BAND Manual

ADF Modeling Suite 2018

www.scm.com

Sep 18, 2018

CONTENTS

1 General **1**
- 1.1 Introduction . . . 1
- 1.2 Feature List . . . 1
  - 1.2.1 Model Hamiltonians . . . 1
  - 1.2.2 Structure and Reactivity . . . 2
  - 1.2.3 Spectroscopy and Properties . . . 2
  - 1.2.4 Charge transport . . . 2
  - 1.2.5 Analysis . . . 2
- 1.3 What's new in Band 2018 . . . 3
  - 1.3.1 New features . . . 3
  - 1.3.2 AMS: a new driver program . . . 3
  - 1.3.3 Renaming / restructuring of input keys . . . 4
- 1.4 Input . . . 5
  - 1.4.1 General remarks on input structure and parsing . . . 6
  - 1.4.2 Keys . . . 7
  - 1.4.3 Blocks . . . 8
  - 1.4.4 Units . . . 9

2 Exploring the PES with AMS **11**
- 2.1 Input Geometry . . . 11
- 2.2 Single Point . . . 11
- 2.3 Geometry Optimization . . . 11
- 2.4 Transition State Search . . . 11
- 2.5 Linear Transit and other PES Scan . . . 11
- 2.6 Molecular Dynamics . . . 12
- 2.7 Nuclear Gradients and Stress Tensor . . . 12

3 Model Hamiltonians **13**
- 3.1 Density Functional (XC) . . . 13
  - 3.1.1 XC functionals . . . 13
    - LDA/GGA/metaGGA . . . 14
    - Dispersion Correction . . . 15
    - Model Potentials . . . 16
    - Non-Collinear Approach . . . 17
    - LibXC Library Integration . . . 17
    - Range-separated hybrid functionals . . . 18
    - Defaults and special cases . . . 19
  - 3.1.2 GGA+U . . . 19
  - 3.1.3 OEP . . . 20
- 3.2 Relativistic Effects and Spin . . . 21

3.2.1 Spin polarization . . . 21
3.2.2 Relativistic Effects . . . 22
3.3 Solvation . . . 22
3.3.1 COSMO: Conductor like Screening Model and the Solvation-key . . . 22
3.3.2 Additional keys for periodic systems . . . 27
3.4 Electric and Magnetic Fields . . . 28
3.4.1 Electric Field . . . 28
3.4.2 Magnetic Field . . . 28
3.4.3 Atoim-wise fuzzy potential . . . 30
3.5 Nuclear Model . . . 30

4 Accuracy and Efficiency **31**
4.1 Basis set . . . 31
4.1.1 Basis input block . . . 32
4.1.2 Which basis set should I use? . . . 32
4.1.3 Available Basis Sets . . . 35
4.1.4 More Basis input options . . . 36
4.1.5 Confinement of basis functions . . . 36
4.1.6 Manually specifying AtomTypes (expert option) . . . 37
4.1.7 Basis Set Superposition Error (BSSE) . . . 39
4.2 K-Space . . . 39
4.2.1 KSpace input block . . . 39
Regular K-Space grid . . . 40
Symmetric K-Space grid (tetrahedron method) . . . 41
4.2.2 Recommendations for k-space . . . 41
4.3 Numerical Integration . . . 43
4.3.1 Becke Grid . . . 43
4.3.2 Radial grid of NAOs . . . 44
4.3.3 Voronoi grid (deprecated) . . . 45
4.4 Density Fitting . . . 45
4.4.1 Zlm Fit . . . 45
Expert options . . . 46
4.4.2 STO Fit (Deprecated) . . . 48
4.5 Hartree–Fock RI . . . 48
4.6 Self Consistent Field (SCF) . . . 49
4.6.1 SCF block . . . 49
4.6.2 Convergence . . . 51
4.6.3 DIIS . . . 52
4.6.4 Multi secant . . . 54
4.6.5 DIRIS . . . 55
4.7 More Technical Settings . . . 55
4.7.1 Linear Scaling . . . 55
4.7.2 Dependency . . . 55
4.7.3 Screening . . . 56
4.7.4 Direct (on the fly) calculation of basis and fit . . . 57
4.7.5 Fermi energy search . . . 57
4.7.6 Block size . . . 58

5 Spectroscopy and Properties **59**
5.1 Frequencies and Phonons . . . 59
5.2 Elastic Tensor . . . 59
5.3 Optical Properties: Time-Dependent Current DFT . . . 59
5.3.1 Insulators, semiconductors and metals . . . 59
5.3.2 Frequency dependent kernel . . . 60

