,  -8.965858, -24.130737,  0.0690,  1.5118
H 216 water   72,  0.3345,   3.080187,  -9.236487, -22.915137,  0.0690,  1.5118
         ANAL 2
O 217 water   73, -0.6690, -16.598416, -12.079831,  -3.027250,  9.3005,  3.0236
H 218 water   73,  0.3345, -16.867921, -10.478502,  -2.230557,  0.0690,  1.5118
H 219 water   73,  0.3345, -17.759278, -12.096633,  -4.414217,  0.0690,  1.5118
         ANAL 2
O 220 water   74, -0.6690,  -7.269666,  11.013178, -13.834325,  9.3005,  3.0236
H 221 water   74,  0.3345,  -8.505335,  11.789167, -12.765401,  0.0690,  1.5118
H 222 water   74,  0.3345,  -5.999215,  12.281945, -14.052989,  0.0690,  1.5118
         ANAL 2
O 223 water   75, -0.6690,  14.926078,   0.729892,  -0.880692,  9.3005,  3.0236
H 224 water   75,  0.3345,  14.684685,   0.444233,   0.888976,  0.0690,  1.5118
H 225 water   75,  0.3345,  13.432620,   1.604653,  -1.406003,  0.0690,  1.5118
         ANAL 2
O 226 water   76, -0.6690, -10.249930, -15.511937, -12.567035,  9.3005,  3.0236
H 227 water   76,  0.3345, -10.815694, -16.802051, -11.432530,  0.0690,  1.5118
H 228 water   76,  0.3345,  -8.811191, -16.233589, -13.392144,  0.0690,  1.5118
         ANAL 2
O 229 water   77, -0.6690,   6.153774,  -6.810349, -18.378764,  9.3005,  3.0236
H 230 water   77,  0.3345,   5.668826,  -7.601313, -16.826090,  0.0690,  1.5118
H 231 water   77,  0.3345,   7.802386,  -6.142324, -18.051050,  0.0690,  1.5118
         ANAL 2
O 232 water   78, -0.6690,   3.767930, -11.873194, -12.000036,  9.3005,  3.0236
H 233 water   78,  0.3345,   5.530162, -12.150411, -12.298786,  0.0690,  1.5118
H 234 water   78,  0.3345,   3.520665, -10.115524, -12.347997,  0.0690,  1.5118
         ANAL 2
```

```
O 235 water   79, -0.6690,   4.238655,   2.169696, -16.376549,  9.3005,  3.0236
H 236 water   79,  0.3345,   3.035922,   0.926926, -16.906254,  0.0690,  1.5118
H 237 water   79,  0.3345,   5.028595,   2.717314, -17.908763,  0.0690,  1.5118
         ANAL 2
O 238 water   80, -0.6690, -14.682710,   0.723814, -16.713445,  9.3005,  3.0236
H 239 water   80,  0.3345, -15.146156,  -0.558170, -17.902276,  0.0690,  1.5118
H 240 water   80,  0.3345, -16.240335,   1.191933, -15.922092,  0.0690,  1.5118
         ANAL 2
O 241 water   81, -0.6690,  -8.561719,  -1.140463, -21.112374,  9.3005,  3.0236
H 242 water   81,  0.3345,  -7.247277,   0.101635, -21.143972,  0.0690,  1.5118
H 243 water   81,  0.3345,  -7.967217,  -2.456683, -22.201304,  0.0690,  1.5118
         ANAL 2
O 244 water   82, -0.6690,   3.098039, -19.304371,  -4.729504,  9.3005,  3.0236
H 245 water   82,  0.3345,   4.475916, -18.539274,  -3.842006,  0.0690,  1.5118
H 246 water   82,  0.3345,   2.316075, -17.936378,  -5.617621,  0.0690,  1.5118
         ANAL 2
O 247 water   83, -0.6690, -11.481971,  -2.627730,   8.985770,  9.3005,  3.0236
H 248 water   83,  0.3345, -12.101813,  -1.963571,   7.421693,  0.0690,  1.5118
H 249 water   83,  0.3345, -12.537175,  -4.058239,   9.320132,  0.0690,  1.5118
         ANAL 2
O 250 water   84, -0.6690, -15.326832,   4.993788,   0.715203,  9.3005,  3.0236
H 251 water   84,  0.3345, -16.282318,   3.624931,   1.411550,  0.0690,  1.5118
H 252 water   84,  0.3345, -14.247083,   5.525313,   2.065484,  0.0690,  1.5118
         ANAL 2
O 253 water   85, -0.6690,   4.699246,  -3.093313, -22.212181,  9.3005,  3.0236
H 254 water   85,  0.3345,   4.440051,  -3.027220, -24.001039,  0.0690,  1.5118
H 255 water   85,  0.3345,   4.907822,  -4.853028, -21.849577,  0.0690,  1.5118
         ANAL 2
O 256 water   86, -0.6690,   5.481254, -14.871569,  -2.300088,  9.3005,  3.0236
H 257 water   86,  0.3345,   6.047930, -15.537006,  -0.716518,  0.0690,  1.5118
H 258 water   86,  0.3345,   4.081230, -13.810063,  -1.870250,  0.0690,  1.5118
         ANAL 2
O 259 water   87, -0.6690,   7.716927,   7.491705,  -0.281469,  9.3005,  3.0236
H 260 water   87,  0.3345,   7.150105,   7.474635,   1.436084,  0.0690,  1.5118
H 261 water   87,  0.3345,   6.704457,   6.239316,  -1.104898,  0.0690,  1.5118
         ANAL 2
O 262 water   88, -0.6690,  -0.099717,  10.178584, -12.503376,  9.3005,  3.0236
H 263 water   88,  0.3345,   1.433195,  10.526454, -11.608534,  0.0690,  1.5118
H 264 water   88,  0.3345,  -1.386488,  10.226885, -11.233132,  0.0690,  1.5118
         ANAL 2
O 265 water   89, -0.6690,   0.174031, -13.358407, -17.219676,  9.3005,  3.0236
H 266 water   89,  0.3345,  -1.538859, -13.329107, -16.639342,  0.0690,  1.5118
H 267 water   89,  0.3345,   1.160347, -13.516677, -15.711781,  0.0690,  1.5118
         ANAL 2
O 268 water   90, -0.6690, -12.876200,  -6.358544,   5.051406,  9.3005,  3.0236
H 269 water   90,  0.3345, -11.965601,  -7.839840,   5.549604,  0.0690,  1.5118
H 270 water   90,  0.3345, -12.072305,  -5.826163,   3.521066,  0.0690,  1.5118
         ANAL 2
O 271 water   91, -0.6690,  10.374968,   3.013642,  -1.655659,  9.3005,  3.0236
H 272 water   91,  0.3345,  10.560607,   4.591503,  -0.791107,  0.0690,  1.5118
H 273 water   91,  0.3345,  10.505731,   3.439691,  -3.408656,  0.0690,  1.5118
         ANAL 2
O 274 water   92, -0.6690, -10.546723,  11.487361,  -1.141891,  9.3005,  3.0236
H 275 water   92,  0.3345, -10.877951,   9.851355,  -1.838588,  0.0690,  1.5118
H 276 water   92,  0.3345, -11.173562,  12.643652,  -2.383534,  0.0690,  1.5118
         ANAL 2
O 277 water   93, -0.6690,  16.771883,   5.466609,  -7.085057,  9.3005,  3.0236
H 278 water   93,  0.3345,  15.059466,   5.871878,  -6.666783,  0.0690,  1.5118
```

```
H 279 water   93,  0.3345,  17.622368,   5.393423,  -5.490399,  0.0690,  1.5118
         ANAL 2
O 280 water   94, -0.6690,   3.754576,  11.312079,  -9.304066,  9.3005,  3.0236
H 281 water   94,  0.3345,   3.860036,  12.752066,  -8.214605,  0.0690,  1.5118
H 282 water   94,  0.3345,   3.061607,  10.005884,  -8.262319,  0.0690,  1.5118
         ANAL 2
O 283 water   95, -0.6690, -12.292259,   3.567992,  15.558774,  9.3005,  3.0236
H 284 water   95,  0.3345, -11.551322,   2.039880,  14.936264,  0.0690,  1.5118
H 285 water   95,  0.3345, -13.884701,   3.052556,  16.244347,  0.0690,  1.5118
         ANAL 2
O 286 water   96, -0.6690,  -6.101086, -22.917487,  -8.300376,  9.3005,  3.0236
H 287 water   96,  0.3345,  -4.397039, -23.519615,  -8.227759,  0.0690,  1.5118
H 288 water   96,  0.3345,  -6.818504, -23.760130,  -9.731051,  0.0690,  1.5118
         ANAL 2
O 289 water   97, -0.6690, -16.474111, -10.067162, -10.990268,  9.3005,  3.0236
H 290 water   97,  0.3345, -16.578249,  -8.410617, -11.709043,  0.0690,  1.5118
H 291 water   97,  0.3345, -14.818156, -10.647769, -11.428823,  0.0690,  1.5118
         ANAL 2
O 292 water   98, -0.6690, -19.797465,  -9.667636,  -6.291155,  9.3005,  3.0236
H 293 water   98,  0.3345, -19.300393,  -9.924127,  -8.011250,  0.0690,  1.5118
H 294 water   98,  0.3345, -20.691511,  -8.095365,  -6.307828,  0.0690,  1.5118
         ANAL 2
O 295 water   99, -0.6690,  -5.173425, -12.712808, -15.395169,  9.3005,  3.0236
H 296 water   99,  0.3345,  -6.298389, -11.960535, -16.595239,  0.0690,  1.5118
H 297 water   99,  0.3345,  -5.886903, -12.300045, -13.785150,  0.0690,  1.5118
         ANAL 2
O 298 water  100, -0.6690,  -0.985868,   2.698559, -21.389839,  9.3005,  3.0236
H 299 water  100,  0.3345,  -2.772026,   2.967796, -21.483095,  0.0690,  1.5118
H 300 water  100,  0.3345,  -0.804982,   1.203436, -20.388070,  0.0690,  1.5118
         ANAL 2
O 301 water  101, -0.6690,  -6.036941,   3.692122, -20.942100,  9.3005,  3.0236
H 302 water  101,  0.3345,  -6.140712,   5.167312, -21.983557,  0.0690,  1.5118
H 303 water  101,  0.3345,  -5.389527,   4.295988, -19.364810,  0.0690,  1.5118
         ANAL 2
O 304 water  102, -0.6690,   8.569219,  -3.794649,   6.339855,  9.3005,  3.0236
H 305 water  102,  0.3345,   9.506348,  -2.259851,   6.534192,  0.0690,  1.5118
H 306 water  102,  0.3345,   9.732446,  -5.097927,   6.808906,  0.0690,  1.5118
         ANAL 2
O 307 water  103, -0.6690,   1.629116,  13.174149,  -1.603484,  9.3005,  3.0236
H 308 water  103,  0.3345,   2.665944,  13.034466,  -0.127976,  0.0690,  1.5118
H 309 water  103,  0.3345,   2.665668,  14.042584,  -2.804722,  0.0690,  1.5118
         ANAL 2
O 310 water  104, -0.6690,  -1.793830,  17.039669,  -2.196744,  9.3005,  3.0236
H 311 water  104,  0.3345,  -0.532011,  15.757282,  -2.009931,  0.0690,  1.5118
H 312 water  104,  0.3345,  -1.427265,  17.806465,  -3.793375,  0.0690,  1.5118
         ANAL 2
O 313 water  105, -0.6690,  -7.668774, -18.374941,  -6.401296,  9.3005,  3.0236
H 314 water  105,  0.3345,  -8.484265, -18.612930,  -4.804439,  0.0690,  1.5118
H 315 water  105,  0.3345,  -7.099674, -20.026694,  -6.869723,  0.0690,  1.5118
         ANAL 2
O 316 water  106, -0.6690,  10.403971,  -3.097182, -22.166270,  9.3005,  3.0236
H 317 water  106,  0.3345,   8.729740,  -2.493267, -21.844085,  0.0690,  1.5118
H 318 water  106,  0.3345,  11.118103,  -1.873618, -23.290792,  0.0690,  1.5118
         ANAL 2
O 319 water  107, -0.6690, -15.303927,   1.380876,   7.300949,  9.3005,  3.0236
H 320 water  107,  0.3345, -14.621808,   2.684836,   8.352602,  0.0690,  1.5118
H 321 water  107,  0.3345, -16.432203,   0.470608,   8.382611,  0.0690,  1.5118
         ANAL 2
```

```
O 322 water  108, −0.6690,   8.586125,  −8.600562,  −9.057275,  9.3005,  3.0236
H 323 water  108,  0.3345,   8.704894,  −9.813188, −10.394062,  0.0690,  1.5118
H 324 water  108,  0.3345,   8.628559,  −9.589696,  −7.543525,  0.0690,  1.5118
        ANAL 2
O 325 water  109, −0.6690,  16.039129,   1.515730, −10.242663,  9.3005,  3.0236
H 326 water  109,  0.3345,  15.156300,   2.196578, −11.666968,  0.0690,  1.5118
H 327 water  109,  0.3345,  16.442442,   2.957895,  −9.228231,  0.0690,  1.5118
        ANAL 2
O 328 water  110, −0.6690, −15.024905,   4.616965,  −7.229413,  9.3005,  3.0236
H 329 water  110,  0.3345, −13.962809,   5.310927,  −8.518592,  0.0690,  1.5118
H 330 water  110,  0.3345, −14.288526,   3.008247,  −6.853418,  0.0690,  1.5118
        ANAL 2
O 331 water  111, −0.6690,  −3.380516, −15.912024,  −1.000610,  9.3005,  3.0236
H 332 water  111,  0.3345,  −4.224093, −15.041864,  −2.343279,  0.0690,  1.5118
H 333 water  111,  0.3345,  −1.903825, −16.614467,  −1.773642,  0.0690,  1.5118
        ANAL 2
O 334 water  112, −0.6690, −15.260480,   6.494595, −20.095316,  9.3005,  3.0236
H 335 water  112,  0.3345, −13.528966,   6.244388, −19.636069,  0.0690,  1.5118
H 336 water  112,  0.3345, −15.477657,   5.509392, −21.596599,  0.0690,  1.5118
        ANAL 2
O 337 water  113, −0.6690, −18.039071,  −8.431035,   0.810660,  9.3005,  3.0236
H 338 water  113,  0.3345, −19.833638,  −8.649204,   0.751210,  0.0690,  1.5118
H 339 water  113,  0.3345, −17.604258,  −8.725109,   2.541579,  0.0690,  1.5118
        ANAL 2
O 340 water  114, −0.6690,  −1.661337, −24.854339,  −7.595623,  9.3005,  3.0236
H 341 water  114,  0.3345,  −0.939122, −26.256081,  −6.709534,  0.0690,  1.5118
H 342 water  114,  0.3345,  −2.163785, −25.538570,  −9.192816,  0.0690,  1.5118
        ANAL 2
O 343 water  115, −0.6690, −13.114116,   4.624061,  10.656335,  9.3005,  3.0236
H 344 water  115,  0.3345, −12.770746,   4.110895,  12.356450,  0.0690,  1.5118
H 345 water  115,  0.3345, −11.486046,   4.717057,   9.873830,  0.0690,  1.5118
        ANAL 2
O 346 water  116, −0.6690, −12.415589,  11.848047,  −6.994473,  9.3005,  3.0236
H 347 water  116,  0.3345, −11.301690,  10.449794,  −6.719342,  0.0690,  1.5118
H 348 water  116,  0.3345, −12.471232,  12.035653,  −8.792606,  0.0690,  1.5118
        ANAL 2
O 349 water  117, −0.6690,  −6.990293,  15.098132,  −9.148407,  9.3005,  3.0236
H 350 water  117,  0.3345,  −8.451161,  15.701720, −10.027680,  0.0690,  1.5118
H 351 water  117,  0.3345,  −7.148900,  15.784293,  −7.482385,  0.0690,  1.5118
        ANAL 2
O 352 water  118, −0.6690, −12.346254,  10.561505, −12.238735,  9.3005,  3.0236
H 353 water  118,  0.3345, −11.449591,   8.995101, −12.357033,  0.0690,  1.5118
H 354 water  118,  0.3345, −12.498207,  11.126631, −13.950212,  0.0690,  1.5118
        ANAL 2
O 355 water  119, −0.6690,  11.408034,  −1.036518,  −9.303117,  9.3005,  3.0236
H 356 water  119,  0.3345,  13.124204,  −0.467286,  −9.351674,  0.0690,  1.5118
H 357 water  119,  0.3345,  11.316274,  −2.308132, −10.586156,  0.0690,  1.5118
        ANAL 2
O 358 water  120, −0.6690,  −5.118967,  −0.122532,  11.023945,  9.3005,  3.0236
H 359 water  120,  0.3345,  −3.926828,   1.229465,  10.874033,  0.0690,  1.5118
H 360 water  120,  0.3345,  −4.376581,  −1.485878,  10.095642,  0.0690,  1.5118
        ANAL 2
O 361 water  121, −0.6690,  −9.994751,  −0.336500,  13.264888,  9.3005,  3.0236
H 362 water  121,  0.3345,  −8.344869,   0.245734,  12.806085,  0.0690,  1.5118
H 363 water  121,  0.3345, −10.596094,  −1.198957,  11.793118,  0.0690,  1.5118
        ANAL 2
O 364 water  122, −0.6690,  −2.302168,  10.694993,   2.997228,  9.3005,  3.0236
H 365 water  122,  0.3345,  −1.936062,  10.018305,   1.360257,  0.0690,  1.5118
```

```
H 366 water  122,  0.3345,  -4.049239,  11.156275,   2.916133,  0.0690,  1.5118
         ANAL 2
O 367 water  123, -0.6690,  13.022958,  -5.274418,   1.238475,  9.3005,  3.0236
H 368 water  123,  0.3345,  14.310336,  -4.634829,   0.140652,  0.0690,  1.5118
H 369 water  123,  0.3345,  13.911524,  -5.785670,   2.728666,  0.0690,  1.5118
         ANAL 2
O 370 water  124, -0.6690,   7.566591,   6.942112,  -9.525069,  9.3005,  3.0236
H 371 water  124,  0.3345,   8.221676,   5.879246, -10.833782,  0.0690,  1.5118
H 372 water  124,  0.3345,   6.672252,   8.244973, -10.405011,  0.0690,  1.5118
         ANAL 2
O 373 water  125, -0.6690,  -2.177550,   7.035558,   6.401299,  9.3005,  3.0236
H 374 water  125,  0.3345,  -2.965515,   5.658507,   5.532692,  0.0690,  1.5118
H 375 water  125,  0.3345,  -2.216274,   8.400616,   5.215260,  0.0690,  1.5118
         ANAL 2
O 376 water  126, -0.6690,   9.007883,  12.608974,  -5.620298,  9.3005,  3.0236
H 377 water  126,  0.3345,  10.262459,  11.860684,  -4.553658,  0.0690,  1.5118
H 378 water  126,  0.3345,   7.974551,  11.220836,  -6.146547,  0.0690,  1.5118
         ANAL 2
O 379 water  127, -0.6690,  -1.447855,  -3.724807, -24.991095,  9.3005,  3.0236
H 380 water  127,  0.3345,   0.236839,  -4.371458, -25.114712,  0.0690,  1.5118
H 381 water  127,  0.3345,  -1.920631,  -4.018901, -23.270180,  0.0690,  1.5118
         ANAL 2
O 382 water  128, -0.6690,  -7.648252,  12.615523,   3.065177,  9.3005,  3.0236
H 383 water  128,  0.3345,  -7.811672,  14.380624,   3.424766,  0.0690,  1.5118
H 384 water  128,  0.3345,  -8.836905,  12.312256,   1.735989,  0.0690,  1.5118
         ANAL 2
END

eor
```

Example: DRF: scripting tool

Download DIMQM_DRF_plams.run

```
#! /bin/sh

# The Discrete Reaction Field (DRF) method is a polarizable QM/MM method. For
# each type of solvent molecule,the DRF method requires two parameters, atomic
# polarizabilities and atomic charges. The main program reads the QM and DRF
# regions, splits the DRF region into unique molecules, computes the atomic
# charges of a single DRF molecule for each unique type, assigns the atomic
# charges and atomic polarizabilities for all the DRF molecules and defines the
# QM_DRF settings. This simple script works for initial configurations that come
# from a MD  simulation, for example.

# In this example a python script is used for DRF calculations, that uses PLAMS
# (Python Library for Automating Molecular Simulation). PLAMS takes care of
# input preparation, job execution, file management and output processing, see
# also the PLAMS documentation in the Scripting manual. The DRF python script
# has default settings which can be easily modified if this is required. In the
# simplest case the user just needs to provide the QM and DRF regions (in xyz
# format). Typically the user may also want to choose the functional, basis set
# and numerical quality for the QM and DRF region. By default, atomic charges
# that are used for the DRF are computed with LDA functional, DZP basis set and
# normal numerical quality. Also by default, atomic polarizabilities (computed
# with the Thole's model) are taken from a inner database including H, C, N, O,
# F, S, Cl, Br, I atoms [146]. The main program defines a series of functions
```

```
# for general settings for a DRF calculation, reading QM and DRF regions,
# classifying DRF molecules of the same type (first by formula and then for
# configurations for a given formula), computing, for a given settings, the
# charges for a single molecule of unique DRF type and assigning the atomic
# charges for the rest of molecules that belong to a given DRF type (preserving
# atom orders).

# The DRF python script DIMQM_DRF_plams.py users will probably want to change or
# extend:

# ============================================================================
# from DIMQM_DRF_plams_module import full_DIMQM_DRF

# qm_region_filename = 'qm.xyz'
# drf_region_filename = 'drf.xyz'

# settings = Settings()
# settings.input.Basis.Type = 'TZP'
# settings.input.XC['_1'] = 'GGA BLYP'
# settings.input.NumericalQuality = 'Normal'

# full_DIMQM_DRF(qm_region_filename, drf_region_filename,
#                settings, drf_charges_type='MDC-q charges',
#                drf_fragment_settings=None)
# ============================================================================


# This DRF python script DIMQM_DRF_plams.py imports DIMQM_DRF_plams_module.py,
# which contains a script full_DIMQM_DRF that splits the DRF region into unique
# molecules, computes the atomic charges of a single DRF molecule for each
# unique type, assignes the atomic charges and atomic polarizabilities for all
# the DRF molecules and defines the QM_DRF settings.

# The run script DIMQM_DRF_plams.run contains an example, how one could use the
# DIMQM_DRF_plams.py script. In this simple example both the QM and DRF region
# consists of a Formic acid molecule.

cp $ADFHOME/examples/adf/DIMQM_DRF_plams/formic_acid_qm.xyz qm.xyz
cp $ADFHOME/examples/adf/DIMQM_DRF_plams/formic_acid_drf.xyz drf.xyz
cp $ADFHOME/examples/adf/DIMQM_DRF_plams/DIMQM_DRF_plams_module.py .
cp $ADFHOME/examples/adf/DIMQM_DRF_plams/DIMQM_DRF_plams.py .

$ADFBIN/plams DIMQM_DRF_plams.py

# Every time you run a PLAMS script, a uniquely named working directory is
# created (plams.*****). This folder will contain one subdirectory per job. Each
# job directory contains the job's input and results files. In this case there
# will be 1 directory for the ADF calculation of the atomic charges that are
# used for DRF. Next there is a directory which calculates the full system.
```

Example: DRF2: Polarizability N2 on Ag68 + H2O

```
Download DIMQM_DRF2.run
```

Example: CPIM: excitation energies N2 on silver cluster Ag68

Download DIMQM_CPIM_Excitations.run

```sh
#! /bin/sh

# == Expert option ==

# This example demonstrates how to calculate excitation energies with the
# Capacitance Polarizability Interaction Model (CPIM). Key EXCITATIONS is used.


$ADFBIN/adf <<eor
TITLE DIM/QM Excitations Test, N2 on Ag68

ATOMS
N     -0.000006    -0.000060     2.302488
N      0.000019     0.000041     3.408628
END

BASIS
 type TZP
 core none
END

SYMMETRY NOSYM

BeckeGrid
  quality good
End

XC
 GGA Becke Perdew
END

EXCITATIONS
 davidson
End
 ALLOWED
 LOWEST 10
END

DIMQM
 CPIM
END

DIMPAR
Ag
 pol 49.9843
 cap 2.7529
SUBEND
XYZ
68
Ag     -1.2626801699    -1.1212553334    -7.5641602542
Ag     -1.5751210922     0.5238327185    -9.9892089940
Ag     -4.1814870442    -1.7663886735    -7.6050834452
Ag     -0.8822430837    -2.7600722440    -5.1959892962
Ag     -2.1240266195    -3.8206731623    -2.8525807753
```

```
Ag       3.4578336576    -2.6744062081    -2.8525742581
Ag      -3.7149452090    -3.2988294197    -5.2034646499
Ag       1.8985756093    -2.1890139391    -5.1959881717
Ag       0.6946811325    -3.3828052555    -2.8061234013
Ag       4.7143519584    -1.5678241605    -5.2034549930
Ag       1.6023915608    -0.5328957432    -7.5641613132
Ag       1.2412345786     1.1021928389    -9.9892023582
Ag       4.5392237030     0.0244593089    -7.6050682929
Ag       0.0000106997     0.0000013921    -5.1462708001
Ag      -0.3396832259     1.6541471619    -7.5641597129
Ag      -2.8450227342    -0.5497019461    -5.1959955618
Ag       2.8314207151     0.6159878635    -5.1959875699
Ag      -3.1884901403     1.0603809604    -7.5634515357
Ag      -1.2364094731    -1.0979217456    -2.7987978362
Ag       1.5690420034    -0.5218070047    -2.7987993856
Ag       4.3708278508     0.0708740258    -2.8525785565
Ag      -4.0450216512    -1.6573690296    -2.8525892410
Ag       2.5125751829     2.2311280445    -7.5634444475
Ag       0.6759414795    -3.2915128984    -7.5634430672
Ag       0.3339235029    -1.6260239766    -9.9892067983
Ag       1.0324285615    -5.0274542660    -5.1718100987
Ag      -2.2484187072    -3.9433040680    -7.6050830302
Ag       3.6204971713    -2.7380910382    -7.6050765656
Ag      -2.4569103660    -2.1817169842    -9.8785349524
Ag      -1.7752961456    -5.4191572521    -5.2192070180
Ag       3.7672926986    -4.2809622136    -5.2191965533
Ag       3.1179002480    -1.0369024811    -9.8785348598
Ag      -1.9491576017     2.1440839896    -5.1959915042
Ag       2.5822650861     2.2930127465    -2.8061239177
Ag      -0.3326165571     1.6197266638    -2.7987957984
Ag       0.9464626101     2.7387205045    -5.1959886567
Ag      -0.5598334280    -4.3819335958    -9.8593641366
Ag      -0.2204924851    -6.0299810786    -7.5512447020
Ag       2.5790329655    -5.4550834512    -7.5512427315
Ag       2.2414383913    -3.8066764388    -9.8593649949
Ag      -2.2907828197     3.9188443502    -7.6050826773
Ag       0.5610225957     4.5044771077    -7.6050819796
Ag      -4.8701069022     1.6196298364    -5.1718125544
Ag      -3.2769317991     1.0897966386    -2.8061340012
Ag      -0.6609450333     3.2186205924    -9.8785392784
Ag      -5.5910570566    -1.1220840757    -5.2192067211
Ag      -2.2467815447     3.7498049589    -2.8525803584
Ag      -0.9993904347     4.8666585643    -5.2034637574
Ag      -3.8054729020     4.2470446290    -5.2192106249
Ag      -3.5149221622     2.6758039407    -9.8593694452
Ag      -4.4173760657    -0.0377981351    -9.8593712497
Ag      -6.0137475245     0.4940463872    -7.5512522368
Ag      -5.1118586671     3.2059431066    -7.5512501894
Ag       0.5871922586     4.3317735603    -2.8525794711
Ag       1.8237836111     5.4030432621    -5.2192000459
Ag       3.8377073288     3.4078339914    -5.1718059272
Ag       5.5807980727     1.1721205307    -5.2191937875
Ag       3.4347425354     4.9610463596    -7.5512390021
Ag       4.0747942310     1.7061247432    -9.8593564188
Ag       5.3323750373     2.8240364209    -7.5512331190
Ag       2.1759755136     3.8444690680    -9.8593641559
Ag      -1.8817544351     2.1073913325    -0.4836595729
Ag       0.8989635719     2.6784284394    -0.4836553279
```

```
Ag      0.0000000000    0.0000000000    0.0000000000
Ag      2.7659350940    0.5759533343   -0.4836497308
Ag     -2.7690678527   -0.5606913044   -0.4836604335
Ag     -0.8841719671   -2.6833451220   -0.4836610364
Ag      1.8701101821   -2.1177433283   -0.4836551986
SUBEND
END


eor
```

Example: CPIM: polarizability N2 on silver cluster Ag68

Download DIMQM_JACOBI.run

```
#! /bin/sh

# == Expert option ==

# This example demonstrates how to do calculate static and frequency dependent
# polarizabilities with the Polarizability Interaction Model (PIM). The example
# consists of 3 calculations: static polarizability, frequency dependent, and
# with inclusion of effect of the local field.

# When the molecule interacts with a metal nanoparticle, there are two types of
# interactions: the image field and the local field. The image field is caused
# by the dipoles induced into the nanoparticle by the molecule's electron
# density. This is always taken into account in a DIM/QM calculation. The local
# field arises by direct interactions of the nanoparticle with an external
# field. Addition of the LOCALFIELD key causes the DIM/QM calculation to include
# this effect, but by default this is not included in a DIM/QM calculation.


#######################
# STATIC POLARIZABILITY
#######################

$ADFBIN/adf <<eor
TITLE PIM Static Polarizability, N2 on Ag68

ATOMS
N     -0.000006    -0.000060     2.302488
N      0.000019     0.000041     3.408628
END

BASIS
 type TZP
 core none
END

SYMMETRY NOSYM

BeckeGrid
 Quality good
End

XC
 GGA Becke Perdew
```

```
END

AORESPONSE
 ALDA
END

DIMQM
 PIM
 JACOBI
 ALGORITHM BRUTE
END

DIMPAR
Ag
 rad 1.4445
 drude 9.2 0.17 eV
 bound 5.0
SUBEND
XYZ
68
Ag    -1.2626801699   -1.1212553334   -7.5641602542
Ag    -1.5751210922    0.5238327185   -9.9892089940
Ag    -4.1814870442   -1.7663886735   -7.6050834452
Ag    -0.8822430837   -2.7600722440   -5.1959892962
Ag    -2.1240266195   -3.8206731623   -2.8525807753
Ag     3.4578336576   -2.6744062081   -2.8525742581
Ag    -3.7149452090   -3.2988294197   -5.2034646499
Ag     1.8985756093   -2.1890139391   -5.1959881717
Ag     0.6946811325   -3.3828052555   -2.8061234013
Ag     4.7143519584   -1.5678241605   -5.2034549930
Ag     1.6023915608   -0.5328957432   -7.5641613132
Ag     1.2412345786    1.1021928389   -9.9892023582
Ag     4.5392237030    0.0244593089   -7.6050682929
Ag     0.0000106997    0.0000013921   -5.1462708001
Ag    -0.3396832259    1.6541471619   -7.5641597129
Ag    -2.8450227342   -0.5497019461   -5.1959955618
Ag     2.8314207151    0.6159878635   -5.1959875699
Ag    -3.1884901403    1.0603809604   -7.5634515357
Ag    -1.2364094731   -1.0979217456   -2.7987978362
Ag     1.5690420034   -0.5218070047   -2.7987993856
Ag     4.3708278508    0.0708740258   -2.8525785565
Ag    -4.0450216512   -1.6573690296   -2.8525892410
Ag     2.5125751829    2.2311280445   -7.5634444475
Ag     0.6759414795   -3.2915128984   -7.5634430672
Ag     0.3339235029   -1.6260239766   -9.9892067983
Ag     1.0324285615   -5.0274542660   -5.1718100987
Ag    -2.2484187072   -3.9433040680   -7.6050830302
Ag     3.6204971713   -2.7380910382   -7.6050765656
Ag    -2.4569103660   -2.1817169842   -9.8785349524
Ag    -1.7752961456   -5.4191572521   -5.2192070180
Ag     3.7672926986   -4.2809622136   -5.2191965533
Ag     3.1179002480   -1.0369024811   -9.8785348598
Ag    -1.9491576017    2.1440839896   -5.1959915042
Ag     2.5822650861    2.2930127465   -2.8061239177
Ag    -0.3326165571    1.6197266638   -2.7987957984
Ag     0.9464626101    2.7387205045   -5.1959886567
Ag    -0.5598334280   -4.3819335958   -9.8593641366
Ag    -0.2204924851   -6.0299810786   -7.5512447020
```

```
Ag       2.5790329655   -5.4550834512   -7.5512427315
Ag       2.2414383913   -3.8066764388   -9.8593649949
Ag      -2.2907828197    3.9188443502   -7.6050826773
Ag       0.5610225957    4.5044771077   -7.6050819796
Ag      -4.8701069022    1.6196298364   -5.1718125544
Ag      -3.2769317991    1.0897966386   -2.8061340012
Ag      -0.6609450333    3.2186205924   -9.8785392784
Ag      -5.5910570566   -1.1220840757   -5.2192067211
Ag      -2.2467815447    3.7498049589   -2.8525803584
Ag      -0.9993904347    4.8666585643   -5.2034637574
Ag      -3.8054729020    4.2470446290   -5.2192106249
Ag      -3.5149221622    2.6758039407   -9.8593694452
Ag      -4.4173760657   -0.0377981351   -9.8593712497
Ag      -6.0137475245    0.4940463872   -7.5512522368
Ag      -5.1118586671    3.2059431066   -7.5512501894
Ag       0.5871922586    4.3317735603   -2.8525794711
Ag       1.8237836111    5.4030432621   -5.2192000459
Ag       3.8377073288    3.4078339914   -5.1718059272
Ag       5.5807980727    1.1721205307   -5.2191937875
Ag       3.4347425354    4.9610463596   -7.5512390021
Ag       4.0747942310    1.7061247432   -9.8593564188
Ag       5.3323750373    2.8240364209   -7.5512331190
Ag       2.1759755136    3.8444690680   -9.8593641559
Ag      -1.8817544351    2.1073913325   -0.4836595729
Ag       0.8989635719    2.6784284394   -0.4836553279
Ag       0.0000000000    0.0000000000    0.0000000000
Ag       2.7659350940    0.5759533343   -0.4836497308
Ag      -2.7690678527   -0.5606913044   -0.4836604335
Ag      -0.8841719671   -2.6833451220   -0.4836610364
Ag       1.8701101821   -2.1177433283   -0.4836551986
SUBEND
END


eor


rm TAPE21 logfile


###################################
# FREQUENCY DEPENDENT POLARIZABILITY
###################################


$ADFBIN/adf <<eor
TITLE PIM Frequency-Dependent Polarizability, N2 on Ag68


ATOMS
N      -0.000006    -0.000060     2.302488
N       0.000019     0.000041     3.408628
END


BASIS
 type TZP
 core none
END


SYMMETRY NOSYM


BeckeGrid
 Quality good
```

```
End

XC
 GGA Becke Perdew
END

AORESPONSE
:: The FREQUENCY is at the Plasmon Max
 FREQUENCY 1 3.55 eV
 LIFETIME 0.0036749
 ALDA
END

DIMQM
 PIM
 JACOBI
 ALGORITHM BRUTE
:: Frequency-Dependent
 FREQUENCY
END

DIMPAR
Ag
 rad 1.4445
 drude 9.2 0.17 eV
 bound 5.0
SUBEND
XYZ
68
Ag     -1.2626801699   -1.1212553334   -7.5641602542
Ag     -1.5751210922    0.5238327185   -9.9892089940
Ag     -4.1814870442   -1.7663886735   -7.6050834452
Ag     -0.8822430837   -2.7600722440   -5.1959892962
Ag     -2.1240266195   -3.8206731623   -2.8525807753
Ag      3.4578336576   -2.6744062081   -2.8525742581
Ag     -3.7149452090   -3.2988294197   -5.2034646499
Ag      1.8985756093   -2.1890139391   -5.1959881717
Ag      0.6946811325   -3.3828052555   -2.8061234013
Ag      4.7143519584   -1.5678241605   -5.2034549930
Ag      1.6023915608   -0.5328957432   -7.5641613132
Ag      1.2412345786    1.1021928389   -9.9892023582
Ag      4.5392237030    0.0244593089   -7.6050682929
Ag      0.0000106997    0.0000013921   -5.1462708001
Ag     -0.3396832259    1.6541471619   -7.5641597129
Ag     -2.8450227342   -0.5497019461   -5.1959955618
Ag      2.8314207151    0.6159878635   -5.1959875699
Ag     -3.1884901403    1.0603809604   -7.5634515357
Ag     -1.2364094731   -1.0979217456   -2.7987978362
Ag      1.5690420034   -0.5218070047   -2.7987993856
Ag      4.3708278508    0.0708740258   -2.8525785565
Ag     -4.0450216512   -1.6573690296   -2.8525892410
Ag      2.5125751829    2.2311280445   -7.5634444475
Ag      0.6759414795   -3.2915128984   -7.5634430672
Ag      0.3339235029   -1.6260239766   -9.9892067983
Ag      1.0324285615   -5.0274542660   -5.1718100987
Ag     -2.2484187072   -3.9433040680   -7.6050830302
Ag      3.6204971713   -2.7380910382   -7.6050765656
Ag     -2.4569103660   -2.1817169842   -9.8785349524
```

```
Ag      -1.7752961456    -5.4191572521    -5.2192070180
Ag       3.7672926986    -4.2809622136    -5.2191965533
Ag       3.1179002480    -1.0369024811    -9.8785348598
Ag      -1.9491576017     2.1440839896    -5.1959915042
Ag       2.5822650861     2.2930127465    -2.8061239177
Ag      -0.3326165571     1.6197266638    -2.7987957984
Ag       0.9464626101     2.7387205045    -5.1959886567
Ag      -0.5598334280    -4.3819335958    -9.8593641366
Ag      -0.2204924851    -6.0299810786    -7.5512447020
Ag       2.5790329655    -5.4550834512    -7.5512427315
Ag       2.2414383913    -3.8066764388    -9.8593649949
Ag      -2.2907828197     3.9188443502    -7.6050826773
Ag       0.5610225957     4.5044771077    -7.6050819796
Ag      -4.8701069022     1.6196298364    -5.1718125544
Ag      -3.2769317991     1.0897966386    -2.8061340012
Ag      -0.6609450333     3.2186205924    -9.8785392784
Ag      -5.5910570566    -1.1220840757    -5.2192067211
Ag      -2.2467815447     3.7498049589    -2.8525803584
Ag      -0.9993904347     4.8666585643    -5.2034637574
Ag      -3.8054729020     4.2470446290    -5.2192106249
Ag      -3.5149221622     2.6758039407    -9.8593694452
Ag      -4.4173760657    -0.0377981351    -9.8593712497
Ag      -6.0137475245     0.4940463872    -7.5512522368
Ag      -5.1118586671     3.2059431066    -7.5512501894
Ag       0.5871922586     4.3317735603    -2.8525794711
Ag       1.8237836111     5.4030432621    -5.2192000459
Ag       3.8377073288     3.4078339914    -5.1718059272
Ag       5.5807980727     1.1721205307    -5.2191937875
Ag       3.4347425354     4.9610463596    -7.5512390021
Ag       4.0747942310     1.7061247432    -9.8593564188
Ag       5.3323750373     2.8240364209    -7.5512331190
Ag       2.1759755136     3.8444690680    -9.8593641559
Ag      -1.8817544351     2.1073913325    -0.4836595729
Ag       0.8989635719     2.6784284394    -0.4836553279
Ag       0.0000000000     0.0000000000     0.0000000000
Ag       2.7659350940     0.5759533343    -0.4836497308
Ag      -2.7690678527    -0.5606913044    -0.4836604335
Ag      -0.8841719671    -2.6833451220    -0.4836610364
Ag       1.8701101821    -2.1177433283    -0.4836551986
SUBEND
END

eor

rm TAPE21 logfile

######################################################
# FREQUENCY DEPENDENT POLARIZABILITY WITH LOCAL FIELDS
######################################################

$ADFBIN/adf <<eor
TITLE PIM Frequency-Dependent Polarizability w/ Local Fields, N2 on Ag68

ATOMS
N      -0.000006    -0.000060     2.302488
N       0.000019     0.000041     3.408628
END
```

```
BASIS
 type TZP
 core none
END

SYMMETRY NOSYM

BeckeGrid
 Quality good
End

XC
 GGA Becke Perdew
END

AORESPONSE
:: The FREQUENCY is at the Plasmon Max
 FREQUENCY 1 3.55 eV
 LIFETIME 0.0036749
 ALDA
END

DIMQM
 PIM
 JACOBI
 ALGORITHM DIRECT
:: Frequency-Dependent
 FREQUENCY
:: Include the local field effects
 LOCALFIELD
END

DIMPAR
Ag
 rad 1.4445
 drude 9.2 0.17 eV
 bound 5.0
SUBEND
XYZ
68
Ag    -1.2626801699   -1.1212553334   -7.5641602542
Ag    -1.5751210922    0.5238327185   -9.9892089940
Ag    -4.1814870442   -1.7663886735   -7.6050834452
Ag    -0.8822430837   -2.7600722440   -5.1959892962
Ag    -2.1240266195   -3.8206731623   -2.8525807753
Ag     3.4578336576   -2.6744062081   -2.8525742581
Ag    -3.7149452090   -3.2988294197   -5.2034646499
Ag     1.8985756093   -2.1890139391   -5.1959881717
Ag     0.6946811325   -3.3828052555   -2.8061234013
Ag     4.7143519584   -1.5678241605   -5.2034549930
Ag     1.6023915608   -0.5328957432   -7.5641613132
Ag     1.2412345786    1.1021928389   -9.9892023582
Ag     4.5392237030    0.0244593089   -7.6050682929
Ag     0.0000106997    0.0000013921   -5.1462708001
Ag    -0.3396832259    1.6541471619   -7.5641597129
Ag    -2.8450227342   -0.5497019461   -5.1959955618
Ag     2.8314207151    0.6159878635   -5.1959875699
Ag    -3.1884901403    1.0603809604   -7.5634515357
```

```
Ag    -1.2364094731    -1.0979217456    -2.7987978362
Ag     1.5690420034    -0.5218070047    -2.7987993856
Ag     4.3708278508     0.0708740258    -2.8525785565
Ag    -4.0450216512    -1.6573690296    -2.8525892410
Ag     2.5125751829     2.2311280445    -7.5634444475
Ag     0.6759414795    -3.2915128984    -7.5634430672
Ag     0.3339235029    -1.6260239766    -9.9892067983
Ag     1.0324285615    -5.0274542660    -5.1718100987
Ag    -2.2484187072    -3.9433040680    -7.6050830302
Ag     3.6204971713    -2.7380910382    -7.6050765656
Ag    -2.4569103660    -2.1817169842    -9.8785349524
Ag    -1.7752961456    -5.4191572521    -5.2192070180
Ag     3.7672926986    -4.2809622136    -5.2191965533
Ag     3.1179002480    -1.0369024811    -9.8785348598
Ag    -1.9491576017     2.1440839896    -5.1959915042
Ag     2.5822650861     2.2930127465    -2.8061239177
Ag    -0.3326165571     1.6197266638    -2.7987957984
Ag     0.9464626101     2.7387205045    -5.1959886567
Ag    -0.5598334280    -4.3819335958    -9.8593641366
Ag    -0.2204924851    -6.0299810786    -7.5512447020
Ag     2.5790329655    -5.4550834512    -7.5512427315
Ag     2.2414383913    -3.8066764388    -9.8593649949
Ag    -2.2907828197     3.9188443502    -7.6050826773
Ag     0.5610225957     4.5044771077    -7.6050819796
Ag    -4.8701069022     1.6196298364    -5.1718125544
Ag    -3.2769317991     1.0897966386    -2.8061340012
Ag    -0.6609450333     3.2186205924    -9.8785392784
Ag    -5.5910570566    -1.1220840757    -5.2192067211
Ag    -2.2467815447     3.7498049589    -2.8525803584
Ag    -0.9993904347     4.8666585643    -5.2034637574
Ag    -3.8054729020     4.2470446290    -5.2192106249
Ag    -3.5149221622     2.6758039407    -9.8593694452
Ag    -4.4173760657    -0.0377981351    -9.8593712497
Ag    -6.0137475245     0.4940463872    -7.5512522368
Ag    -5.1118586671     3.2059431066    -7.5512501894
Ag     0.5871922586     4.3317735603    -2.8525794711
Ag     1.8237836111     5.4030432621    -5.2192000459
Ag     3.8377073288     3.4078339914    -5.1718059272
Ag     5.5807980727     1.1721205307    -5.2191937875
Ag     3.4347425354     4.9610463596    -7.5512390021
Ag     4.0747942310     1.7061247432    -9.8593564188
Ag     5.3323750373     2.8240364209    -7.5512331190
Ag     2.1759755136     3.8444690680    -9.8593641559
Ag    -1.8817544351     2.1073913325    -0.4836595729
Ag     0.8989635719     2.6784284394    -0.4836553279
Ag     0.0000000000     0.0000000000     0.0000000000
Ag     2.7659350940     0.5759533343    -0.4836497308
Ag    -2.7690678527    -0.5606913044    -0.4836604335
Ag    -0.8841719671    -2.6833451220    -0.4836610364
Ag     1.8701101821    -2.1177433283    -0.4836551986
SUBEND
END

eor
```

Example: PIM: H2O on Ag2689

Download DIMQM_LOCAL.run

```sh
#! /bin/sh

# PIM example for H2O on a cluster with 2869 silver atoms.
#
# LocalDIM performs the calculation fully for a specified CutOffDist (in Bohr),
# but ignores the image dipoles from atoms outside of this distance.
# This option should converge to the full simulation for a larger CutOffDist.

$ADFBIN/adf <<eor
TITLE ['Local Dim Test']
basis
 Type TZP
 core None
end

atoms
O       0.016560    0.127604    0.000000
H       0.650009    0.566164    0.000000
H      -0.666568    0.484010    0.000000
end


DIMQM
 PIM
 LOCALDIM
 LOCALDIRECT
 debug
 PRINTLJPAR
 niter 2000
 CutOffDist 50.0
END

DIMPAR
 Ag
  rad=1.4445
 SUBEND
 XYZ
$ADFHOME/examples/adf/DIMQM_LOCAL/ag2869.xyz
 SUBEND

END

symmetry nosym
allpoints

FullSCF

Beckegrid
 Quality good
end
xc
 Model SAOP
end
```

```
end input

eor
```

Example: PIM: Polarizability with local fields

Download DIMQM_READLCLFLD.run

```
#!/bin/sh

######################################################
# FREQUENCY DEPENDENT POLARIZABILITY WITH LOCAL FIELDS
######################################################

$ADFBIN/adf <<eor
TITLE PIM Frequency-Dependent Polarizability w/ Local Fields, N2 on Ag68

ATOMS
N    -0.000006   -0.000060   2.302488
N     0.000019    0.000041   3.408628
END

BASIS
 type TZP
 core none
END

SYMMETRY NOSYM

BeckeGrid
 Quality good
End

XC
 GGA Becke Perdew
END

AORESPONSE
:: The FREQUENCY is at the Plasmon Max
 FREQUENCY 1 3.55 eV
 LIFETIME 0.0036749
 ALDA
END

DIMQM
 PIM
:: Frequency-Dependent
 FREQUENCY
:: Include the local field effects
 LocalField RESP
END

DIMPAR
Ag
 rad 1.4445
 drude 9.2 0.17 eV
 bound 5.0
```

```
SUBEND
XYZ
68
Ag      -1.2626801699      -1.1212553334      -7.5641602542
Ag      -1.5751210922       0.5238327185      -9.9892089940
Ag      -4.1814870442      -1.7663886735      -7.6050834452
Ag      -0.8822430837      -2.7600722440      -5.1959892962
Ag      -2.1240266195      -3.8206731623      -2.8525807753
Ag       3.4578336576      -2.6744062081      -2.8525742581
Ag      -3.7149452090      -3.2988294197      -5.2034646499
Ag       1.8985756093      -2.1890139391      -5.1959881717
Ag       0.6946811325      -3.3828052555      -2.8061234013
Ag       4.7143519584      -1.5678241605      -5.2034549930
Ag       1.6023915608      -0.5328957432      -7.5641613132
Ag       1.2412345786       1.1021928389      -9.9892023582
Ag       4.5392237030       0.0244593089      -7.6050682929
Ag       0.0000106997       0.0000013921      -5.1462708001
Ag      -0.3396832259       1.6541471619      -7.5641597129
Ag      -2.8450227342      -0.5497019461      -5.1959955618
Ag       2.8314207151       0.6159878635      -5.1959875699
Ag      -3.1884901403       1.0603809604      -7.5634515357
Ag      -1.2364094731      -1.0979217456      -2.7987978362
Ag       1.5690420034      -0.5218070047      -2.7987993856
Ag       4.3708278508       0.0708740258      -2.8525785565
Ag      -4.0450216512      -1.6573690296      -2.8525892410
Ag       2.5125751829       2.2311280445      -7.5634444475
Ag       0.6759414795      -3.2915128984      -7.5634430672
Ag       0.3339235029      -1.6260239766      -9.9892067983
Ag       1.0324285615      -5.0274542660      -5.1718100987
Ag      -2.2484187072      -3.9433040680      -7.6050830302
Ag       3.6204971713      -2.7380910382      -7.6050765656
Ag      -2.4569103660      -2.1817169842      -9.8785349524
Ag      -1.7752961456      -5.4191572521      -5.2192070180
Ag       3.7672926986      -4.2809622136      -5.2191965533
Ag       3.1179002480      -1.0369024811      -9.8785348598
Ag      -1.9491576017       2.1440839896      -5.1959915042
Ag       2.5822650861       2.2930127465      -2.8061239177
Ag      -0.3326165571       1.6197266638      -2.7987957984
Ag       0.9464626101       2.7387205045      -5.1959886567
Ag      -0.5598334280      -4.3819335958      -9.8593641366
Ag      -0.2204924851      -6.0299810786      -7.5512447020
Ag       2.5790329655      -5.4550834512      -7.5512427315
Ag       2.2414383913      -3.8066764388      -9.8593649949
Ag      -2.2907828197       3.9188443502      -7.6050826773
Ag       0.5610225957       4.5044771077      -7.6050819796
Ag      -4.8701069022       1.6196298364      -5.1718125544
Ag      -3.2769317991       1.0897966386      -2.8061340012
Ag      -0.6609450333       3.2186205924      -9.8785392784
Ag      -5.5910570566      -1.1220840757      -5.2192067211
Ag      -2.2467815447       3.7498049589      -2.8525803584
Ag      -0.9993904347       4.8666585643      -5.2034637574
Ag      -3.8054729020       4.2470446290      -5.2192106249
Ag      -3.5149221622       2.6758039407      -9.8593694452
Ag      -4.4173760657      -0.0377981351      -9.8593712497
Ag      -6.0137475245       0.4940463872      -7.5512522368
Ag      -5.1118586671       3.2059431066      -7.5512501894
Ag       0.5871922586       4.3317735603      -2.8525794711
Ag       1.8237836111       5.4030432621      -5.2192000459
```

```
Ag       3.8377073288    3.4078339914   -5.1718059272
Ag       5.5807980727    1.1721205307   -5.2191937875
Ag       3.4347425354    4.9610463596   -7.5512390021
Ag       4.0747942310    1.7061247432   -9.8593564188
Ag       5.3323750373    2.8240364209   -7.5512331190
Ag       2.1759755136    3.8444690680   -9.8593641559
Ag      -1.8817544351    2.1073913325   -0.4836595729
Ag       0.8989635719    2.6784284394   -0.4836553279
Ag       0.0000000000    0.0000000000    0.0000000000
Ag       2.7659350940    0.5759533343   -0.4836497308
Ag      -2.7690678527   -0.5606913044   -0.4836604335
Ag      -0.8841719671   -2.6833451220   -0.4836610364
Ag       1.8701101821   -2.1177433283   -0.4836551986
SUBEND
LCLFLD
 $ADFHOME/examples/adf/DIMQM_READLCLFLD/lclfld.txt
SUBEND
END

eor
```

Example: PIM: optimization N2 on silver cluster Ag68

Download DIMQM_Gradients.run

```
#! /bin/sh

# == Expert option ==

# This example demonstrates how to do a geometry optimization with the
# Polarizability Interaction Model (PIM).


$ADFBIN/adf <<eor
TITLE DIM/QM Gradient Test, N2 on Ag68

ATOMS
N      0.0000000000    0.0000000000    2.5000000000
N      0.0000000000    0.0000000000    3.5000000000
END

BASIS
 type TZP
 core none
END

SYMMETRY NOSYM

CHARGE 0

BeckeGrid
 Quality good
End

XC
 GGA Becke Perdew
END
```

```
:: It is suggested to use a full scf for the first
:: geometry iteration of a DIM/QM optimization
FULLSCF

GEOMETRY
 :: DIM/QM geomerty optimization must be done in Cartesian coordinates
 Optim Cartesian
 :: For a DIM/QM geomerty optimization
 :: it is suggested to use tighter convergence criterion
 Converge Grad=1E-4
 :: We are only interested in testing the gradients, so look at first iteration only
 Iterations 1
END

SAVE  TAPE21

DIMQM
:: Choose DIM/QM Methods:
:: Here PIM with screening
 PIM
 ALGORITHM DIRECT
 SCREEN EXP 1.0
:: Choose DIM/QM FORCE FIELD Method
:: First, we choose the Lennard-Jones 12-6 pot.
 FORCEFIELD LJ12-6
:: Next, we make the FORCE FIELD coordination dependent
 COORDDEPEND
:: Finally, we will add Chemisorption corrects
 CHEMISORPTION
:: User many add additonal FORCE FIELD parameters
:: Adding coordination dependent parameters (in a.u.)
::       ATOM     E_0        E_1        R_0        R_1        CN_max     R_max      R_min
 COORDPAR  Ag   0.003000  0.000800   3.250000   3.650000  12.000000  9.448600  5.669200
:: Adding Chemisorption parameters
::       ATOM     E_0        E_1        R_0        R_1        CUTOFF
 CHEMPAR   N    0.275000  0.052500   0.800000   0.900000  4.913000
:: To print the DIM/QM gradients
DEBUG
END

DIMPAR
Ag
 rad=1.4445
SUBEND
XYZ
68
Ag    -1.2626801699   -1.1212553334   -7.5641602542
Ag    -1.5751210922    0.5238327185   -9.9892089940
Ag    -4.1814870442   -1.7663886735   -7.6050834452
Ag    -0.8822430837   -2.7600722440   -5.1959892962
Ag    -2.1240266195   -3.8206731623   -2.8525807753
Ag     3.4578336576   -2.6744062081   -2.8525742581
Ag    -3.7149452090   -3.2988294197   -5.2034646499
Ag     1.8985756093   -2.1890139391   -5.1959881717
Ag     0.6946811325   -3.3828052555   -2.8061234013
Ag     4.7143519584   -1.5678241605   -5.2034549930
Ag     1.6023915608   -0.5328957432   -7.5641613132
```

```
Ag       1.2412345786      1.1021928389     -9.9892023582
Ag       4.5392237030      0.0244593089     -7.6050682929
Ag       0.0000106997      0.0000013921     -5.1462708001
Ag      -0.3396832259      1.6541471619     -7.5641597129
Ag      -2.8450227342     -0.5497019461     -5.1959955618
Ag       2.8314207151      0.6159878635     -5.1959875699
Ag      -3.1884901403      1.0603809604     -7.5634515357
Ag      -1.2364094731     -1.0979217456     -2.7987978362
Ag       1.5690420034     -0.5218070047     -2.7987993856
Ag       4.3708278508      0.0708740258     -2.8525785565
Ag      -4.0450216512     -1.6573690296     -2.8525892410
Ag       2.5125751829      2.2311280445     -7.5634444475
Ag       0.6759414795     -3.2915128984     -7.5634430672
Ag       0.3339235029     -1.6260239766     -9.9892067983
Ag       1.0324285615     -5.0274542660     -5.1718100987
Ag      -2.2484187072     -3.9433040680     -7.6050830302
Ag       3.6204971713     -2.7380910382     -7.6050765656
Ag      -2.4569103660     -2.1817169842     -9.8785349524
Ag      -1.7752961456     -5.4191572521     -5.2192070180
Ag       3.7672926986     -4.2809622136     -5.2191965533
Ag       3.1179002480     -1.0369024811     -9.8785348598
Ag      -1.9491576017      2.1440839896     -5.1959915042
Ag       2.5822650861      2.2930127465     -2.8061239177
Ag      -0.3326165571      1.6197266638     -2.7987957984
Ag       0.9464626101      2.7387205045     -5.1959886567
Ag      -0.5598334280     -4.3819335958     -9.8593641366
Ag      -0.2204924851     -6.0299810786     -7.5512447020
Ag       2.5790329655     -5.4550834512     -7.5512427315
Ag       2.2414383913     -3.8066764388     -9.8593649949
Ag      -2.2907828197      3.9188443502     -7.6050826773
Ag       0.5610225957      4.5044771077     -7.6050819796
Ag      -4.8701069022      1.6196298364     -5.1718125544
Ag      -3.2769317991      1.0897966386     -2.8061340012
Ag      -0.6609450333      3.2186205924     -9.8785392784
Ag      -5.5910570566     -1.1220840757     -5.2192067211
Ag      -2.2467815447      3.7498049589     -2.8525803584
Ag      -0.9993904347      4.8666585643     -5.2034637574
Ag      -3.8054729020      4.2470446290     -5.2192106249
Ag      -3.5149221622      2.6758039407     -9.8593694452
Ag      -4.4173760657     -0.0377981351     -9.8593712497
Ag      -6.0137475245      0.4940463872     -7.5512522368
Ag      -5.1118586671      3.2059431066     -7.5512501894
Ag       0.5871922586      4.3317735603     -2.8525794711
Ag       1.8237836111      5.4030432621     -5.2192000459
Ag       3.8377073288      3.4078339914     -5.1718059272
Ag       5.5807980727      1.1721205307     -5.2191937875
Ag       3.4347425354      4.9610463596     -7.5512390021
Ag       4.0747942310      1.7061247432     -9.8593564188
Ag       5.3323750373      2.8240364209     -7.5512331190
Ag       2.1759755136      3.8444690680     -9.8593641559
Ag      -1.8817544351      2.1073913325     -0.4836595729
Ag       0.8989635719      2.6784284394     -0.4836553279
Ag       0.0000000000      0.0000000000      0.0000000000
Ag       2.7659350940      0.5759533343     -0.4836497308
Ag      -2.7690678527     -0.5606913044     -0.4836604335
Ag      -0.8841719671     -2.6833451220     -0.4836610364
Ag       1.8701101821     -2.1177433283     -0.4836551986
SUBEND
```

```
END

eor
```

Example: PIM: polarizability N2 on silver cluster Ag68

Download DIMQM_PIM_Pol.run

```
#! /bin/sh

# == Expert option ==

# This example demonstrates how to do calculate static and frequency dependent
# polarizabilities with the Polarizability Interaction Model (PIM). The example
# consists of 3 calculations: static polarizability, frequency dependent, and
# with inclusion of effect of the local field.

# When the molecule interacts with a metal nanoparticle, there are two types of
# interactions: the image field and the local field. The image field is caused
# by the dipoles induced into the nanoparticle by the molecule's electron
# density. This is always taken into account in a DIM/QM calculation. The local
# field arises by direct interactions of the nanoparticle with an external
# field. Addition of the LOCALFIELD key causes the DIM/QM calculation to include
# this effect, but by default this is not included in a DIM/QM calculation.


#######################
# STATIC POLARIZABILITY
#######################

$ADFBIN/adf <<eor
TITLE PIM Static Polarizability, N2 on Ag68

ATOMS
N     -0.000006    -0.000060    2.302488
N      0.000019     0.000041    3.408628
END

BASIS
 type TZP
 core none
END

SYMMETRY NOSYM

BeckeGrid
 Quality good
End

XC
 GGA Becke Perdew
END

AORESPONSE
 ALDA
END
```

```
DIMQM
 PIM
 ALGORITHM BRUTE
END

DIMPAR
Ag
 rad 1.4445
 drude 9.2 0.17 eV
 bound 5.0
SUBEND
XYZ
68
Ag    -1.2626801699  -1.1212553334  -7.5641602542
Ag    -1.5751210922   0.5238327185  -9.9892089940
Ag    -4.1814870442  -1.7663886735  -7.6050834452
Ag    -0.8822430837  -2.7600722440  -5.1959892962
Ag    -2.1240266195  -3.8206731623  -2.8525807753
Ag     3.4578336576  -2.6744062081  -2.8525742581
Ag    -3.7149452090  -3.2988294197  -5.2034646499
Ag     1.8985756093  -2.1890139391  -5.1959881717
Ag     0.6946811325  -3.3828052555  -2.8061234013
Ag     4.7143519584  -1.5678241605  -5.2034549930
Ag     1.6023915608  -0.5328957432  -7.5641613132
Ag     1.2412345786   1.1021928389  -9.9892023582
Ag     4.5392237030   0.0244593089  -7.6050682929
Ag     0.0000106997   0.0000013921  -5.1462708001
Ag    -0.3396832259   1.6541471619  -7.5641597129
Ag    -2.8450227342  -0.5497019461  -5.1959955618
Ag     2.8314207151   0.6159878635  -5.1959875699
Ag    -3.1884901403   1.0603809604  -7.5634515357
Ag    -1.2364094731  -1.0979217456  -2.7987978362
Ag     1.5690420034  -0.5218070047  -2.7987993856
Ag     4.3708278508   0.0708740258  -2.8525785565
Ag    -4.0450216512  -1.6573690296  -2.8525892410
Ag     2.5125751829   2.2311280445  -7.5634444475
Ag     0.6759414795  -3.2915128984  -7.5634430672
Ag     0.3339235029  -1.6260239766  -9.9892067983
Ag     1.0324285615  -5.0274542660  -5.1718100987
Ag    -2.2484187072  -3.9433040680  -7.6050830302
Ag     3.6204971713  -2.7380910382  -7.6050765656
Ag    -2.4569103660  -2.1817169842  -9.8785349524
Ag    -1.7752961456  -5.4191572521  -5.2192070180
Ag     3.7672926986  -4.2809622136  -5.2191965533
Ag     3.1179002480  -1.0369024811  -9.8785348598
Ag    -1.9491576017   2.1440839896  -5.1959915042
Ag     2.5822650861   2.2930127465  -2.8061239177
Ag    -0.3326165571   1.6197266638  -2.7987957984
Ag     0.9464626101   2.7387205045  -5.1959886567
Ag    -0.5598334280  -4.3819335958  -9.8593641366
Ag    -0.2204924851  -6.0299810786  -7.5512447020
Ag     2.5790329655  -5.4550834512  -7.5512427315
Ag     2.2414383913  -3.8066764388  -9.8593649949
Ag    -2.2907828197   3.9188443502  -7.6050826773
Ag     0.5610225957   4.5044771077  -7.6050819796
Ag    -4.8701069022   1.6196298364  -5.1718125544
Ag    -3.2769317991   1.0897966386  -2.8061340012
Ag    -0.6609450333   3.2186205924  -9.8785392784
```

```
Ag      -5.5910570566   -1.1220840757   -5.2192067211
Ag      -2.2467815447    3.7498049589   -2.8525803584
Ag      -0.9993904347    4.8666585643   -5.2034637574
Ag      -3.8054729020    4.2470446290   -5.2192106249
Ag      -3.5149221622    2.6758039407   -9.8593694452
Ag      -4.4173760657   -0.0377981351   -9.8593712497
Ag      -6.0137475245    0.4940463872   -7.5512522368
Ag      -5.1118586671    3.2059431066   -7.5512501894
Ag       0.5871922586    4.3317735603   -2.8525794711
Ag       1.8237836111    5.4030432621   -5.2192000459
Ag       3.8377073288    3.4078339914   -5.1718059272
Ag       5.5807980727    1.1721205307   -5.2191937875
Ag       3.4347425354    4.9610463596   -7.5512390021
Ag       4.0747942310    1.7061247432   -9.8593564188
Ag       5.3323750373    2.8240364209   -7.5512331190
Ag       2.1759755136    3.8444690680   -9.8593641559
Ag      -1.8817544351    2.1073913325   -0.4836595729
Ag       0.8989635719    2.6784284394   -0.4836553279
Ag       0.0000000000    0.0000000000    0.0000000000
Ag       2.7659350940    0.5759533343   -0.4836497308
Ag      -2.7690678527   -0.5606913044   -0.4836604335
Ag      -0.8841719671   -2.6833451220   -0.4836610364
Ag       1.8701101821   -2.1177433283   -0.4836551986
SUBEND
END


eor

rm TAPE21 logfile

####################################
# FREQUENCY DEPENDENT POLARIZABILITY
####################################

$ADFBIN/adf <<eor
TITLE PIM Frequency-Dependent Polarizability, N2 on Ag68

ATOMS
N      -0.000006    -0.000060     2.302488
N       0.000019     0.000041     3.408628
END

BASIS
 type TZP
 core none
END

SYMMETRY NOSYM

BeckeGrid
 Quality good
End

XC
 GGA Becke Perdew
END

AORESPONSE
```

```
:: The FREQUENCY is at the Plasmon Max
 FREQUENCY 1 3.55 eV
 LIFETIME 0.0036749
 ALDA
END

DIMQM
 PIM
 ALGORITHM BRUTE
:: Frequency-Dependent
 FREQUENCY
END

DIMPAR
Ag
 rad 1.4445
 drude 9.2 0.17 eV
 bound 5.0
SUBEND
XYZ
68
Ag    -1.2626801699   -1.1212553334   -7.5641602542
Ag    -1.5751210922    0.5238327185   -9.9892089940
Ag    -4.1814870442   -1.7663886735   -7.6050834452
Ag    -0.8822430837   -2.7600722440   -5.1959892962
Ag    -2.1240266195   -3.8206731623   -2.8525807753
Ag     3.4578336576   -2.6744062081   -2.8525742581
Ag    -3.7149452090   -3.2988294197   -5.2034646499
Ag     1.8985756093   -2.1890139391   -5.1959881717
Ag     0.6946811325   -3.3828052555   -2.8061234013
Ag     4.7143519584   -1.5678241605   -5.2034549930
Ag     1.6023915608   -0.5328957432   -7.5641613132
Ag     1.2412345786    1.1021928389   -9.9892023582
Ag     4.5392237030    0.0244593089   -7.6050682929
Ag     0.0000106997    0.0000013921   -5.1462708001
Ag    -0.3396832259    1.6541471619   -7.5641597129
Ag    -2.8450227342   -0.5497019461   -5.1959955618
Ag     2.8314207151    0.6159878635   -5.1959875699
Ag    -3.1884901403    1.0603809604   -7.5634515357
Ag    -1.2364094731   -1.0979217456   -2.7987978362
Ag     1.5690420034   -0.5218070047   -2.7987993856
Ag     4.3708278508    0.0708740258   -2.8525785565
Ag    -4.0450216512   -1.6573690296   -2.8525892410
Ag     2.5125751829    2.2311280445   -7.5634444475
Ag     0.6759414795   -3.2915128984   -7.5634430672
Ag     0.3339235029   -1.6260239766   -9.9892067983
Ag     1.0324285615   -5.0274542660   -5.1718100987
Ag    -2.2484187072   -3.9433040680   -7.6050830302
Ag     3.6204971713   -2.7380910382   -7.6050765656
Ag    -2.4569103660   -2.1817169842   -9.8785349524
Ag    -1.7752961456   -5.4191572521   -5.2192070180
Ag     3.7672926986   -4.2809622136   -5.2191965533
Ag     3.1179002480   -1.0369024811   -9.8785348598
Ag    -1.9491576017    2.1440839896   -5.1959915042
Ag     2.5822650861    2.2930127465   -2.8061239177
Ag    -0.3326165571    1.6197266638   -2.7987957984
Ag     0.9464626101    2.7387205045   -5.1959886567
Ag    -0.5598334280   -4.3819335958   -9.8593641366
```

```
Ag      -0.2204924851    -6.0299810786    -7.5512447020
Ag       2.5790329655    -5.4550834512    -7.5512427315
Ag       2.2414383913    -3.8066764388    -9.8593649949
Ag      -2.2907828197     3.9188443502    -7.6050826773
Ag       0.5610225957     4.5044771077    -7.6050819796
Ag      -4.8701069022     1.6196298364    -5.1718125544
Ag      -3.2769317991     1.0897966386    -2.8061340012
Ag      -0.6609450333     3.2186205924    -9.8785392784
Ag      -5.5910570566    -1.1220840757    -5.2192067211
Ag      -2.2467815447     3.7498049589    -2.8525803584
Ag      -0.9993904347     4.8666585643    -5.2034637574
Ag      -3.8054729020     4.2470446290    -5.2192106249
Ag      -3.5149221622     2.6758039407    -9.8593694452
Ag      -4.4173760657    -0.0377981351    -9.8593712497
Ag      -6.0137475245     0.4940463872    -7.5512522368
Ag      -5.1118586671     3.2059431066    -7.5512501894
Ag       0.5871922586     4.3317735603    -2.8525794711
Ag       1.8237836111     5.4030432621    -5.2192000459
Ag       3.8377073288     3.4078339914    -5.1718059272
Ag       5.5807980727     1.1721205307    -5.2191937875
Ag       3.4347425354     4.9610463596    -7.5512390021
Ag       4.0747942310     1.7061247432    -9.8593564188
Ag       5.3323750373     2.8240364209    -7.5512331190
Ag       2.1759755136     3.8444690680    -9.8593641559
Ag      -1.8817544351     2.1073913325    -0.4836595729
Ag       0.8989635719     2.6784284394    -0.4836553279
Ag       0.0000000000     0.0000000000     0.0000000000
Ag       2.7659350940     0.5759533343    -0.4836497308
Ag      -2.7690678527    -0.5606913044    -0.4836604335
Ag      -0.8841719671    -2.6833451220    -0.4836610364
Ag       1.8701101821    -2.1177433283    -0.4836551986
SUBEND
END

eor

rm TAPE21 logfile

######################################################
# FREQUENCY DEPENDENT POLARIZABILITY WITH LOCAL FIELDS
######################################################

$ADFBIN/adf <<eor
TITLE PIM Frequency-Dependent Polarizability w/ Local Fields, N2 on Ag68

ATOMS
N      -0.000006    -0.000060     2.302488
N       0.000019     0.000041     3.408628
END

BASIS
 type TZP
 core none
END

SYMMETRY NOSYM

BeckeGrid
```

```
  Quality good
End

XC
 GGA Becke Perdew
END

AORESPONSE
:: The FREQUENCY is at the Plasmon Max
 FREQUENCY 1 3.55 eV
 LIFETIME 0.0036749
 ALDA
END

DIMQM
 PIM
 ALGORITHM DIRECT
:: Frequency-Dependent
 FREQUENCY
:: Include the local field effects
 LOCALFIELD
END

DIMPAR
Ag
 rad 1.4445
 drude 9.2 0.17 eV
 bound 5.0
SUBEND
XYZ
68
Ag    -1.2626801699   -1.1212553334   -7.5641602542
Ag    -1.5751210922    0.5238327185   -9.9892089940
Ag    -4.1814870442   -1.7663886735   -7.6050834452
Ag    -0.8822430837   -2.7600722440   -5.1959892962
Ag    -2.1240266195   -3.8206731623   -2.8525807753
Ag     3.4578336576   -2.6744062081   -2.8525742581
Ag    -3.7149452090   -3.2988294197   -5.2034646499
Ag     1.8985756093   -2.1890139391   -5.1959881717
Ag     0.6946811325   -3.3828052555   -2.8061234013
Ag     4.7143519584   -1.5678241605   -5.2034549930
Ag     1.6023915608   -0.5328957432   -7.5641613132
Ag     1.2412345786    1.1021928389   -9.9892023582
Ag     4.5392237030    0.0244593089   -7.6050682929
Ag     0.0000106997    0.0000013921   -5.1462708001
Ag    -0.3396832259    1.6541471619   -7.5641597129
Ag    -2.8450227342   -0.5497019461   -5.1959955618
Ag     2.8314207151    0.6159878635   -5.1959875699
Ag    -3.1884901403    1.0603809604   -7.5634515357
Ag    -1.2364094731   -1.0979217456   -2.7987978362
Ag     1.5690420034   -0.5218070047   -2.7987993856
Ag     4.3708278508    0.0708740258   -2.8525785565
Ag    -4.0450216512   -1.6573690296   -2.8525892410
Ag     2.5125751829    2.2311280445   -7.5634444475
Ag     0.6759414795   -3.2915128984   -7.5634430672
Ag     0.3339235029   -1.6260239766   -9.9892067983
Ag     1.0324285615   -5.0274542660   -5.1718100987
Ag    -2.2484187072   -3.9433040680   -7.6050830302
```

```
Ag      3.6204971713   -2.7380910382   -7.6050765656
Ag     -2.4569103660   -2.1817169842   -9.8785349524
Ag     -1.7752961456   -5.4191572521   -5.2192070180
Ag      3.7672926986   -4.2809622136   -5.2191965533
Ag      3.1179002480   -1.0369024811   -9.8785348598
Ag     -1.9491576017    2.1440839896   -5.1959915042
Ag      2.5822650861    2.2930127465   -2.8061239177
Ag     -0.3326165571    1.6197266638   -2.7987957984
Ag      0.9464626101    2.7387205045   -5.1959886567
Ag     -0.5598334280   -4.3819335958   -9.8593641366
Ag     -0.2204924851   -6.0299810786   -7.5512447020
Ag      2.5790329655   -5.4550834512   -7.5512427315
Ag      2.2414383913   -3.8066764388   -9.8593649949
Ag     -2.2907828197    3.9188443502   -7.6050826773
Ag      0.5610225957    4.5044771077   -7.6050819796
Ag     -4.8701069022    1.6196298364   -5.1718125544
Ag     -3.2769317991    1.0897966386   -2.8061340012
Ag     -0.6609450333    3.2186205924   -9.8785392784
Ag     -5.5910570566   -1.1220840757   -5.2192067211
Ag     -2.2467815447    3.7498049589   -2.8525803584
Ag     -0.9993904347    4.8666585643   -5.2034637574
Ag     -3.8054729020    4.2470446290   -5.2192106249
Ag     -3.5149221622    2.6758039407   -9.8593694452
Ag     -4.4173760657   -0.0377981351   -9.8593712497
Ag     -6.0137475245    0.4940463872   -7.5512522368
Ag     -5.1118586671    3.2059431066   -7.5512501894
Ag      0.5871922586    4.3317735603   -2.8525794711
Ag      1.8237836111    5.4030432621   -5.2192000459
Ag      3.8377073288    3.4078339914   -5.1718059272
Ag      5.5807980727    1.1721205307   -5.2191937875
Ag      3.4347425354    4.9610463596   -7.5512390021
Ag      4.0747942310    1.7061247432   -9.8593564188
Ag      5.3323750373    2.8240364209   -7.5512331190
Ag      2.1759755136    3.8444690680   -9.8593641559
Ag     -1.8817544351    2.1073913325   -0.4836595729
Ag      0.8989635719    2.6784284394   -0.4836553279
Ag      0.0000000000    0.0000000000    0.0000000000
Ag      2.7659350940    0.5759533343   -0.4836497308
Ag     -2.7690678527   -0.5606913044   -0.4836604335
Ag     -0.8841719671   -2.6833451220   -0.4836610364
Ag      1.8701101821   -2.1177433283   -0.4836551986
SUBEND
END

eor
```

Example: PIM: Raman scattering N2 on silver cluster Ag68

Download DIMQM_Raman.run

```
#! /bin/sh

# == Expert option ==

# This example demonstrates how to calculate Raman scattering with the
# Polarizability Interaction Model (PIM). AORESPONSE is used and numerical
# frequencies.
```

```
$ADFBIN/adf <<eor
TITLE DIM/QM Raman Calculation, N2 on Ag68

ATOMS
N     -0.000006    -0.000060    2.302488
N      0.000019     0.000041    3.408628
END

BASIS
 type TZP
 core none
END

SYMMETRY NOSYM

CHARGE 0

BeckeGrid
 Quality good
End

XC
 GGA Becke Perdew
END

AORESPONSE
 RAMAN
:: The FREQUENCY is at the Plasmon Max
 FREQUENCY 1 3.55 EV
 LIFETIME 0.0036749
 ALDA
END
ALLPOINTS

GEOMETRY
 Frequencies
 End
END

SAVE  TAPE21

DIMQM
 PIM
 ALGORITHM DIRECT
 SCREEN EXP 1.0
 FORCEFIELD LJ12-6
 COORDDEPEND
 CHEMISORPTION
::       ATOM   E_0        E_1        R_0        R_1        CN_max     R_max      R_min
 COORDPAR  Ag  0.003000  0.000800  3.250000  3.650000  12.000000  9.448600  5.669200
::       ATOM   E_0        E_1        R_0        R_1        CUTOFF
 CHEMPAR   N   0.275000  0.052500  0.800000  0.900000  4.913000
 FREQUENCY
 LOCALFIELD
END
```

```
DIMPAR
Ag
 rad=1.4445
 drude 9.2 0.17 eV
 bound 5.0
SUBEND
XYZ
68
Ag    -1.2626801699   -1.1212553334   -7.5641602542
Ag    -1.5751210922    0.5238327185   -9.9892089940
Ag    -4.1814870442   -1.7663886735   -7.6050834452
Ag    -0.8822430837   -2.7600722440   -5.1959892962
Ag    -2.1240266195   -3.8206731623   -2.8525807753
Ag     3.4578336576   -2.6744062081   -2.8525742581
Ag    -3.7149452090   -3.2988294197   -5.2034646499
Ag     1.8985756093   -2.1890139391   -5.1959881717
Ag     0.6946811325   -3.3828052555   -2.8061234013
Ag     4.7143519584   -1.5678241605   -5.2034549930
Ag     1.6023915608   -0.5328957432   -7.5641613132
Ag     1.2412345786    1.1021928389   -9.9892023582
Ag     4.5392237030    0.0244593089   -7.6050682929
Ag     0.0000106997    0.0000013921   -5.1462708001
Ag    -0.3396832259    1.6541471619   -7.5641597129
Ag    -2.8450227342   -0.5497019461   -5.1959955618
Ag     2.8314207151    0.6159878635   -5.1959875699
Ag    -3.1884901403    1.0603809604   -7.5634515357
Ag    -1.2364094731   -1.0979217456   -2.7987978362
Ag     1.5690420034   -0.5218070047   -2.7987993856
Ag     4.3708278508    0.0708740258   -2.8525785565
Ag    -4.0450216512   -1.6573690296   -2.8525892410
Ag     2.5125751829    2.2311280445   -7.5634444475
Ag     0.6759414795   -3.2915128984   -7.5634430672
Ag     0.3339235029   -1.6260239766   -9.9892067983
Ag     1.0324285615   -5.0274542660   -5.1718100987
Ag    -2.2484187072   -3.9433040680   -7.6050830302
Ag     3.6204971713   -2.7380910382   -7.6050765656
Ag    -2.4569103660   -2.1817169842   -9.8785349524
Ag    -1.7752961456   -5.4191572521   -5.2192070180
Ag     3.7672926986   -4.2809622136   -5.2191965533
Ag     3.1179002480   -1.0369024811   -9.8785348598
Ag    -1.9491576017    2.1440839896   -5.1959915042
Ag     2.5822650861    2.2930127465   -2.8061239177
Ag    -0.3326165571    1.6197266638   -2.7987957984
Ag     0.9464626101    2.7387205045   -5.1959886567
Ag    -0.5598334280   -4.3819335958   -9.8593641366
Ag    -0.2204924851   -6.0299810786   -7.5512447020
Ag     2.5790329655   -5.4550834512   -7.5512427315
Ag     2.2414383913   -3.8066764388   -9.8593649949
Ag    -2.2907828197    3.9188443502   -7.6050826773
Ag     0.5610225957    4.5044771077   -7.6050819796
Ag    -4.8701069022    1.6196298364   -5.1718125544
Ag    -3.2769317991    1.0897966386   -2.8061340012
Ag    -0.6609450333    3.2186205924   -9.8785392784
Ag    -5.5910570566   -1.1220840757   -5.2192067211
Ag    -2.2467815447    3.7498049589   -2.8525803584
Ag    -0.9993904347    4.8666585643   -5.2034637574
Ag    -3.8054729020    4.2470446290   -5.2192106249
```

```
Ag      -3.5149221622     2.6758039407    -9.8593694452
Ag      -4.4173760657    -0.0377981351    -9.8593712497
Ag      -6.0137475245     0.4940463872    -7.5512522368
Ag      -5.1118586671     3.2059431066    -7.5512501894
Ag       0.5871922586     4.3317735603    -2.8525794711
Ag       1.8237836111     5.4030432621    -5.2192000459
Ag       3.8377073288     3.4078339914    -5.1718059272
Ag       5.5807980727     1.1721205307    -5.2191937875
Ag       3.4347425354     4.9610463596    -7.5512390021
Ag       4.0747942310     1.7061247432    -9.8593564188
Ag       5.3323750373     2.8240364209    -7.5512331190
Ag       2.1759755136     3.8444690680    -9.8593641559
Ag      -1.8817544351     2.1073913325    -0.4836595729
Ag       0.8989635719     2.6784284394    -0.4836553279
Ag       0.0000000000     0.0000000000     0.0000000000
Ag       2.7659350940     0.5759533343    -0.4836497308
Ag      -2.7690678527    -0.5606913044    -0.4836604335
Ag      -0.8841719671    -2.6833451220    -0.4836610364
Ag       1.8701101821    -2.1177433283    -0.4836551986
SUBEND
END

eor
```

Example: PIM: SEROA calculation N2 on silver cluster Ag68

Download DIMQM_SEROA.run

```
#! /bin/sh

# == Expert option ==


# This example demonstrates how to calculate Surface-Enhanced Raman Optical
# Activity (SEROA) with the Polarizability Interaction Model (PIM). AORESPONSE
# is used and numerical frequencies.


$ADFBIN/adf <<eor
TITLE DIM/QM SEROA Calculation, N2 on Ag68

ATOMS
N       -0.000097   -0.000310     2.997901
N        0.000076    0.000133     4.095604
END

BASIS
 type TZP
END

SYMMETRY NoSym

Integration
   Accint 6.0
End

AORESPONSE
```

```
 scf   converge 1d-6 iterations 100
 frequency 1 5.0 eV
 lifetime 0.0037
 El_Dipole_El_Dipole Velocity
 El_Dipole_El_Quadrupole Velocity
 El_Dipole_Mag_Dipole Velocity
 VROA
 Traceless
END
AllPoints


GEOMETRY
 Frequencies
 End
END


ExactDensity

DIMQM
 PIM
 ALGORITHM DIRECT
 SCREEN GAU 1.0
 FORCEFIELD LJ12-6
 FREQUENCY
 LOCALFIELD
END


DIMPAR
Ag
 rad=1.4445
 drude 5.0 0.1 eV
 bound 5.0
SUBEND
XYZ
68
Ag    -1.2626801699   -1.1212553334   -7.5641602542
Ag    -1.5751210922    0.5238327185   -9.9892089940
Ag    -4.1814870442   -1.7663886735   -7.6050834452
Ag    -0.8822430837   -2.7600722440   -5.1959892962
Ag    -2.1240266195   -3.8206731623   -2.8525807753
Ag     3.4578336576   -2.6744062081   -2.8525742581
Ag    -3.7149452090   -3.2988294197   -5.2034646499
Ag     1.8985756093   -2.1890139391   -5.1959881717
Ag     0.6946811325   -3.3828052555   -2.8061234013
Ag     4.7143519584   -1.5678241605   -5.2034549930
Ag     1.6023915608   -0.5328957432   -7.5641613132
Ag     1.2412345786    1.1021928389   -9.9892023582
Ag     4.5392237030    0.0244593089   -7.6050682929
Ag     0.0000106997    0.0000013921   -5.1462708001
Ag    -0.3396832259    1.6541471619   -7.5641597129
Ag    -2.8450227342   -0.5497019461   -5.1959955618
Ag     2.8314207151    0.6159878635   -5.1959875699
Ag    -3.1884901403    1.0603809604   -7.5634515357
Ag    -1.2364094731   -1.0979217456   -2.7987978362
Ag     1.5690420034   -0.5218070047   -2.7987993856
Ag     4.3708278508    0.0708740258   -2.8525785565
Ag    -4.0450216512   -1.6573690296   -2.8525892410
Ag     2.5125751829    2.2311280445   -7.5634444475
```

```
Ag      0.6759414795   -3.2915128984   -7.5634430672
Ag      0.3339235029   -1.6260239766   -9.9892067983
Ag      1.0324285615   -5.0274542660   -5.1718100987
Ag     -2.2484187072   -3.9433040680   -7.6050830302
Ag      3.6204971713   -2.7380910382   -7.6050765656
Ag     -2.4569103660   -2.1817169842   -9.8785349524
Ag     -1.7752961456   -5.4191572521   -5.2192070180
Ag      3.7672926986   -4.2809622136   -5.2191965533
Ag      3.1179002480   -1.0369024811   -9.8785348598
Ag     -1.9491576017    2.1440839896   -5.1959915042
Ag      2.5822650861    2.2930127465   -2.8061239177
Ag     -0.3326165571    1.6197266638   -2.7987957984
Ag      0.9464626101    2.7387205045   -5.1959886567
Ag     -0.5598334280   -4.3819335958   -9.8593641366
Ag     -0.2204924851   -6.0299810786   -7.5512447020
Ag      2.5790329655   -5.4550834512   -7.5512427315
Ag      2.2414383913   -3.8066764388   -9.8593649949
Ag     -2.2907828197    3.9188443502   -7.6050826773
Ag      0.5610225957    4.5044771077   -7.6050819796
Ag     -4.8701069022    1.6196298364   -5.1718125544
Ag     -3.2769317991    1.0897966386   -2.8061340012
Ag     -0.6609450333    3.2186205924   -9.8785392784
Ag     -5.5910570566   -1.1220840757   -5.2192067211
Ag     -2.2467815447    3.7498049589   -2.8525803584
Ag     -0.9993904347    4.8666585643   -5.2034637574
Ag     -3.8054729020    4.2470446290   -5.2192106249
Ag     -3.5149221622    2.6758039407   -9.8593694452
Ag     -4.4173760657   -0.0377981351   -9.8593712497
Ag     -6.0137475245    0.4940463872   -7.5512522368
Ag     -5.1118586671    3.2059431066   -7.5512501894
Ag      0.5871922586    4.3317735603   -2.8525794711
Ag      1.8237836111    5.4030432621   -5.2192000459
Ag      3.8377073288    3.4078339914   -5.1718059272
Ag      5.5807980727    1.1721205307   -5.2191937875
Ag      3.4347425354    4.9610463596   -7.5512390021
Ag      4.0747942310    1.7061247432   -9.8593564188
Ag      5.3323750373    2.8240364209   -7.5512331190
Ag      2.1759755136    3.8444690680   -9.8593641559
Ag     -1.8817544351    2.1073913325   -0.4836595729
Ag      0.8989635719    2.6784284394   -0.4836553279
Ag      0.0000000000    0.0000000000    0.0000000000
Ag      2.7659350940    0.5759533343   -0.4836497308
Ag     -2.7690678527   -0.5606913044   -0.4836604335
Ag     -0.8841719671   -2.6833451220   -0.4836610364
Ag      1.8701101821   -2.1177433283   -0.4836551986
SUBEND
END

eor
```

Example: PIM: Multipole Method N2 on silver cluster Ag1415

Download DIMQM_PIM_CMM.run

```
#! /bin/sh

# == Expert option ==
```

```
# This example demonstrates how to do a single point calculation with the
# Polarizability Interaction Model (PIM), for a large metal cluster.


$ADFBIN/adf <<eor
TITLE PIM Cell Multipole Method Test, N2 on Ag1415

ATOMS
N     -0.000006    -0.000060     2.302488
N      0.000019     0.000041     3.408628
END

BASIS
 type TZP
 core none
END

SYMMETRY NOSYM

BeckeGrid
 Quality good
End

XC
 GGA Becke Perdew
END

DIMQM
 PIM
 ALGORITHM MULTI
:: This is used to efficiently calculate the volume of the nanoparticle
 MULTIPLIER 0.683 0.517 1.1301
END

DIMPAR
Ag
 rad 1.4445
SUBEND
XYZ
1415
Ag   0.00000000    0.00000000  -17.21195420
Ag   2.88900000    0.00000000  -17.21195420
Ag   5.77800000    0.00000000  -17.21195420
Ag   8.66700000    0.00000000  -17.21195420
Ag  11.55600000    0.00000000  -17.21195420
Ag  14.44500000    0.00000000  -17.21195420
Ag  17.33400000    0.00000000  -17.21195420
Ag  20.22300000    0.00000000  -17.21195420
Ag   1.29138300    2.58276600  -17.21195420
Ag   4.18038300    2.58276600  -17.21195420
Ag   7.06938300    2.58276600  -17.21195420
Ag   9.95838300    2.58276600  -17.21195420
Ag  12.84738300    2.58276600  -17.21195420
Ag  15.73638300    2.58276600  -17.21195420
Ag  18.62538300    2.58276600  -17.21195420
Ag   2.58276600    5.16553200  -17.21195420
Ag   5.47176600    5.16553200  -17.21195420
```

```
Ag     8.36076600     5.16553200    -17.21195420
Ag    11.24976600     5.16553200    -17.21195420
Ag    14.13876600     5.16553200    -17.21195420
Ag    17.02776600     5.16553200    -17.21195420
Ag     3.87414900     7.74829800    -17.21195420
Ag     6.76314900     7.74829800    -17.21195420
Ag     9.65214900     7.74829800    -17.21195420
Ag    12.54114900     7.74829800    -17.21195420
Ag    15.43014900     7.74829800    -17.21195420
Ag     5.16553200    10.33106400    -17.21195420
Ag     8.05453200    10.33106400    -17.21195420
Ag    10.94353200    10.33106400    -17.21195420
Ag    13.83253200    10.33106400    -17.21195420
Ag     6.45691500    12.91383000    -17.21195420
Ag     9.34591500    12.91383000    -17.21195420
Ag    12.23491500    12.91383000    -17.21195420
Ag     7.74829800    15.49659600    -17.21195420
Ag    10.63729800    15.49659600    -17.21195420
Ag     9.03968100    18.07936200    -17.21195420
Ag     1.29716100     0.80314200    -14.75919320
Ag     4.18616100     0.80314200    -14.75919320
Ag     7.07516100     0.80314200    -14.75919320
Ag     9.96416100     0.80314200    -14.75919320
Ag    12.85316100     0.80314200    -14.75919320
Ag    15.74216100     0.80314200    -14.75919320
Ag    18.63116100     0.80314200    -14.75919320
Ag     2.58854400     3.38590800    -14.75919320
Ag     5.47754400     3.38590800    -14.75919320
Ag     8.36654400     3.38590800    -14.75919320
Ag    11.25554400     3.38590800    -14.75919320
Ag    14.14454400     3.38590800    -14.75919320
Ag    17.03354400     3.38590800    -14.75919320
Ag     3.87992700     5.96867400    -14.75919320
Ag     6.76892700     5.96867400    -14.75919320
Ag     9.65792700     5.96867400    -14.75919320
Ag    12.54692700     5.96867400    -14.75919320
Ag    15.43592700     5.96867400    -14.75919320
Ag     5.17131000     8.55144000    -14.75919320
Ag     8.06031000     8.55144000    -14.75919320
Ag    10.94931000     8.55144000    -14.75919320
Ag    13.83831000     8.55144000    -14.75919320
Ag     6.46269300    11.13420600    -14.75919320
Ag     9.35169300    11.13420600    -14.75919320
Ag    12.24069300    11.13420600    -14.75919320
Ag     7.75407600    13.71697200    -14.75919320
Ag    10.64307600    13.71697200    -14.75919320
Ag     9.04545900    16.29973800    -14.75919320
Ag     2.59432200     1.60628400    -12.30643220
Ag     5.48332200     1.60628400    -12.30643220
Ag     8.37232200     1.60628400    -12.30643220
Ag    11.26132200     1.60628400    -12.30643220
Ag    14.15032200     1.60628400    -12.30643220
Ag    17.03932200     1.60628400    -12.30643220
Ag     3.88570500     4.18905000    -12.30643220
Ag     6.77470500     4.18905000    -12.30643220
Ag     9.66370500     4.18905000    -12.30643220
Ag    12.55270500     4.18905000    -12.30643220
Ag    15.44170500     4.18905000    -12.30643220
```

```
Ag     5.17708800      6.77181600    -12.30643220
Ag     8.06608800      6.77181600    -12.30643220
Ag    10.95508800      6.77181600    -12.30643220
Ag    13.84408800      6.77181600    -12.30643220
Ag     6.46847100      9.35458200    -12.30643220
Ag     9.35747100      9.35458200    -12.30643220
Ag    12.24647100      9.35458200    -12.30643220
Ag     7.75985400     11.93734800    -12.30643220
Ag    10.64885400     11.93734800    -12.30643220
Ag     9.05123700     14.52011400    -12.30643220
Ag     3.89148300      2.40942600     -9.85367120
Ag     6.78048300      2.40942600     -9.85367120
Ag     9.66948300      2.40942600     -9.85367120
Ag    12.55848300      2.40942600     -9.85367120
Ag    15.44748300      2.40942600     -9.85367120
Ag     5.18286600      4.99219200     -9.85367120
Ag     8.07186600      4.99219200     -9.85367120
Ag    10.96086600      4.99219200     -9.85367120
Ag    13.84986600      4.99219200     -9.85367120
Ag     6.47424900      7.57495800     -9.85367120
Ag     9.36324900      7.57495800     -9.85367120
Ag    12.25224900      7.57495800     -9.85367120
Ag     7.76563200     10.15772400     -9.85367120
Ag    10.65463200     10.15772400     -9.85367120
Ag     9.05701500     12.74049000     -9.85367120
Ag     5.18864400      3.21256800     -7.40091020
Ag     8.07764400      3.21256800     -7.40091020
Ag    10.96664400      3.21256800     -7.40091020
Ag    13.85564400      3.21256800     -7.40091020
Ag     6.48002700      5.79533400     -7.40091020
Ag     9.36902700      5.79533400     -7.40091020
Ag    12.25802700      5.79533400     -7.40091020
Ag     7.77141000      8.37810000     -7.40091020
Ag    10.66041000      8.37810000     -7.40091020
Ag     9.06279300     10.96086600     -7.40091020
Ag     6.48580500      4.01571000     -4.94814920
Ag     9.37480500      4.01571000     -4.94814920
Ag    12.26380500      4.01571000     -4.94814920
Ag     7.77718800      6.59847600     -4.94814920
Ag    10.66618800      6.59847600     -4.94814920
Ag     9.06857100      9.18124200     -4.94814920
Ag     7.78296600      4.81885200     -2.49538820
Ag    10.67196600      4.81885200     -2.49538820
Ag     9.07434900      7.40161800     -2.49538820
Ag     9.08012700      5.62199400     -0.04262720
Ag    -1.28560500      2.09163600    -15.69234020
Ag     0.01155600      2.89188900    -13.23957920
Ag     1.30871700      3.69214200    -10.78681820
Ag     2.60587800      4.49239500     -8.33405720
Ag     3.90303900      5.29264800     -5.88129620
Ag     5.20020000      6.09290100     -3.42853520
Ag     6.49736100      6.89315400     -0.97577420
Ag     0.00577800      4.67440200    -15.69234020
Ag     1.30293900      5.47465500    -13.23957920
Ag     2.60010000      6.27490800    -10.78681820
Ag     3.89726100      7.07516100     -8.33405720
Ag     5.19442200      7.87541400     -5.88129620
Ag     6.49158300      8.67566700     -3.42853520
```

```
Ag     1.29716100     7.25716800    -15.69234020
Ag     2.59432200     8.05742100    -13.23957920
Ag     3.89148300     8.85767400    -10.78681820
Ag     5.18864400     9.65792700     -8.33405720
Ag     6.48580500    10.45818000     -5.88129620
Ag     2.58854400     9.83993400    -15.69234020
Ag     3.88570500    10.64018700    -13.23957920
Ag     5.18286600    11.44044000    -10.78681820
Ag     6.48002700    12.24069300     -8.33405720
Ag     3.87992700    12.42270000    -15.69234020
Ag     5.17708800    13.22295300    -13.23957920
Ag     6.47424900    14.02320600    -10.78681820
Ag     5.17131000    15.00546600    -15.69234020
Ag     6.46847100    15.80571900    -13.23957920
Ag     6.46269300    17.58823200    -15.69234020
Ag    -2.57121000     4.18327200    -14.17272620
Ag    -1.27404900     4.98352500    -11.71996520
Ag     0.02311200     5.78377800     -9.26720420
Ag     1.32027300     6.58403100     -6.81444320
Ag     2.61743400     7.38428400     -4.36168220
Ag     3.91459500     8.18453700     -1.90892120
Ag    -1.27982700     6.76603800    -14.17272620
Ag     0.01733400     7.56629100    -11.71996520
Ag     1.31449500     8.36654400     -9.26720420
Ag     2.61165600     9.16679700     -6.81444320
Ag     3.90881700     9.96705000     -4.36168220
Ag     0.01155600     9.34880400    -14.17272620
Ag     1.30871700    10.14905700    -11.71996520
Ag     2.60587800    10.94931000     -9.26720420
Ag     3.90303900    11.74956300     -6.81444320
Ag     1.30293900    11.93157000    -14.17272620
Ag     2.60010000    12.73182300    -11.71996520
Ag     3.89726100    13.53207600     -9.26720420
Ag     2.59432200    14.51433600    -14.17272620
Ag     3.89148300    15.31458900    -11.71996520
Ag     3.88570500    17.09710200    -14.17272620
Ag    -3.85681500     6.27490800    -12.65311220
Ag    -2.55965400     7.07516100    -10.20035120
Ag    -1.26249300     7.87541400     -7.74759020
Ag     0.03466800     8.67566700     -5.29482920
Ag     1.33182900     9.47592000     -2.84206820
Ag    -2.56543200     8.85767400    -12.65311220
Ag    -1.26827100     9.65792700    -10.20035120
Ag     0.02889000    10.45818000     -7.74759020
Ag     1.32605100    11.25843300     -5.29482920
Ag    -1.27404900    11.44044000    -12.65311220
Ag     0.02311200    12.24069300    -10.20035120
Ag     1.32027300    13.04094600     -7.74759020
Ag     0.01733400    14.02320600    -12.65311220
Ag     1.31449500    14.82345900    -10.20035120
Ag     1.30871700    16.60597200    -12.65311220
Ag    -5.14242000     8.36654400    -11.13349820
Ag    -3.84525900     9.16679700     -8.68073720
Ag    -2.54809800     9.96705000     -6.22797620
Ag    -1.25093700    10.76730300     -3.77521520
Ag    -3.85103700    10.94931000    -11.13349820
Ag    -2.55387600    11.74956300     -8.68073720
Ag    -1.25671500    12.54981600     -6.22797620
```

```
Ag   -2.55965400   13.53207600  -11.13349820
Ag   -1.26249300   14.33232900   -8.68073720
Ag   -1.26827100   16.11484200  -11.13349820
Ag   -6.42802500   10.45818000   -9.61388420
Ag   -5.13086400   11.25843300   -7.16112320
Ag   -3.83370300   12.05868600   -4.70836220
Ag   -5.13664200   13.04094600   -9.61388420
Ag   -3.83948100   13.84119900   -7.16112320
Ag   -3.84525900   15.62371200   -9.61388420
Ag   -7.71363000   12.54981600   -8.09427020
Ag   -6.41646900   13.35006900   -5.64150920
Ag   -6.42224700   15.13258200   -8.09427020
Ag   -8.99923500   14.64145200   -6.57465620
Ag    1.29138300    0.79811860  -19.66831060
Ag    4.18038300    0.79811860  -19.66831060
Ag    7.06938300    0.79811860  -19.66831060
Ag    9.95838300    0.79811860  -19.66831060
Ag   12.84738300    0.79811860  -19.66831060
Ag   15.73638300    0.79811860  -19.66831060
Ag   18.62538300    0.79811860  -19.66831060
Ag    2.58276600    1.59623720  -22.12466710
Ag    5.47176600    1.59623720  -22.12466710
Ag    8.36076600    1.59623720  -22.12466710
Ag   11.24976600    1.59623720  -22.12466710
Ag   14.13876600    1.59623720  -22.12466710
Ag   17.02776600    1.59623720  -22.12466710
Ag    3.87414900    2.39435580  -24.58102350
Ag    6.76314900    2.39435580  -24.58102350
Ag    9.65214900    2.39435580  -24.58102350
Ag   12.54114900    2.39435580  -24.58102350
Ag   15.43014900    2.39435580  -24.58102350
Ag    5.16553200    3.19247430  -27.03737990
Ag    8.05453200    3.19247430  -27.03737990
Ag   10.94353200    3.19247430  -27.03737990
Ag   13.83253200    3.19247430  -27.03737990
Ag    6.45691500    3.99059290  -29.49373640
Ag    9.34591500    3.99059290  -29.49373640
Ag   12.23491500    3.99059290  -29.49373640
Ag    7.74829800    4.78871150  -31.95009280
Ag   10.63729800    4.78871150  -31.95009280
Ag    9.03968100    5.58683010  -34.40644920
Ag    2.58854400    3.37901740  -19.67419920
Ag    5.47754400    3.37901740  -19.67419920
Ag    8.36654400    3.37901740  -19.67419920
Ag   11.25554400    3.37901740  -19.67419920
Ag   14.14454400    3.37901740  -19.67419920
Ag   17.03354400    3.37901740  -19.67419920
Ag    3.87992700    4.17713600  -22.13055570
Ag    6.76892700    4.17713600  -22.13055570
Ag    9.65792700    4.17713600  -22.13055570
Ag   12.54692700    4.17713600  -22.13055570
Ag   15.43592700    4.17713600  -22.13055570
Ag    5.17131000    4.97525460  -24.58691210
Ag    8.06031000    4.97525460  -24.58691210
Ag   10.94931000    4.97525460  -24.58691210
Ag   13.83831000    4.97525460  -24.58691210
Ag    6.46269300    5.77337320  -27.04326850
Ag    9.35169300    5.77337320  -27.04326850
```

```
Ag    12.24069300     5.77337320   -27.04326850
Ag     7.75407600     6.57149180   -29.49962500
Ag    10.64307600     6.57149180   -29.49962500
Ag     9.04545900     7.36961040   -31.95598140
Ag     3.88570500     5.95991630   -19.68008780
Ag     6.77470500     5.95991630   -19.68008780
Ag     9.66370500     5.95991630   -19.68008780
Ag    12.55270500     5.95991630   -19.68008780
Ag    15.44170500     5.95991630   -19.68008780
Ag     5.17708800     6.75803490   -22.13644430
Ag     8.06608800     6.75803490   -22.13644430
Ag    10.95508800     6.75803490   -22.13644430
Ag    13.84408800     6.75803490   -22.13644430
Ag     6.46847100     7.55615350   -24.59280070
Ag     9.35747100     7.55615350   -24.59280070
Ag    12.24647100     7.55615350   -24.59280070
Ag     7.75985400     8.35427210   -27.04915710
Ag    10.64885400     8.35427210   -27.04915710
Ag     9.05123700     9.15239070   -29.50551360
Ag     5.18286600     8.54081520   -19.68597640
Ag     8.07186600     8.54081520   -19.68597640
Ag    10.96086600     8.54081520   -19.68597640
Ag    13.84986600     8.54081520   -19.68597640
Ag     6.47424900     9.33893370   -22.14233290
Ag     9.36324900     9.33893370   -22.14233290
Ag    12.25224900     9.33893370   -22.14233290
Ag     7.76563200    10.13705230   -24.59868930
Ag    10.65463200    10.13705230   -24.59868930
Ag     9.05701500    10.93517090   -27.05504570
Ag     6.48002700    11.12171400   -19.69186500
Ag     9.36902700    11.12171400   -19.69186500
Ag    12.25802700    11.12171400   -19.69186500
Ag     7.77141000    11.91983260   -22.14822150
Ag    10.66041000    11.91983260   -22.14822150
Ag     9.06279300    12.71795120   -24.60457790
Ag     7.77718800    13.70261290   -19.69775360
Ag    10.66618800    13.70261290   -19.69775360
Ag     9.06857100    14.50073150   -22.15411010
Ag     9.07434900    16.28351170   -19.70364220
Ag    -1.28560500     2.09158990   -18.73163170
Ag     0.01155600     4.67159600   -18.73477270
Ag     1.30871700     7.25160210   -18.73791370
Ag     2.60587800     9.83160820   -18.74105470
Ag     3.90303900    12.41161430   -18.74419570
Ag     5.20020000    14.99162040   -18.74733670
Ag     6.49736100    17.57162650   -18.75047770
Ag     0.00577800     2.88970850   -21.18798810
Ag     1.30293900     5.46971460   -21.19112910
Ag     2.60010000     8.04972070   -21.19427010
Ag     3.89726100    10.62972680   -21.19741110
Ag     5.19442200    13.20973290   -21.20055210
Ag     6.49158300    15.78973900   -21.20369310
Ag     1.29716100     3.68782700   -23.64434460
Ag     2.59432200     6.26783310   -23.64748560
Ag     3.89148300     8.84783930   -23.65062660
Ag     5.18864400    11.42784540   -23.65376760
Ag     6.48580500    14.00785150   -23.65690860
Ag     2.58854400     4.48594560   -26.10070100
```

```
Ag    3.88570500    7.06595170   -26.10384200
Ag    5.18286600    9.64595780   -26.10698300
Ag    6.48002700   12.22596400   -26.11012400
Ag    3.87992700    5.28406420   -28.55705740
Ag    5.17708800    7.86407030   -28.56019840
Ag    6.47424900   10.44407640   -28.56333940
Ag    5.17131000    6.08218280   -31.01341390
Ag    6.46847100    8.66218890   -31.01655490
Ag    6.46269300    6.88030140   -33.46977030
Ag   -2.57121000    4.18317970   -20.25130920
Ag   -1.27404900    6.76318580   -20.25445020
Ag    0.02311200    9.34319200   -20.25759120
Ag    1.32027300   11.92319810   -20.26073220
Ag    2.61743400   14.50320420   -20.26387320
Ag    3.91459500   17.08321030   -20.26701420
Ag   -1.27982700    4.98129830   -22.70766560
Ag    0.01733400    7.56130440   -22.71080660
Ag    1.31449500   10.14131050   -22.71394760
Ag    2.61165600   12.72131660   -22.71708860
Ag    3.90881700   15.30132280   -22.72022960
Ag    0.01155600    5.77941690   -25.16402210
Ag    1.30871700    8.35942300   -25.16716310
Ag    2.60587800   10.93942910   -25.17030410
Ag    3.90303900   13.51943520   -25.17344510
Ag    1.30293900    6.57753550   -27.62037850
Ag    2.60010000    9.15754160   -27.62351950
Ag    3.89726100   11.73754770   -27.62666050
Ag    2.59432200    7.37565410   -30.07673490
Ag    3.89148300    9.95566020   -30.07987590
Ag    3.88570500    8.17377270   -32.53309140
Ag   -3.85681500    6.27476960   -21.77098670
Ag   -2.55965400    8.85477570   -21.77412770
Ag   -1.26249300   11.43478180   -21.77726870
Ag    0.03466800   14.01478790   -21.78040970
Ag    1.33182900   16.59479400   -21.78355070
Ag   -2.56543200    7.07288820   -24.22734310
Ag   -1.26827100    9.65289430   -24.23048410
Ag    0.02889000   12.23290040   -24.23362510
Ag    1.32605100   14.81290650   -24.23676610
Ag   -1.27404900    7.87100680   -26.68369960
Ag    0.02311200   10.45101290   -26.68684060
Ag    1.32027300   13.03101900   -26.68998160
Ag    0.01733400    8.66912540   -29.14005600
Ag    1.31449500   11.24913150   -29.14319700
Ag    1.30871700    9.46724390   -31.59641240
Ag   -5.14242000    8.36635950   -23.29066420
Ag   -3.84525900   10.94636560   -23.29380520
Ag   -2.54809800   13.52637170   -23.29694620
Ag   -1.25093700   16.10637780   -23.30008720
Ag   -3.85103700    9.16447810   -25.74702060
Ag   -2.55387600   11.74448420   -25.75016160
Ag   -1.25671500   14.32449030   -25.75330260
Ag   -2.55965400    9.96259660   -28.20337710
Ag   -1.26249300   12.54260270   -28.20651810
Ag   -1.26827100   10.76071520   -30.65973350
Ag   -6.42802500   10.45794930   -24.81034170
Ag   -5.13086400   13.03795540   -24.81348270
Ag   -3.83370300   15.61796160   -24.81662370
```

```
Ag   -5.13664200   11.25606790   -27.26669810
Ag   -3.83948100   13.83607400   -27.26983910
Ag   -3.84525900   12.05418650   -29.72305460
Ag   -7.71363000   12.54953920   -26.33001920
Ag   -6.41646900   15.12954530   -26.33316020
Ag   -6.42224700   13.34765780   -28.78637560
Ag   -8.99923500   14.64112910   -27.84969670
Ag    1.29138300   -2.08950160   -18.73006600
Ag    4.18038300   -2.08950160   -18.73006600
Ag    7.06938300   -2.08950160   -18.73006600
Ag    9.95838300   -2.08950160   -18.73006600
Ag   12.84738300   -2.08950160   -18.73006600
Ag   15.73638300   -2.08950160   -18.73006600
Ag   18.62538300   -2.08950160   -18.73006600
Ag    2.58276600   -4.17900320   -20.24817770
Ag    5.47176600   -4.17900320   -20.24817770
Ag    8.36076600   -4.17900320   -20.24817770
Ag   11.24976600   -4.17900320   -20.24817770
Ag   14.13876600   -4.17900320   -20.24817770
Ag   17.02776600   -4.17900320   -20.24817770
Ag    3.87414900   -6.26850480   -21.76628950
Ag    6.76314900   -6.26850480   -21.76628950
Ag    9.65214900   -6.26850480   -21.76628950
Ag   12.54114900   -6.26850480   -21.76628950
Ag   15.43014900   -6.26850480   -21.76628950
Ag    5.16553200   -8.35800630   -23.28440130
Ag    8.05453200   -8.35800630   -23.28440130
Ag   10.94353200   -8.35800630   -23.28440130
Ag   13.83253200   -8.35800630   -23.28440130
Ag    6.45691500  -10.44750790   -24.80251300
Ag    9.34591500  -10.44750790   -24.80251300
Ag   12.23491500  -10.44750790   -24.80251300
Ag    7.74829800  -12.53700950   -26.32062480
Ag   10.63729800  -12.53700950   -26.32062480
Ag    9.03968100  -14.62651110   -27.83873660
Ag    2.58854400   -1.29756040   -21.18646630
Ag    5.47754400   -1.29756040   -21.18646630
Ag    8.36654400   -1.29756040   -21.18646630
Ag   11.25554400   -1.29756040   -21.18646630
Ag   14.14454400   -1.29756040   -21.18646630
Ag   17.03354400   -1.29756040   -21.18646630
Ag    3.87992700   -3.38706200   -22.70457810
Ag    6.76892700   -3.38706200   -22.70457810
Ag    9.65792700   -3.38706200   -22.70457810
Ag   12.54692700   -3.38706200   -22.70457810
Ag   15.43592700   -3.38706200   -22.70457810
Ag    5.17131000   -5.47656350   -24.22268990
Ag    8.06031000   -5.47656350   -24.22268990
Ag   10.94931000   -5.47656350   -24.22268990
Ag   13.83831000   -5.47656350   -24.22268990
Ag    6.46269300   -7.56606510   -25.74080160
Ag    9.35169300   -7.56606510   -25.74080160
Ag   12.24069300   -7.56606510   -25.74080160
Ag    7.75407600   -9.65556670   -27.25891340
Ag   10.64307600   -9.65556670   -27.25891340
Ag    9.04545900  -11.74506830   -28.77702510
Ag    3.88570500   -0.50561920   -23.64286670
Ag    6.77470500   -0.50561920   -23.64286670
```

```
Ag     9.66370500    -0.50561920   -23.64286670
Ag    12.55270500    -0.50561920   -23.64286670
Ag    15.44170500    -0.50561920   -23.64286670
Ag     5.17708800    -2.59512070   -25.16097840
Ag     8.06608800    -2.59512070   -25.16097840
Ag    10.95508800    -2.59512070   -25.16097840
Ag    13.84408800    -2.59512070   -25.16097840
Ag     6.46847100    -4.68462230   -26.67909020
Ag     9.35747100    -4.68462230   -26.67909020
Ag    12.24647100    -4.68462230   -26.67909020
Ag     7.75985400    -6.77412390   -28.19720200
Ag    10.64885400    -6.77412390   -28.19720200
Ag     9.05123700    -8.86362550   -29.71531370
Ag     5.18286600     0.28632210   -26.09926700
Ag     8.07186600     0.28632210   -26.09926700
Ag    10.96086600     0.28632210   -26.09926700
Ag    13.84986600     0.28632210   -26.09926700
Ag     6.47424900    -1.80317950   -27.61737880
Ag     9.36324900    -1.80317950   -27.61737880
Ag    12.25224900    -1.80317950   -27.61737880
Ag     7.76563200    -3.89268110   -29.13549060
Ag    10.65463200    -3.89268110   -29.13549060
Ag     9.05701500    -5.98218270   -30.65360230
Ag     6.48002700     1.07826330   -28.55566740
Ag     9.36902700     1.07826330   -28.55566740
Ag    12.25802700     1.07826330   -28.55566740
Ag     7.77141000    -1.01123830   -30.07377920
Ag    10.66041000    -1.01123830   -30.07377920
Ag     9.06279300    -3.10073990   -31.59189090
Ag     7.77718800     1.87020450   -31.01206770
Ag    10.66618800     1.87020450   -31.01206770
Ag     9.06857100    -0.21929710   -32.53017950
Ag     9.07434900     2.66214570   -33.46846810
Ag    -1.28560500    -0.79896240   -19.67078050
Ag     0.01155600    -0.00468390   -22.12548280
Ag     1.30871700     0.78959460   -24.58018500
Ag     2.60587800     1.58387300   -27.03488730
Ag     3.90303900     2.37815150   -29.48958950
Ag     5.20020000     3.17243000   -31.94429180
Ag     6.49736100     3.96670840   -34.39899400
Ag     0.00577800    -2.88846400   -21.18889230
Ag     1.30293900    -2.09418550   -23.64359460
Ag     2.60010000    -1.29990700   -26.09829680
Ag     3.89726100    -0.50562860   -28.55299900
Ag     5.19442200     0.28864990   -31.00770130
Ag     6.49158300     1.08292840   -33.46240350
Ag     1.29716100    -4.97796550   -22.70700410
Ag     2.59432200    -4.18368710   -25.16170630
Ag     3.89148300    -3.38940860   -27.61640860
Ag     5.18864400    -2.59513010   -30.07111080
Ag     6.48580500    -1.80085170   -32.52581300
Ag     2.58854400    -7.06746710   -24.22511580
Ag     3.88570500    -6.27318870   -26.67981810
Ag     5.18286600    -5.47891020   -29.13452030
Ag     6.48002700    -4.68463170   -31.58922260
Ag     3.87992700    -9.15696870   -25.74322760
Ag     5.17708800    -8.36269030   -28.19792980
Ag     6.47424900    -7.56841180   -30.65263210
```

```
Ag     5.17131000   -11.24647030   -27.26133940
Ag     6.46847100   -10.45219180   -29.71604160
Ag     6.46269300   -13.33597190   -28.77945110
Ag    -2.57121000    -1.59792470   -22.12960690
Ag    -1.27404900    -0.80364630   -24.58430910
Ag     0.02311200    -0.00936780   -27.03901140
Ag     1.32027300     0.78491070   -29.49371360
Ag     2.61743400     1.57918910   -31.94841590
Ag     3.91459500     2.37346760   -34.40311810
Ag    -1.27982700    -3.68742630   -23.64771870
Ag     0.01733400    -2.89314790   -26.10242090
Ag     1.31449500    -2.09886940   -28.55712310
Ag     2.61165600    -1.30459090   -31.01182540
Ag     3.90881700    -0.51031250   -33.46652760
Ag     0.01155600    -5.77692790   -25.16583040
Ag     1.30871700    -4.98264940   -27.62053270
Ag     2.60587800    -4.18837100   -30.07523490
Ag     3.90303900    -3.39409250   -32.52993720
Ag     1.30293900    -7.86642950   -26.68394220
Ag     2.60010000    -7.07215100   -29.13864440
Ag     3.89726100    -6.27787260   -31.59334670
Ag     2.59432200    -9.95593110   -28.20205390
Ag     3.89148300    -9.16165260   -30.65675620
Ag     3.88570500   -12.04543270   -29.72016570
Ag    -3.85681500    -2.39688710   -24.58843320
Ag    -2.55965400    -1.60260860   -27.04313550
Ag    -1.26249300    -0.80833020   -29.49783770
Ag     0.03466800    -0.01405170   -31.95254000
Ag     1.33182900     0.78022680   -34.40724220
Ag    -2.56543200    -4.48638870   -26.10654500
Ag    -1.26827100    -3.69211020   -28.56124720
Ag     0.02889000    -2.89783180   -31.01594950
Ag     1.32605100    -2.10355330   -33.47065170
Ag    -1.27404900    -6.57589030   -27.62465680
Ag     0.02311200    -5.78161180   -30.07935900
Ag     1.32027300    -4.98733340   -32.53406130
Ag     0.01733400    -8.66539190   -29.14276850
Ag     1.31449500    -7.87111340   -31.59747080
Ag     1.30871700   -10.75489350   -30.66088030
Ag    -5.14242000    -3.19584950   -27.04725960
Ag    -3.84525900    -2.40157100   -29.50196180
Ag    -2.54809800    -1.60729260   -31.95666410
Ag    -1.25093700    -0.81301410   -34.41136630
Ag    -3.85103700    -5.28535110   -28.56537130
Ag    -2.55387600    -4.49107260   -31.02007360
Ag    -1.25671500    -3.69679410   -33.47477580
Ag    -2.55965400    -7.37485270   -30.08348310
Ag    -1.26249300    -6.58057420   -32.53818540
Ag    -1.26827100    -9.46435420   -31.60159490
Ag    -6.42802500    -3.99481190   -29.50608590
Ag    -5.13086400    -3.20053340   -31.96078820
Ag    -3.83370300    -2.40625490   -34.41549040
Ag    -5.13664200    -6.08431340   -31.02419770
Ag    -3.83948100    -5.29003500   -33.47889990
Ag    -3.84525900    -8.17381500   -32.54230950
Ag    -7.71363000    -4.79377420   -31.96491230
Ag    -6.41646900    -3.99949580   -34.41961450
Ag    -6.42224700    -6.88327580   -33.48302400
```

```
Ag   -8.99923500   -5.59273660  -34.42373860
Ag    1.29138300   -2.08950160  -15.69384240
Ag    4.18038300   -2.08950160  -15.69384240
Ag    7.06938300   -2.08950160  -15.69384240
Ag    9.95838300   -2.08950160  -15.69384240
Ag   12.84738300   -2.08950160  -15.69384240
Ag   15.73638300   -2.08950160  -15.69384240
Ag   18.62538300   -2.08950160  -15.69384240
Ag    2.58276600   -4.17900320  -14.17573070
Ag    5.47176600   -4.17900320  -14.17573070
Ag    8.36076600   -4.17900320  -14.17573070
Ag   11.24976600   -4.17900320  -14.17573070
Ag   14.13876600   -4.17900320  -14.17573070
Ag   17.02776600   -4.17900320  -14.17573070
Ag    3.87414900   -6.26850480  -12.65761890
Ag    6.76314900   -6.26850480  -12.65761890
Ag    9.65214900   -6.26850480  -12.65761890
Ag   12.54114900   -6.26850480  -12.65761890
Ag   15.43014900   -6.26850480  -12.65761890
Ag    5.16553200   -8.35800630  -11.13950710
Ag    8.05453200   -8.35800630  -11.13950710
Ag   10.94353200   -8.35800630  -11.13950710
Ag   13.83253200   -8.35800630  -11.13950710
Ag    6.45691500  -10.44750790   -9.62139540
Ag    9.34591500  -10.44750790   -9.62139540
Ag   12.23491500  -10.44750790   -9.62139540
Ag    7.74829800  -12.53700950   -8.10328360
Ag   10.63729800  -12.53700950   -8.10328360
Ag    9.03968100  -14.62651110   -6.58517180
Ag    2.58854400   -4.18095390  -17.20609270
Ag    5.47754400   -4.18095390  -17.20609270
Ag    8.36654400   -4.18095390  -17.20609270
Ag   11.25554400   -4.18095390  -17.20609270
Ag   14.14454400   -4.18095390  -17.20609270
Ag   17.03354400   -4.18095390  -17.20609270
Ag    3.87992700   -6.27045540  -15.68798100
Ag    6.76892700   -6.27045540  -15.68798100
Ag    9.65792700   -6.27045540  -15.68798100
Ag   12.54692700   -6.27045540  -15.68798100
Ag   15.43592700   -6.27045540  -15.68798100
Ag    5.17131000   -8.35995700  -14.16986920
Ag    8.06031000   -8.35995700  -14.16986920
Ag   10.94931000   -8.35995700  -14.16986920
Ag   13.83831000   -8.35995700  -14.16986920
Ag    6.46269300  -10.44945860  -12.65175740
Ag    9.35169300  -10.44945860  -12.65175740
Ag   12.24069300  -10.44945860  -12.65175740
Ag    7.75407600  -12.53896020  -11.13364570
Ag   10.64307600  -12.53896020  -11.13364570
Ag    9.04545900  -14.62846180   -9.61553390
Ag    3.88570500   -6.27240610  -18.71834310
Ag    6.77470500   -6.27240610  -18.71834310
Ag    9.66370500   -6.27240610  -18.71834310
Ag   12.55270500   -6.27240610  -18.71834310
Ag   15.44170500   -6.27240610  -18.71834310
Ag    5.17708800   -8.36190770  -17.20023130
Ag    8.06608800   -8.36190770  -17.20023130
Ag   10.95508800   -8.36190770  -17.20023130
```

```
Ag   13.84408800    -8.36190770   -17.20023130
Ag    6.46847100   -10.45140930   -15.68211950
Ag    9.35747100   -10.45140930   -15.68211950
Ag   12.24647100   -10.45140930   -15.68211950
Ag    7.75985400   -12.54091090   -14.16400780
Ag   10.64885400   -12.54091090   -14.16400780
Ag    9.05123700   -14.63041250   -12.64589600
Ag    5.18286600    -8.36385840   -20.23059340
Ag    8.07186600    -8.36385840   -20.23059340
Ag   10.96086600    -8.36385840   -20.23059340
Ag   13.84986600    -8.36385840   -20.23059340
Ag    6.47424900   -10.45336000   -18.71248160
Ag    9.36324900   -10.45336000   -18.71248160
Ag   12.25224900   -10.45336000   -18.71248160
Ag    7.76563200   -12.54286160   -17.19436980
Ag   10.65463200   -12.54286160   -17.19436980
Ag    9.05701500   -14.63236320   -15.67625810
Ag    6.48002700   -10.45531070   -21.74284370
Ag    9.36902700   -10.45531070   -21.74284370
Ag   12.25802700   -10.45531070   -21.74284370
Ag    7.77141000   -12.54481230   -20.22473190
Ag   10.66041000   -12.54481230   -20.22473190
Ag    9.06279300   -14.63431380   -18.70662010
Ag    7.77718800   -12.54676290   -23.25509400
Ag   10.66618800   -12.54676290   -23.25509400
Ag    9.06857100   -14.63626450   -21.73698220
Ag    9.07434900   -14.63821520   -24.76734430
Ag   -1.28560500    -2.58537580   -17.21191500
Ag    0.01155600    -4.67449080   -18.72586340
Ag    1.30871700    -6.76360580   -20.23981180
Ag    2.60587800    -8.85272080   -21.75376020
Ag    3.90303900   -10.94183580   -23.26770860
Ag    5.20020000   -13.03095090   -24.78165710
Ag    6.49736100   -15.12006590   -26.29560550
Ag    0.00577800    -4.67487740   -15.69380320
Ag    1.30293900    -6.76399240   -17.20775160
Ag    2.60010000    -8.85310740   -18.72170000
Ag    3.89726100   -10.94222240   -20.23564850
Ag    5.19442200   -13.03133740   -21.74959690
Ag    6.49158300   -15.12045250   -23.26354530
Ag    1.29716100    -6.76437890   -14.17569140
Ag    2.59432200    -8.85349400   -15.68963980
Ag    3.89148300   -10.94260900   -17.20358830
Ag    5.18864400   -13.03172400   -18.71753670
Ag    6.48580500   -15.12083900   -20.23148510
Ag    2.58854400    -8.85388050   -12.65757970
Ag    3.88570500   -10.94299550   -14.17152810
Ag    5.18286600   -13.03211060   -15.68547650
Ag    6.48002700   -15.12122560   -17.19942490
Ag    3.87992700   -10.94338210   -11.13946790
Ag    5.17708800   -13.03249710   -12.65341630
Ag    6.47424900   -15.12161220   -14.16736470
Ag    5.17131000   -13.03288370    -9.62135610
Ag    6.46847100   -15.12199870   -11.13530460
Ag    6.46269300   -15.12238530    -8.10324440
Ag   -2.57121000    -5.17075150   -17.21187570
Ag   -1.27404900    -7.25986660   -18.72582410
Ag    0.02311200    -9.34898160   -20.23977260
```

```
Ag     1.32027300   -11.43809660   -21.75372100
Ag     2.61743400   -13.52721160   -23.26766940
Ag     3.91459500   -15.61632660   -24.78161780
Ag    -1.27982700    -7.26025310   -15.69376390
Ag     0.01733400    -9.34936810   -17.20771240
Ag     1.31449500   -11.43848320   -18.72166080
Ag     2.61165600   -13.52759820   -20.23560920
Ag     3.90881700   -15.61671320   -21.74955760
Ag     0.01155600    -9.34975470   -14.17565220
Ag     1.30871700   -11.43886970   -15.68960060
Ag     2.60587800   -13.52798470   -17.20354900
Ag     3.90303900   -15.61709980   -18.71749740
Ag     1.30293900   -11.43925630   -12.65754040
Ag     2.60010000   -13.52837130   -14.17148880
Ag     3.89726100   -15.61748630   -15.68543730
Ag     2.59432200   -13.52875790   -11.13942870
Ag     3.89148300   -15.61787290   -12.65337710
Ag     3.88570500   -15.61825950    -9.62131690
Ag    -3.85681500    -7.75612730   -17.21183650
Ag    -2.55965400    -9.84524230   -18.72578490
Ag    -1.26249300   -11.93435730   -20.23973330
Ag     0.03466800   -14.02347240   -21.75368170
Ag     1.33182900   -16.11258740   -23.26763020
Ag    -2.56543200    -9.84562890   -15.69372470
Ag    -1.26827100   -11.93474390   -17.20767310
Ag     0.02889000   -14.02385890   -18.72162150
Ag     1.32605100   -16.11297400   -20.23557000
Ag    -1.27404900   -11.93513050   -14.17561290
Ag     0.02311200   -14.02424550   -15.68956140
Ag     1.32027300   -16.11336050   -17.20350980
Ag     0.01733400   -14.02463210   -12.65750120
Ag     1.31449500   -16.11374710   -14.17144960
Ag     1.30871700   -16.11413370   -11.13938940
Ag    -5.14242000   -10.34150310   -17.21179720
Ag    -3.84525900   -12.43061810   -18.72574560
Ag    -2.54809800   -14.51973310   -20.23969410
Ag    -1.25093700   -16.60884810   -21.75364250
Ag    -3.85103700   -12.43100470   -15.69368550
Ag    -2.55387600   -14.52011970   -17.20763390
Ag    -1.25671500   -16.60923470   -18.72158230
Ag    -2.55965400   -14.52050620   -14.17557370
Ag    -1.26249300   -16.60962130   -15.68952210
Ag    -1.26827100   -16.61000780   -12.65746190
Ag    -6.42802500   -12.92687880   -17.21175800
Ag    -5.13086400   -15.01599390   -18.72570640
Ag    -3.83370300   -17.10510890   -20.23965480
Ag    -5.13664200   -15.01638040   -15.69364620
Ag    -3.83948100   -17.10549540   -17.20759460
Ag    -3.84525900   -17.10588200   -14.17553450
Ag    -7.71363000   -15.51225460   -17.21171870
Ag    -6.41646900   -17.60136960   -18.72566720
Ag    -6.42224700   -17.60175620   -15.69360700
Ag    -8.99923500   -18.09763040   -17.21167950
Ag     2.58854400    -1.28641120   -13.23381950
Ag     5.47754400    -1.28641120   -13.23381950
Ag     8.36654400    -1.28641120   -13.23381950
Ag    11.25554400    -1.28641120   -13.23381950
Ag    14.14454400    -1.28641120   -13.23381950
```

```
Ag    17.03354400    -1.28641120   -13.23381950
Ag     3.87992700    -0.48829260   -10.77746310
Ag     6.76892700    -0.48829260   -10.77746310
Ag     9.65792700    -0.48829260   -10.77746310
Ag    12.54692700    -0.48829260   -10.77746310
Ag    15.43592700    -0.48829260   -10.77746310
Ag     5.17131000     0.30982600    -8.32110660
Ag     8.06031000     0.30982600    -8.32110660
Ag    10.94931000     0.30982600    -8.32110660
Ag    13.83831000     0.30982600    -8.32110660
Ag     6.46269300     1.10794450    -5.86475020
Ag     9.35169300     1.10794450    -5.86475020
Ag    12.24069300     1.10794450    -5.86475020
Ag     7.75407600     1.90606310    -3.40839380
Ag    10.64307600     1.90606310    -3.40839380
Ag     9.04545900     2.70418170    -0.95203730
Ag     3.88570500    -3.37094100   -11.71204120
Ag     6.77470500    -3.37094100   -11.71204120
Ag     9.66370500    -3.37094100   -11.71204120
Ag    12.55270500    -3.37094100   -11.71204120
Ag    15.44170500    -3.37094100   -11.71204120
Ag     5.17708800    -2.57282240    -9.25568480
Ag     8.06608800    -2.57282240    -9.25568480
Ag    10.95508800    -2.57282240    -9.25568480
Ag    13.84408800    -2.57282240    -9.25568480
Ag     6.46847100    -1.77470390    -6.79932840
Ag     9.35747100    -1.77470390    -6.79932840
Ag    12.24647100    -1.77470390    -6.79932840
Ag     7.75985400    -0.97658530    -4.34297190
Ag    10.64885400    -0.97658530    -4.34297190
Ag     9.05123700    -0.17846670    -1.88661550
Ag     5.18286600    -5.45547080   -10.19026300
Ag     8.07186600    -5.45547080   -10.19026300
Ag    10.96086600    -5.45547080   -10.19026300
Ag    13.84986600    -5.45547080   -10.19026300
Ag     6.47424900    -4.65735220    -7.73390650
Ag     9.36324900    -4.65735220    -7.73390650
Ag    12.25224900    -4.65735220    -7.73390650
Ag     7.76563200    -3.85923370    -5.27755010
Ag    10.65463200    -3.85923370    -5.27755010
Ag     9.05701500    -3.06111510    -2.82119370
Ag     6.48002700    -7.54000060    -8.66848470
Ag     9.36902700    -7.54000060    -8.66848470
Ag    12.25802700    -7.54000060    -8.66848470
Ag     7.77141000    -6.74188200    -6.21212830
Ag    10.66041000    -6.74188200    -6.21212830
Ag     9.06279300    -5.94376350    -3.75577180
Ag     7.77718800    -9.62453040    -7.14670640
Ag    10.66618800    -9.62453040    -7.14670640
Ag     9.06857100    -8.82641190    -4.69035000
Ag     9.07434900   -11.70906020    -5.62492820
Ag    -1.28560500    -0.79888770   -14.75310360
Ag     0.01155600    -2.88431030   -13.23407290
Ag     1.30871700    -4.96973280   -11.71504230
Ag     2.60587800    -7.05515540   -10.19601160
Ag     3.90303900    -9.14057790    -8.67698100
Ag     5.20020000   -11.22600050    -7.15795030
Ag     6.49736100   -13.31142310    -5.63891960
```

```
Ag     0.00577800    -0.00076910   -12.29674720
Ag     1.30293900    -2.08619170   -10.77771650
Ag     2.60010000    -4.17161430    -9.25868580
Ag     3.89726100    -6.25703680    -7.73965520
Ag     5.19442200    -8.34245940    -6.22062450
Ag     6.49158300   -10.42788190    -4.70159390
Ag     1.29716100     0.79734940    -9.84039070
Ag     2.59432200    -1.28807310    -8.32136010
Ag     3.89148300    -3.37349570    -6.80232940
Ag     5.18864400    -5.45891820    -5.28329870
Ag     6.48580500    -7.54434080    -3.76426810
Ag     2.58854400     1.59546800    -7.38403430
Ag     3.88570500    -0.48995450    -5.86500360
Ag     5.18286600    -2.57537710    -4.34597300
Ag     6.48002700    -4.66079960    -2.82694230
Ag     3.87992700     2.39358660    -4.92767790
Ag     5.17708800     0.30816410    -3.40864720
Ag     6.47424900    -1.77725850    -1.88961650
Ag     5.17131000     3.19170520    -2.47132140
Ag     6.46847100     1.10628270    -0.95229080
Ag     6.46269300     3.98982380    -0.01496500
Ag    -2.57121000    -1.59777550   -12.29425300
Ag    -1.27404900    -3.68319800   -10.77522230
Ag     0.02311200    -5.76862060    -9.25619170
Ag     1.32027300    -7.85404310    -7.73716100
Ag     2.61743400    -9.93946570    -6.21813040
Ag     3.91459500   -12.02488820    -4.69909970
Ag    -1.27982700    -0.79965690    -9.83789660
Ag     0.01733400    -2.88507940    -8.31886590
Ag     1.31449500    -4.97050200    -6.79983520
Ag     2.61165600    -7.05592450    -5.28080460
Ag     3.90881700    -9.14134710    -3.76177390
Ag     0.01155600    -0.00153830    -7.38154010
Ag     1.30871700    -2.08696080    -5.86250950
Ag     2.60587800    -4.17238340    -4.34347880
Ag     3.90303900    -6.25780590    -2.82444810
Ag     1.30293900     0.79658030    -4.92518370
Ag     2.60010000    -1.28884220    -3.40615300
Ag     3.89726100    -3.37426480    -1.88712240
Ag     2.59432200     1.59469890    -2.46882730
Ag     3.89148300    -0.49072370    -0.94979660
Ag     3.88570500     2.39281750    -0.01247080
Ag    -3.85681500    -2.39666320    -9.83540240
Ag    -2.55965400    -4.48208570    -8.31637170
Ag    -1.26249300    -6.56750830    -6.79734110
Ag     0.03466800    -8.65293080    -5.27831040
Ag     1.33182900   -10.73835340    -3.75927980
Ag    -2.56543200    -1.59854460    -7.37904600
Ag    -1.26827100    -3.68396710    -5.86001530
Ag     0.02889000    -5.76938970    -4.34098460
Ag     1.32605100    -7.85481230    -2.82195400
Ag    -1.27404900    -0.80042600    -4.92268950
Ag     0.02311200    -2.88584860    -3.40365890
Ag     1.32027300    -4.97127110    -1.88462820
Ag     0.01733400    -0.00230740    -2.46633310
Ag     1.31449500    -2.08773000    -0.94730240
Ag     1.30871700     0.79581120    -0.00997670
Ag    -5.14242000    -3.19555090    -7.37655180
```

```
Ag   -3.84525900   -5.28097350   -5.85752110
Ag   -2.54809800   -7.36639600   -4.33849050
Ag   -1.25093700   -9.45181860   -2.81945980
Ag   -3.85103700   -2.39743230   -4.92019540
Ag   -2.55387600   -4.48285490   -3.40116470
Ag   -1.25671500   -6.56827740   -1.88213400
Ag   -2.55965400   -1.59931370   -2.46383890
Ag   -1.26249300   -3.68473630   -0.94480830
Ag   -1.26827100   -0.80119510   -0.00748250
Ag   -6.42802500   -3.99443860   -4.91770120
Ag   -5.13086400   -6.07986120   -3.39867050
Ag   -3.83370300   -8.16528370   -1.87963990
Ag   -5.13664200   -3.19632000   -2.46134480
Ag   -3.83948100   -5.28174260   -0.94231410
Ag   -3.84525900   -2.39820150   -0.00498830
Ag   -7.71363000   -4.79332640   -2.45885060
Ag   -6.41646900   -6.87874890   -0.93981990
Ag   -6.42224700   -3.99520780   -0.00249420
Ag   -8.99923500   -5.59221410    0.00000000
Ag   -2.88900000    0.00000000  -17.21195420
Ag   -5.77800000    0.00000000  -17.21195420
Ag   -8.66700000    0.00000000  -17.21195420
Ag  -11.55600000    0.00000000  -17.21195420
Ag  -14.44500000    0.00000000  -17.21195420
Ag  -17.33400000    0.00000000  -17.21195420
Ag  -20.22300000    0.00000000  -17.21195420
Ag   -4.18038300    2.08950160  -18.73006600
Ag   -7.06938300    2.08950160  -18.73006600
Ag   -9.95838300    2.08950160  -18.73006600
Ag  -12.84738300    2.08950160  -18.73006600
Ag  -15.73638300    2.08950160  -18.73006600
Ag  -18.62538300    2.08950160  -18.73006600
Ag   -5.47176600    4.17900320  -20.24817770
Ag   -8.36076600    4.17900320  -20.24817770
Ag  -11.24976600    4.17900320  -20.24817770
Ag  -14.13876600    4.17900320  -20.24817770
Ag  -17.02776600    4.17900320  -20.24817770
Ag   -6.76314900    6.26850480  -21.76628950
Ag   -9.65214900    6.26850480  -21.76628950
Ag  -12.54114900    6.26850480  -21.76628950
Ag  -15.43014900    6.26850480  -21.76628950
Ag   -8.05453200    8.35800630  -23.28440130
Ag  -10.94353200    8.35800630  -23.28440130
Ag  -13.83253200    8.35800630  -23.28440130
Ag   -9.34591500   10.44750790  -24.80251300
Ag  -12.23491500   10.44750790  -24.80251300
Ag  -10.63729800   12.53700950  -26.32062480
Ag   -4.18616100    2.09145230  -15.69970390
Ag   -7.07516100    2.09145230  -15.69970390
Ag   -9.96416100    2.09145230  -15.69970390
Ag  -12.85316100    2.09145230  -15.69970390
Ag  -15.74216100    2.09145230  -15.69970390
Ag  -18.63116100    2.09145230  -15.69970390
Ag   -2.58854400    4.18095390  -17.21781570
Ag   -5.47754400    4.18095390  -17.21781570
Ag   -8.36654400    4.18095390  -17.21781570
Ag  -11.25554400    4.18095390  -17.21781570
Ag  -14.14454400    4.18095390  -17.21781570
```

```
Ag   -17.03354400     4.18095390   -17.21781570
Ag    -3.87992700     6.27045540   -18.73592740
Ag    -6.76892700     6.27045540   -18.73592740
Ag    -9.65792700     6.27045540   -18.73592740
Ag   -12.54692700     6.27045540   -18.73592740
Ag   -15.43592700     6.27045540   -18.73592740
Ag    -5.17131000     8.35995700   -20.25403920
Ag    -8.06031000     8.35995700   -20.25403920
Ag   -10.94931000     8.35995700   -20.25403920
Ag   -13.83831000     8.35995700   -20.25403920
Ag    -6.46269300    10.44945860   -21.77215100
Ag    -9.35169300    10.44945860   -21.77215100
Ag   -12.24069300    10.44945860   -21.77215100
Ag    -7.75407600    12.53896020   -23.29026270
Ag   -10.64307600    12.53896020   -23.29026270
Ag    -9.04545900    14.62846180   -24.80837450
Ag    -5.48332200     4.18290450   -14.18745360
Ag    -8.37232200     4.18290450   -14.18745360
Ag   -11.26132200     4.18290450   -14.18745360
Ag   -14.15032200     4.18290450   -14.18745360
Ag   -17.03932200     4.18290450   -14.18745360
Ag    -3.88570500     6.27240610   -15.70556530
Ag    -6.77470500     6.27240610   -15.70556530
Ag    -9.66370500     6.27240610   -15.70556530
Ag   -12.55270500     6.27240610   -15.70556530
Ag   -15.44170500     6.27240610   -15.70556530
Ag    -5.17708800     8.36190770   -17.22367710
Ag    -8.06608800     8.36190770   -17.22367710
Ag   -10.95508800     8.36190770   -17.22367710
Ag   -13.84408800     8.36190770   -17.22367710
Ag    -6.46847100    10.45140930   -18.74178890
Ag    -9.35747100    10.45140930   -18.74178890
Ag   -12.24647100    10.45140930   -18.74178890
Ag    -7.75985400    12.54091090   -20.25990060
Ag   -10.64885400    12.54091090   -20.25990060
Ag    -9.05123700    14.63041250   -21.77801240
Ag    -6.78048300     6.27435680   -12.67520330
Ag    -9.66948300     6.27435680   -12.67520330
Ag   -12.55848300     6.27435680   -12.67520330
Ag   -15.44748300     6.27435680   -12.67520330
Ag    -5.18286600     8.36385840   -14.19331500
Ag    -8.07186600     8.36385840   -14.19331500
Ag   -10.96086600     8.36385840   -14.19331500
Ag   -13.84986600     8.36385840   -14.19331500
Ag    -6.47424900    10.45336000   -15.71142680
Ag    -9.36324900    10.45336000   -15.71142680
Ag   -12.25224900    10.45336000   -15.71142680
Ag    -7.76563200    12.54286160   -17.22953860
Ag   -10.65463200    12.54286160   -17.22953860
Ag    -9.05701500    14.63236320   -18.74765030
Ag    -8.07764400     8.36580910   -11.16295300
Ag   -10.96664400     8.36580910   -11.16295300
Ag   -13.85564400     8.36580910   -11.16295300
Ag    -6.48002700    10.45531070   -12.68106470
Ag    -9.36902700    10.45531070   -12.68106470
Ag   -12.25802700    10.45531070   -12.68106470
Ag    -7.77141000    12.54481230   -14.19917650
Ag   -10.66041000    12.54481230   -14.19917650
```

```
Ag   -9.06279300   14.63431380  -15.71728830
Ag   -9.37480500   10.45726140   -9.65070270
Ag  -12.26380500   10.45726140   -9.65070270
Ag   -7.77718800   12.54676290  -11.16881440
Ag  -10.66618800   12.54676290  -11.16881440
Ag   -9.06857100   14.63626450  -12.68692620
Ag  -10.67196600   12.54871360   -8.13845230
Ag   -9.07434900   14.63821520   -9.65656410
Ag   -1.30293900    6.76399240  -17.21615680
Ag   -2.60010000    8.85310740  -15.70220840
Ag   -3.89726100   10.94222240  -14.18825990
Ag   -5.19442200   13.03133740  -12.67431150
Ag   -6.49158300   15.12045250  -11.16036310
Ag   -2.59432200    8.85349400  -18.73426860
Ag   -3.89148300   10.94260900  -17.22032010
Ag   -5.18864400   13.03172400  -15.70637170
Ag   -6.48580500   15.12083900  -14.19242330
Ag   -3.88570500   10.94299550  -20.25238030
Ag   -5.18286600   13.03211060  -18.73843190
Ag   -6.48002700   15.12122560  -17.22448350
Ag   -5.17708800   13.03249710  -21.77049210
Ag   -6.47424900   15.12161220  -20.25654370
Ag   -6.46847100   15.12199870  -23.28860380
Ag   -0.01733400    9.34936810  -17.21619600
Ag   -1.31449500   11.43848320  -15.70224760
Ag   -2.61165600   13.52759820  -14.18829920
Ag   -3.90881700   15.61671320  -12.67435080
Ag   -1.30871700   11.43886970  -18.73430780
Ag   -2.60587800   13.52798470  -17.22035940
Ag   -3.90303900   15.61709980  -15.70641100
Ag   -2.60010000   13.52837130  -20.25241960
Ag   -3.89726100   15.61748630  -18.73847110
Ag   -3.89148300   15.61787290  -21.77053130
Ag    1.26827100   11.93474390  -17.21623530
Ag   -0.02889000   14.02385890  -15.70228690
Ag   -1.32605100   16.11297400  -14.18833840
Ag   -0.02311200   14.02424550  -18.73434700
Ag   -1.32027300   16.11336050  -17.22039860
Ag   -1.31449500   16.11374710  -20.25245880
Ag    2.55387600   14.52011970  -17.21627450
Ag    1.25671500   16.60923470  -15.70232610
Ag    1.26249300   16.60962130  -18.73438630
Ag    3.83948100   17.10549540  -17.21631380
Ag   -4.18038300   -0.79811860  -19.66831060
Ag   -7.06938300   -0.79811860  -19.66831060
Ag   -9.95838300   -0.79811860  -19.66831060
Ag  -12.84738300   -0.79811860  -19.66831060
Ag  -15.73638300   -0.79811860  -19.66831060
Ag  -18.62538300   -0.79811860  -19.66831060
Ag   -5.47176600   -1.59623720  -22.12466710
Ag   -8.36076600   -1.59623720  -22.12466710
Ag  -11.24976600   -1.59623720  -22.12466710
Ag  -14.13876600   -1.59623720  -22.12466710
Ag  -17.02776600   -1.59623720  -22.12466710
Ag   -6.76314900   -2.39435580  -24.58102350
Ag   -9.65214900   -2.39435580  -24.58102350
Ag  -12.54114900   -2.39435580  -24.58102350
Ag  -15.43014900   -2.39435580  -24.58102350
```

```
Ag    -8.05453200   -3.19247430   -27.03737990
Ag   -10.94353200   -3.19247430   -27.03737990
Ag   -13.83253200   -3.19247430   -27.03737990
Ag    -9.34591500   -3.99059290   -29.49373640
Ag   -12.23491500   -3.99059290   -29.49373640
Ag   -10.63729800   -4.78871150   -31.95009280
Ag    -2.58854400    1.28641120   -21.19008890
Ag    -5.47754400    1.28641120   -21.19008890
Ag    -8.36654400    1.28641120   -21.19008890
Ag   -11.25554400    1.28641120   -21.19008890
Ag   -14.14454400    1.28641120   -21.19008890
Ag   -17.03354400    1.28641120   -21.19008890
Ag    -3.87992700    0.48829260   -23.64644530
Ag    -6.76892700    0.48829260   -23.64644530
Ag    -9.65792700    0.48829260   -23.64644530
Ag   -12.54692700    0.48829260   -23.64644530
Ag   -15.43592700    0.48829260   -23.64644530
Ag    -5.17131000   -0.30982600   -26.10280180
Ag    -8.06031000   -0.30982600   -26.10280180
Ag   -10.94931000   -0.30982600   -26.10280180
Ag   -13.83831000   -0.30982600   -26.10280180
Ag    -6.46269300   -1.10794450   -28.55915820
Ag    -9.35169300   -1.10794450   -28.55915820
Ag   -12.24069300   -1.10794450   -28.55915820
Ag    -7.75407600   -1.90606310   -31.01551460
Ag   -10.64307600   -1.90606310   -31.01551460
Ag    -9.04545900   -2.70418170   -33.47187110
Ag    -3.88570500    3.37094100   -22.71186720
Ag    -6.77470500    3.37094100   -22.71186720
Ag    -9.66370500    3.37094100   -22.71186720
Ag   -12.55270500    3.37094100   -22.71186720
Ag   -15.44170500    3.37094100   -22.71186720
Ag    -5.17708800    2.57282240   -25.16822360
Ag    -8.06608800    2.57282240   -25.16822360
Ag   -10.95508800    2.57282240   -25.16822360
Ag   -13.84408800    2.57282240   -25.16822360
Ag    -6.46847100    1.77470390   -27.62458000
Ag    -9.35747100    1.77470390   -27.62458000
Ag   -12.24647100    1.77470390   -27.62458000
Ag    -7.75985400    0.97658530   -30.08093650
Ag   -10.64885400    0.97658530   -30.08093650
Ag    -9.05123700    0.17846670   -32.53729290
Ag    -5.18286600    5.45547080   -24.23364540
Ag    -8.07186600    5.45547080   -24.23364540
Ag   -10.96086600    5.45547080   -24.23364540
Ag   -13.84986600    5.45547080   -24.23364540
Ag    -6.47424900    4.65735220   -26.69000190
Ag    -9.36324900    4.65735220   -26.69000190
Ag   -12.25224900    4.65735220   -26.69000190
Ag    -7.76563200    3.85923370   -29.14635830
Ag   -10.65463200    3.85923370   -29.14635830
Ag    -9.05701500    3.06111510   -31.60271470
Ag    -6.48002700    7.54000060   -25.75542370
Ag    -9.36902700    7.54000060   -25.75542370
Ag   -12.25802700    7.54000060   -25.75542370
Ag    -7.77141000    6.74188200   -28.21178010
Ag   -10.66041000    6.74188200   -28.21178010
Ag    -9.06279300    5.94376350   -30.66813660
```

```
Ag   -7.77718800     9.62453040   -27.27720200
Ag  -10.66618800     9.62453040   -27.27720200
Ag   -9.06857100     8.82641190   -29.73355840
Ag   -9.07434900    11.70906020   -28.79898020
Ag   -1.30293900     2.08619170   -23.64619190
Ag   -2.60010000     4.17161430   -25.16522260
Ag   -3.89726100     6.25703680   -26.68425320
Ag   -5.19442200     8.34245940   -28.20328390
Ag   -6.49158300    10.42788190   -29.72231450
Ag   -2.59432200     1.28807310   -26.10254830
Ag   -3.89148300     3.37349570   -27.62157900
Ag   -5.18864400     5.45891820   -29.14060970
Ag   -6.48580500     7.54434080   -30.65964030
Ag   -3.88570500     0.48995450   -28.55890480
Ag   -5.18286600     2.57537710   -30.07793540
Ag   -6.48002700     4.66079960   -31.59696610
Ag   -5.17708800    -0.30816410   -31.01526120
Ag   -6.47424900     1.77725850   -32.53429190
Ag   -6.46847100    -1.10628270   -33.47161760
Ag   -0.01733400     2.88507940   -26.10504250
Ag   -1.31449500     4.97050200   -27.62407320
Ag   -2.61165600     7.05592450   -29.14310380
Ag   -3.90881700     9.14134710   -30.66213450
Ag   -1.30871700     2.08696080   -28.56139890
Ag   -2.60587800     4.17238340   -30.08042960
Ag   -3.90303900     6.25780590   -31.59946030
Ag   -2.60010000     1.28884220   -31.01775540
Ag   -3.89726100     3.37426480   -32.53678600
Ag   -3.89148300     0.49072370   -33.47411180
Ag    1.26827100     3.68396710   -28.56389310
Ag   -0.02889000     5.76938970   -30.08292380
Ag   -1.32605100     7.85481230   -31.60195440
Ag   -0.02311200     2.88584860   -31.02024950
Ag   -1.32027300     4.97127110   -32.53928020
Ag   -1.31449500     2.08773000   -33.47660600
Ag    2.55387600     4.48285490   -31.02274370
Ag    1.25671500     6.56827740   -32.54177440
Ag    1.26249300     3.68473630   -33.47910010
Ag    3.83948100     5.28174260   -33.48159430
Ag   -4.18038300    -2.58276600   -17.21195420
Ag   -7.06938300    -2.58276600   -17.21195420
Ag   -9.95838300    -2.58276600   -17.21195420
Ag  -12.84738300    -2.58276600   -17.21195420
Ag  -15.73638300    -2.58276600   -17.21195420
Ag  -18.62538300    -2.58276600   -17.21195420
Ag   -5.47176600    -5.16553200   -17.21195420
Ag   -8.36076600    -5.16553200   -17.21195420
Ag  -11.24976600    -5.16553200   -17.21195420
Ag  -14.13876600    -5.16553200   -17.21195420
Ag  -17.02776600    -5.16553200   -17.21195420
Ag   -6.76314900    -7.74829800   -17.21195420
Ag   -9.65214900    -7.74829800   -17.21195420
Ag  -12.54114900    -7.74829800   -17.21195420
Ag  -15.43014900    -7.74829800   -17.21195420
Ag   -8.05453200   -10.33106400   -17.21195420
Ag  -10.94353200   -10.33106400   -17.21195420
Ag  -13.83253200   -10.33106400   -17.21195420
Ag   -9.34591500   -12.91383000   -17.21195420
```

```
Ag   -12.23491500   -12.91383000   -17.21195420
Ag   -10.63729800   -15.49659600   -17.21195420
Ag    -2.58854400    -3.38590800   -19.66471520
Ag    -5.47754400    -3.38590800   -19.66471520
Ag    -8.36654400    -3.38590800   -19.66471520
Ag   -11.25554400    -3.38590800   -19.66471520
Ag   -14.14454400    -3.38590800   -19.66471520
Ag   -17.03354400    -3.38590800   -19.66471520
Ag    -3.87992700    -5.96867400   -19.66471520
Ag    -6.76892700    -5.96867400   -19.66471520
Ag    -9.65792700    -5.96867400   -19.66471520
Ag   -12.54692700    -5.96867400   -19.66471520
Ag   -15.43592700    -5.96867400   -19.66471520
Ag    -5.17131000    -8.55144000   -19.66471520
Ag    -8.06031000    -8.55144000   -19.66471520
Ag   -10.94931000    -8.55144000   -19.66471520
Ag   -13.83831000    -8.55144000   -19.66471520
Ag    -6.46269300   -11.13420600   -19.66471520
Ag    -9.35169300   -11.13420600   -19.66471520
Ag   -12.24069300   -11.13420600   -19.66471520
Ag    -7.75407600   -13.71697200   -19.66471520
Ag   -10.64307600   -13.71697200   -19.66471520
Ag    -9.04545900   -16.29973800   -19.66471520
Ag    -3.88570500    -4.18905000   -22.11747620
Ag    -6.77470500    -4.18905000   -22.11747620
Ag    -9.66370500    -4.18905000   -22.11747620
Ag   -12.55270500    -4.18905000   -22.11747620
Ag   -15.44170500    -4.18905000   -22.11747620
Ag    -5.17708800    -6.77181600   -22.11747620
Ag    -8.06608800    -6.77181600   -22.11747620
Ag   -10.95508800    -6.77181600   -22.11747620
Ag   -13.84408800    -6.77181600   -22.11747620
Ag    -6.46847100    -9.35458200   -22.11747620
Ag    -9.35747100    -9.35458200   -22.11747620
Ag   -12.24647100    -9.35458200   -22.11747620
Ag    -7.75985400   -11.93734800   -22.11747620
Ag   -10.64885400   -11.93734800   -22.11747620
Ag    -9.05123700   -14.52011400   -22.11747620
Ag    -5.18286600    -4.99219200   -24.57023720
Ag    -8.07186600    -4.99219200   -24.57023720
Ag   -10.96086600    -4.99219200   -24.57023720
Ag   -13.84986600    -4.99219200   -24.57023720
Ag    -6.47424900    -7.57495800   -24.57023720
Ag    -9.36324900    -7.57495800   -24.57023720
Ag   -12.25224900    -7.57495800   -24.57023720
Ag    -7.76563200   -10.15772400   -24.57023720
Ag   -10.65463200   -10.15772400   -24.57023720
Ag    -9.05701500   -12.74049000   -24.57023720
Ag    -6.48002700    -5.79533400   -27.02299820
Ag    -9.36902700    -5.79533400   -27.02299820
Ag   -12.25802700    -5.79533400   -27.02299820
Ag    -7.77141000    -8.37810000   -27.02299820
Ag   -10.66041000    -8.37810000   -27.02299820
Ag    -9.06279300   -10.96086600   -27.02299820
Ag    -7.77718800    -6.59847600   -29.47575920
Ag   -10.66618800    -6.59847600   -29.47575920
Ag    -9.06857100    -9.18124200   -29.47575920
Ag    -9.07434900    -7.40161800   -31.92852020
```

```
Ag   -1.30293900   -5.47465500  -21.18432920
Ag   -2.60010000   -6.27490800  -23.63709020
Ag   -3.89726100   -7.07516100  -26.08985120
Ag   -5.19442200   -7.87541400  -28.54261220
Ag   -6.49158300   -8.67566700  -30.99537320
Ag   -2.59432200   -8.05742100  -21.18432920
Ag   -3.89148300   -8.85767400  -23.63709020
Ag   -5.18864400   -9.65792700  -26.08985120
Ag   -6.48580500  -10.45818000  -28.54261220
Ag   -3.88570500  -10.64018700  -21.18432920
Ag   -5.18286600  -11.44044000  -23.63709020
Ag   -6.48002700  -12.24069300  -26.08985120
Ag   -5.17708800  -13.22295300  -21.18432920
Ag   -6.47424900  -14.02320600  -23.63709020
Ag   -6.46847100  -15.80571900  -21.18432920
Ag   -0.01733400   -7.56629100  -22.70394320
Ag   -1.31449500   -8.36654400  -25.15670420
Ag   -2.61165600   -9.16679700  -27.60946520
Ag   -3.90881700   -9.96705000  -30.06222620
Ag   -1.30871700  -10.14905700  -22.70394320
Ag   -2.60587800  -10.94931000  -25.15670420
Ag   -3.90303900  -11.74956300  -27.60946520
Ag   -2.60010000  -12.73182300  -22.70394320
Ag   -3.89726100  -13.53207600  -25.15670420
Ag   -3.89148300  -15.31458900  -22.70394320
Ag    1.26827100   -9.65792700  -24.22355720
Ag   -0.02889000  -10.45818000  -26.67631820
Ag   -1.32605100  -11.25843300  -29.12907920
Ag   -0.02311200  -12.24069300  -24.22355720
Ag   -1.32027300  -13.04094600  -26.67631820
Ag   -1.31449500  -14.82345900  -24.22355720
Ag    2.55387600  -11.74956300  -25.74317120
Ag    1.25671500  -12.54981600  -28.19593220
Ag    1.26249300  -14.33232900  -25.74317120
Ag    3.83948100  -13.84119900  -27.26278520
Ag   -4.18038300   -0.79811860  -14.75559780
Ag   -7.06938300   -0.79811860  -14.75559780
Ag   -9.95838300   -0.79811860  -14.75559780
Ag  -12.84738300   -0.79811860  -14.75559780
Ag  -15.73638300   -0.79811860  -14.75559780
Ag  -18.62538300   -0.79811860  -14.75559780
Ag   -5.47176600   -1.59623720  -12.29924130
Ag   -8.36076600   -1.59623720  -12.29924130
Ag  -11.24976600   -1.59623720  -12.29924130
Ag  -14.13876600   -1.59623720  -12.29924130
Ag  -17.02776600   -1.59623720  -12.29924130
Ag   -6.76314900   -2.39435580   -9.84288490
Ag   -9.65214900   -2.39435580   -9.84288490
Ag  -12.54114900   -2.39435580   -9.84288490
Ag  -15.43014900   -2.39435580   -9.84288490
Ag   -8.05453200   -3.19247430   -7.38652850
Ag  -10.94353200   -3.19247430   -7.38652850
Ag  -13.83253200   -3.19247430   -7.38652850
Ag   -9.34591500   -3.99059290   -4.93017200
Ag  -12.23491500   -3.99059290   -4.93017200
Ag  -10.63729800   -4.78871150   -2.47381560
Ag   -2.58854400   -3.37901740  -14.74970920
Ag   -5.47754400   -3.37901740  -14.74970920
```

```
Ag    -8.36654400    -3.37901740   -14.74970920
Ag   -11.25554400    -3.37901740   -14.74970920
Ag   -14.14454400    -3.37901740   -14.74970920
Ag   -17.03354400    -3.37901740   -14.74970920
Ag    -3.87992700    -4.17713600   -12.29335270
Ag    -6.76892700    -4.17713600   -12.29335270
Ag    -9.65792700    -4.17713600   -12.29335270
Ag   -12.54692700    -4.17713600   -12.29335270
Ag   -15.43592700    -4.17713600   -12.29335270
Ag    -5.17131000    -4.97525460    -9.83699630
Ag    -8.06031000    -4.97525460    -9.83699630
Ag   -10.94931000    -4.97525460    -9.83699630
Ag   -13.83831000    -4.97525460    -9.83699630
Ag    -6.46269300    -5.77337320    -7.38063990
Ag    -9.35169300    -5.77337320    -7.38063990
Ag   -12.24069300    -5.77337320    -7.38063990
Ag    -7.75407600    -6.57149180    -4.92428340
Ag   -10.64307600    -6.57149180    -4.92428340
Ag    -9.04545900    -7.36961040    -2.46792700
Ag    -3.88570500    -5.95991630   -14.74382060
Ag    -6.77470500    -5.95991630   -14.74382060
Ag    -9.66370500    -5.95991630   -14.74382060
Ag   -12.55270500    -5.95991630   -14.74382060
Ag   -15.44170500    -5.95991630   -14.74382060
Ag    -5.17708800    -6.75803490   -12.28746410
Ag    -8.06608800    -6.75803490   -12.28746410
Ag   -10.95508800    -6.75803490   -12.28746410
Ag   -13.84408800    -6.75803490   -12.28746410
Ag    -6.46847100    -7.55615350    -9.83110770
Ag    -9.35747100    -7.55615350    -9.83110770
Ag   -12.24647100    -7.55615350    -9.83110770
Ag    -7.75985400    -8.35427210    -7.37475130
Ag   -10.64885400    -8.35427210    -7.37475130
Ag    -9.05123700    -9.15239070    -4.91839480
Ag    -5.18286600    -8.54081520   -14.73793200
Ag    -8.07186600    -8.54081520   -14.73793200
Ag   -10.96086600    -8.54081520   -14.73793200
Ag   -13.84986600    -8.54081520   -14.73793200
Ag    -6.47424900    -9.33893370   -12.28157550
Ag    -9.36324900    -9.33893370   -12.28157550
Ag   -12.25224900    -9.33893370   -12.28157550
Ag    -7.76563200   -10.13705230    -9.82521910
Ag   -10.65463200   -10.13705230    -9.82521910
Ag    -9.05701500   -10.93517090    -7.36886270
Ag    -6.48002700   -11.12171400   -14.73204340
Ag    -9.36902700   -11.12171400   -14.73204340
Ag   -12.25802700   -11.12171400   -14.73204340
Ag    -7.77141000   -11.91983260   -12.27568690
Ag   -10.66041000   -11.91983260   -12.27568690
Ag    -9.06279300   -12.71795120    -9.81933050
Ag    -7.77718800   -13.70261290   -14.72615480
Ag   -10.66618800   -13.70261290   -14.72615480
Ag    -9.06857100   -14.50073150   -12.26979830
Ag    -9.07434900   -16.28351170   -14.72026620
Ag    -1.30293900    -5.46971460   -13.23277930
Ag    -2.60010000    -8.04972070   -13.22963830
Ag    -3.89726100   -10.62972680   -13.22649730
Ag    -5.19442200   -13.20973290   -13.22335630
```

```
Ag    -6.49158300   -15.78973900   -13.22021530
Ag    -2.59432200    -6.26783310   -10.77642280
Ag    -3.89148300    -8.84783930   -10.77328180
Ag    -5.18864400   -11.42784540   -10.77014080
Ag    -6.48580500   -14.00785150   -10.76699980
Ag    -3.88570500    -7.06595170    -8.32006640
Ag    -5.18286600    -9.64595780    -8.31692540
Ag    -6.48002700   -12.22596400    -8.31378440
Ag    -5.17708800    -7.86407030    -5.86371000
Ag    -6.47424900   -10.44407640    -5.86056900
Ag    -6.46847100    -8.66218890    -3.40735350
Ag    -0.01733400    -7.56130440   -11.71310180
Ag    -1.31449500   -10.14131050   -11.70996080
Ag    -2.61165600   -12.72131660   -11.70681980
Ag    -3.90881700   -15.30132280   -11.70367880
Ag    -1.30871700    -8.35942300    -9.25674530
Ag    -2.60587800   -10.93942910    -9.25360430
Ag    -3.90303900   -13.51943520    -9.25046330
Ag    -2.60010000    -9.15754160    -6.80038890
Ag    -3.89726100   -11.73754770    -6.79724790
Ag    -3.89148300    -9.95566020    -4.34403250
Ag     1.26827100    -9.65289430   -10.19342430
Ag    -0.02889000   -12.23290040   -10.19028330
Ag    -1.32605100   -14.81290650   -10.18714230
Ag    -0.02311200   -10.45101290    -7.73706780
Ag    -1.32027300   -13.03101900    -7.73392680
Ag    -1.31449500   -11.24913150    -5.28071140
Ag     2.55387600   -11.74448420    -8.67374680
Ag     1.25671500   -14.32449030    -8.67060580
Ag     1.26249300   -12.54260270    -6.21739030
Ag     3.83948100   -13.83607400    -7.15406930
Ag    -2.58854400     1.29756040   -13.23744210
Ag    -5.47754400     1.29756040   -13.23744210
Ag    -8.36654400     1.29756040   -13.23744210
Ag   -11.25554400     1.29756040   -13.23744210
Ag   -14.14454400     1.29756040   -13.23744210
Ag   -17.03354400     1.29756040   -13.23744210
Ag    -3.87992700     3.38706200   -11.71933030
Ag    -6.76892700     3.38706200   -11.71933030
Ag    -9.65792700     3.38706200   -11.71933030
Ag   -12.54692700     3.38706200   -11.71933030
Ag   -15.43592700     3.38706200   -11.71933030
Ag    -5.17131000     5.47656350   -10.20121850
Ag    -8.06031000     5.47656350   -10.20121850
Ag   -10.94931000     5.47656350   -10.20121850
Ag   -13.83831000     5.47656350   -10.20121850
Ag    -6.46269300     7.56606510    -8.68310680
Ag    -9.35169300     7.56606510    -8.68310680
Ag   -12.24069300     7.56606510    -8.68310680
Ag    -7.75407600     9.65556670    -7.16499500
Ag   -10.64307600     9.65556670    -7.16499500
Ag    -9.04545900    11.74506830    -5.64688330
Ag    -3.88570500     0.50561920   -10.78104170
Ag    -6.77470500     0.50561920   -10.78104170
Ag    -9.66370500     0.50561920   -10.78104170
Ag   -12.55270500     0.50561920   -10.78104170
Ag   -15.44170500     0.50561920   -10.78104170
Ag    -5.17708800     2.59512070    -9.26293000
```

```
Ag    -8.06608800     2.59512070    -9.26293000
Ag   -10.95508800     2.59512070    -9.26293000
Ag   -13.84408800     2.59512070    -9.26293000
Ag    -6.46847100     4.68462230    -7.74481820
Ag    -9.35747100     4.68462230    -7.74481820
Ag   -12.24647100     4.68462230    -7.74481820
Ag    -7.75985400     6.77412390    -6.22670640
Ag   -10.64885400     6.77412390    -6.22670640
Ag    -9.05123700     8.86362550    -4.70859470
Ag    -5.18286600    -0.28632210    -8.32464140
Ag    -8.07186600    -0.28632210    -8.32464140
Ag   -10.96086600    -0.28632210    -8.32464140
Ag   -13.84986600    -0.28632210    -8.32464140
Ag    -6.47424900     1.80317950    -6.80652960
Ag    -9.36324900     1.80317950    -6.80652960
Ag   -12.25224900     1.80317950    -6.80652960
Ag    -7.76563200     3.89268110    -5.28841780
Ag   -10.65463200     3.89268110    -5.28841780
Ag    -9.05701500     5.98218270    -3.77030610
Ag    -6.48002700    -1.07826330    -5.86824100
Ag    -9.36902700    -1.07826330    -5.86824100
Ag   -12.25802700    -1.07826330    -5.86824100
Ag    -7.77141000     1.01123830    -4.35012920
Ag   -10.66041000     1.01123830    -4.35012920
Ag    -9.06279300     3.10073990    -2.83201750
Ag    -7.77718800    -1.87020450    -3.41184070
Ag   -10.66618800    -1.87020450    -3.41184070
Ag    -9.06857100     0.21929710    -1.89372890
Ag    -9.07434900    -2.66214570    -0.95544030
Ag    -1.30293900     2.09418550   -10.78031380
Ag    -2.60010000     1.29990700    -8.32561160
Ag    -3.89726100     0.50562860    -5.87090940
Ag    -5.19442200    -0.28864990    -3.41620710
Ag    -6.49158300    -1.08292840    -0.96150490
Ag    -2.59432200     4.18368710    -9.26220210
Ag    -3.89148300     3.38940860    -6.80749980
Ag    -5.18864400     2.59513010    -4.35279760
Ag    -6.48580500     1.80085170    -1.89809540
Ag    -3.88570500     6.27318870    -7.74409030
Ag    -5.18286600     5.47891020    -5.28938810
Ag    -6.48002700     4.68463170    -2.83468580
Ag    -5.17708800     8.36269030    -6.22597860
Ag    -6.47424900     7.56841180    -3.77127630
Ag    -6.46847100    10.45219180    -4.70786680
Ag    -0.01733400     2.89314790    -8.32148750
Ag    -1.31449500     2.09886940    -5.86678530
Ag    -2.61165600     1.30459090    -3.41208300
Ag    -3.90881700     0.51031250    -0.95738080
Ag    -1.30871700     4.98264940    -6.80337570
Ag    -2.60587800     4.18837100    -4.34867350
Ag    -3.90303900     3.39409250    -1.89397120
Ag    -2.60010000     7.07215100    -5.28526400
Ag    -3.89726100     6.27787260    -2.83056170
Ag    -3.89148300     9.16165260    -3.76715220
Ag     1.26827100     3.69211020    -5.86266120
Ag    -0.02889000     2.89783180    -3.40795890
Ag    -1.32605100     2.10355330    -0.95325670
Ag    -0.02311200     5.78161180    -4.34454940
```

```
Ag   -1.32027300    4.98733340   -1.88984710
Ag   -1.31449500    7.87111340   -2.82643760
Ag    2.55387600    4.49107260   -3.40383480
Ag    1.25671500    3.69679410   -0.94913260
Ag    1.26249300    6.58057420   -1.88572300
Ag    3.83948100    5.29003500   -0.94500850
SUBEND
END

eor
```

14.3 Structure and Reactivity

14.3.1 Geometry Optimizations

Example: Geometry Optimization: H2O

Download GO_H2O.run

```
#! /bin/sh

# Summary:
# - geometry optimization in delocalized coordinates
# - geometry optimization in internal coordinates
# - start-up Hessian defined in the input file
# - geometry optimization in Cartesian coordinates


# Geometry optimization of the water molecule, using the (default) local density
# functional approximation (LDA)

# Fair quality basis set: triple zeta with polarization. Four equivalent
# computations are carried out. The first optimization is done in delocalized
# coordinates, which requires that atomic coordinates in the input are
# Cartesian. In the three other optimizations the atomic coordinates are input
# in Z-matrix format. The optimization is carried out by optimization of the
# internal coordinates in the second and third calculations, and by optimizing
# the Cartesian coordinates in the fourth one. In calculation #3 the start-up
# Hessian is defined in the input file; in #1,2, and 4 the default start-up
# Hessian (from a force-field approximation) is applied.

# As expected all final results are the same, within the range that might be
# expected from the convergence thresholds (here: the default values).


# == Optimization in delocalized coordinates ==

# A title is supplied. This title is printed in the output header. It is also
# written to any result files from the calculation and will be printed out when
# such a file is attached to another calculation, for instance as a fragment
# file. In addition, adf constructs a 'jobidentification' string that contains
# the adf release number and the date and time. The jobidentification is also
# printed in the output header and dumped on any result files.

# The atomic positions are given with the key atoms. The Cartesian atomic
```

```
# coordinates are in Angstrom. The structure used here does not necessary imply
# that the two HO bonds must remain equal in the optimization. The symmetry will
# keep them equal.

# The key geometry must be supplied to let the program do an optimization:
# otherwise a single point calculation would be carried out. The geometry data
# block is empty here, meaning that no default values are reset for the options
# that are controlled with this key.

# No symmetry is specified by a Schonfliess type symbol (key symmetry). The
# program will use the true symmetry of the nuclear frame (accounting for any
# fields, if present). In this case that is C(2v). If such symmetry would not be
# acceptable for adf (not all point groups are supported!) or when you want to
# run in a lower symmetry, the symmetry to be used must be specified.

# The fragment files are defined implicitly with the Basis keyword. In this case
# (as well as in most other samples) the fragment files reside in the local
# directory since they were created there in the same job. If they would have
# been located elsewhere you could specify a full path for each of the files, or
# alternatively (if all fragmentfiles are in one single directory) write the
# directory after the keyword fragments (on the same line).


$ADFBIN/adf <<eor
Title WATER Geometry Optimization with Delocalized Coordinates

Atoms
    O              0.000000      0.000000      0.000000
    H              0.000000     -0.689440     -0.578509
    H              0.000000      0.689440     -0.578509
End

Basis
 Type TZP
 Core Small
End

Geometry
  Converge grad=0.0000001
End

eor

rm TAPE21 logfile


# == Optimization in Cartesian coordinates ==


$ADFBIN/adf <<eor
Title WATER Geometry Optimization in Cartesians

Atoms
    O              0.000000      0.000000      0.000000
    H              0.000000     -0.689440     -0.578509
    H              0.000000      0.689440     -0.578509
End
```

```
Basis
 Type TZP
 Core Small
End

Geometry
  Optim Cartesian
  Converge grad=0.0000001
End

eor

rm TAPE21 logfile


# == Optimization with Internal Coordinates ==


$ADFBIN/adf <<eor
Title WATER Geometry Optimization with Internal Coordinates

Atoms    Z-Matrix
 1. O   0 0 0
 2. H   1 0 0   0.9
 3. H   1 2 0   0.9  100
End

Basis
 Type TZP
 Core Small
End

Geometry
  Converge grad=0.0000001
End

eor

rm TAPE21 logfile


# == Definition of (diagonal) start-up Hessian ==

# The key HessDiag defines the start-up Hessian to be diagonal with values 1.0
# and 0.1 for the entries related to bondlengths and angles respectively.


$ADFBIN/adf <<eor
Title WATER   optimization with (partial) specification of Hessian

Atoms    Z-Matrix
 1. O   0 0 0
 2. H   1 0 0   0.9
 3. H   1 2 0   0.9  100
End

HessDiag  rad=1.0  ang=0.1
```

```
Fragments
 H   t21.H
 O   t21.O
End

Geometry
  Converge grad=0.0000001
End

eor

rm TAPE21 logfile


# In the last calculation the atomic coordinates are input in the same way as
# before, but the geometry block now specifies, with the subkey optim, that the
# cartesian coordinates are to be varied and monitored for convergence.

# If different coordinates are specified in the optim instruction than were used
# for the input in the atoms block, no constraints can be used.

# The placement of 0.9 (and 100) in the define block has a completely
# different meaning. define merely associates a numerical value with an
# identifier. Wherever the identifier occurs in input (not only in the atoms
# block) it will be replaced by the numerical value. This means that there are
# now nine (9) variables: the x,y,z coordinates of the three atoms.

# Pure translations and rotations will be filtered out by the program and the
# symmetry (explicitly specified or internally computed), C(2v) here, will be
# enforced on all developments so that the situation is equivalent to the
# previous calculation as regards the degrees of freedom of the system.

# Remark: the define block must occur in input before the variables defined in
# it are used. This is one of the few cases where the relative position of keys
# in the input stream is relevant.


$ADFBIN/adf <<eor
Title WATER Geometry Optimization in Cartesians

Geometry
  Optim Cartesian
  Converge grad=0.0000001
End

Atoms    Z-Matrix
 1. O   0 0 0
 2. H   1 0 0   0.9
 3. H   1 2 0   0.9 100
End

Fragments
 H   t21.H
 O   t21.O
End

eor
```

```
mv TAPE21 H2O.t21
```

Example: Restraint Geometry Optimization: H2O

Download GO_restraint.run

```
#! /bin/sh

# The restraint does not have to be satisfied at the start of the geometry
# optimization. An extra force is added to restrain the bond length, angle, or
# dihedral angle to a certain value.


#  == angle restraint ==


$ADFBIN/adf <<eor
title WATER geometry optimization with angle restraint

ATOMS
    1.O        0.001356    0.000999    0.000000
    2.H        0.994442   -0.037855    0.000000
    3.H       -0.298554    0.948531    0.000000
END

BASIS
 Type DZP
END

RESTRAINT
  ANGLE  3 1 2 125.0
END

GEOMETRY
END

eor

rm TAPE21 logfile


# == bond length restraint ==


$ADFBIN/adf <<eor
title WATER Geometry Optimization with bond length restraint

ATOMS
    1.O        0.001356    0.000999    0.000000
    2.H        0.994442   -0.037855    0.000000
    3.H       -0.298554    0.948531    0.000000
END

BASIS
 Type DZP
END
```

```
RESTRAINT
  DIST  1 2 1.03
  DIST  1 3 1.03
END

GEOMETRY
END

eor

rm TAPE21 logfile


# == Restraining dihedral ==


$ADFBIN/adf <<eor
Title Restraining dihedral of ethane

SYMMETRY NOSYM

ATOMS
  1.C        -0.004115   -0.000021    0.000023
  2.C         1.535711    0.000022    0.000008
  3.H        -0.399693    1.027812   -0.000082
  4.H        -0.399745   -0.513934    0.890139
  5.H        -0.399612   -0.513952   -0.890156
  6.H         1.931188    0.514066    0.890140
  7.H         1.931432    0.513819   -0.890121
  8.H         1.931281   -1.027824    0.000244
END

RESTRAINT
  DIHED  6 2 1 3   20.00
END

BASIS
  type DZP
END

GEOMETRY
END

eor

mv TAPE21 C2H6.t21
```

Example: Constraint Geometry Optimization: H2O

```
Download GO_constraints.run
```

```
#! /bin/sh

# The key CONSTRAINTS can only be used to enforce constraint in a geometry
# optimization. The input for this key is very similar to that of
# the RESTRAINT keyword. The key CONSTRAINTS can, however, also be used to
# constrain Cartesian coordinates.
```

```
# The constraints do not have to be satisfied at the start of the geometry
# optimization.


# == Example for angle constraint ==


$ADFBIN/adf <<eor
title WATER geometry optimization with angle constraint

ATOMS
    1.O         0.001356    0.000999    0.000000
    2.H         0.994442   -0.037855    0.000000
    3.H        -0.298554    0.948531    0.000000
END

BASIS
 Type DZP
END

CONSTRAINTS
  ANGLE  3 1 2 125.0
END

GEOMETRY
END

eor

rm TAPE21 logfile


# == Example for fixed-atom constraint ==

# Note that the optimization should be done in Cartesian:

$ADFBIN/adf <<eor
title WATER geometry optimization with fixed-atom constraint

ATOMS
    1.O         0.001356    0.000999    0.000000
    2.H         0.994442   -0.037855    0.000000
    3.H        -0.298554    0.948531    0.000000
END

BASIS
 Type DZP
END

SYMMETRY NOSYM

CONSTRAINTS
  ATOM 1 0.0 0.0 0.0
  ATOM 2 1.0 0.0 0.0
END

GEOMETRY
```

```
  OPTIM CARTESIAN
END

eor

rm TAPE21 logfile


# == Example for bond length constraint ==


$ADFBIN/adf <<eor
title WATER Geometry Optimization with bond length constraint

ATOMS
    1.O         0.001356    0.000999    0.000000
    2.H         0.994442   -0.037855    0.000000
    3.H        -0.298554    0.948531    0.000000
END

BASIS
 Type DZP
END

CONSTRAINTS
  DIST  1 2 1.03
  DIST  1 3 1.03
END

GEOMETRY
  OPTIM CARTESIAN
END

eor

rm TAPE21 logfile


# == Example for dihedral angle constraint ==


$ADFBIN/adf <<eor
Title Constraining dihedral of ethane

SYMMETRY NOSYM

ATOMS
  1.C        -0.004115   -0.000021    0.000023
  2.C         1.535711    0.000022    0.000008
  3.H        -0.399693    1.027812   -0.000082
  4.H        -0.399745   -0.513934    0.890139
  5.H        -0.399612   -0.513952   -0.890156
  6.H         1.931188    0.514066    0.890140
  7.H         1.931432    0.513819   -0.890121
  8.H         1.931281   -1.027824    0.000244
END

CONSTRAINTS
```

```
  DIHED  6 2 1 3  20.00
END

BASIS
  type DZP
END

GEOMETRY
END

eor

rm TAPE21 logfile


# == Example for Block constraint (with a dihedral constraint) ==


$ADFBIN/adf <<eor
Title Block constraints (with a dihedral constraint)

SYMMETRY NOSYM

ATOMS
  1.C       -0.004115   -0.000021    0.000023 b=b1
  2.C        1.535711    0.000022    0.000008 b=b2
  3.H       -0.399693    1.027812   -0.000082 b=b1
  4.H       -0.399745   -0.513934    0.890139 b=b1
  5.H       -0.399612   -0.513952   -0.890156 b=b1
  6.H        1.931188    0.514066    0.890140 b=b2
  7.H        1.931432    0.513819   -0.890121 b=b2
  8.H        1.931281   -1.027824    0.000244 b=b2
END

CONSTRAINTS
  DIHED  6 2 1 3  20.00
  BLOCK b1
  BLOCK b2
END

BASIS
  type DZP
END

GEOMETRY
END

eor

rm TAPE21 logfile
```

Example: Initial Hessian

```
Download InitHess.run
```

```
#! /bin/sh
```

```
# In this example the Hessian from a DFTB and a MOPAC calculation is used as a
# starting guess for the Hessian in an ADF geometry optimization of Isobutane.
# This example consist of 4 calculations:
# - MOPAC frequency calculation
# - DFTB frequency calculation
# - ADF geometry optimization with initial MOPAC Hessian
# - ADF geometry optimization with initial DFTB Hessian



# =======================
# Generate MOPAC Hessian
# =======================


AMS_JOBNAME=mopac_hessian $ADFBIN/ams << eor

Task SinglePoint

Properties
    NormalModes Yes
End

System
    Atoms
        C      -3.01417223     -2.81892970      0.27443080
        C      -2.32560254     -1.48669854     -0.04099291
        C      -0.81033566     -1.60274167      0.15652441
        H      -2.65505660     -3.61825489     -0.40916512
        H      -4.11459952     -2.71726364      0.16084942
        H      -2.80179295     -3.12071121      1.32226630
        H      -2.71211792     -0.71788031      0.66394035
        C      -2.65045192     -1.03659727     -1.46944155
        H      -0.32255475     -0.62449394     -0.04145480
        H      -0.38172118     -2.36329425     -0.53112669
        H      -0.58228356     -1.89634208      1.20336328
        H      -2.17532684     -0.05436369     -1.67852390
        H      -3.74843413     -0.92249496     -1.59450083
        H      -2.28021380     -1.77910197     -2.20888365
    End
End

Engine Mopac
endengine
eor



mv mopac_hessian.results/mopac.rkf mopac.rkf

# =======================
# Generate a dftb Hessian
# =======================


$ADFBIN/ams <<eor

Task SinglePoint

Properties
```

```
   NormalModes True
End

System
    Atoms
     C       -3.014172230000      -2.818929700000       0.274430800000
     C       -2.325602540000      -1.486698540000      -0.040992910000
     C       -0.810335660000      -1.602741670000       0.156524410000
     H       -2.655056600000      -3.618254890000      -0.409165120000
     H       -4.114599520000      -2.717263640000       0.160849420000
     H       -2.801792950000      -3.120711210000       1.322266300000
     H       -2.712117920000      -0.717880310000       0.663940350000
     C       -2.650451920000      -1.036597270000      -1.469441550000
     H       -0.322554750000      -0.624493940000      -0.041454800000
     H       -0.381721180000      -2.363294250000      -0.531126690000
     H       -0.582283560000      -1.896342080000       1.203363280000
     H       -2.175326840000      -0.054363690000      -1.678523900000
     H       -3.748434130000      -0.922494960000      -1.594500830000
     H       -2.280213800000      -1.779101970000      -2.208883650000
    End

    Charge 0
End

Engine DFTB
    ResourcesDir DFTB.org/mio-1-1
    SCC
    End
EndEngine

eor

mv ams.results/dftb.rkf dftb.rkf

# Use the MOPAC Hessian generated earlier for ADF optimization

"$ADFBIN/adf" <<eor
ATOMS
 C       -3.014172230000      -2.818929700000       0.274430800000
 C       -2.325602540000      -1.486698540000      -0.040992910000
 C       -0.810335660000      -1.602741670000       0.156524410000
 H       -2.655056600000      -3.618254890000      -0.409165120000
 H       -4.114599520000      -2.717263640000       0.160849420000
 H       -2.801792950000      -3.120711210000       1.322266300000
 H       -2.712117920000      -0.717880310000       0.663940350000
 C       -2.650451920000      -1.036597270000      -1.469441550000
 H       -0.322554750000      -0.624493940000      -0.041454800000
 H       -0.381721180000      -2.363294250000      -0.531126690000
 H       -0.582283560000      -1.896342080000       1.203363280000
 H       -2.175326840000      -0.054363690000      -1.678523900000
 H       -3.748434130000      -0.922494960000      -1.594500830000
 H       -2.280213800000      -1.779101970000      -2.208883650000
END

BASIS
type DZ
core Large
createoutput None
```

```
END

GEOMETRY
    InitHess mopac.rkf
END

NOPRINT LOGFILE

eor

# Use the DFTB Hessian generated earlier for ADF optimization

"$ADFBIN/adf" <<eor
ATOMS
 C      -3.014172230000      -2.818929700000       0.274430800000
 C      -2.325602540000      -1.486698540000      -0.040992910000
 C      -0.810335660000      -1.602741670000       0.156524410000
 H      -2.655056600000      -3.618254890000      -0.409165120000
 H      -4.114599520000      -2.717263640000       0.160849420000
 H      -2.801792950000      -3.120711210000       1.322266300000
 H      -2.712117920000      -0.717880310000       0.663940350000
 C      -2.650451920000      -1.036597270000      -1.469441550000
 H      -0.322554750000      -0.624493940000      -0.041454800000
 H      -0.381721180000      -2.363294250000      -0.531126690000
 H      -0.582283560000      -1.896342080000       1.203363280000
 H      -2.175326840000      -0.054363690000      -1.678523900000
 H      -3.748434130000      -0.922494960000      -1.594500830000
 H      -2.280213800000      -1.779101970000      -2.208883650000
END

BASIS
type DZ
core Large
createoutput None
END

GEOMETRY
    InitHess dftb.rkf
END

eor
```

Example: Geometry optimization with an external electric field or point charges: LiF

```
Download GO_LiF_Efield.run
```

```
#! /bin/sh

# In the first example a geometry optimization is performed with an external
# homogeneous electric field. In the second example a geometry optimization is
# performed with an external point charges

# Note that SYMMETRY NOSYM should be used. In case of point charges it is
# important to use the QPNEAR subkeyword of the BECKEGRID key with a large
# enough value that would include some of the point charges.
```

```
$ADFBIN/adf <<eor
Title LiF Geometry Optimization in Cartesians with new optimizer in the presence of␣
↪electric field

Symmetry NOSYM

Atoms
    F    0.000000     0.800000     0.000000
    Li   0.000000    -0.800000     0.000000
End

Basis
 Type TZP
 Core Small
End

Geometry
  Optim Cartesian
  Converge grad=0.0000004
  Iterations 100
End

Efield 0.0 0.0 0.01

eor

mv TAPE21 LiF.t21
rm logfile

$ADFBIN/adf <<eor
Title LiF Geometry Optimization in Cartesians with new optimizer in the presence of␣
↪point charges

Symmetry NOSYM

Atoms
    F    0.000000     0.800000     0.000000
    Li   0.000000    -0.800000     0.000000
End

Basis
 Type TZP
 Core Small
End

Geometry
  Optim Cartesian
  Converge grad=0.0001
  Iterations 100
End

PointCharges
0.0 0.0  5.3  0.5
0.0 0.0 -5.3 -0.5
End

! IT IS IMPORTANT to use the qpnear keyword with a large enough
! value that would include some of the point charges
```

```
BeckeGrid
   qpnear 20
end

eor

mv TAPE21 LiF_pc.t21
rm logfile
```

Example: Excited state geometry optimization with a constraint: CH2O

Download EGO_CH2O_trip_constr.run

```
#!/bin/sh

# Example for an excited triplet state geometry optimization with a constraint
# included.

# Needed for such excited state optimizations are the key EXCITATIONS (to
# calculate excitation energies), the key GEOMETRY (to do a geometry
# optimization) and the key EXCITEDGO (to select for which excitation a geometry
# optimization should be performed). In this case a Z-matrix input for the
# coordinates is used, and a constraint is used using the GEOVAR keyword. Note
# that one has to select the state with the irrep name A'', and not with AAA.


$ADFBIN/adf <<eor
atoms zmatrix
    C         0 0 0       0.0      0.0      0.0
    O         1 0 0       R        0.0      0.0
    H         1 2 0       1.0    110.0      0.0
    H         1 2 3       1.0    110.0    170.0
end

XC
END

GEOVAR
 R=1.2 F
END

GEOMETRY
END

BASIS
TYPE DZ
CORE NONE
end

excitations
davidson
End
LOWEST 5
onlytrip
end
```

```
EXCITEDGO
 STATE A'' 1
 TRIPLET
 OUTPUT 2
end

SYMMETRY C(S)

eor
```

14.3.2 Transition States, Linear Transits, Intrinsic Reaction Coordinates

Example: LT, Frequencies, TS, and IRC: HCN

Download HCN.run

```
#! /bin/sh


# Summary
#
# - For a sequence of intermediates, each defined by a fixed angle H-C-N between
#   the linear extremes HCN and CNH, the remaining geometrical parameters are
#   optimized, giving a Linear Transit point-by-point scan of the energy curve
#   of the Hydrogen atom traveling from one end of the CN fragment to the other.
#   This is a useful way to get a reasonable first guess of the Transition
#   State.
# - At the approximate TS a Frequencies calculation is performed to obtain a
#   fairly accurate Hessian for the next calculation.
# - A TS search is carried out, using the computed Hessian. As variation, the TS
#   search is repeated, first with the automatic (internal) Hessian (based on
#   force fields) and then also with a constraint applied.
# - A full IRC scan of the full path, starting from the TS, down to the two
#   minima.


# == Linear Transit (LT) ==

# The first calculation is a Linear Transit where the Hydrogen atom moves from
# one side of CN to the other by a parametrized step-by-step change of the angle
# H-C-N. The other coordinates of the system are optimized along the path.

# In the atoms block, one coordinate value is represented by an identifier (th).
# In the geovar block this is assigned two values, implying that it is a Linear
# Transit parameter. The initial and final values for the parameter are given.

# Since the geometry block does not have OPTIM SELECTED, all other coordinates
# are optimized for each of the 10 Linear Transit points.

# The subkey iterations in the geometry block carries two arguments: the first
# is the maximum number of optimization steps (per LT point). The second is the
# number of LT points to compute in this run: 4. This implies that only a part
# of the 10-point path defined by the LT parameter(s) will be scanned. The
# remainder will be done in a follow-up run to illustrate usage of the restart
# facility.
```

```
$ADFBIN/adf <<eor
Title    HCN Linear Transit, first part
NoPrint  SFO, Frag, Functions, Computation

Atoms      Internal
  1 C  0 0 0       0    0    0
  2 N  1 0 0      1.3  0    0
  3 H  1 2 0      1.0  th   0
End

Basis
 Type DZP
End

Symmetry NOSYM

BeckeGrid
 Quality good
End

Geometry
  Branch Old
  LinearTransit  10
  Iterations    30  4
  Converge  Grad=3e-2  Rad=3e-2  Angle=2
END

Geovar
  th   180    0
End

eor

mv TAPE21 t21.LT
rm logfile



# The NoPRINT key turns off a lot of default output. There are several PRINT and
# NOPRINT options; see the User's Guides for details.

# Since the geometry changes from linear to planar (and finally back to linear
# again), the symmetry must be given explicitly in the input file. Otherwise the
# program would find a C(lin) symmetry for the initial geometry and assume that
# this symmetry is preserved throughout. This would of course result in an error
# abort when the first LT step is carried out, breaking the linear symmetry.

# The here specified symmetry (NOSYM: no symmetry at all) is not the true
# symmetry of the complete path C(s) but a subgroup. It is always allowed to
# specify a lower symmetry than the actually present symmetry. Such may be
# necessary (for instance when the true symmetry cannot be handled by adf) or in
# special cases required for reasons of analysis. Generally speaking, however,
# we recommend to use the highest symmetry possible (given the case at hand and
# taking into account the symmetries recognizable by ADF) to boost performance.

# Convergence thresholds in the geometry block are set less tight than the
# defaults: we need only a reasonable estimate of the path, but no highly
```

```
# converged geometries.


# At the end of the run the tape21 result file is saved and renamed t21.LT to
# serve as restart file for the follow-up calculation.


# == LT continuation ==



$ADFBIN/adf <<eor
Title    HCN Linear Transit
NoPrint   SFO,Frag,Functions,Computation

Restart
  File t21.LT
End

Fragments
  N    t21.N
  C    t21.C
  H    t21.H
End

Atoms       Internal
  1 C  0 0 0       0    0    0
  2 N  1 0 0      1.3   0    0
  3 H  1 2 0      1.0  th    0
End

symmetry NOSYM

BeckeGrid
 Quality good
End

Geometry
  Branch Old
  LinearTransit   10
  Converge        Grad=3e-2 Rad=3e-2 Angle=2
END

Geovar
  th   180    0
End

eor

rm logfile
mv TAPE21 HCN_LT.t21



# From the restart file, supplied with the key restart, the program reads off
# that the first 4 points of the LT path have been done already and the scan is
# continued with LT point #5. The same path definition is supplied again,
# including the original starting values for the coordinates. The actual
# starting coordinates (for LT point #5) are read from the restart file. The
# input values, however, serve to define and verify consistency of the defined
# LT path and must therefore be supplied correctly.
```

```
# The key noprint is used to suppress major parts of standard output: all
# information pertaining to the sfo analysis, all build-from-fragments
# information, and the lists of elementary functions in the basis sets and fit
# sets.



# == Frequencies at the estimated Transition State ==



# From the results of the Linear Transit run we can sketch the energy barrier
# that H passes over when going from one side of the molecule to the other. This
# yields a reasonable guess for the Transition State.

# To check that the so-obtained estimate is adequate we compute the frequencies
# in that geometry: one of them should be imaginary.

# Apart from serving as a check that the TS estimate is not too bad, the
# computed Hessian will also serve in the follow-up calculation to obtain the
# true TS.



$ADFBIN/adf <<eor
Title    HCN Frequencies in LT max (approx), moderate precision
NoPrint  SFO,Frag,Functions,Computation

BeckeGrid
 Quality good
End

Fragments
  N   t21.N
  C   t21.C
  H   t21.H
End

Atoms      Internal
  1 C  0 0 0      0       0   0
  2 N  1 0 0      1.186   0   0
  3 H  1 2 0      1.223  70   0
End

Geometry
  Frequencies
  End
End

eor

rm logfile
mv TAPE21 HCN_Freq.t21

# Inspection of the output file shows that one of the frequencies is imaginary,
# as expected (printed as negative), signaling the proximity of the Transition
# State.

# The TAPE21 result file of the calculation is renamed and saved. Later we will
# use it as a 'restart' file for a TS search, namely to supply the computed
```

```
# Hessian as the initial 'guess' of the Hessian in the (TS) optimization run.


# == TS search ==

# Now carry out the Transition State search, starting from the lt-derived guess.

# In this first attempt to find the TS, no use is made of the tape21 result file
# from the Frequencies run. That will be done in the next calculation.



$ADFBIN/adf <<eor
Title   HCN Transition State, automatic initial Hessian
NoPrint  SFO,Frag,Functions,Computation

BeckeGrid
 Quality good
End

Atoms     Internal
  1 C  0 0 0     0       0   0
  2 N  1 0 0     1.186   0   0
  3 H  1 2 0     1.223  70   0
End

Fragments
  N    t21.N
  C    t21.C
  H    t21.H
End

Geometry
  TransitionState
  End
  Converge Grad=0.0001
End

eor

rm TAPE21 logfile

# The TS-search run type is specified in the geometryblock.

# No symmetry is specified; the program determines the symmetry to be C(s) and
# consequently carries out the ts search in that symmetry.


#  == TS search, using the Hessian from the Frequencies run ==



$ADFBIN/adf <<eor
Title   HCN Transition State,  initial Hessian from Freq run
NoPrint  SFO,Frag,Functions,Computation

Restart
  File HCN_Freq.t21
End
```

```
Save     TAPE13

BeckeGrid
 Quality good
End

Atoms          Internal
  1 C  0 0 0      0       0   0
  2 N  1 0 0      1.186   0   0
  3 H  1 2 0      1.223  70   0
End

Fragments
  N    t21.N
  C    t21.C
  H    t21.H
End

Geometry
  TransitionState
  End
  Converge Grad=0.0001
End

eor

mv TAPE13  t13.TS
rm logfile
mv TAPE21 HCN_TS.t21


# The CheckPoint file TAPE13, at normal termination automatically deleted by the
# program, is here saved, using the SAVE key. TAPE13 is as good a restart file
# as TAPE21 is, but it is a lot smaller. TAPE21 contains a large amount of
# information for analysis purposes, while TAPE13 contains essentially only
# restart-type data.

# The input is identical to the previous one, except for the restart file. This
# is used here to provide the Hessian computed in the Frequencies run as the
# start-up Hessian for the ts optimization. At the same time the atomic
# coordinates are read off from the restart file and override the values in the
# input file. This latter aspect could have been suppressed; see the User's
# Guide for using the restart key.



#  == Constrained TS search ==

# Finally the ts search where one coordinate is kept frozen, to illustrate a
# constrained optimization.



$ADFBIN/adf <<eor
Title    HCN  constrained TS search
NoPrint  SFO,Frag,Functions,Computation

Restart
```

```
  File HCN_Freq.t21
End

BeckeGrid
 Quality good
End

Atoms     Internal
  1 C  0 0 0      0        0   0
  2 N  1 0 0      rNC      0   0
  3 H  1 2 0      1.223  70   0
End

GeoVar
  rNC=1.186 F
End

Fragments
  N    t21.N
  C    t21.C
  H    t21.H
End

Geometry
  TransitionState
  End
  Converge Grad=0.0001
End
FULLSCF

eor


echo ""
echo "Final Zmatrix and energy at TS"
$ADFBIN/adfreport -i TAPE21 'Geometry%zmatrix#12.6f##3' bonding
rm TAPE21 logfile

# The geovar key specifies that the nc distance, rNC has the initial value 1.15
# and remains frozen ('F').

# The fact that the optimization is now carried out in a different subspace of
# atomic coordinates does not prevent us from using the t21.Freq restart file to
# supply the initial Hessian.

$ADFBIN/adf <<eor
Title    HCN Frequencies at TS
NoPrint  SFO,Frag,Functions,Computation

BeckeGrid
 Quality good
End

Fragments
  N    t21.N
  C    t21.C
  H    t21.H
End
```

```
Atoms       Internal
   1 C        0     0     0     0.000000     0.000000     0.000000
   2 N        1     0     0     1.185055     0.000000     0.000000
   3 H        1     2     0     1.208984    69.196764     0.000000
End

AnalyticalFreq
End

eor

mv TAPE21 HCN_Freq_TS.t21


# == IRC scan of the reaction path ==


# The IRC calculation is split in three steps, to illustrate the Restart
# facility applied to the IRC functionality.

# In the first only a few points are computed, along one of the two paths
# leading from the TS to the adjacent minima. Since no explicit directives are
# given in the input to specify the direction of the first path, the so-called
# 'forward' path is taken. The definition of which is 'forward' and which is
# 'backward' is in fact quite arbitrary and is determined by the program. See
# the User's Guide for details.

# The saved TAPE13 file from one of the TS calculations is used as restart file.
# This provides (a) the optimized coordinates of the TS as starting point, (b)
# the initial Hessian to guide the point-by-point optimizations along the IRC
# path, and (c) the eigenvector of the lowest Hessian eigenvalue to define the
# initial direction of the IRC path.

# The TAPE13 file from this partial IRC scan is saved to serve as start-up file
# for the next calculations, which will continue the IRC scan.

# In the Geometry key block, the run type is set to IRC and the 'Points' option
# is used to limit the number of IRC points to compute.



$ADFBIN/adf  <<eor
Title   HCN  IRC partial path (forward)
NoPrint SFO,Frag,Functions, Computation

BeckeGrid
 Quality good
End

Restart
  File HCN_Freq_TS.t21
End
Save     TAPE13

Atoms          Internal
  1 C  0 0 0      0        0   0
  2 N  1 0 0    1.186      0   0
```

```
   3 H  1 2 0      1.223  70   0
End

Fragments
  N    t21.N
  C    t21.C
  H    t21.H
End

Geometry
  IRC  Points=5
  Converge Grad=0.001
End
SCF
  Converge 0.00000001
End
FULLSCF

eor

mv TAPE13  t13.IRC_1
rm TAPE21 logfile
echo ""
echo "Final Zmatrix after IRC"
$ADFBIN/adfreport -i t13.IRC_1 'Geometry%zmatrix#12.6f##3'



# The IRC is continued in the next calculation, using the TAPE13 file from the
# previous one as restart file. From this file, the program reads the IRC path
# information computed sofar. By default, it would continue on the 'forward'
# path, since that was not yet finished. However, in the Geometry key block, we
# now specify not only that a limited number of points is to be computed in this
# run (5 again), but we instruct the program also to compute only points on the
# 'backward' path.



$ADFBIN/adf  <<eor
Title   HCN  IRC partial part (backward)
NoPrint SFO,Frag,Functions, Computation

Restart
  File t13.IRC_1
End
Save    TAPE13

BeckeGrid
 Quality good
End

Atoms         Internal
  1 C  0 0 0      0       0   0
  2 N  1 0 0      1.186   0   0
  3 H  1 2 0      1.223  70   0
END

Fragments
```

```
   N    t21.N
   C    t21.C
   H    t21.H
End

Geometry
  IRC   Points=5  Forward=False
  Converge Grad=0.001
End
SCF
  Converge 0.00000001
End
FULLSCF

eor

mv TAPE13  t13.IRC_2
rm TAPE21 logfile
echo ""
echo "Final Zmatrix after IRC"
$ADFBIN/adfreport -i t13.IRC_2 'Geometry%zmatrix#12.6f##3'


# In the third IRC run, the IRC scan is finished. We start with the TAPE13 file
# from the previous run and set a maximum of 70 IRC points to compute (which
# turns out to be sufficient for the complete IRC scan). The program starts on
# the forward path, continuing where the first (not the previous) had stopped
# after 5 points, completes the forward path, and then continues on the backward
# path, starting where the second IRC run had stopped. Both paths are finished
# and a summary of the path characteristics is printed in the final part of the
# output.


$ADFBIN/adf  <<eor
Title    HCN  IRC completion
NoPrint  SFO,Frag,Functions, Computation

Restart
  File t13.IRC_2
End

BeckeGrid
 Quality good
End

Atoms          Internal
  1 C  0 0 0     0       0   0
  2 N  1 0 0     1.186   0   0
  3 H  1 2 0     1.223  70   0
End

Fragments
  N    t21.N
  C    t21.C
  H    t21.H
End

Geometry
```

```
  IRC   Points=100 StepMax=0.2 StepMin=0.15
  Converge Grad=0.0001
End
SCF
  Converge 0.00000001
End
FULLSCF


eor


mv TAPE21 HCN_irc.t21


echo ""
echo "Final Zmatrix and energy after IRC"
$ADFBIN/adfreport -i HCN_irc.t21 'Geometry%zmatrix#12.6f##3' bonding
```

Example: Transition state search with the CINEB method: HCN

Download HCN_CINEB.run

```
#! /bin/sh


# This example demonstrates the use of the Nudged Elastic Band method in ADF for
# finding a transition state of the HCN isomerisation reaction. A shell script
# used to run the example calculation is shown below:


# A few important points to note:


# Symmetry is set to Cs explicitly because all images must have the same
# symmetry but symmetry of the reaction products is higher, C_infty. Thus, it is
# necessary to lower the overall symmetry to match the lowest.


# In the GEOVAR section, there are six values specified for the coordinates of
# the hydrogen atom. This is necessary in order to bring the initial guess for
# the reaction path closer to the final result. Moreover, if only two values for
# YH were specified (0.0 and 0.0) then the hydrogen atom would be 'dragged'
# along the C-N bond leading to unrealistic geometries and, eventually, to a
# failure.


# In the GEOMETRY section of the input, the number of NEB images is set to 9;
# the convergence criterion is lowered to 0.001; optimization of only the
# coordinates specified in the GEOVAR block is requested;


# the NEB spring parameter is set to a constant (energy-independent) value of
# 0.06 a.u.. The choice of the parameter depends on the stiffness of the bonds
# involved in the reaction and the value of the parameter should, in principle,
# be of about the same magnitude as the Hessian eigenvalue for the coordinates
# participating in the reaction.



$ADFBIN/adf <<eor
TITLE Test of the CI-NEB method

SYMMETRY C(S)

ATOMS
    1.C        0.000000     0.000000     0.000000
```

```
    2.N          XN            0.000000    0.000000
    3.H          XH            YH          0.000000
END

GEOVAR
XN      1.180  1.163
XH      2.196  1.831 1.006 0.105 -0.718 -1.078
YH      0.000  0.799 1.122 1.163  0.813  0.000
END

BASIS
END

GEOMETRY
  CINEB      9
  OPTIM selected
  converge grad=0.0005
  iterations 100
  NEBOptEnds False
END

SCF
  Converge 0.00000001
END

eor

mv TAPE21 HCN_CINEB.t21

$ADFBIN/adf <<eor
TITLE Test of the CI-NEB method, optimize reaction initial and final states too

SYMMETRY C(S)

ATOMS
    1.C          0.000000    0.000000    0.000000
    2.N          XN            0.000000    0.000000
    3.H          XH            YH          0.000000
END

GEOVAR
XN      1.180  1.163
XH      2.196  1.831 1.006 0.105 -0.718 -1.078
YH      0.000  0.799 1.122 1.163  0.813  0.000
END

FRAGMENTS
  H t21.H
  C t21.C
  N t21.N
END

GEOMETRY
  CINEB      9
  OPTIM selected
  converge grad=0.0005
  iterations 100
END
```

```
SCF
  Converge 0.00000001
END

eor

mv TAPE21 HCN_CINEB_fullopt.t21
```

Example: TS search using partial Hessian: C2H6 internal rotation

Download TS_C2H6.run

```
#! /bin/sh

# Frequently when searching for a transition state, one needs an accurate second
# derivatives matrix, a Hessian. An exact Hessian may be obtained analytically
# but this may be very expensive for large molecules. In such cases it may be
# beneficial to calculate Hessian matrix elements only for atoms directly
# involved in the reaction for which a transition state is sought for. The rest
# of the Hessian can then be approximated using a cheaper method.

# In this example, a saddle point of the ethane internal rotation around C-C
# bond is found. In principle, only hydrogen atoms contribute to the normal mode
# we are interested in. Therefore we calculate a partial Hessian matrix
# including hydrogen atoms only. For this purpose, the AnalyticalFreq block key
# is used. In this block, a NUC keyword is added specifying that the second
# derivatives are calculated for atom 3 (and its symmetry-equivalents) only.
# Note that the Hessian matrix elements between symmetry-equivalent atoms, for
# example between 3,H and 4.H are also calculated. The rest of the matrix is
# estimated using the default method.

"$ADFBIN/adf" <<eor
TITLE Ethane transition state search using partial Hessian

ATOMS
1 C      0.000000000000      0.000000000000      0.767685465031
2 C      0.000000000000      0.000000000000     -0.767685465031
3 H      0.964354016767      0.347635559279      1.177128271450
4 H     -0.181115782790     -1.008972856410      1.177128271450
5 H     -0.783238233981      0.661337297125      1.177128271450
6 H     -0.500471876676      0.894626767091     -1.177128271450
7 H     -0.524533568868     -0.880734742626     -1.177128271450
8 H      1.025005445540     -0.013892024465     -1.177128271450
END

BASIS
type DZ
core Large
END

AnalyticalFreq
 NUC 3
End

BeckeGrid
 quality good
```

```
End

eor

rm TAPE13
mv TAPE21 ethane-frq.t21

# After the Hessian is calculated, the resulting TAPE21 file is used for a
# subsequent transition state search:

"$ADFBIN/adf" <<eor
TITLE Ethane transition state search using partial Hessian

ATOMS
1 C        0.000000000000        0.000000000000        0.767685465031
2 C        0.000000000000        0.000000000000       -0.767685465031
3 H        0.964354016767        0.347635559279        1.177128271450
4 H       -0.181115782790       -1.008972856410        1.177128271450
5 H       -0.783238233981        0.661337297125        1.177128271450
6 H       -0.500471876676        0.894626767091       -1.177128271450
7 H       -0.524533568868       -0.880734742626       -1.177128271450
8 H        1.025005445540       -0.013892024465       -1.177128271450
END

Fragments
  H t21.H
  C t21.C
END

GEOMETRY
    transitionstate mode=1
    iterations 30
    step rad=0.15
    hessupd BOFILL
    converge e=1.0e-4 grad=1.0e-4 rad=1.0e-3
END

RESTART
   File ethane-frq.t21
End

BeckeGrid
 quality good
End
eor

# Important note: care should be taken to specify correct mode in the
# TransitionState keyword. Because a significant part of the Hessian will not be
# calculated exactly, it is possible that it will have more than one negative
# eigenvalue, in which case the one we are interested in may not be the first
# one. In such a case, one needs to specify the correct mode number in the
# TransitionState keyword.
```

Example: Relativistic ZORA TS search: CH4 + HgCl2 <==> CH3HgCl + HCl

Download TS_CH4_HgCl2.run

```
#! /bin/sh


# A ZORA scalar relativistic Transition State calculation.

$ADFBIN/adf <<eor
TITLE  Transition State: CH4 + HgCl2 <==> CH3HgCl + HCl

noprint sfo,frag
print atdist

GEOMETRY
  TransitionState
  End
  step rad=0.1
END
TSRC
 DIST 1 5 1.0
 DIST 5 8 -1.0
END

Relativistic scalar ZORA

Basis
  Type TZP
  Core small
  Cl TZP/Cl.2p
End

ATOMS
   C         0.049484    0.042994    0.000000
   H        -0.068980    0.638928   -0.915972
   H        -0.068980    0.638928    0.915972
   H        -0.841513   -0.626342    0.000000
   H         0.555494   -1.148227    0.000000
   Hg        2.303289   -0.007233    0.000000
   Cl        4.429752    0.776056    0.000000
   Cl        1.342057   -2.676083    0.000000
END

eor

mv TAPE21 CH4_HgCl2.t21

# For the density-functional the Local Density approximation is used (no GGA
# corrections).

# At each geometry cycle the interatomic distance matrix is printed (print
# atdist).

# The initial geometry is a reasonable but not very accurate estimate of the
# Transition State. The program needs quite a few cycles to converge, which is
# rather typical for TS searches: they are a lot more tricky and fail more often
# than a simple minimization. The TSRC key is used to specify a reaction
# coordinate along which the transition state is sought for. This feature is
# especially useful when an accurate Hessian is not available.
```

Example: TS reaction coordinate: F- + CH3Cl

Download TSRC_SN2.run

```
#! /bin/sh


# With the TSRC key one can specify a reaction coordinate along which the
# transition state is sought for. This feature is especially useful when an
# accurate Hessian is not available.