5.3.3 EELS . . . 60
5.3.4 Input Options . . . 60
NewResponse . . . 61
OldResponse . . . 65
5.4 ESR/EPR . . . 67
5.5 Nuclear Quadrupole Interaction (EFG) . . . 68
5.6 NMR . . . 69
5.7 Effective Mass . . . 70
5.8 Properties at Nuclei . . . 71
5.9 X-Ray Form Factors . . . 71

6 Analysis **73**
6.1 Density of States (DOS) . . . 73
6.1.1 Gross populations . . . 75
6.1.2 Overlap populations . . . 75
6.2 Band Structure . . . 76
6.2.1 User-defined path in the Brillouin zone . . . 78
6.2.2 Definition of the Fat Bands . . . 78
6.2.3 Band Gap . . . 78
6.3 Charges . . . 79
6.3.1 Default Atomic Charge Analysis . . . 79
6.3.2 Bader Analysis (AIM) . . . 79
6.4 Fragments . . . 81
6.5 Energy Decomposition Analysis . . . 82
6.5.1 Periodic Energy Decomposition Analysis (PEDA) . . . 82
6.5.2 Periodic Energy Decomposition Analysis and natural orbitals of chemical valency (PEDA-NOCV) . . . 83
6.6 3D field visualization with BAND . . . 83

7 Electronic Transport (NEGF) **91**
7.1 Transport with NEGF in a nutshell . . . 91
7.1.1 Self consistency . . . 92
7.1.2 Contour integral . . . 93
7.1.3 Gate potential . . . 93
7.1.4 Bias potential . . . 94
7.2 Workflow . . . 94
7.3 Input options . . . 95
7.3.1 SGF Input options . . . 95
7.3.2 NEGF Input options (no bias) . . . 95
7.3.3 NEGF Input options (with bias) . . . 98
7.3.4 NEGF Input options (alignment) . . . 100
7.4 Troubleshooting . . . 101
7.5 Miscellaneous remarks on BAND-NEGF . . . 101
7.5.1 Store tight-binding Hamiltonian . . . 101

8 Expert Options **103**
8.1 Restarts . . . 103
8.1.1 Restart key . . . 103
8.1.2 Grid . . . 104
8.1.3 Plots of the density, potential, and many more properties . . . 106
8.1.4 Orbital plots . . . 107
8.1.5 Induced Density Plots of Response Calculations . . . 107
8.1.6 NOCV Orbital Plots . . . 108
8.1.7 NOCV Deformation Density Plots . . . 109

8.1.8 LDOS (STM) . . . 109
8.1.9 Save . . . 110
8.2 Symmetry . . . 110
8.2.1 Symmetry breaking for SCF . . . 112
8.3 Advanced Occupation Options . . . 112

9 Troubleshooting **115**
9.1 Recommendations . . . 115
9.1.1 Model Hamiltonian . . . 115
Relativistic model . . . 115
XC functional . . . 115
9.1.2 Technical Precision . . . 115
9.1.3 Performance . . . 116
Reduced precision . . . 116
Memory usage . . . 116
Reduced basis set . . . 117
Frozen core for 5d elements . . . 117
9.2 Troubleshooting . . . 117
9.2.1 SCF does not converge . . . 117
9.2.2 Geometry does not converge . . . 118
9.2.3 Negative frequencies in phonon spectra . . . 118
9.2.4 Basis set dependency . . . 118
Using confinement . . . 118
Removing basis functions . . . 119
9.2.5 Frozen core too large . . . 119
9.3 Various issues . . . 120
9.3.1 Understanding the logfile . . . 120
9.3.2 Breaking the symmetry . . . 121
9.3.3 Labels for the basis functions . . . 122
9.3.4 Reference and Startup Atoms . . . 122
9.3.5 Numerical Atoms and Basis functions . . . 122
9.4 Warnings . . . 123
9.4.1 Warnings specific to periodic codes (BAND, DFTB) . . . 123

10 Examples **125**
10.1 Introduction . . . 125
10.2 Model Hamiltonians . . . 127
10.2.1 Example: Spin polarization: antiferromagnetic iron . . . 127
10.2.2 Example: Applying a Magnetic Field . . . 128
10.2.3 Example: Graphene sheet with dispersion correction . . . 129
10.2.4 Example: H on perovskite with the COSMO solvation model . . . 131
10.2.5 Example: Applying a homogeneous electric field . . . 132
10.2.6 Example: Finite nucleus . . . 134
10.2.7 Example: Fixing the Band gap of NiO with GGA+U . . . 136
10.2.8 Example: Fixing the band gap of ZnS with the TB-mBJ model potential . . . 137
10.3 Precision and performance . . . 139
10.3.1 Example: Convenient way to specify a basis set . . . 139
10.3.2 Example: Tuning precision and performance . . . 140
10.3.3 Example: Multiresolution . . . 141
10.3.4 Example: BSSE correction . . . 143
10.4 Restarts . . . 146
10.4.1 Example: Restart the SCF . . . 146
10.4.2 Example: Restart SCF for properties calculation . . . 150
10.4.3 Example: Properties on a grid . . . 152