# This example tries to find the TS for the S_N2 reaction of F^- + CH_3Cl <==>
# CH_3F + Cl^-

$ADFBIN/adf <<eor
Title Transition state search for Sn2 reaction of F- + CH3Cl

ANALYTICALFREQ
END

XC
 LDA VWN
 GGA OPBE
END

TSRC
  dist 1 5  1.0
  dist 1 6 -1.0
END

ATOMS
   C              0.000000     0.000000     0.000000
   H             -0.530807     0.919384693  0.012892
   H             -0.530807    -0.919384693  0.012892
   H              1.061614     0.000000     0.012892
   Cl             0.000000     0.000000    -2.124300
   F              0.000000     0.000000     2.019100
END

Geometry
 TransitionState
 End
End

BASIS
 type TZ2P
 core NONE
 createoutput none
END

BeckeGrid
 quality good
End

SCF
 converge 1.0e-6 1.0e-6
 diis ok=0.01
 iterations 99
```

```
END

Charge -1

EPRINT
 SFO noeig noovl
END

eor
```

Example: Constraint Linear Transit: H2O

Download LT_constraint.run

```
#!/bin/sh

# The LINEARCONSTRAINTS keyword allows geometry optimizations (old branch) with
# constraints defined by arbitrary linear combinations of (internal)
# coordinates. The constraint has to be satisfied at the start of the geometry
# optimization.

# Example for bond length constraint, where at the start of the linear transit
# rOH1=R1=1.0, and rOH2=R2=1.5, such that (-1.0)*R1+(1.0)*R2=0.5, and in the
# final geometry -R1+R2=0.0 (Reactcoord 0.5 0.0)


$ADFBIN/adf <<eor
title linearconstraints keyword
XC
  GGA Becke Perdew
END

Geometry
 Branch Old
 LinearTransit  6
End

BeckeGrid
 Quality good
End

Atoms Internal
O  0 0 0
H  1 0 0    R1
H  1 2 0    R2  109.9
End

GeoVar
 R1 =1.
 R2 =1.5
End

LinearConstraints
 ReactCoord 0.5 0.0
  R1  -1.0
  R2   1.0
 SubEnd
```

```
End

Basis
 Type DZP
End

eor

mv TAPE21 H2O.t21
```

Example: (non-)Linear Transit: H2O

Download Transit_H2O.run

```
#! /bin/sh


# ADF is capable of performing both linear transits, and non-linear transits,
# and is the default when the LINEARTRANSIT or TRANSIT sub-block is included in
# the 'Geometry' block.

# The new transit code works differently to the old: the transit is represented
# as a sequence of constrained optimizations. A 'Constraints' block is used to
# delineate the constraints applied at each stage of the transit.

# Non-linear transits are possible, and can even be combined with linear
# transits in other coordinates. To perform a non-linear transit in a particular
# coordinate, explicit values must be given.



$ADFBIN/adf <<eor
Title WATER Transit (non-linear), with the new optimizer branch

Atoms
    O            0.000000     0.000000     0.000000
    H            0.000000    -0.689440    -0.578509
    H            0.000000     0.689440    -0.578509
End

Symmetry NOSYM

Constraints
  dist 1 2 0.8 1.0 1.25 1.5
  angle 2 1 3 start=100.0 end=120.0
End

Basis
 Type SZ
 Core Large
End

Geometry
  LinearTransit 4
  Optim Deloc
  Converge grad=0.0001
!  Constraints PartialConverge
```

```
End

eor

# In the example above, 4 values are given for the distance between atoms 1 and
# 2. This distance constraint will be applied simultaneously with the linear
# transit constraints for the angle, with other degrees of freedom optimized at
# each stage of the transit.

# Finally, it should be pointed out that 'full constraints' are used by
# default in the transit calculations. These constraints are required to be
# fully met at each intermediate geometry. You can use partially converged␣
↪constraints,
# that are fully met only at the converged geometries, by supplying the
# PARTIALCONVERGE option to the 'Constraints' subkey of the 'Geometry' block (not
# to be confused with the 'Constraints' block at root level).
```

14.3.3 Total energy, Multiplet States, S2, Localized hole, CEBE

Example: Total Energy calculation: H2O

Download Energy_H2O.run

```
#! /bin/sh

# If the TOTALENERGY is included the total energy will be calculated.

# This example performs single point runs for H2 O with PBE/DZP with frozen
# cores and all-electron and B3LYP/DZP with all-electron and HARTREEFOCK/DZP
# with all-electron The tests run in C(2v) symmetry. Integration quality is good
# which should give total energies accurate at least up to 10^-4 atomic units.
# The key EXACTDENSITY is used for higher accuracy of the results.

# Note that only energy difference comparisons are meaningful. These are the
# only energies that play a role in chemistry of course, and for this one does
# not need total energies.


# ----------------------------
# H2O PBE/DZP with frozen core
# ----------------------------

"$ADFBIN/adf" <<eor
Title H2O PBE/DZP (frozen core) single point calculation

ATOMS
  O        0.00000        0.00000        0.00000
  H        0.00000       -0.68944       -0.57851
  H        0.00000        0.68944       -0.57851
END

BASIS
  Type DZP
  Core Small
END
```

```
XC
  GGA PBE
END

ZlmFit
  quality good
END
BeckeGrid
  quality good
END

EXACTDENSITY

EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END

NOPRINT BAS FUNCTIONS

TOTALENERGY
NOTOTEN
eor

rm TAPE21 logfile

# -----------------------
# H2O PBE/DZP all-electron
# -----------------------

"$ADFBIN/adf" <<eor
Title H2O PBE/DZP (all-electron) single point calculation

ATOMS
  O        0.00000        0.00000        0.00000
  H        0.00000       -0.68944       -0.57851
  H        0.00000        0.68944       -0.57851
END

ZlmFit
  quality good
END
BeckeGrid
  quality good
END

BASIS
  Type DZP
  Core None
END

XC
  GGA PBE
END

EXACTDENSITY

EPRINT
```

```
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END

NOPRINT BAS FUNCTIONS

TOTALENERGY
NOTOTEN
eor

rm TAPE21 logfile

# -------------------------
# H2O B3LYP/DZP all-electron
# -------------------------

"$ADFBIN/adf" <<eor
Title H2O B3LYP/DZP (all-electron) single point calculation

ATOMS
  O      0.00000       0.00000        0.00000
  H      0.00000      -0.68944       -0.57851
  H      0.00000       0.68944       -0.57851
END

ZlmFit
  quality good
END
BeckeGrid
  quality good
END

BASIS
  Type DZP
  Core None
END

XC
  HYBRID B3LYP
END

EXACTDENSITY

EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END

NOPRINT BAS FUNCTIONS

TOTALENERGY
NOTOTEN
eor

rm TAPE21 logfile

# --------------------------------
# H2O HARTREEFOCK/DZP all-electron
```

```
# -------------------------------

"$ADFBIN/adf" <<eor
Title H2O HARTREEFOCK/DZP (all-electron) single point calculation

ATOMS
   O        0.00000         0.00000         0.00000
   H        0.00000        -0.68944        -0.57851
   H        0.00000         0.68944        -0.57851
END

ZlmFit
  quality good
END
BeckeGrid
  quality good
END

BASIS
  Type DZP
  Core None
END

XC
  HARTREEFOCK
END

EXACTDENSITY

EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END

NOPRINT BAS FUNCTIONS

TOTALENERGY
NOTOTEN
eor

rm TAPE21 logfile
```

Example: Multiplet States: [Cr(NH3)6]3+

Download SD_CrNH3_6.run

```
#! /bin/sh

# The computation of multiplet states corresponding to an open-shell system can
# be carried out with ADF by first computing the 'Average-of-Configuration'
# (aoc) state, where all orbitals in the open shell are degenerate and equally
# occupied. This computation is spin-restricted and serves as a fragment file
# for the multiplet run, where then different occupation numbers are assigned to
# the various orbitals in the open shell. The corresponding energies are
# computed in the field of the aoc, which is achieved by not iterating the self-
# consistency equations to convergence but only computing the orbitals in the
# initial field.
```

```
# Since ADF requires that all symmetry-partners in an irreducible representation
# (irrep) have equal occupations, the multiplet calculation, where such orbitals
# are not equally occupied, must be carried out in a formally lower point group
# symmetry. The point group to select and the appropriate occupation numbers to
# apply must be worked out by the user 'on paper' in advance. An auxiliary
# program asf, developed by the group of Claude Daul in Fribourg can be used to
# determine which calculations are needed, and how to compute the multiplet
# energies from the results. See the discussion of Multiplet energies in the
# Theory document.

# The script starts with the 'creation' of the required basic atoms, N, H, Cr
# using a fair basis set quality.

# The next step is the computation of the ammonia fragment NH3 . This is not a
# crucial step here: the multiplet state computation can equally well be carried
# out by not using any intermediate compound fragments. However, it illustrates
# once more how a bigger molecule can be built up from smaller, but not trivial
# fragments.

$ADFBIN/adf  -n1  <<eor
create Cr file=$ADFRESOURCES/TZP/Cr.2p
eor
mv TAPE21 t21.Cr


$ADFBIN/adf  <<eor
title AMMONIA
NOPRINT sfo,frag,functions

atoms
  N    0                 0                      0
  H   -0.95522523        0                      0.3711068
  H    0.477612615   -0.8272493155158333        0.3711068
  H    0.477612615    0.8272493155158333        0.3711068
end

Basis
 Type TZP
 Core Small
End

symmetry  C(3V)

eor

mv TAPE21 t21.NH3

# The input of the atomic coordinates uses expressions, in this case to enforce
# exact symmetry relations that would otherwise require 14-digit input values or
# some inaccuracy. The symmetry specification is redundant: the program would
# also find it by itself.

# == Average-of-Configuration ==

# The next step is to compute the reference state, with respect to which we will
# later compute the multiplet states. The reference state is the so-called
# 'Average-of-configuration' (aoc) state. The result file (TAPE21) of this
```

```
# calculation will be used as a fragment file.

$ADFBIN/adf  <<eor
title Cr(NH3)6 : Average-of-Configuration run

COMMENT
using NH3-fragments
END

symmetry D(3d)

scf
iterations  25
mixing   0.15
end

atoms
 Cr     0.000000     0.000000     0.000000
  N     0.000000     1.714643     1.212436   f=NH3/1
  H     0.000000     1.466154     2.206635   f=NH3/1
  H    -0.827250     2.293404     1.036727   f=NH3/1
  H     0.827250     2.293404     1.036727   f=NH3/1
  N    -1.484924    -0.857321     1.212436   f=NH3/2
  H    -1.269726    -0.733077     2.206635   f=NH3/2
  H    -1.572521    -1.863121     1.036727   f=NH3/2
  H    -2.399771    -0.430282     1.036727   f=NH3/2
  N     1.484924    -0.857321     1.212436   f=NH3/3
  H     1.269726    -0.733077     2.206635   f=NH3/3
  H     2.399771    -0.430282     1.036727   f=NH3/3
  H     1.572521    -1.863121     1.036727   f=NH3/3
  N     0.000000    -1.714643    -1.212436   f=NH3/4
  H     0.000000    -1.466154    -2.206635   f=NH3/4
  H     0.827250    -2.293404    -1.036727   f=NH3/4
  H    -0.827250    -2.293404    -1.036727   f=NH3/4
  N     1.484924     0.857321    -1.212436   f=NH3/5
  H     1.269726     0.733077    -2.206635   f=NH3/5
  H     1.572521     1.863121    -1.036727   f=NH3/5
  H     2.399771     0.430282    -1.036727   f=NH3/5
  N    -1.484924     0.857321    -1.212436   f=NH3/6
  H    -1.269726     0.733077    -2.206635   f=NH3/6
  H    -2.399771     0.430282    -1.036727   f=NH3/6
  H    -1.572521     1.863121    -1.036727   f=NH3/6
end

fragments
Cr  t21.Cr
NH3 t21.NH3
end

IrrepOccupations
 A1.G 8.75
 A2.G 2
 E1.G 16 1.5 0.75
 A1.U 2
 A2.U 8
 E1.U 20
END
```

```
eor

mv TAPE21 t21.CrA6ES

# Occupation numbers are specified, to make certain what the reference state is
# that we will start from in the subsequent calculations. The result file TAPE21
# is saved to serve as fragment file in the subsequent calculations.

# == One-determinant states ==

# Now, we proceed with the multiplet calculations. In the example they are
# combined in one single run, but they could also be evaluated in separate runs.
# For each calculation it is required to:

# a) Use the aoc TAPE21 file as fragment file
# b) Choose which molecular orbitals in the open shell to occupy: select the
#    appropriate point group symmetry and the UnRestricted key if necessary and
#    specify the occupation numbers, using the irreducible representations of the
#    selected point group.

# The results are one-determinant calculations, which must then, later, be
# combined analytically to obtain the required multiplet energy values.

$ADFBIN/adf  <<eor
title Cr(NH3)6 : SlaterDeterminants run
NOPRINT frag

symmetry C(I)  !  lower symmetry

scf
iterations 0
end

atoms
   Cr       0.000000       0.000000       0.000000  f=CrA6
    N       0.000000       1.714643       1.212436  f=CrA6
    H       0.000000       1.466154       2.206635  f=CrA6
    H      -0.827250       2.293404       1.036727  f=CrA6
    H       0.827250       2.293404       1.036727  f=CrA6
    N      -1.484924      -0.857321       1.212436  f=CrA6
    H      -1.269726      -0.733077       2.206635  f=CrA6
    H      -1.572521      -1.863121       1.036727  f=CrA6
    H      -2.399771      -0.430282       1.036727  f=CrA6
    N       1.484924      -0.857321       1.212436  f=CrA6
    H       1.269726      -0.733077       2.206635  f=CrA6
    H       2.399771      -0.430282       1.036727  f=CrA6
    H       1.572521      -1.863121       1.036727  f=CrA6
    N       0.000000      -1.714643      -1.212436  f=CrA6
    H       0.000000      -1.466154      -2.206635  f=CrA6
    H       0.827250      -2.293404      -1.036727  f=CrA6
    H      -0.827250      -2.293404      -1.036727  f=CrA6
    N       1.484924       0.857321      -1.212436  f=CrA6
    H       1.269726       0.733077      -2.206635  f=CrA6
    H       1.572521       1.863121      -1.036727  f=CrA6
    H       2.399771       0.430282      -1.036727  f=CrA6
    N      -1.484924       0.857321      -1.212436  f=CrA6
    H      -1.269726       0.733077      -2.206635  f=CrA6
    H      -2.399771       0.430282      -1.036727  f=CrA6
```

```
    H      -1.572521      1.863121     -1.036727  f=CrA6
end

fragments
  CrA6  t21.CrA6ES
end

UnRestricted

SlaterDeterminants
  Check AOC
    A1.g   4 0.375         // 4 0.375
    A2.g   1               // 1
    E1.g:1 4 0.375 0.1875  // 4 0.375  0.1875
    E1.g:2 4 0.375 0.1875  // 4 0.375  0.1875
    A1.u   1//1
    A2.u   4//4
    E1.u:1 5//5
    E1.u:2 5//5
  SUBEND
  State1
    A1.g   4 1             // 4 1
    A2.g   1               // 1
    E1.g:1 4 0     0       // 4 0      1
    E1.g:2 4 0     0       // 4 0      0
    A1.u   1//1
    A2.u   4//4
    E1.u:1 5//5
    E1.u:2 5//5
  SUBEND
  State2
    A1.g   4 1             // 4 1
    A2.g   1               // 1
    E1.g:1 4 0     0       // 4 1      0
    E1.g:2 4 0     0       // 4 0      0
    A1.u   1//1
    A2.u   4//4
    E1.u:1 5//5
    E1.u:2 5//5
  SUBEND
  State3
    A1.g   4 1             // 4 1
    A2.g   1               // 1
    E1.g:1 4 0     1       // 4 0      0
    E1.g:2 4 0     0       // 4 0      0
    A1.u   1//1
    A2.u   4//4
    E1.u:1 5//5
    E1.u:2 5//5
  SUBEND
end

eor

mv TAPE21 CrNH3_6.t21

# The SlaterDeterminants block may contain any number of sub blocks, each
# starting with an (arbitrary) title record, followed by a set of occupation
```

```
# numbers and closed by a SubEnd record. Each such subkey block specifies a
# single one-determinant-state calculation. All occupation numbers must
# reference the irreps of the specified point group symmetry, C(I) in the
# example, and must be just a reassignment of the electrons that are equally
# distributed over the corresponding degenerate irreps in the reference aoc
# calculation.

# The so-obtained energies of the one-determinant states can now be combined to
# calculate the desired multiplet energies. See the Theory document and the adf
# User's Guide.

# Note carefully that in the calculation of the SingleDeterminants, the scf
# procedure is prevented to cycle to convergence by setting the subkey
# Iterations to zero in the SCF data block.
```

Example: Calculation of S2: CuH+

Download CuH+_S-squared.run

```
#! /bin/sh

# Example calculates expectation value of S^2 (<S^2>) of CuH+ in various
# symmetries, using unrestricted density functional theory.


$ADFBIN/adf -n 1 <<eor
Create H $ADFRESOURCES/TZP/H
eor

mv TAPE21 t21.H

$ADFBIN/adf -n 1 <<eor
Create Cu $ADFRESOURCES/TZP/Cu.2p
eor

mv TAPE21 t21.Cu

$ADFBIN/adf <<eor
Title calculate expectation value of S-squared

Symmetry Nosym

ATOMS Z-Matrix
 Cu   0 0 0
 H    1 0 0  1.463
END

CHARGE 1.0 1.0
Unrestricted

FRAGMENTS
H  t21.H
Cu t21.Cu
END

eor
```

```
rm TAPE21 logfile

$ADFBIN/adf <<eor
Title calculate expectation value of S-squared

Symmetry C(LIN)

ATOMS Z-Matrix
 Cu    0 0 0
 H     1 0 0  1.463
END

CHARGE 1.0 1.0
Unrestricted

FRAGMENTS
H  t21.H
Cu t21.Cu
END

eor


rm TAPE21 logfile

$ADFBIN/adf <<eor
Title calculate expectation value of S-squared

Symmetry C(S)

ATOMS Z-Matrix
 Cu    0 0 0
 H     1 0 0  1.463
END

CHARGE 1.0 1.0
Unrestricted

FRAGMENTS
H  t21.H
Cu t21.Cu
END

eor

rm TAPE21 logfile

$ADFBIN/adf <<eor
Title calculate expectation value of S-squared

ATOMS Z-Matrix
 Cu    0 0 0
 H     1 0 0  1.463
END

CHARGE 1.0 -1.0
Unrestricted
```

```
FRAGMENTS
H  t21.H
Cu t21.Cu
END

eor

mv TAPE21 CuH_plus.t21
```

Example: Localized Hole: N2+

Download ModStPot_N2+.run

```
#! /bin/sh

# This calculation illustrates:

# - How to specify the net total charge on a molecule
# - How to enforce breaking the symmetry that is present in the start-up
#   situation, in this case to localize a hole in the electron density on one of
#   the two equivalent atoms.
# - How to prevent the scf from oscillating back and forth between the two
#   equivalent situations or from even restoring the unwanted symmetry


$ADFBIN/adf <<eor
title N2+  hole localization

atoms
N 0 0  -2.0
N 0 0   2.0
end

Basis
 Type DZP
 Core Small
End

SCF
  DIIS ok=0.5
END

symmetry C(lin)  ! allow symmetry breaking

unrestricted

IrrepOccupations
  ! do not use keeporbitals as it may lead to a wrong configuration
  sigma 3 // 1 0 1
  pi    2 // 2
end

CHARGE  1  1   ! this duplicates info from "OCCUPATIONS" (check)

modifystartpotential  ! to break the symmetry in the start-up potential
N/1    0.5   0.5
N/2      4    1
```

```
end

eor

# The purpose of this run is to compute the N2+ ion, with the hole localized on
# one of the atoms. In a very small system like N2+ this is a tricky thing to
# do. The program has a tendency towards the symmetric solution, with the hole
# delocalized. A few trial runs, just putting a net +1 charge into the system,
# will reveal that clearly.

# To achieve the desired situation we apply the key modifystartpotential to
# break the symmetry of the initial potential. A potential is generated as if
# the electronic cloud in the second N fragment is spin-polarized in a ratio 4:1
# (this precise value is not very relevant), which achieves that initially a
# non-symmetric solution is obtained. The symmetry must be specified, lest the
# program determine and use the higher symmetry from the nuclear frame. This
# would prevent any symmetry breaking altogether.

# Next, in order to prevent that the system relaxes to the symmetric situation,
# we apply the keeporbitals option of the occupations key. This fixes the
# occupied orbitals in the sense that in each scf cycle the program will try to
# keep the electrons in orbitals that resemble the previously occupied orbitals
# as much as possible.

# The key modifystartpotential here demonstrated has a more relevant and less
# unstable application in larger systems. See the User's Guide for references.
```

Example: Broken spin-symmetry: Fe4S4

Download Fe4S4_BrokenSymm.run

```
#! /bin/sh

# This calculation shows a spin-flip restart feature that allows to exchange
# alpha and beta fit coefficients for selected atoms upon restart. First the
# high spin configuration with 8 more alpha-electrons than beta-electrons is
# calculated (S_z=4). Next the broken spin-symmetry configuration is calculated
# (S_z=0), using the subkey spinflip in the restart key. In this case the spin
# will be flipped for iron atoms 1 and 2. Note that the used symmetry C(2v) is
# lower than the actual symmetry T(d), such that iron atoms 1 and 2 can have a
# different spin than iron atoms 3 and 4.


"$ADFBIN/adf" <<eor
TITLE Fe4S4 High-spin configuration

ATOMS
 Fe     -0.000000000000     -1.256142548900      0.888226914500
 Fe      0.000000000000      1.256142548900      0.888226914500
 Fe     -1.256142548900      0.000000000000     -0.888226914500
 Fe      1.256142548900     -0.000000000000     -0.888226914500
  S     -1.845393493800      0.000000000000      1.304890253400
  S      1.845393493800     -0.000000000000      1.304890253400
  S     -0.000000000000     -1.845393493800     -1.304890253400
  S      0.000000000000      1.845393493800     -1.304890253400
END
```

```
Symmetry C(2v)

CHARGE 0.0 8.0

UNRESTRICTED

BASIS
type DZ
core Large
createoutput None
END

XC
GGA OPBE
END

EPRINT
 ! Omit printing of atomic population numbers since they may change per computer
 ! In real-life calculations you may want to have them
 SCF NoPOP
END

eor

mv TAPE21 Fe4S4-high-spin.t21

"$ADFBIN/adf" <<eor
TITLE Fe4S4 LOW-spin configuration

Restart
   File Fe4S4-high-spin.t21
   ! Make sure atoms specified in the SpinFlip keyword are symmetry-equivalent
   SpinFlip 1 2
End

ATOMS
 Fe     -0.000000000000     -1.256142548900      0.888226914500
 Fe      0.000000000000      1.256142548900      0.888226914500
 Fe     -1.256142548900      0.000000000000     -0.888226914500
 Fe      1.256142548900     -0.000000000000     -0.888226914500
  S     -1.845393493800      0.000000000000      1.304890253400
  S      1.845393493800     -0.000000000000      1.304890253400
  S     -0.000000000000     -1.845393493800     -1.304890253400
  S      0.000000000000      1.845393493800     -1.304890253400
END

Symmetry C(2v)

CHARGE 0.0 0.0

UNRESTRICTED

BASIS
type DZ
core Large
createoutput None
END
```

```
XC
GGA OPBE
END

eor
```

Example: Core-electron binding energies (CEBE): NNO

Download CEBE_NNO.run

```
#!/bin/sh


# ADF is well suited for calculating Core Electron Binding Energies (CEBEs). In
# this example it is shown how one can differentiate between the 1s CEBEs of the
# two non-equivalent nitrogen atoms in N2 O, using a delta-SCF technique. It
# starts with a regular calculation that has the purpose of preparing a
# reference TAPE21 file for the NNO molecule, which will later be useful in the
# energy analysis. The result file is saved to t21.NNO.

# The same GGA functional is specified throughout the run. The amount of output
# is reduced by using some print keys.

# The prepare the nitrogen atom with a core hole (restricted) will be used as a
# fragment later. This enables selection of where the core hole should be.

# prepare the Nitrogen core hole restricted to be used as fragment


$ADFBIN/adf -n1 <<eor
title   N atom core hole
ATOMS
 N   0.0    0.0     0.0
end
Basis
 Type TZ2P
 Core None
end
xc
 GGA pw86x pw91c
end
numericalquality good
IrrepOccupations
 s    1 2
 p    3
end
eor

mv TAPE21 t21.N_ch


# Now perform the restricted ground state molecule for analysis later. The
# TAPE21 result file is saved.
# Remark: In the core hole calculation the 2 N atoms will have different fragments.
# Therefore also in this calculation the 2 N atoms are treated with 2 different
↪fragments,
# which is done here by using 2 different labels N.1 and N.2.
```

```
$ADFBIN/adf <<eor
title   NNO
noprint sfofragpop fragsfo
xc
 GGA pw86x pw91c
end
numericalquality good
ATOMS
 N.1 0.0    0.0    -1.1284
 N.2 0.0    0.0     0.0
 O   0.0    0.0     1.1841
end
Basis
 Type TZ2P
 Core None
 createoutput None
end
eor


mv TAPE21 NNO.t21



# Next follow two sets of almost identical calculations in which a 1s electron
# is removed from one or the other N atom (please note that the deepest s level
# is associated with the 1s of the oxygen atom). The molecular NNO result file
# is used as fragment. An unrestricted calculation is done and a positive charge
# is specified. The final result file for the molecule with the core hole is
# saved. Then another calculation is done to conveniently obtain the energy with
# respect to the normal molecule. This is repeated for a core hole on the other
# N atom.



$ADFBIN/adf <<eor
title NNO unrestricted core hole
noprint sfofragpop fragsfo
ATOMS
 N.1 0.0    0.0    -1.1284        f=N_ch
 N.2 0.0    0.0     0.0
 O   0.0    0.0     1.1841
end
xc
 GGA pw86x pw91c
end
numericalquality good
Basis
 Type TZ2P
 Core None
 createoutput None
end
fragments
 N_ch  t21.N_ch
end
unrestricted
charge 1 1
IrrepOccupations
 sigma  1 1 1 4 // 1 0 1 4
```

```
 pi     4       // 4
end
eor

mv TAPE21 NNO.unr1.t21


# In the second calculation the result file of one of the unrestricted NNO
# calculations is used as restart file, which ensures that the hole stays at its
# place, because the starting density is already correct. The result file
# t21.NNO for the normal NNO calculation is specified as fragment to serve as an
# energy reference. The final Bonding Energy printed by ADF indicates what the
# CEBE is. However, please check Chong, D.P. Accurate DFT Calculation of Core-
# Electron Binding Energies in Reviews in Modern Quantum Chemistry, A
# Celebration of the Contributions of R.G. Parr, edited by K.D. Sen (World
# Scientific Publishing Co., Singapore), 1106-1139 (2002) for more detailed
# information on Core-Electron Binding Energies. This reference also contain
# information on empirical corrections that may have to be made on the final
# numbers.


$ADFBIN/adf <<eor
title   NNO unr. core hole
noprint sfofragpop fragsfo
xc
 GGA pw86x pw91c
end
numericalquality good
restart
    File NNO.unr1.t21
End
ATOMS
 N.1 0.0    0.0    -1.1284        f=NNO
 N.2 0.0    0.0     0.0           f=NNO
 O   0.0    0.0     1.1841        f=NNO
end
fragments
 NNO      NNO.t21
end
unrestricted
charge 1 1
IrrepOccupations
  sigma  1 1 1 4 // 1 0 1 4
  pi     4       // 4
end
eor

mv TAPE21 NNO.unr1.1.t21
rm logfile


# core hole nr. 2

$ADFBIN/adf <<eor
title NNO unrestricted core hole
noprint sfofragpop fragsfo
xc
 GGA pw86x pw91c
```

```
end
numericalquality good
ATOMS
 N.1 0.0    0.0    -1.1284
 N.2 0.0    0.0     0.0            f=N_ch
 O   0.0    0.0     1.1841
end
Basis
 Type TZ2P
 Core None
 createoutput None
end
fragments
 N_ch   t21.N_ch
end
unrestricted
charge 1 1
IrrepOccupations
 sigma  1 1 1 4 // 1 0 1 4
 pi     4       // 4
end
eor

mv TAPE21 NNO.unr2.t21

# analysis for core hole 2

$ADFBIN/adf <<eor
title   NNO unr. core hole
noprint sfofragpop fragsfo
xc
 GGA pw86x pw91c
end
numericalquality good
restart
   File NNO.unr2.t21
End
ATOMS
 N.1 0.0    0.0    -1.1284        f=NNO
 N.2 0.0    0.0     0.0           f=NNO
 O   0.0    0.0     1.1841        f=NNO
end
fragments
 NNO      NNO.t21
end
unrestricted
charge 1 1
IrrepOccupations
  sigma  1 1 1 4 // 1 0 1 4
  pi     4       // 4
end
eor

mv TAPE21 NNO.unr3.t21

# Similarly, one could easily have prepared an oxygen with a core hole and
# determined the CEBE of the oxygen 1s atom.
```

Example: Constrained DFT: H2O+ ... H2O

Download WaterDimer.run

```sh
#!/bin/sh

# Example of Constrained Density Functional Theory (CDFT) for the charged Water
# dimer, in which the charge is constrained to be on one of the waters. In this
# case there is one constraint, with a total charge of 1.0 on the atoms 1-3.

$ADFBIN/adf <<eor
Title water dimer h2o+ ... h2o
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
SYMMETRY NOSYM
XC
  GGA PW91
END
CHARGE 1 1
UNRESTRICTED
BASIS
 Type TZP
 Core none
 CreateOutput None
END
CDFT
 NCONSTRAINTS 1
 NATOMSPERSET 3
 THEATOMS 1 2 3
 CONSTRAINTS  1.0
END
ATOMS
O        0.0000000000         0.0000000000         0.0000000000
H       -0.9358409558          .2646136961         0.0000000000
H       -0.0304663436        -0.9828924420         0.0000000000
O        0.0000000000        -2.9053396088         0.0000000000
H       -0.4092227596        -3.3374838250        -0.7701260000
H       -0.4092227596        -3.3374838250         0.7701260000
END
eor

rm TAPE21 logfile
```

14.4 Spectroscopic Properties

14.4.1 IR Frequencies, (resonance) Raman, VROA, VCD, Franck-Condon factors

Example: Numerical Frequencies: NH3

Download Freq_NH3.run

```sh
#! /bin/sh
```

```
# Summary:
# - Frequencies with symmetric displacements
# - Frequencies with Cartesian displacements
# - Isotope effects in the frequencies


# == Frequencies with symmetric displacements ==

# Computation of frequencies by symmetric displacements. The assumed equilibrium
# input structure should be given in Cartesian coordinates.

# The symmetry is determined automatically by the program as C(3v), from the
# input coordinates. During the calculation first symmetric atomic displacements
# are constructed. The number of such displacements in each irreducible
# representation corresponds to the number of frequencies with the corresponding
# symmetry. All displaced geometries within one representation have the same
# symmetry, which enables us to use it to speed up the computation
# significantly.

$ADFBIN/adf <<eor
title NH3 frequencies in symmetric displacements

atoms
   N              0.0000    0.0000    0.0000
   H              0.4729    0.8190    0.3821
   H             -0.9457    0.0000    0.3821
   H              0.4729   -0.8190    0.3821
end

Basis
 Type TZP
 Core Small
End

geometry
  frequencies Symm=True
end

thermo   TMin=300 TMax=400

BeckeGrid
 quality good
End

eor

mv TAPE21 NH3_symm.t21


# == Frequencies with Cartesian displacements ==

# Computation of frequencies by Cartesian displacements. The assumed equilibrium
# input structure is given in internal coordinates. A dummy atom is used for a
# convenient definition of the Z-matrix such that it reflects the pointgroup
# symmetry C(3v).


$ADFBIN/adf <<eor
```

```
title NH3 frequencies

atoms    Z-matrix
  XX    0 0 0
  N     1 0 0   1.0
  H     2 1 0   1.02    112.
  H     2 1 3   1.02    112.    120.
  H     2 1 4   1.02    112.    120.
end

Fragments
 N t21.N
 H t21.H
End

geometry
  optim   cartesian
  frequencies
  End
end

thermo   TMin=300 TMax=400

BeckeGrid
 quality good
End

eor


# The symmetry is determined automatically by the program as C(3v), from the
# input coordinates. In a Frequencies calculation the symmetry (specified on
# input or computed internally) is used for analysis and in some cases to speed
# up the calculation.

# The equilibrium coordinate values are supplied as identifiers that are
# associated with values in the define block.

# Unlike using the geovar key, applying the define key does not mean anything in
# the sense that the various coordinates that refer to the same identifier would
# be forced to remain equal; it is just a way to display (to the human reader)
# symmetry in the equilibrium values, to avoid typing errors and to allow an
# easy adjustment of starting coordinates for another calculation.

# Since the atomic coordinates are input in Z-matrix format, the program would
# by default carry out displacements in internal coordinates to scan the energy
# surface and hence compute force constants and frequencies. This is overridden
# by specifying in the geometry block optim cartesian: carry out cartesian
# displacements.

# The key thermo addresses the thermodynamical analysis (only available in a
# Frequencies calculation, otherwise ignored). The specification 'T=300,400'
# means that the thermodynamic properties are printed for the temperature range
# 300-400K, in steps of 10K (default) and for a pressure of 1.0 atmosphere
# (default).

# Frequencies calculations suffer easily from numerical inaccuracies. Therefore,
# the default numerical integration precision in a Frequencies calculation is
```

```
# much higher than in an ordinary single-point or minimization run.


# == Isotope effects in the frequencies ==

# Rename the TAPE21 result file of the previous calculation so we can restart
# with other masses. Calculate a different isotope of H, in this case deuterium.
# It will differ from the original one only in the mass of the nucleus. Repeat
# the frequency calculation with different fragments. It is important to
# preserve symmetry at this step so we replace fragment files for ALL H atoms.
# If you want to replace only one fragment then the original calculation must be
# performed the same way, with different fragment names.

mv TAPE21 restart.t21

#  Calculate a different isotope of H. It will differ from the
#  original one only in the mass of the nucleus.

$ADFBIN/adf <<eor
create H M=2.014101779 $ADFRESOURCES/TZP/H
eor

mv TAPE21 t21.D

$ADFBIN/adf <<eor
title NH3 frequencies

atoms    Z-matrix
  XX    0 0 0
  N     1 0 0   1.0
  H     2 1 0   1.02   112.
  H     2 1 3   1.02   112.   120.
  H     2 1 4   1.02   112.   120.
end

Fragments
  N t21.N
! The different isotope mass sits in the next line.
  H t21.D
End

geometry
  optim cartesian
  frequencies
  End
end

! Restart the frequency calculation.
! In fact ADF should perform only one geometry cycle
restart
  File restart.t21
End

thermo  TMin=300 TMax=400

BeckeGrid
 quality good
End
```

```
eor

mv TAPE21 NH3.t21
```

Example: Numerical Frequencies, spin-orbit coupled ZORA: UF6

Download Freq_UF6.run

```sh
#! /bin/sh

# Summary:

# - Geometry optimization
# - Frequencies with symmetric displacements
# - Both for scalar relativistic ZORA and spin-orbit coupled ZORA


# == Geometry optimization ==

# Here only the spin-orbit coupled input file for ADF is given (in the scalar
# relativistic case change 'spinorbit' in 'scalar'). The resulting TAPE21 is
# saved such that it can be used in the frequency calculation.


$ADFBIN/adf  <<eor
Title  UF6 geometry optimization: scalar ZORA

BeckeGrid
 Quality good
End

Geometry
 converge grad=1e-4
End

relativistic scalar zora

Basis
 Type TZP
end

ATOMS
1 U   0.00000   0.00000   0.00000
2 F   2.00000   0.00000   0.00000
3 F  -2.00000   0.00000   0.00000
4 F   0.00000   2.00000   0.00000
5 F   0.00000  -2.00000   0.00000
6 F   0.00000   0.00000   2.00000
7 F   0.00000   0.00000  -2.00000
END

eor

mv TAPE21 UF6.t21
```

```
# == Frequencies with symmetric displacements ==

# Computation of frequencies by symmetric displacements. The assumed equilibrium
# input structure should be given in Cartesian coordinates. The calculation
# starts with the optimized structure read from UF6.t21 (restart file). Again
# only the spin-orbit coupled input file for ADF is given.

# The symmetry is determined automatically by the program as O(H), from the
# input coordinates. During the calculation first symmetric atomic displacements
# are constructed. The number of such displacements in each irreducible
# representation corresponds to the number of frequencies with the corresponding
# symmetry. All displaced geometries within one representation have the same
# symmetry, which enables us to use it to speed up the computation
# significantly.


$ADFBIN/adf  <<eor
Title  UF6 frequencies and IR intensities: scalar ZORA

Restart
   File UF6.t21
End

Geometry
  Frequencies Symm=True
End

BeckeGrid
 Quality good
End

relativistic scalar zora

Fragments
 U t21.U
 F t21.F
end

ATOMS
1 U  0.00000   0.00000   0.00000
2 F  2.00000   0.00000   0.00000
3 F -2.00000   0.00000   0.00000
4 F  0.00000   2.00000   0.00000
5 F  0.00000  -2.00000   0.00000
6 F  0.00000   0.00000   2.00000
7 F  0.00000   0.00000  -2.00000
END

eor

rm TAPE21 logfile UF6.t21


# Optimization and frequencies for spinorbit ZORA


$ADFBIN/adf  <<eor
```

```
Title  UF6 geometry optimization: spinorbit ZORA

BeckeGrid
 Quality good
End

Geometry
 converge grad=1e-4
End

relativistic spinorbit zora

Fragments
 U t21.U
 F t21.F
end

ATOMS
1 U  0.00000   0.00000   0.00000
2 F  2.00000   0.00000   0.00000
3 F -2.00000   0.00000   0.00000
4 F  0.00000   2.00000   0.00000
5 F  0.00000  -2.00000   0.00000
6 F  0.00000   0.00000   2.00000
7 F  0.00000   0.00000  -2.00000
END

eor

mv TAPE21 UF6.t21


$ADFBIN/adf  <<eor
Title  UF6 frequencies and IR intensities: spinorbit ZORA

Restart
   File UF6.t21
End

Geometry
  Frequencies Symm=True
End

BeckeGrid
 Quality good
End

relativistic spinorbit zora

Fragments
 U t21.U
 F t21.F
end

ATOMS
1 U  0.00000   0.00000   0.00000
2 F  2.00000   0.00000   0.00000
3 F -2.00000   0.00000   0.00000
```

```
4 F   0.00000    2.00000    0.00000
5 F   0.00000   -2.00000    0.00000
6 F   0.00000    0.00000    2.00000
7 F   0.00000    0.00000   -2.00000
END

eor

rm TAPE21 logfile UF6.t21
```

Example: Numerical Frequencies, accurate Hartree-Fock: H2O

Download H2O_HF_freq.run

```
#!/bin/sh

# Example shows a Hartree-Fock frequency calculation with an accurate basis set.

# The 'DEPENDENCY' key is set to 1e-4. Note that for hybrids and Hartree-Fock
# the dependency key is always set. The default value in that case is 4e-3. By
# explicitly setting the 'DEPENDENCY' key we can use a lower value, which is
# possible in this case. One should check that the results remain reliable if
# one uses a smaller value for the 'DEPENDENCY' key.



# First a geometry optimization is performed.


$ADFBIN/adf <<eor
title accurate HF geometry optimization with large QZ4P basis set
basis
 type QZ4P
 core None
end

adddiffusefit
dependency bas=1e-4
ATOMS
    O          0.000000    0.000000   -0.007124
    H          0.000000    0.751933    0.556531
    H          0.000000   -0.751933    0.556531
END

NumericalQuality Good
xc
 hartreefock
end

geometry
end

eor
rm logfile
mv TAPE21 t21


# Next the frequency calculation is done. A restart is used to pick up the
```

```
# excited state geometry of the previous calculation. The orbitals are not read
# from the restart file (subkey NoOrb in the restart key). Also the fit
# coefficients are not read from the restart file (subkey NoSCF in the restart
# key).


$ADFBIN/adf <<eor
title accurate HF frequency calculation with large QZ4P basis set
Restart
   File t21
   noscf True
   noorb True
end
basis
 type QZ4P
 core None
end

adddiffusefit
dependency bas=1e-4
ATOMS
    O         0.000000    0.000000   -0.007124
    H         0.000000    0.751933    0.556531
    H         0.000000   -0.751933    0.556531
END

NumericalQuality Good
xc
 hartreefock
end

geometry
 frequencies Symm=True
end

eor
rm TAPE21 logfile
```

Example: Numerical Frequencies of an excited state: PH2

Download EGO_PH2.run

```
#!/bin/sh

# Example for an excited state geometry optimization and frequency calculation.

# Needed for such excited state optimizations are the key EXCITATIONS (to
# calculate excitation energies), the key GEOMETRY (to do a geometry
# optimization) and the key EXCITEDGO (to select for which excitation a geometry
# optimization should be performed). The ground state and excited state are open
# shell.


$ADFBIN/adf <<eor
TITLE PH2 Excited state geometry

atoms
```

```
    P           0.000000    0.000000   0.0
    H           0.7         0.0        0.7
    H          -0.7         0.0        0.7
end

XC
 GGA BP86
END

UNRESTRICTED

CHARGE 0 1
BeckeGrid
 Quality good
End
ExactDensity

GEOMETRY
 ITERATIONS 50
 CONVERGE E=0.0001 grad=0.00001
END
SCF
   converge 1.0e-9
END

basis
 TYPE DZ
 CORE NONE
end

excitations
LOWEST 10
onlysing
end

EXCITEDGO
 STATE  B2 1
 OUTPUT 1
end

eor

cp TAPE21 PH2.t21


# Next the frequencies are calculated of the excited state. A restart is used to
# pick up the excited state geometry of the previous calculation. Note that in a
# numerical FREQUENCIES calculation symmetry is turned off except to reduce the
# number of points calculated. Thus irrespective of the specified point group
# symmetry the symmetry label A of SYMMETRY NOSYM should be used to select the
# excited state. Care should be taken to ensure that the correct state is chosen
# in this frequencies calculation as the excited state number can change when
# the point group is changed. In this case instead of 'B2 1' one needs to select
# 'A 1'. Accurate SCF convergence parameters are used.


$ADFBIN/adf <<eor
TITLE PH2 Excited state frequencies
```

```
atoms
P        0.000000    0.000000    0.002878
H        1.258230    0.000000    0.655775
H       -1.258230    0.000000    0.655775
end

XC
 GGA BP86
END

UNRESTRICTED

CHARGE 0 1
BeckeGrid
 Quality good
End
ExactDensity

GEOMETRY
 FREQUENCIES
 End
END
SCF
   converge 1.0e-9
END

basis
 TYPE DZ
CORE NONE
end

excitations
LOWEST 10
onlysing
end

EXCITEDGO
 STATE  A 1
 OUTPUT 2
 CPKS EPS=0.000001
end

eor
```

Example: Analytic Frequencies: CN

Download CN_SecDeriv.run

```
#! /bin/sh



# Calculation of analytical second derivatives is requested by specifying:
#   AnalyticalFreq
#   End
# in the ADF input.
```

```
# A good quality is specified for the numerical Becke integration to be sure of
# reliable results. In general, it seems advisable to use high accuracy for
# heavy nuclei at the moment, whereas default integration accuracy is usually
# sufficient for light atoms. The precision of the fit may be improved with the
# ZlmFit block keyword.


$ADFBIN/adf <<eor
title CN

atoms
 N  -1.3  0.0   0.0
 C   0.0  0.0   0.0
end

Basis
 Type DZ
 Core None
End

charge -1

XC
 LDA Xonly
End

BeckeGrid
 quality good
End

AnalyticalFreq
End

eor

mv TAPE21 CN.t21
rm -f logfile

# After SCF is completed, the energy second derivatives matrix is calculated and
# analysed, which yields in this case one frequency.
```

Example: Analytic Frequencies: CH4

```
Download CH4_SecDeriv.run
```

```
#! /bin/sh

# In this example we perform a geometry optimization immediately followed by
# calculation of frequencies. This is done by specifying the Geometry and
# AnalyticalFreq input blocks in one file.

# Note: when using this feature, one might improve the precision if one uses
# NUMERICALQUALITY GOOD. In order to save time this is not done in this example.


$ADFBIN/adf <<eor
title CH4 LDA potential
```

```
Atoms
 C  0 0 0
 H  0.6264250420707439  0.6264250420707439  0.6264250420707439
 H  0.6264250420707439 -0.6264250420707439 -0.6264250420707439
 H -0.6264250420707439  0.6264250420707439 -0.6264250420707439
 H -0.6264250420707439 -0.6264250420707439  0.6264250420707439
End

Basis
 Type TZP
 Core None
End

Geometry
 converge grad=0.0001
End

AnalyticalFreq
End

eor

mv TAPE21 CH4.t21
rm logfile
```

Example: Analytic Frequencies, scalar ZORA: HI

Download HI_SecDer_ZORA.run

```
#!/bin/sh

# Analytical frequencies of HI using the scalar ZORA relativistic approximation.
# A good NumericalQuality is used.


$ADFBIN/adf <<eor
TITLE HI scalar, ZORA,

ATOMS
 I 0.0 0.0 0.0
 H 0.0 0.0 1.6090
END

XC
 LDA Xonly
END

RELATIVISTIC scalar ZORA

symmetry NOSYM

Basis
 Type DZ
 Core None
END
```

```
NumericalQuality good

AnalyticalFreq
  print computation numintpar
End

eor

mv TAPE21 HI.t21
```

Example: Mobile Block Hessian (MBH): Ethanol

Download MBH_Ethanol.run

```
#! /bin/sh



# A frequency calculation is performed using the mobile block Hessian (MBH)
# method. The coordinates in the ATOMS section should be the partially optimized
# coordinates (or the fully optimized coordinates would work too). The next
# input for ADF shows how to perform a frequency calculation with MBH. The flag
# b=b1 in the ATOMS section adds the label 'b1' to some of the atoms. Only the
# four atoms labeled 'b1' (CH3 ) will be considered as a block with fixed
# internal geometry.



$ADFBIN/adf <<eor
TITLE ethanol: second derivatives with MBH approach. CH3 is treated as a rigid block

ATOMS
1 C     -0.029587   -0.006554    0.008124     b=b1
2 H     -0.087498   -0.025163    1.109913     b=b1
3 H      1.027473   -0.056237   -0.302751     b=b1
4 H     -0.565305   -0.891154   -0.376242     b=b1
5 C     -0.694908    1.238909   -0.501807     b=b2
6 H     -0.670258    1.265092   -1.608847     b=b2
7 O     -2.069894    1.175059   -0.017251
8 H     -0.182335    2.138977   -0.109315     b=b2
9 H     -2.586972    1.972802   -0.317216
END

SYMMETRY nosym

BASIS
  type DZ
  core Large
  CreateOutput None
END

XC
LDA SCF VWN
END

SCF
  Converge 1.0e-8
END
```

```
GEOMETRY
   frequencies
   End
   mbh b1
END

NumericalQuality good
eor

mv TAPE21 Ethanol_MBH.t21


# For comparison in this example also a calculation is performed without any
# restrictions.


$ADFBIN/adf <<eor
TITLE ethanol: complete vibrational spectrum, compare with MBH above

ATOMS
1 C      -0.029587   -0.006554    0.008124
2 H      -0.087498   -0.025163    1.109913
3 H       1.027473   -0.056237   -0.302751
4 H      -0.565305   -0.891154   -0.376242
5 C      -0.694908    1.238909   -0.501807
6 H      -0.670258    1.265092   -1.608847
7 O      -2.069894    1.175059   -0.017251
8 H      -0.182335    2.138977   -0.109315
9 H      -2.586972    1.972802   -0.317216
END

BASIS
  type DZ
  core Large
  CreateOutput None
END

XC
LDA SCF VWN
END

AnalyticalFreq
End

NumericalQuality good
eor

mv TAPE21 Ethanol_full.t21
```

Example: Mobile Block Hessian: CH4

Download MBH_CH4.run

```
#! /bin/sh

# A frequency calculation is performed using the mobile block Hessian
# (MBH)method. The coordinates in the ATOMS section should be the partially
```

```
# optimized coordinates (or the fully optimized coordinates would work too).

# Example input how to do a block constraint:
# ATOMS
#     C           0.000000     0.000000     0.000000 b=b1
#     H           0.634671     0.634671     0.634671 b=b1
#     H          -0.634671    -0.634671     0.634671 b=b1
#     H          -0.634671     0.634671    -0.634671 b=b1
#     H           0.634671    -0.634671    -0.634671
# END
# CONSTRAINTS
#   block b1
# END

# Such geometry optimization will not be discussed here any further. The next
# input for ADF shows how to perform a frequency calculation with MBH.


$ADFBIN/adf  <<eor
TITLE Methane

BASIS
 Type DZ
 Core None
END

ATOMS
    C           0.000000     0.000000     0.000000 b=b1
    H           0.634671     0.634671     0.634671 b=b1
    H          -0.634671    -0.634671     0.634671 b=b1
    H          -0.634671     0.634671    -0.634671 b=b1
    H           0.634671    -0.634671    -0.634671 b=b2
END

NumericalQuality verygood

SYMMETRY nosym

GEOMETRY
   frequencies disrad=0.001
   mbh b1
   branch new
END
eor

# The flag b=b1 in the ATOMS section adds the label 'b1' to some of the atoms.
# The four atoms labeled 'b1' will be considered as a block with fixed internal
# geometry.

# In the GEOMETRY section, a Mobile Block Hessian calculation is requested by
# using the FREQUENCIES and MBH keywords. Here the atoms with label 'b1' are
# selected to be in the same mobile block. The position/orientation of the block
# are supposed to be optimized in a preceding partial optimization run. In the
# vibrational analys, the block 'b1' is only allowed to vibrate as a whole. The
# number of resulting modes/frequencies is 3 for the fifth atom plus 6 for the
# block 'b1' (3 position/3 orientation), resulting in 9 frequencies in total.
# Since 6 of those frequencies are zero due to translational and rotational
# invariance of the system, one will find 3 non-zero characteristic frequencies
```

```
# in the output. In practice with ADF not exactly 6 zero's are found, but they
# are close to zero.

# The quality of the frequencies/modes depends largely on the block choice. Best
# results are obtained when grouping atoms in a block if those atoms are known
# to form rather rigid structures. For instance, grouping the 11 atoms of
# benzene side group into a block, will usually result in representative
# frequencies. In this example the block choice is only illustrative for the
# methodology.
```

Example: Raman: NH3

Download Freq_NH3_RAMAN.run

```
#! /bin/sh

# Summary:
# - Analytical frequencies with subsequent calculation of Raman intensities in a
#   range
# - Numerical frequencies including all Raman intensities


# == Raman Intensities for Selected Frequencies ==

# The RamanRange keyword can be used to calculate Raman intensities for a range
# of frequencies only. Using this option is a fast alternative for the existing
# method of calculating Raman intensities, which is described in the second part
# of this example.

# Two values defining an interval of frequencies to calculate the Raman
# intensities for. The Raman intensities are calculated by numerical
# differentiation of the polarizability tensor. Only frequencies frequencies
# withing the interval that are known to be Raman-active will be included.


$ADFBIN/adf <<eor
title NH3 frequencies with subsequent calculation of Raman intensities in the range 0-
↪2000 cm-1

atoms
   N      0.0000    0.0000    0.0000
   H      0.4729    0.8190    0.3821
   H     -0.9457    0.0000    0.3821
   H      0.4729   -0.8190    0.3821
end

Basis
 Type TZP
 Core Small
End

AnalyticalFreq
end

thermo   TMin=300 TMax=400

BeckeGrid
```

```
 Quality good
End

eor

mv TAPE21 NH3_freqs.t21

$ADFBIN/adf <<eor
title NH3 Raman intensities in the range 0-2000 cm-1

atoms
   N      0.0000    0.0000    0.0000
   H      0.4729    0.8190    0.3821
   H     -0.9457    0.0000    0.3821
   H      0.4729   -0.8190    0.3821
end

Restart
  File NH3_freqs.t21
End

Fragments
 H t21.H
 N t21.N
End

RamanRange 0.0 2000.0

thermo   TMin=300 TMax=400

BeckeGrid
 Quality good
End

eor


mv TAPE21 NH3_raman_range1.t21

$ADFBIN/adf <<eor
title NH3 Raman intensities in the range 2000-5000 cm-1

atoms
   N      0.0000    0.0000    0.0000
   H      0.4729    0.8190    0.3821
   H     -0.9457    0.0000    0.3821
   H      0.4729   -0.8190    0.3821
end

Restart
  File NH3_freqs.t21
End

Fragments
 H t21.H
 N t21.N
End
```

```
RamanRange 2000.0 5000.0

thermo  TMin=300 TMax=400

BeckeGrid
 Quality good
End

eor

mv TAPE21 NH3_raman_range2.t21


# == Raman Intensities for All Frequencies ==

# Raman scattering intensities and depolarization ratios for all molecular
# vibrations at a certain laser frequency can be calculated in a single run. The
# run type must be Frequencies and the RESPONSE key is used to specify that
# Raman intensities are computed.

# In this example the static Raman scattering is calculated (omega = 0). This
# type of calculation is very similar to an IR intensity calculation. In fact,
# all IR output is automatically generated as well. At all distorted geometries
# the dipole polarizability tensor is calculated. This is very time-consuming
# and is only feasible for small molecules.


$ADFBIN/adf <<eor
title NH3 frequencies with Raman intensities

atoms
   N      0.0000    0.0000    0.0000
   H      0.4729    0.8190    0.3821
   H     -0.9457    0.0000    0.3821
   H      0.4729   -0.8190    0.3821
end

Fragments
 H t21.H
 N t21.N
End

geometry
  frequencies
  End
end

response
  raman
end

thermo  TMin=300 TMax=400

BeckeGrid
 Quality good
End

eor
```

```
mv TAPE21 NH3_raman.t21
```

Example: Raman: HI

Download HI_Raman.run

```
# /bin/sh

# Example shows a calculation of the Raman spectrum of HI
# with a scalar relativistic ZORA and a spin-orbit coupled ZORA Hamiltonian.
# Note that used basis set is too small to get accurate results.

# == scalar relativistic ZORA ==

# 3 calculations are performed.
# First a scalar relativistic geometry optimization is performed
# and analytical frequencies are calculated. This calculation produces a restart file,
# which is used in the second and third alculation as input for the Raman calculation.
# The RamanRange keyword is used to calculate Raman intensities for a range of
→frequencies only.
# In the second calculation RESPONSE is used and in the third calculation AORESPONSE.

$ADFBIN/adf <<eor
BASIS
 Type DZ
 Core None
END
GEOMETRY
END
AnalyticalFreq
end
relativistic scalar zora
Atoms
 H  0.0000 0.0000 0.0000
 I  0.0000 0.0000 1.6092
End
NOPRINT SFO
eor
mv TAPE21 SR.t21
rm logfile

$ADFBIN/adf <<eor
restart
 file SR.t21
end
BASIS
 Type DZ
 Core None
END
RamanRange 0.0 4000.0
relativistic scalar zora
Atoms
 H  0.0000 0.0000 0.0000
 I  0.0000 0.0000 1.6092
End
response
```

```
 raman
 nfreq 1
 frqbeg 2.0
end
NOPRINT SFO
eor
rm TAPE21 logfile

$ADFBIN/adf <<eor
restart
 file SR.t21
end
BASIS
 Type DZ
 Core None
END
RamanRange 0.0 4000.0
relativistic scalar zora
Atoms
 H  0.0000 0.0000 0.0000
 I  0.0000 0.0000 1.6092
End
aoresponse
 frequency 1 2.0 eV
 raman
end
NOPRINT SFO
eor
rm TAPE21 logfile

# == Relativistic Spinorbit ZORA ==

# 4 calculations are performed.
# First a spin-orbit coupled relativistic geometry optimization is performed.
# This calculation produces a restart file which is used in the second calculation
# to calculate numerical frequencies. This calculation produces a restart file
# which is used in the third and fourth calculation as input for the Raman␣
→calculation.
# The RamanRange keyword is used to calculate Raman intensities for a range of␣
→frequencies only.
# In the third calculation RESPONSE is used and in the fourth calculation AORESPONSE.
# Symmetry NOSYM is needed for AORESPONSE icw spin-orbit coupling.

$ADFBIN/adf <<eor
BASIS
 Type DZ
 Core None
END
GEOMETRY
END
relativistic spinorbit zora
Atoms
H  0.0000 0.0000 0.0000
I  0.0000 0.0000 1.6092
End
NOPRINT SFO
eor
mv TAPE21 SOgeo.t21
```

```
rm logfile

$ADFBIN/adf <<eor
restart
 file SOgeo.t21
end
BASIS
 Type DZ
 Core None
END
GEOMETRY
 Frequencies Symm=True
END
relativistic spinorbit zora
Atoms
H  0.0000 0.0000 0.0000
I  0.0000 0.0000 1.6092
End
NOPRINT SFO
eor
mv TAPE21 SO.t21
rm logfile

$ADFBIN/adf <<eor
restart
 file SO.t21
end
BASIS
 Type DZ
 Core None
END
RamanRange 0.0 4000.0
relativistic spinorbit zora
Atoms
H  0.0000 0.0000 0.0000
I  0.0000 0.0000 1.6092
End
response
 raman
 nfreq 1
 frqbeg 2.0
end
NOPRINT SFO
eor
rm TAPE21 logfile

$ADFBIN/adf <<eor
restart
 file SO.t21
end
BASIS
 Type DZ
 Core None
END
symmetry nosym
RamanRange 0.0 4000.0
relativistic spinorbit zora
Atoms
```

```
H  0.0000 0.0000 0.0000
I  0.0000 0.0000 1.6092
End
aoresponse
 frequency 1 2.0 eV
 raman
end
NOPRINT SFO
eor
rm TAPE21 logfile
```

Example: Resonance Raman, excited state finite lifetime: HF

Download HF_ResonanceRaman.run

```
# /bin/sh

# Example shows a calculation of the Resonance Raman spectrum (RRS) of HF. In
# this example the RRS is calculated from the geometrical derivatives of the
# frequency-dependent polarizability, including a finite lifetime.

# In the ADF input one then needs to include the subkey FREQUENCIES of the key
# GEOMETRY (numerical frequencies) and include the subkeys RAMAN and LIFETIME of
# the key AORESPONSE.

# Note that used basis set is too small to get accurate results.

$ADFBIN/adf <<eor
title HF ao-raman

BASIS
 F DZP/F
 H DZP/H
END

GEOMETRY
 Frequencies
 End
END

Symmetry NOSYM

NumericalQuality Good

Atoms
 H 0.0000 0.0000 0.0000
 F 0.0000 0.0000 0.9170
End

aoresponse
 frequency 1 0.52362 Hartree
 lifetime 0.0034
 raman
end

NOPRINT SFO
```

```
eor
```

Example: Resonance Raman, excited state gradient: Formaldehyde

Download Vibron_RR_form.run

```
#! /bin/sh

# Example shows a calculation of the Resonance Raman spectrum (RRS) of
# formaldehyde. In this example the RRS is calculated using the excited-state
# gradient.

# First the frequency calculation is performed, which produces a restart file
# 'restart.freq.t21' used as input in the resonance Raman calculation.

# Next the resonance Raman calculation is performed by setting the 'VIBRON'
# subkey in the 'GEOMETRY' block key, including both the 'EXCITATION' block key
# and the 'VIBRON' block key. The mode number 3 (in the DOMODES sub-key)
# corresponds to the C-O bond stretching.


"$ADFBIN/adf" <<eor
ATOMS
1 C     -0.341709656800     -1.442555818000     -0.0000000
2 O     -0.400432132900     -0.199067921400     -0.0000000
3 H     -1.260200076000     -2.063957146000     -0.0000000
4 H      0.631252737900     -1.974635173000     -0.0000000
END

BASIS
type DZ
core None
createoutput None
END

XC
GGA Becke Perdew
END
Symmetry NOSYM

AnalyticalFreq
END


eor

mv TAPE21 restart.freq.t21

"$ADFBIN/adf" <<eor
ATOMS
1 C     -0.341709656800     -1.442555818000     -0.0000000
2 O     -0.400432132900     -0.199067921400     -0.0000000
3 H     -1.260200076000     -2.063957146000     -0.0000000
4 H      0.631252737900     -1.974635173000     -0.0000000
END
```

```
BASIS
type DZ
core None
createoutput None
END

XC
GGA Becke Perdew
END
Symmetry NOSYM

GEOMETRY
  VIBRON
END

EXCITATIONS
  ONLYSING
  LOWEST 10
END

VIBRON
  NMTAPE restart.freq.t21
  RESRAMAN
  STPSIZE 0.1
  DOMODES 3
  DSCHEME ELCHAR
END

eor

# Note that used basis set is too small to get accurate results.
```

Example: Vibrational Raman optical activity (VROA): H2O2

Download VROA.run

```
#! /bin/sh

# Example shows a calculation of normal vibrational Raman optical activity.

# In the ADF input one then needs to include the subkey FREQUENCIES of the key
# GEOMETRY (numerical frequencies) and include the subkey VROA of the key
# AORESPONSE. A laser frequency need to be added. The other keys in AORESPONSE
# are recommended by the author of the implementation of VROA in ADF. For
# accuracy reasons 'NumericalQuality Good' and EXACTDENSITY are used. Note that
# the subkey FitAOderiv is only used in case of pair fitting (STOFIT), in which
# case it will improve the fit quality. The default fit method is ZlmFit, which
# will not use this key. In case of ZlmFit one should set the numericalquality
# or ZlmFit quality to a high level, for example Good or VeryGood.

$ADFBIN/adf <<eor
title VROA TEST H2O2

basis
 type TZP
 core None
end
```

```
ATOMS
    1.O        -0.750254   -0.034490    0.015133
    2.O         0.750254    0.034490    0.015133
    3.H        -0.943532    0.744006    0.580040
    4.H         0.943532   -0.744006    0.580040
END

xc
 gga BLYP
end

SCF
  Converge 1.0e-10
END

GEOMETRY
frequencies
End
END
exactdensity

NumericalQuality Good

aoresponse
 VROA
 scf  converge 1d-6 iterations 100
 frequency 1 5145 Angstrom
 ALDA
 FitAOderiv
 EL_DIPOLE_EL_DIPOLE VELOCITY
 EL_DIPOLE_EL_QUADRUPOLE VELOCITY
 EL_DIPOLE_MAG_DIPOLE VELOCITY
end

eor

# Note that used basis set is too small to get accurate results. Better is the
# use larger basis sets, like one of the even tempered basis sets (for example
# Type ET/ET-QZ3P-1DIFFUSE), or use augmented basis sets (for example Type
# AUG/ATZ2P).
```

Example: Resonance VROA: H2O2

Download VROA_RESO.run

```
#! /bin/sh

# Example shows a calculation of resonance vibrational Raman optical activity
# (resonance VROA).

# In the ADF input one then needs to include the subkey FREQUENCIES of the key
# GEOMETRY (numerical frequencies) and include the subkeys VROA and LIFETIME of
# the key AORESPONSE. A laser frequency need to be added. The other keys in
# AORESPONSE are recommended by the author of the implementation of VROA in ADF.
# For accuracy reasons 'NumericalQuality Good' and EXACTDENSITY are used. Note
# that the subkey FitAOderiv is only used in case of pair fitting (STOFIT), in
```

```
# which case it will improve the fit quality. The default fit method is ZlmFit,
# which will not use this key. In case of ZlmFit one should set the
# numericalquality or ZlmFit quality to a high level, for example Good or
# VeryGood.

$ADFBIN/adf <<eor
title VROA-RESO TEST H2O2

basis
 type TZP
 core None
end

ATOMS
    1.O       -0.750254   -0.034490    0.015133
    2.O        0.750254    0.034490    0.015133
    3.H       -0.943532    0.744006    0.580040
    4.H        0.943532   -0.744006    0.580040
END

xc
 gga BLYP
end

SCF
  Converge 1.0e-10
END

GEOMETRY
frequencies
End
END
exactdensity

NumericalQuality Good

aoresponse
 VROA
 scf   converge 1d-6 iterations 100
 frequency 1 5.15462 eV
 lifetime 0.0037
 ALDA
 FitAOderiv
 EL_DIPOLE_EL_DIPOLE VELOCITY
 EL_DIPOLE_EL_QUADRUPOLE VELOCITY
 EL_DIPOLE_MAG_DIPOLE VELOCITY
end

eor

# Note that used basis set is too small to get accurate results. Better is the
# use larger basis sets, like one of the even tempered basis sets (for example
# Type ET/ET-QZ3P-1DIFFUSE), or use augmented basis sets (for example Type
# AUG/ATZ2P).
```

Example: Vibrational Circular Dichroism (VCD): NHDT

Download VCD_COG_NHDT.run

```
#! /bin/sh

# Analytical frequencies with subsequent calculation of vibrational circular
# dichroism (VCD)

# The VCD keyword can be used to calculate VCD spectra. It is important to note
# that the VCD keyword only works in combination with the keys AnalyticalFreq
# and symmetry NOSYM.

# Recommended is use to use high accuracy for the geometry optimization which
# one needs to do before the frequency calculation. This simple example is an
# NHDT molecule, which is NH3 where one hydrogen atom is replaced with deuterium
# and another with tritium.

$ADFBIN/adf <<eor
Atoms
    N          0.000000    0.000000    0.010272
    H         -0.471582   -0.816803    0.407861
    H.D        0.943163    0.000000    0.407861
    H.T       -0.471582    0.816803    0.407861
End

AtomProps
H.D m=2.014101778
H.T m=3.01604927
End

Basis
  Type TZP
  Core None
End

Symmetry NOSYM

xc
 GGA BP86
end

BeckeGrid
 quality verygood
End

AnalyticalFreq
End

VCD

eor
```

Example: Franck-Condon Factors: NO2

Download FranckCondon_NO2.run

```
#!/bin/sh

# As an example of a Franck-Condon calculation, lets look at the transition of
# NO2 to NO2 - . NO2 is a small molecule with only three vibrational modes.
# Putting an extra electron on the molecule will cause a big displacement,
# resulting in large electron-phonon couplings.

# In general, the larger the molecule, the smaller the displacement and hence
# the electron-phonon couplings and Franck-Condon factors. Moreover, larger
# molecules have more vibrational modes, meaning that the already smaller
# displacement will generally be smeared out over more modes, resulting in an
# additional decrease in electron-phonon couplings. This is fortunate, since the
# number of Franck-Condon factors increases factorially with the number of
# vibrational modes, making it prohibitively expensive to take more than a few
# vibrational quanta into account for most molecules.

# In order to calculate the Franck-Condon factors for Nitrite and Nitrogen
# dioxide, the equilibrium positions of the nuclei and the vibrational modes
# have to be obtained:


$ADFBIN/adf <<eor
TITLE Nitrogen dioxide
ATOMS
    N         0.000000    0.000000   -0.016179
    O         0.000000    1.098646   -0.492918
    O         0.000000   -1.098646   -0.492918
END
BASIS
    CORE NONE
    TYPE DZP
END
XC
    LDA SCF VWN
END
ANALYTICALFREQ
END
UNRESTRICTED
CHARGE 0 1
GEOMETRY
 Converge grad=1.0e-5
END
eor

mv TAPE21 NO2.t21
rm logfile

# We are using an already optimized geometry for the second calculation but in a
# real experiment one should run geometry optimization first

$ADFBIN/adf <<eor
TITLE Nitrite
ATOMS
    N         0.000000    0.000000    0.126041
    O         0.000000    1.070642   -0.555172
    O         0.000000   -1.070642   -0.555172
END
CHARGE -1.0
```

```
BASIS
    CORE NONE
    TYPE DZP
END
XC
    LDA SCF VWN
END
! Uncomment the 2 lines below if you copy this example
!GEOMETRY
!END

ANALYTICALFREQ
END
eor

mv TAPE21 NO2-.t21
rm logfile


# This runscript produces two TAPE21 files containing the frequencies and the
# normal modes for both charge states. Lets first look at the ground state to
# ground state overlap:


$ADFBIN/fcf <<eor
STATES NO2.t21 NO2-.t21
QUANTA 0 0
TRANSLATE
ROTATE
eor

rm TAPE61 logfile

# Here, zero vibrational quanta are specified for both charge states, which
# corresponds to the vibrational ground state. Looking at the standard output,
# we see for NO2 :

# =================================================
# | Frequency (cm^-1 ) |  lambda (dimensionless) |
# | 756                | 1.979                   |
# | 1380               | 1.489                   |
# | 1716               | 0.000                   |
# =================================================

# And for NO_2^- :

# =================================================
# | Frequency (cm^-1 ) |  lambda (dimensionless) |
# | 785                | 1.552                   |
# | 1265               | 0.000                   |
# | 1338               | 2.231                   |
# =================================================

# Both states have two vibrational modes with a significant electron-phonon
# coupling. The ground state to ground state Franck-Condon factor is therefore
# expected to be quite small. And indeed, looking at the output, we see that it
# is 0.349*10^-1 , less than four percent of the total.
```

```
# Since NO2 has only three vibrational modes, many quanta can be included, and
# this indeed turns out to be necessary. Setting the maximum number of quanta at
# 20 results in 1771 permutations for both states and a total of 3136441 Franck-
# Condon factors. Even with so many factors, the average sum is still only
# 0.463. Including one extra vibrational quanta results in an additional 960135
# Franck-Condon factors, but an average sum of only 0.473, i.e. a percent more.
# This one percent is smeared out over so many factors that their individual
# contributions become negligible.


$ADFBIN/fcf <<eor
STATES NO2.t21 NO2-.t21
QUANTA 20 20
TRANSLATE
ROTATE
eor

rm TAPE61 logfile

$ADFBIN/fcf <<eor
STATES NO2.t21 NO2-.t21
QUANTA 21 21
TRANSLATE
ROTATE
eor

rm TAPE61 logfile

$ADFBIN/fcf <<eor
STATES NO2.t21 NO2-.t21
QUANTA 0 20
TRANSLATE
ROTATE
SPECTRUM 0e3 20e3 1001
eor

rm TAPE61 logfile
```

14.4.2 Excitation energies: UV/Vis spectra, X-ray absorption, CD, MCD

Example: Excitation energies and polarizability: Au2

Download Au2_Resp.run

```
#! /bin/sh

# A calculation of response properties of the Au2 dimer, with ZORA relativistic
# corrections

# In the response module infinite symmetries cannot be handled (see the User's
# Guide), so we specify a lower subgroup in the input file, here D(8h).

# In this sample run the LB94 potential is used. The implementation implies that
# the XC potential is evaluated from the exact charge density, rather than the
# cheaper and faster fitted density (as is the case for other XC functionals).
# This means that the computation times are longer. In a small molecule like Au2
```

```
# this hardly shows, but in larger molecules the differences may be more
# significant. Note that the LB94 is a model potential, thus the calculated bond
# energy has not so much meaning. If the BASIS key is used and the model
# potential LB94 in the create run of the atoms the BP86 functional is used.

# Excitation energies are computed, in principle the lowest 10 in each irrep of
# the symmetry (see, however, the User's Guide).


$ADFBIN/adf <<eor
Title Au2, Response Properties

XC
  GGA  LB94
End

Relativistic Scalar ZORA

Atoms
  Au   0.0 0.0 1.236
  Au   0.0 0.0 -1.236
End

Basis
 Au ZORA/DZ/Au.4d
End

Symmetry D(8h)

Excitations
  Lowest 10
  TOLERANCE 1e-10
End

Response
  AllComponents
End

eor

mv TAPE21 Au2.t21
```

Example: Excitation energies open shell molecule: CN

```
Download CN_unr_exci.run
```

```
#!/bin/sh

# Calculation of the excitation energies of the open shell molecule CN

# In this example, the lowest 20 excitation energies of CN are calculated in a
# spin-unrestricted TDDFT calculation. In the MO -> MO transitions part for the
# excitations of the output file, the spin of each molecular orbitals are also
# specified to help assign the spin state of the excited states. The transitions
# are always from alpha spin-orbital to alpha spin-orbital or from beta spin-
# orbital to beta spin-orbital.
```

```
$ADFBIN/adf <<eor
Title excitation energies of CN

Atoms
 C  .0000  .0000  .0000
 N  .0000  .0000 1.1718
End

unrestricted
charge 0 1

excitations
 lowest 20
 nto
 sfoanalysis
end

Basis
 Type AUG/ADZP
End

eor
rm TAPE21 logfile


# Next the same example for CN is given with the Tamm-Dancoff approximation
# (TDA) approximation (including TDA in the input). Due to this approximation
# the calculated excitation energies will not be exactly the same as in the
# first example.

$ADFBIN/adf <<eor
Title excitation energies (TDA) of CN

Atoms
 C  .0000  .0000  .0000
 N  .0000  .0000 1.1718
End

unrestricted
charge 0 1

excitations
 lowest 20
 nto
 sfoanalysis
end

TDA

Basis
 Type AUG/ADZP
End

eor
rm TAPE21 logfile


# Note that the basis set is still far from complete, the ATZ2P is better. For
```

```
# accuracy reasons one may need to increase the numerical quality. If one uses
# the augmented basis sets for accuracy reasons one could use NumericalQuality
# Good.

# The third calculation is the calculation of spin-flip excitation energies for
# CN. Again these energies will not be exactly the same as in the first example.
# For open-shell molecules, spin-flip transition can result in transition to the
# ground state with a different S_z value, while the symmetry of the transition
# density is A1 (Sigma+ for linear molecules). The excitation energy of this
# transition should be zero and this can be used to test the reliability of
# spin-flip TDDFT. Indeed the calculation of the spin-flip excitation energies
# of CN shows one value which is close to zero and has a transition density of
# Sigma+ symmetry.


$ADFBIN/adf <<eor
Title spin-flip excitation energies (TDA) of CN

Atoms
 C  .0000  .0000  .0000
 N  .0000  .0000 1.1718
End

unrestricted
charge 0 1

excitations
 lowest 20
 nto
 sfoanalysis
end

SFTDDFT
TDA

Basis
 Type AUG/ADZP
End

eor

mv TAPE21 CN.t21
rm logfile
```

Example: Spin-flip excitation energies: SiH2

```
Download SiH2_spinflip.run
```

```
#!/bin/sh

# Calculation of the spin-flip excitation energies of the open shell molecule SiH2

$ADFBIN/adf <<eor
Title spin-flip excitation energies of SiH2
Atoms Zmatrix
 Si 0  0  0
 H  1  0  0  1.5145
```

```
 H  1  2  0  1.5145  92.68
End

excitations
  lowest 20
end

unrestricted
charge 0 2

SFTDDFT
FORCEALDA
TDA

Basis
 Type TZ2P
End

eor
mv TAPE21 SiH2_spinflip.t21
rm logfile

# In this example, the lowest 20 spin-flip excitation energies of SiH2 are
# calculated in a spin-unrestricted TDDFT calculation.

# In this case an excited state is used as reference, which means that there can
# also be a negative excitation energy, which is indeed the case. The electron
# configuration used in the SCF is (a1)^1 (b1)^1, with Sz=1, thus a ^3B_1
# state, which is an excited state. The ^1A_1 state with electron configuration
# (a1)^2 is lower in energy, and is the ground state.

# There is also an excited 1A1 state with electron configuration (b1)^2. The
# transition from the ground 1A1 state to the excited 1A1 state is an excitation
# from the electron configuration (a1)^2 to (b1)^2. This transition is actually
# a double excitation, which means that some double excitations can be reached
# using spin-flip TDDFT with carefully selected reference states.

# In the MO -> MO transitions part for the excitations of the output file, the
# spin of each molecular orbitals are also specified to help assign the spin
# state of the excited states. Note that in these spin-flip calculations the
# transitions are always from alpha spin-orbital to beta or from beta spin-
# orbital to alpha spin-orbital.

# For open-shell molecules, spin-flip transition can result in transition to the
# ground state with a different Sz value, while the symmetry of the transition
# density is A1. The excitation energy of this transition should be zero and
# this can be used to test the reliability of spin-flip TDDFT. Indeed the
# calculation of the spin-flip excitation energies of SiH2 shows one value which
# is close to zero and has a transition density of A1 symmetry.

# The 1A1 state with electron configuration (a1)^2 can also be used in the
# calculation of the excitation energies. This is a closed shell configuration,
# in which case we do not need the spin-flip method.


$ADFBIN/adf <<eor
Title excitation energies of SiH2
Atoms Zmatrix
```

```
 Si 0  0  0
 H  1  0  0  1.5145
 H  1  2  0  1.5145  92.68
End

excitations
  lowest 20
end

Basis
 Type TZ2P
End

eor
mv TAPE21 SiH2_excit.t21
rm logfile

# The transition from the ground ^1A_1 state to the excited ^1A_1 state, which
# is an excitation from the electron configuration (a1)^2 to (b1)^2, can not be
# reached in this calculation, since it has mainly double excitation character.
# Of course, other excited ^1A_1 states can be reached.
```

Example: TDHF excitation energies: N2

Download N2_TDHF.run

```
#! /bin/sh

# Calculation of the excitation energies of N2 using time-dependent Hartree-Fock
# (TDHF). It also shows the possibility to use the Tamm-Dancoff approximation
# (TDA). This examples consists of 4 calculations:

# RPA non-relativistic and spin-orbit coupled
# TDA non-relativistic and spin-orbit coupled (CIS)


# == RPA non-relativistic ==


$ADFBIN/adf <<eor
Atoms
N 0 0 0
N 0 0 1.0977
End

XC
 hartreefock
end

Basis
 Type DZ
 Core None
End

BeckeGrid
 Quality Good
End
```

```
excitations
 lowest 5
end
eor

rm TAPE21 logfile


# == RPA spin-orbit coupled ==


$ADFBIN/adf <<eor
Atoms
N 0 0 0
N 0 0 1.0977
End

XC
 hartreefock
end

Basis
 Type DZ
 Core None
End

BeckeGrid
 Quality Good
End

excitations
 lowest 20
end
relativistic spinorbit zora
eor

rm TAPE21 logfile


# == TDA non-relativistic ==


$ADFBIN/adf <<eor
Atoms
N 0 0 0
N 0 0 1.0977
End

XC
 hartreefock
end

Basis
 Type DZ
 Core None
End
```

```
BeckeGrid
 Quality Good
End

excitations
 lowest 5
end
TDA
eor

rm TAPE21 logfile


# == TDA spin-orbit coupled ==


$ADFBIN/adf <<eor
Atoms
N 0 0 0
N 0 0 1.0977
End

XC
 hartreefock
end

Basis
 Type DZ
 Core None
End

BeckeGrid
 Quality Good
End

excitations
 lowest 20
end
TDA
relativistic spinorbit zora
eor

rm TAPE21 logfile
```

Example: excitation energies CAM-B3LYP: Pyridine

Download `Pyridine_CAM-B3LYP.run`

```
#! /bin/sh

# Calculation of the excitation energies of Pyridine using the range-separated
# functional CAM-B3LYP. First the atoms are calculated. To calculate this range-
# separated functional CAM-B3LYP the RIHARTREEFOCK method is required, and LibXC
# is needed. Note that CAM-B3LYP is not the same as CAMY-B3LYP. For speed
# reasons the atoms are calculated separately, such that these calculations
# might run in parallel. If the BASIS key is used the atoms are calculated
# serially. For the LibXC range separated functionals, like CAM-B3LYP, starting
```

```
# from ADF2016.102 the kernel consists of range separated ALDA plus the kernel
# of the range separated exact exchange part. In ADF2016.101 the kernel for
# LibXC range separated functionals, like CAM-B3LYP, was using a 100% ALDA plus
# range separated exact exchange kernel (the ALDA part was not range-separated
# corrected).

for n in H C N
do
$ADFBIN/adf <<eor
Create $n file=$ADFRESOURCES/DZP/$n
XC
 LibXC CAM-B3LYP
End
eor
mv TAPE21 $n.t21
rm logfile
done

# Next the excitation energies of Pyridine are calculated.

$ADFBIN/adf <<eor
Atoms
 N      0.00000000      0.00000000      1.07112000
 C      0.00000000      0.00000000      3.88763000
 C      0.00000000     -1.14432000      1.77593000
 C      0.00000000      1.14432000      1.77593000
 C      0.00000000      1.19947000      3.17299000
 C      0.00000000     -1.19947000      3.17299000
 H      0.00000000     -2.16297000      3.68508000
 H      0.00000000     -2.06734000      1.18970000
 H      0.00000000      0.00000000      4.97897000
 H      0.00000000      2.16297000      3.68508000
 H      0.00000000      2.06734000      1.18970000
End
Fragments
 H H.t21
 C C.t21
 N N.t21
End
excitations
 onlysing
end
XC
 LibXC CAM-B3LYP
End
eor
```

Example: CAMY-B3LYP excitation energies: H2O

Download CAMY-B3LYP_TDDFT_H2O.run

```
#! /bin/sh


# Example for the range separated functional CAMY-B3LYP for the calculation of
# excitation energies of water. Note that CAMY-B3LYP is not the same as CAM-
# B3LYP.
```

```
$ADFBIN/adf <<eor
   TITLE  H2O excitations with CAMY-B3LYP functional

   ATOMS
      O      0.000000    0.000000    0.000000
      H      0.000000    0.500000   -0.6
      H      0.000000   -0.500000   -0.6
   END

   NOPRINT BAS FIT FUNCTIONS SFO

   XC
      HYBRID CAMY-B3LYP
      xcfun
      RANGESEP GAMMA=0.34
   END

   BeckeGrid
      Quality good
   End

   EXCITATIONS
      LOWEST 10
      ONLYSING
   END

   Basis
      Type DZP
   END

eor
```

Example: Full XC kernel in excitation energy calculation: H2O+

Download XCFUN_FULLKERNEL_TDDFT_H2O.run

```
#! /bin/sh

# Example of using the full XC kernel in the calculation of excitation energies,
# in this case for the BP86 functional. Note that SYMMETRY NOSYM is required and
# XCFUN should be added to the XC block key. In this case spin-flip excitations
# are calculated.

$ADFBIN/adf <<eor
TITLE  H2O+ spin-flip TDDFT with BP and the non-ALDA kernel

ATOMS
   O      0.000000    0.000000    0.000631
   H      0.000000    0.771769   -0.605006
   H      0.000000   -0.771769   -0.605006
END

NOPRINT BAS FIT FUNCTIONS SFO

UNRESTRICTED
```

```
CHARGE 1 1

SFTDDFT
TDA

SYMMETRY NOSYM

XC
 GGA Becke Perdew
 xcfun
END

BeckeGrid
 quality good
End

EXCITATIONS
 LOWEST 5
 FULLKERNEL
END

Basis
 Type DZP
END

eor

rm TAPE21 logfile
```

Example: Use of xcfun in excitation energy calculations: H2O

Download XCFUN_TDDFT_H2O.run

```
#! /bin/sh

# Example for the use of the xcfun library for a calculation of excitation
# energies, in this case for the BP86 functional. Note that for closed shell
# molecules only singlet-singlet can be calculated with xcfun, therefore the
# subkey ONLYSING is used.

$ADFBIN/adf <<eor
TITLE  H2O TDDFT with XCFUN

ATOMS
   O        0.000000    0.000000    0.000631
   H        0.000000    0.771769   -0.605006
   H        0.000000   -0.771769   -0.605006
END

NOPRINT BAS FIT FUNCTIONS SFO

XC
 GGA Becke Perdew
 xcfun
END
```

```
BeckeGrid
 quality good
End

EXCITATIONS
 LOWEST 5
 ONLYSING
END

Basis
 Type DZP
END

eor

rm TAPE21 logfile
```

Example: Core excitation energies: TiCl4

Download TiCl4_CoreExci.run

```
#! /bin/sh

# Calculation of the 2p Ti and 2p Cl core excitation energies of TiCl4

# The state selective method (key SELECTEXCITATION) can be used to calculate
# core excitation energies. The use of the key SELECTEXCITATION is similar as
# the use of the key MODIFYEXCITATION. However, the key SELECTEXCITATION can not
# be used in combination with the key MODIFYEXCITATION. In the state selective
# method (key SELECTEXCITATION) the one-electron excited state configuration
# space remains complete, whereas it is reduced in case the scheme with the
# MODIFYEXCITATION key.

$ADFBIN/adf <<EOR
Title TiCl4 TD-DFT spinorbit relativistic 2p Cl core excitations single orbital␣
↪transitions

Units
 LENGTH BOHR
End

Atoms
 Ti 0.        0.        0.
 Cl    2.36754    2.36754    2.36754
 Cl   -2.36754   -2.36754    2.36754
 Cl    2.36754   -2.36754   -2.36754
 Cl   -2.36754    2.36754   -2.36754
End
SYMMETRY T(D)

EPRINT
 eigval 1000 1000
End

XC
 GGA LB94
End
```

```
relativistic spinorbit zora

SelectExcitation
 UseOccRange -20.0 -16.0
 UseScaledZORA
END
Excitations
 SingleOrbTrans
 allowed
 lowest 40
 kfwrite 0
End

Basis
 Type DZ
 Core None
End

BeckeGrid
 Quality good
End

EOR

rm TAPE21
rm logfile


# == An example with the key MODIFYEXCITATION ==

# In this example, the 12 lowest singlet-singlet excitation energies of T2
# symmetry are calculated, the dipole allowed excitations. This can also be
# achieved using the ALLOWED subkey in the key Excitation.

# In this example only excitations from the 2t2 -orbital are included (see the
# key MODIFYEXCITATION), an almost pure 2p core orbital of titanium. The orbital
# energies of the uninteresting other occupied orbitals are artificially changed
# to a large negative value (-1d6 hartree).


$ADFBIN/adf <<EOR
Title TiCl4 TD-DFT scalar relativistic 2p Ti core excitations

Units
 LENGTH BOHR
End

Atoms
 Ti 0.        0.         0.
 Cl    2.36754     2.36754      2.36754
 Cl   -2.36754    -2.36754      2.36754
 Cl    2.36754    -2.36754     -2.36754
 Cl   -2.36754     2.36754     -2.36754
End
SYMMETRY T(D)

EPRINT
```

```
 eigval 1000 1000
End

XC
 GGA LB94
End

relativistic scalar zora

ModifyExcitation
 UseOccupied
  T2 2
 SubEnd
 UseScaledZORA
END

Excitations
 ONLYSING
 davidson
  T2 12
 SubEnd
End

Basis
 Type DZ
 Core None
End

BeckeGrid
 Quality good
End

EOR

rm TAPE21
rm logfile


# In this example the 2p Cl core excitation energies of TiCl4 are calculated.

# In this example only excitations from occupied orbitals are considered which
# have orbital energies between -8 and -6 hartree, namely the 5a1, 1e, 1t1,
# 4t2, and 5t2 orbitals, which are almost pure 2p core orbitals of chlorine.
# The orbital energies of the uninteresting other occupied orbitals are again
# artificially changed to a large negative value (-1d6 hartree).

$ADFBIN/adf <<EOR
Title TiCl4 TD-DFT scalar relativistic 2p Cl core excitations

Units
 LENGTH BOHR
End

Atoms
 Ti 0.        0.        0.
 Cl   2.36754    2.36754    2.36754
 Cl  -2.36754   -2.36754    2.36754
 Cl   2.36754   -2.36754   -2.36754
```

```
 Cl    -2.36754     2.36754    -2.36754
End
SYMMETRY T(D)

EPRINT
 eigval 1000 1000
End

XC
 GGA LB94
End

relativistic scalar zora

ModifyExcitation
 UseOccRange -8.0 -6.0
 UseScaledZORA
END

Excitations
 allowed
 lowest 50
End

Basis
 Type DZ
 Core None
End

BeckeGrid
 Quality good
End

EOR

rm TAPE21
rm logfile


# Another possibility is the use of the subkey OccVirtRange:

$ADFBIN/adf <<EOR
Title TiCl4 TD-DFT scalar relativistic 2p Cl core excitations

Units
 LENGTH BOHR
End

Atoms
 Ti 0.        0.         0.
 Cl    2.36754     2.36754     2.36754
 Cl   -2.36754    -2.36754     2.36754
 Cl    2.36754    -2.36754    -2.36754
 Cl   -2.36754     2.36754    -2.36754
End
SYMMETRY T(D)

EPRINT
```

```
 eigval 1000 1000
End

XC
 GGA LB94
End

relativistic scalar zora

ModifyExcitation
 UseOccVirtRange 7.0 100.0
 UseScaledZORA
END

Excitations
 allowed
 lowest 50
End

Basis
 Type DZ
 Core None
End

BeckeGrid
 Quality good
End

EOR

rm TAPE21
rm logfile



$ADFBIN/adf <<EOR
Title TiCl4 TD-DFT scalar relativistic 2p Ti core excitations

Units
 LENGTH BOHR
End

Atoms
 Ti 0.        0.         0.
 Cl    2.36754     2.36754     2.36754
 Cl   -2.36754    -2.36754     2.36754
 Cl    2.36754    -2.36754    -2.36754
 Cl   -2.36754     2.36754    -2.36754
End
SYMMETRY T(D)

EPRINT
 eigval 1000 1000
End

XC
 GGA LB94
End
```

```
relativistic scalar zora

SelectExcitation
 UseOccupied
  T2 2
 SubEnd
 UseScaledZORA
END

Excitations
 ONLYSING
 davidson
  T2 12
 SubEnd
End

Basis
 Type DZ
 Core None
End

BeckeGrid
 Quality good
End

EOR

rm TAPE21
rm logfile

$ADFBIN/adf <<EOR
Title TiCl4 TD-DFT scalar relativistic 2p Cl core excitations

Units
 LENGTH BOHR
End

Atoms
 Ti 0.         0.          0.
 Cl    2.36754     2.36754     2.36754
 Cl   -2.36754    -2.36754     2.36754
 Cl    2.36754    -2.36754    -2.36754
 Cl   -2.36754     2.36754    -2.36754
End
SYMMETRY T(D)

EPRINT
 eigval 1000 1000
End

XC
 GGA LB94
End

relativistic scalar zora

SelectExcitation
```

```
 UseOccVirtRange 7.0 100.0
 UseScaledZORA
END

Excitations
 allowed
 lowest 25
End

Basis
 Type DZ
 Core None
End

BeckeGrid
 Quality good
End

EOR


rm TAPE21
rm logfile
```

Example: X-Ray Absorption and Emission Quadrupole Oscillator strengths at the Cl K-edge: TiCl4

Download TiCl4_XASXES.run

```
#! /bin/sh

# Calculation of the 1s Cl core excitation energies of TiCl4. Both X-Ray
# Emission (XES keyword) and X-Ray absorption (XAS keyword in EXCITATION key)
# are calculated. The individual (zeroth and higher order) oscillator strength
# components to the total oscillator strengths will be printed
# (ALLXESQUADRUPOLE, ALLXASQUADRUPOLE).

# In order to calculate a 1s-orbital that is localized on one of the Cl atoms, a
# frozen core approach is used on the other Cl atoms, such that there is only
# one Cl 1s-orbital, which will remain localized in the molecule.

# In order to get information of the population analysis on all core orbitals, a
# large negative value was included as argument for the orbpoper subkey of the
# block key EPRINT. This orbital analysis shows that the 2 A1 orbital is indeed
# the 1s Cl orbital, see subkey corehole of block key XES, and subkey
# useoccupied of block key MODIFYEXCITATION.

"$ADFBIN/adf" <<eor
TITLE TiCl4 X-Ray Absorption and Emission Quadrupole Oscillator strengths at the Cl K-
→edge

ATOMS
    1  Ti            0.0000    0.0000    0.0000
    2  Cl            0.0000    0.0000    2.2930
    3  Cl.1          1.0809   -1.8722   -0.7643
    4  Cl.1         -2.1618    0.0000   -0.7643
    5  Cl.1          1.0809    1.8722   -0.7643
END
```

```
SYMMETRY C(3V)

BASIS
type TZP
core None
createoutput None
Cl.1 $ADFRESOURCES/TZP/Cl.2p
END

XC
GGA Becke Perdew
END

EPRINT
 eigval 1000 1000
 orbpoper -10000 100
End

RELATIVISTIC Scalar ZORA

SCF
iterations 200
END

XES
COREHOLE A1 2
ALLXESQUADRUPOLE
END

MODIFYEXCITATION
UseOccupied
A1 2
SubEnd
END

EXCITATIONS
davidson
End
ONLYSING
lowest 10
XAS
ALLXASQUADRUPOLE
END

eor

mv TAPE21 restart.t21
rm logfile


# In the second example, to test gauge dependency, the molecule is shifted 100
# Angstrom in the z-direction.

"$ADFBIN/adf" <<eor
TITLE TiCl4 X-Ray Absorption and Emission Quadrupole Oscillator strengths at the Cl K-
↪edge shifted 100Ang in z-direction

ATOMS
```

```
    1  Ti               0.0000    0.0000   100.0000
    2  Cl               0.0000    0.0000   102.2930
    3  Cl.1             1.0809   -1.8722    99.2357
    4  Cl.1            -2.1618    0.0000    99.2357
    5  Cl.1             1.0809    1.8722    99.2357
END

SYMMETRY C(3V)

BASIS
type TZP
core None
createoutput None
Cl.1 $ADFRESOURCES/TZP/Cl.2p
END

XC
GGA Becke Perdew
END

EPRINT
 eigval 1000 1000
 orbpoper -10000 100
End

RELATIVISTIC Scalar ZORA

SCF
iterations 200
END

XES
COREHOLE A1 2
ALLXESQUADRUPOLE
END

MODIFYEXCITATION
UseOccupied
A1 2
SubEnd
END

EXCITATIONS
davidson
End
ONLYSING
lowest 10
XAS
ALLXASQUADRUPOLE
END

RESTART
  File restart.t21
  noGEO True
  noEXC True
END

eor
```

```
rm TAPE21
rm restart.t21
rm logfile
```

Example: (Core) Excitation energies including spin-orbit coupling: Ne

Download Ne_exciso.run

```
#! /bin/sh


# Calculation of the (core) excitation energies of Ne including spin-orbit
# coupling.

# The state selective method (key SELECTEXCITATION) can be used to reduce the
# computational costs of, for example, core excitation energies. In this scheme
# a guess vector for the orbital transition has to be provided. It should be
# used in combination with the davidson method to calculate excitation energies.
# An overlap criterion is used to follow the wanted eigenvector. This key can
# also be used in case of spin-orbit coupling. The use of the key
# SELECTEXCITATION is similar as the use of the key MODIFYEXCITATION. However,
# the key SELECTEXCITATION can not be used in combination with the key
# MODIFYEXCITATION. In the state selective method (key SELECTEXCITATION) the
# one-electron excited state configuration space remains complete, whereas it is
# reduced in case the scheme with the MODIFYEXCITATION key.


$ADFBIN/adf  <<eor
Title Ne

Atoms
  Ne .0000  .0000  0.0000
End

Basis
 Type QZ4P
End

relativistic scalar zora

symmetry d(8h)

NumericalQuality Good

xc
 model SAOP
end

Excitations
 lowest 8
End

eor

mv TAPE21 Frag.t21
rm logfile TAPE21
```

```
$ADFBIN/adf -n1 <<eor
Title Ne spin-orbit

Atoms
  Ne .0000  .0000  0.0000  f=Frag
End

relativistic spinorbit zora

symmetry d(8h)

xc
 model SAOP
end

NumericalQuality Good

Excitations
 alsorestricted
 lowest 8
End

Fragments
 Frag Frag.t21
End

STCONTRIB

eor

rm logfile TAPE21

# The difference between the core excitation calculation and the standard
# excitation is the extra subkey MODIFYEXCITATION or SELECTEXCITATION in the
# core excitation calculation (in italic).

# ADF can not handle ATOM and linear symmetries in excitation calculations.
# Therefore a subsymmetry is used, in this case symmetry d(8h).

# A relatively large QZ4P basis set is used, which is still insufficient for
# excitations to Rydberg-like orbitals, one needs more diffuse functions.

# The key STCONTRIB is used, which will give a composition of the spin-orbit
# coupled excitation in terms of singlet-singlet and singlet-triplet scalar
# relativistic excitations. In order to use the key STCONTRIB the scalar
# relativistic fragment should be the complete molecule.

# In this case the key MODIFYEXCITATION or SELECTEXCITATION takes care that only
# excitations from the occupied 1s-orbital (spinor) are included. In symmetry
# d(8H) the 1s-orbital (spinor) is of A1.g (E1/2.g) symmetry.
```

```
Download Ne_CoreExci.run
```

```
#! /bin/sh


$ADFBIN/adf  <<eor
```

```
Title Ne

Atoms
  Ne .0000  .0000  0.0000
End

Basis
 Type QZ4P
End

relativistic scalar zora

symmetry d(8h)

NumericalQuality Good

xc
 model SAOP
end

Excitations
 lowest 8
End

ModifyExcitation
 UseOccupied
  A1.g 1
 SubEnd
 UseScaledZORA
END

eor

mv TAPE21 Fragmod.t21
rm logfile

$ADFBIN/adf -n1 <<eor
Title Ne spin-orbit

Atoms
  Ne .0000  .0000  0.0000  f=Frag
End

relativistic spinorbit zora

symmetry d(8h)

xc
 model SAOP
end

NumericalQuality Good

Excitations
 alsorestricted
 lowest 11
End
```

```
ModifyExcitation
 UseOccupied
  E1/2.g 1
 SubEnd
 UseScaledZORA
END

Fragments
 Frag Fragmod.t21
End

STCONTRIB

eor

mv TAPE21 Excitationsmod.t21
rm logfile

$ADFBIN/adf  <<eor
Title Ne

Atoms
  Ne .0000  .0000  0.0000
End

Basis
 Type QZ4P
End

relativistic scalar zora

symmetry d(8h)

NumericalQuality Good

xc
 model SAOP
end

Excitations
 lowest 8
End

SelectExcitation
 UseOccupied
  A1.g 1
 SubEnd
 UseScaledZORA
END

eor

mv TAPE21 Fragsel.t21
rm logfile

$ADFBIN/adf -n1 <<eor
Title Ne spin-orbit
```

```
Atoms
  Ne .0000  .0000  0.0000  f=Frag
End

relativistic spinorbit zora

symmetry d(8h)

xc
 model SAOP
end

NumericalQuality Good

Excitations
 alsorestricted
 lowest 11
End

SelectExcitation
 UseOccupied
  E1/2.g 1
 SubEnd
 UseScaledZORA
END

Fragments
 Frag Fragsel.t21
End

STCONTRIB

eor

mv TAPE21 Excitationssel.t21
rm logfile
```

Example: Excitation energies perturbative spin-orbit coupling: AgI

Download AgI_asoexcit.run

```
#! /bin/sh

# Calculation of the excitation energies of AgI including spin-orbit coupling in
# a perturbative way.

# ADF can not handle ATOM and linear symmetries in excitation calculations.
# Therefore a subsymmetry is used, in this case symmetry C(7v).

# For such a small molecule, a relatively small TZ2P basis set is used, which is
# not sufficient for excitations to Rydberg-like orbitals, one needs more
# diffuse functions. For larger molecules a TZ2P basis set is in most cases
# sufficient. In that case one might even consider smaller basis sets.

# The key SOPERT is included in scalar relativistic ZORA calculations of
# excitation energies. First scalar relativistic TDDFT calculations are
# performed to determine the lowest 60 singlet-singlet and singlet-triplet
```

```
# excited states and the spin-orbit coupling operator is applied to these
# single-group excited states to obtain the excitation energies with spin-orbit
# coupling effects included.


$ADFBIN/adf  <<eor
Title [AgI]

Atoms
 Ag .0000  .0000  2.5446
  I .0000  .0000  0.0000
End

relativistic scalar zora

symmetry C(7v)

Excitations
 lowest 60
End

SOPERT
End

NumericalQuality Good

Basis
 Type TZ2P
 Core None
End

eor

rm logfile TAPE21
```

Example: Excitation energies including spin-orbit coupling for open shell: PbF

Download PbF_excisou.run

```
#!/bin/sh

# Application of a spin-orbit coupled relativistic excitation energy calculation
# for an open-shell system. Note that the used method used is highly
# approximate. In this example the effect of spin-orbit coupling is large, in
# which case this method can be of use.

# First a scalar relativistic calculation is performed with symmetry. The
# resulting scalar relativistic TAPE21 is used as fragment in the spin-orbit
# coupled calculation. Using a scalar relativistic fragment is not necessary,
# but makes the identification of the spin-orbit coupled results easier.

$ADFBIN/adf   <<eor
Atoms
  Pb   0    0    0
  F    0    0    2.06
End
Basis
```

```
 Type TZ2P
 Core None
End
relativistic scalar zora
xc
 gga bp86
end
eor
mv TAPE21 PbF.t21
rm logfile


# Next the spin-orbit coupled excitation unrestricted calculation is perfomed.
# The Tamm-Dancoff approximation (TDA) is needed and symmetry NOSYM should be
# used. Best is to use the noncollinear approximation.


$ADFBIN/adf   <<eor
Atoms
  Pb   0    0    0     f=PbF
  F    0    0    2.06 f=PbF
End
Fragments
 PbF PbF.t21
End
relativistic spinorbit zora
xc
 gga bp86
end
unrestricted
noncollinear
symmetry nosym
tda
excitations
 lowest 14
end
eor
rm TAPE21 logfile
```

Example: Excited state geometry optimization: N2

Download EGO_N2.run

```
#!/bin/sh

# Example for a singlet excited state geometry optimization for N_2.

# Needed for such excited state optimizations are the key EXCITATIONS (to
# calculate excitation energies), the key GEOMETRY (to do a geometry
# optimization) and the key EXCITEDGO (to select for which excitation a geometry
# optimization should be performed). In this case a hybrid functional B3LYP is
# used.


$ADFBIN/adf <<eor
TITLE N2 Excited state geometry
```

```
atoms
    N          0.0         0.0         -0.7
    N          0.0         0.0          0.7
end

XC
 HYBRID B3LYP
END

BeckeGrid
 Quality good
End

GEOMETRY
 ITERATIONS 30
 CONVERGE grad=0.0001
END

basis
 TYPE DZ
CORE NONE
end

excitations
LOWEST 10
onlysing
end

EXCITEDGO
 STATE  S-.u 1
 OUTPUT 2
end

ALLPOINTS

eor
```

Download EGO_N2_EIGENF.run

```
#!/bin/sh

# Example for a singlet excited state geometry optimization with eigenvector
# following (subkeyword EIGENFOLLOW of key EXCITEDGO), GGA functional used.


$ADFBIN/adf <<eor
TITLE N2 Eigenvector follow. Lowest state at the starting geometry is not the lowest
→at min.

atoms
    N          0.0         0.0         -0.55
    N          0.0         0.0          0.55
end

XC
 GGA Becke Perdew
END
```

```
BeckeGrid
 Quality good
End

GEOMETRY
 ITERATIONS 30
 CONVERGE grad=0.0001
END

basis
 TYPE DZ
CORE NONE
end

excitations
LOWEST 10
onlysing
end

EXCITEDGO
 STATE  A 1
 OUTPUT 2
 EIGENFOLLOW
end

SYMMETRY NOSYM

eor
```

Example: Spin-flip excited state geometry optimization: CH2

Download EGO_CH2_sf.run

```
#!/bin/sh

# Example for a spin-flip excited state geometry optimization with a triplet
# reference, and a frequency calculation afterwards.

# Needed for such excited state optimizations are the key EXCITATIONS (to
# calculate excitation energies), the key GEOMETRY (to do a geometry
# optimization) and the key EXCITEDGO (to select for which excitation a geometry
# optimization should be performed). In this case spin-flip excitations are
# calculated.


$ADFBIN/adf <<eor
TITLE CH2 Excited state geometry with triplet reference and spin-flip excitation

atoms
    C         0.000000    0.000000    0.0
    H         0.7         0.0         0.7
    H        -0.7         0.0         0.7
end

XC
END
```

```
UNRESTRICTED

CHARGE 0 2

GEOMETRY
 ITERATIONS 50
 CONVERGE E=0.0001 grad=0.00001
END

SCF
 Converge 1.0e-9
END

basis
 TYPE DZP
CORE NONE
end

excitations
LOWEST 10
end

TDA
SFTDDFT
FORCEALDA

EXCITEDGO
 STATE  B2 1
 OUTPUT 2
end

SYMMETRY C(2V)
eor

cp TAPE21 CH2.t21


# Next the frequencies are calculated of the excited state. A restart is used to
# pick up the excited state geometry of the previous calculation. Note that in a
# numerical FREQUENCIES calculation symmetry is turned off except to reduce the
# number of points calculated. Thus irrespective of the specified point group
# symmetry the symmetry label A of SYMMETRY NOSYM should be used to select the
# excited state. Care should be taken to ensure that the correct state is chosen
# in this frequencies calculation as the excited state number can change when
# the point group is changed. In this case instead of 'B2 1' one needs to select
# 'A 2'.