10.5 NEGF . . . 154
10.5.1 Example: Main NEGF flavors . . . 154
10.5.2 Example: NEGF with bias . . . 161
10.5.3 Example: NEGF using the non-self consistent method . . . 163
10.6 Structure and Reactivity . . . 168
10.6.1 Example: NaCl: Bulk Crystal . . . 168
10.6.2 Example: Transition-State search using initial Hessian . . . 170
10.6.3 Example: Atomic energies . . . 172
10.6.4 Example: Calculating the atomic forces . . . 176
10.6.5 Example: Optimizing the geometry . . . 177
10.7 Time dependent DFT . . . 179
10.7.1 Example: TD-CDFT for MoS2 Monolayer (NewResponse) . . . 179
10.7.2 Example: TD-CDFT for Copper (NewResponse) . . . 182
10.7.3 Example: TDCDFT: Plot induced density (NewResponse) . . . 183
10.7.4 Example: TD-CDFT for bulk diamond (OldResponse) . . . 185
10.8 Spectroscopy . . . 186
10.8.1 Example: Hyperfine A-tensor . . . 186
10.8.2 Example: Zeeman g-tensor . . . 187
10.8.3 Example: NMR . . . 188
10.8.4 Example: EFG . . . 189
10.8.5 Example: Phonons . . . 190
10.9 Analysis . . . 192
10.9.1 Example: CO absorption on a Cu slab: fragment option and densityplot . . . 192
10.9.2 Example: Grid key for plotting results . . . 198
10.9.3 Example: H2 on [PtCl4]2-: charged molecules and PEDA . . . 200
10.9.4 Example: CO absorption on a MgO slab: fragment option and PEDA . . . 203
10.9.5 Example: CO absorption on a MgO slab: fragment option, PEDA and PEDANOCV . . . 206
10.9.6 Example: Bader analysis . . . 212
10.9.7 Example: Properties at nuclei . . . 213
10.9.8 Example: Band structure plot . . . 214
10.9.9 Example: Effective Mass (electron mobility) . . . 216
10.9.10 Example: Generating an Excited State with and Electron Hole . . . 218
10.10 List of Examples . . . 219

11 Required Citations **221**
11.1 General References . . . 221
11.2 Feature References . . . 221
11.2.1 Geometry optimization . . . 222
11.2.2 TDDFT . . . 222
11.2.3 Relativistic TDDFT . . . 222
11.2.4 Vignale Kohn . . . 222
11.2.5 NMR . . . 223
11.2.6 ESR . . . 223
11.2.7 NEGF . . . 223
11.3 External programs and Libraries . . . 223

12 References **225**

13 Keywords **231**
13.1 Links to manual entries . . . 231
13.2 Summary of all keywords . . . 232

Index **281**

CHAPTER

ONE

GENERAL

1.1 Introduction

The periodic DFT program **BAND** can be used for calculations on periodic systems, i.e. polymers, slabs and crystals. It uses Density Functional Theory (DFT) in the Kohn-Sham approach. BAND shares many of the core algorithms with ADF, although important differences remain (a noteworthy difference is that BAND uses numerical atomic orbitals as basis functions).

This User's Manual describes how to use the program, how input is structured, what files are produced, and so on. The *Examples section* (page 125) explains the most popular features in detail, by commenting on the input and output files in the $ADFHOME/examples/band directory.

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 **installation** of BAND is explained in the Installation manual. There you can also find information about the license file, which you need to run the program.

Graphical User Interface (GUI) tutorials: GUI overview tutorials, BAND-GUI tutorials

This manual and other documentation is available at http://www.scm.com. As mentioned in the license agreement, it is mandatory, for publications in which BAND has been used, to cite the *lead references* (page 221).

1.2 Feature List

1.2.1 Model Hamiltonians

- *XC energy functionals and potentials* (page 13)
  - *LDA* (page 14), *GGA* (page 14), *meta-GGA* (page 15), *Model potentials* (page 16)
  - *Range-separated Hybrids* (page 18)
  - *GGA+U (Hubbard)* (page 19)
  - *LibXC library* (page 17)
  - *Grimme dispersion corrections* (page 15)
- *Relativistic effects: ZORA and spin-orbit coupling* (page 22) (including non-collinear magnetization)
- *COSMO* (page 22) solvation model
- Homogeneous *electric* (page 28) and *magnetic* (page 28) fields

1.2.2 Structure and Reactivity

- Geometry optimization, transition state search, linear transit, PES-scan, molecular dynamics via **AMS**. See the AMS Manual for details.
- Formation energy with respect to isolated atoms (which are computed with a fully numerical Herman-Skillman type subprogram)

1.2.3 Spectroscopy and Properties

- Normal modes, phonon dispersion curves (and related thermodynamic properties) and elastic tensor via **AMS** See the AMS Manual for details.
- Frequency-dependent dielectric function of systems periodic in one, two and three dimensions in the *Time-dependent Current-DFT* (page 59) (TD-CDFT) formalism
- *ESR and EPR* (page 67) (electron paramagnetic resonance) and *EFG* (page 68) (Nuclear Quadrupole Interaction)
- *Form factors* (page 71) (X-ray structures)
- *NMR shielding tensor* (page 69)

1.2.4 Charge transport

- *Non-Equilibrium Green's Function* (page 91) (NEGF) for calculating transmission function and current
- *Effective mass* (page 70) for electrons and holes mobility

1.2.5 Analysis

- *Various Atomic charges* (page 79), including Mulliken, Hirshfeld, CM5 and Voronoi
- Mulliken populations for basis functions, overlap populations between atoms or between basis functions
- *Densities-of-States* (page 73): DOS, PDOS and OPWDOS/COOP
- *Local Densities-of-States* (page 109) LDOS (STM images)
- *3D filed plotting of various properties* (page 83), such as orbitals (Bloch-waves), deformation densities, Coulomb potentials, ...
- *Band Structure plot* (page 76) along edges of the Brillouin zone
- *Fragment* (page 81) orbitals and a Mulliken type population analysis in terms of the fragment orbitals
- *Quantum Theory of Atoms In Molecules* (page 79) (QT-AIM, aka Bader Analysis). Atomic charges and critical points
- Electron Localization Function (*ELF* (page 106))
- Fragment based Periodic Energy Decomposition Analysis (*PEDA* (page 82))
- PEDA combined with Natural Orbitals for Chemical Valency (NOCV) to decompose the orbital relaxation (*PEDA-NOCV* (page 83))

1.3 What's new in Band 2018

1.3.1 New features

- Elastic tensor and related properties (e.g. Bulk modulus) (via AMS)
- Molecular dynamics (via AMS)
- Linear transit and PES scan (via AMS)
- Geometry optimization under pressure (via AMS)
- It's now possible to *specify units in the input file* (page 9)
- *Effective mass* (page 70) for arbitrary k-points
- Multi-Secant method for *SCF* (page 49) convergence

1.3.2 AMS: a new driver program

Important: In the 2018 release of the ADF Modeling Suite we introduced a new driver program call **AMS**. We recommend you to first read the General section of the AMS Manual

If you use BAND exclusively via the Graphical User Interface (GUI), this change should not create any issues. If, on the other hand, you create input files *by hand* (or you use BAND via PLAMS), then you should be aware that **shell scripts for BAND-2017 and previous versions are not compatible with BAND-2018 and have to be adjusted to the new setup.**

The example below shows how a shell script for BAND-2017 is converted to BAND-2018.

BAND-2017 shell script (obsolete):

```
#!/bin/sh

# This is a shell script for BAND-2017 which will not work for BAND-2018

$ADFBIN/band <<eor

   Units
     Length Angstrom
   End

   Atoms
      H 0.0 0.0 0.0
      H 0.9 0.0 0.0
   End

   GeoOpt
      Converge grad=1e-4
   End

   BasisDefaults
      BasisType DZP
   End

   XC
      GGA PBE
```

```
   End
eor
```

BAND-2018 shell script:

```
#!/bin/sh

# This is a shell script for BAND-2018

# The executable '$ADFBIN/band' is no longer present.
# You should use '$ADFBIN/ams' instead.

$ADFBIN/ams <<eor
   # Input options for the AMS driver:

   System
      Atoms
         H 0.0 0.0 0.0
         H 0.9 0.0 0.0
      End
   End

   Task GeometryOptimization

   GeometryOptimization
      Convergence gradients=1e-4
   End

   # The input options for Band, which are described in this manual,
   # should be specified in the 'Engine Band' block:

   Engine Band
      Basis
         Type DZP
      End

      XC
         GGA PBE
      End
   EndEngine
eor
```