$ADFBIN/adf <<eor
TITLE CH2 Excited state frequencies with triplet reference and spin-flip excitation

atoms
    C         0.000000    0.000000    0.0
    H         0.7         0.0         0.7
    H        -0.7         0.0         0.7
end
```

```
RESTART
   File CH2.t21
End

XC
END

UNRESTRICTED

CHARGE 0 2

GEOMETRY
 FREQUENCIES
 End
END

SCF
 Converge 1.0e-9
END

basis
 TYPE DZP
CORE NONE
end

excitations
LOWEST 10
end

TDA
SFTDDFT
FORCEALDA

EXCITEDGO
 STATE  A 2
 OUTPUT 2
end
SYMMETRY C(2V)
eor
```

Example: Circular Dichroism (CD) spectrum: DMO

Download DMO_CD.run

```
#!/bin/sh

# If the subkey CDSPECTRUM is included in the key EXCITATIONS, the rotatory
# strength is calculated for the calculated excitations, in order to calculate
# the CD (Circular Dichroism) spectrum. Only useful for chiral molecules.

# With the VELOCITY keyword also the dipole-velocity representation of the
# rotatory strength is calculated.

# Note: results will be physically meaningless due to small basis set. purpose
# of this job is to provide a test case for the CD implementation

# Do not use less strict convergence criteria than default, better to use
```

```
# tighter criteria. The approximations in the evaluation of the integrals one
# makes with the linear scaling techniques are effectively switched off by
# setting LINEARSCALING 100 (recommended to use this).


$ADFBIN/adf <<eor
TITLE dimethyloxirane excitations + CD

COMMENT
 results will be physically  meaningless due to small basis set.
 purpose of this job is to provide a test case for the CD implementation
END

XC
  gga becke perdew
END

Basis
 Type DZP
 Core Small
End

ATOMS
O       0.000129    1.141417    0.000023
C      -0.597040   -0.094320    0.428262
C       0.596952   -0.094328   -0.428223
H      -0.442927   -0.302650    1.487698
H       0.442944   -0.302474   -1.487720
C      -1.978779   -0.386617   -0.093924
H      -2.723275    0.220579    0.429114
H      -2.043506   -0.157697   -1.159810
H      -2.236045   -1.439970    0.055144
C       1.978716   -0.386693    0.093893
H       2.236030   -1.439985   -0.055498
H       2.723156    0.220701   -0.429005
H       2.043497   -0.158088    1.159845
END

linearscaling 100
End

excitations
 cdspectrum
 onlysing
 velocity
 lowest 10
end
SCF
 converge 1e-7
End

BeckeGrid
 Quality good
End

eor

mv TAPE21 DMO.t21
```

Example: CD spectrum, hybrid functional: Twisted ethene

Download Twist_Ethene_TDDFT.run

```sh
#!/bin/sh


# If the subkey CDSPECTRUM is included in the key EXCITATIONS, the rotatory
# strength is calculated for the calculated excitations, in order to calculate
# the CD (Circular Dichroism) spectrum. Only useful for chiral molecules. A
# hybrid functional is used in this case.

# With the VELOCITY keyword also the dipole-velocity representation of the
# rotatory strength is calculated.

# Note: results will be physically meaningless due to small basis set. purpose
# of this job is to provide a test case for the CD implementation

# Do not use less strict convergence criteria than default, better to use
# tighter criteria. The approximations in the evaluation of the integrals one
# makes with the linear scaling techniques are effectively switched off by
# setting LINEARSCALING 99 (recommended to use this).

$ADFBIN/adf <<eor
TITLE twisted ethene

COMMENT
 purpose of this job is to provide a test case for the CD implementation
 with hybrid functionals, and to test the "velocity" keyword
END

XC
 hybrid PBE0
END

RIHartreeFock
  FitSetQuality Normal
End

LinearScaling 99
End

Basis
 Type TZ2P
 Core Small
 CreateOutput none
End

ATOMS
C 0.000000 0.000000 0.000000
C -0.000000 -0.000000 1.350716
H 0.565703 -0.745428 -0.540179
H -0.565703 0.745428 -0.540179
H 0.035837 -0.935093 1.890895
H -0.035837 0.935093 1.890895
END

excitations
```

```
 cdspectrum
 onlysing
 velocity
 lowest 20
end
SCF
  Converge 1.e-8
END
numericalquality good

eor
```

Example: MCD: H2O

Download H2O_MCD.run

```
#! /bin/sh

# Example for the calculation of magnetic circular dichroism (MCD). If the
# subkey MCD is included in the key EXCITATIONS the MCD parameters of the
# calculated excitations are calculated (A and B terms). The keys RELATIVISTIC
# ZORA and SOMCD are required for a calculation of temperature-dependent C
# terms. The key ALLPOINTS is required for an MCD calculation (if the molecule
# has symmetry).


$ADFBIN/adf <<eor
title water MCD

atoms
 O 0 0 0
 H 0 0 1
 H 0 1 0
end

BASIS
 TYPE DZP
end

ALLPOINTS

SOMCD

UNRESTRICTED

CHARGE 1 1

RELATIVISTIC ZORA

excitations
 lowest 20
 onlysing
 mcd NMCDTERM=5
end

eor
```

Example: MCD including zero-field splitting: H2O

Download `H2O_MCD_ZFS.run`

```sh
#! /bin/sh

# Example for the calculation of magnetic circular dichroism (MCD) including
# zero-field splitting (ZFS). If the subkey MCD is included in the key
# EXCITATIONS the MCD parameters of the calculated excitations are calculated (A
# and B terms). The keys RELATIVISTIC ZORA and SOMCD are required for a
# calculation of temperature-dependent C terms. The key ALLPOINTS is required
# for an MCD calculation (if the molecule has symmetry). For zero-field
# splitting (ZFS) the electron spin S<=1.


$ADFBIN/adf <<eor
title water MCD with zero-field zplitting

atoms
 O 0 0 0
 H 0 0 1
 H 0 1 0
end

BASIS
 TYPE DZP
end

ALLPOINTS

SOMCD

UNRESTRICTED

CHARGE 2 2

RELATIVISTIC ZORA

ZFS

excitations
 lowest 20
 onlysing
 mcd NMCDTERM=5 NODIRECT NTEMP=2 TMIN=1.0 TMAX=300.0 NBFIELD=2 BMIN=1.0 BMAX=8.0
end

eor
```

Example: CV(n)-DFT excitation energies: Formamide

Download `Formamide_CVnDFT.run`

```sh
#!/bin/sh

# Calculation of the excitation energies of Formamide using the CV(n)-DFT
# method. Due to bugs in older versions it is important to use ADF2016.105 or
# later.
```

```
# Example of CV(infinity)-DFT where the molecular orbitals for the particular
# excitation are optimized (relaxed), which is called the R-CV(infinity)-DFT method.
# In this case, the transition U-vector is frozen, while the orbitals are
# relaxed. ALLPOINTS should be used in case of a symmetric molecule.
# EXACTDENSITY should be used for precision reasons. Note that the irrep labels
# for C(s) symmetry that are used in the excitation code (A' and A'') differ
# from those that are used during the SCF (AA and AAA). The example contains of
# 2 parts. In the first example singlet-singlet excitations are calculated using
# the subkeyword ONLYSING in the EXCITATIONS block key. In the second example
# singlet-triplet excitations are calculated replacing the subkeyword ONLYSING
# with the subkeyword ONLYTRIP in the EXCITATIONS block key.


$ADFBIN/adf <<eor
Title Formamide

basis
 Type DZ
 FitType ZORA/QZ4P
 Core None
end
Numericalquality good

atoms
H -0.927427 -0.600301 0.000000
H  1.070498 -1.782390 0.000000
H  2.024514 -0.325050 0.000000
C  0.000000  0.000000 0.000000
O  0.000000  1.225060 0.000000
N  1.119392 -0.775069 0.000000
end

allpoints

xc
 hybrid B3LYP
end

CVnDFT
 ITERATION 10
 TOLERANCE 0.001
 R_CV_DFT &
  RELAXALPHA 1
  RELAXBETA 1
  DAMPORBRELAX 0.2
 SUBEND
END

TDA

EXACTDENSITY

EXCITATIONS
 davidson
  A' 2
  A'' 2
 SUBEND
```

```
 lowest 4
 ONLYSING
 Analytical
 TOLERANCE 1e-5
END

eor

rm TAPE21 logfile



$ADFBIN/adf <<eor
Title Formamide

basis
 Type DZ
 FitType ZORA/QZ4P
 Core None
end
Numericalquality good

atoms
H -0.927427 -0.600301 0.000000
H  1.070498 -1.782390 0.000000
H  2.024514 -0.325050 0.000000
C  0.000000  0.000000 0.000000
O  0.000000  1.225060 0.000000
N  1.119392 -0.775069 0.000000
end

allpoints

xc
 hybrid B3LYP
end

CVnDFT
 ITERATION 10
 TOLERANCE 0.001
 R_CV_DFT &
  RELAXALPHA 1
  RELAXBETA 1
  DAMPORBRELAX 0.2
 SUBEND
END

TDA

EXACTDENSITY

EXCITATIONS
 davidson
  A' 2
  A'' 2
 SUBEND
 lowest 4
 ONLYTRIP
```

```
 ANALYTICAL
 TOLERANCE 1e-5
END

eor

rm TAPE21 logfile
```

Example: TD-DFT+TB excitation energies: beta-Carotene

Download `TD-DFTB_betacarotene.run`

```
#!/bin/sh

# Calculation of the excitation energies of beta-Carotene using the TD-DFT+TB
# method. This method is meant for GGA or LDA functionals. In this example PBE
# is used during the SCF. In the the calculation of the excitation energies the
# TD-DFTB+TB method is used in which the required integrals are approximated in
# the same way as in a TD-DFTB calculation. Symmetry NOSYM is required.

# The key MODIFYEXCITATION allows to remove single orbital transitions based on
# certain criteria. For example, for a large calculation, one can combine
# several of the filtering techniques with the key MODIFYEXCITATION. If one uses
# the next input, only the lowest 10000 single orbital transitions are used
# (which have the lowest orbital energy differences). Out of these 10000 the
# single orbital transitions with an oscillator strength smaller than 0.001 are
# removed, and single orbital transitions are removed that have an energy
# difference larger than 1.0 Hartree. In the space of single-orbital transition
# that are left the lowest 1000 eigenvectors are calculated.

#    ModifyExcitation
#      UseOccVirtNumbers 1 10000
#      OscStrength 0.001
#      UseOccVirtRange 0.0 1.0
#    End
#    Excitations
#      TDDFTB
#      OnlySing
#      Lowest 1000
#    End

# In the actual example only single orbital transitions with an oscillator
# strength smaller than 0.001 are removed, which one might do if one is
# interested only in excitation energies with a high oscillator strength.


$ADFBIN/adf <<eor
Basis
End
XC
 GGA PBE
END
Symmetry NOSYM
Excitations
 TD-DFTB
 OnlySing
End
```

```
ModifyExcitation
 OscStrength 0.001
End
Atoms
        C       -1.67096000        1.41980000       -1.15887000
        C       -0.38686000        2.25210000       -1.41391000
        C       -1.74087000        0.49471000       -0.15347000
        C       -2.78739000        1.72912000       -2.05465000
        C        0.64868000        2.05301000       -0.28395000
        C        0.23104000        1.84552000       -2.77135000
        C       -0.70560000        3.76543000       -1.46182000
        C       -0.54745000        0.11313000        0.69574000
        C       -3.00351000       -0.21803000        0.26657000
        C       -3.76926000        0.89738000       -2.51350000
        C        0.78914000        0.59489000        0.13981000
        C       -4.86625000        1.29159000       -3.36539000
        C       -5.83916000        0.35729000       -3.66241000
        C       -4.93554000        2.71910000       -3.84917000
        C       -7.05227000        0.56749000       -4.37867000
        C       -8.05019000       -0.37384000       -4.49171000
        C       -9.34557000       -0.15733000       -5.06205000
        C      -10.30909000       -1.14399000       -4.91514000
        C       -9.64286000        1.17054000       -5.71730000
        C      -11.69180000       -1.05048000       -5.21459000
        C      -12.61955000       -2.01468000       -4.87148000
        C      -14.02165000       -1.83518000       -4.98190000
        C      -15.01934000       -2.67078000       -4.50243000
        C      -16.37196000       -2.20132000       -4.51598000
        C      -14.71829000       -4.01459000       -3.88235000
        C      -17.44392000       -2.81539000       -3.90884000
        C      -18.74098000       -2.23505000       -3.81068000
        C      -19.81891000       -2.70482000       -3.08600000
        C      -21.01079000       -1.89278000       -3.01886000
        C      -19.77834000       -4.00300000       -2.31809000
        C      -22.11699000       -2.17469000       -2.26831000
        C      -23.34229000       -1.38813000       -2.11260000
        C      -24.62364000       -2.22792000       -1.86832000
        C      -23.37421000       -0.02091000       -2.14864000
        C      -25.81771000       -1.33236000       -1.46707000
        C      -24.97031000       -3.02505000       -3.14664000
        C      -24.41619000       -3.23491000       -0.71197000
        C      -24.66206000        0.77221000       -2.09033000
        C      -22.15090000        0.86196000       -2.20093000
        C      -25.91751000       -0.06612000       -2.31089000
        H       -2.80910000        2.75370000       -2.43250000
        H        1.61418000        2.46808000       -0.61595000
        H        0.33480000        2.64516000        0.59383000
        H       -0.50118000        1.95999000       -3.58430000
        H        1.10267000        2.47962000       -3.00160000
        H        0.55846000        0.79551000       -2.76878000
        H       -1.27083000        4.08204000       -0.57268000
        H        0.23706000        4.33391000       -1.48690000
        H       -1.27659000        4.05053000       -2.35573000
        H       -0.69913000        0.51646000        1.71643000
        H       -0.54631000       -0.98424000        0.82135000
        H       -3.05490000       -1.23815000       -0.15105000
        H       -3.01240000       -0.33500000        1.36235000
        H       -3.91214000        0.31400000       -0.03789000
```

```
        H       -3.75416000      -0.15646000      -2.22797000
        H        1.07957000      -0.02480000      -0.72369000
        H        1.58320000       0.48035000       0.89321000
        H       -5.69282000      -0.64720000      -3.25046000
        H       -4.00625000       2.99962000      -4.36844000
        H       -5.76674000       2.88684000      -4.54145000
        H       -5.05424000       3.41602000      -3.00441000
        H       -7.22923000       1.55534000      -4.80743000
        H       -7.87841000      -1.35648000      -4.03966000
        H       -9.98472000      -2.07170000      -4.43134000
        H       -8.86494000       1.42343000      -6.45273000
        H      -10.60514000       1.16993000      -6.23979000
        H       -9.66280000       1.98327000      -4.97344000
        H      -12.06891000      -0.12920000      -5.66646000
        H      -12.24343000      -2.93190000      -4.41065000
        H      -14.35090000      -0.88585000      -5.41809000
        H      -16.54169000      -1.22776000      -4.98799000
        H      -15.37918000      -4.79019000      -4.29660000
        H      -13.68496000      -4.33124000      -4.05827000
        H      -14.88466000      -3.99412000      -2.79324000
        H      -17.27327000      -3.77423000      -3.41639000
        H      -18.87740000      -1.27545000      -4.32124000
        H      -20.99304000      -0.98523000      -3.62576000
        H      -19.85755000      -3.82238000      -1.23433000
        H      -20.62535000      -4.64779000      -2.59868000
        H      -18.85754000      -4.56711000      -2.49752000
        H      -22.10976000      -3.13142000      -1.74142000
        H      -26.74148000      -1.93003000      -1.53138000
        H      -25.70276000      -1.04055000      -0.40816000
        H      -24.12173000      -3.65146000      -3.45913000
        H      -25.83556000      -3.68339000      -2.96560000
        H      -25.21348000      -2.35763000      -3.98638000
        H      -24.03490000      -2.73308000       0.18974000
        H      -25.38111000      -3.70177000      -0.46026000
        H      -23.72534000      -4.04703000      -0.97564000
        H      -24.71649000       1.28228000      -1.10831000
        H      -24.60164000       1.58888000      -2.83152000
        H      -21.95198000       1.22417000      -3.22414000
        H      -22.31941000       1.76042000      -1.58543000
        H      -21.24680000       0.35543000      -1.84399000
        H      -26.00957000      -0.32720000      -3.37736000
        H      -26.81690000       0.51083000      -2.04725000
End
eor
```

Example: sTDA excitation energies: Adenine

Download Adenine_sTDA.run

```
#! /bin/sh

# Calculation of the excitation energies of Adenine using the simplified Tamm-
# Dancoff (sTDA) method by Grimme. This method is meant for hybrid functionals
# or range-separated functionals during the SCF. In this example the hybrid PBE0
# is used during the SCF. In the the calculation of the excitation energies the
# sTDA method is used in which the required integrals are approximated. For
# hybrid functionals ADF will automatically set the parameters that are needed
```

```
# in this method. For range-separated functional one needs to set the parameters
# manually. Symmetry NOSYM is required. A TZP or TZ2P basis set is recommended
# for this method.


$ADFBIN/adf <<eor
   Atoms
       1.N          1.966533    -0.556678      0.000000
       2.C          1.351210    -1.750556      0.000000
       3.N          0.039008    -2.019043      0.000000
       4.C         -0.679938    -0.889685      0.000000
       5.C         -0.192782     0.421840      0.000000
       6.C          1.210298     0.557343      0.000000
       7.N         -2.051805    -0.746568      0.000000
       8.C         -2.311388     0.608600      0.000000
       9.N         -1.225604     1.347025      0.000000
      10.N          1.828584     1.760602      0.000000
      11.H          2.012099    -2.612788      0.000000
      12.H         -2.725918    -1.499744      0.000000
      13.H         -3.323166     0.988113      0.000000
      14.H          1.288936     2.610953      0.000000
      15.H          2.835883     1.795025      0.000000
   End

   Basis
      Type DZ
   End

   Symmetry nosym

   Excitations
      stda
      onlysing
   end

   XC
      Hybrid PBE0
   end

eor
```

Example: sTDDFT excitation energies: Adenine

Download Adenine_sTDDFT.run

```
#! /bin/sh

# Calculation of the excitation energies of Adenine using the simplified time-
# dependent DFT (sTDDFT) method by Grimme. This method is meant for hybrid
# functionals or range-separated functionals during the SCF. In this example the
# hybrid PBE0 is used during the SCF. In the the calculation of the excitation
# energies the sTDDFT method is used in which the required integrals are
# approximated. For hybrid functionals ADF will automatically set the parameters
# that are needed in this method. For range-separated functional one needs to
# set the parameters manually. Symmetry NOSYM is required A TZP or TZ2P basis
# set is recommended for this method.
```

```
$ADFBIN/adf <<eor
  Atoms
     1.N        1.966533    -0.556678     0.000000
     2.C        1.351210    -1.750556     0.000000
     3.N        0.039008    -2.019043     0.000000
     4.C       -0.679938    -0.889685     0.000000
     5.C       -0.192782     0.421840     0.000000
     6.C        1.210298     0.557343     0.000000
     7.N       -2.051805    -0.746568     0.000000
     8.C       -2.311388     0.608600     0.000000
     9.N       -1.225604     1.347025     0.000000
    10.N        1.828584     1.760602     0.000000
    11.H        2.012099    -2.612788     0.000000
    12.H       -2.725918    -1.499744     0.000000
    13.H       -3.323166     0.988113     0.000000
    14.H        1.288936     2.610953     0.000000
    15.H        2.835883     1.795025     0.000000
  End

  Basis
     Type DZ
  End

  Symmetry nosym

  Excitations
     stddft
     onlysing
  End

  XC
   Hybrid pbe0
  End

eor
```

Example: sTDA excitation energies RS functional: Bimane

Download `Bimane_wB97X.run`

```
#! /bin/sh



# Calculation of the excitation energies of Bimane using the simplified Tamm-
# Dancoff (sTDA) method by Grimme, for a range-separated function. This method
# is meant for hybrid functionals or range-separated functionals during the SCF.
# In this example the range-separated functional wB97X is used during the SCF.
# In the the calculation of the excitation energies the sTDA method is used
# in which the required integrals are approximated. For range-separated
# functional one needs to set the parameters manually. Note that the alpha and
# beta parameters for sTDA for range separated functionals in the Risthaus paper
# (dx.doi.org/10.1039/C3CP54517B) are mixed up. A TZP or TZ2P basis set is
# recommended for this method.

# First the atoms are calculated. To calculate this range-separated functional
# wB97X the RIHARTREEFOCK method is required, and LibXC is needed. For speed
```

```
# reasons the atoms are calculated separately, such that these calculations
# might run in parallel. If the BASIS key is used the atoms are calculated
# serially.


for n in H C N O
do
$ADFBIN/adf <<eor
Create $n file=$ADFRESOURCES/DZ/$n
XC
 LibXC wB97X
End
eor
mv TAPE21 $n.t21
rm logfile
done


# Next the excitation energies of Bimane are calculated. Symmetry NOSYM is
# required. The sTDA parameters should be set with the subkeys GRIMMEAEX,
# GRIMMEALPHA, and GRIMMEBETA of the key MODIFYEXCITATION.


$ADFBIN/adf <<eor
Atoms
 C      1.331698   -1.879605   -0.108603
 C      2.132097   -0.654088   -0.003948
 C      1.302873    0.426430   -0.104903
 C     -1.300787    0.433951   -0.101824
 C     -2.136017   -0.641707    0.001280
 C     -1.342993   -1.871870   -0.104659
 N     -0.000534   -0.021248   -0.324354
 N     -0.004568   -1.403598   -0.331622
 O     -1.679176   -3.039253   -0.043168
 O      1.661467   -3.048960   -0.049039
 C     -1.615093    1.891323   -0.058117
 C     -3.618641   -0.678771    0.170347
 C      1.626055    1.881942   -0.064754
 C      3.614962   -0.700043    0.160744
 H     -4.049414    0.325596    0.200326
 H     -3.887430   -1.197656    1.097229
 H     -4.086348   -1.233771   -0.650635
 H     -1.173280    2.374317    0.820379
 H     -2.696603    2.026980   -0.008719
 H     -1.243008    2.407629   -0.949030
 H      2.707594    2.010775   -0.000483
 H      1.174301    2.373283    0.803861
 H      1.270709    2.395192   -0.964373
 H      4.079173   -1.241294   -0.671464
 H      3.883147   -1.237421    1.077108
 H      4.049818    0.301869    0.208419
End
Fragments
 H H.t21
 C C.t21
 N N.t21
 O O.t21
End
```

```
symmetry nosym
excitations
 stda
 onlysing
end
XC
 LibXC wB97X
END
modifyexcitation
 GRIMMEPERTC
 GRIMMEAEX   0.56
 GRIMMEALPHA 4.51
 GRIMMEBETA  8.00
end
eor
```

Example: sTDA excitation energies wB97: TCNE-Benzene

Download TCNE-Benzene_wB97.run

```
#! /bin/sh


# Calculation of the excitation energies of the complex TCNE-Benzene using the
# simplified Tamm-Dancoff (sTDA) method by Grimme, for a range-separated
# function. This method is meant for hybrid functionals or range-separated
# functionals during the SCF. In this example the range-separated functional
# wB97 is used during the SCF. In the the calculation of the excitation
# energies the sTDA method is used in which the required integrals are
# approximated. For range-separated functional one needs to set the parameters
# manually. Note that the alpha and beta parameters for sTDA for range separated
# functionals in the Risthaus paper (dx.doi.org/10.1039/C3CP54517B) are mixed
# up. A TZP or TZ2P basis set is recommended for this method.

# First the atoms are calculated. To calculate this range-separated functional
# wB97 the RIHARTREEFOCK method is required, and LibXC is needed. For speed
# reasons the atoms are calculated separately, such that these calculations
# might run in parallel. If the BASIS key is used the atoms are calculated
# serially.

for n in H C N
do
$ADFBIN/adf <<eor
Create $n file=$ADFRESOURCES/DZ/$n
XC
 LibXC wB97
End
eor
mv TAPE21 $n.t21
rm logfile
done


# Next the excitation energies of the complex TCNE-Benzene are calculated, in
# which charge transfer (CT) states are important. Symmetry NOSYM is required.
# The subkey DESCRIPTORS is added to the key EXCITATIONS such that CT descriptors
# are calculated. The the atomic distance criterion used for the calculation of
# CT_AT was set to 3.0 Angstrom.
# The sTDA parameters should be set with the subkeys GRIMMEAEX, GRIMMEALPHA, and
```

```
# GRIMMEBETA of the key MODIFYEXCITATION.

$ADFBIN/adf <<eor
Atoms
C      -2.274205    -1.398789    -0.000000
C      -2.274206     1.398791    -0.000000
C      -2.274506    -0.699551     1.213150
C      -2.274505    -0.699551    -1.213150
C      -2.274506     0.699551     1.213150
C      -2.274505     0.699551    -1.213150
H      -2.278385    -2.491323    -0.000000
H      -2.278386     2.491321    -0.000000
C       1.368996    -0.686302     0.000000
C       1.368996     0.686301     0.000000
C       1.379968    -1.434052    -1.220900
C       1.379968    -1.434052     1.220895
C       1.379968     1.434051    -1.220900
C       1.379967     1.434051     1.220895
N       1.400955    -2.054479    -2.204820
N       1.400955    -2.054479     2.204824
N       1.400955     2.054481    -2.204820
N       1.400954     2.054481     2.204824
H      -2.276046    -1.246672     2.158733
H      -2.276045    -1.246672    -2.158730
H      -2.276045     1.246671    -2.158730
H      -2.276046     1.246671     2.158733
End
Fragments
 H H.t21
 C C.t21
 N N.t21
End
symmetry nosym
excitations
 stda
 onlysing
 descriptors
 descriptors_CT_AT_Rab 3.0
end
XC
 LibXC wB97
END
modifyexcitation
 GRIMMEPERTC
 GRIMMEAEX   0.61
 GRIMMEALPHA 4.41
 GRIMMEBETA  8.00
end
eor
```

14.4.3 (Hyper-)Polarizabilities, dispersion coefficients, ORD, magnetizabilities, Verdet constants

Example: Polarizabilities including spin-orbit coupling: AgI

Download `AgI_SO_Pol.run`

```sh
#! /bin/sh

# Example shows an frequency-dependent ZORA calculation of complex linear
# response of AgI including spin-orbit coupling.

# The polarizability is calculated using the AORESPONSE key, in which spin-orbit
# coupling is taken into account. In this case a spin-restricted calculation is
# required, but, unlike the rest of AORESPONSE, also NOSYM. In this example
# lifetime effects are included.


"$ADFBIN/adf" <<eor
TITLE AgI,SO-ZORA

basis
 type TZ2P
 core None
end

ATOMS
  Ag   0.0 0.0  1.2723
  I    0.0 0.0  -1.2723
END

symmetry nosym
allpoints

BeckeGrid
 quality good
End

linearscaling 9.0
End

xc
  lda VWN
end

Relativistic zora spinorbit

aoresponse
 scf
 frequency 1 0.085 HARTREE
 lifetime 0.007
 ALDA
 EL_DIPOLE_EL_DIPOLE
end

eor
```

Example: damped first hyperpolarizability: LiH

Download LiH_DampedBeta.run

```sh
#! /bin/sh

# If the subkey lifetime and BETA or QUADRATIC is included in the key
```

```
# AORESPONSE, the damped (frequency dependent) first hyperpolarizability is
# calculated. This test example consists of two calculations calculations: one
# with BETA and the other with QUADRATIC.

# The subkey EOPE is used, which means the electro-optical Pockels effect
# beta(-omega;omega,0). This example can easily be modified to calculate the
# static beta(0;0,0), the optical rectification beta(0;omega,-omega), the second
# harmonic generation beta(-2omega;omega,omega), or the general case
# beta(-(omega1+omega2);omega1,omega2).

# Note: results will be physically meaningless due to small basis set. Purpose
# of this job is to provide a test case for the first hyperpolarizability
# implementation

# In the first example the first hyperpolarizability is calculated with the
# subkey BETA, for EOPE, the electro-optical Pockels effect
# beta(-omega;omega,0).

###########################
# Damped EOPE, 2n+1
###########################

$ADFBIN/adf <<eor
title Damped EOPE of LiH, 2n+1

basis
 Type SZ
 core None
end

atoms
Li    0.00000000     0.00000000     3.49467000
H     0.00000000     0.00000000     1.89402300
end

symmetry nosym
allpoints

numericalquality good

aoresponse
 scf  iterations 50
 frequency 2 0.1000 0.0000 Hartree
 beta
 eope
 ALDA
 lifetime 0.0034
end

eor

rm TAPE21 logfile


# In the second example the first hyperpolarizability is calculated with the
# subkey QUADRATIC, again for EOPE, the electro-optical Pockels effect
# beta(-omega;omega,0).
```

```
################################
# Damped EOPE, Quadratic
################################

$ADFBIN/adf <<eor
title Damped EOPE of LiH, Quadratic

basis
 Type SZ
 core None
end

atoms
Li    0.00000000    0.00000000    3.49467000
H     0.00000000    0.00000000    1.89402300
end

symmetry nosym
allpoints

numericalquality good

aoresponse
 scf  iterations 50
 frequency 2 0.1000 0.0000 Hartree
 quadratic
 eope
 ALDA
 lifetime 0.0034
end

eor

rm TAPE21 logfile

# For the static case beta(0;0,0) use the subkey STATIC
#   aoresponse
#   ...
#   frequency 2 0.0000 0.0000 Hartree
#   static
#   end

# For optical rectification beta(0;omega,-omega) use the subkey OPTICALR.
#   aoresponse
#   ...
#   frequency 2 0.1000 -0.1000 Hartree
#   opticalr
#   end

# For the second harmonic generation beta(-2omega;omega,omega) use the subkey SHG
#   aoresponse
#   ...
#   frequency 2 0.1000 0.1000 Hartree
#   shg
#   end

# Or in the general case for beta(-(omega1+omega2);omega1,omega2) choose two
# input frequencies omega1 and omega2
```

```
#    aoresponse
#    ...
#    frequency 2 omega1 omega2 Hartree
#    end
```

Example: damped second hyperpolarizability: LiH

Download LiH_DampedGamma.run

```
#! /bin/sh

# If the subkey lifetime and GAMMA or CUBIC is included in the key AORESPONSE,
# the damped (frequency dependent) second hyperpolarizability is calculated.
# This test example consists of three calculations calculations: two with GAMMA
# and one with CUBIC.

# The the first two examples subkey EFIOR is used, which means the electric
# field induced optical rectification gamma(0;omega,-omega,0). In the third
# example two-photon absorption (TPA) cross sections are calculated, the reduced
# form of gamma(-omega;omega,omega,-omega). Examples can easily be modified to
# calculate the static case gamma(0;0,0,0), the optical Kerr effect
# gamma(-omega;omega,0,0), the intensity dependent refractive index
# gamma(-omega;omega,omega,-omega), the electric field induced second harmonic
# generation gamma(-2omega;omega,omega,0), the third harmonic generation
# gamma(-3omega;omega,omega,omega) use the subkey THG, or the general case
# gamma(-(omega1+omega2+omega3);omega1,omega2,omega3).

# Note: results will be physically meaningless due to small basis set. Purpose
# of this job is to provide a test case for the second hyperpolarizability
# implementation

# In the first example the second hyperpolarizability is calculated with the
# subkey GAMMA, for EFIOR, the electric field induced optical rectification
# gamma(0;omega,-omega,0).


####################
# Damped EFIOR, 2n+1
####################

$ADFBIN/adf <<eor
title Damped EFIOR of LiH, 2n+1

basis
 Type SZ
 core None
end

atoms
Li    0.00000000    0.00000000    3.49467000
H     0.00000000    0.00000000    1.89402300
end

symmetry nosym
allpoints

numericalquality good
```

```
aoresponse
 scf iterations 50
 frequency 3 0.1000 -0.1000 0.0000 Hartree
 gamma
 efior
 ALDA
 lifetime 0.0034
end

eor

rm TAPE21 logfile

# In the second example the second hyperpolarizability is calculated with the
# subkey CUBIC, again for EFIOR, the electric field induced optical
# rectification gamma(0;omega,-omega,0).

#####################
# Damped EFIOR, Cubic
#####################

$ADFBIN/adf <<eor
title Damped EFIOR of LiH, Cubic

basis
 Type SZ
 core None
end

atoms
Li    0.00000000    0.00000000    3.49467000
H     0.00000000    0.00000000    1.89402300
end

symmetry nosym
allpoints

numericalquality good

aoresponse
 scf iterations 50
 frequency 3 0.1000 -0.1000 0.0000 Hartree
 cubic
 efior
 ALDA
 lifetime 0.0034
end

eor

rm TAPE21 logfile

# In the third example the subkeyword TPA is be used to calculate the gammagamma
# corresponding to the two photon absorption process (i.e., the reduced form of
# gamma(-omega;omega,omega,-omega)), however, TPA can ONLY be used with keyword
# GAMMA.
```

```
##################
# Damped TPA, 2n+1
##################

$ADFBIN/adf <<eor
title Damped TPA of LiH, 2n+1

basis
 Type SZ
 core None
end

atoms
Li    0.00000000    0.00000000    3.49467000
H     0.00000000    0.00000000    1.89402300
end

symmetry nosym
allpoints

numericalquality good

aoresponse
 scf iterations 50
 frequency 3 0.1000 0.1000 -0.1000 Hartree
 gamma
 tpa
 ALDA
 lifetime 0.0034
end

eor

rm TAPE21 logfile

# For the static case gamma(0;0,0,0) use the subkey STATIC
#   aoresponse
#   ...
#   frequency 3 0.0000 0.0000 0.0000 Hartree
#   static
#   end

# For the optical Kerr effect gamma(-omega;omega,0,0) use the subkey OKE
#   aoresponse
#   ...
#   frequency 3 0.1000 0.0000 0.0000 Hartree
#   oke
#   end

# For the intensity dependent refractive index gamma(-omega;omega,omega,-omega)
# use the subkey IDRI
#   aoresponse
#   ...
#   frequency 3 0.1000 0.1000 -0.1000 Hartree
#   idri
#   end

# For the electric field induced second harmonic generation
```

```
# gamma(-2omega;omega,omega,0) use the subkey EFISHG
#   aoresponse
#    ...
#    frequency 3 0.1000 0.1000 0.0000 Hartree
#    efishg
#   end

# For the third harmonic generation gamma(-3omega;omega,omega,omega) use the
# subkey THG
#   aoresponse
#    ...
#    frequency 3 0.1000 0.1000 0.1000 Hartree
#    thg
#   end

# Or in the general case for gamma(-(omega1+omega2+omega3);omega1,omega2,omega3)
# choose three input frequencies omega1, omega2, and omega3
#   aoresponse
#    ...
#    frequency 3 omega1 omega2 omega3 Hartree
#   end
```

Example: Verdet constants: H2O

Download H2O_Verdet.run

```
#! /bin/sh

# Specify the subkey MAGOPTROT in the AORESPONSE key to calculate the Verdet
# constant.


$ADFBIN/adf <<eor
title water

basis
 type TZP
 core None
end

atoms
 O         0.000000    0.134692    0.000000
 H         0.869763   -0.538741    0.000000
 H        -0.869763   -0.538794    0.000000
end

symmetry nosym
allpoints

NumericalQuality Good

xc
 lda vwn
 gga revPBE
end

aoresponse
```

```
 scf   converge 1d-6 iterations 25
 frequency 1 0.088558 Hartree
 ALDA
 giao
 magoptrot
end


eor
```

Example: Dispersion Coefficients: HF

Download Disper_HF.run

```
#! /bin/sh

# General dispersion coefficients (beyond de dipole-dipole C6 interaction
# coefficient) are computed with the auxiliary program DISPER. It uses two
# output files from previous ADF Response calculations. In the example, the two
# ADF runs are one and the same and the relevant TENSOR output file is used
# twice.


$ADFBIN/adf <<EOR
title Van der Waals coefficients HF

atoms
H  0 0  -0.8708056087
F  0 0   0.04619439132
end

Basis
 Type DZP
 Core Small
End

symmetry C(8v)

RESPONSE
  MAXWAALS      8
  VANDERWAALS  7
  ALLTENSOR
  ALLCOMPONENTS
END

EOR

mv TAPE21 HF.t21
rm logfile

# Polarizabilities are computed at 7 (imaginary) frequencies between 0 and
# infinity. The program determines internally the actual frequency values in
# this range to use. The user only specifies the number of them, thereby
# determining the precision of, in fact, a numerical integration over the zero-
# infinity frequency range. A value of 7 is rather low.

# MaxWaals determines that not only the C6 but also C7 and C8 coefficients are
```

```
# computed. A value higher than 8 would not be recommended, because the
# available basis sets would be inadequate for higher coefficients.

# In DISPER calculations the preparatory Response calculation must use the
# AllTensor and AllComponents subkeys.

# The calculation produces a file TENSOR. The subsequent DISPER run uses two
# such files. In this example, both are taken from the same ADF run, copying the
# TENSOR file to, respectively, tensorA and tensorB. These names are prescribed
# for a DISPER calculation.

cp TENSOR tensorA
cp TENSOR tensorB

$ADFBIN/disper -n1 <<eor
eor

# The DISPER program needs no other input than just the files tensorA and
# tensorB, which must both be present as local files. Note the '-n1' flag: this
# enforces that a single-node (non-parallel) run is performed. The current
# implementation does not support parallelization of DISPER, because the kid
# processes may not have the (local to the master!) files tensorA and tensorB.
```

Example: Optical Rotation Dispersion (ORD): DMO

Download DMO_ORD.run

```
#!/bin/sh

# If the subkey OPTICALROTATION is included in the key RESPONSE, the (frequency
# dependent) optical rotation is calculated.

# Note: results will be physically meaningless due to small basis set. purpose
# of this job is to provide a test case for the ORD implementation

# Do not use less strict convergence criteria than default, better to use
# tighter criteria. The approximations in the evaluation of the integrals one
# makes with the linear scaling techniques are effectively switched off by
# setting LINEARSCALING 100 (recommended to use this).


$ADFBIN/adf <<eor
TITLE dimethyloxirane, ORD

COMMENT
 results will be physically meaningless due to small basis set.
 purpose of this job is to provide a test case for the ORD implementation
END

XC
  gga becke perdew
END

Basis
 Type DZP
 Core Small
End
```

```
ATOMS
O       0.000129    1.141417    0.000023
C      -0.597040   -0.094320    0.428262
C       0.596952   -0.094328   -0.428223
H      -0.442927   -0.302650    1.487698
H       0.442944   -0.302474   -1.487720
C      -1.978779   -0.386617   -0.093924
H      -2.723275    0.220579    0.429114
H      -2.043506   -0.157697   -1.159810
H      -2.236045   -1.439970    0.055144
C       1.978716   -0.386693    0.093893
H       2.236030   -1.439985   -0.055498
H       2.723156    0.220701   -0.429005
H       2.043497   -0.158088    1.159845
END


LinearScaling 100
End

response
 allcomponents
 opticalrotation
end

BeckeGrid
 Quality good
End
eor

mv TAPE21 DMO.t21
```

Example: ORD, lifetime effects (key AORESPONSE): DMO

Download DMO_ORD_aoresponse.run

```
#!/bin/sh

# If the subkey OPTICALROTATION is included in the key AORESPONSE, the
# (frequency dependent) optical rotation is calculated. In this example lifetime
# effects are included. This test example consists of two ORD calculations: one
# with and one without the velocity gauge.

# Note: results will be physically meaningless due to small basis set. purpose
# of this job is to provide a test case for the ORD implementation


$ADFBIN/adf <<eor
TITLE dimethyloxirane, ORD

COMMENT
 results will be physically meaningless due to small basis set.
 You may also want to increase the NumericalQuality to "Good".
 purpose of this job is to provide a test case for the ORD implementation
END

XC
```

```
 gga becke perdew
END

Basis
 Type DZP
 Core Small
 CreateOutput none
End

ATOMS
O        0.000129    1.141417    0.000023
C       -0.597040   -0.094320    0.428262
C        0.596952   -0.094328   -0.428223
H       -0.442927   -0.302650    1.487698
H        0.442944   -0.302474   -1.487720
C       -1.978779   -0.386617   -0.093924
H       -2.723275    0.220579    0.429114
H       -2.043506   -0.157697   -1.159810
H       -2.236045   -1.439970    0.055144
C        1.978716   -0.386693    0.093893
H        2.236030   -1.439985   -0.055498
H        2.723156    0.220701   -0.429005
H        2.043497   -0.158088    1.159845
END

allpoints
aoresponse
 ALDA
 opticalrotation
 frequency 1 5893 angstrom
 scf iter 20
 lifetime 0.007
end
NumericalQuality Normal

eor

mv TAPE21 DMO_1.t21
rm -f logfile


# In the second example the subkey OPTICALROTATION of the key AORESPONSE is
# changed into VELOCITYORD:


$ADFBIN/adf <<eor
TITLE dimethyloxirane, ORD with velocity gauge

COMMENT
 results will be physically  meaningless due to small basis set.
 You may also want to increase the NumericalQuality to "Good".
 purpose of this job is to provide a test case for the ORD implementation
END

XC
  gga becke perdew
END
```

```
Basis
 Type DZP
 Core Small
 CreateOutput none
End

ATOMS
O       0.000129    1.141417    0.000023
C      -0.597040   -0.094320    0.428262
C       0.596952   -0.094328   -0.428223
H      -0.442927   -0.302650    1.487698
H       0.442944   -0.302474   -1.487720
C      -1.978779   -0.386617   -0.093924
H      -2.723275    0.220579    0.429114
H      -2.043506   -0.157697   -1.159810
H      -2.236045   -1.439970    0.055144
C       1.978716   -0.386693    0.093893
H       2.236030   -1.439985   -0.055498
H       2.723156    0.220701   -0.429005
H       2.043497   -0.158088    1.159845
END

allpoints
aoresponse
 ALDA
 VelocityORD
 frequency 1 5893 angstrom
 scf iter 20
 lifetime 0.007
end

NumericalQuality Normal

eor

mv TAPE21 DMO_2.t21
```

Example: Polarizability: first order perturbed density

Download ALPHA_DENSITY.run

```
#! /bin/sh

# In this example the polarizability is calculated with AORESONSE.
# If the ALPHA subkey of AORESPONSE is included
# the perturbed density matrix is written to TAPE16.
# densf can use this TAPE16 to calculate the perturbed density on a grid.

$ADFBIN/adf <<eor

Basis
  type TZP
  core None
End

ATOMS
  1 O     0.00000000   -0.06692444    0.00000000
```

```
  2 H     0.76766392    0.53123258    0.00000000
  3 H    -0.76766392    0.53123258    0.00000000
END

SYMMETRY NOSYM

XC
  GGA Becke Perdew
end

BeckeGrid
  Quality verygood
end

aoresponse
  scf  iter 200
  frequency 1 2.2489 ev
  lifetime 0.0037
  alpha
  ALDA
end

eor

$ADFBIN/densf << eor

Grid Inline
-4.0   -3.0   -3.0
-4.0   -3.0   -2.2
-4.0   -3.0   -1.4
-4.0   -3.0   -0.6
-4.0   -3.0   0.2
-4.0   -3.0   1.0
-4.0   -3.0   1.8
-4.0   -3.0   2.6
-4.0   -2.2   -3.0
-4.0   -2.2   -2.2
-4.0   -2.2   -1.4
-4.0   -2.2   -0.6
-4.0   -2.2   0.2
-4.0   -2.2   1.0
-4.0   -2.2   1.8
-4.0   -2.2   2.6
-4.0   -1.4   -3.0
-4.0   -1.4   -2.2
-4.0   -1.4   -1.4
-4.0   -1.4   -0.6
-4.0   -1.4   0.2
-4.0   -1.4   1.0
-4.0   -1.4   1.8
-4.0   -1.4   2.6
-4.0   -0.6   -3.0
-4.0   -0.6   -2.2
-4.0   -0.6   -1.4
-4.0   -0.6   -0.6
-4.0   -0.6   0.2
-4.0   -0.6   1.0
-4.0   -0.6   1.8
```

```
-4.0    -0.6    2.6
-4.0    0.2    -3.0
-4.0    0.2    -2.2
-4.0    0.2    -1.4
-4.0    0.2    -0.6
-4.0    0.2    0.2
-4.0    0.2    1.0
-4.0    0.2    1.8
-4.0    0.2    2.6
-4.0    1.0    -3.0
-4.0    1.0    -2.2
-4.0    1.0    -1.4
-4.0    1.0    -0.6
-4.0    1.0    0.2
-4.0    1.0    1.0
-4.0    1.0    1.8
-4.0    1.0    2.6
-4.0    1.8    -3.0
-4.0    1.8    -2.2
-4.0    1.8    -1.4
-4.0    1.8    -0.6
-4.0    1.8    0.2
-4.0    1.8    1.0
-4.0    1.8    1.8
-4.0    1.8    2.6
-4.0    2.6    -3.0
-4.0    2.6    -2.2
-4.0    2.6    -1.4
-4.0    2.6    -0.6
-4.0    2.6    0.2
-4.0    2.6    1.0
-4.0    2.6    1.8
-4.0    2.6    2.6
-4.0    3.4    -3.0
-4.0    3.4    -2.2
-4.0    3.4    -1.4
-4.0    3.4    -0.6
-4.0    3.4    0.2
-4.0    3.4    1.0
-4.0    3.4    1.8
-4.0    3.4    2.6
-3.2    -3.0    -3.0
-3.2    -3.0    -2.2
-3.2    -3.0    -1.4
-3.2    -3.0    -0.6
-3.2    -3.0    0.2
-3.2    -3.0    1.0
-3.2    -3.0    1.8
-3.2    -3.0    2.6
-3.2    -2.2    -3.0
-3.2    -2.2    -2.2
-3.2    -2.2    -1.4
-3.2    -2.2    -0.6
-3.2    -2.2    0.2
-3.2    -2.2    1.0
-3.2    -2.2    1.8
-3.2    -2.2    2.6
-3.2    -1.4    -3.0
```

```
-3.2   -1.4   -2.2
-3.2   -1.4   -1.4
-3.2   -1.4   -0.6
-3.2   -1.4   0.2
-3.2   -1.4   1.0
-3.2   -1.4   1.8
-3.2   -1.4   2.6
-3.2   -0.6   -3.0
-3.2   -0.6   -2.2
-3.2   -0.6   -1.4
-3.2   -0.6   -0.6
-3.2   -0.6   0.2
-3.2   -0.6   1.0
-3.2   -0.6   1.8
-3.2   -0.6   2.6
-3.2   0.2    -3.0
-3.2   0.2    -2.2
-3.2   0.2    -1.4
-3.2   0.2    -0.6
-3.2   0.2    0.2
-3.2   0.2    1.0
-3.2   0.2    1.8
-3.2   0.2    2.6
-3.2   1.0    -3.0
-3.2   1.0    -2.2
-3.2   1.0    -1.4
-3.2   1.0    -0.6
-3.2   1.0    0.2
-3.2   1.0    1.0
-3.2   1.0    1.8
-3.2   1.0    2.6
-3.2   1.8    -3.0
-3.2   1.8    -2.2
-3.2   1.8    -1.4
-3.2   1.8    -0.6
-3.2   1.8    0.2
-3.2   1.8    1.0
-3.2   1.8    1.8
-3.2   1.8    2.6
-3.2   2.6    -3.0
-3.2   2.6    -2.2
-3.2   2.6    -1.4
-3.2   2.6    -0.6
-3.2   2.6    0.2
-3.2   2.6    1.0
-3.2   2.6    1.8
-3.2   2.6    2.6
-3.2   3.4    -3.0
-3.2   3.4    -2.2
-3.2   3.4    -1.4
-3.2   3.4    -0.6
-3.2   3.4    0.2
-3.2   3.4    1.0
-3.2   3.4    1.8
-3.2   3.4    2.6
-2.4   -3.0   -3.0
-2.4   -3.0   -2.2
-2.4   -3.0   -1.4
```

```
-2.4    -3.0    -0.6
-2.4    -3.0    0.2
-2.4    -3.0    1.0
-2.4    -3.0    1.8
-2.4    -3.0    2.6
-2.4    -2.2    -3.0
-2.4    -2.2    -2.2
-2.4    -2.2    -1.4
-2.4    -2.2    -0.6
-2.4    -2.2    0.2
-2.4    -2.2    1.0
-2.4    -2.2    1.8
-2.4    -2.2    2.6
-2.4    -1.4    -3.0
-2.4    -1.4    -2.2
-2.4    -1.4    -1.4
-2.4    -1.4    -0.6
-2.4    -1.4    0.2
-2.4    -1.4    1.0
-2.4    -1.4    1.8
-2.4    -1.4    2.6
-2.4    -0.6    -3.0
-2.4    -0.6    -2.2
-2.4    -0.6    -1.4
-2.4    -0.6    -0.6
-2.4    -0.6    0.2
-2.4    -0.6    1.0
-2.4    -0.6    1.8
-2.4    -0.6    2.6
-2.4    0.2     -3.0
-2.4    0.2     -2.2
-2.4    0.2     -1.4
-2.4    0.2     -0.6
-2.4    0.2     0.2
-2.4    0.2     1.0
-2.4    0.2     1.8
-2.4    0.2     2.6
-2.4    1.0     -3.0
-2.4    1.0     -2.2
-2.4    1.0     -1.4
-2.4    1.0     -0.6
-2.4    1.0     0.2
-2.4    1.0     1.0
-2.4    1.0     1.8
-2.4    1.0     2.6
-2.4    1.8     -3.0
-2.4    1.8     -2.2
-2.4    1.8     -1.4
-2.4    1.8     -0.6
-2.4    1.8     0.2
-2.4    1.8     1.0
-2.4    1.8     1.8
-2.4    1.8     2.6
-2.4    2.6     -3.0
-2.4    2.6     -2.2
-2.4    2.6     -1.4
-2.4    2.6     -0.6
-2.4    2.6     0.2
```

```
-2.4   2.6   1.0
-2.4   2.6   1.8
-2.4   2.6   2.6
-2.4   3.4  -3.0
-2.4   3.4  -2.2
-2.4   3.4  -1.4
-2.4   3.4  -0.6
-2.4   3.4   0.2
-2.4   3.4   1.0
-2.4   3.4   1.8
-2.4   3.4   2.6
-1.6  -3.0  -3.0
-1.6  -3.0  -2.2
-1.6  -3.0  -1.4
-1.6  -3.0  -0.6
-1.6  -3.0   0.2
-1.6  -3.0   1.0
-1.6  -3.0   1.8
-1.6  -3.0   2.6
-1.6  -2.2  -3.0
-1.6  -2.2  -2.2
-1.6  -2.2  -1.4
-1.6  -2.2  -0.6
-1.6  -2.2   0.2
-1.6  -2.2   1.0
-1.6  -2.2   1.8
-1.6  -2.2   2.6
-1.6  -1.4  -3.0
-1.6  -1.4  -2.2
-1.6  -1.4  -1.4
-1.6  -1.4  -0.6
-1.6  -1.4   0.2
-1.6  -1.4   1.0
-1.6  -1.4   1.8
-1.6  -1.4   2.6
-1.6  -0.6  -3.0
-1.6  -0.6  -2.2
-1.6  -0.6  -1.4
-1.6  -0.6  -0.6
-1.6  -0.6   0.2
-1.6  -0.6   1.0
-1.6  -0.6   1.8
-1.6  -0.6   2.6
-1.6   0.2  -3.0
-1.6   0.2  -2.2
-1.6   0.2  -1.4
-1.6   0.2  -0.6
-1.6   0.2   0.2
-1.6   0.2   1.0
-1.6   0.2   1.8
-1.6   0.2   2.6
-1.6   1.0  -3.0
-1.6   1.0  -2.2
-1.6   1.0  -1.4
-1.6   1.0  -0.6
-1.6   1.0   0.2
-1.6   1.0   1.0
-1.6   1.0   1.8
```

```
-1.6   1.0    2.6
-1.6   1.8   -3.0
-1.6   1.8   -2.2
-1.6   1.8   -1.4
-1.6   1.8   -0.6
-1.6   1.8    0.2
-1.6   1.8    1.0
-1.6   1.8    1.8
-1.6   1.8    2.6
-1.6   2.6   -3.0
-1.6   2.6   -2.2
-1.6   2.6   -1.4
-1.6   2.6   -0.6
-1.6   2.6    0.2
-1.6   2.6    1.0
-1.6   2.6    1.8
-1.6   2.6    2.6
-1.6   3.4   -3.0
-1.6   3.4   -2.2
-1.6   3.4   -1.4
-1.6   3.4   -0.6
-1.6   3.4    0.2
-1.6   3.4    1.0
-1.6   3.4    1.8
-1.6   3.4    2.6
-0.8  -3.0   -3.0
-0.8  -3.0   -2.2
-0.8  -3.0   -1.4
-0.8  -3.0   -0.6
-0.8  -3.0    0.2
-0.8  -3.0    1.0
-0.8  -3.0    1.8
-0.8  -3.0    2.6
-0.8  -2.2   -3.0
-0.8  -2.2   -2.2
-0.8  -2.2   -1.4
-0.8  -2.2   -0.6
-0.8  -2.2    0.2
-0.8  -2.2    1.0
-0.8  -2.2    1.8
-0.8  -2.2    2.6
-0.8  -1.4   -3.0
-0.8  -1.4   -2.2
-0.8  -1.4   -1.4
-0.8  -1.4   -0.6
-0.8  -1.4    0.2
-0.8  -1.4    1.0
-0.8  -1.4    1.8
-0.8  -1.4    2.6
-0.8  -0.6   -3.0
-0.8  -0.6   -2.2
-0.8  -0.6   -1.4
-0.8  -0.6   -0.6
-0.8  -0.6    0.2
-0.8  -0.6    1.0
-0.8  -0.6    1.8
-0.8  -0.6    2.6
-0.8   0.2   -3.0
```

```
-0.8    0.2    -2.2
-0.8    0.2    -1.4
-0.8    0.2    -0.6
-0.8    0.2     0.2
-0.8    0.2     1.0
-0.8    0.2     1.8
-0.8    0.2     2.6
-0.8    1.0    -3.0
-0.8    1.0    -2.2
-0.8    1.0    -1.4
-0.8    1.0    -0.6
-0.8    1.0     0.2
-0.8    1.0     1.0
-0.8    1.0     1.8
-0.8    1.0     2.6
-0.8    1.8    -3.0
-0.8    1.8    -2.2
-0.8    1.8    -1.4
-0.8    1.8    -0.6
-0.8    1.8     0.2
-0.8    1.8     1.0
-0.8    1.8     1.8
-0.8    1.8     2.6
-0.8    2.6    -3.0
-0.8    2.6    -2.2
-0.8    2.6    -1.4
-0.8    2.6    -0.6
-0.8    2.6     0.2
-0.8    2.6     1.0
-0.8    2.6     1.8
-0.8    2.6     2.6
-0.8    3.4    -3.0
-0.8    3.4    -2.2
-0.8    3.4    -1.4
-0.8    3.4    -0.6
-0.8    3.4     0.2
-0.8    3.4     1.0
-0.8    3.4     1.8
-0.8    3.4     2.6
 0.0   -3.0    -3.0
 0.0   -3.0    -2.2
 0.0   -3.0    -1.4
 0.0   -3.0    -0.6
 0.0   -3.0     0.2
 0.0   -3.0     1.0
 0.0   -3.0     1.8
 0.0   -3.0     2.6
 0.0   -2.2    -3.0
 0.0   -2.2    -2.2
 0.0   -2.2    -1.4
 0.0   -2.2    -0.6
 0.0   -2.2     0.2
 0.0   -2.2     1.0
 0.0   -2.2     1.8
 0.0   -2.2     2.6
 0.0   -1.4    -3.0
 0.0   -1.4    -2.2
 0.0   -1.4    -1.4
```

```
0.0    -1.4    -0.6
0.0    -1.4    0.2
0.0    -1.4    1.0
0.0    -1.4    1.8
0.0    -1.4    2.6
0.0    -0.6    -3.0
0.0    -0.6    -2.2
0.0    -0.6    -1.4
0.0    -0.6    -0.6
0.0    -0.6    0.2
0.0    -0.6    1.0
0.0    -0.6    1.8
0.0    -0.6    2.6
0.0    0.2    -3.0
0.0    0.2    -2.2
0.0    0.2    -1.4
0.0    0.2    -0.6
0.0    0.2    0.2
0.0    0.2    1.0
0.0    0.2    1.8
0.0    0.2    2.6
0.0    1.0    -3.0
0.0    1.0    -2.2
0.0    1.0    -1.4
0.0    1.0    -0.6
0.0    1.0    0.2
0.0    1.0    1.0
0.0    1.0    1.8
0.0    1.0    2.6
0.0    1.8    -3.0
0.0    1.8    -2.2
0.0    1.8    -1.4
0.0    1.8    -0.6
0.0    1.8    0.2
0.0    1.8    1.0
0.0    1.8    1.8
0.0    1.8    2.6
0.0    2.6    -3.0
0.0    2.6    -2.2
0.0    2.6    -1.4
0.0    2.6    -0.6
0.0    2.6    0.2
0.0    2.6    1.0
0.0    2.6    1.8
0.0    2.6    2.6
0.0    3.4    -3.0
0.0    3.4    -2.2
0.0    3.4    -1.4
0.0    3.4    -0.6
0.0    3.4    0.2
0.0    3.4    1.0
0.0    3.4    1.8
0.0    3.4    2.6
0.8    -3.0    -3.0
0.8    -3.0    -2.2
0.8    -3.0    -1.4
0.8    -3.0    -0.6
0.8    -3.0    0.2
```

```
0.8   -3.0   1.0
0.8   -3.0   1.8
0.8   -3.0   2.6
0.8   -2.2  -3.0
0.8   -2.2  -2.2
0.8   -2.2  -1.4
0.8   -2.2  -0.6
0.8   -2.2   0.2
0.8   -2.2   1.0
0.8   -2.2   1.8
0.8   -2.2   2.6
0.8   -1.4  -3.0
0.8   -1.4  -2.2
0.8   -1.4  -1.4
0.8   -1.4  -0.6
0.8   -1.4   0.2
0.8   -1.4   1.0
0.8   -1.4   1.8
0.8   -1.4   2.6
0.8   -0.6  -3.0
0.8   -0.6  -2.2
0.8   -0.6  -1.4
0.8   -0.6  -0.6
0.8   -0.6   0.2
0.8   -0.6   1.0
0.8   -0.6   1.8
0.8   -0.6   2.6
0.8    0.2  -3.0
0.8    0.2  -2.2
0.8    0.2  -1.4
0.8    0.2  -0.6
0.8    0.2   0.2
0.8    0.2   1.0
0.8    0.2   1.8
0.8    0.2   2.6
0.8    1.0  -3.0
0.8    1.0  -2.2
0.8    1.0  -1.4
0.8    1.0  -0.6
0.8    1.0   0.2
0.8    1.0   1.0
0.8    1.0   1.8
0.8    1.0   2.6
0.8    1.8  -3.0
0.8    1.8  -2.2
0.8    1.8  -1.4
0.8    1.8  -0.6
0.8    1.8   0.2
0.8    1.8   1.0
0.8    1.8   1.8
0.8    1.8   2.6
0.8    2.6  -3.0
0.8    2.6  -2.2
0.8    2.6  -1.4
0.8    2.6  -0.6
0.8    2.6   0.2
0.8    2.6   1.0
0.8    2.6   1.8
```

14.4. Spectroscopic Properties

```
0.8    2.6    2.6
0.8    3.4    -3.0
0.8    3.4    -2.2
0.8    3.4    -1.4
0.8    3.4    -0.6
0.8    3.4    0.2
0.8    3.4    1.0
0.8    3.4    1.8
0.8    3.4    2.6
1.6    -3.0   -3.0
1.6    -3.0   -2.2
1.6    -3.0   -1.4
1.6    -3.0   -0.6
1.6    -3.0   0.2
1.6    -3.0   1.0
1.6    -3.0   1.8
1.6    -3.0   2.6
1.6    -2.2   -3.0
1.6    -2.2   -2.2
1.6    -2.2   -1.4
1.6    -2.2   -0.6
1.6    -2.2   0.2
1.6    -2.2   1.0
1.6    -2.2   1.8
1.6    -2.2   2.6
1.6    -1.4   -3.0
1.6    -1.4   -2.2
1.6    -1.4   -1.4
1.6    -1.4   -0.6
1.6    -1.4   0.2
1.6    -1.4   1.0
1.6    -1.4   1.8
1.6    -1.4   2.6
1.6    -0.6   -3.0
1.6    -0.6   -2.2
1.6    -0.6   -1.4
1.6    -0.6   -0.6
1.6    -0.6   0.2
1.6    -0.6   1.0
1.6    -0.6   1.8
1.6    -0.6   2.6
1.6    0.2    -3.0
1.6    0.2    -2.2
1.6    0.2    -1.4
1.6    0.2    -0.6
1.6    0.2    0.2
1.6    0.2    1.0
1.6    0.2    1.8
1.6    0.2    2.6
1.6    1.0    -3.0
1.6    1.0    -2.2
1.6    1.0    -1.4
1.6    1.0    -0.6
1.6    1.0    0.2
1.6    1.0    1.0
1.6    1.0    1.8
1.6    1.0    2.6
1.6    1.8    -3.0
```

```
1.6    1.8    -2.2
1.6    1.8    -1.4
1.6    1.8    -0.6
1.6    1.8    0.2
1.6    1.8    1.0
1.6    1.8    1.8
1.6    1.8    2.6
1.6    2.6    -3.0
1.6    2.6    -2.2
1.6    2.6    -1.4
1.6    2.6    -0.6
1.6    2.6    0.2
1.6    2.6    1.0
1.6    2.6    1.8
1.6    2.6    2.6
1.6    3.4    -3.0
1.6    3.4    -2.2
1.6    3.4    -1.4
1.6    3.4    -0.6
1.6    3.4    0.2
1.6    3.4    1.0
1.6    3.4    1.8
1.6    3.4    2.6
2.4    -3.0   -3.0
2.4    -3.0   -2.2
2.4    -3.0   -1.4
2.4    -3.0   -0.6
2.4    -3.0   0.2
2.4    -3.0   1.0
2.4    -3.0   1.8
2.4    -3.0   2.6
2.4    -2.2   -3.0
2.4    -2.2   -2.2
2.4    -2.2   -1.4
2.4    -2.2   -0.6
2.4    -2.2   0.2
2.4    -2.2   1.0
2.4    -2.2   1.8
2.4    -2.2   2.6
2.4    -1.4   -3.0
2.4    -1.4   -2.2
2.4    -1.4   -1.4
2.4    -1.4   -0.6
2.4    -1.4   0.2
2.4    -1.4   1.0
2.4    -1.4   1.8
2.4    -1.4   2.6
2.4    -0.6   -3.0
2.4    -0.6   -2.2
2.4    -0.6   -1.4
2.4    -0.6   -0.6
2.4    -0.6   0.2
2.4    -0.6   1.0
2.4    -0.6   1.8
2.4    -0.6   2.6
2.4    0.2    -3.0
2.4    0.2    -2.2
2.4    0.2    -1.4
```

```
2.4   0.2   -0.6
2.4   0.2   0.2
2.4   0.2   1.0
2.4   0.2   1.8
2.4   0.2   2.6
2.4   1.0   -3.0
2.4   1.0   -2.2
2.4   1.0   -1.4
2.4   1.0   -0.6
2.4   1.0   0.2
2.4   1.0   1.0
2.4   1.0   1.8
2.4   1.0   2.6
2.4   1.8   -3.0
2.4   1.8   -2.2
2.4   1.8   -1.4
2.4   1.8   -0.6
2.4   1.8   0.2
2.4   1.8   1.0
2.4   1.8   1.8
2.4   1.8   2.6
2.4   2.6   -3.0
2.4   2.6   -2.2
2.4   2.6   -1.4
2.4   2.6   -0.6
2.4   2.6   0.2
2.4   2.6   1.0
2.4   2.6   1.8
2.4   2.6   2.6
2.4   3.4   -3.0
2.4   3.4   -2.2
2.4   3.4   -1.4
2.4   3.4   -0.6
2.4   3.4   0.2
2.4   3.4   1.0
2.4   3.4   1.8
2.4   3.4   2.6
3.2   -3.0   -3.0
3.2   -3.0   -2.2
3.2   -3.0   -1.4
3.2   -3.0   -0.6
3.2   -3.0   0.2
3.2   -3.0   1.0
3.2   -3.0   1.8
3.2   -3.0   2.6
3.2   -2.2   -3.0
3.2   -2.2   -2.2
3.2   -2.2   -1.4
3.2   -2.2   -0.6
3.2   -2.2   0.2
3.2   -2.2   1.0
3.2   -2.2   1.8
3.2   -2.2   2.6
3.2   -1.4   -3.0
3.2   -1.4   -2.2
3.2   -1.4   -1.4
3.2   -1.4   -0.6
3.2   -1.4   0.2
```

```
3.2   -1.4   1.0
3.2   -1.4   1.8
3.2   -1.4   2.6
3.2   -0.6   -3.0
3.2   -0.6   -2.2
3.2   -0.6   -1.4
3.2   -0.6   -0.6
3.2   -0.6   0.2
3.2   -0.6   1.0
3.2   -0.6   1.8
3.2   -0.6   2.6
3.2   0.2    -3.0
3.2   0.2    -2.2
3.2   0.2    -1.4
3.2   0.2    -0.6
3.2   0.2    0.2
3.2   0.2    1.0
3.2   0.2    1.8
3.2   0.2    2.6
3.2   1.0    -3.0
3.2   1.0    -2.2
3.2   1.0    -1.4
3.2   1.0    -0.6
3.2   1.0    0.2
3.2   1.0    1.0
3.2   1.0    1.8
3.2   1.0    2.6
3.2   1.8    -3.0
3.2   1.8    -2.2
3.2   1.8    -1.4
3.2   1.8    -0.6
3.2   1.8    0.2
3.2   1.8    1.0
3.2   1.8    1.8
3.2   1.8    2.6
3.2   2.6    -3.0
3.2   2.6    -2.2
3.2   2.6    -1.4
3.2   2.6    -0.6
3.2   2.6    0.2
3.2   2.6    1.0
3.2   2.6    1.8
3.2   2.6    2.6
3.2   3.4    -3.0
3.2   3.4    -2.2
3.2   3.4    -1.4
3.2   3.4    -0.6
3.2   3.4    0.2
3.2   3.4    1.0
3.2   3.4    1.8
3.2   3.4    2.6
4.0   -3.0   -3.0
4.0   -3.0   -2.2
4.0   -3.0   -1.4
4.0   -3.0   -0.6
4.0   -3.0   0.2
4.0   -3.0   1.0
4.0   -3.0   1.8
```

```
4.0    -3.0    2.6
4.0    -2.2   -3.0
4.0    -2.2   -2.2
4.0    -2.2   -1.4
4.0    -2.2   -0.6
4.0    -2.2    0.2
4.0    -2.2    1.0
4.0    -2.2    1.8
4.0    -2.2    2.6
4.0    -1.4   -3.0
4.0    -1.4   -2.2
4.0    -1.4   -1.4
4.0    -1.4   -0.6
4.0    -1.4    0.2
4.0    -1.4    1.0
4.0    -1.4    1.8
4.0    -1.4    2.6
4.0    -0.6   -3.0
4.0    -0.6   -2.2
4.0    -0.6   -1.4
4.0    -0.6   -0.6
4.0    -0.6    0.2
4.0    -0.6    1.0
4.0    -0.6    1.8
4.0    -0.6    2.6
4.0     0.2   -3.0
4.0     0.2   -2.2
4.0     0.2   -1.4
4.0     0.2   -0.6
4.0     0.2    0.2
4.0     0.2    1.0
4.0     0.2    1.8
4.0     0.2    2.6
4.0     1.0   -3.0
4.0     1.0   -2.2
4.0     1.0   -1.4
4.0     1.0   -0.6
4.0     1.0    0.2
4.0     1.0    1.0
4.0     1.0    1.8
4.0     1.0    2.6
4.0     1.8   -3.0
4.0     1.8   -2.2
4.0     1.8   -1.4
4.0     1.8   -0.6
4.0     1.8    0.2
4.0     1.8    1.0
4.0     1.8    1.8
4.0     1.8    2.6
4.0     2.6   -3.0
4.0     2.6   -2.2
4.0     2.6   -1.4
4.0     2.6   -0.6
4.0     2.6    0.2
4.0     2.6    1.0
4.0     2.6    1.8
4.0     2.6    2.6
4.0     3.4   -3.0
```

```
4.0   3.4   -2.2
4.0   3.4   -1.4
4.0   3.4   -0.6
4.0   3.4   0.2
4.0   3.4   1.0
4.0   3.4   1.8
4.0   3.4   2.6
End
Density scf
aoresponse alpha lifetime imag
eor
```

Example: Hyperpolarizabilities of He and H2

Download Hyperpol.run

```
#! /bin/sh



# This sample illustrates the computation of (hyper) polarizability tensors for
# the He atom and the H2 molecule.

# The symmetry is specified, because the Response module in ADF cannot yet
# handle the infinite symmetries ATOM, C(lin), D(lin).


$ADFBIN/adf  -n1 <<EOR
create He  $ADFRESOURCES/Special/Vdiff/He.2s2p2d2f
EOR
mv TAPE21 t21.He8


$ADFBIN/adf  -n1 <<EOR
create H  $ADFRESOURCES/Special/Vdiff/H.2p
EOR
mv TAPE21 t21.H7


$ADFBIN/adf  <<EOR
Title expt geometrie H2(VII),VWN
noprint sfo,frag,functions

Symmetry C(8v)

Atoms
  H 0 0 -0.37305
  H 0 0  0.37305
End

Fragments
  H t21.H7
End

Response
  HyperPol 0.03
  DynaHyp
  AllComponents
```

```
End

EField 0 0 0.001

NumericalQuality Good

EOR

rm TAPE21 logfile

# The Response data block specifies (AllComponents) that not only the (default)
# zz-dipole polarizability is to be computed, but the complete tensor. The
# subkey HyperPol instructs the program to compute hyper*polarizabilities and
# not only polarizabilities. The DynaHyp subkey implies that the *frequency-
# dependent (hyper)polarizability is calculated. In that case the main laser
# frequency has to be specified, in hartree units, after the HyperPol subkey.

# Only the first hyperpolarizability has been implemented in ADF. Some
# information on second hyperpolarizabilities can be obtained from the
# calculation of the first one in a finite field (EFIELD).

# In similar fashion the frequency-dependent hyperpolarizability is computed for
# He, but only the zzz-component because now the AllComponents subkey is
# omitted.


$ADFBIN/adf  <<EOR
Title hyperpolarizability He with the LB94 potential
noprint sfo,frag,functions

Atoms
  He 0 0 0
End

XC
  GGA LB94
END

Fragments
  He t21.He8
End

Response
  HyperPol 0.07
  DynaHyp
End

NumericalQuality Good

EField 0 0 0.001

SCF
 Converge 1e-8
 AccelerationMethod LISTi
End

Symmetry C(8v)
```

```
EOR
```

Example: Damped Verdet constants: Propene

Download DampedVerdet.run

```sh
#!/bin/sh

# Specify the subkey MAGOPTROT in the AORESPONSE key to calculate the Verdet
# constant. Here it is specified together with the LIFETIME key, such that the
# real and imaginary part of the damped Verdet constant will be calculated.


$ADFBIN/adf <<eor
title Propene

ATOMS
C   0.867000     1.441800     3.000000
C   0.849400     2.777300     3.000000
C   2.115500     0.591200     3.000000
H  -0.088300     0.909000     3.000000
H  -0.085900     3.336500     3.000000
H   1.772400     3.363200     3.000000
H   2.737100     0.793300     2.115200
H   1.876900    -0.479100     3.000000
H   2.737100     0.793300     3.884800
END

basis
 type DZP
 core None
end

symmetry nosym
allpoints

BeckeGrid
 quality good
End

XC
 Model SAOP
END

noprint sfo

aoresponse
 scf  converge 1d-5 iterations 25
 frequency 1 0.2 Hartree
 ALDA
 giao
 magoptrot
 lifetime 0.007
end

eor
```

Example: Static magnetizability: H2O

Download H2O_magnet.run

```
#! /bin/sh

# Basic example for a magnetizability calculation.

# One should set iterations=0 for STATIC magnetizability. If one does not use
# SYMMETRY NOSYM, one should set use ALLPOINTS for correct results in
# AORESPONSE. If a line starts with :: it will be skipped during the reading of
# the input.


$ADFBIN/adf <<eor
title H2O magnetizability test

basis
 type DZP
 core None
end

units
 length bohr
 angle degree
end

atoms zmatrix
O 0 0 0 0.        0.     0.
H 1 0 0 1.808846 0.     0.
H 1 2 0 1.808846 104.52 0.
end

xc
 lda
 gga revPBE
end

LinearScaling 100
End

Comment
 New optiond fro AOResponse below
End

allpoints :: needed for correct results in AOResponse

AOResponse
ALDA
:: opticalrotation :: opt. rot. calculated instead of polarizability or
→magnetizability
magneticpert :: needed for magnetizability
:: frequency 5893 Angstroms :: single frequency
:: frequency 2 0.182253 0.197012 Hartree :: select several frequencies
:: freqrange 0.182253 0.25515 25 Hartree :: select freq. range. MUST be 2 or more.
:: components zz :: select xx, yy, zz components instead of whole tensor
scf iterations 0  converge 1e-3 :: set iterations=0 for STATIC magnetizability
:: lifetime 0.0037 :: parameter in Hartree for lifetime broadening / damping
```

```
End


eor

rm -f TAPE21

$ADFBIN/adf <<eor
title H2O magnetizability test

Fragments
 O t21.O
 H t21.H
end

units
 length bohr
 angle degree
end

atoms zmatrix
O 0 0 0 0.        0.     0.
H 1 0 0 1.808846 0.      0.
H 1 2 0 1.808846 104.52 0.
end

LinearScaling 100
End

xc
 lda
 gga revPBE
end

allpoints :: needed for correct results in AOResponse

AOResponse
  ALDA
  magneticpert
  frequency 2 0.182253 0.197012 Hartree
  scf iterations 30  converge 1e-3
End


eor
```

Example: Dynamic magnetizability: H2O

Download H2O_TD_magnet.run

```
#!/bin/sh

# Example for time-dependent magnetizability with GIAOs (Gauge including atomic
# orbitals).


$ADFBIN/adf <<eor
```

```
TITLE H2O, TD magnetizability

COMMENT
 purpose of this job is to calculate time-dependent magnetizability
 with GIAOs
END

basis
 type TZP
 core None
end

ATOMS
 O          0.000000    0.134692    0.000000
 H          0.869763   -0.538741    0.000000
 H         -0.869763   -0.538794    0.000000
END

LinearScaling 100
End

symmetry nosym
allpoints

NumericalQuality good

xc
 lda vwn
 gga revPBE
end

aoresponse
 scf  conv 1d-6 iter 25
 frequency 1 5893 Angstrom
 giao
 ALDA
 magneticpert
 FitAOderiv
end

eor
```

Example: Time-dependent current-density-functional theory: C2H4:

Download C2H4_TDCDFT.run

```
#!/bin/sh



# Calculation of excitation energies and response properties of C2 H4 , with the
# VK functional, thus using time-dependent current-density-functional theory.


$ADFBIN/adf <<eor
title C2H4 excitation energy calculation with the VK functional

ATOMS
```

```
1. C 0.000000    0.000000    0.666318
2. C 0.000000    0.000000   -0.666318
3. H 0.000000    0.928431    1.239388
4. H 0.000000   -0.928431    1.239388
5. H 0.000000    0.928431   -1.239388
6. H 0.000000   -0.928431   -1.239388
END

BASIS
C ET/ET-pVQZ/C
H ET/ET-pVQZ/H
END

EXCITATIONS
END

CURRENTRESPONSE
END

eor

mv TAPE21 C2H4_ET-pVQZ.t21
rm logfile


$ADFBIN/adf <<eor
title C2H4 response calculation with the VK functional

ATOMS
1. C 0.000000    0.000000    0.666318
2. C 0.000000    0.000000   -0.666318
3. H 0.000000    0.928431    1.239388
4. H 0.000000   -0.928431    1.239388
5. H 0.000000    0.928431   -1.239388
6. H 0.000000   -0.928431   -1.239388
END

BASIS
Type TZ2P
END

RESPONSE
ALLCOMPONENTS
END

CURRENTRESPONSE
END

eor


mv TAPE21 C2H4_TZ2P.t21
```

Example: Damped complex polarizabilities with POLTDDFT: Au10

Download `Au10_POLTDDFT.run`

```
#!/bin/sh

$ADFBIN/adf <<eor
Title Au10 DZ GGA LB94

Atoms
Au      -1.44000000        0.00000000       2.88000000
Au       1.44000000       -0.00000000      -2.88000000
Au      -1.44000000        0.00000000      -2.88000000
Au       1.44000000       -0.00000000       2.88000000
Au      -0.00000000       -2.03647000      -1.44000000
Au       0.00000000        2.03647000      -1.44000000
Au       1.44000000       -0.00000000       0.00000000
Au      -0.00000000       -2.03647000       1.44000000
Au      -1.44000000        0.00000000      -0.00000000
Au       0.00000000        2.03647000       1.44000000
End

Basis
 Au ZORA/DZ/REDFIT/Au.4f
End

relativistic scalar zora

XC
 GGA LB94
END

POLTDDFT
 Irrep
   B1.u
   B2.u
   B3.u
 subend
END
eor
```

14.4.4 Ligand Field DFT (LFDFT)

Example: Ligand Field DFT: Co 2+

Download Co_LFDFT.run

```
#!/bin/sh

# Application of the Ligand Field DFT approach for a Co^2+ d^7 electron
# configuration. Here Co is surrounded by 6 Water molecules.

# First an average of configuration calculation (AOC) is performed, where 7
# electrons are equally distributed over the 5 orbitals that have the most
# dominant Co 3d character. Depending on the electron configuration this might
# be a non-trivial task. Symmetry NOSYM should be specified.


SCM_LFDFT="$ADFHOME/examples/adf/Co_LFDFT/LFDFT"
export SCM_LFDFT
```

```
$ADFBIN/adf <<eor
Atoms
 Co        0.000000     0.000000     0.000000
 O         2.113495     0.000000     0.000000
 H         2.687264    -0.787336     0.000000
 H         2.687264     0.787336     0.000000
 O         0.000000    -2.016255     0.000000
 H         0.000000    -2.587300     0.791160
 H         0.000000    -2.587300    -0.791160
 O        -2.113495     0.000000     0.000000
 H        -2.687264    -0.787336     0.000000
 H        -2.687264     0.787336     0.000000
 O         0.000000     2.016255     0.000000
 H         0.000000     2.587300     0.791160
 H         0.000000     2.587300    -0.791160
 O         0.000000     0.000000     2.124849
 H         0.785128     0.000000     2.701737
 H        -0.785128     0.000000     2.701737
 O         0.000000     0.000000    -2.124849
 H         0.785128     0.000000    -2.701737
 H        -0.785128     0.000000    -2.701737
End

Charge 2

IrrepOccupations
  A 56 1.4 1.4 1.4 1.4 1.4
End

Symmetry nosym

Basis
type TZP
core small
End
relativistic scalar zora

XC
gga bp86
End
eor


# When the AOC calculation is ready, you need to make sure that indeed the
# partially occupied orbitals are dominantly d orbitals. In the ADF output you
# can find the character of the MOs in the list of all MOs, ordered by energy,
# with the most significant SFO gross populations.

# Next two LFDFT calculations are performed, first without spin-orbit coupling
# (soc 0), next including spin-orbit coupling (soc 1). In this case there is 1
# shell, and the nlval for 3d is '3 2'. The MO indices should be the
# fractionally occupied levels of the AOC calculation (in this case 29 30 31 32
# 33).


$ADFBIN/lfdft <<eor
```

```
nshell 1
nlval1 3 2
MOind1 29 30 31 32 33
soc    0
eor

$ADFBIN/lfdft <<eor
nshell 1
nlval1 3 2
MOind1 29 30 31 32 33
soc    1
eor
```

Example: Ligan Field DFT: f-d transitions in Pr 3+

Download Pr_LFDFT.run

```
#!/bin/sh

# Application of the Ligand Field DFT approach for the atomic Pr 4f^2 -> Pr 4f^1
# 5d^1 transition.

# First an average of configuration calculation (AOC) is performed for the 4f^2
# electron configuration, where 2 electrons are equally distributed over the 7
# orbitals that have the most dominant Pr 4f character. Depending on the
# electron configuration this might be a non-trivial task.

SCM_LFDFT="$ADFHOME/examples/adf/Pr_LFDFT/LFDFT"
export SCM_LFDFT

$ADFBIN/adf <<eor
 Atoms
  Pr        0.000000    0.000000    0.000000
 End
 IrrepOccupations
   A 54 0.2857143 0.2857143 0.2857143 0.2857143 0.2857143 0.2857143 0.2857143
 End
 Symmetry nosym
 Basis
  type ZORA/TZ2P+
  core none
 End
 relativistic scalar zora
 numericalquality good
eor

# Next the an LFDFT calculation is performed including spin-orbit coupling (soc
# 1). In this case there is 1 shell, and the nlval for 4f is '4 3'. The MO
# indices should be the fractionally occupied levels of the AOC calculation (28
# 29 30 31 32 33 34). Results are added on TAPE21.

$ADFBIN/lfdft <<eor
 nshel 1
 nlval 4 3
 MOind 28 29 30 31 32 33 34
 soc 1
eor
```

```
mv TAPE21 Pr_f2.t21
rm logfile

# Next an average of configuration calculation (AOC) is performed for the 4f^1
# 5d^1 electron configuration, where 1 electron is equally distributed over the
# 7 orbitals that have the most dominant Pr 4f character, and 1 electron is
# equally distributed over the 7 orbitals that have the most dominant Pr 5d
# character. Depending on the electron configuration this might be a non-trivial
# task.

$ADFBIN/adf <<eor
 Atoms
  Pr        0.000000    0.000000    0.000000
 End
 IrrepOccupations
   A 54 0.1428571 0.1428571 0.1428571 0.1428571 0.1428571 0.1428571 0.1428571 0.2 0.2⤶
→0.2 0.2 0.2
 End
 Symmetry nosym
 Basis
  type ZORA/TZ2P+
  core none
 End
 relativistic scalar zora
 numericalquality good
eor

# Next the an LFDFT calculation is performed including spin-orbit coupling for
# both 4f and 5d (soc 1 1). In this case there are 2 shells, and the nlval for
# 4f is '4 3', and for 5d '5 2'. The MO indices should be the fractionally
# occupied levels of the AOC calculation for 4f (28 29 30 31 32 33 34), and for
# 5d (35 36 37 38 39). Results are added on TAPE21.

$ADFBIN/lfdft <<eor
 nshel 2
 nlval1 4 3
 nlval2 5 2
 MOind1 28 29 30 31 32 33 34
 MOind2 35 36 37 38 39
 soc 1 1
eor

mv TAPE21 Pr_f1d1.t21
rm logfile

# Next excitation energies are calculated for exciting from the ground state of
# the Pr 4f^2 -> Pr 4f^1 5d^1. Input for lfdft_tdm are the 2 .t21 files that were
# calculated previously. Results of the excitation energies are added on the
# .t21 file belonging to state2.

$ADFBIN/lfdft_tdm <<eor
 state1 Pr_f2.t21
 state2 Pr_f1d1.t21
eor
```

14.4.5 NMR chemical shifts and spin-spin coupling constants

Example: NMR Chemical Shifts: HBr

Download HBr.run

```sh
#! /bin/sh

# Computation of the NMR chemical shifts for HBr.


# == Non-relativistic ==


$ADFBIN/adf <<eor
TITLE HBr non-relativistic
ATOMS
 1. H   .0000  .0000  .0000
 2. Br  .0000  .0000 1.4140
End
Basis
 Type DZ
 Core Large
End
XC
 GGA Becke Perdew
End
ZlmFit
 Quality good
End
SAVE TAPE10
eor

# The TAPE21 result file and TAPE10 of ADF must be present under those names for
# the NMR calculation
# The NMR program uses mostly only one input (block) key NMR. The subkeys
# specify what output is produced (OUT) and for which Nuclei the NMR data are
# computed and printed (NUC). See the ADF manual.

$ADFBIN/nmr <<eor
NMR
  Out  TENS
  Nuc  1  2
End
eor

mv TAPE21 HBr_nonrel.t21
rm TAPE10 TAPE15 logfile


# == Scalar relativistic ZORA ==


$ADFBIN/adf <<eor
TITLE HBr scalar relativistic ZORA
ATOMS
 1. H   .0000  .0000  .0000
 2. Br  .0000  .0000 1.4140
```

```
End
Basis
 Type DZ
 Core Large
End
XC
 GGA Becke Perdew
End
ZlmFit
 Quality good
End
Relativistic Scalar ZORA
SAVE TAPE10
eor

$ADFBIN/nmr <<eor
NMR
  Out   TENS
  Nuc  1  2
End
eor

mv TAPE21 HBr_scalar_zora.t21
rm TAPE10 TAPE15 logfile


# == Relativistic spin-orbit ZORA ==


# Using Spin-Orbit implies that NOSYM symmetry must be used in the ADF
# calculation: the NMR program cannot handle symmetry calculations in
# combination with spin-orbit terms and will stop with an error message if you
# try to do so.


$ADFBIN/adf <<eor
TITLE HBr relativistic spinorbit ZORA
Atoms
 1. H   .0000  .0000  .0000
 2. Br  .0000  .0000 1.4140
End
Basis
 Type DZ
 Core Large
End
Symmetry NoSYM
XC
 GGA Becke Perdew
End
SAVE TAPE10
ZlmFit
 Quality good
End
Relativistic SpinOrbit ZORA
eor

rm t12.rel
```

```
$ADFBIN/nmr <<eor
 NMR
   U1K BEST
   OUT TENS
   NUC  1 2
 End
eor

mv TAPE21 HBr_SO_zora.t21
rm TAPE10 TAPE15 logfile
```

Example: NMR Chemical Shifts: HgMeBr

Download HgMeBr_pnr.run

```
#!/bin/sh

# NMR data are computed for the 1st and 3rd nucleus only. The UIK subkey is used
# to indicate that certain terms are to be included in the 'U-matrix', which
# goes into the first-order change of the MO's due to the applied magnetic
# field. See the documentation for more information.

# The 'BEST' specification means the best (recommended) options for each
# relativistic option are included for this sub key. In a non-relativistic run
# it has no meaning. In a spin-orbit run it would include the ZORA Spin-Orbit
# terms for a ZORA calculation.

$ADFBIN/adf <<eor
Title   HgMeBr, frozen core
NoPrint SFO, Frag, Functions

Atoms
 Hg  0.0                0.0                   0.0
 Br  0.0                0.0                  -2.406
 C   0.0                0.0                   2.074
 H   1.0289885545161663  0.0                   2.4628220089896287
 H  -0.5144942772580832  0.8911302284144289   2.4628220089896287
 H  -0.5144942772580832 -0.8911302284144289   2.4628220089896287
End

XC
 GGA PW91
End

NumericalQuality Good

Basis
 Type DZ
 Core Small
 C C
End

SAVE TAPE10

eor

$ADFBIN/nmr << eor
```

```
NMR
  OUT ISO
  U1K BEST
  NUC 1 3
END
eor

mv TAPE21 HgMeBr.t21
```

Download HgMeBr_zsc.run

```
#!/bin/sh


$ADFBIN/adf  <<eor
TITLE  HgMeBr, scalar, ZORA

Atoms
 Hg  0.0                 0.0                 0.0
 Br  0.0                 0.0                -2.406
 C   0.0                 0.0                 2.074
 H   1.0289885545161663  0.0                 2.4628220089896287
 H  -0.5144942772580832  0.8911302284144289  2.4628220089896287
 H  -0.5144942772580832 -0.8911302284144289  2.4628220089896287
End

Basis
 Type DZ
 Core None
End

XC
 GGA PW91
End

NumericalQuality Good

Relativistic  Scalar  ZORA

SAVE TAPE10

eor

rm -f t12.rel


$ADFBIN/nmr  <<eor
NMR
  OUT ISO
  U1K BEST
  NUC 1 3
END
eor
mv TAPE21 HgMeBr.t21
```

Download HgMeBr_zso.run

```
#!/bin/sh


$ADFBIN/adf  <<eor
TITLE  HgMeBr, spinorbit, ZORA

Atoms
 Hg  0.0                  0.0                  0.0
 Br  0.0                  0.0                 -2.406
 C   0.0                  0.0                  2.074
 H   1.0289885545161663  0.0                  2.4628220089896287
 H  -0.5144942772580832  0.8911302284144289  2.4628220089896287
 H  -0.5144942772580832 -0.8911302284144289  2.4628220089896287
End

Basis
 Type DZ
 Core None
End

Symmetry  NOSYM

XC
 GGA PW91
End

NumericalQuality Good

Relativistic  SpinOrbit  ZORA

SAVE TAPE10

eor

rm -f t12.rel


$ADFBIN/nmr  <<eor
NMR
  OUT ISO
  U1K BEST
  NUC 1 3
END
eor
mv TAPE21 HgMeBr.t21
```

Example: NMR Chemical Shifts, SAOP potential: CH4

Download CH4_SAOP.run

```
#! /bin/sh

# Computation of the NMR chemical shifts for CH4, with the model potential
# SAOP.

# Important: use SAVE TAPE10. This is necessary for SAOP, since the nmr program
# does not know about SAOP or other model potentials. On TAPE10 the SCF
```

```
# potential is written, which is read in by the nmr program.


# Note: For SAOP one needs an all-electron basis set



$ADFBIN/adf <<eor
xc
 model saop
end

Atoms
 C  0 0 0
 H  0.6264250420707439  0.6264250420707439  0.6264250420707439
 H  0.6264250420707439 -0.6264250420707439 -0.6264250420707439
 H -0.6264250420707439  0.6264250420707439 -0.6264250420707439
 H -0.6264250420707439 -0.6264250420707439  0.6264250420707439
End

Basis
 Type TZ2P
 Core None
End

save TAPE10
eor

$ADFBIN/nmr <<eor
NMR
  Out   TENS
  Nuc  1  2
End
eor

mv TAPE21 CH4.t21
rm TAPE10 logfile
```

Example: NMR Nucleus-independent chemical shifts (NICS): PF3

Download NMR_NICS.run

```
#!/bin/sh

# The NMR program enables the calculation of so-called nucleus-independent
# chemical shifts (NICS). More details are available in the Properties Programs
# User's Guide.

# In the ADF run, the Efield key is used to define points charges with zero
# charge. The GHOSTS key in the nmr program then basically copies this block.
# For the interpretation of the results we refer to the literature.



$ADFBIN/adf <<eor
Title PF3 - scf

Basis
 Type TZP
 Core Large
```

```
End

Symmetry NOSYM

Atoms Z-mat
  1  Xx 0  0  0
  2  P  1  0  0   1.0
  3  F  2  1  0   1.641314   119.702107
  4  F  2  1  3   1.641314   119.702107   120.
  5  F  2  1  3   1.641314   119.702107  -120.
End

PointCharges
    3.0 4.0 5.0 0.0
    1.0 2.0 3.0 0.0
End

NumericalQuality Good

XC
   GGA revPBE
End

SAVE TAPE10

eor


$ADFBIN/nmr <<eor
NMR
  Out Iso Tens
  GHOSTS
     3.0 4.0 5.0
     1.0 2.0 3.0
  SUBEND
END
eor
```

Example: NMR with B3LYP: PF3

Download NMR_B3LYP.run

```
#!/bin/sh

# This example shows how to do hybrid calculation of NMR chemical shifts.

# One needs of course a hybrid functional in the XC block key in ADF. One should
# also use SAVE TAPE10, such that it is an input file in the nmr module.

# In the input for the nmr module one can add the key ZSOAO2007 to approximate
# the effect of spin on the nucleus in the spin-orbit coupled calculations.

# In the last example spin-orbit coupling is included. Symmetry should be NOSYM.


$ADFBIN/adf <<eor
title PF3-NMR-B3LYP
```

```
basis
 type DZP
 core None
end

Atoms
P      0.00000000      0.00000000      1.00000000
F     -0.71283358      1.23466398      1.81325568
F     -0.71283358     -1.23466398      1.81325568
F      1.42566716      0.00000000      1.81325568
End

noprint sfo

xc
 hybrid B3LYP
end

RIHartreeFock
  FitSetQuality Normal
End

NumericalQuality Good

save TAPE10

eor
rm logfile


$ADFBIN/nmr <<eor
NMR
  U1K  BEST
  Out  TENS
  Nuc  1  2
  SCF 1.0e-4
End
eor


mv TAPE21 PF3_1.t21
rm TAPE15 TAPE10
rm logfile


$ADFBIN/adf <<eor
title PF3-NMR-B3LYP ZORA SCALAR

basis
 type DZP
 core None
end

Atoms
P      0.00000000      0.00000000      1.00000000
F     -0.71283358      1.23466398      1.81325568
F     -0.71283358     -1.23466398      1.81325568
```

```
F        1.42566716        0.00000000        1.81325568
End

noprint sfo

xc
 hybrid B3LYP
end

RIHartreeFock
  FitSetQuality Normal
End

RELATIVISTIC SCALAR ZORA

NumericalQuality Good

save TAPE10

eor
rm logfile


$ADFBIN/nmr <<eor
NMR
  U1K  BEST
  Out   TENS
  Nuc  1  2
  SCF 1.0e-4
End
eor

mv TAPE21 PF3_2.t21
rm TAPE15 TAPE10
rm logfile


$ADFBIN/adf <<eor
title PF3-NMR-B3LYP ZORA SPINORBIT

basis
 type DZP
 core None
end

Atoms Z-mat
  1  Xx 0  0  0
  2  P  1  0  0   1.0
  3  F  2  1  0   1.641314   119.702107
  4  F  2  1  3   1.641314   119.702107   120.
  5  F  2  1  3   1.641314   119.702107  -120.
End

symmetry nosym

noprint sfo

xc
```

```
 hybrid B3LYP
end

RIHartreeFock
  FitSetQuality Normal
End

RELATIVISTIC SPINORBIT ZORA

NumericalQuality Good

save TAPE10

eor
rm logfile

cp TAPE21 t21
cp TAPE10 t10

$ADFBIN/nmr <<eor
NMR
  USE  SO1C
  U1K  BEST
  Out  TENS
  Nuc  1  2
  SCF 1.0e-4
End
eor
rm TAPE10 TAPE21 TAPE15 logfile

cp t21 TAPE21
cp t10 TAPE10

$ADFBIN/nmr <<eor
NMR
  U1K  BEST
  Out  TENS
  Nuc  1  2
  SCF 1.0e-4
End
eor
```

Example: NMR Spin-spin coupling constants: C2H2

```
Download CPL_C2H2.run
```

```
#!/bin/sh

# Non-relativistic calculation
# ============================

# A calculation of NMR nuclear spin-spin coupling constants (NSCCs).

# As explained in the ADF manual, the quality of a calculation for spin-spin
# coupling constants, using the program 'CPL', depends largely on the preceding
# ADF calculation, which produces the Kohn-Sham orbitals and orbital energies,
# used as a starting point.
```

```
# One of the quality-determining factors is the chosen basis set. It should be
# sufficiently flexible near the nucleus. Although the all-electron basis TZ2P
# is chosen in this example, it is recommendable to add more functions to the
# basis sets near the nucleus in case of heavy elements. One could start from a
# ZORA/QZ4P basis for example.

# The NOSYM symmetry currently needs to be specified in ADF to enable the CPL
# program to work correctly.


$ADFBIN/adf <<eor
TITLE C2H2 nrel

BASIS
 Type TZ2P
 Core None
END

ATOMS
C 0.0  0.0  0.0
C 0.0  0.0  1.20692
H 0.0  0.0  2.26672
H 0.0  0.0 -1.0598
END

XC
 GGA Becke Perdew
END

BeckeGrid
 Quality good
end

SYMMETRY NOSYM
SAVE TAPE10

eor
rm logfile
cp TAPE10 t10
cp TAPE21 t21


# The CPL program can run in parallel.

# The specification of what needs to be calculated is given in the nmrcoupling
# block key.

# coupling without the SD term
$ADFBIN/cpl <<eor
nmrcoupling
dso
pso
scf converge=1e-7
nuclei 1 2 3 4
nuclei 3 4
end
eor
```

```
rm logfile
cp t10 TAPE10
cp t21 TAPE21


# In this first example, the SD subkey is left out, as this would lead to a very
# strong increase in the required CPU time. The SD subkey is included in the
# second CPL run. That subkey controls the calculation of the so-called spin-
# dipole term.

# The subkeys dso and pso specify that, respectively, the diamagnetic and
# paramagnetic orbital terms will be calculated. The often dominant Fermi
# contact term (FC) is calculated by default and therefore does not have to be
# specified explicitly.

# The scf convergence subkey, in this context, refers to the convergence for the
# solution of the coupled-perturbed Kohn-sham equations which need to be solved
# to obtain to spin-spin couplings.

# The following lines
#    nuclei 1 2 3 4
#    nuclei 3 4
# specify that one coupled-perturbed Kohn-Sham calculation is performed where
# nucleus number 1 (according to the ordering in the ADF output) is the
# perturbing nucleus, and nuclei 2, 3, and 4 are the perturbed nuclei, and
# another coupled-perturbed Kohn-Sham calculation is performed where nucleus 3
# is the perturbing nucleus and nucleus 4 is the perturbed nucleus.

# The second CPL run also includes the spin-dipole (SD) term, through the SD
# subkey.

# The output of the CPL program first contains a lot of general information, a
# summary of the specified input, and then produces the desired numbers:

# It prints separately the different contributions (FC, DSO, PSO, SD) if
# specified in input and sums them up to a total number. Experimental NSCCs
# between two nuclei A and B are usually reported as J(A,B) in Hertz. From a
# computational point of view, the so-called reduced NSCCs K(A,B) are more
# convenient for comparisons. CPL outputs both. In this example, the Fermi-
# contact term is indeed dominant.

# The first part of the output refers to the line
#    nuclei 1 2 3 4
# then the same thing is done for the second similar line where nucleus 3 is the
# perturbing nucleus.

# The output for the second CPL run looks very similar, but now the SD term is
# added to the Fermi contact term, resulting in much longer execution times.


# == Scalar relativistic and spin-orbit calculations ==

# The CPL program also enables calculations using scalar relativistic effects
# (ZORA) and/or spin-orbit effects.

# Schematically, this requires the following changes to the input file with
# respect to a regular spin-orbit calculation and a non-relativistic CPL
# calculation:
```

```
# steep (1s) functions may need to be added to the standard basis sets. the
# full-potential option for ZORA is needed in the create runs and all further
# runs: relativistic zora scalar full the molecular ADF calculation should
# contain the line relativistic zora full spinorbit the CPL input is unmodified
# with respect to the example given here. Please check the ADF manual for
# details on relativistic input options.


# coupling including the SD term

$ADFBIN/cpl <<eor
nmrcoupling
dso
pso
sd
scf converge=1e-7
nuclei 1 2 3 4
nuclei 3 4
end
eor
rm logfile

mv TAPE21 C2H2_1.t21

# redo the same set of calculations, but with a different orientation
# of the molecule this time. Only the CPL calculation without the SD
# term is carried out in order to keep the execution time for this example
# within reasonable limits

rm logfile
rm TAPE10

$ADFBIN/adf <<eor
TITLE C2H2 nrel

BASIS
 Type TZ2P
 Core None
 CreateOutput None
END

ATOMS
C 0.0  0.0     0.0
C 0.0  1.20692 0.0
H 0.0  2.26672 0.0
H 0.0 -1.0598  0.0
END

XC
 GGA Becke Perdew
END

BeckeGrid
 Quality good
end
SAVE TAPE10
```

```
SYMMETRY NOSYM

eor
rm logfile

# coupling without the SD term

$ADFBIN/cpl <<eor
nmrcoupling
dso
pso
scf converge=1e-7
nuclei 1 2 3 4
nuclei 3 4
end
eor
rm logfile

mv TAPE21 C2H2_2.t21
```

Example: NMR Spin-spin coupling constants, hybrid PBE0: HF

Download CPL_HF_hybrid.run

```
#!/bin/sh

# A calculation of NMR nuclear spin-spin coupling constants (NSCCs) for the
# hybrid PBE0.

# The hybrid PBE0 is chosen as exchange-correlation potential in the ADF
# calculation. The key 'usespcode' is required for consistency reasons of the
# PBE0 implementation in ADF and the kernel that is used in the 'CPL' program,
# that calculates NMR spin-spin coupling constants. Symmetry should be NOSYM.
# The basis sets used are specially optimized all-electron basis sets for NMR
# spin-spin coupling calculations (in the directory
# $ADFHOME/atomicdata/ZORA/jcpl), which have extra tight functions, compared to
# a default ADF basis set. The integration accuracy is extra high (Quality
# VeryGood).

$ADFBIN/adf <<eor
   ! experimental bond length
   ATOMS
      F 0.0000 0.0000 0.0000
      H 0.0000 0.0000 0.9170
   END

   BASIS
      Type ZORA/jcpl
      Core None
   END

   usespcode

   XC
      hybrid PBE0
   END
```

```
   SYMMETRY nosym

   BeckeGrid
      Quality verygood
   End

   scf
      converge 1e-8 1e-7
   End

eor


# The first call to cpl is as follows.
# The key 'gga' is included to use the first-order GGA potential instead of the
# first-order VWN potential. The Hartree-Fock part of the kernel is included
# automatically if a hybrid potential is used in the ADF calculation.


$ADFBIN/cpl <<eor
   gga
   nmrcoupling
   dso
   pso
   scf converge=1e-6 iterations=20
      nuclei 1 2
   End
eor


# The second CPL run also includes the spin-dipole (SD) term, through the SD
# subkey, which is much more time-consuming.


$ADFBIN/cpl <<eor
   gga
   nmrcoupling
   dso
   pso
   sd
   scf converge=1e-6 iterations=20
      nuclei 1 2
   End
eor
```

Example: NMR Spin-spin coupling constants, finite nucleus: PbH4

Download PbH4_finitenuc.run

```
#!/bin/sh

# Example for a finite nucleus calculation and the calculation of NMR spin-spin
# coupling constants.

# One of the quality-determining factors for the calculation of NMR coupling
# constants is the chosen basis set, especially one needs enough tight s
```

```
# functions. If one has a large enough basis set in the core region one can see
# an effect of using a finite size of the nucleus instead of a point nucleus,
# especially for heavy nuclei. Such large basis sets can be found for some
# elements in $ADFRESOURCES/ZORA/jcpl, which are basis sets especially designed
# for NMR spin-spin coupling calculations. In this example first a basis set for
# Pb is made which has many tight s functions.


cat <<eor > $PWD/Pb
Pb Basis TZ2P - all steep, plus high exponent fcts. added: iteration 3

BASIS
 1s    39358.2303582207
 1s    20080.7297746024
 1s    10245.2702931645
 1s    5227.17872100229
 1s    2666.92791887872
 1s    1360.67750963200
 1s    694.223219200000
 1s    354.195520000000
 1s    180.712000000000
 2p    39358.2303582207
 2p    20080.7297746024
 2p    10245.2702931645
 2p    5227.17872100229
 2p    2666.92791887872
 2p    1360.67750963200
 2p    694.223219200000
 2p    354.195520000000
 2p    180.712000000000
        1S   92.200
        2S  106.000
        2S   40.200
        3S   51.150
        3S   21.850
        4S   13.400
        4S    9.550
        5S    7.650
        5S    5.200
        6S    3.700
        6S    2.400
        6S    1.580
        2P  196.000
        2P   70.200
        2P   38.300
        3P   21.750
        3P   15.900
        4P   14.500
        4P    9.900
        5P    6.600
        5P    4.200
        6P    2.700
        6P    1.600
        6P     .960
        3D   29.400
        3D   18.800
        4D   13.000
        4D    8.450
```

```
        5D    6.000
        5D    3.650
        5D    2.200
        4F   17.560
        4F    9.950
        4F    5.750
        6D    1.600
        5F    2.500
END

CORE   0 0 0 0
END

DESCRIPTION

END

FIT
 1S   78716.46
 1S   47664.51
 1S   28861.89
 1S   17476.49
 1S   10582.39
 1S    6407.87
 1S    3880.10
 1S    2349.49
 1S    1422.66
 1S     861.45
 1S     521.63
 1S     315.86
 1S     191.26
 1S     115.81
 1S      70.13
 2S      78.58
 2S      53.32
 3S      53.08
 3S      38.26
 4S      36.45
 4S      27.31
 5S      25.48
 5S      19.63
 6S      18.11
 6S      14.24
 7S      13.06
 7S      10.45
 8S       9.55
 8S       7.74
 9S       7.06
 9S       5.79
10S       5.28
10S       4.37
10S       3.62
11S       3.30
11S       2.75
11S       2.30
11S       1.92
 2P 130.50
 3P 105.00
```

```
 4P  83.37
 5P  66.14
 6P  52.66
 6P  35.96
 7P  28.77
 7P  20.17
 8P  16.23
 8P  11.63
 9P   9.41
 9P   6.86
10P   5.58
10P   4.14
11P   3.38
11P   2.54
 3D 111.00
 4D  81.70
 5D  60.58
 6D  45.35
 7D  34.28
 7D  22.74
 8D  17.35
 8D  11.80
 9D   9.08
 9D   6.31
10D   4.89
10D   3.46
 4F 102.15
 5F  63.70
 6F  40.72
 7F  26.59
 8F  17.70
 9F  11.98
 9F   7.34
10F   5.04
10F   3.16
 5G  48.25
 6G  32.57
 7G  22.37
 8G  15.60
 8G   9.73
 9G   6.88
 9G   4.40
END

eor


# This large basis set for Pb is used in ADF calculations on PbH4 and the
# calculation of the NMR spin-spin coupling constants.


$ADFBIN/adf <<eor
UNITS
    length Angstrom
    angle Degree
END

NuclearModel Gaussian
```

```
Print Nuclei

ATOMS
 Pb      0.000000      0.000000      0.000000
  H     -1.023703      1.023703      1.023703
  H      1.023703     -1.023703      1.023703
  H     -1.023703     -1.023703     -1.023703
  H      1.023703      1.023703     -1.023703
END

BASIS
type DZP
core None
Pb $PWD/Pb
END

XC
 gga PBE
END
usespcode

SCF
iterations 100
converge 1.0e-8 1.0e-6
END

BeckeGrid
 quality verygood
End

relativistic scalar zora

SAVE TAPE10

eor

## end scalar run, now do the coupling constant

$ADFBIN/cpl <<eor
gga
Print Nuclei
nmrcoupling
scf converge=1e-5 iterations=25
 nuclei 1 2
end
eor
```

14.4.6 ESR/EPR g-tensor, A-tensor, Q-tensor, ZFS

Example: ESR g-tensor, A-tensor, Q-tensor, D-tensor: HfV

```
Download ESR_HfV.run
```

```
#!/bin/sh

# For the ESR g-tensor and D-tensor (zero-field splitting, ZFS) the effect of
```

```
# spin-orbit coupling is important. For the ESR A-tensor and Q-tensor (EFG)
# spin-orbit coupling is less important.

# In this example first spin-orbit coupling is taken into account
# perturbatively. Next spin-orbit coupling is taken into account self-
# consistent, using the COLLINEAR keyword.

# Note that an all-electron calculation is carried out. This is relevant for the
# computation of the A-tensor, the nuclear magnetic dipole hyperfine
# interaction, where an accurate value of the spin-polarization density at the
# nucleus is important. For the g-tensor this plays a minor role. However, for
# the g-tensor calculation that includes spin-orbit coupling perturbatively, all
# electron basis sets are necessary.

# In the first ADF calculation the A-tensor (block key ESR) is calculated
# without the effect of spin-orbit coupling included. The zero-field splitting
# (key ZFS) is calculated by including spin-orbit coupling perturbatively.


"$ADFBIN/adf" <<eor
Atoms
  Hf  0.0 0.0 0.0
  V   0.0 0.0 2.033
End
ESR
END
Unrestricted
Symmetry NoSym
Charge 0 3
Basis
  Type TZ2P
  Core None
End
ZFS
QTENS
BeckeGrid
 Quality good
End
Relativistic Scalar ZORA
SAVE  TAPE21 TAPE10
eor
cp TAPE21 hfv.t21
cp TAPE10 hfv.t10
rm logfile


# In the next calculation the module nmr calculates the g-tensor (subkey
# GFACTORS) using spin-orbit coupling and the external magnetic field as
# perturbation.


$ADFBIN/nmr <<eor
nmr
 gfactors
 u1k best
 out iso tens
end
eor
```

```
rm TAPE21 TAPE10 logfile TAPE15
cp hfv.t21 TAPE21
cp hfv.t10 TAPE10


# The module cpl can calculate the A-tensor (key HYPERFINE) using spin-orbit
# coupling and the nuclear magnetic field as perturbation. Note that one needs
# to set the SCF convergence criterium to a small value.


$ADFBIN/cpl <<eor
hyperfine
 atoms 1 2 :: calculates A-tensor for atom 1 and 2, input order
 SCF Converge=1e-7
end
eor
rm TAPE21 TAPE10 logfile


# ADF can calculate the g-tensor and A-tensor (block key ESR) using only the
# nuclear or external magnetic field as perturbation, since spin-orbit coupling
# can be taken into account self-consistently. However, in this case, degenerate
# perturbation theory is used. The collinear approximation is used (and symmetry
# NOSYM) to account for spin-polarization effects.


"$ADFBIN/adf" <<eor
Atoms
  Hf  0.0 0.0 0.0
  V   0.0 0.0 2.033
End
ESR
END
QTENS
collinear
Unrestricted
Symmetry NoSym
Basis
  Type TZ2P
  Core None
End
BeckeGrid
 Quality good
End
Relativistic Spinorbit ZORA
eor
```

Example: ESR g-tensor, A-tensor, self consistent spin-orbit coupling: VO

Download VO_collinear.run

```
#! /bin/sh

# The ESR parameters of VO are calculated with the collinear approximation for
# unrestricted Spin-Orbit coupled calculations. In this example the VO-molecule
# has three unpaired electrons.
```

```
# You calculate Electron Spin Resonance properties with the keywords ESR and
# QTENS. ESR is a block-type key and is used to compute the G-tensor or the
# Nuclear Magnetic Dipole Hyperfine interaction. QTENS is a simple key and
# invokes the computation of the Nuclear Electric Quadrupole Hyperfine
# interaction.

# Proper usage of the key ESR requires that you do one of the following:

# A Spin-Orbit calculation, spin-restricted, with exactly one unpaired electron,
# or (b) A Spin-Orbit calculation, spin-unrestricted in the collinear
# approximation, or (c) No Spin-Orbit terms and spin-unrestricted. In case (a)
# and (b) you obtain the G-tensor. In case (b) and (c) you get the Magnetic
# Dipole Hyperfine interaction.

# Note: in case (a) the program also prints a Magnetic Dipole Hyperfine
# interaction data, but these have then been computed without the terms from the
# spin-density at the nucleus. Note: in case (b) and (c) one can have more than
# one unpaired electron. Note: in case (b) one has to use symmetry NOSYM.

# Two calculations are performed:

# Scalar relativistic spin-unrestricted (case c)
# Spin-Orbit relativistic spin-unrestricted collinear (case b)

# After the preliminary calculations (DIRAC, to get the relativistic TAPE12 file
# with relativistic potentials, and the Create runs), we first calculate the
# Dipole Hyperfine interaction: a spin- unrestricted calculation without Spin-
# Orbit coupling.

# Note that one has to use ALLPOINTS in the calculation for a linear molecule to
# get results for the nuclear magnetic dipole hyperfine interaction. For an
# accurate calculation of the hyperfine interaction the numerical quality is set
# to VeryGood.

$ADFBIN/adf  <<eor
Atoms
 V 0 0 0
 O 0 0 1.589
End
XC
 GGA Becke Perdew
End
ESR
End
Qtens
Allpoints
Unrestricted
ZFS
Charge 0 3
Relativistic Scalar ZORA
Basis
 Type TZ2P
 Core None
End
NumericalQuality verygood
eor
```

```
rm TAPE21 logfile

# Then a spin-orbit coupled spin-unrestricted calculation is performed using the
# collinear approximation. Note that symmetry NOSYM is used.

$ADFBIN/adf  <<eor
Atoms
 V 0 0 0
 O 0 0 1.589
End
XC
 GGA Becke Perdew
End
SCF
 AccelerationMethod LISTf
END
ESR
End
Qtens
Symmetry nosym
Unrestricted
Collinear
Relativistic Spinorbit ZORA
Basis
 Type TZ2P
 Core None
End
NumericalQuality verygood
eor

rm TAPE21 logfile
```

Example: ESR g-tensor, A-tensor, perturbative spin-orbit coupling: HgF

Download ESR_HgF_2der.run

```
#!/bin/sh

# This example calculates the ESR g-tensor and A-tensor for HgF. In this example
# first spin-orbit coupling is taken into account perturbatively.

# Note that an all-electron calculation is carried out. This is relevant for the
# computation of the A-tensor, the nuclear magnetic dipole hyperfine
# interaction, where an accurate value of the spin-polarization density at the
# nucleus is important. For the g-tensor this plays a minor role. However, for
# the g-tensor calculation that includes spin-orbit coupling perturbatively, all
# electron basis sets are necessary.

# In the first example the module nmr calculates the g-tensor (subkey GFACTORS)
# using spin-orbit coupling and the external magnetic field as perturbation.


"$ADFBIN/adf" <<eor
Atoms
  Hg        0.0 0.0 0.0
  F         0.0 0.0 2.804
End
```

```
Unrestricted
Symmetry NoSym
Charge 0 1
Basis
  Type TZ2P
  Core None
End
usespcode
XC
  GGA PBE
End
BeckeGrid
 Quality verygood
End
Relativistic Scalar ZORA
SCF
 Iterations 500
 Converge 1e-7 1e-7
END
SAVE  TAPE21 TAPE10
eor

$ADFBIN/nmr <<eor
nmr
 gfactors
 u1k best
 out iso tens
end
eor

rm TAPE21 TAPE10


# In the second example the module cpl calculates the A-tensor (key HYPERFINE)
# using spin-orbit coupling and the nuclear magnetic field as perturbation. Note
# that one needs to set the SCF convergence criterium to a small value. For an
# accurate calculation of the A-tensor one needs a very large basis set in the
# core region (especially tight s-functions), especially for heavy nuclei. If
# one has such a large basis set in the core region, one can also see an effect
# of using a finite size of the nucleus instead of a point nucleus. Such large
# basis sets can be found for some elements in $ADFRESOURCES/ZORA/jcpl, which
# are basis sets especially designed for ESR A-tensor and NMR spin-spin coupling
# calculations.


"$ADFBIN/adf" <<eor
Atoms
  Hg        0.0 0.0 0.0
  F         0.0 0.0 2.804
End
Unrestricted
Symmetry NoSym
Charge 0 1
Basis
  Hg ZORA/jcpl/Hg
  F  ZORA/jcpl/F
End
usespcode
```

```
XC
  GGA PBE
End
BeckeGrid
 Quality verygood
End
NuclearModel Gaussian
Relativistic Scalar ZORA
SCF
 Iterations 500
 Converge 1e-7 1e-7
END
eor

$ADFBIN/cpl <<eor
gga
hyperfine
 atoms 1 2 :: calculates A-tensor for atom 1 and 2, input order
 SCF Converge=1e-7
end
eor
```

Example: ESR spin-restricted and spin-unrestricted: TiF3

```
Download ESR_TiF3.run
```

```
#!/bin/sh

# You calculate Electron Spin Resonance properties with the keywords ESR and
# QTENS. ESR is a block-type key and is used to compute the G-tensor or the
# Nuclear Magnetic Dipole Hyperfine interaction. QTENS is a simple key and
# invokes the computation of the Nuclear Electric Quadrupole Hyperfine
# interaction.

# Proper usage of the key ESR requires that you do one of the following:

# (a) A Spin-Orbit calculation, spin-restricted, with exactly one unpaired
#     electron, or
# (b) A Spin-Orbit calculation, spin-unrestricted in the collinear
#     approximation, or
# (c) No Spin-Orbit terms and spin-unrestricted.

# In case (a) and (b) you obtain the G-tensor. In case (b) and (c) you get the
# Magnetic Dipole Hyperfine interaction.

# Note: in case (a) the program also prints a Magnetic Dipole Hyperfine
# interaction data, but these have then been computed without the terms from the
# spin-density at the nucleus. Note: in case (b) and (c) one can have more than
# one unpaired electron. Note: in case (b) one has to use symmetry NOSYM.

# Five calculations are performed:
# - Scalar relativistic spin-restricted
# - Scalar relativistic open shell spin-restricted
# - Scalar relativistic spin-unrestricted
# - Spin-Orbit relativistic spin-restricted
# - Spin-Orbit relativistic spin-unrestricted collinear
```

```
# First a scalar relativistic spin-restricted calculation is performed. The
# TAPE21 of this calculation is saved as a fragment in the next spin-
# unrestricted calculation, using only 1 SCF iteration, which is a way to get
# the scalar relativistic spin-restricted open shell result for the magnetic
# dipole hyperfine interaction.


$ADFBIN/adf   <<eor
title  TiF3  scalar relativistic restricted
noprint sfo frag functions
Atoms
  Ti    0    0    0
  F   1.780   0    0
  F  -0.89  1.5415252187363007 0
  F  -0.89 -1.5415252187363007 0
End
Basis
 Type TZ2P
 Core None
End
XC
  GGA Becke Perdew
End
relativistic scalar zora
eor

mv TAPE21 t21.TiF3
rm logfile

$ADFBIN/adf   <<eor
title  TiF3  scalar relativistic open shell restricted
noprint sfo frag functions
ESR
End
qtens
Atoms
  Ti    0    0    0 f=TiF3
  F   1.780   0    0 f=TiF3
  F  -0.89  1.5415252187363007 0 f=TiF3
  F  -0.89 -1.5415252187363007 0 f=TiF3
End
Fragments
  TiF3   t21.TiF3
End
XC
  GGA Becke Perdew
End
charge 0 1
unrestricted
scf
 Iterations 0
End
relativistic scalar zora
eor

rm TAPE21 logfile
```

```
# Next a spin-unrestricted SCF calculation is performed to get the scalar
# relativistic spin-unrestricted result for the magnetic dipole hyperfine
# interaction.


$ADFBIN/adf   <<eor
title  TiF3  relativistic open shell unrestricted
noprint sfo frag functions
ESR
End
qtens
Atoms
  Ti    0    0    0 f=TiF3
  F    1.780   0    0 f=TiF3
  F   -0.89  1.5415252187363007 0 f=TiF3
  F   -0.89 -1.5415252187363007 0 f=TiF3
End
Fragments
  TiF3   t21.TiF3
End
XC
  GGA Becke Perdew
End
charge 0 1
unrestricted
relativistic scalar zora
eor

mv TAPE21 TiF3_scalar_zora.t21
rm logfile



# Then, for the same molecule, we compute the G-tensor in a Spin-Orbit run
# (spin-restricted).

# The here-computed and printed Dipole Hyperfine interaction misses the terms
# from the spin-density at the nucleus: compare with the outcomes from the first
# calculation.

# In each of the calculations, the QTENS key invokes the computation of the
# Electric Quadrupole Hyperfine interaction.

# Note that an all-electron calculation is carried out. This is relevant for the
# computation of the A-tensor, the nuclear magnetic dipole hyperfine
# interaction, where an accurate value of the spin-polarization density at the
# nucleus is important. For the G-tensor (and also for the Q-tensor) this plays
# a minor role, but for reasons of consistency both calculations use the same
# basis set and (absence of) frozen core.



$ADFBIN/adf   <<eor
title  TiF3  relativistic spinorbit open shell restricted
noprint sfo frag functions
ESR
End
qtens
Atoms
  Ti    0    0    0 f=TiF3
```

```
  F   1.780   0   0 f=TiF3
  F  -0.89  1.5415252187363007 0 f=TiF3
  F  -0.89 -1.5415252187363007 0 f=TiF3
End
Fragments
  TiF3   t21.TiF3
End
XC
  GGA Becke Perdew
End
relativistic spinorbit zora
eor

mv TAPE21 TiF3_SO_zora.t21
rm logfile


# Finally a spin-orbit coupled spin-unrestricted calculation is performed using
# the collinear approximation. Note that symmetry NOSYM is used.


$ADFBIN/adf   <<eor
title  TiF3  relativistic spinorbit open shell unrestricted collinear
noprint sfo frag functions
ESR
End
qtens
symmetry nosym
unrestricted
collinear
Atoms
  Ti    0    0   0 f=TiF3
  F   1.780   0   0 f=TiF3
  F  -0.89  1.5415252187363007 0 f=TiF3
  F  -0.89 -1.5415252187363007 0 f=TiF3
End
Fragments
  TiF3   t21.TiF3
End
XC
  GGA Becke Perdew
End
relativistic spinorbit zora
eor

mv TAPE21 TiF3_SO_coll.t21
rm logfile
```

Example: ESR, X2C and RA-X2C: PdH

Download PdH_ESR_X2C.run

```
#! /bin/sh

# This example calculates the ESR A-tensor and g-tensor of PdH, using the X2C
# and RA-X2C relativistic methods. Four calculations are performed:
# - Scalar relativistic X2C open shell spin-restricted
```

```
# - Scalar relativistic RA-X2C open shell spin-restricted
# - Spin-Orbit relativistic X2C spin-unrestricted collinear
# - Spin-Orbit relativistic RA-X2C spin-unrestricted collinear

# The spin-orbit coupled spin-unrestricted calculation is performed using the
# collinear approximation, symmetry NOSYM is required in that case. A finite
# nucleaus is used, which will be important for the calculated A-tensor,
# especially for heavy nuclei.

# == Scalar relativistic X2C open shell spin-restricted ==

$ADFBIN/adf <<eor
ATOMS
 Pd 0 0 0
 H  0 0 1.529
END
Basis
 Type QZ4P
 Core None
End
esr
end
nuclearmodel gaussian
unrestricted
charge 0 1
relativistic scalar X2C
eor
rm TAPE21 logfile

# == Scalar relativistic RA-X2C open shell spin-restricted ==

$ADFBIN/adf <<eor
ATOMS
 Pd 0 0 0
 H  0 0 1.529
END
Basis
 Type QZ4P
 Core None
End
esr
end
nuclearmodel gaussian
unrestricted
charge 0 1
relativistic scalar RA-X2C
eor
rm TAPE21 logfile

# == Spin-Orbit relativistic X2C spin-unrestricted collinear ==

$ADFBIN/adf <<eor
ATOMS
 Pd 0 0 0
 H  0 0 1.529
END
Basis
 Type QZ4P
```

```
 Core None
End
esr
end
nuclearmodel gaussian
unrestricted
symmetry nosym
collinear
relativistic spinorbit X2C
eor
rm TAPE21 logfile


# == Spin-Orbit relativistic RA-X2C spin-unrestricted collinear ==

$ADFBIN/adf <<eor
ATOMS
 Pd 0 0 0
 H  0 0 1.529
END
Basis
 Type QZ4P
 Core None
End
esr
end
nuclearmodel gaussian
unrestricted
symmetry nosym
collinear
relativistic spinorbit RA-X2C
eor
rm TAPE21 logfile
```

Example: Zero-field splitting (ZFS), ESR D-tensor: NH

Download NH_ZFS.run

```
#! /bin/sh

# The zero-field splitting (ZFS) can be calculated for open shell molecules with
# electron spin S<=1, using the key ZFS.

# Only the spin-orbit contribution to ZFS is evaluated. Can be used in
# combination with LDA and GGAs. RELATIVISTIC ZORA is also required.


$ADFBIN/adf <<eor
TITLE NH Zero-field splitting

RELATIVISTIC ZORA

ATOMS
   N        0.000000    0.000000    0.007767
   H        0.000000    0.000000   -1.043445
END

BASIS
```

```
   TYPE DZP
END

XC
   GGA BECKE PERDEW
END

UNRESTRICTED

CHARGE 0.0 2.0

ZFS

NumericalQuality Good

eor
```

Example: ZFS D tensor, including direct electron spin-spin part: PhenyInitrene

```
Download Phenylnitrene_ZFS.run
```

```sh
#! /bin/sh

# The zero-field splitting (ZFS) can be calculated for open shell molecules with
# electron spin S<=1, using the key ZFS.

# The direct electron spin-spin part and the spin-orbit contribution to ZFS is
# evaluated. Can be used in combination with LDA and GGAs. RELATIVISTIC ZORA is
# also required. The direct electron spin-spin part is calculated if HARTREEFOCK
# is included as separate keyword. Both Coulomb and (Hartree-Fock like) exchange
# contributions to the direct electron spin-spin term are calculated. In the
# spin-orbit coupling no Hartree-Fock like exchange contributions are included.

$ADFBIN/adf <<eor
Atoms
C  0.000000  0.000000  1.072671
C  0.000000  1.232517  0.336828
C  0.000000  1.216609 -1.046705
C  0.000000  0.000000 -1.747681
C  0.000000 -1.216609 -1.046705
C  0.000000 -1.232517  0.336828
N  0.000000  0.000000  2.400797
H  0.000000  2.164354  0.892961
H  0.000000  2.155064 -1.594614
H  0.000000  0.000000 -2.833691
H  0.000000 -2.155064 -1.594614
H  0.000000 -2.164354  0.892961
End
Basis
 Type TZ2P
 Core None
END
XC
 gga blyp
end
Relativistic ZORA
Charge 0.0 2.0
```

```
Unrestricted
HartreeFock
ZFS
eor
```

14.4.7 EFG, Mössbauer

Example: Mössbauer spectroscopy: Ferrocene

Download Mossbauer.run

```
#! /bin/sh

# By default in ADF the electron density at the nuclei is calculated, no input
# key is required. The electron density at the nuclei could be used for the
# interpretation of isomer shifts in Mossbauer spectroscopy. The absolute
# electron density at a nucleus heavily depends on the accuracy of the basis set
# in the core region of this nucleus, especially if relativistic effects are
# included. Important is to use the same basis set, same exchange correlation
# functional, same numerical accuracy, if electron densities at nuclei in
# different molecules are compared. For the calculation of Mossbauer quadrupole
# splittings the key QTENS is required For 57 Fe quadrupole splittings will be
# written in units of mm/s, used in Mossbauer spectroscopy Example shows a
# calculation on ferrocene with anon-relativistic, and two with a scalar
# relativistic ZORA Hamiltonian using a different all electron basis set.

# First a non-relativistic calculation.

$ADFBIN/adf <<eor
title ferrocene

Atoms
    FE       0.000000    0.000000    0.000000
    C        1.215650    0.000000    1.600813
    C        0.375656   -1.156152    1.600813
    C       -0.983481   -0.714541    1.600813
    C       -0.983481    0.714541    1.600813
    C        0.375656    1.156152    1.600813
    C        1.215650    0.000000   -1.600813
    C        0.375656    1.156152   -1.600813
    C       -0.983481    0.714541   -1.600813
    C       -0.983481   -0.714541   -1.600813
    C        0.375656   -1.156152   -1.600813
    H        2.310827    0.000000    1.629796
    H        0.714085   -2.197727    1.629796
    H       -1.869498   -1.358270    1.629796
    H       -1.869498    1.358270    1.629796
    H        0.714085    2.197727    1.629796
    H        2.310827    0.000000   -1.629796
    H        0.714085    2.197727   -1.629796
    H       -1.869498    1.358270   -1.629796
    H       -1.869498   -1.358270   -1.629796
    H        0.714085   -2.197727   -1.629796
End

xc
```

```
 gga blyp
end

Basis
     Type TZP
     Core none
End

qtens

NumericalQuality Good
exactdensity

eor


# Next the scalar relativistic ZORA calculations. ADF will also calculate the
# quadrupole splittings including the small component density, also called SR
# ZORA-4.


$ADFBIN/adf <<eor
title ferrocene

Atoms
    FE        0.000000    0.000000    0.000000
    C         1.215650    0.000000    1.600813
    C         0.375656   -1.156152    1.600813
    C        -0.983481   -0.714541    1.600813
    C        -0.983481    0.714541    1.600813
    C         0.375656    1.156152    1.600813
    C         1.215650    0.000000   -1.600813
    C         0.375656    1.156152   -1.600813
    C        -0.983481    0.714541   -1.600813
    C        -0.983481   -0.714541   -1.600813
    C         0.375656   -1.156152   -1.600813
    H         2.310827    0.000000    1.629796
    H         0.714085   -2.197727    1.629796
    H        -1.869498   -1.358270    1.629796
    H        -1.869498    1.358270    1.629796
    H         0.714085    2.197727    1.629796
    H         2.310827    0.000000   -1.629796
    H         0.714085    2.197727   -1.629796
    H        -1.869498    1.358270   -1.629796
    H        -1.869498   -1.358270   -1.629796
    H         0.714085   -2.197727   -1.629796
End

xc
 gga blyp
end

Basis
     Type TZP
     Core none
End

qtens
```

```
relativistic scalar zora
NumericalQuality Good
exactdensity


eor


# Next a scalar relativistic calculation is performed with a much larger basis
# set (QZ4P) in the core region. Changing the basis set will have a large effect
# on the electron density at the nucleus and a noticeable effect on the
# calculated quadrupole splittings.


$ADFBIN/adf <<eor
title ferrocene

Atoms
    FE        0.000000    0.000000    0.000000
    C         1.215650    0.000000    1.600813
    C         0.375656   -1.156152    1.600813
    C        -0.983481   -0.714541    1.600813
    C        -0.983481    0.714541    1.600813
    C         0.375656    1.156152    1.600813
    C         1.215650    0.000000   -1.600813
    C         0.375656    1.156152   -1.600813
    C        -0.983481    0.714541   -1.600813
    C        -0.983481   -0.714541   -1.600813
    C         0.375656   -1.156152   -1.600813
    H         2.310827    0.000000    1.629796
    H         0.714085   -2.197727    1.629796
    H        -1.869498   -1.358270    1.629796
    H        -1.869498    1.358270    1.629796
    H         0.714085    2.197727    1.629796
    H         2.310827    0.000000   -1.629796
    H         0.714085    2.197727   -1.629796
    H        -1.869498    1.358270   -1.629796
    H        -1.869498   -1.358270   -1.629796
    H         0.714085   -2.197727   -1.629796
End

xc
 gga blyp
end

Basis
    Type QZ4P
    Core none
End

qtens

relativistic scalar zora
NumericalQuality Good
exactdensity
```

```
eor
```

Example: Mössbauer with X2C: Hg compounds

Download Hg_Mossbauer_X2C.run

```
#! /bin/sh

# This example uses the relativistic spin-orbit coupled X2C method. By default
# in ADF the electron density at the nuclei is calculated, no input key is
# required. The electron density at the nuclei could be used for the
# interpretation of isomer shifts in Mossbauer spectroscopy. Typically one needs
# to perform a fit of the experimentally measured isomer shifts versus the
# calculated electron densities. The absolute electron density at a nucleus
# heavily depends on the accuracy of the basis set in the core region of this
# nucleus, especially if relativistic effects are included. Important is to use
# the same basis set, same exchange correlation functional, same numerical
# accuracy, if electron densities at nuclei in different molecules are compared.
# For the calculation of Mossbauer qadrupole splittings the key QTENS is
# required This example calculates the compounds Hg, HgF, HgF2, HgF4, and HgCl2.


$ADFBIN/adf <<eor
ATOMS
 Hg 0 0 0
END
Basis
Type ZORA/DZ
Core None
End
nuclearmodel gaussian
exactdensity
numericalquality good
relativistic spinorbit X2C
eor
rm TAPE21 logfile


$ADFBIN/adf <<eor
ATOMS
 Hg 0 0 0
 F  0 0 2.007
END
Basis
Type ZORA/DZ
Core None
End
unrestricted
nuclearmodel gaussian
symmetry nosym
noncollinear
exactdensity
numericalquality good
relativistic spinorbit X2C
qtens
eor
rm TAPE21 logfile
```

```
$ADFBIN/adf <<eor
ATOMS
 Hg 0 0  0
 F  0 0  1.914
 F  0 0 -1.914
END
Basis
Type ZORA/DZ
Core None
End
nuclearmodel gaussian
exactdensity
numericalquality good
relativistic spinorbit X2C
qtens
eor
rm TAPE21 logfile


$ADFBIN/adf <<eor
ATOMS
 Hg  0      0      0
 F   0    -1.885 0
 F   0     1.885 0
 F  -1.885 0      0
 F   1.885 0      0
END
Basis
Type ZORA/DZ
Core None
End
nuclearmodel gaussian
exactdensity
numericalquality good
relativistic spinorbit X2C
qtens
eor
rm TAPE21 logfile


$ADFBIN/adf <<eor
ATOMS
 Hg 0 0  0
 Cl 0 0  2.252
 Cl 0 0 -2.252
END
Basis
Type ZORA/DZ
Core None
End
nuclearmodel gaussian
exactdensity
numericalquality good
relativistic spinorbit X2C
qtens
eor
```

```
rm TAPE21 logfile
```

14.5 Transport properties

14.5.1 Charge transfer integrals (transport properties)

Example: Charge transfer integrals: AT base pair

Download AT_transferintegrals.run

```
#! /bin/sh



# ADF can can calculate charge transfer integrals, that are needed in
# approximate methods that model charge transport properties. The molecular
# system typically should be build from 2 fragments. In this example charge
# transfer integrals are calculated between Adenine and Thymine. First these two
# molecules are calculated. In the fragment calculation full symmetry can be
# used. For precision reasons the ZlmFit quality is set to good.



"$ADFBIN/adf" <<eor
TITLE A fragment

ATOMS
 1 N        0.000000000000        0.656191000000        4.473450000000
 2 C        0.000000000000        1.850911000000        5.098850000000
 3 N        0.000000000000        2.094911000000        6.411070000000
 4 C        0.000000000000        0.951291000000        7.115010000000
 5 C        0.000000000000       -0.355699000000        6.611740000000
 6 C        0.000000000000       -0.487619000000        5.203330000000
 7 N        0.000000000000        0.791131000000        8.484350000000
 8 C        0.000000000000       -0.567649000000        8.729290000000
 9 N        0.000000000000       -1.292469000000        7.631450000000
10 N        0.000000000000       -1.672349000000        4.572610000000
11 H        0.000000000000        2.715551000000        4.433920000000
12 H        0.000000000000        1.540301000000        9.166150000000
13 H        0.000000000000       -0.961519000000        9.739820000000
14 H        0.000000000000       -2.515699000000        5.129900000000
15 H        0.000000000000       -1.718459000000        3.541030000000
END

ZlmFit
 Quality good
End

BASIS
 type DZ
 core None
END
eor
mv TAPE21 Adenine.t21

"$ADFBIN/adf" <<eor
```

```
TITLE T fragment

ATOMS
 1 N        0.000000000000       0.617991000000       1.666040000000
 2 C        0.000000000000       1.851251000000       1.046260000000
 3 N        0.000000000000       1.768641000000      -0.347380000000
 4 C        0.000000000000       0.582611000000      -1.042160000000
 5 C        0.000000000000      -0.621999000000      -0.417040000000
 6 C        0.000000000000      -0.627269000000       1.045880000000
 7 O        0.000000000000      -1.670479000000       1.720780000000
 8 O        0.000000000000       2.924531000000       1.636600000000
 9 C        0.000000000000      -1.937039000000      -1.138130000000
10 H        0.000000000000       0.635221000000       2.733380000000
11 H        0.000000000000       2.660141000000      -0.830100000000
12 H        0.000000000000       0.676731000000      -2.127100000000
13 H        0.880180000000      -2.533409000000      -0.860650000000
14 H        0.000000000000      -1.793509000000      -2.225780000000
15 H       -0.880180000000      -2.533409000000      -0.860650000000
END

ZlmFit
 Quality good
End

BASIS
 type DZ
 core None
END
eor
mv TAPE21 Thymine.t21



# Next the the base pair is calculated that consists of Adenine and Thymine. To
# calculate the charge transfer integrals, spatial overlap integrals and site
# energies, include the key TRANSFERINTEGRALS in the input for ADF. Symmetry
# NOSYM should be used.



"$ADFBIN/adf" <<eor
TITLE AT

ATOMS
 1 N        0.000000000000       0.656191000000       4.473450000000  f=Adenine
 2 C        0.000000000000       1.850911000000       5.098850000000  f=Adenine
 3 N        0.000000000000       2.094911000000       6.411070000000  f=Adenine
 4 C        0.000000000000       0.951291000000       7.115010000000  f=Adenine
 5 C        0.000000000000      -0.355699000000       6.611740000000  f=Adenine
 6 C        0.000000000000      -0.487619000000       5.203330000000  f=Adenine
 7 N        0.000000000000       0.791131000000       8.484350000000  f=Adenine
 8 C        0.000000000000      -0.567649000000       8.729290000000  f=Adenine
 9 N        0.000000000000      -1.292469000000       7.631450000000  f=Adenine
10 N        0.000000000000      -1.672349000000       4.572610000000  f=Adenine
11 H        0.000000000000       2.715551000000       4.433920000000  f=Adenine
12 H        0.000000000000       1.540301000000       9.166150000000  f=Adenine
13 H        0.000000000000      -0.961519000000       9.739820000000  f=Adenine
14 H        0.000000000000      -2.515699000000       5.129900000000  f=Adenine
15 H        0.000000000000      -1.718459000000       3.541030000000  f=Adenine
16 N        0.000000000000       0.617991000000       1.666040000000  f=Thymine
```

```
17 C        0.000000000000        1.851251000000        1.046260000000   f=Thymine
18 N        0.000000000000        1.768641000000       -0.347380000000   f=Thymine
19 C        0.000000000000        0.582611000000       -1.042160000000   f=Thymine
20 C        0.000000000000       -0.621999000000       -0.417040000000   f=Thymine
21 C        0.000000000000       -0.627269000000        1.045880000000   f=Thymine
22 O        0.000000000000       -1.670479000000        1.720780000000   f=Thymine
23 O        0.000000000000        2.924531000000        1.636600000000   f=Thymine
24 C        0.000000000000       -1.937039000000       -1.138130000000   f=Thymine
25 H        0.000000000000        0.635221000000        2.733380000000   f=Thymine
26 H        0.000000000000        2.660141000000       -0.830100000000   f=Thymine
27 H        0.000000000000        0.676731000000       -2.127100000000   f=Thymine
28 H        0.880180000000       -2.533409000000       -0.860650000000   f=Thymine
29 H        0.000000000000       -1.793509000000       -2.225780000000   f=Thymine
30 H       -0.880180000000       -2.533409000000       -0.860650000000   f=Thymine
END


ZlmFit
 Quality good
End

Fragments
    Adenine Adenine.t21
    Thymine Thymine.t21
end

SYMMETRY NOSYM
TRANSFERINTEGRALS
eor

# After the calculation has finished in the output one will find the charge
# transfer (overlap integrals and site energies) that are needed to calculate
# hole mobility or electron mobility calculations:

#␣
↪================================================================================
# Electronic coupling V (also known as effective (generalized) transfer integrals J_
↪eff)
# V = (J-S(e1+e2)/2)/(1-S^2)
#
# V for hole transfer:          0.000 eV
# V for electron transfer:     -0.036 eV
#
# The effective transfer integral, or electronic coupling, is calculated from these␣
↪components:
#
# e1(hole) Site energy HOMO fragment 1:      -6.88 eV
# e2(hole) Site energy HOMO fragment 2:      -6.46 eV
# J(hole) Charge transfer integral HOMO fragment 1 - HOMO fragment 2:       0.000 eV
# S(hole) Overlap integral HOMO fragment 1 - HOMO fragment 2:       0.000
#
# e1(electron) Site energy LUMO fragment 1:      -2.24 eV
# e2(electron) Site energy LUMO fragment 2:      -2.62 eV
# J(electron) Charge transfer integral LUMO fragment 1 - LUMO fragment 2:     -0.046␣
↪eV
# S(electron) Overlap integral LUMO fragment 1 - LUMO fragment 2:       0.004
#␣
↪================================================================================
```

Example: Charge transfer integrals with FDE: water dimer

Download ElectronTransfer_FDE_H2O.run

```
#!/bin/sh

# == Expert Option ==

# The electron transfer calculation of a water dimer radical cation in this
# example is aimed at:

# - calculate site energies and couplings of FDE-derived charge-localized states
# - calculate the charge-transfer excitation energy from a two-state model that
#   includes the two charge-localized states involved in the calculation
# - the output also includes a rough evaluation of the error introduced by the
#   density fitting on the site energies and coupling


# First the isolated neutral fragments are obtained. Symmetry NOSYM is used.
# Next in the first FDE calculation the localized state D+A is calculated, which
# means that the first water molecule has charge +1, and the second water
# molecule is neutral. The resulting TAPE21 files must be renamed to fragA1.t21
# and fragA2.t21. In the second FDE calculation the localized state DA+ is
# calculated, now the second water molecule has charge +1, and the first water
# molecule is neutral. The resulting TAPE21 files must be renamed to fragB1.t21
# and fragB2.t21. The FDE freeze and thaw cycle is done manually, and a spin-
# unrestricted calculation is performed.

# The electron transfer calculation is next. The files fragA1.t21, fragA2.t21,
# fragB1.t21, and fragB2.t21 must exist and must have these names. The program
# must be execute in serial mode. Hybrids are not supported. An integration
# parameter of 6 is needed for accuracy.


# == The output of this example is discussed here ==


# ============   Electron Transfer RESULTS ==================

# Electronic Coupling =          0.000000 eV
# Electronic Coupling =         -0.000006 cm-1
# H11-H22             =         -1.396836 eV
# Excitation Energy   =          1.396836 eV
# Overlap             =          0.000000
# H11 H22 H12 =  -152.443044906236  -152.391712133030  -151.743951196449 Eh
# S11 S22 S12 =     0.981761438554     0.980941502465     -0.000000000038


# =========== END Electron Transfer RESULTS ===============

# Due to symmetry, the overlap is almost diagonal (Overlap = 0.00), thus the
# transition density is evaluated with one less electron as explained in Ref.
# [353] in the ADF Manual.

# The electronic coupling between the state with a positive charge localized on
# one water molecule and another with the charge localized on the other water
# molecule is given by 'Electronic Coupling' and is reported in eV and cm^-1.
```

```
# 'H11-H22' is the difference of the site energies in eV. Values of the site
# energies are given by the first two values of 'H11 H22 H12' in atomic units.

# 'Excitation Energy' reports the value of the transfer excitation energy as
# calculated by diagonalization of the 2X2 generalized eigenvalue problem in
# the basis of the charge-localized states, see Refs.[352,353] in the ADF
# Manual.

# 'S11 S22 S12' are the values of the non-normalized overlaps.



# ===================================================
#          OBTAIN ISOLATED FRAGMENTS
# ===================================================


$ADFBIN/adf -n 1  <<eor
Title Isolated water molecule 1

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS


SYMMETRY NOSYM

XC
  GGA PW91
END

SCF
  iterations 90
END

ATOMS
O        0.0000000000        0.0000000000        0.0000000000
H       -0.9358409558         .2646136961        0.0000000000
H       -0.0304663436       -0.9828924420        0.0000000000
END

NumericalQuality good

BASIS
 Type TZP
 Core None
END

eor

mv TAPE21 t21.iso.rho1

$ADFBIN/adf -n 1  <<eor
Title isolated water molecule 2

EPRINT
  SFO NOEIG NOOVL NOORBPOP
```

```
END

NOPRINT BAS FUNCTIONS


SYMMETRY NOSYM

XC
  GGA PW91
END

SCF
  iterations 90
END

ATOMS
 O       0.0000000000      -2.9053396088       0.0000000000
 H      -0.4092227596      -3.3374838250      -0.7701260000
 H      -0.4092227596      -3.3374838250       0.7701260000
END

NumericalQuality good

BASIS
 Type TZP
 Core None
END

eor

mv TAPE21 t21.iso.rho2

# ==================================================
#  FIRST FDE CALCULATION: STATE D+A
#  note: the freeze and thaw is done manually
#        as we have open-shell fragments
# ==================================================


$ADFBIN/adf -n 1 <<eor
Title MODCO: Fragment no.         1; (polarized)

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS


SYMMETRY NOSYM

XC
  GGA PW91
END

SCF
  iterations 90
END
```

```
CHARGE 1 1
UNRESTRICTED

FRAGMENTS
  rho1 t21.iso.rho1
  rho2 t21.iso.rho2 type=fde
END

ATOMS
O         0.0000000000       0.0000000000       0.0000000000    f=rho1
H        -0.9358409558        .2646136961       0.0000000000    f=rho1
H        -0.0304663436      -0.9828924420       0.0000000000    f=rho1
O         0.0000000000      -2.9053396088       0.0000000000    f=rho2
H        -0.4092227596      -3.3374838250      -0.7701260000    f=rho2
H        -0.4092227596      -3.3374838250       0.7701260000    f=rho2
END

NumericalQuality good

ALLOW PARTIALSUPERFRAGS

FDE
  PW91K
  GGAPOTXFD pw91x
  GGAPOTCFD pw91c
END

eor

mv TAPE21 t21.emb.rho1

$ADFBIN/adf -n 1 <<eor
Title MODCO: Fragment no.          2; (polarized)

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS


SYMMETRY NOSYM

XC
  GGA PW91
END

SCF
  iterations 90
END

FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active type=fde
  rho2 t21.iso.rho2
END

ATOMS
```

```
O         0.0000000000        0.0000000000        0.0000000000   f=rho1
H        -0.9358409558         .2646136961        0.0000000000   f=rho1
H        -0.0304663436       -0.9828924420        0.0000000000   f=rho1
O         0.0000000000       -2.9053396088        0.0000000000   f=rho2
H        -0.4092227596       -3.3374838250       -0.7701260000   f=rho2
H        -0.4092227596       -3.3374838250        0.7701260000   f=rho2
END


NumericalQuality good

ALLOW PARTIALSUPERFRAGS

FDE
  PW91K
  GGAPOTXFD pw91x
  GGAPOTCFD pw91c
END

eor

mv TAPE21 t21.emb.rho2


$ADFBIN/adf -n 1 <<eor
Title MODCO: Fragment no.          1; relaxed

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS


SYMMETRY NOSYM

XC
  GGA PW91
END

SCF
  iterations 90
END

CHARGE 1 1
UNRESTRICTED

FRAGMENTS
  rho1 t21.iso.rho1
  rho2 t21.emb.rho2 subfrag=active type=fde
END

restart
  File t21.emb.rho1
End

ATOMS
O         0.0000000000        0.0000000000        0.0000000000   f=rho1
H        -0.9358409558         .2646136961        0.0000000000   f=rho1
```

```
H         -0.0304663436        -0.9828924420         0.0000000000    f=rho1
O          0.0000000000        -2.9053396088         0.0000000000    f=rho2
H         -0.4092227596        -3.3374838250        -0.7701260000    f=rho2
H         -0.4092227596        -3.3374838250         0.7701260000    f=rho2
END

NumericalQuality good

ALLOW PARTIALSUPERFRAGS

FDE
  PW91K
  GGAPOTXFD pw91x
  GGAPOTCFD pw91c
END

eor

mv TAPE21 t21.emb.rho1

$ADFBIN/adf -n 1 <<eor
Title MODCO: Fragment no.        2; relaxed

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS


SYMMETRY NOSYM

XC
  GGA PW91
END

SCF
  iterations 90
END

FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active type=fde
  rho2 t21.emb.rho2 subfrag=active
END

ATOMS
O          0.0000000000         0.0000000000         0.0000000000    f=rho1
H         -0.9358409558          .2646136961         0.0000000000    f=rho1
H         -0.0304663436        -0.9828924420         0.0000000000    f=rho1
O          0.0000000000        -2.9053396088         0.0000000000    f=rho2
H         -0.4092227596        -3.3374838250        -0.7701260000    f=rho2
H         -0.4092227596        -3.3374838250         0.7701260000    f=rho2
END

NumericalQuality good

ALLOW PARTIALSUPERFRAGS
```

```
FDE
  PW91K
  GGAPOTXFD pw91x
  GGAPOTCFD pw91c
END

eor

mv TAPE21 t21.emb.rho2

mv t21.emb.rho1 fragA1.t21
mv t21.emb.rho2 fragA2.t21


# ===================================================
#   SECOND FDE CALCULATION: STATE DA+
#   note: the freeze and thaw is done manually
#         as we have open-shell fragments
#   note: now rho1 is the second fragment
#         as rho2 is the cation
# ===================================================


$ADFBIN/adf -n 1 <<eor
Title MODCO: Fragment no.         2; (polarized)

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS


SYMMETRY NOSYM

XC
  GGA PW91
END

SCF
  iterations 90
END

CHARGE 1 1
UNRESTRICTED

FRAGMENTS
  rho1 t21.iso.rho1 type=fde
  rho2 t21.iso.rho2
END

ATOMS
O       0.0000000000        0.0000000000        0.0000000000    f=rho1
H      -0.9358409558         .2646136961        0.0000000000    f=rho1
H      -0.0304663436       -0.9828924420        0.0000000000    f=rho1
O       0.0000000000       -2.9053396088        0.0000000000    f=rho2
H      -0.4092227596       -3.3374838250       -0.7701260000    f=rho2
H      -0.4092227596       -3.3374838250        0.7701260000    f=rho2
```

```
END

NumericalQuality good

ALLOW PARTIALSUPERFRAGS

FDE
  PW91K
  GGAPOTXFD pw91x
  GGAPOTCFD pw91c
END

eor

mv TAPE21 t21.emb.rho2


$ADFBIN/adf -n 1 <<eor
Title MODCO: Fragment no.          1; (polarized)

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS


SYMMETRY NOSYM

XC
  GGA PW91
END

SCF
  iterations 90
END

FRAGMENTS
  rho1 t21.iso.rho1
  rho2 t21.emb.rho2 subfrag=active type=fde
END

ATOMS
O      0.0000000000      0.0000000000      0.0000000000   f=rho1
H     -0.9358409558       .2646136961      0.0000000000   f=rho1
H     -0.0304663436     -0.9828924420      0.0000000000   f=rho1
O      0.0000000000     -2.9053396088      0.0000000000   f=rho2
H     -0.4092227596     -3.3374838250     -0.7701260000   f=rho2
H     -0.4092227596     -3.3374838250      0.7701260000   f=rho2
END

NumericalQuality good

ALLOW PARTIALSUPERFRAGS

FDE
  PW91K
  GGAPOTXFD pw91x
```

```
  GGAPOTCFD pw91c
END


eor

mv TAPE21 t21.emb.rho1

$ADFBIN/adf -n 1 <<eor
Title MODCO: Fragment no.         2; relaxed

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END


NOPRINT BAS FUNCTIONS


SYMMETRY NOSYM

XC
  GGA PW91
END

SCF
  iterations 90
END

CHARGE 1 1
UNRESTRICTED

restart
  File t21.emb.rho2
End

FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active type=fde
  rho2 t21.iso.rho2
END

ATOMS
O        0.0000000000        0.0000000000        0.0000000000    f=rho1
H       -0.9358409558         .2646136961        0.0000000000    f=rho1
H       -0.0304663436       -0.9828924420        0.0000000000    f=rho1
O        0.0000000000       -2.9053396088        0.0000000000    f=rho2
H       -0.4092227596       -3.3374838250       -0.7701260000    f=rho2
H       -0.4092227596       -3.3374838250        0.7701260000    f=rho2
END

NumericalQuality good

ALLOW PARTIALSUPERFRAGS

FDE
  PW91K
  GGAPOTXFD pw91x
  GGAPOTCFD pw91c
END
```

```
eor

mv TAPE21 t21.emb.rho2

$ADFBIN/adf -n 1 <<eor
Title MODCO: Fragment no.            1; relaxed

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS


SYMMETRY NOSYM

XC
  GGA PW91
END

SCF
  iterations 90
END

FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active
  rho2 t21.emb.rho2 subfrag=active type=fde
END

ATOMS
O        0.0000000000        0.0000000000        0.0000000000    f=rho1
H       -0.9358409558         .2646136961        0.0000000000    f=rho1
H       -0.0304663436       -0.9828924420        0.0000000000    f=rho1
O        0.0000000000       -2.9053396088        0.0000000000    f=rho2
H       -0.4092227596       -3.3374838250       -0.7701260000    f=rho2
H       -0.4092227596       -3.3374838250        0.7701260000    f=rho2
END

NumericalQuality good

ALLOW PARTIALSUPERFRAGS

FDE
  PW91K
  GGAPOTXFD pw91x
  GGAPOTCFD pw91c
END

eor

mv TAPE21 t21.emb.rho1

# =====================================
# NOW THE TAPE21 FILES MUST BE RENAMED
#
#  TO: fragA1.t21, fragA2.t21
#      .... up to a max of fragA999.t21
# AND: fragB1.t21, fragB2.t21
```

```
#     .... up to a max of fragB999.t21
# ====================================

mv t21.emb.rho1 fragB1.t21
mv t21.emb.rho2 fragB2.t21


# ====================================
# Electron Transfer calculation is next
#
# 1) note the ELECTRONTRANSFER block keyword
# 2) the program must be execute in
#    serial mode (parallel mode not
#    supported)
# 3) hybrids not supported
# 4) an integration parameter of 6 is
#    needed for accuracy
# ====================================

$ADFBIN/adf -n 1 <<eor
Title ElectronTransfer calculation

EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END

NOPRINT BAS FUNCTIONS



FRAGMENTS
 rho1 t21.iso.rho1
 rho2 t21.iso.rho2
END

ATOMS
O       0.0000000000       0.0000000000       0.0000000000   f=rho1
H      -0.9358409558        .2646136961       0.0000000000   f=rho1
H      -0.0304663436      -0.9828924420       0.0000000000   f=rho1
O       0.0000000000      -2.9053396088       0.0000000000   f=rho2
H      -0.4092227596      -3.3374838250      -0.7701260000   f=rho2
H      -0.4092227596      -3.3374838250       0.7701260000   f=rho2
END

NumericalQuality good

CHARGE 1 1
UNRESTRICTED

XC
  GGA PW91
END

SYMMETRY nosym

SCF
  iterations 0
END
```

```
ELECTRONTRANSFER
 numfrag 2
END

eor
```

Example: Charge Recombination Calculation of Toluene and TCNE

Download ChargeSeparation_Toluene_TCNE.run

```
#!/bin/sh

# Charge Recombination Calculation of Toluene and TCNE
# tol + TCNE ==> tol+ + TCNE-
# aimed at:

# 1) calculate site energies and couplings of fde-derived neutral states
#    (Diabat 1) and charge-localized states (Diabat 2)
# 2) calculate the charge-transfer excitation energy from a two-state model that
#    includes the two Diabatic states involved in the calculation

# First the isolated neutral fragments are obtained. Symmetry NOSYM is used.
# Resulting files: t21.iso.rho1 and t21.iso.rho2. Next in the first FDE
# calculations: state Toluene (neutral) and TCNE (neutral), resulting files:
# fragA1.t21 and fragA2.t21. The freeze and thaw cycles are done manually since
# open-shell fragments were used. Next in the second FDE calculations: state
# Toluene+ and TCNE-, resulting files fragB1.t21 and fragB2.t21. Again freeze
# and thaw cycles are done manually since open-shell fragments were used.

# In the final charge separation calculation the ELECTRONTRANSFER block keyword
# The program must be executed in serial mode.

#  NOTE: For production-quality results use a larger basis set


# ==================================================
#          OBTAIN ISOLATED FRAGMENTS
# ==================================================

$ADFBIN/adf <<eor
Title Isolated Toluene
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
SYMMETRY NOSYM
XC
  Model SAOP
END
Basis
 Type DZ
 Core none
 CreateOutput none
END
ATOMS
    1 C        -1.07097000000000     -2.32168000000000      0.00000000000000
    2 C        -1.39599000000000     -1.69511000000000      1.20863700000000
```

```
       3 C           -1.39599000000000        -1.69511000000000        -1.20864000000000
       4 H           -0.57394000000000        -3.29447000000000         0.00000000000000
       5 C           -2.04242000000000        -0.45596000000000         1.20579700000000
       6 H           -1.14539000000000        -2.17411000000000         2.15797500000000
       7 C           -2.04242000000000        -0.45596000000000        -1.20580000000000
       8 H           -1.14539000000000        -2.17411000000000        -2.15798000000000
       9 C           -2.38186000000000         0.18220800000000         0.00000000000000
      10 H           -2.29006000000000         0.02569300000000         2.15529900000000
      11 H           -2.29006000000000         0.02569300000000        -2.15530000000000
      12 C           -3.12057000000000         1.49908200000000         0.00000000000000
      13 H           -2.88337000000000         2.09962700000000         0.89190300000000
      14 H           -2.88337000000000         2.09962700000000        -0.89190000000000
      15 H           -4.21273000000000         1.33541500000000         0.00000000000000
END
eor

mv TAPE21 t21.iso.rho1
rm logfile

$ADFBIN/adf <<eor
Title Isolated TCNE
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
SYMMETRY NOSYM
XC
  Model SAOP
END
Basis
 Type DZ
 Core none
 CreateOutput none
END
ATOMS
       1 C            1.83245600000000        -0.14587000000000         0.00000000000000
       2 C            1.20994900000000         1.07690800000000         0.00000000000000
       3 C            2.18809800000000        -0.80630000000000        -1.21971000000000
       4 C            2.18809800000000        -0.80630000000000         1.21971000000000
       5 C            0.88208300000000         1.75218800000000        -1.21935000000000
       6 C            0.88208300000000         1.75218800000000         1.21934500000000
       7 N            2.50290100000000        -1.35237000000000        -2.19775000000000
       8 N            2.50290100000000        -1.35237000000000         2.19774900000000
       9 N            0.61754300000000         2.32819700000000        -2.19523000000000
      10 N            0.61754300000000         2.32819700000000         2.19522600000000
END
eor

mv TAPE21 t21.iso.rho2
rm logfile

# ================================================================
#   FIRST FDE CALCULATION: STATE tol(neutral) and TCNE (neutral)
#   note: the freeze and thaw is done manually
#         as we have open-shell fragments
# ================================================================

$ADFBIN/adf <<eor
```

```
Title tol(neutral), First freeze--and--thaw cycle
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
SYMMETRY NOSYM
XC
  Model SAOP
END
CHARGE 0 0
UNRESTRICTED
FRAGMENTS
  rho1 t21.iso.rho1
  rho2 t21.iso.rho2 type=fde
END
ATOMS
    1 C       -1.07097000000000     -2.32168000000000      0.00000000000000   ␣
↪f=rho1
    2 C       -1.39599000000000     -1.69511000000000      1.20863700000000   ␣
↪f=rho1
    3 C       -1.39599000000000     -1.69511000000000     -1.20864000000000   ␣
↪f=rho1
    4 H       -0.57394000000000     -3.29447000000000      0.00000000000000   ␣
↪f=rho1
    5 C       -2.04242000000000     -0.45596000000000      1.20579700000000   ␣
↪f=rho1
    6 H       -1.14539000000000     -2.17411000000000      2.15797500000000   ␣
↪f=rho1
    7 C       -2.04242000000000     -0.45596000000000     -1.20580000000000   ␣
↪f=rho1
    8 H       -1.14539000000000     -2.17411000000000     -2.15798000000000   ␣
↪f=rho1
    9 C       -2.38186000000000      0.18220800000000      0.00000000000000   ␣
↪f=rho1
   10 H       -2.29006000000000      0.02569300000000      2.15529900000000   ␣
↪f=rho1
   11 H       -2.29006000000000      0.02569300000000     -2.15530000000000   ␣
↪f=rho1
   12 C       -3.12057000000000      1.49908200000000      0.00000000000000   ␣
↪f=rho1
   13 H       -2.88337000000000      2.09962700000000      0.89190300000000   ␣
↪f=rho1
   14 H       -2.88337000000000      2.09962700000000     -0.89190000000000   ␣
↪f=rho1
   15 H       -4.21273000000000      1.33541500000000      0.00000000000000   ␣
↪f=rho1
   16 C        1.83245600000000     -0.14587000000000      0.00000000000000   ␣
↪f=rho2
   17 C        1.20994900000000      1.07690800000000      0.00000000000000   ␣
↪f=rho2
   18 C        2.18809800000000     -0.80630000000000     -1.21971000000000   ␣
↪f=rho2
   19 C        2.18809800000000     -0.80630000000000      1.21971000000000   ␣
↪f=rho2
   20 C        0.88208300000000      1.75218800000000     -1.21935000000000   ␣
↪f=rho2
   21 C        0.88208300000000      1.75218800000000      1.21934500000000   ␣
↪f=rho2
```

```
    22 N        2.50290100000000    -1.35237000000000    -2.19775000000000    ␣
→f=rho2
    23 N        2.50290100000000    -1.35237000000000     2.19774900000000    ␣
→f=rho2
    24 N        0.61754300000000     2.32819700000000    -2.19523000000000    ␣
→f=rho2
    25 N        0.61754300000000     2.32819700000000     2.19522600000000    ␣
→f=rho2
END
ALLOW PARTIALSUPERFRAGS
FDE
 PW91k
 XCNADD PW91
END
eor

mv TAPE21 t21.emb.rho1
rm logfile

$ADFBIN/adf <<eor
Title TCNE(neutral), First freeze--and--thaw cycle
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
SYMMETRY NOSYM
XC
  Model SAOP
END
CHARGE 0 0
UNRESTRICTED
FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active type=fde
  rho2 t21.iso.rho2
END
ATOMS
    1 C        -1.07097000000000    -2.32168000000000     0.00000000000000    ␣
→f=rho1
    2 C        -1.39599000000000    -1.69511000000000     1.20863700000000    ␣
→f=rho1
    3 C        -1.39599000000000    -1.69511000000000    -1.20864000000000    ␣
→f=rho1
    4 H        -0.57394000000000    -3.29447000000000     0.00000000000000    ␣
→f=rho1
    5 C        -2.04242000000000    -0.45596000000000     1.20579700000000    ␣
→f=rho1
    6 H        -1.14539000000000    -2.17411000000000     2.15797500000000    ␣
→f=rho1
    7 C        -2.04242000000000    -0.45596000000000    -1.20580000000000    ␣
→f=rho1
    8 H        -1.14539000000000    -2.17411000000000    -2.15798000000000    ␣
→f=rho1
    9 C        -2.38186000000000     0.18220800000000     0.00000000000000    ␣
→f=rho1
    10 H       -2.29006000000000     0.02569300000000     2.15529900000000    ␣
→f=rho1
    11 H       -2.29006000000000     0.02569300000000    -2.15530000000000    ␣
→f=rho1
```

```
    12 C        -3.12057000000000      1.49908200000000      0.00000000000000    ␣
↪f=rho1
    13 H        -2.88337000000000      2.09962700000000      0.89190300000000    ␣
↪f=rho1
    14 H        -2.88337000000000      2.09962700000000     -0.89190000000000    ␣
↪f=rho1
    15 H        -4.21273000000000      1.33541500000000      0.00000000000000    ␣
↪f=rho1
    16 C         1.83245600000000     -0.14587000000000      0.00000000000000    ␣
↪f=rho2
    17 C         1.20994900000000      1.07690800000000      0.00000000000000    ␣
↪f=rho2
    18 C         2.18809800000000     -0.80630000000000     -1.21971000000000    ␣
↪f=rho2
    19 C         2.18809800000000     -0.80630000000000      1.21971000000000    ␣
↪f=rho2
    20 C         0.88208300000000      1.75218800000000     -1.21935000000000    ␣
↪f=rho2
    21 C         0.88208300000000      1.75218800000000      1.21934500000000    ␣
↪f=rho2
    22 N         2.50290100000000     -1.35237000000000     -2.19775000000000    ␣
↪f=rho2
    23 N         2.50290100000000     -1.35237000000000      2.19774900000000    ␣
↪f=rho2
    24 N         0.61754300000000      2.32819700000000     -2.19523000000000    ␣
↪f=rho2
    25 N         0.61754300000000      2.32819700000000      2.19522600000000    ␣
↪f=rho2
END
ALLOW PARTIALSUPERFRAGS
FDE
 PW91k
 XCNADD PW91
END


eor

mv TAPE21 t21.emb.rho2
rm logfile


it=1
while [ $it -le 2 ]; do

$ADFBIN/adf <<eor
Title tol(neutral), Relaxation cycles
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
SYMMETRY NOSYM
XC
  Model SAOP
END
CHARGE 0 0
UNRESTRICTED
FRAGMENTS
```

```
  rho1 t21.iso.rho1
  rho2 t21.emb.rho2 subfrag=active type=fde
END
RESTART
   File t21.emb.rho1
End
ATOMS
     1 C       -1.07097000000000      -2.32168000000000       0.00000000000000    ␣
↪f=rho1
     2 C       -1.39599000000000      -1.69511000000000       1.20863700000000    ␣
↪f=rho1
     3 C       -1.39599000000000      -1.69511000000000      -1.20864000000000    ␣
↪f=rho1
     4 H       -0.57394000000000      -3.29447000000000       0.00000000000000    ␣
↪f=rho1
     5 C       -2.04242000000000      -0.45596000000000       1.20579700000000    ␣
↪f=rho1
     6 H       -1.14539000000000      -2.17411000000000       2.15797500000000    ␣
↪f=rho1
     7 C       -2.04242000000000      -0.45596000000000      -1.20580000000000    ␣
↪f=rho1
     8 H       -1.14539000000000      -2.17411000000000      -2.15798000000000    ␣
↪f=rho1
     9 C       -2.38186000000000       0.18220800000000       0.00000000000000    ␣
↪f=rho1
    10 H       -2.29006000000000       0.02569300000000       2.15529900000000    ␣
↪f=rho1
    11 H       -2.29006000000000       0.02569300000000      -2.15530000000000    ␣
↪f=rho1
    12 C       -3.12057000000000       1.49908200000000       0.00000000000000    ␣
↪f=rho1
    13 H       -2.88337000000000       2.09962700000000       0.89190300000000    ␣
↪f=rho1
    14 H       -2.88337000000000       2.09962700000000      -0.89190000000000    ␣
↪f=rho1
    15 H       -4.21273000000000       1.33541500000000       0.00000000000000    ␣
↪f=rho1
    16 C        1.83245600000000      -0.14587000000000       0.00000000000000    ␣
↪f=rho2
    17 C        1.20994900000000       1.07690800000000       0.00000000000000    ␣
↪f=rho2
    18 C        2.18809800000000      -0.80630000000000      -1.21971000000000    ␣
↪f=rho2
    19 C        2.18809800000000      -0.80630000000000       1.21971000000000    ␣
↪f=rho2
    20 C        0.88208300000000       1.75218800000000      -1.21935000000000    ␣
↪f=rho2
    21 C        0.88208300000000       1.75218800000000       1.21934500000000    ␣
↪f=rho2
    22 N        2.50290100000000      -1.35237000000000      -2.19775000000000    ␣
↪f=rho2
    23 N        2.50290100000000      -1.35237000000000       2.19774900000000    ␣
↪f=rho2
    24 N        0.61754300000000       2.32819700000000      -2.19523000000000    ␣
↪f=rho2
    25 N        0.61754300000000       2.32819700000000       2.19522600000000    ␣
↪f=rho2
END
```

```
ALLOW PARTIALSUPERFRAGS
FDE
 PW91k
 XCNADD PW91
END
eor

mv TAPE21 t21.emb.rho1
rm logfile

$ADFBIN/adf <<eor
Title TCNE(neutral), Relaxation cycles
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
SYMMETRY NOSYM
XC
  Model SAOP
END
CHARGE 0 0
UNRESTRICTED
FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active type=fde
  rho2 t21.iso.rho2
END
RESTART
   File t21.emb.rho2
End
ATOMS
    1 C        -1.07097000000000    -2.32168000000000     0.00000000000000    ↵
→f=rho1
    2 C        -1.39599000000000    -1.69511000000000     1.20863700000000    ↵
→f=rho1
    3 C        -1.39599000000000    -1.69511000000000    -1.20864000000000    ↵
→f=rho1
    4 H        -0.57394000000000    -3.29447000000000     0.00000000000000    ↵
→f=rho1
    5 C        -2.04242000000000    -0.45596000000000     1.20579700000000    ↵
→f=rho1
    6 H        -1.14539000000000    -2.17411000000000     2.15797500000000    ↵
→f=rho1
    7 C        -2.04242000000000    -0.45596000000000    -1.20580000000000    ↵
→f=rho1
    8 H        -1.14539000000000    -2.17411000000000    -2.15798000000000    ↵
→f=rho1
    9 C        -2.38186000000000     0.18220800000000     0.00000000000000    ↵
→f=rho1
   10 H        -2.29006000000000     0.02569300000000     2.15529900000000    ↵
→f=rho1
   11 H        -2.29006000000000     0.02569300000000    -2.15530000000000    ↵
→f=rho1
   12 C        -3.12057000000000     1.49908200000000     0.00000000000000    ↵
→f=rho1
   13 H        -2.88337000000000     2.09962700000000     0.89190300000000    ↵
→f=rho1
   14 H        -2.88337000000000     2.09962700000000    -0.89190000000000    ↵
→f=rho1
```

```
    15 H        -4.21273000000000     1.33541500000000     0.00000000000000    ␣
↪f=rho1
    16 C         1.83245600000000    -0.14587000000000     0.00000000000000    ␣
↪f=rho2
    17 C         1.20994900000000     1.07690800000000     0.00000000000000    ␣
↪f=rho2
    18 C         2.18809800000000    -0.80630000000000    -1.21971000000000    ␣
↪f=rho2
    19 C         2.18809800000000    -0.80630000000000     1.21971000000000    ␣
↪f=rho2
    20 C         0.88208300000000     1.75218800000000    -1.21935000000000    ␣
↪f=rho2
    21 C         0.88208300000000     1.75218800000000     1.21934500000000    ␣
↪f=rho2
    22 N         2.50290100000000    -1.35237000000000    -2.19775000000000    ␣
↪f=rho2
    23 N         2.50290100000000    -1.35237000000000     2.19774900000000    ␣
↪f=rho2
    24 N         0.61754300000000     2.32819700000000    -2.19523000000000    ␣
↪f=rho2
    25 N         0.61754300000000     2.32819700000000     2.19522600000000    ␣
↪f=rho2
END
ALLOW PARTIALSUPERFRAGS
FDE
 PW91k
 XCNADD PW91
END
eor

mv TAPE21 t21.emb.rho2
rm logfile


it=$(($it+1))
done

mv t21.emb.rho1 fragA1.t21
mv t21.emb.rho2 fragA2.t21


# =====================================================
#   SECOND FDE CALCULATION: STATE tol+ TCNE-
#   note: the freeze and thaw is done manually
#         as we have open-shell fragments
# =====================================================

$ADFBIN/adf <<eor
Title tol+, First freeze--and--thaw cycle
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
SYMMETRY NOSYM
XC
  Model SAOP
END
CHARGE 1 1
```

```
UNRESTRICTED
FRAGMENTS
  rho1 t21.iso.rho1
  rho2 t21.iso.rho2 type=fde
END
ATOMS
    1 C       -1.07097000000000     -2.32168000000000      0.00000000000000    ␣
↪f=rho1
    2 C       -1.39599000000000     -1.69511000000000      1.20863700000000    ␣
↪f=rho1
    3 C       -1.39599000000000     -1.69511000000000     -1.20864000000000    ␣
↪f=rho1
    4 H       -0.57394000000000     -3.29447000000000      0.00000000000000    ␣
↪f=rho1
    5 C       -2.04242000000000     -0.45596000000000      1.20579700000000    ␣
↪f=rho1
    6 H       -1.14539000000000     -2.17411000000000      2.15797500000000    ␣
↪f=rho1
    7 C       -2.04242000000000     -0.45596000000000     -1.20580000000000    ␣
↪f=rho1
    8 H       -1.14539000000000     -2.17411000000000     -2.15798000000000    ␣
↪f=rho1
    9 C       -2.38186000000000      0.18220800000000      0.00000000000000    ␣
↪f=rho1
   10 H       -2.29006000000000      0.02569300000000      2.15529900000000    ␣
↪f=rho1
   11 H       -2.29006000000000      0.02569300000000     -2.15530000000000    ␣
↪f=rho1
   12 C       -3.12057000000000      1.49908200000000      0.00000000000000    ␣
↪f=rho1
   13 H       -2.88337000000000      2.09962700000000      0.89190300000000    ␣
↪f=rho1
   14 H       -2.88337000000000      2.09962700000000     -0.89190000000000    ␣
↪f=rho1
   15 H       -4.21273000000000      1.33541500000000      0.00000000000000    ␣
↪f=rho1
   16 C        1.83245600000000     -0.14587000000000      0.00000000000000    ␣
↪f=rho2
   17 C        1.20994900000000      1.07690800000000      0.00000000000000    ␣
↪f=rho2
   18 C        2.18809800000000     -0.80630000000000     -1.21971000000000    ␣
↪f=rho2
   19 C        2.18809800000000     -0.80630000000000      1.21971000000000    ␣
↪f=rho2
   20 C        0.88208300000000      1.75218800000000     -1.21935000000000    ␣
↪f=rho2
   21 C        0.88208300000000      1.75218800000000      1.21934500000000    ␣
↪f=rho2
   22 N        2.50290100000000     -1.35237000000000     -2.19775000000000    ␣
↪f=rho2
   23 N        2.50290100000000     -1.35237000000000      2.19774900000000    ␣
↪f=rho2
   24 N        0.61754300000000      2.32819700000000     -2.19523000000000    ␣
↪f=rho2
   25 N        0.61754300000000      2.32819700000000      2.19522600000000    ␣
↪f=rho2
END
ALLOW PARTIALSUPERFRAGS
```

```
FDE
 PW91k
 XCNADD PW91
END
eor

mv TAPE21 t21.emb.rho1
rm logfile

$ADFBIN/adf <<eor
Title TCNE-, First freeze--and--thaw cycle
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
SYMMETRY NOSYM
XC
  Model SAOP
END
CHARGE -1 -1
UNRESTRICTED
FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active type=fde
  rho2 t21.iso.rho2
END
ATOMS
    1 C       -1.07097000000000     -2.32168000000000      0.00000000000000    ⮐
↪f=rho1
    2 C       -1.39599000000000     -1.69511000000000      1.20863700000000    ⮐
↪f=rho1
    3 C       -1.39599000000000     -1.69511000000000     -1.20864000000000    ⮐
↪f=rho1
    4 H       -0.57394000000000     -3.29447000000000      0.00000000000000    ⮐
↪f=rho1
    5 C       -2.04242000000000     -0.45596000000000      1.20579700000000    ⮐
↪f=rho1
    6 H       -1.14539000000000     -2.17411000000000      2.15797500000000    ⮐
↪f=rho1
    7 C       -2.04242000000000     -0.45596000000000     -1.20580000000000    ⮐
↪f=rho1
    8 H       -1.14539000000000     -2.17411000000000     -2.15798000000000    ⮐
↪f=rho1
    9 C       -2.38186000000000      0.18220800000000      0.00000000000000    ⮐
↪f=rho1
   10 H       -2.29006000000000      0.02569300000000      2.15529900000000    ⮐
↪f=rho1
   11 H       -2.29006000000000      0.02569300000000     -2.15530000000000    ⮐
↪f=rho1
   12 C       -3.12057000000000      1.49908200000000      0.00000000000000    ⮐
↪f=rho1
   13 H       -2.88337000000000      2.09962700000000      0.89190300000000    ⮐
↪f=rho1
   14 H       -2.88337000000000      2.09962700000000     -0.89190000000000    ⮐
↪f=rho1
   15 H       -4.21273000000000      1.33541500000000      0.00000000000000    ⮐
↪f=rho1
   16 C        1.83245600000000     -0.14587000000000      0.00000000000000    ⮐
↪f=rho2
```

```
    17 C        1.20994900000000      1.07690800000000      0.00000000000000    ␣
→f=rho2
    18 C        2.18809800000000     -0.80630000000000     -1.21971000000000    ␣
→f=rho2
    19 C        2.18809800000000     -0.80630000000000      1.21971000000000    ␣
→f=rho2
    20 C        0.88208300000000      1.75218800000000     -1.21935000000000    ␣
→f=rho2
    21 C        0.88208300000000      1.75218800000000      1.21934500000000    ␣
→f=rho2
    22 N        2.50290100000000     -1.35237000000000     -2.19775000000000    ␣
→f=rho2
    23 N        2.50290100000000     -1.35237000000000      2.19774900000000    ␣
→f=rho2
    24 N        0.61754300000000      2.32819700000000     -2.19523000000000    ␣
→f=rho2
    25 N        0.61754300000000      2.32819700000000      2.19522600000000    ␣
→f=rho2
END
ALLOW PARTIALSUPERFRAGS
FDE
 PW91k
 XCNADD PW91
END
eor


mv TAPE21 t21.emb.rho2
rm logfile



it=1
while [  $it -le 2 ]; do

$ADFBIN/adf <<eor
Title tol+, Relaxation cycles
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
SYMMETRY NOSYM
XC
  Model SAOP
END
CHARGE 1 1
UNRESTRICTED
FRAGMENTS
  rho1 t21.iso.rho1
  rho2 t21.emb.rho2 subfrag=active type=fde
END
RESTART
   File t21.emb.rho1
End
ATOMS
    1 C       -1.07097000000000     -2.32168000000000      0.00000000000000    ␣
→f=rho1
    2 C       -1.39599000000000     -1.69511000000000      1.20863700000000    ␣
→f=rho1
    3 C       -1.39599000000000     -1.69511000000000     -1.20864000000000    ␣
→f=rho1
```

```
     4 H        -0.57394000000000     -3.29447000000000      0.00000000000000    ␣
↪f=rho1
     5 C        -2.04242000000000     -0.45596000000000      1.20579700000000    ␣
↪f=rho1
     6 H        -1.14539000000000     -2.17411000000000      2.15797500000000    ␣
↪f=rho1
     7 C        -2.04242000000000     -0.45596000000000     -1.20580000000000    ␣
↪f=rho1
     8 H        -1.14539000000000     -2.17411000000000     -2.15798000000000    ␣
↪f=rho1
     9 C        -2.38186000000000      0.18220800000000      0.00000000000000    ␣
↪f=rho1
    10 H        -2.29006000000000      0.02569300000000      2.15529900000000    ␣
↪f=rho1
    11 H        -2.29006000000000      0.02569300000000     -2.15530000000000    ␣
↪f=rho1
    12 C        -3.12057000000000      1.49908200000000      0.00000000000000    ␣
↪f=rho1
    13 H        -2.88337000000000      2.09962700000000      0.89190300000000    ␣
↪f=rho1
    14 H        -2.88337000000000      2.09962700000000     -0.89190000000000    ␣
↪f=rho1
    15 H        -4.21273000000000      1.33541500000000      0.00000000000000    ␣
↪f=rho1
    16 C         1.83245600000000     -0.14587000000000      0.00000000000000    ␣
↪f=rho2
    17 C         1.20994900000000      1.07690800000000      0.00000000000000    ␣
↪f=rho2
    18 C         2.18809800000000     -0.80630000000000     -1.21971000000000    ␣
↪f=rho2
    19 C         2.18809800000000     -0.80630000000000      1.21971000000000    ␣
↪f=rho2
    20 C         0.88208300000000      1.75218800000000     -1.21935000000000    ␣
↪f=rho2
    21 C         0.88208300000000      1.75218800000000      1.21934500000000    ␣
↪f=rho2
    22 N         2.50290100000000     -1.35237000000000     -2.19775000000000    ␣
↪f=rho2
    23 N         2.50290100000000     -1.35237000000000      2.19774900000000    ␣
↪f=rho2
    24 N         0.61754300000000      2.32819700000000     -2.19523000000000    ␣
↪f=rho2
    25 N         0.61754300000000      2.32819700000000      2.19522600000000    ␣
↪f=rho2
END
ALLOW PARTIALSUPERFRAGS
FDE
 PW91k
 XCNADD PW91
END
eor

mv TAPE21 t21.emb.rho1
rm logfile

$ADFBIN/adf <<eor
Title TCNE-, Relaxation cycles
EPRINT
```

```
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
SYMMETRY NOSYM
XC
  Model SAOP
END
CHARGE -1 -1
UNRESTRICTED
FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active type=fde
  rho2 t21.iso.rho2
END
RESTART
    File t21.emb.rho2
End
ATOMS
     1 C       -1.07097000000000    -2.32168000000000     0.00000000000000    ⌴
↪f=rho1
     2 C       -1.39599000000000    -1.69511000000000     1.20863700000000    ⌴
↪f=rho1
     3 C       -1.39599000000000    -1.69511000000000    -1.20864000000000    ⌴
↪f=rho1
     4 H       -0.57394000000000    -3.29447000000000     0.00000000000000    ⌴
↪f=rho1
     5 C       -2.04242000000000    -0.45596000000000     1.20579700000000    ⌴
↪f=rho1
     6 H       -1.14539000000000    -2.17411000000000     2.15797500000000    ⌴
↪f=rho1
     7 C       -2.04242000000000    -0.45596000000000    -1.20580000000000    ⌴
↪f=rho1
     8 H       -1.14539000000000    -2.17411000000000    -2.15798000000000    ⌴
↪f=rho1
     9 C       -2.38186000000000     0.18220800000000     0.00000000000000    ⌴
↪f=rho1
    10 H       -2.29006000000000     0.02569300000000     2.15529900000000    ⌴
↪f=rho1
    11 H       -2.29006000000000     0.02569300000000    -2.15530000000000    ⌴
↪f=rho1
    12 C       -3.12057000000000     1.49908200000000     0.00000000000000    ⌴
↪f=rho1
    13 H       -2.88337000000000     2.09962700000000     0.89190300000000    ⌴
↪f=rho1
    14 H       -2.88337000000000     2.09962700000000    -0.89190000000000    ⌴
↪f=rho1
    15 H       -4.21273000000000     1.33541500000000     0.00000000000000    ⌴
↪f=rho1
    16 C        1.83245600000000    -0.14587000000000     0.00000000000000    ⌴
↪f=rho2
    17 C        1.20994900000000     1.07690800000000     0.00000000000000    ⌴
↪f=rho2
    18 C        2.18809800000000    -0.80630000000000    -1.21971000000000    ⌴
↪f=rho2
    19 C        2.18809800000000    -0.80630000000000     1.21971000000000    ⌴
↪f=rho2
    20 C        0.88208300000000     1.75218800000000    -1.21935000000000    ⌴
↪f=rho2
    21 C        0.88208300000000     1.75218800000000     1.21934500000000    ⌴
↪f=rho2
```

```
    22 N        2.50290100000000    -1.35237000000000    -2.19775000000000    ␣
↪f=rho2
    23 N        2.50290100000000    -1.35237000000000     2.19774900000000    ␣
↪f=rho2
    24 N        0.61754300000000     2.32819700000000    -2.19523000000000    ␣
↪f=rho2
    25 N        0.61754300000000     2.32819700000000     2.19522600000000    ␣
↪f=rho2
END
ALLOW PARTIALSUPERFRAGS
FDE
 PW91k
 XCNADD PW91
END
eor


mv TAPE21 t21.emb.rho2
rm logfile



it=$(($it+1))
done



mv t21.emb.rho1 fragB1.t21
mv t21.emb.rho2 fragB2.t21



# =====================================
# Charge Separation calculation is next
#
# 1) note the ELECTRONTRANSFER block keyword
# 2) the program must be execute in
#    serial mode (parallel mode not
#    supported)
# 3) hybrids not supported
# =====================================

$ADFBIN/adf -n 1 <<eor
Title ChargeSeparation Calculation
EPRINT
  SFO NOEIG NOOVL NOORBPOP
END
NOPRINT BAS FUNCTIONS
SYMMETRY NOSYM
XC
  GGA PW91
END
SCF
  iterations 0
END
CHARGE  0 0
UNRESTRICTED
FRAGMENTS
  rho1 t21.iso.rho1
  rho2 t21.iso.rho2
END
ATOMS
```

```
     1 C        -1.07097000000000       -2.32168000000000        0.00000000000000     ␣
→f=rho1
     2 C        -1.39599000000000       -1.69511000000000        1.20863700000000     ␣
→f=rho1
     3 C        -1.39599000000000       -1.69511000000000       -1.20864000000000     ␣
→f=rho1
     4 H        -0.57394000000000       -3.29447000000000        0.00000000000000     ␣
→f=rho1
     5 C        -2.04242000000000       -0.45596000000000        1.20579700000000     ␣
→f=rho1
     6 H        -1.14539000000000       -2.17411000000000        2.15797500000000     ␣
→f=rho1
     7 C        -2.04242000000000       -0.45596000000000       -1.20580000000000     ␣
→f=rho1
     8 H        -1.14539000000000       -2.17411000000000       -2.15798000000000     ␣
→f=rho1
     9 C        -2.38186000000000        0.18220800000000        0.00000000000000     ␣
→f=rho1
    10 H        -2.29006000000000        0.02569300000000        2.15529900000000     ␣
→f=rho1
    11 H        -2.29006000000000        0.02569300000000       -2.15530000000000     ␣
→f=rho1
    12 C        -3.12057000000000        1.49908200000000        0.00000000000000     ␣
→f=rho1
    13 H        -2.88337000000000        2.09962700000000        0.89190300000000     ␣
→f=rho1
    14 H        -2.88337000000000        2.09962700000000       -0.89190000000000     ␣
→f=rho1
    15 H        -4.21273000000000        1.33541500000000        0.00000000000000     ␣
→f=rho1
    16 C         1.83245600000000       -0.14587000000000        0.00000000000000     ␣
→f=rho2
    17 C         1.20994900000000        1.07690800000000        0.00000000000000     ␣
→f=rho2
    18 C         2.18809800000000       -0.80630000000000       -1.21971000000000     ␣
→f=rho2
    19 C         2.18809800000000       -0.80630000000000        1.21971000000000     ␣
→f=rho2
    20 C         0.88208300000000        1.75218800000000       -1.21935000000000     ␣
→f=rho2
    21 C         0.88208300000000        1.75218800000000        1.21934500000000     ␣
→f=rho2
    22 N         2.50290100000000       -1.35237000000000       -2.19775000000000     ␣
→f=rho2
    23 N         2.50290100000000       -1.35237000000000        2.19774900000000     ␣
→f=rho2
    24 N         0.61754300000000        2.32819700000000       -2.19523000000000     ␣
→f=rho2
    25 N         0.61754300000000        2.32819700000000        2.19522600000000     ␣
→f=rho2
END
ELECTRONTRANSFER
 NUMFRAG 2
 Debug
END
eor
```

Example: XCDFT: Charge Separation of an ethylene dimer

Download XCDFT_FDE_ET_ethylene.run

```sh
#!/bin/sh

# ⌴
↪================================================================================================
#
#   Charge Separation Calculation of a ethylene dimer:
#   C2H4* + C2H4 --> C2H4+ + C2H4-
#   aimed at:
#
#   1) calculate site energies and couplings of fde-derived of a localized exciton
#      (Diabat 1) and charge-separated state (Diabat 2). The exciton is obtained by⌴
↪using
#      XCDFT method.
#
#   2) calculate the charge-transfer excitation energy from a two-state model that
#        includes the two charge-localized states involved in the calculation
#
#   First the isolated (restricted) neutral fragments are obtained. Symmetry NOSYM is⌴
↪used.
#   In addition, an isolated unrestricted neutral calculation of the fragment to be⌴
↪excited is performed.
#   Resulting files: t21.iso.rho1, t21.iso.rho2 and gs.t21. Next in the first FDE⌴
↪calculation:
#   state C2H4* (neutral-XCDFT excitation) and C2H4(neutral), resulting files:
#   fragA1.t21 and fragA2.t21. The freeze and thaw cycles are done manually since
#   open-shell fragments were used. Next in the second FDE calculation: state C2H4+⌴
↪and C2H4-,
#   resulting files fragB1.t21 and fragB2.t21. Again freeze and thaw cycles are done⌴
↪manually since
#   open-shell fragments were used.
#
#   In the final charge separation calculation the ELECTRONTRANSFER block keyword is⌴
↪used.
#   The files fragA1.t21, fragA2.t21, fragB1.t21, and fragB2.t21 must exist and must⌴
↪have these names.
# ⌴
↪================================================================================================

# ==================================================
#          OBTAIN ISOLATED FRAGMENTS
# ==================================================


$ADFBIN/adf  << eor
Title MODCO: Fragment no.          1; (isolated)

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS

SYMMETRY NOSYM
```

```
XC
  model SAOP
END

ATOMS
 H         0.930241    1.239503    0.000000
 H        -0.930241    1.239503    0.000000
 C         0.000000    0.667450    0.000000
 C         0.000000   -0.667450    0.000000
 H         0.930241   -1.239503    0.000000
 H        -0.930241   -1.239503    0.000000
END

BASIS
 Type TZP
 Core None
END
eor

rm logfile
mv TAPE21 t21.iso.rho1

$ADFBIN/adf << eor
Title MODCO: Fragment no.        2; (isolated)

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS

SYMMETRY NOSYM

XC
  model SAOP
END

ATOMS
 H         0.930241    1.239503    8.000000
 H        -0.930241    1.239503    8.000000
 C         0.000000    0.667450    8.000000
 C         0.000000   -0.667450    8.000000
 H         0.930241   -1.239503    8.000000
 H        -0.930241   -1.239503    8.000000
END

BASIS
 Type TZP
 Core None
END
eor

rm logfile
mv TAPE21 t21.iso.rho2

# ====================================================
#       OBTAIN ISOLATED UNRESTRICTED FRAGMENT
# ====================================================
```

```
$ADFBIN/adf  << eor
Title  Fragment 1; (isolated_unrestricted)

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS

SYMMETRY NOSYM

XC
  model SAOP
END

CHARGE 0 0
UNRESTRICTED

ATOMS
 H        0.930241    1.239503    0.000000
 H       -0.930241    1.239503    0.000000
 C        0.000000    0.667450    0.000000
 C        0.000000   -0.667450    0.000000
 H        0.930241   -1.239503    0.000000
 H       -0.930241   -1.239503    0.000000
END

BASIS
 Type TZP
 Core None
END
eor

rm logfile

# ===================================================
#    RENAME TAPE21 file as gs.t21, as required in
#    XCDFT methodology.
# ===================================================
mv TAPE21 gs.t21


# ===================================================
#   FIRST FDE CALCULATION: STATE D* (XCDFT) + A
#   note: the freeze and thaw is done manually
#         as we have open-shell fragments
# ===================================================

$ADFBIN/adf << eor
Title MODCO: Fragment no.          1; (polarized)

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS

SYMMETRY NOSYM
```

```
XC
  model SAOP
END

CDFT
  MAXITER 250
  NCONSTRAINTS 1
  EXCITEDCDFT
  METRIC
  ALLATOMS
  NATOMSPERSET 36
  CONSTRAINTS 1.0
  INITIALMULTIPLIERS 0.0
  THRESHOLD 1.0e-9
  STEPSIZE 0.4
  PRINT LOW
END

CHARGE 0 0
UNRESTRICTED

FRAGMENTS
  rho1 t21.iso.rho1
  rho2 t21.iso.rho2 type=fde
END

ATOMS
H       0.930241    1.239503    0.000000   f=rho1
H      -0.930241    1.239503    0.000000   f=rho1
C       0.000000    0.667450    0.000000   f=rho1
C       0.000000   -0.667450    0.000000   f=rho1
H       0.930241   -1.239503    0.000000   f=rho1
H      -0.930241   -1.239503    0.000000   f=rho1
H       0.930241    1.239503    8.000000   f=rho2
H      -0.930241    1.239503    8.000000   f=rho2
C       0.000000    0.667450    8.000000   f=rho2
C       0.000000   -0.667450    8.000000   f=rho2
H       0.930241   -1.239503    8.000000   f=rho2
H      -0.930241   -1.239503    8.000000   f=rho2
END

ALLOW PARTIALSUPERFRAGS

FDE
  PW91K
  GGAPOTXFD pw91x
  GGAPOTCFD pw91c
END
eor

rm logfile
mv TAPE21 t21.emb.rho1

$ADFBIN/adf << eor
Title MODCO: Fragment no.         2; (polarized)

EPRINT
```

```
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS

SYMMETRY NOSYM

XC
  model SAOP
END

CHARGE 0 0
UNRESTRICTED

FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active type=fde
  rho2 t21.iso.rho2
END

ATOMS
H        0.930241   1.239503    0.000000   f=rho1
H       -0.930241   1.239503    0.000000   f=rho1
C        0.000000   0.667450    0.000000   f=rho1
C        0.000000  -0.667450    0.000000   f=rho1
H        0.930241  -1.239503    0.000000   f=rho1
H       -0.930241  -1.239503    0.000000   f=rho1
H        0.930241   1.239503    8.000000   f=rho2
H       -0.930241   1.239503    8.000000   f=rho2
C        0.000000   0.667450    8.000000   f=rho2
C        0.000000  -0.667450    8.000000   f=rho2
H        0.930241  -1.239503    8.000000   f=rho2
H       -0.930241  -1.239503    8.000000   f=rho2
END

ALLOW PARTIALSUPERFRAGS

FDE
  PW91K
  GGAPOTXFD pw91x
  GGAPOTCFD pw91c
END
eor

rm logfile
mv TAPE21 t21.emb.rho2


it=1
while [ $it -le 2 ]; do

$ADFBIN/adf << eor
Title MODCO: Fragment no.          1; relaxed

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS
```

```
SYMMETRY NOSYM

XC
  model SAOP
END

NumericalQuality Good

CDFT
  MAXITER 250
  NCONSTRAINTS 1
  EXCITEDCDFT
  METRIC
  ALLATOMS
  NATOMSPERSET 36
  CONSTRAINTS 1.0
  INITIALMULTIPLIERS 0.0
  THRESHOLD 1.0e-9
  STEPSIZE 0.4
  PRINT LOW
END

CHARGE 0 0

UNRESTRICTED

FRAGMENTS
  rho1 t21.iso.rho1
  rho2 t21.emb.rho2 subfrag=active type=fde
END

restart
  file t21.emb.rho1
end

ATOMS
H       0.930241   1.239503   0.000000   f=rho1
H      -0.930241   1.239503   0.000000   f=rho1
C       0.000000   0.667450   0.000000   f=rho1
C       0.000000  -0.667450   0.000000   f=rho1
H       0.930241  -1.239503   0.000000   f=rho1
H      -0.930241  -1.239503   0.000000   f=rho1
H       0.930241   1.239503   8.000000   f=rho2
H      -0.930241   1.239503   8.000000   f=rho2
C       0.000000   0.667450   8.000000   f=rho2
C       0.000000  -0.667450   8.000000   f=rho2
H       0.930241  -1.239503   8.000000   f=rho2
H      -0.930241  -1.239503   8.000000   f=rho2
END

ALLOW PARTIALSUPERFRAGS

FDE
  PW91K
  GGAPOTXFD pw91x
  GGAPOTCFD pw91c
  FullGrid
```

```
  Energy
END
eor

rm logfile
mv TAPE21 t21.emb.rho1

$ADFBIN/adf << eor
Title MODCO: Fragment no.          2; relaxed

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS

SYMMETRY NOSYM

XC
  model SAOP
END

NumericalQuality Good

restart
  file t21.emb.rho2
end

CHARGE 0 0
UNRESTRICTED
FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active type=fde
  rho2 t21.iso.rho2
END

ATOMS
H        0.930241    1.239503    0.000000    f=rho1
H       -0.930241    1.239503    0.000000    f=rho1
C        0.000000    0.667450    0.000000    f=rho1
C        0.000000   -0.667450    0.000000    f=rho1
H        0.930241   -1.239503    0.000000    f=rho1
H       -0.930241   -1.239503    0.000000    f=rho1
H        0.930241    1.239503    8.000000    f=rho2
H       -0.930241    1.239503    8.000000    f=rho2
C        0.000000    0.667450    8.000000    f=rho2
C        0.000000   -0.667450    8.000000    f=rho2
H        0.930241   -1.239503    8.000000    f=rho2
H       -0.930241   -1.239503    8.000000    f=rho2
END

ALLOW PARTIALSUPERFRAGS

FDE
  PW91K
  GGAPOTXFD pw91x
  GGAPOTCFD pw91c
  FullGrid
  Energy
```

```
END
eor

rm logfile
mv TAPE21 t21.emb.rho2


it=$(($it+1))
done

# ======================================
# NOW THE TAPE21 FILES MUST BE RENAMED
#
#  TO: fragA1.t21, fragA2.t21
#      .... up to a max of fragA999.t21
# AND: fragB1.t21, fragB2.t21
#      .... up to a max of fragB999.t21
# ======================================

mv t21.emb.rho1 fragA1.t21
mv t21.emb.rho2 fragA2.t21




# ==================================================
#   SECOND FDE CALCULATION: STATE D+ + A-
#   note: the freeze and thaw is done manually
#         as we have open-shell fragments
# ==================================================

$ADFBIN/adf << eor
Title MODCO: Fragment no.         1; (polarized)

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS

SYMMETRY NOSYM

XC
  model SAOP
END

CHARGE 1 -1
UNRESTRICTED

FRAGMENTS
  rho1 t21.iso.rho1
  rho2 t21.iso.rho2 type=fde
END

ATOMS
H        0.930241    1.239503    0.000000    f=rho1
H       -0.930241    1.239503    0.000000    f=rho1
C        0.000000    0.667450    0.000000    f=rho1
C        0.000000   -0.667450    0.000000    f=rho1
```

```
H          0.930241   -1.239503    0.000000    f=rho1
H         -0.930241   -1.239503    0.000000    f=rho1
H          0.930241    1.239503    8.000000    f=rho2
H         -0.930241    1.239503    8.000000    f=rho2
C          0.000000    0.667450    8.000000    f=rho2
C          0.000000   -0.667450    8.000000    f=rho2
H          0.930241   -1.239503    8.000000    f=rho2
H         -0.930241   -1.239503    8.000000    f=rho2
END

ALLOW PARTIALSUPERFRAGS

FDE
  PW91K
  GGAPOTXFD pw91x
  GGAPOTCFD pw91c
END
eor

rm logfile
mv TAPE21 t21.emb.rho1

$ADFBIN/adf << eor
Title MODCO: Fragment no.        2; (polarized)

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS

SYMMETRY NOSYM

XC
  model SAOP
END

CHARGE -1  1
UNRESTRICTED

FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active type=fde
  rho2 t21.iso.rho2
END

ATOMS
H          0.930241    1.239503    0.000000    f=rho1
H         -0.930241    1.239503    0.000000    f=rho1
C          0.000000    0.667450    0.000000    f=rho1
C          0.000000   -0.667450    0.000000    f=rho1
H          0.930241   -1.239503    0.000000    f=rho1
H         -0.930241   -1.239503    0.000000    f=rho1
H          0.930241    1.239503    8.000000    f=rho2
H         -0.930241    1.239503    8.000000    f=rho2
C          0.000000    0.667450    8.000000    f=rho2
C          0.000000   -0.667450    8.000000    f=rho2
H          0.930241   -1.239503    8.000000    f=rho2
H         -0.930241   -1.239503    8.000000    f=rho2
```

```
END

ALLOW PARTIALSUPERFRAGS

FDE
  PW91K
  GGAPOTXFD pw91x
  GGAPOTCFD pw91c
END
eor

rm logfile
mv TAPE21 t21.emb.rho2

it=1
while [ $it -le 2 ]; do

$ADFBIN/adf << eor
Title MODCO: Fragment no.          1; relaxed

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS

SYMMETRY NOSYM

XC
  model SAOP
END

NumericalQuality Good

CHARGE 1 -1
UNRESTRICTED

FRAGMENTS
  rho1 t21.iso.rho1
  rho2 t21.emb.rho2 subfrag=active type=fde
END

restart
  file t21.emb.rho1
end

ATOMS
H       0.930241    1.239503    0.000000    f=rho1
H      -0.930241    1.239503    0.000000    f=rho1
C       0.000000    0.667450    0.000000    f=rho1
C       0.000000   -0.667450    0.000000    f=rho1
H       0.930241   -1.239503    0.000000    f=rho1
H      -0.930241   -1.239503    0.000000    f=rho1
H       0.930241    1.239503    8.000000    f=rho2
H      -0.930241    1.239503    8.000000    f=rho2
C       0.000000    0.667450    8.000000    f=rho2
C       0.000000   -0.667450    8.000000    f=rho2
H       0.930241   -1.239503    8.000000    f=rho2
```

```
H       -0.930241  -1.239503   8.000000   f=rho2
END

ALLOW PARTIALSUPERFRAGS

FDE
  PW91K
  GGAPOTXFD pw91x
  GGAPOTCFD pw91c
  FullGrid
  Energy
END
eor

rm logfile
mv TAPE21 t21.emb.rho1

$ADFBIN/adf << eor
Title MODCO: Fragment no.        2; relaxed

EPRINT
  SFO NOEIG NOOVL NOORBPOP
END

NOPRINT BAS FUNCTIONS

SYMMETRY NOSYM

XC
  model SAOP
END

NumericalQuality Good

restart
  file t21.emb.rho2
end

CHARGE -1  1
UNRESTRICTED
FRAGMENTS
  rho1 t21.emb.rho1 subfrag=active type=fde
  rho2 t21.iso.rho2
END

ATOMS
H        0.930241   1.239503   0.000000   f=rho1
H       -0.930241   1.239503   0.000000   f=rho1
C        0.000000   0.667450   0.000000   f=rho1
C        0.000000  -0.667450   0.000000   f=rho1
H        0.930241  -1.239503   0.000000   f=rho1
H       -0.930241  -1.239503   0.000000   f=rho1
H        0.930241   1.239503   8.000000   f=rho2
H       -0.930241   1.239503   8.000000   f=rho2
C        0.000000   0.667450   8.000000   f=rho2
C        0.000000  -0.667450   8.000000   f=rho2
H        0.930241  -1.239503   8.000000   f=rho2
H       -0.930241  -1.239503   8.000000   f=rho2
```

```
END

ALLOW PARTIALSUPERFRAGS

FDE
  PW91K
  GGAPOTXFD pw91x
  GGAPOTCFD pw91c
  FullGrid
  Energy
END
eor

rm logfile
mv TAPE21 t21.emb.rho2

it=$(($it+1))
done
# ======================================
# NOW THE TAPE21 FILES MUST BE RENAMED
#
#  TO: fragA1.t21, fragA2.t21
#      .... up to a max of fragA999.t21
# AND: fragB1.t21, fragB2.t21
#      .... up to a max of fragB999.t21
# ======================================

mv t21.emb.rho1 fragB1.t21
mv t21.emb.rho2 fragB2.t21


# ======================================
# Electron Transfer calculation is next
#
# 1) note the ELECTRONTRANSFER block keyword
# 2) hybrids not supported
# 3) NumericalQuality GOOD is needed for accuracy
# ======================================

$ADFBIN/adf << eor
Title ET calculation at R=0.0 Angstrom

EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END

NOPRINT BAS FUNCTIONS

FRAGMENTS
 rho1 t21.iso.rho1
 rho2 t21.iso.rho2
END

ATOMS
 H        0.930241    1.239503    0.000000 f=rho1
 H       -0.930241    1.239503    0.000000 f=rho1
 C        0.000000    0.667450    0.000000 f=rho1
 C        0.000000   -0.667450    0.000000 f=rho1
```

```
 H          0.930241   −1.239503    0.000000 f=rho1
 H         −0.930241   −1.239503    0.000000 f=rho1
 H          0.930241    1.239503    8.000000 f=rho2
 H         −0.930241    1.239503    8.000000 f=rho2
 C          0.000000    0.667450    8.000000 f=rho2
 C          0.000000   −0.667450    8.000000 f=rho2
 H          0.930241   −1.239503    8.000000 f=rho2
 H         −0.930241   −1.239503    8.000000 f=rho2
END

NumericalQuality Good

UNRESTRICTED

CHARGE 0 0
XC
  GGA PW91
END

SYMMETRY nosym

SCF
  iterations 0
END

Beckegrid
 Quality Good
END

ELECTRONTRANSFER
 numfrag 2
END
eor

rm logfile
```

14.5.2 Non-self-consistent Green's function calculation

Example: DOS and transmission: Aluminium

Download green_Al.run

```
#!/bin/sh

# As an example of a non-self-consistent Green's function calculation, we will
# look at the density of states (DOS) and transmission of an infinite 1D chain
# of Aluminum atoms.

# First we need to perform a single-point calculation with ADF on a principal
# layer, consisting, in this case, of four atoms. Since bulk Aluminum has an FCC
# structure with a lattice constant of 4.05 Angstrom, the nearest neighbor
# distance is approximately 2.83 Angstrom. green requires SYMMETRY NOSYM, so we
# have the following input file for the principal layer:

$ADFBIN/adf <<eor
TITLE Principal layer
```

```
ATOMS
    Al      -4.290000    0.000000    0.000000
    Al      -1.430000    0.000000    0.000000
    Al       1.430000    0.000000    0.000000
    Al       4.290000    0.000000    0.000000
END
SYMMETRY NOSYM
BASIS
    Type DZP
    Core Large
    CreateOutput None
END
SCF
  Converge 1.0e-8
END
XC
    LDA SCF VWN
END
eor

mv TAPE21 layer.t21

# The bulk contact geometry consists of three principal layers:

$ADFBIN/adf <<eor
TITLE Bulk
ATOMS
    Al     -15.730000    0.000000    0.000000 f=left
    Al     -12.870000    0.000000    0.000000 f=left
    Al     -10.010000    0.000000    0.000000 f=left
    Al      -7.150000    0.000000    0.000000 f=left
    Al      -4.290000    0.000000    0.000000 f=center
    Al      -1.430000    0.000000    0.000000 f=center
    Al       1.430000    0.000000    0.000000 f=center
    Al       4.290000    0.000000    0.000000 f=center
    Al       7.150000    0.000000    0.000000 f=right
    Al      10.010000    0.000000    0.000000 f=right
    Al      12.870000    0.000000    0.000000 f=right
    Al      15.730000    0.000000    0.000000 f=right
END
SYMMETRY NOSYM
FRAGMENTS
    left    layer.t21
    center  layer.t21
    right   layer.t21
END
XC
    LDA SCF VWN
END
SCF
    Converge 1.0e-10
    AccelerationMethod LISTi
END
eor

mv TAPE21 bulk.t21

# Notice that we have increased the number of SCF iterations. The combination of
```

```
# SYMMETRY NOSYM with a 1D chain of metal atoms generally leads to convergence
# problems. This is the main reason why the principal layer consists of only
# four atoms. Fortunately, for larger 3D contacts, the convergence is generally
# better.

# From the bulk TAPE21 file green can calculate the self-energies of the left
# and right contacts. As discussed in the introduction, the self-energy of the
# left contact needs the center and right fragments of the bulk calculation, and
# the self-energy of the right contact needs the center and left fragments.
# Since we need a self-energy matrix for every energy for which we want to
# calculate the DOS and transmission, already here we have to specify the energy
# range. We take 1000 points between -0.4 and 0 Hartree.

$ADFBIN/green <<eor
SURFACE bulk.t21
    FRAGMENTS center right
END
EPS -0.4 0 1000
ETA 1e-6
eor

mv SURFACE left.kf

$ADFBIN/green <<eor
SURFACE bulk.t21
    FRAGMENTS center left
END
EPS -0.4 0 1000
ETA 1e-6
eor

mv SURFACE right.kf

# Since we want to calculate the DOS and transmission of bare aluminum, we can
# reuse the bulk.t21 file for the extended molecule. We couple the left self-
# energy to the 'left' fragment and the right self-energy to the 'right'
# fragment in bulk.t21. Since we performed restricted ADF calculations, there is
# no difference between spin-A and spin-B and we can omit spin-B from the
# calculation.

$ADFBIN/green <<eor
DOS bulk.t21
TRANS bulk.t21
EPS -0.4 0 1000
ETA 1e-6
LEFT left.kf
    FRAGMENT left
END
RIGHT right.kf
    FRAGMENT right
END
NOSAVE DOS_B, TRANS_B
eor

# As would be expected for a 1D system, the DOS shows Van Hove singularities at
# the band edges. Apart from oscillations due to the finite size of the system
# in ADF, the transmission only reaches integer values. Between approximately
# -0.35 and -0.15 Hartree, only the sigma channel contributes to the
```

```
# transmission. Above -0.15 Hartree also the two pi channels start to
# contribute.

echo ""
echo "Contents of DOS_A:"
cat DOS_A
echo "END"
echo ""
echo "Contents of TRANS_A:"
cat TRANS_A
echo "END"
```

Example: Gold electrodes

Download `green_Au.run`

In this example of *green*, the self-energies are calculated of gold electrodes, the material most often used in molecular electronics. In *the example for the Benzenedithiol junction* (page 804) these self-energies will be used to calculate the DOS and transmission of a benzenedithiol junction. The geometry of the electrodes is shown in Fig. 1.
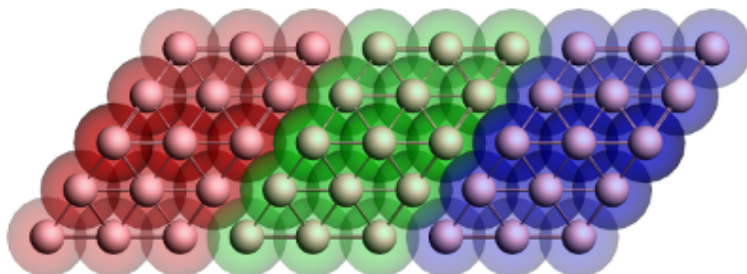
Figure 1: Geometry of the gold contact used in the calculation of the self-energy. The lead consists of two surface layers, left (red) and right (blue), and a bulk layer (green). Each principal layer in turn consists of three atomic layers. This should be sufficient to ensure that the Hamiltonian of the central (green) layer is a bulk Hamiltonian.

Each principal layer contains 3x3x3=27 gold atoms. For the calculation of the self-energies three principal layers are needed, and therefore 81 gold atoms in total. To keep the runtimes manageable it is therefore important to choose the basis set as small as possible. For transport calculations, a DZ basis with a large frozen core is generally sufficient. Unfortunately, even with the largest frozen core (Au.4f), the basis set for Au still contains 19 electrons. A significant speedup can be obtained by limiting this to 11 electrons (only the outer *d* and *s* shells). Be advised that even with this reduction the total runtime of calculation can be long.

To facilitate the calculation of the electrodes, first a gold atom fragment will be calculated with the smallest possible basis. The sample directory contains the required Au.5p and Au.5p.dirac files. Note that for gold relativistic effects are important. Therefore RELATIVISTIC Scalar ZORA will be used throughout this example.

```
$ADFBIN/dirac < Au.5p.dirac

mv TAPE12 t12.rel

$ADFBIN/adf -n1 << eor
CREATE Au    file=Au.5p
RELATIVISTIC Scalar ZORA
COREPOTENTIALS t12.rel
XC
    LDA SCF VWN
END
```

```
eor

mv TAPE21 t21.Au

$ADFBIN/adf << eor
TITLE Gold atom
ATOMS
    Au        0.000000   0.000000   0.000000
END
RELATIVISTIC Scalar ZORA
FRAGMENTS
    Au   t21.Au
END
XC
    LDA SCF VWN
END
eor

mv TAPE21 Au.t21
```

A principal layer of gold consists of three atomic layers, which should be sufficient due to the small screening length. An atomic layer contains 3x3=9 atoms in a (111) FCC configuration. This allows one to use the top-, bride-, and hollow-site binding configurations for a molecule. For the following calculations it is necessary to first construct a fragment of a principal layer.

```
$ADFBIN/adf << eor
TITLE Principal layer
ATOMS
    Au       -2.355588   -6.662612    0.000000
    Au       -2.355589   -4.164133   -1.442498
    Au       -2.355589   -4.164133    1.442498
    Au       -2.355589   -1.665653   -2.884996
    Au       -2.355589   -1.665653    0.000000
    Au       -2.355589   -1.665653    2.884996
    Au       -2.355589    0.832826   -1.442498
    Au       -2.355589    0.832826    1.442498
    Au       -2.355589    3.331306    0.000000
    Au        0.000000   -4.996959    0.000000
    Au        0.000000   -2.498480   -1.442498
    Au        0.000000   -2.498480    1.442498
    Au        0.000000    0.000000   -2.884996
    Au        0.000000    0.000000    0.000000
    Au        0.000000    0.000000    2.884996
    Au        0.000000    2.498480   -1.442498
    Au        0.000000    2.498480    1.442498
    Au        0.000000    4.996959    0.000000
    Au        2.355589   -3.331306    0.000000
    Au        2.355589   -0.832826   -1.442498
    Au        2.355589   -0.832826    1.442498
    Au        2.355589    1.665653   -2.884996
    Au        2.355589    1.665653    0.000000
    Au        2.355589    1.665653    2.884996
    Au        2.355589    4.164133   -1.442498
    Au        2.355589    4.164133    1.442498
    Au        2.355588    6.662612    0.000000
END
SYMMETRY NOSYM
RELATIVISTIC Scalar ZORA
```

```
FRAGMENTS
    Au  Au.t21
END
XC
    LDA SCF VWN
END
eor

mv TAPE21 layer.t21
```

Three principal layers are stacked together to calculate the self-energies (see Fig. 1). The names of the fragments are significant, since one needs to refer to them by name in the calculation of the self-energies.

```
$ADFBIN/adf << eor
TITLE Bulk gold
ATOMS
    Au       -9.422355  -11.659571    0.000000 f=left
    Au       -9.422356   -9.161092   -1.442498 f=left
    Au       -9.422356   -9.161092    1.442498 f=left
    Au       -9.422356   -6.662612   -2.884996 f=left
    Au       -9.422356   -6.662612    0.000000 f=left
    Au       -9.422356   -6.662612    2.884996 f=left
    Au       -9.422356   -4.164133   -1.442498 f=left
    Au       -9.422356   -4.164133    1.442498 f=left
    Au       -9.422356   -1.665653    0.000000 f=left
    Au       -7.066767   -9.993918    0.000000 f=left
    Au       -7.066767   -7.495439   -1.442498 f=left
    Au       -7.066767   -7.495439    1.442498 f=left
    Au       -7.066767   -4.996959   -2.884996 f=left
    Au       -7.066767   -4.996959    0.000000 f=left
    Au       -7.066767   -4.996959    2.884996 f=left
    Au       -7.066767   -2.498479   -1.442498 f=left
    Au       -7.066767   -2.498479    1.442498 f=left
    Au       -7.066767    0.000000    0.000000 f=left
    Au       -4.711178   -8.328265    0.000000 f=left
    Au       -4.711178   -5.829785   -1.442498 f=left
    Au       -4.711178   -5.829785    1.442498 f=left
    Au       -4.711178   -3.331306   -2.884996 f=left
    Au       -4.711178   -3.331306    0.000000 f=left
    Au       -4.711178   -3.331306    2.884996 f=left
    Au       -4.711178   -0.832826   -1.442498 f=left
    Au       -4.711178   -0.832826    1.442498 f=left
    Au       -4.711179    1.665653    0.000000 f=left
    Au       -2.355588   -6.662612    0.000000 f=center
    Au       -2.355589   -4.164133   -1.442498 f=center
    Au       -2.355589   -4.164133    1.442498 f=center
    Au       -2.355589   -1.665653   -2.884996 f=center
    Au       -2.355589   -1.665653    0.000000 f=center
    Au       -2.355589   -1.665653    2.884996 f=center
    Au       -2.355589    0.832826   -1.442498 f=center
    Au       -2.355589    0.832826    1.442498 f=center
    Au       -2.355589    3.331306    0.000000 f=center
    Au        0.000000   -4.996959    0.000000 f=center
    Au        0.000000   -2.498480   -1.442498 f=center
    Au        0.000000   -2.498480    1.442498 f=center
    Au        0.000000    0.000000   -2.884996 f=center
    Au        0.000000    0.000000    0.000000 f=center
    Au        0.000000    0.000000    2.884996 f=center
```

```
    Au        0.000000     2.498480    -1.442498 f=center
    Au        0.000000     2.498480     1.442498 f=center
    Au        0.000000     4.996959     0.000000 f=center
    Au        2.355589    -3.331306     0.000000 f=center
    Au        2.355589    -0.832826    -1.442498 f=center
    Au        2.355589    -0.832826     1.442498 f=center
    Au        2.355589     1.665653    -2.884996 f=center
    Au        2.355589     1.665653     0.000000 f=center
    Au        2.355589     1.665653     2.884996 f=center
    Au        2.355589     4.164133    -1.442498 f=center
    Au        2.355589     4.164133     1.442498 f=center
    Au        2.355588     6.662612     0.000000 f=center
    Au        4.711179    -1.665653     0.000000 f=right
    Au        4.711178     0.832826    -1.442498 f=right
    Au        4.711178     0.832826     1.442498 f=right
    Au        4.711178     3.331306    -2.884996 f=right
    Au        4.711178     3.331306     0.000000 f=right
    Au        4.711178     3.331306     2.884996 f=right
    Au        4.711178     5.829785    -1.442498 f=right
    Au        4.711178     5.829785     1.442498 f=right
    Au        4.711178     8.328265     0.000000 f=right
    Au        7.066767     0.000000     0.000000 f=right
    Au        7.066767     2.498479    -1.442498 f=right
    Au        7.066767     2.498479     1.442498 f=right
    Au        7.066767     4.996959    -2.884996 f=right
    Au        7.066767     4.996959     0.000000 f=right
    Au        7.066767     4.996959     2.884996 f=right
    Au        7.066767     7.495439    -1.442498 f=right
    Au        7.066767     7.495439     1.442498 f=right
    Au        7.066767     9.993918     0.000000 f=right
    Au        9.422356     1.665653     0.000000 f=right
    Au        9.422356     4.164133    -1.442498 f=right
    Au        9.422356     4.164133     1.442498 f=right
    Au        9.422356     6.662612    -2.884996 f=right
    Au        9.422356     6.662612     0.000000 f=right
    Au        9.422356     6.662612     2.884996 f=right
    Au        9.422356     9.161092    -1.442498 f=right
    Au        9.422356     9.161092     1.442498 f=right
    Au        9.422355    11.659571     0.000000 f=right
END
SYMMETRY NOSYM
RELATIVISTIC Scalar ZORA
FRAGMENTS
    left    layer.t21
    center  layer.t21
    right   layer.t21
END
XC
    LDA SCF VWN
END
eor

mv TAPE21 bulk.t21
```

Similar to the other examples, the self-energies of the left and right contacts is calculated for 1000 energy points between -0.5 and 0 Hartree. This results in two keyfiles of approximately 2.5 GB each. Since the self-energies are independent of whatever is placed between the contacts, they can be reused many times.

```
$ADFBIN/green << eor
SURFACE bulk.t21
    FRAGMENTS center right
END
EPS -0.5 0 1000
ETA 1e-6
eor

mv SURFACE left.kf

$ADFBIN/green << eor
SURFACE bulk.t21
    FRAGMENTS center left
END
EPS -0.5 0 1000
ETA 1e-6
eor

mv SURFACE right.kf
```

In order to interpret transmissions calculated with these self-energies, it is necessary to know the location of the Fermi energy. An estimate for the Fermi energy can be obtained from the bulk SCF calculation by taking the average of the HOMO and LUMO energies, which in this case equals -0.195 Hartree.

Usually the self-energies will be used to calculate the transmission of a molecular junction. However, it is instructive to use a principal layer of gold as the "molecule" and study the DOS and transmission of bulk gold.

```
$ADFBIN/green << eor
DOS bulk.t21
TRANS bulk.t21
EPS -0.5 0 1000
ETA 1e-6
LEFT left.kf
    FRAGMENT left
END
RIGHT right.kf
    FRAGMENT right
END
NOSAVE DOS_B, TRANS_B
eor
```

The results are shown in the following figure:
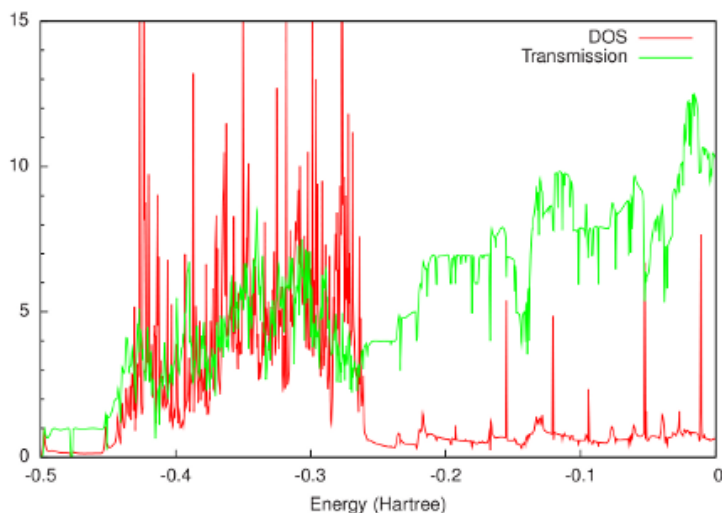
From this figure it can be seen that around the Fermi energy (-0.2 Hartree), both the DOS and the transmission of gold are relatively constant. This feature makes gold an attractive material for electrodes, since one can expect that the transmission of a molecular junction will be dominated by the molecular properties.

Example: Benzenedithiol junction: Wide-Band-Limit

Download green_WBL.run

In the wide-band limit (WBL) the coupling to the leads is assumed to be independent of energy. Therefore one does not need to calculate any self-energies. This also means that the eigenspace of the Green's function is independent of energy. It can therefore be diagonalized in advance, greatly speeding up the calculation of the DOS and the transmission.

In the example $ADFHOME/examples/adf/green_Al/green_WBL/green_WBL.run of *green*, the transmission of *benzenedithiol junction* (page 804) in the wide-band limit (WBL) is calculated. In order to model the molecule-metal interface, we do need to include a few gold layers in the calculation. However, unlike *before* (page 798), only a single atomic layer as the principal layer is used.

Compare green_BDT.run

```
FRAGMENTS
    left        layer.t21
    molecule    molecule.t21
    right       layer.t21
END
```

with green_WBL/green_WBL.run, where each layer of gold atoms has its own fragment:

```
FRAGMENTS
    left        layer.t21
    left.2      layer.t21
    left.3      layer.t21
    molecule    molecule.t21
    right.3     layer.t21
    right.2     layer.t21
    right       layer.t21
END
```

Because a single atomic layer is an unnatural configuration for gold, a minor amount of smearing is necessary to make the calculation converge. The molecule is sandwiched in between the electrodes just like before (see Fig. 2 in

---

the example for benzenedithiol (page 804)). However, this time each atomic layer of gold gets its own fragment. The reason for this configuration is that if the WBL is used on the entire gold contact the result is an an unphysical coupling to the leads; even the gold atoms contacting the molecule would have a direct coupling to the environment. A much better result can be obtained by only using the WBL on the back-most atomic layer and letting the electrons propagate naturally through the rest of the contact. Because the WBL is computationally so inexpensive, we can easily calculate the DOS and transmission for 10,000 points instead of 1000.

```
$ADFBIN/green << eor
DOS fock.t21
TRANS fock.t21
EPS -0.5 0 10000
ETA 1e-6
LEFT
    FRAGMENT left
    ETA 2e-2
END
RIGHT
    FRAGMENT right
    ETA 2e-2
END
NOSAVE DOS_B, TRANS_B
eor
```

A comparison of the resulting transmission with the calculation with self-energies is shown in the following figure:

The WBL shows good agreement with the non-WBL transmission around the Fermi energy (-0.195 Hartree or -5.306 eV). Note that the quality of the WBL depends on the choice of the coupling (ETA). For this particular contact geometry we obtain good agreement for ETA = 0.02 Hartree, but a better value may be found for other electrodes. Finally, the WBL can be incrementally improved by adding more gold layers to the extended molecule. For many layers it converges to the calculation with full self-energies.

Example: Benzenedithiol junction

Download green_BDT.run

In this example of *green*, the DOS and transmission of a benzenedithiol molecule between gold electrodes is calculated. The calculation uses the self-energies obtained in *the example for the gold electrodes* (page 798). Note that this is a relatively expensive calculation.

First a fragment of the isolated molecule is constructed:

```
$ADFBIN/adf << eor
TITLE Benzenedithiol
ATOMS
    C        -1.400000    0.000000    0.000000
    C        -0.700000    0.000000   -1.200000
    C        -0.700000    0.000000    1.200000
    C         0.700000    0.000000   -1.200000
    C         0.700000    0.000000    1.200000
    C         1.400000    0.000000    0.000000
    H        -1.200000    0.000000   -2.200000
    H        -1.200000    0.000000    2.200000
    H         1.200000    0.000000   -2.200000
    H         1.200000    0.000000    2.200000
    S        -3.200000    0.000000    0.000000
    S         3.200000    0.000000    0.000000
END
SYMMETRY NOSYM
RELATIVISTIC Scalar ZORA
BASIS
    type DZP
    core Large
    createOutput None
END
XC
    LDA SCF VWN
END
eor

mv TAPE21 molecule.t21
```

Next the molecule is sandwiched between the electrodes in the configuration of Fig. 2. For this the fragment of the principal layer obtained in *the example for the gold electrodes* (page 798) is needed.
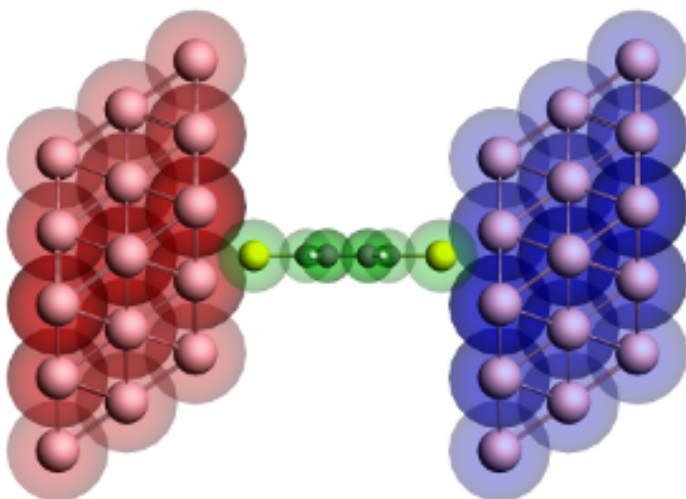

Figure 2: Geometry of the extended molecule used in the calculation of a benzenedithiol junction. The molecule is shown in green, while the left and right contact regions are shown in red and blue, respectively. Note that the red region corresponds to the blue surface layer in Figure 1 of *the example for the gold electrodes* (page 798) and vice versa.

```
$ADFBIN/adf << eor
TITLE Benzenedithiol
ATOMS
    Au      -9.911177   -6.662612    0.000000 f=left
    Au      -9.911178   -4.164133   -1.442498 f=left
    Au      -9.911178   -4.164133    1.442498 f=left
    Au      -9.911178   -1.665653   -2.884996 f=left
    Au      -9.911178   -1.665653    0.000000 f=left
    Au      -9.911178   -1.665653    2.884996 f=left
    Au      -9.911178    0.832826   -1.442498 f=left
    Au      -9.911178    0.832826    1.442498 f=left
    Au      -9.911178    3.331306    0.000000 f=left
    Au      -7.555589   -4.996959    0.000000 f=left
    Au      -7.555589   -2.498480   -1.442498 f=left
    Au      -7.555589   -2.498480    1.442498 f=left
    Au      -7.555589    0.000000   -2.884996 f=left
    Au      -7.555589    0.000000    0.000000 f=left
    Au      -7.555589    0.000000    2.884996 f=left
    Au      -7.555589    2.498480   -1.442498 f=left
    Au      -7.555589    2.498480    1.442498 f=left
    Au      -7.555589    4.996959    0.000000 f=left
    Au      -5.200000   -3.331306    0.000000 f=left
    Au      -5.200000   -0.832826   -1.442498 f=left
    Au      -5.200000   -0.832826    1.442498 f=left
    Au      -5.200000    1.665653   -2.884996 f=left
    Au      -5.200000    1.665653    0.000000 f=left
    Au      -5.200000    1.665653    2.884996 f=left
    Au      -5.200000    4.164133   -1.442498 f=left
    Au      -5.200000    4.164133    1.442498 f=left
    Au      -5.200001    6.662612    0.000000 f=left
    C       -1.400000    0.000000    0.000000 f=molecule
    C       -0.700000    0.000000   -1.200000 f=molecule
    C       -0.700000    0.000000    1.200000 f=molecule
    C        0.700000    0.000000   -1.200000 f=molecule
    C        0.700000    0.000000    1.200000 f=molecule
    C        1.400000    0.000000    0.000000 f=molecule
    H       -1.200000    0.000000   -2.200000 f=molecule
    H       -1.200000    0.000000    2.200000 f=molecule
    H        1.200000    0.000000   -2.200000 f=molecule
    H        1.200000    0.000000    2.200000 f=molecule
    S       -3.200000    0.000000    0.000000 f=molecule
    S        3.200000    0.000000    0.000000 f=molecule
    Au       5.200001   -6.662612    0.000000 f=right
    Au       5.200000   -4.164133   -1.442498 f=right
    Au       5.200000   -4.164133    1.442498 f=right
    Au       5.200000   -1.665653   -2.884996 f=right
    Au       5.200000   -1.665653    0.000000 f=right
    Au       5.200000   -1.665653    2.884996 f=right
    Au       5.200000    0.832826   -1.442498 f=right
    Au       5.200000    0.832826    1.442498 f=right
    Au       5.200000    3.331306    0.000000 f=right
    Au       7.555589   -4.996959    0.000000 f=right
    Au       7.555589   -2.498480   -1.442498 f=right
    Au       7.555589   -2.498480    1.442498 f=right
    Au       7.555589    0.000000   -2.884996 f=right
    Au       7.555589    0.000000    0.000000 f=right
    Au       7.555589    0.000000    2.884996 f=right
    Au       7.555589    2.498480   -1.442498 f=right
```

```
    Au          7.555589    2.498480    1.442498 f=right
    Au          7.555589    4.996959    0.000000 f=right
    Au          9.911178   -3.331306    0.000000 f=right
    Au          9.911178   -0.832826   -1.442498 f=right
    Au          9.911178   -0.832826    1.442498 f=right
    Au          9.911178    1.665653   -2.884996 f=right
    Au          9.911178    1.665653    0.000000 f=right
    Au          9.911178    1.665653    2.884996 f=right
    Au          9.911178    4.164133   -1.442498 f=right
    Au          9.911178    4.164133    1.442498 f=right
    Au          9.911177    6.662612    0.000000 f=right
END
SYMMETRY NOSYM
RELATIVISTIC Scalar ZORA
FRAGMENTS
    left        layer.t21
    molecule    molecule.t21
    right       layer.t21
END
XC
    LDA SCF VWN
END
eor

mv TAPE21 fock.t21
```

The DOS and transmission can now be calculated:

```
$ADFBIN/green << eor
DOS fock.t21
TRANS fock.t21
EPS -0.5 0 1000
ETA 1e-6
LEFT left.kf
    FRAGMENT left
END
RIGHT right.kf
    FRAGMENT right
END
NOSAVE DOS_B, TRANS_B
eor
```

The results are shown in the following figure:
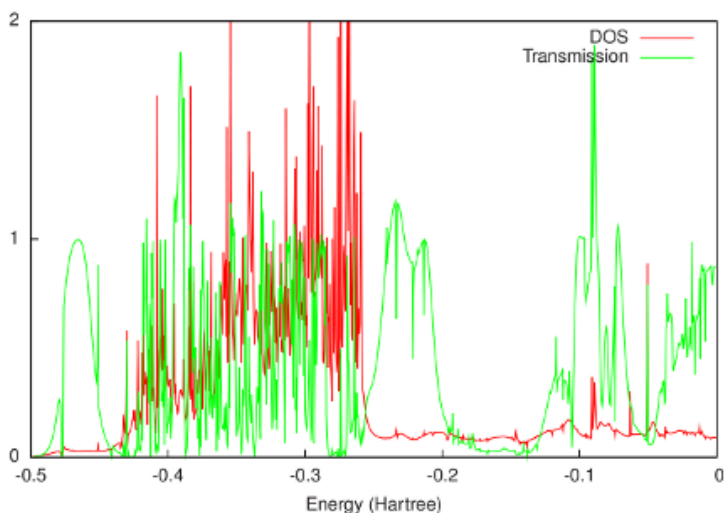
The Fermi energy of the electrodes is -0.2 Hartree (see *the example for the gold electrodes* (page 798)). This is just above the HOMO of the molecular junction. Consistent with literature, the HOMO and lower orbitals are combined into a broad peak just below the Fermi energy, while the LUMO is much sharper and situated about 2 eV above the Fermi energy.

The current can be calculated by integrating the transmission around the Fermi energy. At low temperatures this means that the differential conductance is equal to the transmission times the quantum of conductance.

14.6 Analysis

14.6.1 Fragment orbitals, bond energy decomposition, charge analysis

Example: Compound Fragments: Ni(CO)4

```
Download Frags_NiCO4.run
```

```
#! /bin/sh

# An illustration of the fragment feature of ADF.

# A transition metal complex is built from a Nickel atom and four CO fragments.
# The outcomes allows for an analysis (of molecular orbitals and the Bonding
# energy) in terms of the fragment properties. It is a Single Point calculation.
# Geometry optimization would not have been possible in this set-up because an
# optimization requires that only single-atom fragments are used.

# The three atoms are created first: C, O, and Ni. For Carbon and Oxygen a type-
# DZ basis set is used (double-zeta) using the Basis key, while Ni gets a type-
# TZP basis (triple-zeta plus polarization).



# == CO ==



# The CO molecule, to serve as a fragment template in Ni(CO)4 , is computed from
# the atomic fragments C and O. The coordinate values (atoms) are in bohr,
# rather than in Angstrom because the unit-of-length is redefined by the key
# units with subkey length.
```

```
# The key scf is used to specify a somewhat tighter convergence criterion than
# the default, just to illustrate how to do this (normal settings are quite
# adequate).

# The TAPE21 result file is renamed t21.CO.


$ADFBIN/adf -n1 <<eor
create Ni  $ADFRESOURCES/TZP/Ni.2p
eor

mv TAPE21 t21.Ni
rm logfile


$ADFBIN/adf -n1 <<eor
title CO (as fragment for NiCO4)

SCF
converge  1e-8
end

EPRINT
SFO eig ovl
END

units
length  bohr
end

atoms
C   0  0  0
O   0  0  2.15617844
end

Basis
  Type DZ
  Core Small
End

eor

mv TAPE21 CO.t21
rm logfile


# One needs to include the subkey SFO of the key EPRINT with arguments eig and
# ovl in order to get the SFO MO coefficients and SFO overlap matrix printed on
# standard output.


# == Main calculation ==


# Apart from the title, the input contains comment. This does not specify
# computational parameters but is only echoed in the output header, similar to
# the title. Contrary to the title, however, such comments are not preserved,
```

```
# apart from their echo in output and they are not written to TAPE21 or any
# other result file.

# The atomic coordinates (atoms) are given in bohr (Units). To supply the
# numerical values use is made of user-defined constants (define): 2.0053211 and
# 3.2501913. This is convenient and it prevents typing errors when several
# coordinate values are identical, in particular when they carry a lot of
# decimal places.

# The Atoms records contain also a specification of the fragments to which the
# respective atoms belong: four different CO fragments. No fragment is specified
# for the Ni atom, which implies that it is a fragment on its own.

# The numbers at the very left of the records (1 through 9, with (optionally) a
# period after them), have no relevance. You can set them for ease of reference
# or counting.


$ADFBIN/adf <<eor
title Ni(CO)4,  from fragments Ni and CO

COMMENT
No geometry optimization possible, because not all fragments
are single atoms
END

units
 length  bohr
end

EPRINT
SFO eig ovl
END

atoms
1. Ni   0        0          0
2. C    2.0053211    2.0053211     2.0053211      f=CO/1
3. C   -2.0053211   -2.0053211     2.0053211      f=CO/2
4. C    2.0053211   -2.0053211    -2.0053211      f=CO/3
5. C   -2.0053211    2.0053211    -2.0053211      f=CO/4
6. O    3.2501913    3.2501913     3.2501913      f=CO/1
7. O   -3.2501913   -3.2501913     3.2501913      f=CO/2
8. O    3.2501913   -3.2501913    -3.2501913      f=CO/3
9. O   -3.2501913    3.2501913    -3.2501913      f=CO/4
end

fragments
CO  CO.t21
Ni  t21.Ni
end

eor

mv TAPE21 NiCO4.t21
rm logfile
```

Example: Fragments: PtCl4H2 2-

Download Frags_PtCl4H2.run

```
#! /bin/sh

# The (scalar) ZORA relativistic option formalism) is used because of the
# presence of the heavy Pt atom. The complex is built from fragments H2 and
# PtCl_4^2- .

# The calculations of the molecule and larger fragments are performed with
# GGA's.


# == fragments H2 and PtCl_4^2- ==

# The two fragments H2 and PtCl_4^2- are first calculated, from which we are
# going to build the final complex.


$ADFBIN/adf <<eor
Title   H2  R=1.68a.u.
NoPrint sfo,frag,functions
Units
  length bohr
End
Atoms
  H       0.0              0.0               0.84
  H       0.0              0.0              -0.84
End
Basis
  Type DZP
End
XC
  GGA  becke perdew
End
Relativistic Scalar ZORA
eor

mv TAPE21 H2.t21
rm logfile


$ADFBIN/adf <<eor
title   PtCl4 (2-)
noprint sfo,frag,functions
units
  length   bohr
end
ATOMS
  Pt   0            0           0
  Cl   4.361580   0.000000   0
  Cl   0.000000   4.361580   0
  Cl  -4.361580   0.000000   0
  Cl   0.000000  -4.361580   0
end

Basis
```

```
  Type DZP
  Pt ZORA/DZ/Pt.4d
End
xc
  GGA  becke perdew
end
relativistic scalar ZORA
charge  -2
eor

mv TAPE21 PtCl4.t21
rm logfile

# The key charge is used to specify the net total charge. The default for the
# net total charge is the sum-of-fragment-charges. The fragments (Pt and Cl
# atoms) have been computed neutrally, but we want to calculate the PtCl4
# complex as a 2- ion.


# == Main calculation ==

# Finally we compute PtCl4 H2 2- from the fragments PtCl_4^2- and H2/


$ADFBIN/adf <<eor
title   PtCl4 H2
units
  length bohr
end
EPRINT
  SFO eig ovl
END
xc
  GGA  becke perdew
end
relativistic scalar ZORA
ATOMS
  Pt   0             0             0              f=PtCl4
  Cl   0.000000     -4.361580     0.00000000     f=PtCl4
  Cl   0.000000      4.361580     0.00000000     f=PtCl4
  Cl  -4.361580      0.000000     0.00000000     f=PtCl4
  Cl   4.361580      0.000000     0.00000000     f=PtCl4
  H    0.0           0.0          5.58           f=H2
  H    0.0           0.0          7.26           f=H2
end
fragments
  PtCl4     PtCl4.t21
  H2        H2.t21
end
eor

mv TAPE21 PtCl4H2.t21


# Note that, although the key charge is not supplied, the molecule is not
# neutral: the default charge (that is, omitting the keys charge, occupations)
# is the sum-of-fragments: the fragments here are H2 and PtCl4 2- , yielding a
# net charge for the molecule of minus two.
```

```
# Note the f=fragment specification in the Atoms block. No fragment-numbering
# suffix (/n) is required because there is only one fragment of each fragment
# type.
```

Example: Spin-unrestricted Fragments: H2

Download UnrFrag_H2.run

```
#! /bin/sh

# This is a small but important example to illustrate what goes into an accurate
# calculation of the 'true' bond energy of a molecule. The (ADF-specific)
# problem is that in a straightforward molecular calculation, the bond energy is
# computed as the energy difference between at the one hand the molecule, and at
# the other hand the isolated spherically symmetric spin-restricted atoms. The
# italic-typed features imply that the reference (comparison) state is usually
# not the physical ground state of the reference system (isolated atoms) and
# hence the computed energy difference has no direct relation to experimental
# data. To account for the true atomic ground states, one has to add correction
# terms. Study this sample carefully to make sure that you fully understand the
# steps to take and consult the User's Guide for details. See also the this
# document for a discussion of multiplet states.

# See also the example, SD_Cr(NH3)6.

# The H2 case consists of a sequence of simple calculations to demonstrate the
# Unrestricted Fragments option. The energy difference between an unrestricted
# fragment as it is used in adf and a self-consistent unrestricted fragment is
# also computed. This turns out to be quite small, confirming that the adf
# approach, although not formally exact, is adequate for practical purposes.

$ADFBIN/adf -n1 <<eor
create H    file=$ADFRESOURCES/DZP/H
eor

mv TAPE21 t21H

$ADFBIN/adf <<eor
title H unrestr., not self-consistent (as used in unr.frag. calcs)

EPRINT
SFO eig ovl
END

scf
iterations 0  ! prohibit relaxation
end

unrestricted

charge 0 1   !  if not specified up and down electrons
!             will both get 0.5 electron: in fact restricted

fragments
H  t21H
end
```

```
atoms
H 0 0 0
end

eor

rm TAPE21 logfile

# By setting the scf iterations to zero (a value of one (1) would give the same
# result) we prevent cycling to self-consistency. The energy of the 'final' one-
# electron orbitals is consequently computed in the start-up potential, i.e. the
# field of the restricted (basic) atom, where spin-alpha and spin-beta are equally
# occupied, in this case by 0.5 electron each. The not-self-consistent,
# unrestricted H atom is precisely the 'unrestricted' fragment as it can be used
# in an adf calculation with unrestricted fragments. The fragment file must be
# the TAPE21 result file from a restricted run, but at start-up you can specify
# that the Fragment Orbitals are, for purposes of reference and comparison,
# occupied in an unrestricted way in the final molecule.

# A calculation that uses restricted fragments right away computes the bonding
# energy relative to the restricted fragments. The difference between using
# restricted and unrestricted fragments is the 'bonding' energy computed in the
# run above.


$ADFBIN/adf <<eor
title  H  unr. self-consistent from unr.0

EPRINT
SFO eig ovl
END

unrestricted
charge 0  1

fragoccupations
H
s  1 // 0
subend
end

Atoms
H  0 0 0
end

fragments
H       t21H
end

eor

rm TAPE21 logfile


# Here we start with the unrestricted fragment and relax to self-consistency.
# The 'bonding energy', i.e. the relaxation energy, is very small, demonstrating
# that using non-self-consistent unrestricted fragments involves only a small
```

```
# error (which, moreover, can be computed as shown here).

# The key UnRestricted sets the spin-unrestricted mode. The key Charge is used
# to specify a net total charge of zero and a net total spin polarization by an
# excess of 1.0 spin-alpha electrons over spin-beta.

$ADFBIN/adf <<eor
title   H2 restricted, from restricted fragments

EPRINT
SFO eig ovl
END

ATOMS
H   0  0   0.375
H   0  0  -0.375
end

fragments
H   t21H
end

eor

rm TAPE21 logfile

# This is the simplest approach, using restricted fragments. The bonding energy
# must be corrected because the reference (restricted H atoms, with 0.5
# electrons in spin-alpha and 0.5 in spin-beta) is far from the true H-atom
# ground state: see the previous runs on the single H atom.

$ADFBIN/adf <<eor
title   H2 from unrestricted fragments

EPRINT
SFO eig ovl
END

ATOMS
H.1   0   0   0.375
H.2   0   0  -0.375
end

fragments !  two different fragment types are necessary
!            because the two atoms get different FragOccupations
!            (see below), while the key FragOc.. addresses
!            only fragmentTYPES
H.1  t21H
H.2  t21H
end

charge  0

IrrepOccupations
  sigma   2  !  specify the state (not always
             !  necessary)
end
```

```
fragoccupations
  H.1
    s  1 // 0
  subend
  H.2
    s  0 // 1
  subend
end

modifystartpotential
H.1  1 // 0    !  this helps SCF start-up
H.2  0 // 1    !  but is here not necessary
end

eor

rm TAPE21 logfile

# This should be a fair approximation (in the lda model) to the bonding energy
# of H2 with respect to the unrestricted H atoms. The difference between the
# bonding energies of this and the previous run should be very close to the
# energy of the not-self-consistent unrestricted H-atom with respect to the
# restricted basic atom (calculation #2).

# == Excited state ==

$ADFBIN/adf <<eor
title   H2  excited

EPRINT
SFO eig ovl
END

ATOMS
H    0 .0      0.375
H    0 .0     -0.375
end

fragments
H    t21H
end

fragoccupations
H
s  1 // 0
subend
end

unrestricted

charge 0  2

IrrepOccupations
sigma.g   1 // 0
sigma.u   1 // 0
end

eor
```

```
# Finally the calculation of an excited state, with respect to unrestricted
# fragments. The excitation energy is obtained by comparing the energy with the
# energy of the ground state calculation. This difference compares reasonably,
# but not accurately, to the difference in one-electron ground state energies of
# the involved orbitals (Koopman's theorem).

# Note that excitation energies can also be calculated with Time-Dependent DFT,
# using the RESPONSE module of ADF. See related sample runs.
```

Example: Bond Energy analysis open-shell fragments: PCCP

Download PCCP_Unr_BondEnergy.run

```
#! /bin/sh


# This example illustrates advanced usage of the bond energy decomposition
# scheme used in ADF.

# A proper decomposition of an electron-pair bond energy requires specifying
# opposite spins for the unpaired electrons of the respective radical fragments,
# which can be done with the input key FragOccupations. The specified alpha- and
# beta-spin configurations of the radical fragments are shown in the output
# section B U I L D.

# Please note that if one neglects explicitly specifying opposite spins for the
# unpaired electrons of the fragments, each of them is treated as being half an
# alpha and half a beta electron and consequently, they enter into a spurious
# Pauli repulsive interaction. This results, among others, into the Pauli
# repulsion term being too repulsive and the orbital interaction term being too
# much stabilizing.

# The example consists of an analysis of the C-C single bond between two CP
# radicals in the four-atomic molecule PCCP. The CP fragment calculations used
# to provide the TAPE21 for the overall PCCP calculation must be done, for
# technical reasons, in the restricted mode ('cp_fpccp_asr'). The proper spins
# are then specified in the calculation of the overall molecule using the
# FragOccupations key ('pccp_fa1_as'). Note that this implies a slight
# approximation because the bond energy computed in this way refers to the
# energy difference between closed-shell PCCP and two CP radicals that are
# described by orbitals from a spin-restricted SCF calculation, which have been
# given an unrestricted occupation. In other words, the set of alpha- and beta-
# spin orbitals are identical and the effect of spin polarization is missing. In
# practice, this leads to minor energy differences with respect to the correct
# bond energy, that is, the energy difference between closed-shell PCCP and two
# CP radicals treated in the unrestricted mode, i.e., for which the set of
# alpha- and beta-spin orbitals are allowed to relax toward different solutions
# in the SCF procedure. This correction term can be computed directly by
# carrying out an unrestricted computation of the CP radical ('cp_fpccp_asu') using␣
↪the
# restricted CP radical ('cp_fpccp_asr') as a fragment.

# The pure orbital interaction effect of forming the electron bonding combination
# of the two radicals can be isolated from the full orbital interaction by carrying␣
↪out a
# separate calculation. In this calculation (pccp_fa1_pb) the bond energy analysis is␣
↪performed in
```

```
# the absence of any virtual CP fragment orbitals, using the key REMOVEFRAGORBITALS.


$ADFBIN/adf<<eor
TITLE cp_fpccp_asr
EPRINT
SFO eig ovl
END
XC
  GGA  BECKE PERDEW
END
ATOMS
  C        .0000    .0000    .6681
  P        .0000    .0000   2.2555
END
BASIS
 Type TZ2P
 Core Large
END
NumericalQuality Good
eor

rm logfile
mv TAPE21 t21cp_fpccp

$ADFBIN/adf<<eor
TITLE cp_fpccp_asu
EPRINT
SFO eig ovl
END
XC
  GGA  BECKE PERDEW
END
ATOMS
  C        .0000    .0000    .6681  f=CP
  P        .0000    .0000   2.2555  f=CP
END
FRAGMENTS
CP   t21cp_fpccp
END
FRAGOCCUPATIONS
CP
  SIGMA 3//2
  PI    2//2
SUBEND
END
UNRESTRICTED
CHARGE    0   1
NumericalQuality Good
eor

rm TAPE21 logfile

$ADFBIN/adf<<eor
TITLE pccp_fa1_pb
EPRINT
SFO eig ovl
ORBPOP  20  20
```

```
SUBEND
END
XC
  GGA  BECKE PERDEW
END
ATOMS
  P          .0000    .0000   2.2555  f=CP_A
  C          .0000    .0000    .6681  f=CP_A
  C          .0000    .0000   -.6681  f=CP_B
  P          .0000    .0000  -2.2555  f=CP_B
END
NumericalQuality Good
FRAGMENTS
CP_A   t21cp_fpccp
CP_B   t21cp_fpccp
END
SYMMETRY   C(LIN)
RemoveAllFragVirtuals
FRAGOCCUPATIONS
CP_A
  SIGMA 3//2
  PI    2//2
SUBEND
CP_B
  SIGMA 2//3
  PI    2//2
SUBEND
END
eor

rm logfile
mv TAPE21 PCCPpb.t21

$ADFBIN/adf<<eor
TITLE pccp_fa1_as
EPRINT
SFO eig ovl
ORBPOP  20  20
SUBEND
END
XC
  GGA  BECKE PERDEW
END
ATOMS
  P          .0000    .0000   2.2555  f=CP_A
  C          .0000    .0000    .6681  f=CP_A
  C          .0000    .0000   -.6681  f=CP_B
  P          .0000    .0000  -2.2555  f=CP_B
END
NumericalQuality Good
FRAGMENTS
CP_A   t21cp_fpccp
CP_B   t21cp_fpccp
END
SYMMETRY   C(LIN)
FRAGOCCUPATIONS
CP_A
  SIGMA 3//2
```

```
  PI    2//2
SUBEND
CP_B
  SIGMA 2//3
  PI    2//2
SUBEND
END
eor

rm logfile
mv TAPE21 PCCP.t21

$ADFBIN/adf<<eor
TITLE B3LYP restricted
XC
 Hybrid B3LYP
End

Atoms
  C         .0000    .0000    .6681
  P         .0000    .0000   2.2555
End
Basis
 Type TZ2P
 Core None
END
NumericalQuality Good
eor

rm logfile
mv TAPE21 CP.t21

$ADFBIN/adf<<eor
TITLE B3LYP unrestricted
XC
 Hybrid B3LYP
End

Atoms
  C         .0000    .0000    .6681  f=CP
  P         .0000    .0000   2.2555  f=CP
End
Fragments
 CP   CP.t21
End
Unrestricted
Charge 0 1
Fragoccupations
 CP
  SIGMA 7//6
  PI    4//4
 Subend
End
NumericalQuality Good
eor

rm TAPE21 logfile
```

```
$ADFBIN/adf<<eor
TITLE PCCP B3LYP PAIRBONDING
XC
 Hybrid B3LYP
End

Atoms
  P         .0000    .0000   2.2555  f=CP_A
  C         .0000    .0000    .6681  f=CP_A
  C         .0000    .0000   -.6681  f=CP_B
  P         .0000    .0000  -2.2555  f=CP_B
End
NumericalQuality Good
Fragments
 CP_A    CP.t21
 CP_B    CP.t21
End
RemoveAllFragVirtuals
Fragoccupations
 CP_A
  SIGMA 7//6
  PI    4//4
 Subend
 CP_B
  SIGMA 6//7
  PI    4//4
 Subend
End
eor

rm logfile
mv TAPE21 PCCPpb_hybrid.t21

$ADFBIN/adf<<eor
TITLE PCCP B3LYP
XC
 Hybrid B3LYP
End

Atoms
  P         .0000    .0000   2.2555  f=CP_A
  C         .0000    .0000    .6681  f=CP_A
  C         .0000    .0000   -.6681  f=CP_B
  P         .0000    .0000  -2.2555  f=CP_B
End
NumericalQuality Good
Fragments
 CP_A    CP.t21
 CP_B    CP.t21
End
Fragoccupations
 CP_A
  SIGMA 7//6
  PI    4//4
 Subend
 CP_B
  SIGMA 6//7
  PI    4//4
```

```
 Subend
End
eor

rm logfile
mv TAPE21 PCCPhybrid.t21
```

Example: Analysis of NaCl using ionic fragments: Na+ and Cl-

Download NaCl_ionicbonding.run

```
#! /bin/sh

# This example illustrates advanced usage of the bond energy decomposition
# scheme used in ADF.

# This example shows the energy decomposition analysis of NaCl with
# ionic fragments: Na+ and Cl-
# See also the example with neutral fragments: Na and Cl, both open shell atoms
#
# If the key REMOVEALLFRAGORBITALS is used one can approximately
# see what the electrostatic effect is on the fragment orbitals if one places
# the two ions together.
#
# Another way to see what the electrostatic effect is on the fragment orbitals is
# to use the key 'PRINT FMATSFO' in the full molecule calculation, where one can
# look at the expectation value of the Fock matrix of a fragment orbital in the
# presence of the other fragment.


$ADFBIN/adf<<eor
Atoms
 Cl  0.0 0.0 0.0
End
XC
  GGA  BECKE PERDEW
End
BASIS
 Type TZ2P
 Core None
End
charge -1
NumericalQuality Good
relativistic scalar zora
eor

rm logfile
mv TAPE21 Cl_min.t21

$ADFBIN/adf<<eor
Atoms
 Na 0.0 0.0 0.0
End
XC
  GGA  BECKE PERDEW
End
BASIS
```

```
 Type TZ2P
 Core None
End
charge 1
NumericalQuality Good
relativistic scalar zora
eor

rm logfile
mv TAPE21 Na_plus.t21

$ADFBIN/adf<<eor
TITLE ionic bonding
EPRINT
   SFO eig ovl
  ORBPOP  20  20
  SubEnd
End
XC
  GGA  BECKE PERDEW
End
Atoms
  Na 0.0 0.0 0.0 f=Na
  Cl  0.0 0.0 2.361 f=Cl
End
NumericalQuality Good
relativistic scalar zora
Fragments
  Na  Na_plus.t21
  Cl  Cl_min.t21
End
RemoveAllFragVirtuals
eor

rm logfile
mv TAPE21 NaCles.t21

$ADFBIN/adf<<eor
TITLE ionic bonding
EPRINT
  SFO eig ovl
  ORBPOP  20  20
  SubEnd
End
XC
  GGA  BECKE PERDEW
End
Atoms
  Na 0.0 0.0 0.0 f=Na
  Cl  0.0 0.0 2.361 f=Cl
End
NumericalQuality Good
relativistic scalar zora
Fragments
  Na  Na_plus.t21
  Cl  Cl_min.t21
End
eor
```

```
rm logfile
mv TAPE21 NaCl.t21
```

Example: Electron Pair bonding in NaCl: open shell fragments

```
Download NaCl_pairbonding.run
```

```
#! /bin/sh


# This example illustrates advanced usage of the bond energy decomposition
# scheme used in ADF.

# This example shows the energy decomposition analysis of NaCl with
# neutral fragments: Na and Cl, both open shell atoms
# See also the example with ionic fragments: Na+ and Cl-.

# A proper decomposition of an electron-pair bond energy requires specifying
# opposite spins for the unpaired electrons of the respective radical fragments,
# which can be done with the input key FragOccupations. The specified alpha- and
# beta-spin configurations of the radical fragments are shown in the output
# section B U I L D.

# Please note that if one neglects explicitly specifying opposite spins for the
# unpaired electrons of the fragments, each of them is treated as being half an
# alpha and half a beta electron and consequently, they enter into a spurious
# Pauli repulsive interaction. This results, among others, into the Pauli
# repulsion term being too repulsive and the orbital interaction term being too
# much stabilizing.

# The example consists of an analysis of the Na-Cl single bond between two
# open shell atoms. For technical reasons these atoms must be calculated spin-
↪restricted.
# The proper spins are then specified in the calculation of the overall molecule␣
↪using the
# FragOccupations key. Note that this implies a slight
# approximation because the bond energy computed in this way refers to the
# energy difference between closed-shell NaCl and two open shell atoms that are
# described by orbitals from a spin-restricted SCF calculation, which have been
# given an unrestricted occupation. In other words, the set of alpha- and beta-
# spin orbitals are identical and the effect of spin polarization is missing. In
# practice, this leads to minor energy differences with respect to the correct
# bond energy, that is, the energy difference between closed-shell NaCl and the two
# open shell atoms treated in the unrestricted mode, i.e., for which the set of
# alpha- and beta-spin orbitals are allowed to relax toward different solutions
# in the SCF procedure. This correction term can be computed directly by
# carrying out an unrestricted computation of the Na and Cl atom using the spin
# restricted atoms as a fragment.

# Note that the spin-restricted Cl atom that is used as fragment is given an electron
# configuration that is so called # 'prepared for bonding' in order to minimize
# the Pauli repulsion in the electron pair bond.
# In this case it means that there is a hole in the valence Cl p_z orbital,
# that will later form an electron pair bond with the valence s orbital of Na.

# The pure orbital interaction effect of forming the electron bonding combination
```

```
# of the two radicals can be isolated from the full orbital interaction by carrying␣
↪out a
# separate calculation. In this calculation the bond energy analysis is performed in
# the absence of any virtual fragment orbitals, using the key REMOVEALLFRAGVIRTUALS.


$ADFBIN/adf<<eor
Atoms
 Cl  0.0 0.0 0.0
End
XC
  GGA  BECKE PERDEW
End
BASIS
 Type TZ2P
 Core None
End
Symmetry D(LIN)
IrrepOccupations
  SIGMA.g 6
  SIGMA.u 2 1
  PI.u    8
End
NumericalQuality Good
relativistic scalar zora
eor

rm logfile
mv TAPE21 Cl_r.t21

$ADFBIN/adf<<eor
Atoms
 Na 0.0 0.0 0.0
End
XC
  GGA  BECKE PERDEW
End
BASIS
 Type TZ2P
 Core None
End
NumericalQuality Good
relativistic scalar zora
eor

rm logfile
mv TAPE21 Na_r.t21

$ADFBIN/adf<<eor
TITLE unrestricted Na
XC
  GGA  BECKE PERDEW
End
Atoms
  Na 0.0 0.0 0.0 f=Na
End
NumericalQuality Good
relativistic scalar zora
```

```
unrestricted
charge 0 1
Fragments
  Na  Na_r.t21
End
FragOccupations
  Na
    S 3 // 2
    P 3 // 3
  SubEnd
End
eor

rm logfile
mv TAPE21 Na_u.t21

$ADFBIN/adf<<eor
TITLE unrestricted Cl
XC
  GGA  BECKE PERDEW
End
Atoms
  Cl  0.0 0.0 0.0 f=Cl
End
NumericalQuality Good
relativistic scalar zora
unrestricted
charge 0 -1
symmetry nosym
Occupations IntegerAufbau
Fragments
  Cl  Cl_r.t21
End
FragOccupations
  Cl
    SIGMA.g 3 // 3
    SIGMA.u 1 // 2
    PI.u    4 // 4
  SubEnd
End
eor

rm logfile
mv TAPE21 Cl_u.t21

$ADFBIN/adf<<eor
TITLE pair bonding
EPRINT
   SFO eig ovl
  ORBPOP  20  20
  SubEnd
End
XC
  GGA  BECKE PERDEW
End
Atoms
  Na 0.0 0.0 0.0 f=Na
  Cl  0.0 0.0 2.361 f=Cl
```

```
End
NumericalQuality Good
relativistic scalar zora
Fragments
  Na  Na_r.t21
  Cl  Cl_r.t21
End
RemoveAllFragVirtuals
FragOccupations
  Na
    S 3 // 2
    P 3 // 3
  SubEnd
  Cl
    SIGMA.g 3 // 3
    SIGMA.u 1 // 2
    PI.u    4 // 4
  SubEnd
End
eor

rm logfile
mv TAPE21 NaClpb.t21

$ADFBIN/adf<<eor
TITLE pair bonding
EPRINT
  SFO eig ovl
  ORBPOP  20  20
  SubEnd
End
XC
  GGA  BECKE PERDEW
End
Atoms
  Na 0.0 0.0 0.0 f=Na
  Cl  0.0 0.0 2.361 f=Cl
End
NumericalQuality Good
relativistic scalar zora
Fragments
  Na  Na_r.t21
  Cl  Cl_r.t21
End
FragOccupations
  Na
    S 3 // 2
    P 3 // 3
  SubEnd
  Cl
    SIGMA.g 3 // 3
    SIGMA.u 1 // 2
    PI.u    4 // 4
  SubEnd
End
eor

rm logfile
```

```
mv TAPE21 NaCl.t21
```

Example: Bond Energy analysis meta-GGA, (meta-)hybrids: Zn2, Cr2, CrH

Download EDA_meta_gga_hybrid.run

```
#! /bin/sh

# This example illustrates the bond energy decomposition scheme using metaGGA or
# metahybrid or hybrid functionals in ADF.

# The first example is straightforward with closed shell atomic fragments: Zn_2.
# The second example has open shell atomic fragments: Cr_2, and the extra
# complication that spin symmetry breaking lowers the energy of the molecule,
# although the total S_z-value is zero. The third example has open shell atomic
# fragments, and the molecule is open shell: CrH.

# == Zn2 ==

# In the first example for Zn_2 the metahybrid TPSSh is used. In the bond energy
# analysis, the bond energy is split in a Pauli repulsion term, a steric
# interaction, and an orbital interaction.

$ADFBIN/adf <<eor
   Atoms
      Zn 0.0 0.0 0.0
      Zn 0.0 0.0 3.2
   End
   XC
    metahybrid TPSSh
   end
   Basis
    Type TZ2P
    Core None
   End
   dependency bas=1e-4
   BeckeGrid
      Quality good
   End
eor

rm TAPE21 logfile


# == Cr2 ==

# In the second example for Cr2 the metaGGA TPSS is used. Since the fragments
# are open shell, one may want to use unrestricted fragments, however, this is
# not possible in ADF. A fair approximation to a computation with unrestricted
# fragments can be achieved with the key FRAGOCCUPATIONS. You tell ADF that you
# want to treat the fragments as if they were unrestricted; this causes the
# program to duplicate the one-electron orbitals of the fragment: one set for
# spin-alpha and one set for spin-beta. You can then specify occupation numbers
# for these spin-unrestricted fragments, and occupy spin-alpha orbitals
# differently from spin-beta orbitals. Especially for the Pauli-repulsion it is
# important that one chooses the spin-occupations on the different fragments
# such that they are 'prepared for bonding'.
```

```
# Of course, the unrestricted fragments that you use in this way, are not self-
# consistent: different numbers of spin-alpha and spin-beta electrons usually
# result in different spatial orbitals and different energy eigenvalues for
# spin-alpha and spin-beta when you go to self-consistency, while here you have
# spatially identical fragment orbitals. Nevertheless it is often a fair
# approximation which gives you a considerable extension of analysis
# possibilities.

# Spin-symmetry breaking is enforced by the use of the key ModifyStartPotential
# in combination with the key key UNRESTRICTED. In the ADF output one can find
# that there is spin-density on both of the atoms.



$ADFBIN/adf <<eor
  Atoms
     Cr.1 0.0 0.0 0.0
     Cr.2 0.0 0.0 1.8
  End
  XC
     metagga TPSS
  end
  Basis
     Type TZ2P
     Core None
  End
  dependency bas=1e-4
  BeckeGrid
     Quality good
  End
  unrestricted
  charge 0 0
  ModifyStartPotential
     Cr.1 1 // 0
     Cr.2 0 // 1
  End
  FragOccupations
     Cr.1
        S 4 // 3
        P 6 // 6
        D 5 // 0
     SubEnd
     Cr.2
        S 3 // 4
        P 6 // 6
        D 0 // 5
     SubEnd
  End
eor

rm TAPE21 logfile



# In order to calculate the effect of self-consistency one should calculate the
# Cr atom spin-unrestrictedly.
```

```
$ADFBIN/adf <<eor
   Atoms
      Cr 0.0 0.0 0.0
   End
   XC
      metagga TPSS
   end
   Basis
      Type TZ2P
      Core None
   End
   BeckeGrid
      Quality good
   End
   unrestricted
   charge 0 6
   FragOccupations
      Cr
         S 4 // 3
         P 6 // 6
         D 5 // 0
    SubEnd
   End
eor

rm TAPE21 logfile


# == CrH ==

# In the third example for CrH the hybrid B3LYP is used.


$ADFBIN/adf <<eor
   Atoms
      Cr 0.0 0.0 0.0
      H  0.0 0.0 1.65
   End
   XC
      hybrid B3LYP
   end

   Basis
      Type TZ2P
      Core None
   End
   dependency bas=1e-4
   BeckeGrid
      Quality good
   End
   unrestricted
   charge 0 5
   FragOccupations
      Cr
         S 4 // 3
         P 6 // 6
         D 5 // 0
    SubEnd
```

```
      H
         S 0 // 1
    SubEnd
   End
eor

rm TAPE21 logfile


# In order to calculate the effect of self-consistency of spin-polarization on
# the atoms one should calculate the Cr and H atom spin-unrestrictedly.


$ADFBIN/adf <<eor
   Atoms
      Cr 0.0 0.0 0.0
   End
   XC
      hybrid B3LYP
   end
   Basis
      Type TZ2P
      Core None
   End
   dependency bas=1e-4
   BeckeGrid
      Quality good
   End
   unrestricted
   charge 0 6
   FragOccupations
      Cr
         S 4 // 3
         P 6 // 6
         D 5 // 0
      SubEnd
   End
eor

rm TAPE21 logfile

$ADFBIN/adf <<eor
   Atoms
      H 0.0 0.0 0.0
   End
   XC
      hybrid B3LYP
   end
   Basis
      Type TZ2P
      Core None
   End
   dependency bas=1e-4
   BeckeGrid
      Quality good
   End
   unrestricted
   charge 0 1
```

```
   FragOccupations
       H
          S 1 // 0
       SubEnd
    End
eor

rm TAPE21 logfile
```

Example: Spin-Orbit SFO analysis: TlH

Download TlH_SO_analysis.run

```
#! /bin/sh

# Application of the Spin-Orbit relativistic option (using double-group
# symmetry) to TlH with a detailed analysis of the spinors in terms of SFOs
# (Symmetrized Fragment Orbitals).

# In order to get the population analysis, one should have one scalar
# relativistic fragment, which is the whole molecule. The SFOs in this case are
# the scalar relativistic orbitals, which are already orthonormal, because one
# has only one fragment which is the whole molecule.

$ADFBIN/adf <<eor
title   TlH, scalar relativistic zora
BeckeGrid
 quality good
End
EPRINT
  SFO eig ovl
END
relativistic scalar zora
ATOMS
  Tl        0.0             0.0             0.0
  H         0.0             0.0             1.870
end
Basis
  Type TZ2P
  Core None
end
xc
  GGA BP86
end
PRINT SFO
eor

mv TAPE21 TlH.t21

# In order to get the population analysis, one should have one scalar
# relativistic fragment, which is the whole molecule, which is TlH in this case.

$ADFBIN/adf <<eor
title   TlH from fragment TlH,  with SpinOrbit coupling
BeckeGrid
 quality good
End
```

```
EPRINT
  SFO eig ovl
END
relativistic spinorbit zora
ATOMS
  Tl        0.0              0.0              0.0     f=TlH
  H         0.0              0.0              1.870   f=TlH
end
fragments
  TlH     TlH.t21
end
xc
  GGA BP86
end
PRINT SFO
eor

mv TAPE21 TlH_spinorbit.t21

# The output gives something like:
# ================================================================================
#
# =======================
#  Double group symmetry :  *   J1/2   *
# =======================
#                                     ===  J1/2:1 ===
#  Spinors expanded in SFOs
# Spinor:              21              22              23              24
#  occup:             1.00            1.00            1.00            0.00
#  ------             ----            ----            ----            ----
#  SFO SIGMA
#    13.alpha:   0.7614+0.0000i   0.0096+0.0000i   0.0052+0.0000i  -0.0006+0.0000i
#    14.alpha:   0.0154+0.0000i  -0.9996+0.0000i   0.0208+0.0000i  -0.0077+0.0000i
#    15.alpha:  -0.0146+0.0000i   0.0185+0.0000i   0.9849+0.0000i   0.1625+0.0000i
#  SFO PI:x
#     8.beta :   0.4578+0.0000i   0.0091+0.0000i   0.0112+0.0000i   0.0030+0.0000i
#     9.beta :   0.0005+0.0000i  -0.0074+0.0000i  -0.1119+0.0000i   0.6910+0.0000i
#  SFO PI:y
#     8.beta :   0.0000+0.4578i   0.0000+0.0091i   0.0000+0.0112i   0.0000+0.0030i
#     9.beta :   0.0000+0.0005i   0.0000-0.0074i   0.0000-0.1119i   0.0000+0.6910i
#
# ================================================================================

# Left out are a lot of small numbers. The meaning is that a spinor of J_z=1/2
# symmetry can have SIGMA and PI character, for example, the 21st spinor with
# occupation number 1.0, is approximately (21 J_z=1/2) = 0.76 (13 SIGMA alpha) +
# 0.46 (8 PI:x beta) + i 0.46 (8 PI:y beta)

# Next in the SFO contributions per spinor the real and imaginary spin alpha
# part and real and imaginary spin beta part are all summed together to give a
# percentage of a certain SFO. are summed. For example the 21st spinor has
# almost 60% (13 SIGMA) character.

# ========================================
#
# SFO contributions (%) per spinor
# Spinor:      21     22     23     24
```

```
# occup:      1.00   1.00   1.00   0.00
# ------      ----   ----   ----   ----

# SFO SIGMA
#    13:    57.97   0.01   0.00   0.00
#    14:     0.02  99.92   0.04   0.01
#    15:     0.02   0.03  97.01   2.64
# SFO PI:x
#     8:    20.96   0.01   0.01   0.00
#     9:     0.00   0.01   1.25  47.75
# SFO PI:y
#     8:    20.96   0.01   0.01   0.00
#     9:     0.00   0.01   1.25  47.75
#
# =====================================
```

Example: (Perturbed) localized molecular orbitals in twisted Ethene

Download LocMO_Resp.run

```
#!/bin/sh

# This is example for the calculation of localized molecular orbitals, and
# perturbed localized molecular orbitals. This example uses the Pipek-Mezey
# localization method (criterion PM in the LOCORB block key). With the
# AORESPONSE block key and the subkey opticalrotation the optical rotation
# dispersion is calculated. The PERTLOC key is specified in order to calculate
# perturbed localized molecular orbitals. The subkey gprime and beta of the key
# PERTLOC are needed to analyze the optical rotation. Note that PERTLOC only
# makes sense in combination with AORESPONSE and LOCORB.

$ADFBIN/adf <<eor
TITLE twisted ethene PM localization and response prop analysis

symmetry nosym

Basis
  Type DZP
  Core None
End

ATOMS
  C 0.000000 0.000000 0.020000
  C 1.350716 0.000000 -0.000000
  H -0.540179 -0.745428 -0.565703
  H -0.540179 0.745428 0.565703
  H 1.890895 -0.935093 -0.035837
  H 1.890895 0.935093 0.035837
END

PertLoc
  Efield
  Alfa
  Gprime
  Beta
  Dynamic
  Fulltens
```

```
End

LocOrb
  criterion PM
End

AOResponse
  opticalrotation
  giao
  frequency 1 .05 Hartree
End

eor
```

Example: Bader Analysis (AIM)

Download Bader.run

```sh
 #! /bin/sh



# The Bader atomic charges are calculated using a grid based method. Another
# possibility for Bader's analysis is to use the adf2aim utility such that a
# third party program Xaim can be used.

# With the QTAIM input block the ADF program will calculate Bader charges (AIM
# charges) using a grid based method.


$ADFBIN/adf <<eor
Title Calculate  Bader analysis for water

Atoms
    O        0.000000   0.000000   -0.001658
    H       -0.769048   0.000000    0.595209
    H        0.769048   0.000000    0.595209
End

Basis
  Type TZP
  Core none
  CreateOutput none
End

QTAIM
  AnalysisLevel Extended
  Spacing 0.1
End

eor

rm TAPE21 logfile


# Next a similar calculation for ferrocene is given
```

```
$ADFBIN/adf <<eor
Title Calculate Bader analysis for ferrocene

Atoms
    FE        0.000000    0.000000    0.000000
    C         1.215650    0.000000    1.600813
    C         0.375656   -1.156152    1.600813
    C        -0.983481   -0.714541    1.600813
    C        -0.983481    0.714541    1.600813
    C         0.375656    1.156152    1.600813
    C         1.215650    0.000000   -1.600813
    C         0.375656    1.156152   -1.600813
    C        -0.983481    0.714541   -1.600813
    C        -0.983481   -0.714541   -1.600813
    C         0.375656   -1.156152   -1.600813
    H         2.310827    0.000000    1.629796
    H         0.714085   -2.197727    1.629796
    H        -1.869498   -1.358270    1.629796
    H        -1.869498    1.358270    1.629796
    H         0.714085    2.197727    1.629796
    H         2.310827    0.000000   -1.629796
    H         0.714085    2.197727   -1.629796
    H        -1.869498    1.358270   -1.629796
    H        -1.869498   -1.358270   -1.629796
    H         0.714085   -2.197727   -1.629796
End

Basis
  Type TZP
  Core none
  CreateOutput none
End

QTAIM
  AnalysisLevel Extended
End

NumericalQuality Good

eor
```

Example: Bader Reactivity

Download Bader_Reactivity.run

```
#!/bin/sh

# With the CONCEPTUALDFT input block the ADF program will calculate
# certain conceptual DFT descriptors,
# such as atomic Fukui functions f+ and f-, the Koopmans dual descriptor, the
# Jenkin's metallicity and the atomic electronegativity by integrating
# corresponding properties over the atomic basin. Besides, MO populations per
# basin, the localization and delocalization indices (LI/DI), and the matrix of
# the condensed linear response function are computed.
# ConceptualDFT requires the symmetry to be NOSYM.
```

```
$ADFBIN/adf <<eor
Title Calculate Bader analysis for benzene-radical

Atoms
C       1.06434610      -0.70982731      -0.00004522
C      -0.17846421      -1.32042953      -0.00163715
C      -0.19884953      -2.73466243      -0.00296149
C       1.00825259      -3.45497625      -0.00262955
C       2.24377213      -2.78457605      -0.00098836
C       2.28117568      -1.37069091       0.00034973
H      -1.10692850      -0.74587638      -0.00187350
H      -1.15592647      -3.26173790      -0.00423999
H       0.98594687      -4.54659148      -0.00365721
H       3.17851894      -3.35030989      -0.00073866
H       3.23233560      -0.83454684       0.00163171
End


Basis
  Type TZ2P
  Core none
  CreateOutput none
End

UNRESTRICTED

CHARGE 0  1

BECKEGRID
   Quality good
End

CONCEPTUALDFT
  AnalysisLevel Full
  Electronegativity true
End

Symmetry NOSYM

XC
   GGA PBE
END

NOPRINT logfile

eor
```

Example: IQA/QTAIM analysis

```
Download IQA.run
```

```
#! /bin/sh

# An IQA/QTAIM analysis is performed to analyze selected interactions
# in formic acid dimer

$ADFBIN/adf -n1 <<eor
```

```
Title IQA Analysis for (HCOOH)2

ATOMS
1 C      -4.777280616716      -2.237679070420       0.027134922119
2 O      -4.316098016052      -1.091055032354      -0.045982864295
3 O      -4.095233471752      -3.376211708158      -0.036303239453
4 H      -5.858352595246      -2.416180987684       0.158163319700
5 H      -3.096816965327      -3.182195213950      -0.161436385033
6 C      -1.043719708731      -1.744256936204      -0.434121365892
7 O      -1.505389993838      -2.890945383925      -0.365122472158
8 O      -1.725417179069      -0.605694491898      -0.367592891810
9 H      -2.724119170709      -0.799847829461      -0.245243925045
10 H      0.037480148811      -1.565749121998      -0.563863149179
END


GUIBONDS
1 2 1 2.0
2 3 1 1.0
3 4 1 1.0
4 5 3 1.0
5 7 6 2.0
6 8 6 1.0
7 9 8 1.0
8 10 6 1.0
END


SYMMETRY NOSYM

BASIS
type TZP
core None
createoutput None
END

XC
GGA BLYP
DISPERSION Grimme3
END

NOPRINT LOGFILE

IQA
    AtomsToDo 2 5 7 9
    Print Normal
End

eor

mv TAPE21 IQA1.t21

# Another IQA analysis for a halogen bonded complex: F-Cl...NH3
# All interactions are calculated - Details printed (verbose mode)

$ADFBIN/adf -n1 <<eor
Title IQA Analysis for F-Cl...NH3

ATOMS
   F       0.000001582319       0.000000000000      -3.410515944395
```

```
   Cl      0.000001582319      0.000000000000     -1.623156530960
   N       0.000001582319      0.000000000000      0.662184762067
   H      -0.482577891417      0.835852167201      0.991662308171
   H      -0.482577891417     -0.835852167201      0.991662308171
   H       0.965160529792      0.000000000000      0.991662308171
END

GUIBONDS
1 2 1 1.0
2 4 3 1.0
3 5 3 1.0
4 6 3 1.0
END

SYMMETRY NOSYM

BASIS
type TZ2P
core None
createoutput None
END

XC
GGA BLYP
DISPERSION Grimme3
END

NOPRINT LOGFILE

IQA
    Print Verbose
End

eor

mv TAPE21 IQA2.t21
```

Example: Aromaticity indices with QTAIM

Download Aromaticity.run

```
#! /bin/sh

# The aromaticity indices Iring and MCI can be calculated with ADF. Calculation
# of these aromaticity indices invokes the QTAIM analysis automatically and uses
# its results so all accuracy considerations for the QTAIM key are also valid
# here. The rings should be specified using the AROMATICITY input block, one
# line per ring. The atoms must be listed in the order they are connected in the
# ring. Note that although a partial QTAIM analysis is performed, its results
# are not printed to the output file.


"$ADFBIN/adf" <<eor
Aromaticity
 1 2 3 4 5 6
 5 6 7 8 9 10
End
```

```
ATOMS
  1 C      -0.000000000000     -1.402840000000     -1.219800000000
  2 C       0.000000000000     -0.698820000000     -2.427560000000
  3 C       0.000000000000      0.698820000000     -2.427560000000
  4 C       0.000000000000      1.402840000000     -1.219800000000
  5 C       0.000000000000      0.708650000000     -0.000000000000
  6 C      -0.000000000000     -0.708650000000      0.000000000000
  7 C      -0.000000000000     -1.402840000000      1.219800000000
  8 C      -0.000000000000     -0.698820000000      2.427560000000
  9 C      -0.000000000000      0.698820000000      2.427560000000
 10 C       0.000000000000      1.402840000000      1.219800000000
 11 H      -0.000000000000     -2.486200000000     -1.238920000000
 12 H      -0.000000000000     -1.237890000000     -3.366130000000
 13 H       0.000000000000      1.237890000000     -3.366130000000
 14 H       0.000000000000      2.486200000000     -1.238920000000
 15 H      -0.000000000000     -2.486200000000      1.238920000000
 16 H      -0.000000000000     -1.237890000000      3.366130000000
 17 H       0.000000000000      1.237890000000      3.366130000000
 18 H       0.000000000000      2.486200000000      1.238920000000
END

BASIS
 type TZP
 core None
 createoutput None
END

XC
 GGA PBE
END

NOPRINT LOGFILE

eor
```

Example: Charge model 5 (CM5)

Download CM5_chargemodel.run

```
#!/bin/sh



# With the key CM5 charge model 5 (CM5) atomic charges are calculated The CM5
# model is especially designed such that one can reasonably accurately predict
# dipole moments with only atomic CM5 charges.


$ADFBIN/adf <<eor
Title Acetone
Atoms
 O      0.000000    0.000000    1.396962
 C      0.000000    0.000000    0.188800
 H      0.871902    1.314824   -1.265092
 C      0.000000   -1.275657   -0.611044
 H      0.000000    2.142271    0.042196
 H      0.000000   -2.142271    0.042196
```

```
 H       0.871902   -1.314824   -1.265092
 C       0.000000    1.275657   -0.611044
 H      -0.871902    1.314824   -1.265092
 H      -0.871902   -1.314824   -1.265092
end

Basis
   type TZP
   core none
End

CM5

XC
 MetaGGA M06-L
End
NumericalQuality good
eor

rm TAPE21 logfile


$ADFBIN/adf <<eor
Title Nitrobenzene
Atoms
 C       0.000000    0.000000   -2.503225
 C       0.000000    1.205864   -1.812565
 C       0.000000    1.214022   -0.427327
 C       0.000000    0.000000    0.239613
 C       0.000000   -1.214022   -0.427327
 C       0.000000   -1.205864   -1.812565
 H       0.000000   -2.131001    0.143627
 H       0.000000   -2.142362   -2.354096
 H       0.000000    2.142362   -2.354096
 H       0.000000    2.131001    0.143627
 H       0.000000    0.000000   -3.585546
 N       0.000000    0.000000    1.715151
 O       0.000000   -1.084613    2.278800
 O       0.000000    1.084613    2.278800
end

Basis
   type TZP
   core none
End

CM5

XC
 MetaGGA M06-L
End
NumericalQuality good
eor

rm TAPE21 logfile
```

Example: Bond Orders

Download BondOrder.run

```sh
#!/bin/sh


# With the key BONDORDER a bond order analysis is performed based on SFOs. Note
# that symmetry used in the calculation should be NOSYM.


$ADFBIN/adf <<eor
title benzene BP/SZ bondorders tol=0.05

atoms Z-matrix
 C  0 0 0
 C  1 0 0  1.38476576
 C  2 1 0  1.38476576 120.0
 C  3 2 1  1.38476576 120.0 0.0
 C  4 3 2  1.38476576 120.0 0.0
 C  5 4 3  1.38476576 120.0 0.0
 H  2 1 3  1.07212846 120.0 180.0
 H  3 2 4  1.07212846 120.0 180.0
 H  4 3 5  1.07212846 120.0 180.0
 H  5 4 3  1.07212846 120.0 180.0
 H  6 5 4  1.07212846 120.0 180.0
 H  1 2 3  1.07212846 120.0 180.0
end

basis
 Type SZ
 Core None
end

symmetry NOSYM

xc
  gga becke perdew
end

bondorder tol=0.05 printall

noprint sfo
eor

mv TAPE21 benzene.t21
rm logfile

$ADFBIN/adf <<eor
title ethylene BP/SZ bondorders-standard output

atoms
 C             0.0000    0.0000    0.0000
 C             1.3319    0.0000    0.0000
 H            -0.5723    0.9299    0.0000
 H            -0.5725   -0.9297    0.0000
 H             1.9043    0.9298    0.0000
 H             1.9042   -0.9298    0.0000
```

```
end

basis
 Type SZ
 Core None
end

symmetry NOSYM

xc
 gga becke perdew
end

bondorder
noprint sfo
eor

mv TAPE21 ethelene.t21
rm logfile

$ADFBIN/adf <<eor
title Cr2 BP/TZP - bondorder - standard printout

atoms
 Cr 0.0 0.0 -0.8
 Cr 0.0 0.0  0.8
end

basis
 Type TZP
 Core None
end

symmetry NOSYM

xc
 gga becke perdew
end

geometry
  Converge grad=0.0001
end

bondorder

noprint sfo
eor

mv TAPE21 Cr2.t21
rm logfile

$ADFBIN/adf <<eor
title Mo2 BP/TZP - bondorder - standard printout

atoms
 Mo 0.0 0.0 -1.0
 Mo 0.0 0.0  1.0
end
```

```
basis
 Type TZP
 Core Small
end

symmetry NOSYM

relativistic scalar ZORA
xc
 gga becke perdew
end

geometry
end

bondorder
noprint sfo
eor

mv TAPE21 Mo2.t21
rm logfile


$ADFBIN/adf <<eor
title NaCl BP/TZP - bondorder - standard printout

atoms
 Na 0.0 0.0 0.0
 Cl 0.0 0.0 2.377
end

basis
 Type DZP
 Core None
end

symmetry NOSYM

xc
 gga becke perdew
end

bondorder printall

noprint sfo
eor

mv TAPE21 NaCl.t21
rm logfile
```

Example: NOCV: ethylene – Ni-diimina and H+ – CO

```
Download Diimina_NOCV.run
```

```
#!/bin/sh

# Example for calculation of ETS-NOCV for spin-restricted fragments. ETS-NOCV:
# energy analysis using the Natural Orbitals for Chemical Valence. The ethylene
# molecule and a Ni-diimina form a complex together. This example will be
# discussed first. The other example is H+ and CO form together HCO+ , this
# example is similar to the discussed example. All electron basis sets are
# required.

# First the two fragments are calculated.


$ADFBIN/adf <<eor
Title et-----Ni-diimina: ethylene run


atoms cartesian
C   -0.430177075 -1.815433265  0.860288229
C   -0.363705637 -1.910722338 -0.515633302
H    0.533109934 -2.284970854 -1.016904201
H   -1.279922499 -1.884673940 -1.115144723
H   -1.389295819 -1.753589602  1.377541080
H    0.440296224 -2.041861443  1.484489314
end

basis
 Type DZP
 Core Small
end

symmetry NOSYM

xc
 gga scf becke perdew
end

eor

mv TAPE21 t21.etfrag


$ADFBIN/adf <<eor
Title et-----Ni-diimina: Ni-diimina run

atoms cartesian
Ni   0.022615419  0.037783871  0.025751533
N    0.386170317  1.871072585  0.306265538
C    1.612863056  2.248007643  0.148716016
C    2.540686607  1.163409862 -0.183603690
N    1.976290003  0.008161589 -0.301176178
H   -0.288333328  2.609667211  0.546869047
H    1.942601454  3.283060847  0.269249237
H    3.613259273  1.338293482 -0.302134814
H    2.621707427 -0.766258151 -0.517479818
H   -1.351756655  0.253389698  0.386197419
end

charge 1
```

```
basis
 Type DZP
 Core Small
end

symmetry NOSYM

xc
  gga scf becke perdew
end

eor

mv TAPE21 t21.Nifrag


# Next these fragments are used in the calculation of the full complex. The keys
# ETSNOCV and 'PRINT etslowdin' are needed in this case to to analyze the
# bonding in the molecule with respect to the fragments. The symmetry must be
# NOSYM.


$ADFBIN/adf <<eor
Title et-----Ni-diimina run

atoms
Ni   0.022615419  0.037783871  0.025751533 f=k
N    0.386170317  1.871072585  0.306265538 f=k
C    1.612863056  2.248007643  0.148716016 f=k
C    2.540686607  1.163409862 -0.183603690 f=k
N    1.976290003  0.008161589 -0.301176178 f=k
H   -0.288333328  2.609667211  0.546869047 f=k
H    1.942601454  3.283060847  0.269249237 f=k
H    3.613259273  1.338293482 -0.302134814 f=k
H    2.621707427 -0.766258151 -0.517479818 f=k
H   -1.351756655  0.253389698  0.386197419 f=k
C   -0.430177075 -1.815433265  0.860288229 f=m
C   -0.363705637 -1.910722338 -0.515633302 f=m
H    0.533109934 -2.284970854 -1.016904201 f=m
H   -1.279922499 -1.884673940 -1.115144723 f=m
H   -1.389295819 -1.753589602  1.377541080 f=m
H    0.440296224 -2.041861443  1.484489314 f=m
end

charge 1

fragments
m t21.etfrag
k t21.Nifrag
end

symmetry NOSYM

xc
 gga scf becke perdew
end
```

```
ETSNOCV
End

print etslowdin

eor


# Next one could do densf calculations, to view the natural orbitals in this
# method, see also the the documentation for the densf analysis program and the
# ADF-GUI. Input is the TAPE21 of the molecular calculation
#
#    $ADFBIN/densf << eor
#    GRID MEDIUM
#    End
#    NOCV
#      THRESH 0.01
#    END
#    eor
#    mv TAPE41 nocv2.t41
#
#    $ADFBIN/densf << eor
#    GRID MEDIUM
#    End
#    NOCV
#      ALL
#    END
#    eor
#    mv TAPE41 nocv3.t41
```

Download Hplus_CO_etsnocv.run

```
#! /bin/sh


$ADFBIN/adf << eor
Title [H]+-----[CO] run from H+ and CO fragments, FULL electron calc.!

atoms cartesian
H       -0.010992    0.000002    0.148581
end

charge 1

basis
H  $ADFRESOURCES/SZ/H
end

symmetry NOSYM

SCF
  Iterations 2500
  Converge 1E-6
end

xc
  gga becke perdew
end
```

```
NumericalQuality Good

eor

mv TAPE21 t21.Hydrogen

$ADFBIN/adf << eor
Title [CO] run

atoms cartesian
    C         0.005191    0.000001    1.248610
    O         0.021536    0.000000    2.359675
end

basis
C   $ADFRESOURCES/SZ/C
O   $ADFRESOURCES/SZ/O
end

symmetry NOSYM

SCF
  Iterations 2500
  Converge 1E-6
end

xc
  gga becke perdew
end

NumericalQuality Good

eor

mv TAPE21 t21.COfragment


$ADFBIN/adf << eor
Title [H]+--[CO], etsnocv acitivated by etsnocv and print etslowdin

atoms
    H        -0.010992    0.000002    0.148581 f=f1
    C         0.005191    0.000001    1.248610 f=f2
    O         0.021536    0.000000    2.359675 f=f2
end

charge 1

fragments
f1 t21.Hydrogen
f2 t21.COfragment
end

symmetry NOSYM

SCF
  Iterations 800
  Converge 1E-6
```

```
end

xc
  gga becke perdew
end

NumericalQuality Good

ETSNOCV RHOKMIN=1.e-3 EKMIN=1.5 ENOCV=0.05

print etslowdin

eor


$ADFBIN/densf <<eor
Grid Medium
End
NOCV
  1
  11
  2
  3
  9
  10
END
eor
mv TAPE41 nocv1.t41

$ADFBIN/densf <<eor
Grid Medium
End
NOCV
  THRESH 0.01
END
eor
mv TAPE41 nocv2.t41

$ADFBIN/densf <<eor
Grid Medium
End
NOCV
  ALL
END
eor
mv TAPE41 nocv3.t41
```

Example: NOCV: CH2 – Cr(CO)5

Download NOCV_CrCO5-CH2.run

```
#! /bin/sh

# Example for calculation of ETS-NOCV for spin-restricted fragments. ETS-NOCV:
# energy analysis using the Natural Orbitals for Chemical Valence. The CH2
# molecule and Cr(CO)5 are the fragments, which form Cr(CO)5 CH2 molecule.
```

```
# First the two fragments are calculated.


$ADFBIN/adf <<eor
Title CrCO5--[CH2] run from CrCO5 and CH2 closed shell fragments,FULL electron calc.!

atoms cartesian
    C        -0.429104    1.732058   -0.225052
    H         0.407023    2.440417   -0.352323
    H        -1.385325    2.281354   -0.254124
end

basis
 Type DZP
 Core None
end

symmetry NOSYM

xc
  gga becke perdew
end

eor

mv TAPE21 t21.CH2


$ADFBIN/adf <<eor
Title [CrCO5] run

atoms cartesian
    Cr       -0.248053   -0.169062    0.005810
    C        -0.072963   -2.080685    0.229583
    O         0.030811   -3.223220    0.361925
    C        -0.182894    0.049840    1.909128
    O        -0.142780    0.212309    3.050403
    C        -0.299940   -0.409118   -1.894730
    O        -0.331795   -0.521589   -3.042336
    C        -2.138631   -0.242152    0.075713
    O        -3.295036   -0.249916    0.115045
    C         1.624487    0.092244   -0.083118
    O         2.763411    0.288575   -0.140976
end

basis
 Type DZP
 Core None
 Cr $ADFRESOURCES/TZP/Cr
end

symmetry NOSYM

xc
  gga becke perdew
end

eor
```

```
mv TAPE21 t21.Crfragment

# Next these fragments are used in the calculation of the full complex. The keys
# ETSNOCV and 'PRINT etslowdin' are needed in this case to to analyze the
# bonding in the molecule with respect to the fragments. The symmetry must be
# NOSYM.


$ADFBIN/adf <<eor
Title CrCO5--[CH2], etsnocv acitivated by etsnocv and print etslowdin

atoms
    C         -0.429104    1.732058   -0.225052 f=f1
    Cr        -0.248053   -0.169062    0.005810 f=f2
    C         -0.072963   -2.080685    0.229583 f=f2
    O          0.030811   -3.223220    0.361925 f=f2
    C         -0.182894    0.049840    1.909128 f=f2
    O         -0.142780    0.212309    3.050403 f=f2
    C         -0.299940   -0.409118   -1.894730 f=f2
    O         -0.331795   -0.521589   -3.042336 f=f2
    C         -2.138631   -0.242152    0.075713 f=f2
    O         -3.295036   -0.249916    0.115045 f=f2
    C          1.624487    0.092244   -0.083118 f=f2
    O          2.763411    0.288575   -0.140976 f=f2
    H          0.407023    2.440417   -0.352323 f=f1
    H         -1.385325    2.281354   -0.254124 f=f1
end

fragments
f1 t21.CH2
f2 t21.Crfragment
end

symmetry NOSYM

xc
  gga becke perdew
end

ETSNOCV RHOKMIN=1.e-3 EKMIN=1.5 ENOCV=0.05

print etslowdin

eor
```

Example: NOCV: CH3 – CH3

Download CH3_CH3_etsnocv.run

```
#! /bin/sh


# Example for calculation of ETS-NOCV for unrestricted fragments. ETS-NOCV:
# energy analysis using the Natural Orbitals for Chemical Valence. The ethane
# molecule is built from two methyl radicals
```

```
# First the two methyl fragments are calculated. The fragments should be spin-
# restricted.


$ADFBIN/adf <<eor
Title CH3-CH3 built from CH3 radicals,  FULL electron calc.!

atoms cartesian
    C        0.019664    -0.034069     0.009101
    H        0.039672    -0.069395     1.109620
    H        1.063205    -0.065727    -0.341092
    H       -0.474230    -0.953693    -0.341621
end

basis
H  $ADFRESOURCES/DZP/H
C  $ADFRESOURCES/DZP/C
end

symmetry NOSYM

SCF
  Iterations 2500
  Converge 1E-6
end

xc
  gga scf becke perdew
end


eor

mv TAPE21 t21.frag1

$ADFBIN/adf <<eor
Title CH3 radical


atoms cartesian
    C       -0.703210     1.217999    -0.497874
    H       -0.723753     1.252869    -1.598316
    H       -1.746567     1.250049    -0.147169
    H       -0.208833     2.137544    -0.147653
end

basis
H  $ADFRESOURCES/DZP/H
C  $ADFRESOURCES/DZP/C
end

symmetry NOSYM

SCF
  Iterations 2500
  Converge 1E-6
end
```

```
xc
  gga scf becke perdew
end


eor

mv TAPE21 t21.frag2


# Next these fragments are used in the calculation of the molecule ethane, using
# the key FRAGOCCUPATIONS to use an unrestricted fragment occupation for the
# methyl radicals, such that they are prepared for bonding. In the one fragment
# the singly occupied orbital will be an alpha-orbital, and in the other fragment
# the singly occupied orbital will be a beta-orbital, such that the calculated
# Pauli repulsion between the fragments will be small.

# The keys ETSNOCV and 'PRINT etslowdin-unrestricted' are needed in this case to
# to analyze the bonding in a molecule with unpaired electrons in the fragments.
# The symmetry must be NOSYM.


$ADFBIN/adf <<eor
Title final [CH3]-[CH3], etsnocv activated by etsnocv and etslowdin-unrestricted

atoms
    C         0.019664   -0.034069    0.009101 f=f1
    H         0.039672   -0.069395    1.109620 f=f1
    H         1.063205   -0.065727   -0.341092 f=f1
    H        -0.474230   -0.953693   -0.341621 f=f1
    C        -0.703210    1.217999   -0.497874 f=f2
    H        -0.723753    1.252869   -1.598316 f=f2
    H        -1.746567    1.250049   -0.147169 f=f2
    H        -0.208833    2.137544   -0.147653 f=f2
end

fragments
f1 t21.frag1
f2 t21.frag2
end

fragoccupations
f1
A 5 // 4
subend
f2
A 4 // 5
subend
end

symmetry NOSYM

SCF
  Iterations 800
  Converge 1E-6
end

xc
```

```
  gga scf becke perdew
end

ETSNOCV RHOKMIN=1.e-3 EKMIN=1.5 ENOCV=0.05

PRINT etslowdin-unrestricted


eor


# Next densf calculations, to view the natural orbitals in this method, see
# also the the documentation for the densf analysis program and the ADF-GUI.
# Input is the TAPE21 of the molecular calculation.


$ADFBIN/densf <<eor
Grid Medium
End

NOCV
 Alpha
  1
  2
  59
  60
 Beta
  1
  2
  59
  60
 END
eor
mv TAPE41 nocv1.t41

$ADFBIN/densf <<eor
Grid Medium
End

NOCV
  THRESH 0.01
  RESTSUM
END
eor
mv TAPE41 nocv2.t41

$ADFBIN/densf <<eor
Grid Medium
End

NOCV
  ALPHA
    SUM 1
  BETA
    SUM 1
  RESTSUM
END
eor
```

```
mv TAPE41 nocv3.t41

$ADFBIN/densf <<eor
Grid Medium
End

NOCV
  SUMBELOW 0.5
END
eor
mv TAPE41 nocv4.t41
```

Example: Activation Strain Model Analysis using PyFrag

Download pyfrag.run

```bash
#!/bin/bash



# The trajectory is a concatenated xyz-file:

cat << eor > coords.xyz
C       -1.741968   -2.220880    0.000000
Pd      -2.137509   -0.237843    0.000000
H       -2.809570   -2.499547    0.000000
H       -1.265288   -2.629932    0.895677
H       -1.265288   -2.629932   -0.895677
H       -0.755099   -0.885698    0.00000

C       -1.741968   -2.220880    0.000000
Pd      -2.137509   -0.250000    0.000000
H       -2.809570   -2.499547    0.000000
H       -1.265288   -2.629932    0.895677
H       -1.265288   -2.629932   -0.895677
H       -0.755099   -0.885698    0.00000
eor

# Input options for ADF:

cat << eor > adfinputfile
NumericalQuality=VeryGood
XC.GGA=OPBE
Basis.core=None
Charge=0
Symmetry=AUTO
RELATIVISTIC=Scalar ZORA
eprint.sfo=eig ovl
eor

# Here we call the PyFrag script.

# PyFrag conducts a total of six single point calculations, one for each fragment and
↪the
# whole system for each of the two geometries in the trajectory, respectively.
# The intermediate results for each geometry are printed during the calculation.
# At the end of its run PyFrag summarizes all results in a table which is also
↪written
```

```
# to an output file PyFrag<SystemName>.txt

$ADFBIN/startpython $ADFHOME/scripting/standalone/pyfrag/PyFrag.py  \
--xyzpath coords.xyz \
--fragment 2 --fragment 1 3 4 5 6 --strain 0 --strain -554.1 \
--bondlength 1 6 1.093 \
--angle 2 1 6 180 \
--VDD 2 1 6 \
--hirshfeld frag1 \
--irrepOI AA \
--irrepOI AAA \
--orbitalenergy frag1 HOMO \
--orbitalenergy frag2 LUMO \
--orbitalenergy AA frag2 5 \
--population frag1 HOMO \
--population AA frag2 5 \
--overlap frag1 HOMO frag2 LUMO \
--overlap S frag1 5 AA frag2 4 \
--adfinput basis.type=DZP \
--adfinputfile adfinputfile
```

14.6.2 DOS: Density of states

Example: Density of States: Cu4CO

Download DOS_Cu4CO.run

```
#! /bin/sh

# This sample illustrates the DOS property program to compute density-of-states
# data, for energy-dependent analysis.

# First, the Cu_4 CO molecule is calculated (ADF), using single-atom fragments.


$ADFBIN/adf <<eor
title  Cu4CO (3,1) from atoms

units
length bohr
end

atoms
1. Cu    2.784     0.0               0.0
2. Cu   -1.392    2.411014724135877  0.0
3. Cu   -1.392   -2.411014724135877  0.0
4. Cu    0.0      0.0               -3.9371705576466964
5. C     0.0      0.0                2.65
6. O     0.0      0.0                4.91
end

Basis
  Type TZP
  Core small
end
```

```
XC
  GGA  PostSCF    Becke Perdew
END
PRINT SFO

eor


# The PostSCF feature in the specification of the XC functional is used: the
# 'Becke-Perdew' GGA corrections are not included self-consistently but applied
# to the energy evaluation after the self-consistent LDA solution has been
# obtained.

# The utility program dos requires a file named TAPE21 in the current directory,
# unless overridden using a TAPE21 keyword (not used in this example).


$ADFBIN/dos <<eor
file dostxt

energyrange npoint=36 e-start=-25 e-end=10

tdos

! Cu 3d partial DOS
gpdos
  a1   14:22 32:34
  a2    5:10
  e1:1 18:32 37:42
  e1:2 18:32 37:42
end

! The same but using BAS
gpdos
  BAS 17:34 57:74 97:114 137:154
end

! The same as above, but using much less complicated input
gpdos
  ATYPE Cu d
end

! Overlap PDOS Cu 3d and CO 2p
opdos
  ATYPE Cu 3d
SUBEND
  ATOM 5 2p
  ATOM 6 2p
end

eor


echo "DOS result file"
echo "==============="
cat dostxt
echo "END of DOS result file"
```

```
mv TAPE21 Cu4CO.t21

# Here, the total density of states, as well as various partial densities of
# states, are computed. You may feed the results found in the dostxt file into a
# plotting program such as gnuplot. The result is not displayed here. See the
# ADF manual for more detailed info about the dos program.
```

14.6.3 Third party analysis software

Example: adf2aim: convert an ADF TAPE21 to WFN format (for Bader analysis)

Download AIM_HF.run

```
#! /bin/sh

# One can calculate Bade atomic charges and other Atoms in Molecule properties
# directly in ADF using a grid based method, see also an example that uses the
# grid based method. Another possibility for Bader's analysis, an example is
# described here, is to use the adf2aim utility such that a third party program
# Xaim can be used.

# The ADF utility adf2aim (original name rdt21, now part of the ADF package)
# developed by Xavi Lopez, Engelber Sans and Carles Bo converts an ADF TAPE21 to
# WFN format (for Bader analysis).

# The WFN file is an input file for the third party program Xaim (see
# http://www.quimica.urv.es/XAIM for details), which is a graphical user
# interface to programs that can perform the Bader analysis.

# Usage of adf2aim:


$ADFBIN/adf << eor
TITLE HF
ATOMS
  1. H  .0000  .0000  .0000
  2. F  .0000  .0000  0.917
End
Basis
End
eor

$ADFBIN/adf2aim TAPE21
echo ""
echo 'Contents of rdt21.res:'
cat rdt21.res | grep -v RunTime
echo 'Contents of WFN:'
cat WFN | grep -v RunTime

mv TAPE21 HF_restricted.t21
rm logfile rdt21.res WFN


# In an unrestricted calculation adf2aim will ask if the MOs should be split in
# two files. In this example the answer is yes (y).
```

```
$ADFBIN/adf << eor
TITLE HF
ATOMS
  1. H  .0000  .0000  .0000
  2. F  .0000  .0000  0.917
End
UNRESTRICTED
CHARGE 0 0
Basis
End
eor


$ADFBIN/adf2aim TAPE21 <<eor
y
eor

mv TAPE21 HF_unrestricted.t21

echo ""
echo 'Contents of rdt21.res:'
cat rdt21.res| grep -v RunTime
echo 'Contents of WFN-alpha:'
cat WFN-alpha| grep -v RunTime
echo 'Contents of WFN-beta:'
cat WFN-beta| grep -v RunTime
```

Example: NBO analysis: adfnbo, gennbo

Download H2O_ADFNBO.run

```
#! /bin/sh

# Dr. Autschbach, SCM, and Prof. Weinhold have collaborated to prepare a simple
# in put file generator, called adfnbo, for the GENNBO program of Prof.
# Weinholds Natural Bond Orbital (NBO) 6.0 package: http://nbo6.chem.wisc.edu
# The GENNBO executable is included in the ADF distribution and can be enabled
# via the license file ( info@scm.com).


$ADFBIN/adf <<eor
Title simple NBO example for water

Atoms    Z-Matrix
O   0 0 0
H   1 0 0   0.9
H   1 2 0   0.9  100
End

Basis
CORE NONE
TYPE DZ
End

FULLFOCK
AOMAT2FILE
```

```
SAVE TAPE15
SYMMETRY NOSYM

eor

# Note added: recommended is to use the key 'spherical' in the adfnbo input.

# A File named FILE47 is generated by adfnbo which is an input file for the
# general NBO program gennbo. ADF needs to write some data to file, which is
# done by including these keywords in the adf input file:


# == GENNBO ==

# Next a brief summary of the capabilities of GENNBO is given (by Prof.
# Weinhold). GENNBO implements most capabilities of the full NBO 6.0 program
# suite as described on the NBO website: http://nbo6.chem.wisc.edu These include
# determination of natural atomic orbitals (NAOs), bond orbitals (NBOs), and
# localized MOs (NLMOs), as well as the associated NPA (atomic charges and
# orbital populations) and NRT (resonance structures, weightings, bond orders)
# valence descriptors, for a wide variety of uncorrelated and correlated
# (variational, perturbative, or density functional) theoretical levels. GENNBO-
# supported options include all keywords except those explicitly requiring
# interactive communication with the host electronic structure system (viz.,
# $DEL deletions, NEDA, NCS, NJC). The GENNBO program typically sits
# conveniently on the PC desktop, ready to analyze (or re-analyze at will, with
# altered options) the final results of a complex ADF calculation performed on a
# remote cluster.

# GENNBO 'communicates' with the original ADF calculation through an archive
# file (JOB.47 file, preserving all necessary details of the final density) that
# is initially generated by ADF and subsequently becomes the input file for
# GENNBO. The .47 file contains a standard $NBO ... $END keylist that can be
# edited with a standard word processor or text editor to include chosen NBO
# keyword options, just as though they might have appeared in the original input
# stream of an interactive ADFNBO run. The stand-alone GENNBO program therefore
# allows many alternative NBO analysis options to be explored at leisure,
# without costly re-calculation of the wave function.

# Using the GENNBO executable is possible only if NBO6 is enabled in your
# license file by SCM (info@scm.com). In that case you will get access to an
# NBO 6.0 manual in electronic form that explains in detail how GENNBO can be
# used and how the output should be interpreted.


$ADFBIN/adfnbo <<eor
write
fock
TESTJOB
eor

echo " "
echo " "
echo "Contents of FILE47 ="
echo " "
echo " "
cat FILE47
```

```
$ADFBIN/gennbo6 FILE47

echo " "
echo " "
echo "Contents of adfnbo.37 = "
echo " "
echo " "
cat adfnbo.37

echo " "
echo " "
echo "Contents of adfnbo.39 ="
echo " "
echo " "

cat adfnbo.39

echo " "
echo " "
echo "Contents of adfnbo.49 ="
echo " "
echo " "
cat adfnbo.49
```

Example: NBO analysis: EFG

Download AlCl3_efgnbo.run

```
#! /bin/sh

# Example shows an NBO analysis of an EFG calculation for AlCl3 .

# In the ADF input one then needs to include the QTENS (EFG calculation) and
# include the subkey EFG of the key AORESPONSE. The BeckeGrid quality is good,
# which may be needed to accurately calculate core properties. Other keywords
# are necessary because of the NBO analysis afterwards. Note that ADF, ADFNBO,
# and GENNBO have to run several times.


$ADFBIN/adf <<eor
Title AlCl3

atoms
 Al     0.000000     0.000000     -0.237368
 Cl     1.808813     0.000000      0.807083
 Cl     0.000000     0.000000     -2.326083
 Cl    -1.808813     0.000000      0.807083
End

xc
 lda vwn
 gga revPBE
end

Symmetry NOSYM

BeckeGrid
```

```
 Quality good
End

Basis
 Type TZP
 Core none
End

Aoresponse
 efg 1 nbo
end

qtens

save TAPE15
FULLFOCK
AOMAT2FILE

eor


# end ADF, run gennbo
# ==================


$ADFBIN/adfnbo <<eor
 write
 spherical
 fock
TESTJOB
eor

rm -f adfnbo.37 adfnbo.39 adfnbo.49 adfnbo.48

$ADFBIN/gennbo6 FILE47

$ADFBIN/adfnbo <<eor
 copy
 spherical
 fock
TESTJOB
eor

$ADFBIN/adfnbo <<eor
 read
 spherical
 fock
TESTJOB
eor

rm -f adfnbo.37 adfnbo.39 adfnbo.49 adfnbo.48


# end gennbo, rerun ADF with NBO property analysis
# =================================================

rm -f TAPE21 TAPE13 TAPE15
```

```
$ADFBIN/adf <<eor
Title AlCl3

atoms
 Al      0.000000      0.000000     -0.237368
 Cl      1.808813      0.000000      0.807083
 Cl      0.000000      0.000000     -2.326083
 Cl     -1.808813      0.000000      0.807083
End

xc
 lda vwn
 gga revPBE
end

Symmetry NOSYM

BeckeGrid
 quality good
End

Basis
 Type TZP
 Core none
 End

Aoresponse
 efg 1 nbo
end

qtens

eor
```

Example: NBO analysis: NMR chemical shift

Download CH4_nmrnbo.run

```
#! /bin/sh



# Example shows an NBO analysis of an NMR shielding calculation for CH4.

# Some keywords are necessary because of the NBO analysis afterwards. A 'Good'
# integration grid is used. First the scalar relativistic calculation is
# performed and the scalar relativistic localized orbitals are made. Next the
# spin-orbit coupled ADF calculation is done, and a calculation of NMR shielding
# constants is performed with an analysis in scalar relativistic localized
# orbitals.


$ADFBIN/adf <<eor

ATOMS
C 0.0000 0.0000 0.0000
H 0.6316 0.6316 0.6316
H 0.6316 -0.6316 -0.6316
```

```
H -0.6316 0.6316 -0.6316
H -0.6316 -0.6316 0.6316
END

save TAPE15
FULLFOCK
AOMAT2FILE

BASIS
type DZP
core None
END

XC
LDA SCF VWN
END

BeckeGrid
 Quality good
End

relativistic scalar zora

eor

###### end scalar, run gennbo ##########


$ADFBIN/adfnbo <<eor
write
spherical
:: read
fock
TESTJOB
eor

rm -f adfnbo.37 adfnbo.39 adfnbo.49 adfnbo.48
$ADFBIN/gennbo6 FILE47

# run adfnbo in COPY mode

$ADFBIN/adfnbo <<eor
 spherical
 fock
 copy
eor

rm -f adfnbo.37 adfnbo.39 adfnbo.49 adfnbo.48


##### end gennbo

### scalar done, redo computation with spin-orbit coupling

rm -f TAPE15 TAPE21 logfile

$ADFBIN/adf <<eor
```

```
ATOMS
C 0.0000 0.0000 0.0000
H 0.6316 0.6316 0.6316
H 0.6316 −0.6316 −0.6316
H −0.6316 0.6316 −0.6316
H −0.6316 −0.6316 0.6316
END

BASIS
type DZP
core None
END

XC
LDA SCF VWN
END

BeckeGrid
 Quality good
End

SYMMETRY nosym
SCF
 scalableso False
End

relativistic spinorbit zora

noprint sfo

SAVE TAPE10

eor

rm −f TAPE15 logfile SINFO IINFO

$ADFBIN/nmr <<eor
nmr
 atoms 2 1
 u1k best
 out iso tens
end
analysis
 print 0.01
 canonical
 nbo
 components
end
eor
```

Example: NBO analysis: NMR spin-spin coupling

Download CPL_CH3OH_NBO.run

```
#!/bin/sh
```

```
# Example shows an NBO analysis of an NMR spin-spin coupling constants
# calculation for CH3 OH.

# Some keywords are necessary because of the NBO analysis afterwards. A good
# quality integration is used. First the scalar relativistic calculation is
# performed and the scalar relativistic localized orbitals are made, and a
# calculation of NMR spin-spin coupling constants is performed with an analysis
# in scalar relativistic localized orbitals. Next the spin-orbit coupled ADF
# calculation is done, and a calculation of NMR spin-spin coupling constants is
# performed with an analysis in scalar relativistic localized orbitals.



$ADFBIN/adf <<eor

ATOMS
1 O       0.151078120000     -0.158942890000     -0.184382010000
2 H       0.762854510000      0.480823600000      0.187867830000
3 C       0.654254930000     -1.481762230000      0.026343630000
4 H       1.616760580000     -1.581906770000     -0.455670800000
5 H      -0.035909520000     -2.200223490000     -0.393433960000
6 H       0.761359880000     -1.661537720000      1.087000640000
END

save TAPE15
FULLFOCK
AOMAT2FILE

BASIS
 type DZP
 core None
END

XC
 LDA SCF VWN
END

SYMMETRY nosym

BeckeGrid
 Quality good
end

relativistic scalar zora

eor

###### end scalar, run gennbo ##########


rm -f adfnbo.kf

$ADFBIN/adfnbo <<eor
write
spherical
:: read
fock
eor
```

```
rm -f adfnbo.37 adfnbo.39 adfnbo.49 adfnbo.48
$ADFBIN/gennbo6 FILE47

# run adfnbo in COPY mode

$ADFBIN/adfnbo <<eor
 spherical
 fock
 copy
eor

# run adfnbo in READ mode: prepare locorb on TAPE21

$ADFBIN/adfnbo <<eor
 spherical
 fock
 read
eor

rm -f adfnbo.37 adfnbo.39 adfnbo.49 adfnbo.48

##### end gennbo

## end scalar and gennbo run, now do the coupling constant

rm -f TAPE15 TAPE10 TAPE13

$ADFBIN/cpl <<eor
nmrcoupling
 xalpha
 dso
 pso
 sd
 scf converge=1e-5 iterations=10
 contributions 1e19 nbo
 nuclei 3  5 6
end
eor

# redo the coupling constant without the SD term

rm -f TAPE10 TAPE15 TAPE13

$ADFBIN/cpl <<eor
nmrcoupling
 xalpha
 dso
 pso
 scf converge=1e-5 iterations=10
 contributions 1e19 nbo
 nuclei 3  5 6
end
eor


### scalar CPL done, redo computation with spin-orbit coupling
for f in TAPE*; do rm -f $f ; done
```

```
$ADFBIN/adf <<eor

ATOMS
1 O       0.151078120000    -0.158942890000    -0.184382010000
2 H       0.762854510000     0.480823600000     0.187867830000
3 C       0.654254930000    -1.481762230000     0.026343630000
4 H       1.616760580000    -1.581906770000    -0.455670800000
5 H      -0.035909520000    -2.200223490000    -0.393433960000
6 H       0.761359880000    -1.661537720000     1.087000640000
END

BASIS
 type DZP
 core None
END

XC
 LDA SCF VWN
END

SYMMETRY nosym

BeckeGrid
 Quality good
end

relativistic spinorbit zora

noprint sfo

eor

## end spinorbit run, now do the coupling constant

rm TAPE15

$ADFBIN/cpl <<eor
nmrcoupling
 xalpha
 dso
 pso
 sd
 scf converge=1e-5 iterations=10
 contributions 1e19 nbo
 nuclei 3  5 6
end
eor
```

Example: Multiple excited state gradients: H2O

Download EGO_H2O_multi.run

```
#! /bin/sh

# Basic example of TDDFT calculation with different number of singlets and triplets␣
↪(LOWEST 10 8)
```

```
# and multiple gradient calculation (SING_GRADS, TRIP_GRADS)
# Could be useful for external programs like SHARC.

$ADFBIN/adf <<eor
title H2O TDDFT Multiple Gradients

ATOMS
1 O     0.0000000    0.0000000   -0.0668805
2 H    -0.7658756    0.0000000    0.5307937
3 H     0.7658756    0.0000000    0.5307937
END

SYMMETRY C(2v)

BASIS
   Type SZ
   Core None
End

XC
   GGA BP86
End

TDA

EXCITATIONS
  Lowest 10 8
End

GRADIENT

EXCITEDGO
   SING_GRADS
    A1 1 2
    A2 1
    B1 1
   SUBEND
   TRIP_GRADS
    A1 1 2 3
    A2 1
    B1 3
    B2 1
   SUBEND
END

RELATIVISTIC scalar zora
SOPERT
End
GSCORR
PRINT SOMATRIX

NumericalQuality Good
eor

rm logfile
mv TAPE21 t21
```

Example: Calculation of overlap of primitive basis functions

Download CalcOverlapOnly.run

```sh
#! /bin/sh

# Example for calculating overlaps of primitive basis functions for almost
# coinciding nuclear coordinates, using the key CALCOVERLAPONLY. ADF will stop
# after the calculation of these overlaps. The overlap is printed in the output,
# but is also present on TAPE15 in binary format in Matrices%Smat.
# Could be useful for external programs like SHARC.


"$ADFBIN/adf" <<eor
TITLE CH2S overlap of near identical geometries

ATOMS
 C     0.1662008   -0.0114225    0.0220295
 S     1.7387976    0.003246    -0.0021948
 H    -0.3207087    0.9655128   -0.0963368
 H    -0.479404    -0.9324828   -0.0963367
 C.1   0.167746    -0.0109425    0.0203042
 S.1   1.7382268    0.002927    -0.0020154
 H.1  -0.329981     0.9728265   -0.0893183
 H.1  -0.4704213   -0.9353924   -0.088502
END

BASIS
 type DZP
 core NONE
 createoutput NONE
END

CALCOVERLAPONLY
PRINT SMAT
SAVE TAPE15

eor

rm TAPE15 TAPE21
rm logfile
```

14.7 Accuracy and Efficiency

14.7.1 BSSE, SCF convergence, Frequencies

Example: Basis Set Superposition Error (BSSE): Cr(CO)5 +CO

Download BSSE_CrCO6.run

```sh
#! /bin/sh

# A study of the Basis Set Superposition Error (BSSE) in the formation of
# Cr(CO)_6. from CO and Cr(CO)_5.
```

```
# For the BSSE calculation special chemical elements must be created to describe
# the 'ghost' atoms, which have zero nuclear charge and mass. They do have basis
# functions (and fit functions), however, and they are used to calculate the
# lowering of the energy of the system to which the ghost atoms are added, due
# to the enlargement of the basis by the ghost basis. The ghost atom has the
# same basis and fit set as the normal element but no nuclear charge and no
# frozen core. The BASIS key recognizes elements denoted with Gh.atom in the
# ATOMS key as being ghost atoms. If the basis file specifies a frozen core ADF
# will treat it as if no frozen core is present.

# The following calculations are carried out:

# 1. CO from C and O. This yields the bond energy of CO with respect to
#    the (restricted) basic atoms.
# 2. CO from the fragments CO (as calculated in 1) and the ghost atom Cr
#    and 5 Carbon and 5 Oxygen ghost atoms. The ghost atomic fragments
#    provide basis and fit functions but do not contribute charge or
#    potential to the molecule. The bond energy of this calculation is
#    the energy lowering of CO due to the additional basis functions.
#    This is the BSSE for CO.
# 3. Cr(CO)5 from Cr and 5 CO's. This yields the ('normal') bond energy
#    with respect to the given fragments.
# 4. Cr(CO)5 from Cr(CO)5 as fragment (as calculated in 3) but with the
#    CO basis functions added on the position of the 6th CO ('ghost' CO).
#    The bond energy is the BSSE for Cr(CO)5 .
# 5. Cr(CO)6 with Cr(CO)5 and CO as fragments. The bond energy is the one
#    without BSSE. This bond energy can now be corrected by the sum of
#    superposition contributions of calculations 2 and 4.



# This series of calculations is carried out with basis set DZ.

# Finally, the whole thing might be redone with basis set TZP, to show that the
# BSSE decreases with larger basis.

# The calculations for the type DZ basis are contained in the sample script
# (with input- and output files). Those for type TZP bases can be set up easily
# and may be done as an exercise.

# For the first series of calculations, with basis type DZ, the input files are
# discussed below and the relevant parts are echoed from the output files where
# the energy decomposition and the total bond energy are printed.

# For the other series, using type TZP basis sets, only a summary of the results
# is given.

# ====================
# Computational details
# ====================

# The calculations in this example all use:

# Small core DZ basis set Frozen core level for the Chromium atom: 2p, for
# Carbon and Oxygen: 1s Numerical integration precision 4.0 (in Create runs
# 10.0, the default) Default settings for model parameters such as density
# functional (key XC) and for the remaining computational settings For the BSSE
# calculations we first do the 'normal' calculations of CO and Cr(CO)5 ,
```

```
# yielding the fragment files t21.CO and t21.CrCO5. The input files for these
# calculations are not shown here.


# ===========
# BSSE for CO
# ===========

# For the CO BSSE calculation the standard CO must have been computed first. In
# the BSSE run a Cr(CO)5 ghost fragment basis set is then added to the 'normal'
# CO input. Important is the use of the BASIS key. In this case the BASIS key is
# used for the generation of the ghost atoms, it should have the same definition
# for the atoms as will be used later for the Cr(CO)5 fragment. The FRAGMENTS
# key is used for the fragment CO. The energy change (the printed 'bond energy'
# in the output) is the BSSE.


$ADFBIN/adf <<eor
title CO (normal run)

atoms
 C    0   0   1.86
 O    0   0   3.03
end

Basis
 Type DZ
 Core Small
end

symmetry C(lin)

eor

mv TAPE21 CO.t21
rm logfile


# The input file for the CO-BSSE run is:

$ADFBIN/adf <<eor
title  BSSE for CO due to Cr(CO)5 ghost
noprint sfo,frag,functions

atoms
 Gh.Cr    0         0         0
 Gh.C   -1.86       0         0
 Gh.C    1.86       0         0
 Gh.C    0         1.86       0
 Gh.C    0        -1.86       0
 Gh.C    0         0        -1.86
 Gh.O    3.03      0         0
 Gh.O   -3.03      0         0
 Gh.O    0         3.03      0
 Gh.O    0        -3.03      0
 Gh.O    0         0        -3.03
    C     0         0         1.86       f=CO
    O     0         0         3.03       f=CO
```

```
end

Basis
 Type DZ
 Core Small
end

fragments
 CO      CO.t21
end

symmetry  C(4V)

eor

mv TAPE21 CO_with_fake_CrC5O5.t21
rm logfile

# In the output we find in the Bond Energy section: The BSSE for CO is computed
# as 2.43 kcal/mol




# ================
# BSSE for Cr(CO)5
# ================

# In similar fashion the BSSE is computed for Cr(CO)_5 . In the BSSE run a ghost
# atoms C and O at the positions they will have in the Cr(CO)_6 molecule are
# added to the normal Cr(CO)_5 input:



$ADFBIN/adf <<eor
title Cr(CO)5  (normal run)
noprint sfo,frag,functions

SCF
 mixing 0.1
END

atoms
 Cr    0        0        0
 C     1.86     0        0        f=CO/1
 C    -1.86     0        0        f=CO/2
 C     0        1.86     0        f=CO/3
 C     0       -1.86     0        f=CO/4
 C     0        0       -1.86     f=CO/5
 O     3.03     0        0        f=CO/1
 O    -3.03     0        0        f=CO/2
 O     0        3.03     0        f=CO/3
 O     0       -3.03     0        f=CO/4
 O     0        0       -3.03     f=CO/5
end

Basis
 Type DZ
 Core Small
```

```
end

fragments
 CO  CO.t21
end

symmetry C(4v)

eor

mv TAPE21 CrCO5.t21
rm logfile


$ADFBIN/adf <<eor
title BSSE for Cr(CO)5 due to CO ghost
noprint sfo,frag,functions

atoms
 Cr    0        0        0         f=CrCO5
 C     1.86     0        0         f=CrCO5
 C    -1.86     0        0         f=CrCO5
 C     0        1.86     0         f=CrCO5
 C     0       -1.86     0         f=CrCO5
 C     0        0       -1.86      f=CrCO5
 O     3.03     0        0         f=CrCO5
 O    -3.03     0        0         f=CrCO5
 O     0        3.03     0         f=CrCO5
 O     0       -3.03     0         f=CrCO5
 O     0        0       -3.03      f=CrCO5
 Gh.C  0        0        1.86
 Gh.O  0        0        3.03
end

Basis
 Type DZ
 Core Small
end

fragments
 CrCO5   CrCO5.t21
end

symmetry C(4v)

eor

rm TAPE21 logfile

# The Bond Energy result yields 1.93 kcal/mole for the BSSE.


# ===============================================
# Bond Energy calculation with BSSE correction
# ===============================================

# The bonding of CO to Cr(CO)5 is computed in the normal way: from fragments CO
# and Cr(CO)5 . The obtained value for the bond energy can then simply corrected
```

```
# for the two BSSE terms, 4.36 kcal/mole together.


# ===========================
# Relevance of Core Functions
# ===========================


# The two BSSE runs can also be repeated, but now with the core
# orthogonalization functions omitted from the ghost bases. To to this one can
# not use the BASIS key, but one needs to explicitly 'create' the ghost atoms.
# This will not be done here, but only the results will be discussed. One may
# argue about whether these functions should be included in the ghost basis
# sets, but since they are very contracted around the ghost nuclei they are not
# expected to contribute significantly anyway and may then just as well be
# omitted. This has been explicitly verified by test examples. /p The Core
# Functions (the functions in the valence basis set that serve only for core-
# orthogonalization, for instance the 1S 5.40 in the Carbon basis set (see the
# $ADFHOME/atomicdata/DZ/C.1s basis set file) are removed from the Create data
# files used for the creation of the ghost atoms. This yields as BSSE values for
# CO and Cr(CO)5 respectively 2.33 and 1.87 kcal/mole (compare 2.43 and 1.93
# kcal/mole for the case with Core Functions included). The net total effect of
# including/removing the Core Functions is therefore
# (2.43-2.33)+(1.93-1.87)=0.16 kcal/mole. This is an order of magnitude smaller
# than the BSSE effect itself.


# ==================================
# BSSE and the size of the basis set
# ==================================


# BSSE effects should diminish with larger bases and disappear in the limit of a
# perfect basis. This can be studied by comparing the BSSE for basis DZ, see
# above, with the BSSE for basis TZP. The procedure is completely similar to the
# one above and yields: For the BSSE terms: 0.7 kcal/mole for CO (compare: 2.4
# kcal/mole for basis DZ), and 0.6 kcal/mole for Cr(CO)5 (1.9 for basis DZ) The
# total BSSE drops from 4.4 kcal/mole in basis DZ to 1.3 in basis TZP.


# =========
# Reference
# =========


# A systematic study with adf of the BSSE in metal-carbonyl complexes can be
# found in Rosa, A., et al., Basis Set Effects in Density Functional
# Calculations on the Metal-Ligand and Metal-Metal Bonds of Cr(CO)5-CO and
# (CO)5. Journal of Physical Chemistry, 1996, 100: p. 5690-5696.



$ADFBIN/adf <<eor
title Bond energy without BSSE for Cr(CO)6 made of Cr(CO)5 and CO
noprint sfo,frag,functions

atoms
 Cr    0       0       0        f=CrCO5
 C     1.86    0       0        f=CrCO5
 C    -1.86    0       0        f=CrCO5
 C     0       1.86    0        f=CrCO5
```

```
 C     0     −1.86    0        f=CrCO5
 C     0      0      −1.86     f=CrCO5
 O     3.03   0       0        f=CrCO5
 O    −3.03   0       0        f=CrCO5
 O     0      3.03    0        f=CrCO5
 O     0     −3.03    0        f=CrCO5
 O     0      0      −3.03     f=CrCO5
 C     0      0       1.86     f=CO
 O     0      0       3.03     f=CO
end

Basis
 Type DZ
 Core Small
end

fragments
 CrCO5   CrCO5.t21
 CO      CO.t21
end

eor

rm TAPE21 logfile
```

Example: Troubleshooting SCF convergence: Ti2O4

Download SCF_Ti2O4.run

```
#! /bin/sh

# One can run into SCF convergence problems when calculating certain types of
# systems. Some of the notorious examples are transition metal oxides and
# lanthanide compounds. Below, several approaches to solving the SCF convergence
# problem are demonstrated.


$ADFBIN/adf −n1 <<eor
Title Ti2O4 SCF aid test (default case)
Atoms
  Ti  1.730   0.000   0.000
  Ti −1.730   0.000   0.000
  O   0.000   1.224   0.000
  O   0.000  −1.224   0.000
  O   3.850   0.000   0.000
  O  −3.850   0.000   0.000
End
XC
 GGA Becke Perdew
End
Basis
 Type DZ
 Core Small
 CreateOutput none
End

SCF
```

```
 Converge 1.0e-6 1.0e-6
End

eor

rm logfile
mv TAPE21 SCF_Ti2O4_default.t21

$ADFBIN/adf -n1 <<eor
Title Ti2O4 SCF aid test (default case)
Atoms
  Ti  1.730   0.000   0.000
  Ti -1.730   0.000   0.000
  O   0.000   1.224   0.000
  O   0.000  -1.224   0.000
  O   3.850   0.000   0.000
  O  -3.850   0.000   0.000
End
XC
 GGA Becke Perdew
End
Basis
 Type DZ
 Core Small
 CreateOutput none
End

SCF
 AccelerationMethod MESA
 Converge 1.0e-6 1.0e-6
End

eor

rm logfile
mv TAPE21 SCF_Ti2O4_MultiLIST.t21

$ADFBIN/adf -n1 <<eor
Title Ti2O4 SCF aid test (LISTb by Alex Wang)
Atoms
  Ti  1.730   0.000   0.000
  Ti -1.730   0.000   0.000
  O   0.000   1.224   0.000
  O   0.000  -1.224   0.000
  O   3.850   0.000   0.000
  O  -3.850   0.000   0.000
End
XC
 GGA Becke Perdew
End
Basis
 Type DZ
 Core Small
 CreateOutput none
End

SCF
 Converge 1.0e-6 1.0e-6
```

```
 AccelerationMethod LISTb
End

eor

rm logfile
mv TAPE21 SCF_Ti2O4_LISTb.t21

$ADFBIN/adf -n1 <<eor
Title Ti2O4 SCF aid test (LISTf by Alex Wang)
Atoms
  Ti  1.730   0.000   0.000
  Ti -1.730   0.000   0.000
  O   0.000   1.224   0.000
  O   0.000  -1.224   0.000
  O   3.850   0.000   0.000
  O  -3.850   0.000   0.000
End
XC
 GGA Becke Perdew
End
Basis
 Type DZ
 Core Small
 CreateOutput none
End

SCF
 Converge 1.0e-6 1.0e-6
 AccelerationMethod LISTf
End

eor

rm logfile
mv TAPE21 SCF_Ti2O4_LISTf.t21

$ADFBIN/adf -n1 <<eor
Title Ti2O4 SCF aid test (fDIIS by Alex Wang)
Atoms
  Ti  1.730   0.000   0.000
  Ti -1.730   0.000   0.000
  O   0.000   1.224   0.000
  O   0.000  -1.224   0.000
  O   3.850   0.000   0.000
  O  -3.850   0.000   0.000
End
XC
 GGA Becke Perdew
End
Basis
 Type DZ
 Core Small
 CreateOutput none
End

SCF
 Converge 1.0e-6 1.0e-6
```

```
 AccelerationMethod fDIIS
End

eor

rm logfile
mv TAPE21 SCF_Ti2O4_fDIIS.t21



# A-DIIS method. The A-DIIS method combines the strength of the ARH and DIIS
# methods. It does not require energy evaluation so it is much cheaper than the
# ARH and Energy-DIIS methods.



$ADFBIN/adf -n1 <<eor
Title Ti2O4 SCF aid test (A-DIIS by Hu and Yang)
Atoms
  Ti  1.730   0.000   0.000
  Ti -1.730   0.000   0.000
  O   0.000   1.224   0.000
  O   0.000  -1.224   0.000
  O   3.850   0.000   0.000
  O  -3.850   0.000   0.000
End
XC
 GGA Becke Perdew
End
Basis
 Type DZ
 Core Small
 CreateOutput none
End

Occupations KeepOrbitals=100

SCF
 Iterations 300
 Mixing 0.05
 AccelerationMethod ADIIS
 Converge 1.0e-6 1.0e-6
End

eor

rm logfile
mv TAPE21 SCF_Ti2O4_adiis.t21



# This example uses the LISTi method. LISTi is very
# similar to the usual DIIS but typically it performs much better. It is also
# computationally less expensive and scales better in parallel even though DIIS
# is rarely a scaling bottleneck.



$ADFBIN/adf -n1 <<eor
Title Ti2O4 SCF aid test (LISTi)
Atoms
  Ti  1.730   0.000   0.000
```

```
  Ti -1.730   0.000   0.000
  O   0.000   1.224   0.000
  O   0.000  -1.224   0.000
  O   3.850   0.000   0.000
  O  -3.850   0.000   0.000
End
XC
 GGA Becke Perdew
End
Basis
 Type DZ
 Core Small
 CreateOutput none
End

SCF
 Iterations 300
 AccelerationMethod LISTi
End

eor

rm logfile
mv TAPE21 SCF_Ti2O4_LISTi.t21


# This example uses the Augmented Roothaan-Hall (ARH) method.
# The basic idea of this method is that the density matrix is
# optimized directly to minimize the total energy. Important: the ARH method can
# be used with SYMMETRY NOSYM only.


$ADFBIN/adf -n1 <<eor
Title Ti2O4 SCF aid test (Augmented Roothaan-Hall)
Atoms
  Ti  1.730   0.000   0.000
  Ti -1.730   0.000   0.000
  O   0.000   1.224   0.000
  O   0.000  -1.224   0.000
  O   3.850   0.000   0.000
  O  -3.850   0.000   0.000
End
XC
 GGA Becke Perdew
End
Basis
 Type DZ
 Core Small
 CreateOutput none
End

! Important: the ARH method can be used with SYMMETRY NOSYM only
Symmetry NOSYM

SCF
 Iterations 300
 Mixing 0.05
 ARH
```

```
 End
End

eor

rm logfile
mv TAPE21 SCF_Ti2O4_ARH.t21


# This is an extension to the so-called 'electron smearing' method.
# In this method, the electrons are distributed among orbitals around Fermi-
# level using a pseudo-thermal distribution function. Although the result with
# fractional occupation number has no physical sense, the method can be used to
# achieve integer occupation numbers by reducing the smearing parameter step-
# wise.
# A few notes:

# You can specify up to ten comma-delimited values after Smear= (no spaces are
# allowed). ADF will start from the first value and try to converge SCF using
# it. If it succeeds, the next value will be picked and so on.

# Because the whole process may require many cycles to converge it is important
# to set the number of SCF cycles to a large value to avoid a premature
# termination.


$ADFBIN/adf -n1 <<eor
Title Ti2O4 SCF aid test (Smear)
Atoms
  Ti  1.730   0.000   0.000
  Ti -1.730   0.000   0.000
  O   0.000   1.224   0.000
  O   0.000  -1.224   0.000
  O   3.850   0.000   0.000
  O  -3.850   0.000   0.000
End
XC
 GGA Becke Perdew
End
Basis
 Type DZ
 Core Small
 CreateOutput none
End
DEBUG DIIS SDIIS
SCF
  Iterations 300
  OLDSCF
End

Occupations Smear=0.2,0.1,0.07,0.05,0.03,0.02,0.01,0.007,0.005,0.001

eor

rm logfile
mv TAPE21 SCF_Ti2O4_smear.t21

# This example demonstrates the use of the Occupations Steep= option (see
```

```
# the User's Guide for details).

# One difference is, obviously, in the Occupations keyword. The other difference
# is more subtle. For stable convergence, it is often essential to switch off
# DIIS and set the mixing parameter to a low value. Of course, it will make
# convergence quite (sometimes very) slow. Ultimately you should get either an
# aufbau configuration or a configuration with exactly degenerate HOMO. In this
# example, the result is an aufbau solution.

# Both methods should, in principle, give the same result, which is the case in
# this example.


$ADFBIN/adf -n1 <<eor
Title Ti2O4 SCF aid test (Steep)
Atoms
  Ti  1.730   0.000   0.000
  Ti -1.730   0.000   0.000
  O   0.000   1.224   0.000
  O   0.000  -1.224   0.000
  O   3.850   0.000   0.000
  O  -3.850   0.000   0.000
End
XC
 GGA Becke Perdew
End
Basis
 Type DZ
 Core Small
 CreateOutput none
End

SCF
 Iterations 300
 Mixing 0.05
 DIIS N=0
End

Occupations Steep=0.5,0.3

eor

rm logfile
mv TAPE21 SCF_Ti2O4_steep.t21


# This example uses the so called Energy-DIIS method. Please note that similar
# to ARH and unlike the standard SCF procedure in ADF this method requires
# energy evaluation at each SCF cycle, which makes it significantly slower
# compared to energy-free SCF.


$ADFBIN/adf -n1 <<eor
Title Ti2O4 SCF aid test (Energy-DIIS)
Atoms
  Ti  1.730   0.000   0.000
  Ti -1.730   0.000   0.000
  O   0.000   1.224   0.000
```

```
  O   0.000  -1.224   0.000
  O   3.850   0.000   0.000
  O  -3.850   0.000   0.000
End
XC
 GGA Becke Perdew
End
Basis
 Type DZ
 Core Small
 CreateOutput none
End

SCF
 Iterations 300
 Mixing 0.05
 EDIIS
 Converge 1.0e-6 1.0e-6
End

eor

rm logfile
mv TAPE21 SCF_Ti2O4_ediis.t21
```

Example: Rescan frequencies: NH3

Download Freq_NH3_Scan.run

```
#! /bin/sh

# Sometimes spurious imaginary frequencies are calculated where one would expect
# a very low (nearly zero) frequency. Most frequently this happens when there is
# a barrier-free rotation of, for example, methyl groups. The SCANFREQ keyword
# allows one to rescan calculated frequencies in order to find out if they were
# calculated accurately.

# In this example analytical frequencies are calculated. Next recalculation of
# certain NH3 frequencies are performed by scanning along normal modes from a
# restart file. In this calculation the frequencies are calculated numerically
# with finite displacements using symmetry.

$ADFBIN/adf <<eor
title NH3 analytic frequencies

atoms
  N              0.0000   0.0000   0.0000
  H              0.4729   0.8190   0.3821
  H             -0.9457   0.0000   0.3821
  H              0.4729  -0.8190   0.3821
end

Basis
 Type TZP
 Core Small
End
```

```
AnalyticalFreq
End

BeckeGrid
 Quality verygood
End

eor

mv TAPE21 NH3_anl.t21

$ADFBIN/adf <<eor
title Re-calculate NH3 frequencies by scanning along normal modes from a restart file

atoms
   N              0.0000    0.0000    0.0000
   H              0.4729    0.8190    0.3821
   H             -0.9457    0.0000    0.3821
   H              0.4729   -0.8190    0.3821
end

Fragments
  N t21.N
  H t21.H
End

ScanFreq 0 4000
Restart
   File NH3_anl.t21
End

BeckeGrid
 Quality verygood
End

eor

mv TAPE21 NH3_symm2.t21
```

14.7.2 Speed

Example: Multiresolution

Download Multiresolution_H2O.run

```
#! /bin/sh

# This example demonstrates how to use different levels of numerical precision
# for different regions, with the aim of increasing computational efficiency.

# Let us assume that we are interested in having an accurate description only
# for a subregion of a large chemical system (in this simple example, the
# central water molecule). The system can be divided into sub-regions and
# different levels of numerical accuracy can be used for each of these sub-
# regions.
```

```
# In this example we will tweak:
# - the basis set (Basis)
# - the numerical integration (BeckeGrid)
# - the density fitting for Coulomb potential (ZlmFit)
# - the fit-set used in the Hartree-Fock Resolution of identity (RIHartreeFock)

# Note: For the atoms that have not been explicitly defined in the
# AtomDepQuality sub-blocks, the quality defined in NumericalQuality will be
# used (Normal, in this example).


"$ADFBIN/adf" <<eor
  Atoms
      1   O.Accurate   0.00000000      0.00000000      0.00000000
      2   H.Accurate   0.40399229     -0.65578342      0.63241539
      3   H.Accurate   0.81410032      0.55624569     -0.23129097
      4   O.Near      -3.02535626     -0.08473104     -0.47678489
      5   H.Near      -2.56531481      0.62644005      0.07759386
      6   H.Near      -2.25289644     -0.61700366     -0.80790649
      7   O.Near       2.95394790     -0.54939973     -0.38206034
      8   H.Near       3.91427727     -0.21304908     -0.44738291
      9   H.Near       2.87780992     -1.13241278     -1.20853726
     10   O.Far       -5.95425742     -0.56764616     -0.02016682
     11   H.Far       -5.26308282     -0.46969096      0.69255963
     12   H.Far       -5.42117992     -0.54361203     -0.86443121
     13   O.Far        6.25171470     -0.62004899     -0.03702467
     14   H.Far        6.16508647     -1.38696453      0.58541903
     15   H.Far        7.09161199     -0.16700550      0.23679419
  End

  ! ================================================================
  ! Set the NumerialQuality to be used for the atoms that are not
  ! explicitely defined in AtomDepQuality
  ! ================================================================

  NumericalQuality Normal

  XC
     LibXC B3LYP
  End

  ! ==========================================================
  ! Set different basis sets for atoms in different regions:
  ! ==========================================================

  Basis
     O.Accurate TZP/O
     H.Accurate TZP/H
     O.Near      DZ/O
     H.Near      DZ/H
     O.Far       SZ/O
     H.Far       SZ/H
  End

  ! =====================
  ! Numerical integration:
  ! =====================
```

```
   BeckeGrid
      AtomDepQuality
          1   Good
          2   Good
          3   Good
          10  Basic
          11  Basic
          12  Basic
          13  Basic
          14  Basic
          15  Basic
      SubEnd
   End

   ! =====================================
   ! Density fitting for Coulomb potential:
   ! =====================================

   ZlmFit
      AtomDepQuality
          1   Good
          2   Good
          3   Good
          10  Basic
          11  Basic
          12  Basic
          13  Basic
          14  Basic
          15  Basic
      SubEnd
   End

   ! =============================================================
   ! Hartree-Fock Resolution of identity (for hybrid functionals)
   ! =============================================================

   RIHartreeFock
      AtomDepQuality
          1   Good
          2   Good
          3   Good
          10  Basic
          11  Basic
          12  Basic
          13  Basic
          14  Basic
          15  Basic
      SubEnd
   End
eor
```

14.8 Scripting

14.8.1 Prepare an ADF job and generate a report

Example: Single point for multiple xyz files: Bakerset

Download BakersetSP.run

```sh
#! /bin/sh


# In this example you will find how to use adfprep to run a particular job (a
# single point calculation in this case) for all molecules in the Baker set. The
# molecules are simply xyz files and contain no ADF specific information.
# adfreport is used to collect the resulting bonding energies.


rm -f runset
for f in $ADFHOME/examples/adf/BakersetSP/Bakerset/*.xyz
do
    "$ADFBIN/adfprep" -t SP -i 2.5 -b DZ -c Large -m "$f" -j `basename $f .xyz`>>␣
↪runset
done

chmod +x runset
./runset

echo Results
ls -t -1 *.t21 | while read f
do
    "$ADFBIN/adfreport" "$f" BondingEnergy
done
echo Ready
```

Example: Basis set and integration accuracy convergence test: Methane

Download ConvergenceTestCH4.run

```sh
#! /bin/sh


# In this example you will find how to use adfprep to test convergence of the
# bonding energy with respect to basis set and integration accuracy. adfreport
# is used to collect the resulting bonding energies.


rm -f runset
for b in SZ DZ DZP TZP TZ2P QZ4P
do
    "$ADFBIN/adfprep" -t "$ADFHOME/examples/adf/ConvergenceTestCH4/methane.adf" -b $b␣
↪-j methane.$b>> runset
done

chmod +x runset
```

```
./runset

echo Results
echo Basis set convergence of Bonding Energy, SZ DZ DZP TZP TZ2P QZ4P
for b in SZ DZ DZP TZP TZ2P QZ4P
do
    "$ADFBIN/adfreport" "methane.$b.t21" BondingEnergy
done

rm -f runset
for i in 2 3 4 5
do
    "$ADFBIN/adfprep" -t "$ADFHOME/examples/adf/ConvergenceTestCH4/methane.adf" -b
→DZP -i $i -j methane.$i>> runset
done
"$ADFBIN/adfprep" -t "$ADFHOME/examples/adf/ConvergenceTestCH4/methane.adf" -b DZP -i
→Basic -j methane.bb>> runset
"$ADFBIN/adfprep" -t "$ADFHOME/examples/adf/ConvergenceTestCH4/methane.adf" -b DZP -i
→Normal -j methane.bn>> runset
"$ADFBIN/adfprep" -t "$ADFHOME/examples/adf/ConvergenceTestCH4/methane.adf" -b DZP -i
→Good -j methane.bg>> runset
"$ADFBIN/adfprep" -t "$ADFHOME/examples/adf/ConvergenceTestCH4/methane.adf" -b DZP -i
→VeryGood -j methane.bv>> runset
"$ADFBIN/adfprep" -t "$ADFHOME/examples/adf/ConvergenceTestCH4/methane.adf" -b DZP -i
→Excellent -j methane.be>> runset

chmod +x runset
./runset

echo Integration convergence of Bonding Energy, 2 3 4 5
for i in 2 3 4 5
do
    "$ADFBIN/adfreport" "methane.$i.t21" BondingEnergy
done

echo Integration Becke convergence of Bonding Energy, Basic, Normal, Good, VeryGood,
→Excellent
"$ADFBIN/adfreport" "methane.bb.t21" BondingEnergy
"$ADFBIN/adfreport" "methane.bn.t21" BondingEnergy
"$ADFBIN/adfreport" "methane.bg.t21" BondingEnergy
"$ADFBIN/adfreport" "methane.bv.t21" BondingEnergy
"$ADFBIN/adfreport" "methane.be.t21" BondingEnergy

echo Ready
```

Example: adfprep: Replace atom with ligand

Download LoopAtomsLigands.run

```
#! /bin/sh


# generate a run file for methane, with 1 H replaced by some ligand (run adfprep
# -liststructures to get full list of available structures) note the -a flag
# makes adfprep produce an .adf file as well, which can be opened by ADFinput to
# check your system
```

```
for ligand in CN CO CO3 NC NH2 NH2CH3 NH3 OC OCH3 OH PH3 Pyridine; do
    "$ADFBIN/adfprep" -t GO -m "$ADFHOME/examples/adf/LoopAtomsLigands/methane.xyz" -
→structure "2 Ligands/$ligand.adf" -a methane.$ligand.adf > methane.$ligand.run
done


# similar, replace the C atom by other atoms

for atom in C Si Ge Sn Pb; do
    "$ADFBIN/adfprep" -t GO -m "$ADFHOME/examples/adf/LoopAtomsLigands/methane.xyz" -
→atomtype "1 $atom" -a methane.$atom.adf > methane.$atom.run
done


# following part of the script is just to check that the run files have been
# generated correctly in the test enviornment

for ligand in CN CO CO3 NC NH2 NH2CH3 NH3 OC OCH3 OH PH3 Pyridine; do
    cat methane.$ligand.run >> all
done
for atom in C Si Ge Sn Pb; do
    cat methane.$atom.run >> all
done

DIFF=`diff -w -i -q all "$ADFHOME/examples/adf/LoopAtomsLigands/ref-all"`

echo Diff result: $DIFF

if test -z "$DIFF"; then
    echo "Ligand run files are correct 0.0"
else
    echo "Ligand run files differ 1.0"
fi

echo Done
```

14.9 List of Examples

- *VROA_RESO* (page 631)
- *XCDFT_FDE_ET_ethylene* (page 783)
- *XCFUN_FULLKERNEL_TDDFT_H2O* (page 645)
- *XCFUN_TDDFT_H2O* (page 646)
- *WaterDimer* (page 606)

REQUIRED CITATIONS

When you publish results in the scientific literature that were obtained with programs of the ADF package, you are required to include references to the program package with the appropriate release number, and a few key publications.

In addition to these general references, references to special features are mandatory, in case you have used them.

15.1 General References

For calculations with the molecular ADF program, version 2019: G.te Velde, F.M. Bickelhaupt, E.J. Baerends, C. Fonseca Guerra, S.J.A. van Gisbergen, J.G. Snijders and T. Ziegler, *Chemistry with ADF*, Journal of Computational Chemistry 22, 931 (2001) (https://doi.org/10.1002/jcc.1056) (`bibtex`)

C.Fonseca Guerra, J.G. Snijders, G. te Velde and E.J. Baerends, *Towards an order-N DFT method*, Theoretical Chemistry Accounts 99, 391 (1998) (https://doi.org/10.1007/s002140050353)

ADF2019, SCM, Theoretical Chemistry, Vrije Universiteit, Amsterdam, The Netherlands, http://www.scm.com. Optionally, you may add the following list of authors and contributors: E.J. Baerends, T. Ziegler, A.J. Atkins, J. Autschbach, O. Baseggio, D. Bashford, A. Bérces, F.M. Bickelhaupt, C. Bo, P.M. Boerrigter, L. Cavallo, C. Daul, D.P. Chong, D.V. Chulhai, L. Deng, R.M. Dickson, J.M. Dieterich, D.E. Ellis, M. van Faassen, L. Fan, T.H. Fischer, C. Fonseca Guerra, M. Franchini, A. Ghysels, A. Giammona, S.J.A. van Gisbergen, A. Goez, A.W. Götz, J.A. Groeneveld, O.V. Gritsenko, M. Grüning, S. Gusarov, F.E. Harris, P. van den Hoek, Z. Hu, C.R. Jacob, H. Jacobsen, L. Jensen, L. Joubert, J.W. Kaminski, G. van Kessel, C. König, F. Kootstra, A. Kovalenko, M.V. Krykunov, E. van Lenthe, D.A. McCormack, A. Michalak, M. Mitoraj, S.M. Morton, J. Neugebauer, V.P. Nicu, L. Noodleman, V.P. Osinga, S. Patchkovskii, M. Pavanello, C.A. Peeples, P.H.T. Philipsen, D. Post, C.C. Pye, H. Ramanantoanina, P. Ramos, W. Ravenek, J.I. Rodríguez, P. Ros, R. Rüger, P.R.T. Schipper, D. Schlüns, H. van Schoot, G. Schreckenbach, J.S. Seldenthuis, M. Seth, J.G. Snijders, M. Solà, M. Stener, M. Swart, D. Swerhone, V. Tognetti, G. te Velde, P. Vernooijs, L. Versluis, L. Visscher, O. Visser, F. Wang, T.A. Wesolowski, E.M. van Wezenbeek, G. Wiesenekker, S.K. Wolff, T.K. Woo, A.L. Yakovlev (`bibtex`)

Note: if you have used a modified (by yourself, for instance) version of the code, you should mention in the citation that a modified version has been used.

15.2 Feature References

When you have used special features, you should include one (or more, as the case may be) lead reference(s) to the implementation. Additional references to related publications are suggested.

15.2.1 Coordinates, basis sets, fragments

Basis Sets E. van Lenthe and E.J. Baerends, *Optimized Slater-type basis sets for the elements 1-118*, Journal of Computational Chemistry 24, 1142 (2003) (https://doi.org/10.1002/jcc.10255)

Spherical Gaussian nuclear charge distribution model J. Autschbach, *Magnitude of Finite-Nucleus-Size Effects in Relativistic Density Functional Computations of Indirect NMR Nuclear Spin-Spin Coupling Constants*, ChemPhysChem 10, 2274 (2009) (https://doi.org/10.1002/cphc.200900271)

15.2.2 Geometry optimizations, transition states, and reaction paths

Transition State search L. Versluis and T. Ziegler, *The determination of Molecular Structure by Density Functional Theory*, Journal of Chemical Physics 88, 322 (1988) (https://doi.org/10.1063/1.454603)

L. Fan and T. Ziegler, *Nonlocal density functional theory as a practical tool in calculations on transition states and activation energies*, Journal of the American Chemical Society 114, 10890 (1992) (https://doi.org/10.1021/ja00053a027)

IRC L. Deng, T. Ziegler and L. Fan, *A combined density functional and intrinsic reaction coordinate study on the ground state energy surface of H2 CO*, Journal of Chemical Physics 99, 3823 (1993) (https://doi.org/10.1063/1.466129)

L. Deng and T. Ziegler, *The determination of Intrinsic Reaction Coordinates by density functional theory*, International Journal of Quantum Chemistry 52, 731 (1994) (https://doi.org/10.1002/qua.560520406)

Nudged Elastic Band G. Henkelman, B.P. Uberuaga and H. Jónsson, *A climbing image nudged elastic band method for finding saddle points and minimum energy paths*, Journal of Chemical Physics 113, 9901 (2000) (https://doi.org/10.1063/1.1329672)

15.2.3 Model Hamiltonians

Range Separated Functionals M. Seth and T. Ziegler, *Range-Separated Exchange Functionals with Slater-Type Functions*, Journal of Chemical Theory and Computation 8, 901 (2012) (https://doi.org/10.1021/ct300006h)

OEP M. Krykunov and T. Ziegler, *On the use of the exact exchange optimized effective potential method for static response properties*, International Journal of Quantum Chemistry 109, 3246 (2009) (https://doi.org/10.1002/qua.21937)

15.2.4 Relativistic Effects

ZORA

Lead references E. van Lenthe, E.J. Baerends and J.G. Snijders, *Relativistic regular two-component Hamiltonians*, Journal of Chemical Physics 99, 4597 (1993) (https://doi.org/10.1063/1.466059)

E. van Lenthe, E.J. Baerends and J.G. Snijders, *Relativistic total energy using regular approximations*, Journal of Chemical Physics 101, 9783 (1994) (https://doi.org/10.1063/1.467943)

E. van Lenthe, A.E. Ehlers and E.J. Baerends, *Geometry optimization in the Zero Order Regular Approximation for relativistic effects*, Journal of Chemical Physics 110, 8943 (1999) (https://doi.org/10.1063/1.478813)

Suggested related references E. van Lenthe, J.G. Snijders and E.J. Baerends, *The zero-order regular approximation for relativistic effects: The effect of spin.orbit coupling in closed shell molecules*, Journal of Chemical Physics 105, 6505 (1996) (https://doi.org/10.1063/1.472460)

E. van Lenthe, R. van Leeuwen, E.J. Baerends and J.G. Snijders, *Relativistic regular two-component Hamiltonians*, International Journal of Quantum Chemistry 57, 281 (1996) (https://doi.org/10.1002/(SICI)1097-461X(1996)57:3%3C281::AID-QUA2%3E3.0.CO;2-U)

Pauli J.G. Snijders, E.J. Baerends and P. Ros, *A perturbation theory approach to relativistic calculations. II. Molecules*, Molecular Physics 38, 1909 (1979) (https://doi.org/10.1080/00268977900102941)

P.M. Boerrigter, E.J. Baerends and J.G. Snijders, *A relativistic LCAO Hartree-Fock-Slater investigation of the electronic structure of the actinocenes M(COT)2 , M=Th, Pa, U, Np and Pu*, Chemical Physics 122, 357 (1988) (https://doi.org/10.1016/0301-0104(88)80018-1)

T. Ziegler, V. Tschinke, E.J. Baerends, J.G. Snijders and W. Ravenek, *Calculation of bond energies in compounds of heavy elements by a quasi-relativistic approach*, Journal of Physical Chemistry 93, 3050 (1989) (https://doi.org/10.1021/j100345a036)

15.2.5 Solvents and other environments

COSMO: Conductor like Screening Model C.C. Pye and T. Ziegler, *An implementation of the conductor-like screening model of solvation within the Amsterdam density functional package*, Theoretical Chemistry Accounts 101, 396 (1999) (https://doi.org/10.1007/s002140050457)

SM12: Solvation Model 12 C.A. Peeples and G. Schreckenbach, *Implementation of the SM12 Solvation Model into ADF and Comparison with COSMO*, Journal of Chemical Theory and Computation 12, 4033 (2016) (https://doi.org/10.1021/acs.jctc.6b00410)

QM/MM: Quantum mechanical and Molecular Mechanics model

Lead T. K. Woo, L. Cavallo and T. Ziegler, *Implementation of the IMOMM methodology for performing combined QM/MM molecular dynamics simulations and frequency calculations*, Theoretical Chemistry Accounts 100, 307 (1998) (https://doi.org/10.1007/s002140050391)

Suggested T. K. Woo, S. Patchkovskii, and T. Ziegler, *Atomic Scale Modeling of Polymerization Catalysts*, Computing in Science & Engineering, 2, 28-37 (2000) (https://doi.org/10.1109/5992.881705)

For AddRemove model M. Swart, *AddRemove: A new link model for use in QM/MM studies*, International Journal of Quantum Chemistry 91, 177 (2003) (https://doi.org/10.1002/qua.10463)

FDE: Frozen Density Embedding T.A. Wesolowski and A. Warshel, *Frozen Density Functional Approach for ab-initio Calculations of Solvated Molecules*, Journal of Physical Chemistry 97, 8050 (1993) (https://doi.org/10.1021/j100132a040)

J. Neugebauer, C.R. Jacob, T.A. Wesolowski and E.J. Baerends, *An Explicit Quantum Chemical Method for Modeling Large Solvation Shells Applied to Aminocoumarin C151* Journal of Physical Chemistry A 109, 7805 (2005) (https://doi.org/10.1021/jp0528764)

C.R. Jacob, J. Neugebauer and L. Visscher, *A flexible implementation of frozen-density embedding for use in multilevel simulations*, Journal of Computational Chemistry 29, 1011 (2008) (https://doi.org/10.1002/jcc.20861)

DIM/QM: Discrete Interaction Model/Quantum Mechanics J.L. Payton, S.M. Morton, Justin E. Moore, and Lasse Jensen, *A discrete interaction model/quantum mechanical method for simulating surface-enhanced Raman spectroscopy*, Journal of Chemical Physics 136, 214103 (2012) (https://doi.org/10.1063/1.4722755)

DRF: Discrete Solvent Reaction Field model L. Jensen, P.T. van Duijnen and J.G. Snijders, *A discrete solvent reaction field model within density functional theory* Journal of Chemical Physics 118, 514 (2003) (https://doi.org/10.1063/1.1527010)

SCRF: Self-Consistent Reaction Field J.L. Chen, L. Noodleman, D.A. Case and D. Bashford, *Incorporating solvation effects into density functional electronic structure calculations*, Journal of Physical Chemistry 98, 11059 (1994) (https://doi.org/10.1021/j100094a013)

VSCRF (vertical excitation self-consistent reaction field) T. Liu, W.-G Han, F. Himo, G.M. Ullmann, D. Bashford, A. Toutchkine, K.M. Hahn, and L. Noodleman, *Density Functional Vertical Self-Consistent Reaction Field Theory for Solvatochromism Studies of Solvent-Sensitive Dyes*, Journal of Physical Chemistry A 108, 3545 (2004) (https://doi.org/10.1021/jp031062p)

W.-G. Han, T. Liu, F. Himo, A. Toutchkine, D. Bashford, K.M. Hahn, L. Noodleman, *A Theoretical Study of the UV/Visible Absorption and Emission Solvatochromic Properties of Solvent-Sensitive Dyes*, ChemPhysChem 4, 1084 (2003) (https://doi.org/10.1002/cphc.200300801)

3D-RISM: Three-Dimensional Reference Interaction Site Model

Lead S. Gusarov, T. Ziegler, and A. Kovalenko, *Self-Consistent Combination of the Three-Dimensional RISM Theory of Molecular Solvation with Analytical Gradients and the Amsterdam Density Functional Package*, Journal of Physical Chemistry A 110, 6083 (2006) (https://doi.org/10.1021/jp054344t)

Suggested A. Kovalenko and F. Hirata, *Self-consistent description of a metal-water interface by the Kohn-Sham density functional theory and the three-dimensional reference interaction site model*, Journal of Chemical Physics 110, 10095 (1999) (https://doi.org/10.1063/1.478883)

A. Kovalenko, *Three-dimensional RISM theory for molecular liquids and solid-liquid interfaces*, In Molecular Theory of Solvation; Hirata, Fumio, Ed.; Understanding Chemical Reactivity (series); Mezey, Paul G., Series Ed.; Kluwer Acadamic Publishers: Dordrecht, The Netherlands, 2003; Vol. 24, pp 169-275.

MM Dispersion: Molecular Mechanics dispersion-corrected functionals S. Grimme, *Semiempirical GGA-Type Density Functional Constructed with a Long-Range Dispersion Correction*, Journal of Computational Chemistry 27, 1787 (2006) (https://doi.org/10.1002/jcc.20495)

old implementation S. Grimme, *Accurate description of van der Waals complexes by density functional theory including empirical corrections*, Journal of Computational Chemistry 25, 1463 (2004) (https://doi.org/10.1002/jcc.20078)

J.-M. Ducéré and L. Cavallo, *Parametrization of an Empirical Correction Term to Density Functional Theory for an Accurate Description of pi-Stacking Interactions in Nucleic Acids*, Journal of Physical Chemistry B 111, 13124 (2007) (https://doi.org/10.1021/jp072317s) contact: J.M. Ducere, L. Cavallo, University of Salerno, Italy

15.2.6 Frequencies, IR Intensities, Raman, VCD

Numerical Differentiation of Gradients L. Fan and T. Ziegler, *Application of density functional theory to infrared absorption intensity calculations on main group molecules*, Journal of Chemical Physics 96, 9005 (1992) (https://doi.org/10.1063/1.462258)

L. Fan and T. Ziegler, *Application of density functional theory to infrared absorption intensity calculations on transition-metal carbonyls*, Journal of Physical Chemistry 96, 6937 (1992) (https://doi.org/10.1021/j100196a016)

Analytical Second Derivatives A. Bérces, R. M. Dickson, L. Fan, H. Jacobsen, D. Swerhone and T. Ziegler, *An implementation of the coupled perturbed Kohn-Sham equations: perturbation due to nuclear displacements*, Computer Physics Communications 100, 247 (1997) (https://doi.org/10.1016/S0010-4655(96)00120-8)

H. Jacobsen, A. Bérces, D. Swerhone and T. Ziegler, *Analytic second derivatives of molecular energies: a density functional implementation*, Computer Physics Communications 100, 263 (1997) (https://doi.org/10.1016/S0010-4655(96)00119-1)

S. K. Wolff, *Analytical second derivatives in the Amsterdam density functional package*, International Journal of Quantum Chemistry 104, 645 (2005) (https://doi.org/10.1002/qua.20653)

Mobile Block Hessian (MBH)

Lead A. Ghysels, D. Van Neck, V. Van Speybroeck, T. Verstraelen and M. Waroquier, *Vibrational Modes in partially optimized molecular systems* Journal of Chemical Physics**126, 224102 (2007) (https://doi.org/10.1063/1.2737444)

Suggested A. Ghysels, D. Van Neck and M. Waroquier, *Cartesian formulation of the Mobile Block Hessian Approach to vibrational analysis in partially optimized systems* Journal of Chemical Physics**127, 164108 (2007) (https://doi.org/10.1063/1.2789429)

Raman scattering S.J.A. van Gisbergen, J.G. Snijders and E.J. Baerends, *Application of time-dependent density functional response theory to Raman scattering*, Chemical Physics Letters 259, 599 (1996) (https://doi.org/10.1016/0009-2614(96)00858-5)

S.J.A. van Gisbergen, J.G. Snijders and E.J. Baerends, *Implementation of time-dependent density functional response equations*, Computer Physics Communications 118, 119 (1999) (https://doi.org/10.1016/S0010-4655(99)00187-3)

Resonance Raman: excited-state finite lifetime L. Jensen, L. Zhao, J. Autschbach and G.C. Schatz, *Theory and method for calculating resonance Raman scattering from resonance polarizability derivatives*, Journal of Chemical Physics 123, 174110 (2005) (https://doi.org/10.1063/1.2046670)

Resonance Raman: excited-state gradient J. Neugebauer, E.J. Baerends, E. Efremov, F. Ariese and C. Gooijer, *Combined Theoretical and Experimental Deep-UV Resonance Raman Studies of Substituted Pyrenes*, Journal of Physical Chemistry A 109, 2100 (2005) (https://doi.org/10.1021/jp045360d)

VROA: (Resonance) vibrational Raman optical activity L. Jensen, J. Autschbach, M. Krykunov, and G.C. Schatz, *Resonance vibrational Raman optical activity: A time-dependent density functional theory approach*, Journal of Chemical Physics 127, 134101 (2007) (https://doi.org/10.1063/1.2768533)

Vibrational Circular Dichroism (VCD) V.P. Nicu J. Neugebauer S.K. Wolff and E.J. Baerends, *A vibrational circular dichroism implementation within a Slater-type-orbital based density functional framework and its application to hexa- and hepta-helicenes*, Theoretical Chemical Accounts 119, 245 (2008) (https://doi.org/10.1007/s00214-006-0234-x)

VCD analysis: VCDtools V.P. Nicu, J. Neugebauer and E.J. Baerends, *Effects of Complex Formation on Vibrational Circular Dichroism Spectra*, Journal of Physical Chemistry A 112, 6978 (2008) (https://doi.org/10.1021/jp710201q)

M.A.J. Koenis, W.J. Buma and V.P. Nicu, *Analysis of VCD spectra of peptides: a generalised coupled oscillator approach using VCDtools*, manuscript in preparation, (2018)

V.P. Nicu, *Revisiting an old concept: the coupled oscillator model for VCD. Part 1: the generalised coupled oscillator mechanism and its intrinsic connection to the strength of VCD signals*, Physical Chemistry Chemical Physics 18, 21202 (2016) (https://doi.org/10.1039/C6CP01282E).

Franck-Condon factors J.S. Seldenthuis, H.S.J. van der Zant, M.A. Ratner and J.M. Thijssen, *Vibrational Excitations in Weakly Coupled Single-Molecule Junctions: A Computational Analysis*, ACS Nano 2, 1445 (2008) (https://doi.org/10.1021/nn800170h)

15.2.7 Time-Dependent DFT

For all Time-Dependent DFT features (Excitation Energies, (Hyper) Polarizabilities, Dispersion Coefficients, Raman Scattering S.J.A. van Gisbergen, J.G. Snijders and E.J. Baerends, *Implementation of time-dependent density functional response equations*, Computer Physics Communications 118, 119 (1999) (https://doi.org/10.1016/S0010-4655(99)00187-3)

Excitation Energies and Oscillator Strengths

Lead reference S.J.A. van Gisbergen, J.G. Snijders and E.J. Baerends, *Implementation of time-dependent density functional response equations*, Computer Physics Communications 118, 119 (1999) (https://doi.org/10.1016/S0010-4655(99)00187-3)

Suggested (when ZORA relativistic results are used) A. Rosa, E.J. Baerends, S.J.A. van Gisbergen, E. van Lenthe, J.A. Groeneveld and J. G. Snijders, *Article Electronic Spectra of M(CO)6 (M = Cr, Mo, W) Revisited by a Relativistic TDDFT Approach*, Journal of the American Chemical Society 121, 10356 (1999) (https://doi.org/10.1021/ja990747t)

Open Shell ground state F. Wang and T. Ziegler, Mol. Phys.102, 2585 (2004)

Spin-flip transitions F. Wang and T. Ziegler, *Time-dependent density functional theory based on a non-collinear formulation of the exchange-correlation potential*, Journal of Chemical Physics 121, 12191 (2004) (https://doi.org/10.1063/1.1821494)

F. Wang and T. Ziegler, *The performance of time-dependent density functional theory based on a noncollinear exchange-correlation potential in the calculations of excitation energies*, Journal of Chemical Physics 122, 74109 (2005) (https://doi.org/10.1063/1.1844299)

Core excitations M. Stener, G. Fronzoni and M. de Simone, *Time dependent density functional theory of core electrons excitations*, Chemical Physics Letters 373, 115 (2003) (https://doi.org/10.1016/S0009-2614(03)00543-8)

Quadrupole intensities S. Bernadotte, A.J. Atkins, Ch.R. Jacob, *Origin-independent calculation of quadrupole intensities in X-ray spectroscopy*, Journal of Chemical Physics 137, 204106 (2012) (https://doi.org/10.1063/1.4766359)

XES: X-ray emission spectroscopy A.J. Atkins, M. Bauer, and Ch.R. Jacob, *The chemical sensitivity of X-ray spectroscopy: high energy resolution XANES versus X-ray emission spectroscopy of substituted ferrocenes*, Physical Chemistry Chemical Physics 15, 8095 (2013) (https://doi.org/10.1039/C3CP50999K)

Excitations including spin-orbit coupling F. Wang, T. Ziegler, E. van Lenthe, S.J.A. vand Gisbergen and E.J. Baerends, *The calculation of excitation energies based on the relativistic two-component zeroth-order regular approximation and time-dependent density-functional with full use of symmetry*, Journal of Chemical Physics 122, 204103 (2005) (https://doi.org/10.1063/1.1899143)

Perturbative approach to include spin-orbit coupling F. Wang and T. Ziegler, *A simplified relativistic time-dependent density-functional theory formalism for the calculations of excitation energies including spin-orbit coupling effect*, Journal of Chemical Physics 123, 154102 (2005) (https://doi.org/10.1063/1.2061187)

CV(n)-DFT: Constricted Variational DFT J. Cullen, M. Krykunov, and T. Ziegler, *The formulation of a self-consistent constricted variational density functional theory for the description of excited states*, Chemical Physics 391, 11 (2011) (https://doi.org/10.1016/j.chemphys.2011.05.021)

M. Krykunov and T. Ziegler, *Self-consistent Formulation of Constricted Variational Density Functional Theory with Orbital Relaxation. Implementation and Applications*, Journal of Chemical Theory and Computation 9, 2761 (2013) (https://doi.org/10.1021/ct300891k)

CV(n)-DFT: triplet states Y.C. Park, F. Senn, M. Krykunov, and T. Ziegler, *Self-Consistent Constricted Variational Theory RSCF-CV(∞)-DFT and Its Restrictions To Obtain a Numerically Stable Δ SCF-DFT-like Method: Theory and Calculations for Triplet States*, Journal of Chemical Theory and Computation 12, 5438 (2016) (https://doi.org/10.1021/acs.jctc.6b00333)

TD-DFT+TB: Tight-Binding approximations to time-dependent Density Functional Theory R. Rüger, E. van Lenthe, T. Heine, L. Visscher, *Tight-Binding Approximations to Time-Dependent Density Functional Theory - a fast approach for the calculation of electronically excited states*, Journal of Chemical Physics 144, 184103 (2016) (https://doi.org/10.1063/1.4948647)

Excited state gradients M. Seth, G. Mazur, and T. Ziegler, *Time-dependent density functional theory gradients in the Amsterdam density functional package: geometry optimizations of spin-flip excitations*, Theoretical Chemistry Accounts 129, 331 (2011) (https://doi.org/10.1007/s00214-010-0819-2)

Polarizabilities S.J.A. van Gisbergen, J.G. Snijders and E.J. Baerends, *A Density Functional Theory study of frequency-dependent polarizabilities and van der Waals dispersion coefficients for polyatomic molecules*, Journal of Chemical Physics 103, 9347 (1995) (https://doi.org/10.1063/1.469994)

Polarizabilities including spin-orbit coupling A. Devarajan, A. Gaenko, and J. Autschbach, *Two-component relativistic density functional method for computing nonsingular complex linear response of molecules based on the zeroth order regular approximation*, Journal of Chemical Physics 130, 194102 (2009) (https://doi.org/10.1063/1.3123765)

Suggested V.P. Osinga, S.J.A. van Gisbergen, J.G. Snijders and E.J. Baerends, *Density functional results for isotropic and anisotropic multipole polarizabilities and C6 , C7 , and C8 Van der Waals dispersion coefficients for molecules*, Journal of Chemical Physics 106, 5091 (1997) (https://doi.org/10.1063/1.473555)

Damped complex polarizabilities, POLTDDFT scheme O. Baseggio, G. Fronzoni, and M. Stener, *A new time dependent density functional algorithm for large systems and plasmons in metal clusters*, Journal of Chemical Physics 143, 024106 (2015) (https://doi.org/10.1063/1.4923368)

Hyperpolarizabilities

Hyperpolarizabilities with RESPONSE S.J.A. van Gisbergen, J.G. Snijders and E.J. Baerends, *Calculating frequency-dependent hyperpolarizabilities using time-dependent density functional theory*, Journal of Chemical Physics 109, 10644 (1998) (https://doi.org/10.1063/1.477762)

Suggested: S.J.A. van Gisbergen, J.G. Snijders, and E.J. Baerends, *Time-dependent Density Functional Results for the Dynamic Hyperpolarizability of C60*, Physical Review Letters 78, 3097 (1997) (https://doi.org/10.1103/PhysRevLett.78.3097)

First hyperpolarizabilities with AORESPONSE Z. Hu, J. Autschbach, and L. Jensen, *Simulation of resonance hyper-Rayleigh scattering of molecules and metal clusters using a time-dependent density functional theory approach*, Journal of Chemical Physics 141, 124305 (2014) (https://doi.org/10.1063/1.4895971)

Second hyperpolarizabilities with AORESPONSE Z. Hu, J. Autschbach, and L. Jensen, *Simulating Third-Order Nonlinear Optical Properties Using Damped Cubic Response Theory within Time-Dependent Density Functional Theory*, Journal of Chemical Theory and Computation 12, 1294 (2016) (https://doi.org/10.1021/acs.jctc.5b01060)

Dispersion Coefficients

Lead V.P. Osinga, S.J.A. van Gisbergen, J.G. Snijders and E.J. Baerends, *Density functional results for isotropic and anisotropic multipole polarizabilities and C6 , C7 , and C8 Van der Waals dispersion coefficients for molecules*, Journal of Chemical Physics 106, 5091 (1997) (https://doi.org/10.1063/1.473555)

Suggested S.J.A. van Gisbergen, J.G. Snijders and E.J. Baerends, *A Density Functional Theory study of frequency-dependent polarizabilities and van der Waals dispersion coefficients for polyatomic molecules*, Journal of Chemical Physics 103, 9347 (1995) (https://doi.org/10.1063/1.469994)

Circular Dichroism (CD) J. Autschbach and T. Ziegler, *Calculating molecular electric and magnetic properties from time-dependent density functional response theory*, Journal of Chemical Physics 116, 891 (2002) (https://doi.org/10.1063/1.1420401)

J. Autschbach, T. Ziegler, S.J.A. van Gisbergen and E.J. Baerends, *Chiroptical properties from time-dependent density functional theory. I. Circular dichroism spectra of organic molecules*, Journal of Chemical Physics 116, 6930 (2002) (https://doi.org/10.1063/1.1436466)

Optical Rotation (OR), Optical Rotation Dispersion (ORD) J. Autschbach and T. Ziegler, *Calculating molecular electric and magnetic properties from time-dependent density functional response theory*, Journal of Chemical Physics 116, 891 (2002) (https://doi.org/10.1063/1.1420401)

J. Autschbach, S. Patchkovskii, T. Ziegler, S.J.A. van Gisbergen and E.J. Baerends, *Chiroptical properties from time-dependent density functional theory. II. Optical rotations of small to medium sized organic molecules*, Journal of Chemical Physics 117, 581 (2002) (https://doi.org/10.1063/1.1477925)

Magnetizability M. Krykunov and J. Autschbach, *Calculation of static and dynamic linear magnetic response in approximate time-dependent density functional theory*, Journal of Chemical Physics 126, 24101 (2007) (https://doi.org/10.1063/1.2423007)

Magnetic Circular Dichroism (MCD) M. Seth, M. Krykunov, T. Ziegler, J. Autschbach and A. Banerjee, *Application of magnetically perturbed time-dependent density functional theory to magnetic circular dichroism: Calculation of B terms*, Journal of Chemical Physics 128, 144105 (2008) (https://doi.org/10.1063/1.2901967)

M. Seth, M. Krykunov, T. Ziegler and J. Autschbach, *Application of magnetically perturbed time-dependent density functional theory to magnetic circular dichroism. II. Calculation of A terms*, Journal of Chemical Physics 128, 234102 (2008) (https://doi.org/10.1063/1.2933550)

M. Seth, T. Ziegler and J. Autschbach, *Application of magnetically perturbed time-dependent density functional theory to magnetic circular dichroism. III. Temperature-dependent magnetic circular dichroism induced by spin-orbit coupling*, Journal of Chemical Physics 129, 104105 (2008) (https://doi.org/10.1063/1.2976568)

Verdet constant and Faraday term M. Krykunov, A. Banerjee, T. Ziegler and J. Autschbach, *Calculation of Verdet constants with time-dependent density functional theory. Implementation and results for small molecules*, Journal of Chemical Physics 122, 074105 (2005) (https://doi.org/10.1063/1.1850919)

M. Krykunov, M. Seth, T. Ziegler and J. Autschbach, *Calculation of the magnetic circular dichroism B term from the imaginary part of the Verdet constant using damped time-dependent density functional theory*, Journal of Chemical Physics 127, 244102 (2007) (https://doi.org/10.1063/1.2806990)

15.2.8 LFDFT

Lead reference M. Atanasov, C.A. Daul, C. Rauzy, *A DFT Based Ligand Field Theory*, Structure & Bonding 106, 97 (2004) (https://doi.org/10.1007/b11308)

LFDFT for two-open-shell configuration H. Ramanantoanina, W. Urland, F. Cimpoesu, and C. Daul, *Ligand field density functional theory calculation of the $4f^2 \rightarrow 4f^1 5d^1$ transitions in the quantum cutter $Cs_2 KYF_6 Pr^{3+}$*, Physical Chemistry Chemical Physics 15, 13902 (2013) (https://doi.org/10.1039/C3CP51344K).

15.2.9 NMR

NMR Chemical Shifts

Lead reference G. Schreckenbach and T. Ziegler, *The calculation of NMR shielding tensors using GIAO's and modern density functional theory*, Journal of Physical Chemistry 99, 606 (1995) (https://doi.org/10.1021/j100002a024)

NMR chemical shifts with hybrid functionals M. Krykunov, T. Ziegler and E. van Lenthe, *Hybrid density functional calculations of nuclear magnetic shieldings using Slater-type orbitals and the zeroth-order regular approximation*, International Journal of Quantum Chemistry 109, 1676 (2009) (https://doi.org/10.1002/qua.21985)

NMR chemical shifts with NBO analysis J. Autschbach and S. Zheng, *Analyzing Pt chemical shifts calculated from relativistic density functional theory using localized orbitals: The role of Pt 5d lone pairs*, Magnetic Resonance in Chemistry 46, S45 (2008) (https://doi.org/10.1002/mrc.2289)

J. Autschbach, *Analyzing NMR shielding tensors calculated with two-component relativistic methods using spin-free localized molecular orbitals*, Journal of Chemical Physics 128, 164112 (2008) (https://doi.org/10.1063/1.2905235)

Paramagnetic NMR chemical shifts J. Autschbach, S. Patchkovskii, and B. Pritchard, *Calculation of Hyperfine Tensors and Paramagnetic NMR Shifts Using the Relativistic Zeroth-Order Regular Approximation and Density Functional Theory*, Journal of Chemical Theory and Computation 7, 2175 (2011) (https://doi.org/10.1021/ct200143w)

Suggested G. Schreckenbach and T. Ziegler, *The calculation of NMR shielding tensors based on density functional theory and the frozen-core approximation*, International Journal of Quantum Chemistry 60, 753 (1996) (https://doi.org/10.1002/(SICI)1097-461X(1996)60:3%3C753::AID-QUA4%3E3.0.CO;2-W)

G. Schreckenbach and T. Ziegler, *Calculation of NMR shielding tensors based on density functional theory and a scalar relativistic Pauli-type Hamiltonian. The application to transition metal complexes*, International Journal of Quantum Chemistry 61, 899 (1997) (https://doi.org/10.1002/(SICI)1097-461X(1997)61:6%3C899::AID-QUA3%3E3.0.CO;2-R)

S.K. Wolff and T. Ziegler, *Calculation of DFT-GIAO NMR shifts with inclusion of spin-orbit coupling*, Journal of Chemical Physics 109, 895 (1998) (https://doi.org/10.1063/1.476630)

S.K. Wolff, T. Ziegler, E. van Lenthe and E.J. Baerends, *Density functional calculations of nuclear magnetic shieldings using the zeroth-order regular approximation (ZORA) for relativistic effects: ZORA nuclear magnetic resonance*, Journal of Chemical Physics 110, 7689 (1999) (https://doi.org/10.1063/1.478680)

NMR spin-spin coupling

Lead J. Autschbach and T. Ziegler, *Nuclear spin-spin coupling constants from regular approximate density functional calculations. I. Formalism and scalar relativistic results for heavy metal compounds*, Journal of Chemical Physics 113, 936 (2000) (https://doi.org/10.1063/1.481874)

J. Autschbach, and T. Ziegler, *Nuclear spin-spin coupling constants from regular approximate relativistic density functional calculations. II. Spin-orbit coupling effects and anisotropies*, Journal of Chemical Physics 113, 9410 (2000) (https://doi.org/10.1063/1.1321310)

NMR spin-spin couplings with PBE0 J. Autschbach, *Two-component relativistic hybrid density functional computations of nuclear spin-spin coupling tensors using Slater-type basis sets and density-fitting techniques*, Journal of Chemical Physics 129, 094105 (2008) (https://doi.org/10.1063/1.2969100), Erratum: Journal of Chemical Physics 130, 209901 (2009) (https://doi.org/10.1063/1.3131724)

NMR spin-spin couplings with NBO analysis J. Autschbach, *Analyzing molecular properties calculated with two-component relativistic methods using spin-free Natural Bond Orbitals: NMR spin-spin coupling constants* Journal of Chemical Physics 127, 124106 (2007) (https://doi.org/10.1063/1.2768363)

Suggested R.M. Dickson and T. Ziegler, *NMR Spin.Spin Coupling Constants from Density Functional Theory with Slater-Type Basis Functions*, Journal of Physical Chemistry 100, 5286 (1996) (https://doi.org/10.1021/jp951930l)

J. Khandogin and T. Ziegler, *A density functional study of nuclear magnetic resonance spin.spin coupling constants in transition-metal systems*, Spectrochimica Acta 55, 607 (1999) (https://doi.org/10.1016/S1386-1425(98)00265-0)

J. Autschbach and T. Ziegler, *Solvent Effects on Heavy Atom Nuclear Spin.Spin Coupling Constants: A Theoretical Study of Hg.C and Pt.P Couplings*, Journal of the American Chemical Society 123, 3341 (2001) (https://doi.org/10.1021/ja003481v)

J. Autschbach and T. Ziegler, *A Theoretical Investigation of the Remarkable Nuclear Spin.Spin Coupling Pattern in [(NC)5 Pt-Tl(CN)]-*, Journal of the American Chemical Society 123, 5320 (2001) (https://doi.org/10.1021/ja003866d)

Suggested book reference J. Autschbach, T. Ziegler, in *Encyclopedia of Nuclear Magnetic Resonance*, Eds. D.M. Grant, R. K. Harris, John Wiley and Sons, Chichester, 2002, Vol. 9 *Advances in NMR*.

15.2.10 ESR/EPR

G-tensor: Zeeman interaction

Lead reference (self-consistent spin-orbit coupling) E. van Lenthe, A. van der Avoird and P.E.S. Wormer, *Density functional calculations of molecular g-tensors in the zero order regular approximation for relativistic effects*, Journal of Chemical Physics 107, 2488 (1997) (https://doi.org/10.1063/1.474590)

Lead reference (perturbative inclusion spin-orbit coupling) J. Autschbach and B. Pritchard, *Calculation of molecular g-tensors using the zeroth-order regular approximation and density functional theory: expectation value versus linear response approaches*, Theoretical Chemistry Accounts 129, 453 (2011) (https://doi.org/10.1007/s00214-010-0880-x)

Lead references (perturbative inclusion spin-orbit coupling with EPR/NMR program) G. Schreckenbach and T. Ziegler, *Calculation of the G-tensor of electron paramagnetic resonance spectroscopy using Gauge-Including Atomic Orbitals and Density Functional Theory*, Journal of Physical Chemistry A 101, 3388 (1997) (https://doi.org/10.1021/jp963060t) (for ESR/EPR g-tensor)

S. Patchkovskii and T. Ziegler, *Calculation of the EPR g-Tensors of High-Spin Radicals with Density Functional Theory*, Journal of Physical Chemistry A 105, 5490 (2001) (https://doi.org/10.1021/jp010457a) (for high-spin ESR/EPR g-tensor)

A-tensor: Nuclear magnetic dipole hyperfine interaction

Lead reference E. van Lenthe, A. van der Avoird and P.E.S. Wormer, *Density functional calculations of molecular hyperfine interactions in the zero order regular approximation for relativistic effects*, Journal of Chemical Physics 108, 4783 (1998) (https://doi.org/10.1063/1.475889)

Lead reference (perturbative inclusion spin-orbit coupling) J. Autschbach, S. Patchkovskii, and B. Pritchard, *Calculation of Hyperfine Tensors and Paramagnetic NMR Shifts Using the Relativistic Zeroth-Order Regular Approximation and Density Functional Theory*, Journal of Chemical Theory and Computation 7, 2175 (2011) (https://doi.org/10.1021/ct200143w)

Electric Field Gradient, NQCC

Lead reference (in ESR called Q-tensor: Nuclear electric quadrupole hyperfine interaction) E. van Lenthe and E.J. Baerends, *Density functional calculations of nuclear quadrupole coupling constants in the zero-order regular approximation for relativistic effects*, Journal of Chemical Physics 112, 8279 (2000) (https://doi.org/10.1063/1.481433)

EFG with NBO analysis A.J. Rossini, R.W. Mills, G.A. Briscoe, E.L. Norton, S.J. Geier, I. Hung, S. Zheng, J. Autschbach, and R.W. Schurko, *Solid-State Chlorine NMR of Group IV Transition Metal Organometallic Complexes*, Journal of the American Chemical Society 131, 3317 (2009) (https://doi.org/10.1021/ja808390a)

J. Autschbach, S. Zheng, and R.W. Schurko, *Analysis of Electric Field Gradient Tensors at Quadrupolar Nuclei in Common Structural Motifs*, Concepts in Magnetic Resonance Part A 36A, 84 (2010) (https://doi.org/10.1002/cmr.a.20155)

15.2.11 Transport properties: Non-self-consistent Green's function

Lead reference Chapter 2 and appendix C of J.S. Seldenthuis, Electrical and mechanical effects in single-molecule junctions (http://www.scm.com/Doc/Seldenthuis2011.pdf), PhD thesis, Delft University of Technology, 2011

Wide-band limit C.J.O. Verzijl, J.S. Seldenthuis, and J.M. Thijssen, *Applicability of the wide-band limit in DFT-based molecular transport calculations*, Journal of Chemical Physics 138, 094102 (2013) (https://doi.org/10.1063/1.4793259)

15.2.12 Analysis

Bond Energy Analysis T. Ziegler and A. Rauk, *A theoretical study of the ethylene-metal bond in complexes between* Cu^+, Ag^+, Au^+, Pt^0 or Pt^{2+} *and ethylene, based on the Hartree-Fock-Slater transition-state method*, Inorganic Chemistry 18, 1558 (1979) (https://doi.org/10.1021/ic50196a034)

T. Ziegler and A. Rauk, *Carbon monoxide, carbon monosulfide, molecular nitrogen, phosphorus trifluoride, and methyl isocyanide as sigma donors and pi acceptors. A theoretical study by the Hartree-Fock-Slater transition-state method*, Inorganic Chemistry 18, 1755 (1979) (https://doi.org/10.1021/ic50197a006)

F.M. Bickelhaupt and E.J. Baerends, In: Rev. Comput. Chem.; K.B. Lipkowitz and D.B. Boyd, Eds.; Wiley, New York, 2000, Vol. 15, p.1-86

ETS-NOCV M. Mitoraj, A. Michalak and T. Ziegler, *A Combined Charge and Energy Decomposition Scheme for Bond Analysis*, Journal of Chemical Theory and Computation 5, 962 (2009) (https://doi.org/10.1021/ct800503d)

QTAIM, Bader analysis

Grid-based algorithm J.I. Rodríguez, R.F.W. Bader, P.W. Ayers, C. Michel, A.W. Götz and C. Bo, *A high performance grid-based algorithm for computing QTAIM properties*, Chemical Physics Letters 472, 149 (2009) (https://doi.org/10.1016/j.cplett.2009.02.081)

J.I. Rodríguez, *An Efficient Method for Computing the QTAIM Topology of a Scalar Field: The Electron Density Case*, Journal of Computational Chemistry 34, 681 (2013) (https://doi.org/10.1002/jcc.23180)

Localized molecular orbitals J. Autschbach and H.F. King, *Analyzing molecular static linear response properties with perturbed localized orbitals*, Journal of Chemical Physics 133, 044109 (2010) (https://doi.org/10.1063/1.3455709)

15.2.13 Accuracy and efficiency

Numerical integration M. Franchini, P.H.T. Philipsen, L. Visscher, *The Becke Fuzzy Cells Integration Scheme in the Amsterdam Density Functional Program Suite*, Journal of Computational Chemistry 34, 1818 (2013) (https://doi.org/10.1002/jcc.23323).

Density fitting M. Franchini, P.H.T. Philipsen, E. van Lenthe, L. Visscher, *Accurate Coulomb Potentials for Periodic and Molecular Systems through Density Fitting*, Journal of Chemical Theory and Computation 10, 1994 (2014) (https://doi.org/10.1021/ct500172n)

15.3 External programs and Libraries

Click here for the list of programs and/or libraries used in the ADF package. On some platforms optimized libraries have been used and/or vendor specific MPI implementations.

REFERENCES

1. E.J. Baerends, V. Branchadell and M. Sodupe, *Atomic reference-energies for density functional calculations*, Chemical Physics Letters 265,481 (1997) (https://doi.org/10.1016/S0009-2614(96)01449-2)

2. F.M. Bickelhaupt and E.J. Baerends, *Kohn-Sham DFT: Predicting and Understanding Chemistry*, in *Reviews in Computational Chemistry*, D.B. Boyd and K.B. Lipkowitz, Editors. 2000, Wiley-VCH: New York. p. 1-86.

3. T. Ziegler and A. Rauk, *On the calculation of Bonding Energies by the Hartree Fock Slater method. I. The Transition State Method*, Theoretica Chimica Acta 46, 1 (1977) (https://doi.org/10.1007/BF00551648)

4. P.J. van den Hoek, A.W. Kleyn and E.J. Baerends, *What is the origin of the repulsive wall in atom-atom potentials*, Comments Atomic and Molecular Physics 23, 93 (1989)

5. E.J. Baerends, *Pauli repulsion effects in scattering from and catalysis by surface*, in *Cluster models for surface and bulk phenomena*, ISBN13: 9780306441028, G. Puccchiori, P.S. Bagus and F. Parmigiani, Editors. 1992, Springer: New-York. p. 189-207.

6. L. Versluis and T. Ziegler, *The determination of Molecular Structure by Density Functional Theory*, Journal of Chemical Physics 88, 322 (1988) (https://doi.org/10.1063/1.454603)

7. L. Versluis, The determination of molecular structures by the HFS method, PhD thesis, University of Calgary, 1989

8. L. Fan and T. Ziegler, *Optimization of molecular structures by self consistent and non-local density functional theory*, Journal of Chemical Physics 95, 7401 (1991) (https://doi.org/10.1063/1.461366)

9. L. Deng, T. Ziegler and L. Fan, *A combined density functional and intrinsic reaction coordinate study on the ground state energy surface of H_2 CO*, Journal of Chemical Physics 99, 3823 (1993) (https://doi.org/10.1063/1.466129)

10. L. Deng and T. Ziegler, *The determination of Intrinsic Reaction Coordinates by density functional theory*, International Journal of Quantum Chemistry 52, 731 (1994) (https://doi.org/10.1002/qua.560520406)

11. T.H. Fischer and J. Almlöf, *General Methods for Geometry and Wave Function Optimization*, Journal of Physical Chemistry 96, 9768 (1992) (https://doi.org/10.1021/j100203a036)

12. L. Fan and T. Ziegler, *Application of density functional theory to infrared absorption intensity calculations on main group molecules*, Journal of Chemical Physics 96, 9005 (1992) (https://doi.org/10.1063/1.462258)

13. L. Fan and T. Ziegler, *Nonlocal density functional theory as a practical tool in calculations on transition states and activation energies*, Journal of the American Chemical Society 114, 10890 (1992) (https://doi.org/10.1021/ja00053a027)

14. A. Bérces, Application of density functional theory to the vibrational characterization of transition metal complexes, PhD thesis, University of Calgary, 1995

15. R. van Leeuwen and E.J. Baerends, *Exchange-correlation potential with correct asymptotic behavior*, Physical Review A 49, 2421 (1994) (https://doi.org/10.1103/PhysRevA.49.2421)

16. M. Grüning, O.V. Gritsenko, S.J.A. van Gisbergen and E.J. Baerends, *Shape corrections to exchange-correlation Kohn-Sham potentials by gradient-regulated seamless connection of model potentials for inner and outer region*, Journal of Chemical Physics 114, 652 (2001) (https://doi.org/10.1063/1.1327260)

17. P.R.T. Schipper, O.V. Gritsenko, S.J.A. van Gisbergen and E.J. Baerends, *Molecular calculations of excitation energies and (hyper)polarizabilities with a statistical average of orbital model exchange-correlation potentials*, Journal of Chemical Physics 112, 1344 (2000) (https://doi.org/10.1063/1.480688)

18. M. Grüning, O.V. Gritsenko, S.J.A. van Gisbergen and E.J. Baerends, *On the required shape correction to the LDA and GGA Kohn Sham potentials for molecular response calculations of (hyper)polarizabilities and excitation energies*, Journal of Chemical Physics 116, 9591 (2002) (https://doi.org/10.1063/1.1476007)

19. D.P. Chong, O.V. Gritsenko and E.J. Baerends, *Interpretation of the Kohn-Sham orbital energies as approximate vertical ionization potentials*, Journal of Chemical Physics 116, 1760 (2002) (https://doi.org/10.1063/1.1430255)

20. S.H. Vosko, L. Wilk and M. Nusair, *Accurate spin-dependent electron liquid correlation energies for local spin density calculations: a critical analysis*, Canadian Journal of Physics 58 (8), 1200 (1980) (https://doi.org/10.1139/p80-159)

21. H. Stoll, C.M.E. Pavlidou, and H. Preuss, *On the calculation of correlation energies in the spin-density functional formalism*, Theoretica Chimica Acta 49, 143 (1978) (https://doi.org/10.1007/BF00553794)

22. A.D. Becke, *Density-functional exchange-energy approximation with correct asymptotic behavior*, Physical Review A 38, 3098 (1988) (https://doi.org/10.1103/PhysRevA.38.3098)

23. J.P. Perdew and Y. Wang, *Accurate and simple density functional for the electronic exchange energy: generalized gradient approximation*, Physical Review B 33, 8822 (1986) (https://doi.org/10.1103/PhysRevB.33.8800)

24. J.P. Perdew, J.A. Chevary, S.H. Vosko, K.A. Jackson, M.R. Pederson, D.J. Sing and C. Fiolhais, *Atoms, molecules, solids, and surfaces: Applications of the generalized gradient approximation for exchange and correlation*, Physical Review B 46, 6671 (1992) (https://doi.org/10.1103/PhysRevB.46.6671)

25. C. Adamo and V. Barone, *Exchange functionals with improved long-range behavior and adiabatic connection methods without adjustable parameters: The mPW and mPW1PW models*, Journal of Chemical Physics 108, 664 (1998) (https://doi.org/10.1063/1.475428)

26. J.P. Perdew, K. Burke and M. Ernzerhof, *Generalized Gradient Approximation Made Simple*, Physical Review Letters 77, 3865 (1996) (https://doi.org/10.1103/PhysRevLett.77.3865)

27. B. Hammer, L.B. Hansen, and J.K. Norskøv, *Improved adsorption energetics within density-functional theory using revised Perdew-Burke-Ernzerhof functionals*, Physical Review B 59, 7413 (1999) (https://doi.org/10.1103/PhysRevB.59.7413)

28. Y. Zhang and W. Yang, *Comment on "Generalized Gradient Approximation Made Simple"*, Physical Review Letters 80, 890 (1998) (https://doi.org/10.1103/PhysRevLett.80.890)

29. N.C. Handy and A.J. Cohen, *Left-right correlation energy*, Molecular Physics 99, 403 (2001) (https://doi.org/10.1080/00268970010018431)

30. J.P. Perdew, *Density-functional approximation for the correlation energy of the inhomogeneous electron gas*, Physical Revied B 33, 8822 (1986) (https://doi.org/10.1103/PhysRevB.33.8822) Erratum: J.P. Perdew, Physical Review B 34, 7406 (1986) (https://doi.org/10.1103/PhysRevB.34.7406)

31. C. Lee, W. Yang and R.G. Parr, *Development of the Colle-Salvetti correlation-energy formula into a functional of the electron density*, Physical Review B 37, 785 (1988) (https://doi.org/10.1103/PhysRevB.37.785)

32. B.G. Johnson, P.M.W. Gill and J.A. Pople, *The performance of a family of density functional methods*, Journal of Chemical Physics 98, 5612 (1993) (https://doi.org/10.1063/1.464906)

33. T.V. Russo, R.L. Martin and P.J. Hay, *Density Functional calculations on first-row transition metals*, Journal of Chemical Physics 101, 7729 (1994) (https://doi.org/10.1063/1.468265)

34. R. Neumann, R.H. Nobes and N.C. Handy, *Exchange functionals and potentials*, Molecular Physics 87, 1 (1996) (https://doi.org/10.1080/00268979600100011)

35. J.P. Perdew, K. Burke and M. Ernzerhof, *ERRATA for "Generalized Gradient Approximation Made Simple [Phys. Rev. Lett. 77, 3865 (1996)]"* Physical Review Letters 78, 1396 (1997) (https://doi.org/10.1103/PhysRevLett.78.1396)

36. F.A. Hamprecht, A.J. Cohen, D.J. Tozer and N.C. Handy, *Development and assessment of new exchange-correlation functionals*, Journal of Chemical Physics 109, 6264 (1988) (https://doi.org/10.1063/1.477267)

37. A.D. Boese, N.L. Doltsinis, N.C. Handy and M. Sprik, *New generalized gradient approximation functionals*, Journal of Chemical Physics 112, 1670 (2000) (https://doi.org/10.1063/1.480732)

38. A.D. Boese and N.C. Handy, *A new parametrization of exchange-correlation generalized gradient approximation functionals*, Journal of Chemical Physics 114, 5497 (2001) (https://doi.org/10.1063/1.1347371)

39. T. Tsuneda, T. Suzumura and K. Hirao, *A new one-parameter progressive Colle.Salvetti-type correlation functional*, Journal of Chemical Physics 110, 10664 (1999) (https://doi.org/10.1063/1.479012)

40. J.B. Krieger, J. Chen, G.J. Iafrate and A. Savin, in *Electron Correlations and Materials Properties*, ISBN13: 9780306462825, A. Gonis and N. Kioussis, Editors. 1999, Plenum: New York.

41. J.P. Perdew, S. Kurth, A. Zupan and P. Blaha, *Erratum: Accurate Density Functional with Correct Formal Properties: A Step Beyond the Generalized Gradient Approximation [Phys. Rev. Lett. 82, 2544 (1999)]*, Physical Review Letters 82, 5179 (1999) (https://doi.org/10.1103/PhysRevLett.82.5179).

42. T. van Voorhis and G.E. Scuseria, *A novel form for the exchange-correlation energy functional*, Journal of Chemical Physics 109, 400 (1998) (https://doi.org/10.1063/1.476577)

43. M. Filatov and W. Thiel, *A new gradient-corrected exchange-correlation density functional*, Molecular Physics 91, 847 (1997) (https://doi.org/10.1080/002689797170950)

44. M. Filatov and W. Thiel, *Exchange-correlation density functional beyond the gradient approximation*, Physical Review A 57, 189 (1998) (https://doi.org/10.1103/PhysRevA.57.189)

45. E.I. Proynov, S. Sirois and D.R. Salahub, *Extension of the LAP functional to include parallel spin correlation*, International Journal of Quantum Chemistry 64, 427 (1997) (https://doi.org/10.1002/(SICI)1097-461X(1997)64:4%3C427::AID-QUA5%3E3.0.CO;2-Y/abstract)

46. E.I. Proynov, H. Chermette and D.R. Salahub, *New tau-dependent correlation functional combined with a modified Becke exchange*, Journal of Chemical Physics 113, 10013 (2000) (https://doi.org/10.1063/1.1321309)

47. S. Patchkovskii, J. Autschbach and T. Ziegler, *Curing difficult cases in magnetic properties prediction with self-interaction corrected density functional theory*, Journal of Chemical Physics 115, 26 (2001) (https://doi.org/10.1063/1.1370527)

48. S. Patchkovskii and T. Ziegler, *Improving "difficult" reaction barriers with self-interaction corrected density functional theory*, Journal of Chemical Physics 1167806 (2002) (https://doi.org/10.1063/1.1468640)

49. S. Patchkovskii and T. Ziegler, *Phosphorus NMR chemical shifts with self-interaction free, gradient-corrected DFT*, Journal of Physical Chemistry A 106, 1088 (2002) (https://doi.org/10.1021/jp014184v)

50. P.H.T. Philipsen, E. van Lenthe, J.G. Snijders and E.J. Baerends, *Relativistic calculations on the adsorption of CO on the (111) surfaces of Ni, Pd, and Pt within the zeroth-order regular approximation*, Physical Review B 56, 13556 (1997) (https://doi.org/10.1103/PhysRevB.56.13556)

51. J.G. Snijders and E.J. Baerends, *A perturbation theory approach to relativistic calculations. I. Atoms*, Molecular Physics 36, 1789 (1978) (https://doi.org/10.1080/00268977800102771)

52. J.G. Snijders, E.J. Baerends and P. Ros, *A perturbation theory approach to relativistic calculations. II. Molecules*, Molecular Physics 38, 1909 (1979) (https://doi.org/10.1080/00268977900102941)

53. T. Ziegler, J.G. Snijders and E.J. Baerends, *Relativistic effects on bonding*, Journal of Chemical Physics 74, 1271 (1981) (https://doi.org/10.1063/1.441187)

54. R.L. DeKock, E.J. Baerends, P.M. Boerrigter and J.G. Snijders, *On the nature of the first excited states of the uranyl ion*, Chemical Physics Letters 105, 308 (1984) (https://doi.org/10.1016/0009-2614(84)85036-8)

55. R.L. DeKock, E.J. Baerends, P.M. Boerrigter and R. Hengelmolen, *Electronic structure and bonding of* $Hg(CH_3)_2$, $Hg(CN)_2$, $Hg(CH_3)(CN)$, $Hg(CCCH_3)_2$, *and* $Au(PMe)_3 (CH_3)$, Journal of the American Chemical Society 106, 3387 (1984) (https://doi.org/10.1021/ja00324a001)

56. P.M. Boerrigter, *Spectroscopy and bonding of heavy element compounds*, 1987, Vrije Universiteit.

57. P.M. Boerrigter, M.A. Buijse and J.G. Snijders, *Spin-Orbit interaction in the excited states of the dihalogen ions* F_2^+, Cl_2^+ and Br_2^+, Chemical Physics 111, 47 (1987) (https://doi.org/10.1016/0301-0104(87)87007-6)

58. P.M. Boerrigter, E.J. Baerends and J.G. Snijders, *A relativistic LCAO Hartree-Fock-Slater investigation of the electronic structure of the actinocenes* $M(COT)_2$, *M=Th, Pa, U, Np and Pu*, Chemical Physics 122, 357 (1988) (https://doi.org/10.1016/0301-0104(88)80018-1)

59. T. Ziegler, V. Tschinke, E.J. Baerends, J.G. Snijders and W. Ravenek, *Calculation of bond energies in compounds of heavy elements by a quasi-relativistic approach*, Journal of Physical Chemistry 93, 3050 (1989) (https://doi.org/10.1021/j100345a036)

60. J. Li, G. Schreckenbach and T. Ziegler, *A Reassessment of the First Metal-Carbonyl Dissociation Energy in* $M(CO)_4$ *(M = Ni, Pd, Pt),* $M(CO)_5$ *(M = Fe, Ru, Os), and* $M(CO)_6$ *(M = Cr, Mo, W) by a Quasirelativistic Density Functional Method*, Journal of the American Chemical Society 117, 486 (1995) (https://doi.org/10.1021/ja00106a056)

61. E. van Lenthe, A.E. Ehlers and E.J. Baerends, *Geometry optimization in the Zero Order Regular Approximation for relativistic effects*, Journal of Chemical Physics 110, 8943 (1999) (https://doi.org/10.1063/1.478813)

62. E. van Lenthe, E.J. Baerends and J.G. Snijders, *Relativistic regular two-component Hamiltonians*, Journal of Chemical Physics 99, 4597 (1993) (https://doi.org/10.1063/1.466059)

63. E. van Lenthe, E.J. Baerends and J.G. Snijders, *Relativistic total energy using regular approximations*, Journal of Chemical Physics 101, 9783 (1994) (https://doi.org/10.1063/1.467943)

64. E. van Lenthe, J.G. Snijders and E.J. Baerends, *The zero-order regular approximation for relativistic effects: The effect of spin-orbit coupling in closed shell molecules*, Journal of Chemical Physics 105, 6505 (1996) (https://doi.org/10.1063/1.472460)

65. E. van Lenthe, R. van Leeuwen, E.J. Baerends and J.G. Snijders, *Relativistic regular two-component Hamiltonians*, International Journal of Quantum Chemistry 57, 281 (1996) (https://doi.org/10.1002/(SICI)1097-461X(1996)57:3%3C281::AID-QUA2%3E3.0.CO;2-U/abstract)

66. C.C. Pye and T. Ziegler, *An implementation of the conductor-like screening model of solvation within the Amsterdam density functional package*, Theoretical Chemistry Accounts 101, 396 (1999) (https://doi.org/10.1007/s002140050457)

67. A. Klamt and G. Schüürmann, *COSMO: a new approach to dielectric screening in solvents with explicit expressions for the screening energy and its gradient*, Journal of the Chemical Society: Perkin Transactions 2, 799 (1993) (https://doi.org/10.1039/P29930000799)

68. A. Klamt, *Conductor-like Screening Model for real solvents: A new approach to the quantitative calculation of solvation phenomena*, Journal of Physical Chemistry 99, 2224 (1995) (https://doi.org/10.1021/j100007a062)

69. A. Klamt and V. Jones, *Treatment of the outlying charge in continuum solvation models*, Journal of Chemical Physics 105, 9972 (1996) (https://doi.org/10.1063/1.472829)

70. J.L. Pascual-ahuir, E. Silla and I. Tuñon, *GEPOL: An improved description of molecular surfaces. III. A new algorithm for the computation of a solvent-excluding surface*, Journal of Computational Chemistry 15, 1127 (1994) (https://doi.org/10.1002/jcc.540151009)

71. S.J.A. van Gisbergen, J.G. Snijders and E.J. Baerends, *Implementation of time-dependent density functional response equations*, Computer Physics Communications 118, 119 (1999) (https://doi.org/10.1016/S0010-4655(99)00187-3)

72. S.J.A. van Gisbergen, *Molecular Response Property Calculations using Time Dependent Density Functional Theory*, in *Chemistry*. 1998, (http://www.scm.com/Doc/gisbergen.pdf.tar.gz) Vrije Universiteit: Amsterdam. p. 190.

73. E.K.U. Gross, J.F. Dobson and Petersilka, in *Density Functional Theory*, R.F. Nalewajski, Editor. 1996, Springer: Heidelberg.

74. S.J.A. van Gisbergen, V.P. Osinga, O.V. Gritsenko, R. van Leeuwen, J.G. Snijders and E.J. Baerends, *Improved density functional theory results for frequency-dependent polarizabilities, by the use of an exchange-correlation potential with correct asymptotic behavior*, Journal of Chemical Physics 105, 3142 (1996) (https://doi.org/10.1063/1.472182)

75. S.J.A. van Gisbergen, J.G. Snijders, and E.J. Baerends, *Time-dependent Density Functional Results for the Dynamic Hyperpolarizability of* C_{60}, Physical Review Letters 78, 3097 (1997) (https://doi.org/10.1103/PhysRevLett.78.3097)

76. S.J.A. van Gisbergen, J.G. Snijders and E.J. Baerends, *Calculating frequency-dependent hyperpolarizabilities using time-dependent density functional theory*, Journal of Chemical Physics 109, 10644 (1998) (https://doi.org/10.1063/1.477762)

77. S.J.A. van Gisbergen, J.G. Snijders and E.J. Baerends, *Accurate density functional calculations on frequency-dependent hyperpolarizabilities of small molecules*, Journal of Chemical Physics 109, 10657 (1998) (https://doi.org/10.1063/1.477763)

78. M.E. Casida, C. Jamorski, K.C. Casida and D.R. Salahub, *Molecular excitation energies to high-lying bound states from time-dependent density-functional response theory: Characterization and correction of the time-dependent local density approximation ionization threshold*, Journal of Chemical Physics 108, 4439 (1998) (https://doi.org/10.1063/1.475855)

79. V.P. Osinga, S.J.A. van Gisbergen, J.G. Snijders and E.J. Baerends, *Density functional results for isotropic and anisotropic multipole polarizabilities and* C_6 , C_7 , *and* C_8 *Van der Waals dispersion coefficients for molecules*, Journal of Chemical Physics 106, 5091 (1997) (https://doi.org/10.1063/1.473555)

80. J. Autschbach and T. Ziegler, *Calculating molecular electric and magnetic properties from time-dependent density functional response theory*, Journal of Chemical Physics 116, 891 (2002) (https://doi.org/10.1063/1.1420401)

81. J. Autschbach, T. Ziegler, S.J.A. van Gisbergen and E.J. Baerends, *Chiroptical properties from time-dependent density functional theory. I. Circular dichroism spectra of organic molecules*, Journal of Chemical Physics 116, 6930 (2002) (https://doi.org/10.1063/1.1436466)

82. S.J.A. van Gisbergen, F. Kootstra, P.R.T. Schipper, O.V. Gritsenko, J.G. Snijders and E.J. Baerends, *Density-functional-theory response-property calculations with accurate exchange-correlation potentials*, Physical Review A 57, 2556 (1998) (https://doi.org/10.1103/PhysRevA.57.2556)

83. S.J.A. van Gisbergen, A. Rosa, G. Ricciardi and E.J. Baerends, *Time-dependent density functional calculations on the electronic absorption spectrum of free base porphin*, Journal of Chemical Physics 111, 2499 (1999) (https://doi.org/10.1063/1.479617)

84. A. Rosa, G. Ricciardi, E.J. Baerends and S.J.A. van Gisbergen, *The Optical Spectra of NiP, Nipz, NiTBP, and NiPc. Electronic effects of meso-tetraaza substitution and tetrabenzoannulation*, Journal of Physical Chemistry A 105, 3311 (2001) (https://doi.org/10.1021/jp003508x)

85. G. Ricciardi, A. Rosa and E.J. Baerends, *Ground and Excited States of Zinc Phthalocyanine studied by Density Functional Methods*, Journal of Physical Chemistry A 105, 5242 (2001) (https://doi.org/10.1021/jp0042361)

86. S.J.A. van Gisbergen, J.A. Groeneveld, A. Rosa, J.G. Snijders and E.J.Baerends, *Excitation energies for transition metal compounds from time-dependent density functional theory. Applications to* $MnO4^-$, $Ni(CO)_4$ and $Mn^2 (CO)_{10}$, Journal of Physical Chemistry A 103, 6835 (1999) (https://doi.org/10.1021/jp991060y)

87. A. Rosa, E.J. Baerends, S.J.A. van Gisbergen, E. van Lenthe, J.A. Groeneveld and J. G. Snijders, *Article Electronic Spectra of* $M(CO)_6$ (M = Cr, Mo, W) *Revisited by a Relativistic TDDFT Approach*, Journal of the American Chemical Society 121, 10356 (1999) (https://doi.org/10.1021/ja990747t)

88. S.J.A. van Gisbergen, C. Fonseca Guerra and E.J. Baerends, *Towards excitation energies and (hyper)polarizability calculations of large molecules. Application of parallelization and linear scaling techniques to time-dependent density functional response theory*, Journal of Computational Chemistry 21, 1511 (2000) (https://doi.org/10.1002/1096-987X(200012)21:16%3C1511::AID-JCC8%3E3.0.CO;2-C/abstract)

89. S.J.A. van Gisbergen, J.G. Snijders and E.J. Baerends, *A Density Functional Theory study of frequency-dependent polarizabilities and van der Waals dispersion coefficients for polyatomic molecules*, Journal of Chemical Physics 103, 9347 (1995) (https://doi.org/10.1063/1.469994)

90. B. Champagne, E.A. Perpète, S.J.A. van Gisbergen, E.J. Baerends, J.G. Snijders, C. Soubra-Ghaoui, K.A. Robins and B.Kirtman, *Assessment of conventional density functional schemes for computing the polarizabilities and hyperpolarizabilities of conjugated oligomers: An ab initio investigation of polyacetylene chains*, Journal of Chemical Physics 109, 10489 (1998) (https://doi.org/10.1063/1.477731) Erratum: Journal of Chemical Physics 111, 6652 (1999) (https://doi.org/10.1063/1.479106)

91. D.M. Bishop, *Aspects of Non-Linear-Optical Calculations*, Advances in Quantum Chemistry 25, 3 (1994) (https://doi.org/10.1016/S0065-3276(08)60017-9)

92. A. Willets, J.E. Rice, D.M. Burland and D.P. Shelton, *Problems in comparison of experimental and theoretical hyperpolarizabilities*, Journal of Chemical Physics 97, 7590 (1992) (https://doi.org/10.1063/1.463479)

93. D.P. Shelton and J.E. Rice, *Measurements and calculations of the hyperpolarizabilities of atoms and small molecules in the gas phase*, Chemical Reviews 94, 3 (1994) (https://doi.org/10.1021/cr00025a001)

94. J. Autschbach, S. Patchkovskii, T. Ziegler, S.J.A. van Gisbergen and E.J. Baerends, *Chiroptical properties from time-dependent density functional theory. II. Optical rotations of small to medium sized organic molecules*, Journal of Chemical Physics 117, 581 (2002) (https://doi.org/10.1063/1.1477925)

95. E. van Lenthe, A. van der Avoird and P.E.S. Wormer, *Density functional calculations of molecular g-tensors in the zero order regular approximation for relativistic effects*, Journal of Chemical Physics 107, 2488 (1997) (https://doi.org/10.1063/1.474590)

96. E. van Lenthe, A. van der Avoird and P.E.S. Wormer, *Density functional calculations of molecular hyperfine interactions in the zero order regular approximation for relativistic effects*, Journal of Chemical Physics 108, 4783 (1998) (https://doi.org/10.1063/1.475889)

97. E. van Lenthe and E.J. Baerends, *Density functional calculations of nuclear quadrupole coupling constants in the zero-order regular approximation for relativistic effects*, Journal of Chemical Physics 112, 8279 (2000) (https://doi.org/10.1063/1.481433)

98. R.E. Bulo, A.W. Ehlers, S. Grimme and K. Lammertsma, *Vinylphosphirane.Phospholene Rearrangements: Pericyclic [1,3]-Sigmatropic Shifts or Not?* Journal of the American Chemical Society 124, 13903 (2002) (https://doi.org/10.1021/ja027925u)

99. R. Pauncz, *Spin Eigenfunctions*, ISBN13: 9780306401411, 1979, New York: Plenum Press

100. A. Szabo and N.S. Ostlund, *Modern Quantum Chemistry*, ISBN13: 9780070627390, 1st ed. revised ed. 1989: McGraw-Hill

101. H. Eschrig and V.D.P. Servedio, *Relativistic density functional approach to open shells*, Journal of Computational Chemistry 20, 23 (1999) (https://doi.org/10.1002/(SICI)1096-987X(19990115)20:1%3C23::AID-JCC5%3E3.0.CO;2-N)

102. C. van Wüllen, *Spin densities in two-component relativistic density functional calculations: Noncollinear versus collinear approach*, Journal of Computational Chemistry 23, 779 (2002) (https://doi.org/10.1002/jcc.10043)

103. S.G. Wang and W.H.E. Schwarz, *Simulation of nondynamical correlation in density functional calculations by the optimized fractional orbital occupation approach: Application to the potential energy surfaces of* O_3 *and* SO_2, Journal of Chemical Physics 105, 4641 (1996) (https://doi.org/10.1063/1.472307)

104. P.M. Boerrigter, G. te Velde and E.J. Baerends, *Three-dimensional Numerical Integration for Electronic Structure Calculations*, International Journal of Quantum Chemistry 33, 87 (1988) (https://doi.org/10.1002/qua.560330204)

105. G. te Velde and E.J. Baerends, *Numerical Integration for Polyatomic Systems*, Journal of Computational Physics 99, 84 (1992) (https://doi.org/10.1016/0021-9991(92)90277-6)

106. A. Bérces and T. Ziegler, *The harmonic force field of benzene calculated by local density functional theory*, Chemical Physics Letters 203, 592 (1993) (https://doi.org/10.1016/0009-2614(93)85316-G)

107. A. Bérces and T. Ziegler, *The harmonic force field of benzene. A local density functional study*, Journal of Chemical Physics 98, 4793 (1993) (https://doi.org/10.1063/1.464983)

108. F. Neese and E. I. Solomon, *MCD C-Term Signs, Saturation Behavior, and Determination of Band Polarizations in Randomly Oriented Systems with Spin $S \leq 1/2$. Applications to S = 1/2 and S = 5/2*, Inorganic Chemistry 38, 1847 (1999) (https://doi.org/10.1021/ic981264d)

109. G. te Velde, *Numerical integration and other methodological aspects of bandstructure calculations*, in *Chemistry*. 1990, Vrije Universiteit: Amsterdam.

110. T. Ziegler and A. Rauk, *A theoretical study of the ethylene-metal bond in complexes between copper(1+), silver(1+), gold(1+), platinum(0) or platinum(2+) and ethylene, based on the Hartree-Fock-Slater transition-state method*, Inorganic Chemistry 18, 1558 (1979) (https://doi.org/10.1021/ic50196a034)

111. L. Noodleman, and E.J. Baerends, *Electronic Structure, Magnetic Properties, ESR, and Optical Spectra for 2-Fe Ferredoxin Models by LCAO-Xa Valence Bond Theory*, Journal of the American Chemical Society 106, 2316 (1984) (https://doi.org/10.1021/ja00320a017)

112. F.M. Bickelhaupt, N.M. Nibbering, E.M. van Wezenbeek and E.J. Baerends, *The Central Bond in the Three CN Dimers NC_CN, CN-CN, and CN-NC: Electron Pair Bonding and Pauli Repulsion Effects*, Journal of Physical Chemistry 96, 4864 (1992) (https://doi.org/10.1021/j100191a027)

113. G. Schreckenbach and T. Ziegler, *The calculation of NMR shielding tensors using GIAO's and modern density functional theory*, Journal of Physical Chemistry 99, 606 (1995) (https://doi.org/10.1021/j100002a024)

114. G. Schreckenbach and T. Ziegler, *The calculation of NMR shielding tensors based on density functional theory and the frozen-core approximation*, International Journal of Quantum Chemistry 60, 753 (1996) (https://doi.org/10.1002/(SICI)1097-461X(1996)60:3%3C753::AID-QUA4%3E3.0.CO;2-W)

115. G. Schreckenbach and T. Ziegler, *Calculation of NMR shielding tensors based on density functional theory and a scalar relativistic Pauli-type Hamiltonian. The application to transition metal complexes*, International Journal of Quantum Chemistry 61, 899 (1997) (https://doi.org/10.1002/(SICI)1097-461X(1997)61:6%3C899::AID-QUA3%3E3.0.CO;2-R)

116. S.K. Wolff and T. Ziegler, *Calculation of DFT-GIAO NMR shifts with inclusion of spin-orbit coupling*, Journal of Chemical Physics 109, 895 (1998) (https://doi.org/10.1063/1.476630)

117. S.K. Wolff, T. Ziegler, E. van Lenthe and E.J. Baerends, *Density functional calculations of nuclear magnetic shieldings using the zeroth-order regular approximation (ZORA) for relativistic effects: ZORA nuclear magnetic resonance* , Journal of Chemical Physics 110, 7689 (1999) (https://doi.org/10.1063/1.478680)

118. J. Autschbach and T. Ziegler, *Nuclear spin-spin coupling constants from regular approximate density functional calculations. I. Formalism and scalar relativistic results for heavy metal compounds*, Journal of Chemical Physics 113, 936 (2000) (https://doi.org/10.1063/1.481874)

119. J. Autschbach, and T. Ziegler, *Nuclear spin-spin coupling constants from regular approximate relativistic density functional calculations. II. Spin-orbit coupling effects and anisotropies*, Journal of Chemical Physics 113, 9410 (2000) (https://doi.org/10.1063/1.1321310)

120. G. Schreckenbach and T. Ziegler, *Calculation of the G-tensor of electron paramagnetic resonance spectroscopy using Gauge-Including Atomic Orbitals and Density Functional Theory*, Journal of Physical Chemistry A 101, 3388 (1997) (https://doi.org/10.1021/jp963060t)

121. S. Patchkovskii and T. Ziegler, *Calculation of the EPR g-Tensors of High-Spin Radicals with Density Functional Theory*, Journal of Physical Chemistry A 105, 5490 (2001) (https://doi.org/10.1021/jp010457a)

122. C. Edmiston and K. Rudenberg, *Localized Atomic and Molecular Orbitals*, Reviews of Modern Physics 35, 457 (1963) (https://doi.org/10.1103/RevModPhys.35.457)

123. J.M Foster and S.F. Boys, *Canonical Configurational Interaction Procedure*, Reviews of Modern Physics 32, 300 (1960) (https://doi.org/10.1103/RevModPhys.32.300)

124. W. von Niessen, *Density Localization of Atomic and Molecular Orbitals. I*, Journal of Chemical Physics 56, 4290 (1972) (https://doi.org/10.1063/1.1677859)

125. F.L. Hirshfeld, *Bonded-atom fragments for describing molecular charge densities*, Theoretica Chimica Acta 44, 129 (1977) (https://doi.org/10.1007/BF00549096)

126. K.B. Wiberg and P.R. Rablen, *Comparison of atomic charges derived via different procedures*, Journal of Computational Chemistry 14, 1504 (1993) (https://doi.org/10.1002/jcc.540141213)

127. F.M. Bickelhaupt, N.J.R. van Eikema Hommes, C. Fonseca Guerra and E.J. Baerends, *The Carbon-Lithium Electron Pair Bond in* $(CH3Li)_n$ (n = 1, 2, 4), Organometallics 15, 2923 (1996) (https://doi.org/10.1021/om950966x)

128. C. Fonseca Guerra, J.-W. Handgraaf, E. J. Baerends and F. M. Bickelhaupt, *Voronoi Deformation Density (VDD) charges. Assessment of the Mulliken, Bader, Hirshfeld, Weinhold and VDD methods for Charge Analysis*, Journal of Computational Chemistry 25, 189 (2004) (https://doi.org/10.1002/jcc.10351)

129. C. Fonseca Guerra, F.M. Bickelhaupt, J.G. Snijders and E.J. Baerends, *The Nature of the Hydrogen Bond in DNA Base Pairs: The Role of Charge Transfer and Resonance Assistance*, Chemistry - A European Journal 5, 3581 (1999) (https://doi.org/10.1002/(SICI)1521-3765(19991203)5:12%3C3581::AID-CHEM3581%3E3.0.CO;2-Y)

130. K. Kitaura and K. Morokuma, *A new energy decomposition scheme for molecular interactions within the Hartree-Fock approximation*, International Journal of Quantum Chemistry 10, 325 (1976) (https://doi.org/10.1002/qua.560100211)

131. T. Ziegler and A. Rauk, *Carbon monoxide, carbon monosulfide, molecular nitrogen, phosphorus trifluoride, and methyl isocyanide as sigma donors and pi acceptors. A theoretical study by the Hartree-Fock-Slater transition-state method*, Inorganic Chemistry 18, 1755 (1979) (https://doi.org/10.1021/ic50197a006)

132. H. Fujimoto, J. Osamura and T. Minato, *Orbital interaction and chemical bonds. Exchange repulsion and rehybridization in chemical reactions*, Journal of the American Chemical Society 100, 2954 (1978) (https://doi.org/10.1021/ja00478a004)

133. S. Wolfe, D.J. Mitchell and M.-H. Whangbo, *On the role of steric effects in the perturbational molecular orbital method of conformational analysis*, Journal of the American Chemical Society 100, 1936 (1978) (https://doi.org/10.1021/ja00474a055)

134. A.J. Stone and R.W. Erskine, *Intermolecular self-consistent-field perturbation theory for organic reactions. I. Theory and implementation; nucleophilic attack on carbonyl compounds*, Journal of the American Chemical Society 102, 7185 (1980) (https://doi.org/10.1021/ja00544a003)

135. F. Bernardi, A. Bottoni, A. Mangini and G. Tonachini, *Quantitative orbital analysis of ab initio SCF=MO computations : Part II. Conformational preferences in H2N—OH and H2N—SH*, Journal of Molecular Structure: THEOCHEM 86, 163 (1981) (https://doi.org/10.1016/0166-1280(81)85082-8)

136. P.J. van den Hoek and E.J. Baerends, *Chemical bonding at metal-semiconductor interfaces*, Applied Surface Science 41/42, 236 (1989) (https://doi.org/10.1016/0169-4332(89)90063-9)

137. J. Autschbach, *On the calculation of relativistic effects and how to understand their trends in atoms and molecules*, in *Chemistry*. 1999, (http://www.ub.uni-siegen.de/epub/diss/autschbach.htm) University of Siegen: Siegen.

138. M.A. Watson, N.C. Handy and A.J. Cohen, *Density functional calculations, using Slater basis sets, with exact exchange*, Journal of Chemical Physics 119, 6475 (2003) (https://doi.org/10.1063/1.1604371)

139. Ö. Farkas and H.B. Schlegel, *Methods for optimizing large molecules. Part III. An improved algorithm for geometry optimization using direct inversion in the iterative subspace (GDIIS)*, Physical Chemistry Chemical Physics 4, 11 (2002) (https://doi.org/10.1039/B108658H)

140. I. Mayer, *Charge, bond order and valence in the ab inition SCF theory*, Chemical Physics Letters 97, 270 (1983) (https://doi.org/10.1016/0009-2614(83)80005-0)

141. L. Jensen, P.T. van Duijnen and J.G. Snijders, *A discrete solvent reaction field model within density functional theory*, Journal of Chemical Physics 118, 514 (2003) (https://doi.org/10.1063/1.1527010)

142. L. Jensen, P.T. van Duijnen and J.G. Snijders, *A discrete solvent reaction field model for calculating molecular linear response properties in solution*, Journal of Chemical Physics 119, 3800 (2003) (https://doi.org/10.1063/1.1590643)

143. L. Jensen, P.T. van Duijnen and J.G. Snijders, *A discrete solvent reaction field model for calculating frequency-dependent hyperpolarizabilities of molecules in solution*, Journal of Chemical Physics 119, 12998 (2003) (https://doi.org/10.1063/1.1627760)

144. L. Jensen, M. Swart and P.T. van Duijnen, *Microscopic and macroscopic polarization within a combined quantum mechanics and molecular mechanics model*, Journal of Chemical Physics 122, 34103 (2005) (https://doi.org/10.1063/1.1831271)

145. L. Jensen, Modelling of optical response properties: Application to nanostructures, (http://www.scm.com/Doc/jensen.pdf) PhD thesis, Rijksuniversiteit Groningen, 2004.

146. P.T. van Duijnen and M. Swart, *Molecular and Atomic Polarizabilities: Thole's Model Revisited*, Journal of Physical Chemistry A 102, 2399 (1998) (https://doi.org/10.1021/jp980221f)

147. L. Jensen, P.-O. Astrand, A. Osted, J. Kongsted and K.V. Mikkelsen, *Polarizability of molecular clusters as calculated by a dipole interaction model*, Journal of Chemical Physics 116, 4001 (2002) (https://doi.org/10.1063/1.1433747)

148. A. Michalak, R.L. De Kock and T. Ziegler, *Bond Multiplicity in Transition-Metal Complexes: Applications of Two-Electron Valence Indices*, Journal of Physical Chemistry A 112, 7256 (2008) (https://doi.org/10.1021/jp800139g)

149. R.F. Nalewajski and J. Mrozek, *Modified valence indices from the two-particle density matrix*, International Journal of Quantum Chemistry 51, 187 (1994) (https://doi.org/10.1002/qua.560510403)

150. R.F. Nalewajski, J. Mrozek and A. Michalak, *Two-electron valence indices from the Kohn-Sham orbitals*, International Journal of Quantum Chemistry 61, 589 (1997) (https://doi.org/10.1002/(SICI)1097-461X(1997)61:3%3C589::AID-QUA28%3E3.0.CO;2-2)

151. R.F. Nalewajski, J. Mrozek and A. Michalak, *Exploring Bonding Patterns of Molecular Systems Using Quantum Mechanical Bond Multiplicities*, Polish Journal of Chemistry 72, 1779 (1998) (http://ichf.edu.pl/pjch/pj-1998/pj07s98.htm#1779)

152. R.F. Nalewajski, J. Mrozek and G. Mazur, *Quantum chemical valence indices from the one-determinantal difference approach*, Canadian Journal of Chemistry 74, 1121 (1996) (https://doi.org/10.1139/v96-126)

153. M.S. Gopinathan and K. Jug, *Valency. I. A quantum chemical definition and properties*, Theoretica Chimica Acta 1983 63, 497 (1983) (https://doi.org/10.1007/BF00552652)

154. F. Wang and T. Ziegler, *Excitation energies of some d1 systems calculated using time-dependent density functional theory: an implementation of open-shell TDDFT theory for doublet-doublet excitations*, Molecular Physics 102, 2585 (2004) (https://doi.org/10.1080/0026897042000275080)

155. Z. Rinkevicius, I. Tunell, P. Salek, O. Vahtras and H. Agren, *Restricted density functional theory of linear time-dependent properties in open-shell molecules*, Journal of Chemical Physics 119, 34 (2003) (https://doi.org/10.1063/1.1577329)

156. F. Wang and T. Ziegler, *Time-dependent density functional theory based on a noncollinear formulation of the exchange-correlation potential*, Journal of Chemical Physics 121, 12191 (2004) (https://doi.org/10.1063/1.1821494)

157. F. Wang and T. Ziegler, *The performance of time-dependent density functional theory based on a noncollinear exchange-correlation potential in the calculations of excitation energies*, Journal of Chemical Physics 122, 74109 (2005) (https://doi.org/10.1063/1.1844299)

158. S. Hirata and M. Head-Gordon, *Time-dependent density functional theory within the Tamm-Dancoff approximation*, Chemical Physics Letters 314, 291 (1999) (https://doi.org/10.1016/S0009-2614(99)01149-5)

159. G. Henkelman, B.P. Uberuaga and H. Jonssón, *A climbing image nudged elastic band method for finding saddle points and minimum energy paths*, Journal of Chemical Physics 113, 9901 (2000) (https://doi.org/10.1063/1.1329672)

160. G. Vignale and W. Kohn, *Current-Dependent Exchange-Correlation Potential for Dynamical Linear Response Theory*, Physical Review Letters 77, 2037 (1996) (https://doi.org/10.1103/PhysRevLett.77.2037)

161. G. Vignale and W. Kohn, in *Electronic Density Functional Theory: Recent Progress and New Direction*, ISBN13: 9780306458347, J. F. Dobson, G. Vignale, and M. P. Das, Editors. 1998, Plenum: New York.

162. M. van Faassen, P. L. de Boeij, R. van Leeuwen, J. A. Berger and J. G. Snijders, *Ultranonlocality in Time-Dependent Current-Density-Functional Theory: Application to Conjugated Polymers*, Physical Review Letters 88, 186401 (2002) (https://doi.org/10.1103/PhysRevLett.88.186401)

163. M. van Faassen, P. L. de Boeij, R. van Leeuwen, J. A. Berger and J. G. Snijders, *Application of time-dependent current-density-functional theory to nonlocal exchange-correlation effects in polymers*, Journal of Chemical Physics 118, 1044 (2003) (https://doi.org/10.1063/1.1529679)

164. M. van Faassen and P. L. de Boeij, *Excitation energies for a benchmark set of molecules obtained within time-dependent current-density functional theory using the Vignale-Kohn functional*, Journal of Chemical Physics 120, 8353 (2004) (https://doi.org/10.1063/1.1697372)

165. M. van Faassen and P. L. de Boeij, *Excitation energies of Π -conjugated oligomers within time-dependent current-density-functional theory*, Journal of Chemical Physics 121, 10707 (2004) (https://doi.org/10.1063/1.1810137)

166. M. van Faassen, Time-Dependent Current-Density-Functional Theory for Molecules, (http://www.scm.com/Doc/faassen.pdf) PhD thesis, Rijksuniversiteit Groningen, 2004.

167. R. Nifosi, S. Conti and M. P. Tosi, *Dynamic exchange-correlation potentials for the electron gas in dimensionality D=3 and D=2*, Physical Review B 58: p. 12758 (1998) (https://doi.org/10.1103/PhysRevB.58.12758)

168. Z.X. Qian and G. Vignale, *Dynamical exchange-correlation potentials for the electron liquid in the spin channel*, Physical Review B 68, 195113 (2003) (https://doi.org/10.1103/PhysRevB.68.195113)

169. M. Stener, G. Fronzoni and M. de Simone, *Time dependent density functional theory of core electrons excitations*, Chemical Physics Letters 373, 115 (2003) (https://doi.org/10.1016/S0009-2614(03)00543-8)

170. M. Swart, P.Th. van Duijnen and J.G. Snijders, *A charge analysis derived from an atomic multipole expansion*, Journal of Computational Chemistry 22, 79 (2001) (https://doi.org/10.1002/1096-987X(20010115)22:1%3C79::AID-JCC8%3E3.0.CO;2-B)

171. T.W. Keal and D.J. Tozer, *The exchange-correlation potential in Kohn.Sham nuclear magnetic resonance shielding calculations*, Journal of Chemical Physics 119, 3015 (2003) (https://doi.org/10.1063/1.1590634)

172. X. Xu and W.A. Goddard III, *The X3LYP extended density functional for accurate descriptions of nonbond interactions, spin states, and thermochemical properties*, Proceedings of the National Academy of Sciences 101, 2673 (2004) (https://doi.org/10.1073/pnas.0308730100)

173. J. Baker, A. Kessi and B. Delley, *The generation and use of delocalized internal coordinates in geometry optimization*, Journal of Chemical Physics 1996 105, 192 (1996) (https://doi.org/10.1063/1.471864)

174. C. Adamo and V. Barone, *Physically motivated density functionals with improved performances: The modified Perdew.Burke.Ernzerhof model*, Journal of Chemical Physics 1996 116, 5933 (1996) (https://doi.org/10.1063/1.1458927)

175. M. Swart, A.W. Ehlers and K. Lammertsma, *Performance of the OPBE exchange-correlation functional*, Molecular Physics 2004 102, 2467 (2004) (https://doi.org/10.1080/0026897042000275017)

176. P.J. Stephens, F.J. Devlin, C.F. Chabalowski and M.J. Frisch, *Ab Initio Calculation of Vibrational Absorption and Circular Dichroism Spectra Using Density Functional Force Fields*, Journal of Physical Chemistry 98, 11623 (1994) (https://doi.org/10.1021/j100096a001)

177. M. Reiher, O. Salomon and B.A. Hess, *Reparameterization of hybrid functionals based on energy differences of states of different multiplicity*, Theoretical Chemistry Accounts 107, 48 (2001) (https://doi.org/10.1007/s00214-001-0300-3)

178. C. Adamo and V. Barone, *Toward reliable adiabatic connection models free from adjustable parameters*, Chemical Physics Letters 274, 242 (1997) (https://doi.org/10.1016/S0009-2614(97)00651-9)

179. J.K. Kang and C.B. Musgrave, *Prediction of transition state barriers and enthalpies of reaction by a new hybrid density-functional approximation*, Journal of Chemical Physics 115, 11040 (2001) (https://doi.org/10.1063/1.1415079)

180. A.J. Cohen and N.C. Handy, *Dynamic correlation*, Molecular Physics 99, 607 (2001) (https://doi.org/10.1080/00268970010023435)

181. B.J. Lynch, P.L. Fast, M. Harris and D.G. Truhlar, *Adiabatic Connection for Kinetics*, Journal of Physical Chemistry A 104, 4811 (2000) (https://doi.org/10.1021/jp000497z)

182. F. Wang, T. Ziegler, E. van Lenthe, S.J.A. van Gisbergen and E.J. Baerends, *The calculation of excitation energies based on the relativistic two-component zeroth-order regular approximation and time-dependent density-functional with full use of symmetry*, Journal of Chemical Physics 122, 204103 (2005) (https://doi.org/10.1063/1.1899143)

183. F. Wang and T. Ziegler, *Theoretical study of the electronic spectra of square-planar platinum (II) complexes based on the two-component relativistic time-dependent density-functional theory*, Journal of Chemical Physics 123, 194102 (2005) (https://doi.org/10.1063/1.2104427)

184. T.A. Wesolowski and A. Warshel, *Frozen Density Functional Approach for ab-initio Calculations of Solvated Molecules*, Journal of Physical Chemistry 97, 8050 (1993) (https://doi.org/10.1021/j100132a040)

185. J. Neugebauer, C.R. Jacob, T.A. Wesolowski and E.J. Baerends, *An Explicit Quantum Chemical Method for Modeling Large Solvation Shells Applied to Aminocoumarin C151*, Journal of Physical Chemistry A 109, 7805 (2005) (https://doi.org/10.1021/jp0528764)

186. M.E. Casida and T.A. Wesolowski, *Generalization of the Kohn-Sham equations with constrained electron density formalism and its time-dependent response theory formulation*, International Journal of Quantum Chemistry 96, 577 (2004) (https://doi.org/10.1002/qua.10744)

187. T.A. Wesolowski, *Hydrogen-Bonding-Induced Shifts of the Excitation Energies in Nucleic Acid Bases: An Interplay between Electrostatic and Electron Density Overlap Effects*, Journal of the American Chemical Society 126, 11444 (2004) (https://doi.org/10.1021/ja048846g)

188. T.A. Wesolowski, *Density functional theory with approximate kinetic energy functionals applied to hydrogen bonds*, Journal of Chemical Physics 106, 8516 (1997) (https://doi.org/10.1063/1.473907)

189. C.R. Jacob, T.A. Wesolowski and L. Visscher, *Orbital-free embedding applied to the calculation of induced dipole moments in* $CO2^{...}$ *X (X=He, Ne, Ar, Kr, Xe, Hg) van der Waals complexes*, Journal of Chemical Physics 123, 174104 (2005) (https://doi.org/10.1063/1.2107567)

190. C.R. Jacob, J. Neugebauer, L. Jensen and L. Visscher, *Comparison of frozen-density embedding and discrete reaction field solvent models for molecular properties*, Physical Chemistry Chemical Physics 8, 2349 (2006) (https://doi.org/10.1039/B601997H)

191. J. Neugebauer, M.J. Louwerse, E.J. Baerends and T.A. Wesolowski, *The merits of the frozen-density embedding scheme to model solvatochromic shifts*, Journal of Chemical Physics 122, 94115 (2005) (https://doi.org/10.1063/1.1858411)

192. J. Neugebauer, M.J. Louwerse, P. Belanzoni, T.A. Wesolowski and E.J. Baerends, *Modeling solvent effects on electron-spin-resonance hyperfine couplings by frozen-density embedding*, Journal of Chemical Physics 123, 114101 (2005) (https://doi.org/10.1063/1.2033749)

193. J. Neugebauer and E.J. Baerends, *Exploring the Ability of Frozen-Density Embedding to Model Induced Circular Dichroism*, Journal of Physical Chemistry A 110, 8786 (2006) (https://doi.org/10.1021/jp0622280)

194. L.H. Thomas, *The calculation of atomic fields*, Mathematical Proceedings of the Cambridge Philosophical Society 23, 542 (1927) (https://doi.org/10.1017/S0305004100011683)

195. E. Fermi, *Eine statistische Methode zur Bestimmung einiger Eigenschaften des Atoms und ihre Anwendung auf die Theorie des periodischen Systems der Elemente*, Zeitschrift für Physik 48, 73 (1928) (https://doi.org/10.1007/BF01351576)

196. C.F. von Weizsäcker, *Zur Theorie der Kernmassen*, Zeitschrift für Physik 96, 431 (1935) (https://doi.org/10.1007/BF01337700)

197. A. Lembarki and H. Chermette, *Obtaining a gradient-corrected kinetic-energy functional from the Perdew-Wang exchange functional*, Physical Review A 50, 5328 (1994) (https://doi.org/10.1103/PhysRevA.50.5328)

198. H. Lee, C. Lee and R.G. Parr, *Conjoint gradient correction to the Hartree-Fock kinetic- and exchange-energy density functionals*, Physical Review A 44, 768 (1991) (https://doi.org/10.1103/PhysRevA.44.768)

199. J.P. Perdew and Wang Yue, *Accurate and simple density functional for the electronic exchange energy: Generalized gradient approximation*, Physical Review B 33, 8800 (1986) (https://doi.org/10.1103/PhysRevB.33.8800), Erratum Physical Review B 40, 3399 (1989) (https://doi.org/10.1103/PhysRevB.40.3399)

200. H. Ou-Yang and M. Levy, *Approximate noninteracting kinetic energy functionals from a nonuniform scaling requirement*, International Journal of Quantum Chemistry 40, 379 (1991) (https://doi.org/10.1002/qua.560400309)

201. A.J. Thakkar, *Comparison of kinetic-energy density functionals*, Physical Review A 46, 6920 (1992) (https://doi.org/10.1103/PhysRevA.46.6920)

202. J. Neugebauer, E.J. Baerends, E. Efremov, F. Ariese and C. Gooijer, *Combined Theoretical and Experimental Deep-UV Resonance Raman Studies of Substituted Pyrenes*, Journal of Physical Chemistry A 109, 2100 (2005) (https://doi.org/10.1021/jp045360d)

203. J. Neugebauer, E.J. Baerends and M. Nooijen, *Vibronic coupling and double excitations in linear response time-dependent density functional calculations: Dipole-allowed states of* N_2 , Journal of Chemical Physics 121, 6155 (2004) (https://doi.org/10.1063/1.1785775)

204. J. Neugebauer, Vibronic Coupling Calculations using ADF, documentation on the VIBRON module available on request.

205. T.A. Wesolowski, in: Computational Chemistry: Reviews of Current Trends - Vol. 10, World Scientific, 2006.

206. M. Zbiri, M. Atanasov, C. Daul, J.-M. Garcia Lastra and T.A. Wesolowski, *Application of the density functional theory derived orbital-free embedding potential to calculate the splitting energies of lanthanide cations in chloroelpasolite crystals*, Chemical Physics Letters 397, 441 (2004) (https://doi.org/10.1016/j.cplett.2004.09.010)

207. M. Zbiri, C.A. Daul and T.A. Wesolowski, *Effect of the f-Orbital Delocalization on the Ligand-Field Splitting Energies in Lanthanide-Containing Elpasolites*, Journal of Chemical Theory and Computation 2, 1106 (2006) (https://doi.org/10.1021/ct060035a)

208. A. Bérces, R. M. Dickson, L. Fan, H. Jacobsen, D. Swerhone and T. Ziegler, *An implementation of the coupled perturbed Kohn-Sham equations: perturbation due to nuclear displacements*, Computer Physics Communications 100, 247 (1997) (https://doi.org/10.1016/S0010-4655(96)00120-8)

209. H. Jacobsen, A. Bérces, D. Swerhone and T. Ziegler, *Analytic second derivatives of molecular energies: a density functional implementation*, Computer Physics Communications 100, 263 (1997) (https://doi.org/10.1016/S0010-4655(96)00119-1)

210. S.K. Wolff, *Analytical second derivatives in the Amsterdam density functional package*, International Journal of Quantum Chemistry 104, 645 (2005) (https://doi.org/10.1002/qua.20653)

211. S. Grimme, *Accurate description of van der Waals complexes by density functional theory including empirical corrections*, Journal of Computational Chemistry 25, 1463 (2004) (https://doi.org/10.1002/jcc.20078)

212. M. Ernzerhof and G. Scuseria, *Assessment of the Perdew.Burke.Ernzerhof exchange-correlation functional*, Journal of Chemical Physics 110, 5029 (1999) (https://doi.org/10.1063/1.478401)

213. C. Adamo and V. Barone, *Toward reliable density functional methods without adjustable parameters: The PBE0 model*, Journal of Chemical Physics 110, 6158 (1999) (https://doi.org/10.1063/1.478522)

214. A.D. Buckingham, P.W. Fowler and J.M. Hutson, *Theoretical studies of van der Waals molecules and intermolecular forces*, Chemical Reviews 88, 963 (1988) (https://doi.org/10.1021/cr00088a008)

215. J.M. Ducéré and L. Cavallo, *Parametrization of an Empirical Correction Term to Density Functional Theory for an Accurate Description of Π -Stacking Interactions in Nucleic Acids*, Journal of Physical Chemistry B 111, 13124 (2007) (https://doi.org/10.1021/jp072317s)

216. V.P. Nicu J. Neugebauer S.K. Wolff and E.J. Baerends, *A vibrational circular dichroism implementation within a Slater-type-orbital based density functional framework and its application to hexa- and hepta-helicenes*, Theoretical Chemical Accounts 119, 245 (2008) (https://doi.org/10.1007/s00214-006-0234-x)

217. C.R. Jacob, J. Neugebauer and L. Visscher, *A flexible implementation of frozen-density embedding for use in multilevel simulations*, Journal of Computational Chemistry 29, 1011 (2008) (https://doi.org/10.1002/jcc.20861)

218. C.R. Jacob and L. Visscher, *Calculation of nuclear magnetic resonance shieldings using frozen-density embedding*, Journal of Chemical Physics 125, 194104 (2006) (https://doi.org/10.1063/1.2370947)

219. V. Bakken and T. Helgaker, *The efficient optimization of molecular geometries using redundant internal coordinates*, Journal of Chemical Physics 117, 9160 (2002) (https://doi.org/10.1063/1.1515483)

220. C.R. Jacob, S.M., Beyhan and L. Visscher, *Exact functional derivative of the nonadditive kinetic-energy bifunctional in the long-distance limit*, Journal of Chemical Physics 126, 234116 (2007) (https://doi.org/10.1063/1.2743013)

221. Y. Zhao, N.E. Schultz and D.G. Truhlar, *Exchange-correlation functional with broad accuracy for metallic and nonmetallic compounds, kinetics, and noncovalent interactions*, Journal of Chemical Physics 123, 161103 (2005) (https://doi.org/10.1063/1.2126975)

222. Y. Zhao, N.E. Schultz and D.G. Truhlar, *Design of Density Functionals by Combining the Method of Constraint Satisfaction with Parametrization for Thermochemistry, Thermochemical Kinetics, and Noncovalent Interactions*, Journal of Chemical Theory and Computation 2, 364 (2006) (https://doi.org/10.1021/ct0502763)

223. Y. Zhao and D.G. Truhlar, *A new local density functional for main-group thermochemistry, transition metal bonding, thermochemical kinetics, and noncovalent interactions*, Journal of Chemical Physics 125, 194101 (2006) (https://doi.org/10.1063/1.2370993)

224. Y. Zhao and D.G. Truhlar, *The M06 suite of density functionals for main group thermochemistry, thermochemical kinetics, noncovalent interactions, excited states, and transition elements: two new functionals and systematic testing of four M06-class functionals and 12 other functionals*, Theoretical Chemical Accounts 120, 215 (2008) (https://doi.org/10.1007/s00214-007-0310-x)

225. M. Swart and F.M. Bickelhaupt, *Optimization of strong and weak coordinates*, International Journal of Quantum Chemistry 106, 2536 (2006) (https://doi.org/10.1002/qua.21049)

226. S. Grimme, *Semiempirical GGA-Type Density Functional Constructed with a Long-Range Dispersion Correction*, Journal of Computational Chemistry 27, 1787 (2006) (https://doi.org/10.1002/jcc.20495)

227. S. Grimme, , J. Antony, T. Schwabe and C. Mück-Lichtenfeld, *Density Functional Theory with Dispersion Corrections for Supramolecular Structures, Aggregates, and Complexes of (Bio)Organic Molecules*, Organic & Biomolecular Chemistry 5, 741 (2007) (https://doi.org/10.1039/B615319B)

228. J.I. Rodríguez, A.M. Köster, P.W. Ayers, A. Santos-Valle, A. Vela and G. Merino, *An efficient grid-based scheme to compute QTAIM atomic properties without explicit calculation of zero-flux surfaces*, Journal of Computational Chemistry 30, 1082 (2009) (https://doi.org/10.1002/jcc.21134)

229. J.I. Rodríguez, R.F.W. Bader, P.W. Ayers, C. Michel, A.W. Götz and C. Bo, *A high performance grid-based algorithm for computing QTAIM properties*, Chemical Physics Letters 472, 149 (2009) (https://doi.org/10.1016/j.cplett.2009.02.081)

230. M. Krykunov and J. Autschbach, *Calculation of static and dynamic linear magnetic response in approximate time-dependent density functional theory*, Journal of Chemical Physics 126, 24101 (2007) (https://doi.org/10.1063/1.2423007)

231. M. Krykunov, M.D. Kundrat and J. Autschbach, *Calculation of CD spectra from optical rotatory dispersion, and vice versa, as complementary tools for theoretical studies of optical activity using time-dependent density functional theory*, Journal of Chemical Physics 125, 194110 (2006) (https://doi.org/10.1063/1.2363372)

232. M. Krykunov and J. Autschbach, *Calculation of origin independent optical rotation tensor components for chiral oriented systems in approximate time-dependent density functional theory*, Journal of Chemical Physics 125, 34102 (2006) (https://doi.org/10.1063/1.2210474)

233. J. Autschbach, L. Jensen, G.C. Schatz, Y.C.E. Tse and M. Krykunov, *Time-dependent density functional calculations of optical rotatory dispersion including resonance wavelengths as a potentially useful tool for determining absolute configurations of chiral molecules*, Journal of Physical Chemistry A 110, 2461 (2006) (https://doi.org/10.1021/jp054847z)

234. M. Krykunov and J. Autschbach, *Calculation of optical rotation with time-periodic magnetic field-dependent basis functions in approximate time-dependent density functional theory*, Journal of Chemical Physics 123, 114103 (2005) (https://doi.org/10.1063/1.2032428)

235. A. Baev, M. Samoc, P.N. Prasad, M. Krykunov and J. Autschbach, *A Quantum Chemical Approach to the Design of Chiral Negative Index Materials*, Optics Express 15, 5730 (2007) (https://doi.org/10.1364/OE.15.005730)

236. M. Krykunov, A. Banerjee, T. Ziegler and J. Autschbach, *Calculation of Verdet constants with time-dependent density functional theory: Implementation and results for small molecules*, Journal of Chemical Physics 122, 74105 (2005) (https://doi.org/10.1063/1.1850919)

237. P. Cortona, *Self-consistently determined properties of solids without band-structure calculations*, Physical Review B 44, 8454 (1991) (https://doi.org/10.1103/PhysRevB.44.8454)

238. T.A. Wesolowski and J. Weber, *Kohn-Sham equations with constrained electron density: The effect of various kinetic energy functional parametrizations on the ground-state molecular properties*, International Jour-

nal of Quantum Chemistry 61, 303 (1997) (https://doi.org/10.1002/(SICI)1097-461X(1997)61:2%3C303::AID-QUA13%3E3.0.CO;2-C)

239. T.A. Wesolowski, H. Chermette and J. Weber, *Accuracy of Approximate Kinetic Energy Functionals in the Model of Kohn-Sham Equations with Constrained Electron Density: the* FH$^{..}$ NCH *complex as a Test Case*, Journal of Chemical Physics 105, 9182 (1996) (https://doi.org/10.1063/1.472823)

240. T.A. Wesolowski and J. Weber, *Kohn-Sham equations with constrained electron density: an iterative evaluation of the ground-state electron density of interacting molecules*, Chemical Physics Letters 248, 71 (1996) (https://doi.org/10.1016/0009-2614(95)01281-8)

241. Y.A. Bernard, M. Dulak, J.W. Kaminski and T.A. Wesolowski, *The energy-differences based exact criterion for testing approximations to the functional for the kinetic energy of non-interacting electrons*, Journal of Physics A 41, 55302 (2008) (https://doi.org/10.1088/1751-8113/41/5/055302)

242. D.A. Kirzhnits, Soviet Physics JETP-USSR 5, 64 (1957)

243. P. Fuentealba and O. Reyes, *Further evidence of the conjoint correction to the local kinetic and exchange energy density functionals*, Chemical Physics Letters 232, 31 (1995) (https://doi.org/10.1016/0009-2614(94)01321-L)

244. O.V. Gritsenko, P.R.T. Schipper and E.J. Baerends, *Approximation of the exchange-correlation Kohn-Sham potential with a statistical average of different orbital model potentials*, Chemical Physics Letters 302, 199 (1999) (https://doi.org/10.1016/S0009-2614(99)00128-1)

245. B. Delley, *The conductor-like screening model for polymers and surfaces*, Molecular Simulation 32, 117 (2006) (https://doi.org/10.1080/08927020600589684)

246. J. Tao, J.P. Perdew, V.N. Staroverov and G.E. Scuseria, *Climbing the Density Functional Ladder: Nonempirical MetaGeneralized Gradient Approximation Designed for Molecules and Solids* Physical Review Letters 91, 146401 (2003) (https://doi.org/10.1103/PhysRevLett.91.146401)

247. V.N. Staroverov, G.E. Scuseria, J. Tao and J.P. Perdew, *Comparative assessment of a new nonempirical density functional: Molecules and hydrogen-bonded complexes* Journal of Chemical Physics 119, 12129 (2003) (https://doi.org/10.1063/1.1626543)

248. S. Ivanov, S. Hirata, R. J. Bartlett, *Exact Exchange Treatment for Molecules in Finite-Basis-Set Kohn-Sham Theory*, Physical Review Letters 83, 5455 (1999) (https://doi.org/10.1103/PhysRevLett.83.5455)

249. A.F. Izmaylov, V.N. Staroverov, G.E. Scuseria, E.R. Davidson, G. Stoltz, E. Cancès, *The effective local potential method: Implementation for molecules and relation to approximate optimized effective potential techniques*, Journal of Chemical Physics 126, 084107 (2007) (https://doi.org/10.1063/1.2434784)

250. M. Krykunov and T. Ziegler, *On the use of the exact exchange optimized effective potential method for static response properties*, International Journal of Quantum Chemistry 109, 3246 (2009) (https://doi.org/10.1002/qua.21937)

251. M.L. Connolly, *Solvent-accessible surfaces of proteins and nucleic acids*, Science, 221, 709 (1983) (https://doi.org/10.1126/science.6879170)

252. L. Onsager, *Electric moments of molecules in liquids*, Journal of the American Chemical Society 58, 1486 (1936) (https://doi.org/10.1021/ja01299a050)

253. S. Miertus, E. Scrocco and J. Tomasi, *Electrostatic interaction of a solute with a continuum: a direct utilization of ab initio molecular potentials for the prevision of solvent effects*, Chemical Physics 55, 117 (1981) (https://doi.org/10.1016/0301-0104(81)85090-2)

254. J. Tomasi, R. Bonaccorsi, R. Cammi and F.J. Olivares del Valle, *Theoretical chemistry in solution. Some results and perspectives of the continuum methods and in particular of the polarizable continuum model*, Journal of Molecular Structure: THEOCHEM 234, 401 (1991) (https://doi.org/10.1016/0166-1280(91)89026-W)

255. J.L. Chen, L. Noodleman, D.A. Case and D. Bashford, *Incorporating solvation effects into density functional electronic structure calculations*, Journal of Physical Chemistry 98, 11059 (1994) (https://doi.org/10.1021/j100094a013)

256. J.-M. Mouesca, J.L. Chen, L. Noodleman, D. Bashford and D.A. Case, *Density functional/Poisson-Boltzmann calculations of redox potentials for iron-sulfur clusters*, Journal of the American Chemical Society 116, 11898 (1994) (https://doi.org/10.1021/ja00105a033)

257. A. Fortunelli and J. Tomasi, *The implementation of density functional theory within the polarizable continuum model for solvation*, Chemical Physics Letters 231, 34 (1994) (https://doi.org/10.1016/0009-2614(94)01253-9)

258. C.M. Breneman and K.B. Wiberg, *Determining atom-centered monopoles from molecular electrostatic potentials. the need for high sampling density in formamide conformational analysis*, Journal of Computational Chemistry 11, 361 (1990) (https://doi.org/10.1002/jcc.540110311)

259. F.M. Richards, *Areas, volumes, packing and protein structures*, Annual Review of Biophysics and Bioengineering 6, 151 (1977) (https://doi.org/10.1146/annurev.bb.06.060177.001055)

260. T. You and D. Bashford, *An analytical algorithm for the rapid determination of the solvent accessibility of points in a three-dimensional lattice around a solute molecule*, Journal of Computational Chemistry 16, 743 (1995) (https://doi.org/10.1002/jcc.540160610)

261. M. Mitoraj, A. Michalak and T. Ziegler, *A Combined Charge and Energy Decomposition Scheme for Bond Analysis*, Journal of Chemical Theory and Computation 5, 962 (2009) (https://doi.org/10.1021/ct800503d)

262. M. Mitoraj, A. Michalak and T. Ziegler, *On the Nature of the Agostic Bond between Metal Centers and Beta-Hydrogen Atoms in Alkyl Complexes. An Analysis Based on the Extended Transition State Method and the Natural Orbitals for Chemical Valence Scheme (ETS-NOCV)*, Organometallics 28, 3727 (2009) (https://doi.org/10.1021/om900203m)

263. A.W. Götz, S.M. Beyhan and L. Visscher, *Performance of Kinetic Energy Functionals for Interaction Energies in a Subsystem Formulation of Density Functional Theory*, Journal of Chemical Theory and Computation 5, 3161 (2009) (https://doi.org/10.1021/ct9001784)

264. M. Ernzerhof, *The role of the kinetic energy density in approximations to the exchange energy*, Journal of Molecular Structure: THEOCHEM 501-502, 59 (2000) (https://doi.org/10.1016/S0166-1280(99)00414-5)

265. J.P. Perdew, *Generalized gradient approximation for the fermion kinetic energy as a functional of the density*, Physics Letters A 165, 79 (1992) (https://doi.org/10.1016/0375-9601(92)91058-Y)

266. L. Jensen, L. Zhao, J. Autschbach and G.C. Schatz, *Theory and method for calculating resonance Raman scattering from resonance polarizability derivatives*, Journal of Chemical Physics 123, 174110 (2005) (https://doi.org/10.1063/1.2046670)

267. L. Jensen, L. Zhao, J. Autschbach and G.C. Schatz, *Resonance Raman Scattering of Rhodamine 6G as calculated using Time-Dependent Density Functional Theory*, Journal of Physical Chemistry A 110, 5973 (2006) (https://doi.org/10.1021/jp0610867)

268. L.L. Zhao, L. Jensen and G.C. Schatz, Pyridine - Ag_{20} *Cluster: A Model System for Studying Surface-Enhanced Raman Scattering*, Journal of the American Chemical Society 128, 2911 (2006) (https://doi.org/10.1021/ja0556326)

269. L. Jensen, L.L. Zhao and G.C. Schatz, *Size-Dependence of the Enhanced Raman Scattering of Pyridine Adsorbed on Ag_n (n=2-8,20) Clusters*, Journal of Physical Chemistry C 111, 4756 (2007) (https://doi.org/10.1021/jp067634y)

270. J. Autschbach, *Magnitude of Finite-Nucleus-Size Effects in Relativistic Density Functional Computations of Indirect NMR Nuclear Spin-Spin Coupling Constants*, ChemPhysChem 10, 2274 (2009) (https://doi.org/10.1002/cphc.200900271)

271. S. Høst, J. Olsen, B. Jansík, L. Thøgersen, P. Jørgensen and T. Helgaker, *The augmented Roothaan-Hall method for optimizing Hartree-Fock and Kohn-Sham density matrices*, Journal of Chemical Physics 129, 124106 (2008) (https://doi.org/10.1063/1.2974099)

272. M. Krykunov, M. Seth, T. Ziegler and J. Autschbach, *Calculation of the magnetic circular dichroism B term from the imaginary part of the Verdet constant using damped time-dependent density functional theory*, Journal of Chemical Physics 127, 244102 (2007) (https://doi.org/10.1063/1.2806990)

273. S.B. Piepho and P. N. Schatz, *Group Theory in Spectroscopy With Application to Magnetic Circular Dichroism*, (Wiley, New York, 1983).

274. W.R. Mason, *A Practical Guide to Magnetic Circular Dichroism Spectroscopy*, (Wiley, New Jersey, 2007).

275. M. Seth and T. Ziegler, *Formulation of magnetically perturbed time-dependent density functional theory*, Journal of Chemical Physics 127, 134108 (2007) (https://doi.org/10.1063/1.2772849)

276. M. Seth, M. Krykunov, T. Ziegler, J. Autschbach and A. Banerjee, *Application of magnetically perturbed time-dependent density functional theory to magnetic circular dichroism: Calculation of B terms*, Journal of Chemical Physics 128, 144105 (2008) (https://doi.org/10.1063/1.2901967)

277. M. Seth, M. Krykunov, T. Ziegler and J. Autschbach, *Application of magnetically perturbed time-dependent density functional theory to magnetic circular dichroism. II. Calculation of A terms*, Journal of Chemical Physics 128, 234102 (2008) (https://doi.org/10.1063/1.2933550)

278. M. Seth, T. Ziegler and J. Autschbach, *Application of magnetically perturbed time-dependent density functional theory to magnetic circular dichroism. III. Temperature-dependent magnetic circular dichroism induced by spin-orbit coupling*, Journal of Chemical Physics 129, 104105 (2008) (https://doi.org/10.1063/1.2976568)

279. J.M. Garcia Lastra, J.W. Kaminski and T.A. Wesolowski, *Orbital-free effective embedding potential at nuclear cusps*, Journal of Chemical Physics 129, 074107 (2008) (https://doi.org/10.1063/1.2969814)

280. F. Wang and T. Ziegler, *A simplified relativistic time-dependent density-functional theory formalism for the calculations of excitation energies including spin-orbit coupling effect*, Journal of Chemical Physics 123, 154102 (2005) (https://doi.org/10.1063/1.2061187)

281. W.-G. Han, T. Liu, T. Lovell and L. Noodleman, *DFT calculations of [57] Fe Mössbauer isomer shifts and quadrupole splittings for iron complexes in polar dielectric media: Applications to methane monooxygenase and ribonucleotide reductase*, Journal of Computational Chemistry 27, 1292 (2006) (https://doi.org/10.1002/jcc.20402)

282. A. Ghysels, D. Van Neck, V. Van Speybroeck, T. Verstraelen and M. Waroquier, *Vibrational Modes in partially optimized molecular systems*, Journal of Chemical Physics126, 224102 (2007) (https://doi.org/10.1063/1.2737444)

283. A. Ghysels, D. Van Neck and M. Waroquier, *Cartesian formulation of the Mobile Block Hessian Approach to vibrational analysis in partially optimized systems*, Journal of Chemical Physics 127, 164108 (2007) (https://doi.org/10.1063/1.2789429)

284. J.J. Mortensen, K. Kaasbjerg, S.L. Frederiksen, J.K. Nørskov, J.P. Sethna, and K.W. Jacobsen, *Bayesian Error Estimation in Density-Functional Theory*, Physical Review Letters 95, 216401 (2005) (https://doi.org/10.1103/PhysRevLett.95.216401)

285. J.P. Perdew, A. Ruzsinszky, G.I. Csonka, O.A. Vydrov, G.E. Scuseria, *Restoring the Density-Gradient Expansion for Exchange in Solids and Surfaces*, Physical Review Letters 100, 136406 (2008) (https://doi.org/10.1103/PhysRevLett.100.136406)

286. M. Swart, M. Solà and F.M. Bickelhaupt, *A new all-round DFT functional based on spin states and SN2 barriers*, Journal of Chemical Physics 131, 094103 (2009) (https://doi.org/10.1063/1.3213193)

287. M. Swart, M. Solà and F.M. Bickelhaupt, *Switching between OPTX and PBE exchange functionals*, Journal of Computational Methods in Science and Engineering 9, 69 (2009) (https://doi.org/10.3233/JCM-2009-0230)

288. J.P. Perdew and Y. Wang, *Accurate and simple analytic representation of the electron-gas correlation energy*, Physical Review B 45, 13244 (1992) (https://doi.org/10.1103/PhysRevB.45.13244)

289. K.N. Kudin, G.E. Scuseria and E. Cances, *A black-box self-consistent field convergence algorithm: One step closer*, Journal of Chemical Physics 116, 8255 (2002) (https://doi.org/10.1063/1.1470195)

290. N.L. Allinger, X. Zhou, J. Bergsma, *Molecular mechanics parameters*, Journal of Molecular Structure: THEOCHEM 312, 69 (1994) (https://doi.org/10.1016/S0166-1280(09)80008-0)

291. X. Hu and W. Yang, *Accelerating self-consistent field convergence with the augmented Roothaan-Hall energy function*, Journal of Chemical Physics 132, 054109 (2010) (https://doi.org/10.1063/1.3304922)

292. S. Grimme, J. Anthony, S. Ehrlich, and H. Krieg, *A consistent and accurate* ab initio *parametrization of density functional dispersion correction (DFT-D) for the 94 elements H-Pu*, Journal of Chemical Physics 132, 154104 (2010) (https://doi.org/10.1063/1.3382344).

293. M.D. Newton, *Quantum chemical probes of electron-transfer kinetics: the nature of donor-acceptor interactions*, Chemical Reviews 91, 767 (1991) (https://doi.org/10.1021/cr00005a007).

294. K. Senthilkumar, F.C. Grozema, F.M. Bickelhaupt, and L.D.A. Siebbeles, *Charge transport in columnar stacked triphenylenes: Effects of conformational fluctuations on charge transfer integrals and site energies*, Journal of Chemical Physics 119, 9809 (2003) (https://doi.org/10.1063/1.1615476).

295. K. Senthilkumar, F.C. Grozema, C. Fonseca Guerra, F.M. Bickelhaupt, F.D. Lewis, Y.A. Berlin, M.A. Ratner, and L.D.A. Siebbeles, *Absolute Rates of Hole Transfer in DNA*, Journal of the American Chemical Society 127, 14894 (2005) (https://doi.org/10.1021/ja054257e)

296. J. Neugebauer, *Couplings between electronic transitions in a subsystem formulation of time-dependent density functional theory*, Journal of Chemical Physics 126, 134116 (2007) (https://doi.org/10.1063/1.2713754).

297. J. Neugebauer, *Photophysical Properties of Natural Light-Harvesting Complexes Studied by Subsystem Density Functional Theory*, Journal of Physical Chemistry B 112, 2207 (2008) (https://doi.org/10.1021/jp709956k)

298. J. Neugebauer, *On the calculation of general response properties in subsystem density functional theory*, Journal of Chemical Physics 131, 084104 (2009) (https://doi.org/10.1063/1.3212883).

299. T.N. Truong and E.V. Stefanovich, *A new method for incorporating solvent effect into the classical, ab initio molecular orbital and density functional theory frameworks for arbitrary shape cavity*, Chemical Physics Letters 240, 253 (1995) (https://doi.org/10.1016/0009-2614(95)00541-B)

300. S. Gusarov, T. Ziegler, and A. Kovalenko, *Self-Consistent Combination of the Three-Dimensional RISM Theory of Molecular Solvation with Analytical Gradients and the Amsterdam Density Functional Package*, Journal of Physical Chemistry A 110, 6083 (2006) (https://doi.org/10.1021/jp054344t)

301. D. Casanova, S. Gusarov, A. Kovalenko, and T. Ziegler, *Evaluation of the SCF Combination of KS-DFT and 3D-RISM-KH; Solvation Effect on Conformational Equilibria, Tautomerization Energies, and Activation Barriers*, Journal of Chemical Theory and Computation 3, 458 (2007) (https://doi.org/10.1021/ct6001785)

302. A. Kovalenko and F. Hirata, *Self-consistent description of a metal-water interface by the Kohn-Sham density functional theory and the three-dimensional reference-interaction site model*, Journal of Chemical Physics 110, 10095 (1999) (https://doi.org/10.1063/1.478883)

303. A. Kovalenko and F. Hirata, *Potentials of mean force of simple ions in ambient aqueous solution. I. Three-dimensional reference-interaction site model approach*, Journal of Chemical Physics 112, 10391 (2000) (https://doi.org/10.1063/1.481676)

304. A. Kovalenko, *Three-dimensional RISM theory for molecular liquids and solid-liquid interfaces*, In Molecular Theory of Solvation; Hirata, Fumio, Ed.; Understanding Chemical Reactivity (series); Mezey, Paul G., Series Ed.; Kluwer Acadamic Publishers: Dordrecht, The Netherlands, 2003; Vol. 24, pp 169-275.

305. J.W. Kaminski, S. Gusarov, A. Kovalenko, T.A. Wesolowski, *Modeling solvatochromic shifts using the orbital-free embedding potential at statistically mechanically averaged solvent density*, Journal of Physical Chemistry A 114, 6082 (2010) (https://doi.org/10.1021/jp100158h)

306. L. Jensen, J. Autschbach, M. Krykunov, and G.C. Schatz, *Resonance vibrational Raman optical activity: A time-dependent density functional theory approach*, Journal of Chemical Physics 127, 134101 (2007) (https://doi.org/10.1063/1.2768533)

307. M. Krykunov, A. Banerjee, T. Ziegler and J. Autschbach, *Calculation of Verdet constants with time-dependent density functional theory. Implementation and results for small molecules*, Journal of Chemical Physics 122, 074105 (2005) (https://doi.org/10.1063/1.1850919)

308. E.J. Baerends, D.E. Ellis and P. Ros, *Self-consistent molecular Hartree-Fock-Slater calculations I. The computational procedure*, Chemical Physics 2, 41 (1973) (https://doi.org/10.1016/0301-0104(73)80059-X)

309. E.J. Baerends and P. Ros, *Evaluation of the LCAO Hartree-Fock-Slater method: Applications to transition-metal complexes*, International Journal of Quantum Chemistry 14, S12, 169 (1978) (https://doi.org/10.1002/qua.560140814)

310. G. te Velde, F.M. Bickelhaupt, E.J. Baerends, C. Fonseca Guerra, S.J.A. van Gisbergen, J.G. Snijders, T. Ziegler, *Chemistry with ADF*, Journal of Computational Chemistry 22, 931 (2001) (https://doi.org/10.1002/jcc.1056)

311. A. Devarajan, A. Gaenko, and J. Autschbach, *Two-component relativistic density functional method for computing nonsingular complex linear response of molecules based on the zeroth order regular approximation*, Journal of Chemical Physics 130, 194102 (2009) (https://doi.org/10.1063/1.3123765)

312. M.R. Pederson, S.N. Khanna, *Magnetic anisotropy barrier for spin tunneling in $Mn_{12} O_{12}$ molecules*, Physical Review B 60, 9566 (1999) (https://doi.org/10.1103/PhysRevB.60.9566)

313. F. Neese, *Calculation of the zero-field splitting tensor on the basis of hybrid density functional and Hartree-Fock theory*, Journal of Chemical Physics 127, 164112 (2007) (https://doi.org/10.1063/1.2772857)

314. C. van Wüllen, *Magnetic anisotropy from density functional calculations. Comparison of different approaches: $Mn_{12} O_{12}$ acetate as a test case*, Journal of Chemical Physics 130, 194109 (2009) (https://doi.org/10.1063/1.3134430)

315. S. Schmitt, P. Jost, C. van Wüllen, *Zero-field splittings from density functional calculations: Analysis and improvement of known methods*, Journal of Chemical Physics 134, 194113 (2011) (https://doi.org/10.1063/1.3590362)

316. S.N. Steinmann, and C. Corminboeuf, *Comprehensive Benchmarking of a Density-Dependent Dispersion Correction*, Journal of Chemical Theory and Computation 7, 3567 (2011) (https://doi.org/10.1021/ct200602x).

317. R.C. Raffenetti, *Eventempered atomic orbitals. II. Atomic SCF wavefunctions in terms of eventempered exponential bases*, Journal of Chemical Physics 59, 5936 (1973) (https://doi.org/10.1063/1.1679962)

318. D.P. Chong, *Completeness profiles of one-electron basis sets*, Canadian Journal of Chemistry 73, 79 (1995) (https://doi.org/10.1139/v95-011)

319. G.D. Zeiss, W.R. Scott, N. Suzuki, D.P. Chong, S.R. Langhoff, *Finite-field calculations of molecular polarizabilities using field-induced polarization functions: second- and third-order perturbation correlation corrections to the coupled Hartree-Fock polarizability of $H_2 O$*, Molecular Physics 37, 1543 (1979) (https://doi.org/10.1080/00268977900101121)

320. E. van Lenthe and E.J. Baerends, *Optimized Slater-type basis sets for the elements 1-118*, Journal of Computational Chemistry 24, 1142 (2003) (https://doi.org/10.1002/jcc.10255)

321. D.P. Chong, E. van Lenthe, S.J.A. van Gisbergen and E.J. Baerends, *Even-tempered Slater-Type orbitals revisited: From Hydrogen to Krypton*, Journal of Computational Chemistry 25, 1030 (2004) (https://doi.org/10.1002/jcc.20030)

322. D.P. Chong, *Augmenting basis set for time-dependent density functional theory calculation of excitation energies: Slater-type orbitals for hydrogen to krypton*, Molecular Physics 103, 749 (2005) (https://doi.org/10.1080/00268970412331333618)

323. T. Ziegler, A. Rauk and E.J. Baerends, *On the calculation of Multiplet Energies by the Hartree Fock Slater method*, Theoretica Chimica Acta 43, 261 (1977) (https://doi.org/10.1007/BF00551551)

324. C. Daul, *DFT applied to excited states*, International Journal of Quantum Chemistry 52, 867 (1994) (https://doi.org/10.1002/qua.560520414)

325. E.J. Baerends, V. Branchadell and M. Sodupe, *Atomic reference-energies for density functional calculations*, Chemical Physics Letters 265, 481 (1997) (https://doi.org/10.1016/S0009-2614(96)01449-2)

326. P.J. van den Hoek, E.J. Baerends, and R.A. van Santen, *Ethylene epoxidation on silver(110): the role of subsurface oxygen*, Journal of Physical Chemistry 93, 6469 (1989) (https://doi.org/10.1021/j100354a038)

327. J. Autschbach, S. Zheng, and R.W. Schurko, *Analysis of Electric Field Gradient Tensors at Quadrupolar Nuclei in Common Structural Motifs*, Concepts in Magnetic Resonance Part A 36A, 84 (2010) (https://doi.org/10.1002/cmr.a.20155)

328. A.J. Rossini, R.W. Mills, G.A. Briscoe, E.L. Norton, S.J. Geier, I. Hung, S. Zheng, J. Autschbach, and R.W. Schurko, *Solid-State Chlorine NMR of Group IV Transition Metal Organometallic Complexes*, Journal of the American Chemical Society 131, 3317 (2009) (https://doi.org/10.1021/ja808390a)

329. J. Autschbach, *Analyzing NMR shielding tensors calculated with two-component relativistic methods using spin-free localized molecular orbitals*, Journal of Chemical Physics 128, 164112 (2008) (https://doi.org/10.1063/1.2905235)

330. J. Autschbach and S. Zheng, *Analyzing Pt chemical shifts calculated from relativistic density functional theory using localized orbitals: The role of Pt 5d lone pairs*, Magnetic Resonance in Chemistry 46, S45 (2008) (https://doi.org/10.1002/mrc.2289)

331. J. Autschbach and S. Zheng, *Relativistic computations of NMR parameters from first principles: Theory and applications*, Annual Reports on NMR Spectroscopy 67, 1 (2009) (https://doi.org/10.1016/S0066-4103(09)06701-5)

332. J. Autschbach, *Analyzing molecular properties calculated with two-component relativistic methods using spin-free Natural Bond Orbitals: NMR spin-spin coupling constants* Journal of Chemical Physics 127, 124106 (2007) (https://doi.org/10.1063/1.2768363)

333. J. Autschbach and B. Le Guennic, *Analyzing and interpreting NMR spin-spin coupling constants from molecular orbital calculations*, Journal of Chemical Education 84, 156 (2007) (https://doi.org/10.1021/ed084p156)

334. A.M.A. Boshaala, S.J. Simpson, J. Autschbach and S. Zheng, *Synthesis and Characterization of the Trihalophosphine Compounds of Ruthenium* [RuX$_2$ (η^6-cymene)(PY$_3$)] (X = Cl, Br, Y = F, Cl, Br) *and the Related* PF$_2$ (NMe$_2$) *and* P(NMe$_2$)$_3$ *Compounds; Multinuclear NMR Spectroscopy and the X-ray Single Crystal Structures of* [RuBr$_2$ (η^6-cymene)(PF$_3$)], [RuBr$_2$ (η^6-cymene)(PF$_2$ {NMe$_2$ })], *and* [RuI$_2$ (η^6-cymene)(P{NMe$_2$ }$_3$)], Inorganic Chemistry 47, 9279 (2008) (https://doi.org/10.1021/ic800611h)

335. A. Michalak, M. Mitoraj, and T. Ziegler, *Bond Orbitals from Chemical Valence Theory*, Journal of Physical Chemistry A 112, 1933 (2008) (https://doi.org/10.1021/jp075460u)

336. E. Clementi, C. Roetti, *Roothaan-Hartree-Fock atomic wavefunctions: Basis functions and their coefficients for ground and certain excited states of neutral and ionized atoms, Z ≤ 54*, Atomic Data and Nuclear Data Tables 14, 177 (1974) (https://doi.org/10.1016/S0092-640X(74)80016-1)

337. A.D. McLean, R.S. McLean, *Roothaan-Hartree-Fock atomic wave functions Slater basis-set expansions for Z = 55-92*, Atomic Data and Nuclear Data Tables 26, 197 (1981) (https://doi.org/10.1016/0092-640X(81)90012-7)

338. J.G. Snijders, P. Vernooijs, E.J. Baerends, *Roothaan-Hartree-Fock-Slater atomic wave functions: Single-zeta, double-zeta, and extended Slater-type basis sets for* $_{87}$Fr-$_{103}$Lr, Atomic Data and Nuclear Data Tables 26, 483 (1981) (https://doi.org/10.1016/0092-640X(81)90004-8)

339. J. Autschbach and B. Pritchard, *Calculation of molecular g-tensors using the zeroth-order regular approximation and density functional theory: expectation value versus linear response approaches*, Theoretical Chemistry Accounts 129, 453 (2011) (https://doi.org/10.1007/s00214-010-0880-x)

340. J. Autschbach, S. Patchkovskii, and B. Pritchard, *Calculation of Hyperfine Tensors and Paramagnetic NMR Shifts Using the Relativistic Zeroth-Order Regular Approximation and Density Functional Theory*, Journal of Chemical Theory and Computation 7, 2175 (2011) (https://doi.org/10.1021/ct200143w)

341. S. Moon, and S. Patchkovskii, *First-principles calculations of paramagnetic NMR shifts*, in Calculation of NMR and EPR parameters, ISBN13: 9783527307791, M. Kaupp, M. Bühl, V.G. Malkin, Editors, (Wiley, Weinheim, 2004).

342. P. Hrobárik, Ro. Reviakine, A.V. Arbuznikov, O.L. Malkina, V.G. Malkin, F.H. Köhler, and M. Kaupp, *Density functional calculations of NMR shielding tensors for paramagnetic systems with arbitrary spin multiplicity: Validation on 3d metallocenes*, Journal of Chemical Physics 126, 024107 (2007) (https://doi.org/10.1063/1.2423003)

343. J. Li, M.R. Nelson, C.Y. Peng, D. Bashford, and L. Noodleman, *Incorporating Protein Environments in Density Functional Theory: A Self-Consistent Reaction Field Calculation of Redox Potentials of [2Fe2S] Clusters in Ferredoxin and Phthalate Dioxygenase Reductase*, Journal of Physical Chemistry A 102, 6311 (1998) (https://doi.org/10.1021/jp980753w)

344. T. Liu, W.-G Han, F. Himo, G.M. Ullmann, D. Bashford, A. Toutchkine, K.M. Hahn, and L. Noodleman, *Density Functional Vertical Self-Consistent Reaction Field Theory for Solvatochromism Studies of Solvent-Sensitive Dyes*, Journal of Physical Chemistry A 108, 3545 (2004) (https://doi.org/10.1021/jp031062p)

345. W.-G. Han, T. Liu, F. Himo, A. Toutchkine, D. Bashford, K.M. Hahn, L. Noodleman, *A Theoretical Study of the UV/Visible Absorption and Emission Solvatochromic Properties of Solvent-Sensitive Dyes*, ChemPhysChem 4, 1084 (2003) (https://doi.org/10.1002/cphc.200300801)

346. A. Kovyrshin, J. Neugebauer, *State-selective optimization of local excited electronic states in extended systems*, Journal of Chemical Physics 133, 174114 (2010) (https://doi.org/10.1063/1.3488230)

347. M. Swart, E. Rösler, and F. M. Bickelhaupt, *Proton affinities of maingroup-element hydrides and noble gases: Trends across the periodic table, structural effects, and DFT validation*, Journal of Computational Chemistry 27, 1486 (2006) (https://doi.org/10.1002/jcc.20431)

348. M. Swart, and F. M. Bickelhaupt, *Proton Affinities of Anionic Bases: Trends Across the Periodic Table, Structural Effects, and DFT Validation*, Journal of Chemical Theory and Computation 2, 281 (2006) (https://doi.org/10.1021/ct200602x).

349. A. Kovalenko and F. Hirata, *Potentials of mean force of simple ions in ambient aqueous solution. II. Solvation structure from the three-dimensional reference-interaction site model approach, and comparison with simulations*, Journal of Chemical Physics 112, 10403 (2000) (https://doi.org/10.1063/1.481677)

350. M. Seth, G. Mazur, and T. Ziegler, *Time-dependent density functional theory gradients in the Amsterdam density functional package: geometry optimizations of spin-flip excitations*, Theoretical Chemistry Accounts 129, 331 (2011) (https://doi.org/10.1007/s00214-010-0819-2)

351. S.Y. Quek, L. Venkataraman, H.J. Choi, S.G. Louie, M.S. Hybertsen and J.B. Neaton, *mine-Gold Linked Single-Molecule Circuits: Experiment and Theory*, Nano Letters 7, 3477 (2007) (https://doi.org/10.1021/nl072058i)

352. M. Pavanello and J. Neugebauer, *Modelling charge transfer reactions with the frozen density embedding formalism*, Journal of Chemical Physics 135, 234103 (2011) (https://doi.org/10.1063/1.3666005)

353. M. Pavanello, T. Van Voorhis, L. Visscher, and J. Neugebauer, *An accurate and linear-scaling method for calculating charge-transfer excitation energies and diabatic couplings*, Journal of Chemical Physics 138, 054101 (2013) (https://doi.org/10.1063/1.4789418)

354. U. Ekström, L. Visscher, R. Bast, A.J. Thorvaldsen, and K. Ruud, *Arbitrary-Order Density Functional Response Theory from Automatic Differentiation*, Journal of Chemical Theory and Computation 6, 1971 (2010)

(https://doi.org/10.1021/ct100117s)

355. M. Seth and T. Ziegler, *Range-Separated Exchange Functionals with Slater-Type Functions*, Journal of Chemical Theory and Computation 8, 901 (2012) (https://doi.org/10.1021/ct300006h)

356. E.R. Johnson, S. Keinan, P. Mori-Sánchez, J. Contreras-García A.J. Cohen, and W. Yang, *Revealing Non-Covalent Interactions*, Journal of the American Chemical Society 132, 6498 (2010) (https://doi.org/10.1021/ja100936w)

357. J. Contreras-García E.R. Johnson, S. Keinan, R. Chaudret, J-P. Piquemal, D.N. Beratan, and W. Yang, *NCIPLOT: A Program for Plotting Noncovalent Interaction Regions*, Journal of Chemical Theory and Computation 7, 625 (2011) (https://doi.org/10.1021/ct100641a)

358. P. de Silva, J. Korchowiec, T.A. Wesolowski, *Revealing the Bonding Pattern from the Molecular Electron Density Using Single Exponential Decay Detector: An Orbital-Free Alternative to the Electron Localization Function*, ChemPhysChem 13, 3462 (2012) (https://doi.org/10.1002/cphc.201200500)

359. R. De Francesco, M. Stener, and G. Fronzoni, *Theoretical Study of Near-Edge X-ray Absorption Fine Structure Spectra of Metal Phthalocyanines at C and N K-Edges*, Journal of Physical Chemistry A, 116 2285 (2012) (https://doi.org/10.1021/jp2109913)

360. B. Delley, *An all-electron numerical method for solving the local density functional for polyatomic molecules*, Journal of Chemical Physics 92, 508 (1992) (https://doi.org/10.1063/1.458452)

361. A.D. Becke, *A multicenter numerical integration scheme for polyatomic molecules*, Journal of Chemical Physics 88, 2547 (1988) (https://doi.org/10.1063/1.454033)

362. J.L. Payton, S.M. Morton, Justin E. Moore, and Lasse Jensen, *A discrete interaction model/quantum mechanical method for simulating surface-enhanced Raman spectroscopy*, Journal of Chemical Physics 136, 214103 (2012) (https://doi.org/10.1063/1.4722755)

363. J.I. Rodríguez, *An Efficient Method for Computing the QTAIM Topology of a Scalar Field: The Electron Density Case*, Journal of Computational Chemistry 34, 681 (2013) (https://doi.org/10.1002/jcc.23180)

364. C. König, N. Schlüter. J. Neugebauer, *Direct Determination of Exciton Couplings from Subsystem TDDFT within the Tamm-Dancoff Approximation*, Journal of Chemical Physics 138, 034104 (2013) (https://doi.org/10.1063/1.4774117)

365. C.J.O. Verzijl, J.S. Seldenthuis, and J.M. Thijssen, *Applicability of the wide-band limit in DFT-based molecular transport calculations*, Journal of Chemical Physics 138, 094102 (2013) (https://doi.org/10.1063/1.4793259)

366. H. Kim, J.-M. Choi, W.A. Goddard, *Universal Correction of Density Functional Theory to Include London Dispersion (up to Lr, Element 103)*, Journal of Physical Chemistry Letters 3, 360 (2012) (https://doi.org/10.1021/jz2016395)

367. M. Swart, *A new family of hybrid density functionals*, Chemical Physics Letters 580, 166 (2013) (https://doi.org/10.1016/j.cplett.2013.06.045)

368. J. Autschbach, C.D. Igna, T. Ziegler, *A theoretical investigation of the apparently irregular behavior of Pt-Pt spin-spin coupling constants* Journal of the American Chemical Society 125, 1028 (2003) (https://doi.org/10.1021/ja027931q)

369. B.L. Guennic, K. Matsumoto, J. Autschbach, *On the NMR properties of platinum thallium bonded complexes: Analysis of relativistic density functional theory results*, Magnetic Resonance in Chemistry 42, S99 (2004) (https://doi.org/10.1002/mrc.1450)

370. J. Khandogin, T. Ziegler, *A density functional study of nuclear magnetic resonance spin-spin coupling constants in transition metal systems*, Spectrochimica Acta Part A 55, 607 (1999) (https://doi.org/10.1016/S1386-1425(98)00265-0)

371. N.F. Ramsey, *Electron Coupled Interactions between Nuclear Spins in Molecules*, Physical Review 91, 303 (1953) (https://doi.org/10.1103/PhysRev.91.303)

372. R.M. Dickson, T. Ziegler, *NMR Spin-Spin Coupling Constants from Density Functional Theory with Slater-Type Basis Functions*, Journal of Physical Chemistry 100, 5286 (1996) (https://doi.org/10.1021/jp951930l)

373. D.L. Bryce, R. Wasylishen, *Indirect Nuclear Spin-Spin Coupling Tensors in Diatomic Molecules: A Comparison of Results Obtained by Experiment and First Principles Calculations*, Journal of the American Chemical Society 122, 3197 (2000) (https://doi.org/10.1021/ja9942134)

374. G. Schreckenbach, S.K. Wolff, T. Ziegler, *Covering the Entire Periodic Table: Relativistic Density Functional Calculations of NMR Chemical Shifts in Diamagnetic Actinide Compounds*, in Modeling NMR chemical shifts, ACS Symposium Series, Vol 732, J.C. Facelli, A.C. de Dios, Editors (American Chemical Society, Washington DC, 1999), Chapter 7 (https://doi.org/10.1021/bk-1999-0732.ch007)

375. M. Franchini, P.H.T. Philipsen, L. Visscher, *The Becke Fuzzy Cells Integration Scheme in the Amsterdam Density Functional Program Suite*, Journal of Computational Chemistry 34, 1818 (2013) (https://doi.org/10.1002/jcc.23323).

376. A. Tkatchenko, R.A. DiStasio Jr., R. Car, M. Scheffler *Accurate and Efficient Method for Many-Body van der Waals Interactions*, Physical Review Letters 108, 236402 (2012) (https://doi.org/10.1103/PhysRevLett.108.236402)

377. A. Ambrosetti, A.M. Reilly, Robert A. DiStasio Jr., A. Tkatchenko, *Long-range correlation energy calculated from coupled atomic response functions*, Journal of Chemical Physics 140, 18A508 (2014) (https://doi.org/10.1063/1.4865104)

378. A.V. Marenich, S.V. Jerome, C.J. Cramer, D.G. Truhlar, *Charge Model 5: An Extension of Hirshfeld Population Analysis for the Accurate Description of Molecular Interactions in Gaseous and Condensed Phases*, Journal of Chemical Theory and Computation 8, 527 (2012) (https://doi.org/10.1021/ct200866d)

379. M. Franchini, P.H.T. Philipsen, E. van Lenthe, L. Visscher, *Accurate Coulomb Potentials for Periodic and Molecular Systems through Density Fitting*, Journal of Chemical Theory and Computation 10, 1994 (2014) (https://doi.org/10.1021/ct500172n)

380. A.D. Becke, R.M. Dickson, *Numerical solution of Poisson's equation in polyatomic molecules*, Journal of Chemical Physics 89, 2993 (1988) (https://doi.org/10.1063/1.455005)

381. M.-C. Kim, E. Sim, and K. Burke, *Understanding and Reducing Errors in Density Functional Calculations*, Physical Review Letters 111, 2073003 (2013) (https://doi.org/10.1103/PhysRevLett.111.073003)

382. D.V. Chulhai and L. Jensen, *Simulating Surface-Enhanced Raman Optical Activity Using Atomistic Electrodynamics-Quantum Mechanical Models*, Journal of Physical Chemistry A 118, 9069 (2014) (https://doi.org/10.1021/jp502107f)

383. D.V. Chulhai and L. Jensen, *Plasmonic Circular Dichroism of 310- and α-Helix Using a Discrete Interaction Model/Quantum Mechanics Method*, Journal of Physical Chemistry A 119, 5218 (2015) (https://doi.org/10.1021/jp5099188)

384. D.V. Chulhai and L. Jensen, *Frozen Density Embedding with External Orthogonality in Delocalized Covalent Systems*, Journal of Chemical Theory and Computation 11, 3080 (2015) (https://doi.org/10.1021/acs.jctc.5b00293).

385. S. Bernadotte, A.J. Atkins, Ch.R. Jacob, *Origin-independent calculation of quadrupole intensities in X-ray spectroscopy*, Journal of Chemical Physics 137, 204106 (2012) (https://doi.org/10.1063/1.4766359)

386. N. Lee, T. Petrenko, U, Bergmann, F. Neese, and S. DeBeer, *Probing Valence Orbital Composition with Iron $K\beta$ X-ray Emission Spectroscopy*, Journal of the American Chemical Society 132, 9715 (2010) (https://doi.org/10.1021/ja101281e)

387. A.J. Atkins, M. Bauer, and Ch.R. Jacob, *The chemical sensitivity of X-ray spectroscopy: high energy resolution XANES versus X-ray emission spectroscopy of substituted ferrocenes*, Physical Chemistry Chemical Physics 15, 8095 (2013) (https://doi.org/10.1039/C3CP50999K)

388. P. de Silva and C. Corminboeuf, *Communication: A new class of non-empirical explicit density functionals on the third rung of Jacob's ladder*, Journal of Chemical Physics 143, 111105 (2015) (https://doi.org/10.1063/1.4931628)

389. S. Grimme, *A simplified Tamm-Dancoff density functional approach for the electronic excitation spectra of very large molecules*, Journal of Chemical Physics 138, 244104 (2013) (https://doi.org/10.1063/1.4811331)

390. C. Bannwarth and S. Grimme, *A simplified time-dependent density functional theory approach for electronic ultraviolet and circular dichroism spectra of very large molecules*, Computational and Theoretical Chemistry 1040-1041, 45 (2014) (https://doi.org/10.1016/j.comptc.2014.02.023)

391. T. Risthaus, A. Hansen, and S. Grimme, *Excited states using the simplified Tamm-Dancoff-Approach for range-separated hybrid density functionals: development and application*, Physical Chemistry Chemical Physics 16, 14408 (2014) (https://doi.org/10.1039/C3CP54517B)

392. M.A.L. Marques, M.J.T. Oliveira, and T. Burnus, *Libxc: a library of exchange and correlation functionals for density functional theory*, Computer Physics Communications 183, 2272 (2012) (https://doi.org/10.1016/j.cpc.2012.05.007)

393. J. Cullen, M. Krykunov, and T. Ziegler, *The formulation of a self-consistent constricted variational density functional theory for the description of excited states*, Chemical Physics 391, 11 (2011) (https://doi.org/10.1016/j.chemphys.2011.05.021)

394. M. Krykunov and T. Ziegler, *Self-consistent Formulation of Constricted Variational Density Functional Theory with Orbital Relaxation. Implementation and Applications*, Journal of Chemical Theory and Computation 9, 2761 (2013) (https://doi.org/10.1021/ct300891k)

395. J. Sun, J.P. Perdew, and A. Ruzsinszky, *Semilocal density functional obeying a strongly tightened bound for exchange*, Proceedings of the National Academy of Sciences 112, 685 (2015) (https://doi.org/10.1073/pnas.1423145112)

396. R.L. Martin, *Natural transition orbitals*, Journal of Chemical Physics 118, 4775 (2003) (https://doi.org/10.1063/1.1558471)

397. R. López, J.F. Rico, G. Ramírez, I. Ema, D. Zorrilla, *DAMQT 2.0: A new version of the DAMQT package for the analysis of electron density in molecules*, Computer Physics Communications 192, 289 (2015) (https://doi.org/10.1016/j.cpc.2015.02.027)

398. S. Grimme and A. Hansen, *A Practicable Real-Space Measure and Visualization of Static Electron-Correlation Effects*, Angewandte Chemie International Edition 54, 12308 (2015) (https://doi.org/10.1002/anie.201501887)

399. A.V. Marenich, C.J. Cramer, and D.G. Truhlar, *Generalized Born Solvation Model SM12*, Journal of Chemical Theory and Computation 9, 609 (2013) (https://doi.org/10.1021/ct300900e)

400. W.C. Still, A. Tempczyk, R.C. Hawley, and T. Hendrickson, *Semianalytical treatment of solvation for molecular mechanics and dynamics*, Journal of the American Chemical Society 112, 6127 (1990) (https://doi.org/10.1021/ja00172a038)

401. D.A. Liotard, G.D. Hawkins, G.C. Lynch, C.J. Cramer, and D.G. Truhlar, *Improved Methods for Semiempirical Solvation Models*, Journal of Computational Chemistry 16, 422 (1995) (https://doi.org/10.1002/jcc.540160405)

402. M. Mantina, R. Valero, C.J. Cramer, D.G. Truhlar, in *CRC Handbook of Chemistry and Physics*, 91st ed. (2010-2011), ISBN13 9781439820773, W.M. Haynes, Ed., CRC Press: Boca Raton, FL, 2010.

403. J. Autschbach and T. Ziegler, *Solvent Effects on Heavy Atom Nuclear Spin.Spin Coupling Constants: A Theoretical Study of Hg.C and Pt.P Couplings*, Journal of the American Chemical Society 123, 3341 (2001) (https://doi.org/10.1021/ja003481v)

404. J. Autschbach and T. Ziegler, *A Theoretical Investigation of the Remarkable Nuclear Spin.Spin Coupling Pattern in* $[(NC)_5 Pt\text{-}Tl(CN)]^-$, Journal of the American Chemical Society 123, 5320 (2001) (https://doi.org/10.1021/ja003866d)

405. P. Ramos, M. Mankarious, M. Pavanello, *A critical look at methods for calculating charge transfer couplings fast and accurately*, in *Practical Aspects of Computational Chemistry IV*, Jerzy Leszczynski and Manoj Shukla (eds.), 2016, Springer (Accepted).

406. A. Solovyeva, M. Pavanello, J. Neugebauer, *Describing Long-Range Charge-Separation Processes with Subsystem Density-Functional Theory*, J. Chem. Phys. 140, 164103 (2014) (https://doi.org/10.1063/1.4871301)

407. P. Ramos, M. Pavanello, *Constrained Subsystem Density Functional Theory*, Physical Chemistry Chemical Physics 18, 21172 (2016) (https://doi.org/10.1039/C6CP00528D).

408. P. Ramos, M. Papadakis, M. Pavanello, *Performance of Frozen Density Embedding for Modeling Hole Transfer Reactions*, Journal of Physical Chemistry B 119, 7541 (2015) (https://doi.org/10.1021/jp511275e)

409. Q. Wu, T. Van Voorhis, *Direct optimization method to study constrained systems within density-functional theory*, Physical Review A 72, 024502 (2005) (https://doi.org/10.1103/PhysRevA.72.024502)

410. P. Ramos, M. Pavanello, *Constrained subsystem density functional theory*, Physical Chemistry Chemical Physics 18, 21172 (2016) (https://doi.org/10.1039/C6CP00528D)

411. F. Senn, M. Krykunov, *Excited State Studies of Polyacenes Using the All-Order Constricted Variational Density Functional Theory with Orbital Relaxation*, Journal of Physical Chemistry A 119, 10575 (2015) (https://doi.org/10.1021/acs.jpca.5b07075)

412. T. Ziegler, M. Krykunov, J. Cullen, *The implementation of a self-consistent constricted variational density functional theory for the description of excited states*, Journal of Chemical Physics 136, 124107 (2012) (https://doi.org/10.1063/1.3696967)

413. T. Ziegler, M. Krykunov, I. Seidu, Y.C. Park, *Constricted Variational Density Functional Theory Approach to the Description of Excited States*, Density Functional Methods for Excited States: Topics in Current Chemistry 368, 61 (2016) (https://doi.org/10.1007/128_2014_611)

414. A. van der Avoird, P.E.S. Wormer, F. Mulder, R.M. Berns, Topics in Current Chemistry 93, 1 (1980)

415. C.J. Pickard and F. Mauri, *First-Principles Theory of the EPR g Tensor in Solids: Defects in Quartz*, Physical Review Letters 88, 86403 (2002) (https://doi.org/10.1103/PhysRevLett.88.086403)

416. S. Patchkovskii and G. Schreckenbach in Calculation of NMR and EPR parameters, ISBN13: 9783527307791, M. Kaupp, M. Bühl, V.G. Malkin, Editors, (Wiley, Weinheim, 2004).

417. S. Patchkovskii, R.S. Strong, C.J. Pickard and S. Un, *Gauge invariance of the spin-other-orbit contribution to the g-tensors of electron paramagnetic resonance*, Journal of Chemical Physics 122, 214101 (2005) (https://doi.org/10.1063/1.1917840)

418. J.S. Seldenthuis, H.S.J. van der Zant, M.A. Ratner and J.M. Thijssen, *Vibrational Excitations in Weakly Coupled Single-Molecule Junctions: A Computational Analysis*, ACS Nano 2, 1445 (2008) (https://doi.org/10.1021/nn800170h)

419. G.M. Sando and K.G. Spears, *Ab Initio Computation of the Duschinsky Mixing of Vibrations and Nonlinear Effects*, Journal of Physical Chemistry A 105, 5326 (2001) (https://doi.org/10.1021/jp004230b)

420. P.T. Ruhoff and M.A. Ratner, *Algorithms for computing Franck-Condon overlap integrals*, International Journal of Quantum Chemistry 77, 383 (2000) (https://doi.org/10.1002/(SICI)1097-461X(2000)77:1%3C383::AID-QUA38%3E3.0.CO;2-0)

421. J. Poater, E. van Lenthe and E.J. Baerends, *Nuclear magnetic resonance chemical shifts with the statistical average of orbital-dependent model potentials in Kohn.Sham density functional theory*, Journal of Chemical Physics 118, 8584 (2003) (https://doi.org/10.1063/1.1567252)

422. M. Krykunov, T. Ziegler and E. van Lenthe, *Hybrid density functional calculations of nuclear magnetic shieldings using Slater-type orbitals and the zeroth-order regular approximation*, International Journal of Quantum Chemistry 109, 1676 (2009) (https://doi.org/10.1002/qua.21985)

423. M. Krykunov, T. Ziegler and E. van Lenthe, *Implementation of a hybrid DFT method for calculating NMR shieldings using Slater-type orbitals with spin-orbital coupling included. Applications to* [187] Os, [195] Pt and [13] C *in heavy metal complexes*, Journal of Physical Chemistry A 113, 11495 (2009) (https://doi.org/10.1021/jp901991s)

424. R. Rüger, E. van Lenthe, T. Heine, L. Visscher, *Tight-Binding Approximations to Time-Dependent Density Functional Theory - a fast approach for the calculation of electronically excited states*, Journal of Chemical Physics 144, 184103 (2016) (https://doi.org/10.1063/1.4948647)

425. J. Autschbach, *Two-component relativistic hybrid density functional computations of nuclear spin-spin coupling tensors using Slater-type basis sets and density-fitting techniques*, Journal of Chemical Physics 129, 094105 (2008) (https://doi.org/10.1063/1.2969100), Erratum: Journal of Chemical Physics 130, 209901 (2009) (https://doi.org/10.1063/1.3131724)

426. D.L. Bryce and J. Autschbach, *Relativistic hybrid density functional calculations of indirect nuclear spin-spin coupling tensors . Comparison with experiment for diatomic alkali metal halides*, Canadian Journal of Chemistry 87, 927 (2009) (https://doi.org/10.1139/V09-040)

427. J. Autschbach, *The role of the exchange-correlation response kernel and scaling corrections in relativistic density functional nuclear magnetic shielding calculations with the zeroth-order regular approximation*, Molecular Physics 111, 2544 (2013) (https://doi.org/10.1080/00268976.2013.796415)

428. J. Autschbach and H.F. King, *Analyzing molecular static linear response properties with perturbed localized orbitals*, Journal of Chemical Physics 133, 044109 (2010) (https://doi.org/10.1063/1.3455709)

429. J. Autschbach, *Analyzing NMR shielding tensors calculated with two-component relativistic methods using spin-free localized molecular orbitals*, Journal of Chemical Physics 128, 164112 (2008) (https://doi.org/10.1063/1.2905235)

430. K.G. Dyall, *An exact separation of the spin-free and spin-dependent terms of the Dirac-Coulomb-Breit Hamiltonian*, Journal of Chemical Physics 100, 2118 (1994) (https://doi.org/10.1063/1.466508)

431. L. Visscher and E. van Lenthe, *On the distinction between scalar and spin-orbit relativistic effects*, Chemical Physics Letters 306, 357 (1999) (https://doi.org/10.1016/S0009-2614(99)00458-3)

432. Y.A. Wang, C.Y. Yam, Y.K. Chen, G. Chen, *Communication: Linear-expansion shooting techniques for accelerating self-consistent field convergence*, Journal of Chemical Physics 134, 241103 (2011) (https://doi.org/10.1063/1.3609242)

433. W.L. Jorgensen, J.D. Madura, C.J. Swenson, *Optimized intermolecular potential functions for liquid hydrocarbons,* Journal of the American Chemical Society 106, 6638 (1984) (https://doi.org/10.1021/ja00334a030)

434. A.E. Kobryn, A. Kovalenko, *Molecular theory of hydrodynamic boundary conditions in nanofluidics,* Journal of Chemical Physics 129, 134701 (2008) (https://doi.org/10.1063/1.2972978)

435. O. Acevedo, W.L. Jorgensen, *Influence of Inter- and Intramolecular Hydrogen Bonding on Kemp Decarboxylations from QM/MM Simulations,* Journal of the American Chemical Society 127, 8829 (2005) (https://doi.org/10.1021/ja051793y)

436. S. Grimme, S. Ehrlich, and L. Goerigk, *Effect of the Damping Function in Dispersion Corrected Density Functional Theory*, Journal of Computational Chemistry 32, 1457 (2011) (https://doi.org/10.1002/jcc.21759).

437. Ph. Haas, F. Tran, P. Blaha, and K.H. Schwartz, *Construction of an optimal GGA functional for molecules and solids*, Physical Review B83, 205117 (2011) (https://doi.org/10.1103/PhysRevB.83.205117).

438. J.P. Perdew, A. Ruzsinszky, G.I. Csonka, L.A. Constantin, and J. Sun, *Workhorse Semilocal Density Functional for Condensed Matter Physics and Quantum Chemistry*, Physical Review Letters 103, 026403 (2009) (https://doi.org/10.1103/PhysRevLett.103.026403).

439. C.A. Peeples and G. Schreckenbach, *Implementation of the SM12 Solvation Model into ADF and Comparison with COSMO*, Journal of Chemical Theory and Computation 12, 4033 (2016) (https://doi.org/10.1021/acs.jctc.6b00410)

440. F. Feixas, E. Matito, J. Poater and M. Sola, *Quantifying aromaticity with electron delocalisation measures*, Chem. Soc. Rev. 44, 6434 (2015) (https://doi.org/10.1039/c5cs00066a)

441. P. de Silva and C. Corminboeuf, *Simultaneous Visualization of Covalent and Noncovalent Interactions Using Regions of Density Overlap*, Journal of Chemical Theory and Computation 10, 3745 (2014) (https://doi.org/10.1021/ct500490b)

442. N. Raimbault, P.L. de Boeij, P. Romaniello, and J.A. Berger, *Gauge-Invariant Calculation of Static and Dynamical Magnetic Properties from the Current Density*, Physical Review Letters 114, 066404 (2015) (https://doi.org/10.1103/PhysRevLett.114.066404)

443. N. Raimbault, P.L. de Boeij, P. Romaniello, and J.A. Berger, *Gauge-Invariant Formulation of Circular Dichroism*, Journal of Chemical Theory and Computation 12, 3278 (2016) (https://doi.org/10.1021/acs.jctc.6b00068)

444. N. Raimbault, Gauge-invariant magnetic properties from the current (https://tel.archives-ouvertes.fr/tel-01235055), PhD thesis, Université Paul Sabatier, LCPQ, Toulouse, 2015

445. A.J. Sadlej and J.G. Snijders, *Spin separation in the regular Hamiltonian approach to solutions of the Dirac equation*, Chemical Physics Letters 229, 435 (1994) (https://doi.org/10.1016/0009-2614(94)01067-6)

446. E. van Lenthe, E.J. Baerends, and J.G. Snijders, *Construction of the Foldy-Wouthuysen transformation and solution of the Dirac equation using large components only*, Journal of Chemical Physics 105, 2373 (1996) (https://doi.org/10.1063/1.472104)

447. K.G. Dyall, *Interfacing relativistic and nonrelativistic methods. I. Normalized elimination of the small component in the modified Dirac equation*, Journal of Chemical Physics 106, 9618 (1997) (https://doi.org/10.1063/1.473860)

448. W. Kutzelnigg and W. Liu, *Quasirelativistic theory equivalent to fully relativistic theory*, Journal of Chemical Physics 123, 241102 (2005) (https://doi.org/10.1063/1.2137315)

449. A. Goez, and J. Neugebauer, *A Local Variant of the Conductor-Like Screening Model for Fragment-Based Electronic-Structure Method*, Journal of Chemical Theory and Computation 11, 5277 (2015) (https://doi.org/10.1021/acs.jctc.5b00832)

450. A. Goez, and J. Neugebauer, *Including protein density relaxation effects in first-principles embedding calculations of cofactor excitation energies*, Molecular Physics 115, 526 (2016) (https://doi.org/10.1080/00268976.2016.1199823)

451. Z. Hu, J. Autschbach, and L. Jensen, *Simulation of resonance hyper-Rayleigh scattering of molecules and metal clusters using a time-dependent density functional theory approach*, Journal of Chemical Physics 141, 124305 (2014) (https://doi.org/10.1063/1.4895971)

452. Z. Hu, J. Autschbach, and L. Jensen, *Simulating Third-Order Nonlinear Optical Properties Using Damped Cubic Response Theory within Time-Dependent Density Functional Theory*, Journal of Chemical Theory and Computation 12, 1294 (2016) (https://doi.org/10.1021/acs.jctc.5b01060)

453. J.I. Rodríguez, .W. Ayers, A.W. Götz, and F.L. Castillo-Alvarado, *Virial theorem in the Kohn-Sham density-functional theory formalism: Accurate calculation of the atomic quantum theory of atoms in molecules energies*, Journal of Chemical Physics 131, 021101 (2009) (https://doi.org/10.1063/1.3160670)

454. J.S.M. Anderson, J.I. Rodriguez, P.W. Ayers, and A.W. Götz, *Relativistic (SR-ZORA) Quantum Theory of Atoms in Molecules Properties*, Journal of Computational Chemistry 82, 81 (2017) (https://doi.org/10.1002/jcc.24520).

455. P.L.A. Popelier, *Atoms in Molecules An Introduction*, Pearson Education, Harlow, 2000.

456. P.W Ayers, S. Jenkins, *Bond metallicity measures*, Comput. Theor. Chem. 1053, 112 (2015) (https://doi.org/10.1016/j.comptc.2014.10.040).

457. V. Tognetti, L. Joubert, *Density functional theory and Bader's atoms-in-molecules theory: towards a vivid dialogue*, Phys. Chem. Chem. Phys. 16, 14539 (2014) (https://doi.org/10.1039/C3CP55526G).

458. Y.A. Abramov, *On the Possibility of Kinetic Energy Density Evaluation from the Experimental Electron-Density Distribution*, Acta Cryst. A53, 264 (1997) (https://doi.org/10.1107/S010876739601495X).

459. P. Geerlings, F. de Proft, W. Langenaecker, *Conceptual Density Functional Theory*, Chem. Rev. 103, 1793 (2003) (https://doi.org/10.1021/cr990029p).

460. F. Zielinski, V. Tognetti, L. Joubert, *Condensed descriptors for reactivity: A methodological study*, Chem. Phys. Lett. 527, 67 (2012) (https://doi.org/10.1016/j.cplett.2012.01.011).

461. V. Tognetti, C. Morell, L. Joubert, *Atomic electronegativities in molecules*, Chem. Phys. Lett. 635, 111 (2015) (https://doi.org/10.1016/j.cplett.2015.05.057).

462. X. Fradera, M.A Austen, *The Lewis model and beyond*, R.F.W. Bader, J. Phys. Chem. A 103, 304 (1999) (https://doi.org/10.1021/jp983362q).

463. J. Poater, M. Solà, M. Duran, X. Fradera, *The calculation of electron localization and delocalization indices at the Hartree–Fock, density functional and post-Hartree–Fock levels of theory*, Theor. Chem. Acc. 107, 362 (2002) (https://doi.org/10.1007/s00214-002-0356-8).

464. P. Geerlings, S. Fias, Z. Boisdenghien, F. De Proft, *Conceptual DFT: chemistry from the linear response function*, Chem. Soc. Rev. 43, 4989 (2014) (https://doi.org/10.1039/C3CS60456J).

465. M. Atanasov, C.A. Daul, C. Rauzy, *New insights into the effects of covalency on the ligand field parameters: a DFT study*, Chemical Physics Letters 367, 737 (2003) (https://doi.org/10.1016/S0009-2614(02)01762-1)

466. M. Atanasov, C.A. Daul, C. Rauzy, *A DFT Based Ligand Field Theory*, Structure & Bonding 106, 97 (2004) (https://doi.org/10.1007/b11308)

467. A. Borel, C.A. Daul, L. Helm, *Hybrid ligand-field theory/quantum chemical calculation of the fine structure and ZFS in lanthanide(III) complexes*, Chemical Physics Letters 383, 584 (2004) (https://doi.org/10.1016/j.cplett.2003.11.082)

468. M. Atanasov, E.J. Baerends, P. Beattig, R. Bruyndockx, C. Daul, C. Rauzy, *The calculation of ESR parameters by density functional theory: the g- and A-tensors of Co(acacen)*, Chemical Physics Letters 399, 433 (2004) (https://doi.org/10.1016/j.cplett.2004.10.041)

469. M. Atanasov, C.A. Daul, *A DFT based ligand field model for magnetic exchange coupling in transition metal dimmer complexes: (i) principles*, Chemical Physics Letters 379, 209 (2003) (https://doi.org/10.1016/S0009-2614(03)01325-3)

470. M. Atanasov, C.A. Daul, *A DFT based ligand field model for magnetic exchange coupling in transition metal dimer complexes:: (ii) application to magnetic systems with more than one unpaired electron per site*, Chemical Physics Letters 381, 584 (2003) (https://doi.org/10.1016/j.cplett.2003.10.024)

471. F. Senn, C.A. Daul, *Calculation of :sup:'59'Co shielding tensor :math:'sigma' using LF-DFT*, Journal of Molecular Structure: THEOCHEM 954, 105 (2010) (https://doi.org/10.1016/j.theochem.2010.02.027)

472. C. Daul, *Non-empirical Prediction of the Photophysical and Magnetic Properties of Systems with Open d- and f-Shells Based on Combined Ligand Field and Density Functional Theory (LFDFT)*, Chimia 68 (2014) (https://doi.org/10.2533/chimia.2014.633)

473. H. Ramanantoanina, W. Urland, F. Cimpoesu, and C. Daul, *Ligand field density functional theory calculation of the 4f^2 → 4f^1 5d^1 transitions in the quantum cutter* Cs$_2$ KYF$_6$ Pr^{3+}, Physical Chemistry Chemical Physics 15, 13902 (2013) (https://doi.org/10.1039/C3CP51344K).

474. H. Ramanantoanina, W. Urland, A. Garcia-Fuente, F. Cimpoesu, and C. Daul, *Ligand field density functional theory for the prediction of future domestic lighting*, Physical Chemistry Chemical Physics 16, 14625 (2014) (https://doi.org/10.1039/C3CP55521F).

475. H. Ramanantoanina, M. Sahnoun, A. Barbiero, M. Ferbinteanu, F. Cimpoesu, *Development and applications of the LFDFT: the non-empirical account of ligand field and the simulation of the f–d transitions by density functional theory*, Physical Chemistry Chemical Physics 17, 18547 (2015) (https://doi.org/10.1039/C5CP02349A).

476. V.P. Nicu, J. Neugebauer and E.J. Baerends, *Effects of Complex Formation on Vibrational Circular Dichroism Spectra*, Journal of Physical Chemistry A 112, 6978 (2008) (https://doi.org/10.1021/jp710201q)

477. M.A.J. Koenis, W.J. Buma and V.P. Nicu, *Analysis of VCD spectra of peptides: a generalised coupled oscillator approach using VCDtools*, *manuscript in preparation, (2018)*

478. V.P. Nicu, *Revisiting an old concept: the coupled oscillator model for VCD. Part 1: the generalised coupled oscillator mechanism and its intrinsic connection to the strength of VCD signals*, Physical Chemistry Chemical Physics 18, 21202 (2016) (https://doi.org/10.1039/C6CP01282E).

479. Y.C. Park, F. Senn, M. Krykunov, and T. Ziegler, *Self-Consistent Constricted Variational Theory RSCF-CV(∞)-DFT and Its Restrictions To Obtain a Numerically Stable Δ SCF-DFT-like Method: Theory and Calculations for Triplet States*, Journal of Chemical Theory and Computation 12, 5438 (2016) (https://doi.org/10.1021/acs.jctc.6b00333)

480. O. Baseggio, G. Fronzoni, and M. Stener, *A new time dependent density functional algorithm for large systems and plasmons in metal clusters*, Journal of Chemical Physics 143, 024106 (2015) (https://doi.org/10.1063/1.4923368)

481. O. Baseggio, D. Toffoli, G. Fronzoni, M. Stener, L. Sementa, and A. Fortunelli, *Extension of the Time-Dependent Density Functional Complex Polarizability Algorithm to Circular Dichroism: Implementation and Applications to* Ag_8 *and* Au_{38} $(SC_2 H_4 C_6 H_5)_{24}$, Journal of Physical Chemistry C 120, 24335 (2016) (https://doi.org/10.1021/acs.jpcc.6b07323)

482. S. Bernadotte, F. Evers, and C.R. Jacob, *Plasmons in Molecules*, Journal of Physical Chemistry C 117, 1863 (2013) (https://doi.org/10.1021/jp3113073)

483. R.G. Parr, W.T. Yang, *Density functional approach to the frontier-electron theory of chemical reactivity*, J. Am. Chem. Soc. 106, 4049 (1984) (https://doi.org/10.1021/ja00326a036)

484. C. Morell, A. Grand, A. Toro-Labbé, *New Dual Descriptor for Chemical reactivity*, J. Phys. Chem. A 109, 205 (2005) (https://doi.org/10.1021/jp046577a)

485. A. Klamt, V. Jonas, T. Bürger and J.C. Lohrenz, *Refinement and Parametrization of COSMO-RS.* J. Phys. Chem. A 102, 5074 (1998) (https://doi.org/10.1021/jp980017s)

486. D.Schlüns, M. Franchini, A. W. Götz, J. Neugebauer, C. R. Jacob ,L. Visscher, *Analytical gradients for subsystem density functional theory within the slater-function-based amsterdam density functional program*, J. Comput. Chem. 38, 4 (2017) (https://doi.org/10.1002/jcc.24670)

487. Z. Hu, D.V. Chulhai, and L. Jensen, *Simulating Surface-Enhanced Hyper-Raman Scattering Using Atomistic Electrodynamics-Quantum Mechanical Models*, Journal of Chemical Theory and Computation 12, 5968 (2016) (https://doi.org/10.1021/acs.jctc.6b00940)

488. S.Lehtola, C. Steigemann, M.J.T. Oliveira, M.A.L. Marques, *Recent developments in LibXC – A comprehensive library of functionals for density functional theory*, SoftwareX 7, 1 (2018) (https://doi.org/10.1016/j.softx.2017.11.002)

489. F.M. Bickelhaupt, M. Solà, C. Fonseca Guerra, *Highly polar bonds and the meaning of covalency and ionicity – structure and bonding of alkali metal hydride oligomers*, Faraday Discussions 135, 451 (2007) (https://doi.org/10.1039/B606093E)

490. M. Mitoraj, and A. Michalak, *Natural orbitals for chemical valence as descriptors of chemical bonding in transition metal complexes*, Journal of Molecular Modeling 13, 347 (2007) (https://doi.org/10.1007/s00894-006-0149-4)

491. J.Martínez, *Local Reactivity Descriptors from Degenerate Frontier Molecular Orbitals*, Chem. Phys. Lett. 478, 310-322 (2009) (https://doi.org/10.1016/j.cplett.2009.07.086)

492. P. Ramos, M. Pavanello, *Low-lying excited states by constrained DFT*, Journal of Chemical Physics 148, 144103 (2018) (https://doi.org/10.1063/1.5018615)

493. M.J.G. Peach, P. Benfield, T. Helgaker, and D.J. Tozer, *Excitation energies in density functional theory: An evaluation and a diagnostic test*, Journal of Chemical Physics 128, 044118 (2008) (https://doi.org/10.1063/1.2831900)

494. C.A. Guido, P. Cortona, B. Mennucci, and C. Adamo, *On the Metric of Charge Transfer Molecular Excitations: A Simple Chemical Descriptor*, Journal of Chemical Theory and Computation 9, 3118 (2013) (https://doi.org/10.1021/ct400337e)

495. F. Plasser, and H. Lischka, *Analysis of Excitonic and Charge Transfer Interactions from Quantum Chemical Calculations*, Journal of Chemical Theory and Computation 8, 2777 (2012) (https://doi.org/10.1021/ct300307c)

496. F. Plasser, M. Wormit, and A. Dreuw, *New tools for the systematic analysis and visualization of electronic excitations. I. Formalism*, Journal of Chemical Physics 141, 024106 (2008) (https://doi.org/10.1063/1.4885819)

497. F. Plasser, S.A. Bäppler, Wormit, and A. Dreuw, *New tools for the systematic analysis and visualization of electronic excitations. II. Applications*, Journal of Chemical Physics 141, 024107 (2008) (https://doi.org/10.1063/1.4885820)

498. S.A. Mewes, J.-M. Mewes, A. Dreuw, and F. Plasser, *Excitons in poly(para phenylene vinylene): a quantum-chemical perspective based on high-level ab initio calculations*, Physical Chemistry Chemical Physics 18, 2548 (2016) (https://doi.org/10.1039/c5cp07077e).

499. M.A. Blanco, A.M. Pendás and E. Francisco, *Interacting Quantum Atoms: a correlated energy decomposition scheme based on the quantum theory of atoms in molecules*, Journal of Chemical Theory and Computation 1, 1096 (2005) (https://doi.org/10.1021/ct0501093).

500. A.M. Pendás, MA. Blanco and E. Francisco, *A molecular energy decomposition scheme for atoms in molecules*, Journal of Chemical Theory and Computation 2, 90 (2006) (https://doi.org/10.1021/ct0502209).

501. V. Tognetti and L. Joubert, *On Atoms-in-Molecules Energies from Kohn–Sham calculations*, ChemPhysChem 18, 2675 (2017) (https://doi.org/10.1002/cphc.201700637).

502. V. Tognetti and L. Joubert, *On the physical role of exchange in the formation of an intramolecular bond path between two electronegative atoms*, Journal of Chemical Physics 138, 024102 (2013) (https://doi.org/10.1063/1.4770495).

503. G. Hoffmann, V. Tognetti and L. Joubert, *Can molecular and atomic descriptors predict the electrophilicity of Michael acceptors?*, J. Mol. Model. 24, 281 (2018) (https://doi.org/10.1007/s00894-018-3802-9).

504. V. Tognetti, C. Morell, L. Joubert, *Quantifying electro/nucleophilicity by partitioning the dual descriptor*, J. Comput. Chem. 36, 649 (2015) (https://doi.org/10.1002/jcc.23840).

505. J. Sun, B. Xiao, A. Ruzsinszky, *Communication: Effect of the orbital-overlap dependence in the meta generalized gradient approximation*, Journal of Chemical Physics 137, 051101 (2012) (https://doi.org/10.1063/1.4742312).

506. J. Sun, R. Haunschild, B. Xiao, I.W. Bulik, G.E. Scuseria, J.P. Perdew, *Semilocal and hybrid meta-generalized gradient approximations based on the understanding of the kinetic-energy-density dependence*, Journal of Chemical Physics 138, 044113 (2013) (https://doi.org/10.1063/1.4789414).

507. J. Sun, A. Ruzsinszky, J.P. Perdew, *Strongly Constrained and Appropriately Normed Semilocal Density Functional*, Physical Review Letters 115, 036402 (2015) (https://doi.org/10.1103/PhysRevLett.115.036402).

508. J.G. Brandenburg, J.E. Bates, J. Sun, and J.P. Perdew, *Benchmark tests of a strongly constrained semilocal functional with a long-range dispersion correction*, Physical Review B 94, 115144 (2016) (https://doi.org/10.1103/PhysRevB.94.115144).

509. C. Morell, A. Grand, A. Toro-Labbé, *Is hyper-hardness more chemically relevant than expected ?*, J. Mol. Model. 19, 2893 (2013) (https://doi.org/10.1007/s00894-013-1778-z).

510. R.G. Parr, L.v. Szenpály, S. Liu, *Electrophilicity index*, J. Am. Chem. Soc. 121, 1922 (1999) (https://doi.org/10.1021/ja983494x)

511. P.W. Ayers, J.S.M. Anderson, J.I. Rodrigueza, Z. Jaweda, *Indices for predicting the quality of leaving groups*, Phys. Chem. Chem. Phys. 7, 1918 (2005) (https://doi.org/10.1039/b500996k)

512. J.L. Gázquez, A. Cedillo, A. Vela, *Electrodonating and electroaccepting powers*, J. Phys. Chem. A 111, 1966 (2007) (https://doi.org/10.1021/jp065459f)

513. P.K. Chattaraj, A. Chakraborty, S. Giri, *Net electrophilicity*, J. Phys. Chem. A 113, 10068 (2009) (https://doi.org/10.1021/jp904674x)

514. F. Guégan, P. Mignon, V. Tognetti, L. Joubert, C. Morell, *Dual descriptor and molecular electrostatic potential: complementary tools for the study of the coordination chemistry of ambiphilic ligands*, Phys. Chem. Chem. Phys. 16, 15558 (2014) (https://doi.org/10.1039/c4cp01613k)

515. A. Toro-Labbé, P. Jaque, J.S. Murray and P. Politzer, *Connection between the average local ionization energy and the Fukui function*, Chem. Phys. Lett. 407, 143 (2005) (https://doi.org/10.1016/j.cplett.2005.03.041)

516. J. Padmanabhan, R. Parthasarathi, M. Elango, V. Subramanian, B.S. Krishnamoorthy, S. Gutierrez-Oliva, A. Toro-Labbé, D. R. Roy, P. K. Chattaraj, *Multiphilic descriptor for chemical reactivity and selectivity*, J. Phys. Chem. A 111, 130 (2007) (https://doi.org/10.1021/jp0718909)

517. P.K. Chattaraj, B. Maiti, U. Sarkar, *Philicity: A unified treatment of chemical reactivity and selectivity*, J. Phys. Chem. A 107, 4973 (2003) (https://doi.org/10.1021/jp034707u)

518. C. Morell C. , J.L. Gázquez, A. Vela, F. Guégan, H. Chermette, *Revisiting electroaccepting and electrodonating powers: proposals for local electrophilicity and local nucleophilicity descriptors*, Phys. Chem. Chem. Phys. 16, 26832 (2014) (https://doi.org/10.1039/c4cp03167a)

519. S. Fias, F. Heidar-Zadeh, P. Geerlings, P.W. Ayers, *Chemical transferability of functional groups follows from the nearsightedness of electronic matter*, PNAS 114, 11633 (2017) (https://doi.org/10.1073/pnas.1615053114)

520. H. Chermette, *Chemical reactivity indexes in density functional theory*, J. Comput. Chem. 20, 129 (1999) (https://doi.org/10.1002/(SICI)1096-987X(19990115)20:1%3C129::AID-JCC13%3E3.0.CO;2-A)

KEYWORDS