1.3.3 Renaming / restructuring of input keys

The input syntax of several option has changed from the 2017 to the 2018 version of BAND. This is an incomplete list of key/blocks for which the input syntax has changed:

BAND-2017 key	BAND-2018 key / comments
Accuracy	Option removed (use *NumericalQuality* (page 31) instead)
AIMCriticalPoints	Added 'Enabled' sub-key to *AIMCriticalPoints* (page 80)
ATensor	Added 'Enabled' sub-key to *ATensor* (page 67)
AtomProps	Feature moved to AMS
Atoms	'Atoms' is now part of AMS
BasisDefaults	*Basis* (page 32)
BZStruct	*BandStructure* (page 76)
Charge	'Charge' is now part of AMS
Constraints	Constraints is now part of AMS
coordinates	This is now part of AMS
Define	*None* (option removed)
EffectiveMass	Added 'Enabled' sub-key and removed UniqueKPoints to *EffectiveMass* (page 70)
EFG	Added 'Enabled' sub-key to *EFG* (page 68)
ESR	Added 'Enabled' sub-key to *ESR* (page 68)
FractionalCoords	This is now part of AMS
GeoOpt	GeometryOptimization (now part of AMS)
GridBasedAIM	Added 'Enabled' sub-key to *GridBasedAIM* (page 79)
KSpace	Several sub-keys of the *KSpace* (page 39) block changed
Lattice	Lattice is now part of AMS
NMR	Added 'Enabled' sub-key to *NMR* (page 69)
OldResponse	Added 'Enabled' sub-key to *OldResponse* (page 65)
PhononConfig	NumericalPhonons (now part of AMS)
Relativistic	*Relativity* (page 22)
RunType	Task (now part of AMS)
Symmetry	*UseSymmetry* (page 111) and *SubSymmetry* (page 111)
Units	See *units in the input file* (page 9)

1.4 Input

The input options for Band are specified in a text file consisting of a series of key-value pairs, possibly nested in blocks. The input is usually embedded in an executable shell script. This is the content of a typical shell script for running a Band calculation:

```
#!/bin/sh

$ADFBIN/ams <<eor
   # This is the beginning of the input.
   # The input consists of key-value pairs and blocks.
   # Here we define the input option for the AMS driver:

   Task GeometryOptimization

   System
      Atoms
         H 0.0 0.0 0.0
         H 0.9 0.0 0.0
      End
   End

   # Next comes the Band "Engine" block. The input options for Band, which are
   # described in this manual, should be specified in this block:
```

```
   Engine Band
      Basis
         Type DZP
      End

      XC
         GGA PBE
      End
   EndEngine
eor
```

To run the calculation above from command-line you should:

1. Create a text file called, for example, `test.run` and copy-paste the content of the script above
2. Make the script executable by typing in your shell `chmod u+x test.run`
3. Execute the script and redirect the output to a file: `./test.run > out`

The program will create a directory called `ams.results`. Inside it, you will find the *logfile* `ams.log` (which can be used to monitor the progress of the calculation) and the binary results files `ams.rkf` and `band.rkf`. After the calculation is completed, you can examine the output file `out`. For more details, see the AMS documentation.

See also:

The *Examples* (page 125) section contains a large number of input examples.

Important: All options described in this manual should be specified in the Band Engine block:

```
# All Band keywords should be specified inside the 'Engine Band' block
Engine Band
   Basis
      Type DZP
   End

   XC
      GGA PBE
   End
EndEngine
```

1.4.1 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
```

1.4.2 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
```

1.4.3 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:

- Spaces (blanks) between the key, the equal sign and the value are not allowed:

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

  # This is NOT OK:
  BlockName KeyName1 = value1 KeyName2= value2
  ```

- The compact notation can be used only for following types of keys: bool, integer, float and multiple_choice. It should **not** be used for sting, integer_list and float_list keys
- The compact notation cannot be used for blocks with headers
- input units cannot be defined in compact notation

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).

1.4.4 Units

Some keys have a default unit associated (not all keys have units). For such keys, the default unit is mention in the key documentation. One can specify a different unit within square brackets at the end of the line:

```
KeyName value [unit]
```

For example, assuming the key `EnergyThreshold` has as default unit `Hartree`, then the following definitions are equivalent:

```
# Use defaults unit:
EnergyThreshold 1.0

# use eV as unit:
EnergyThreshold 27.211 [eV]

# use kcal/mol as unit:
EnergyThreshold 627.5 [kcal/mol]

# Hartree is the atomic unit of energy:
EnergyThreshold 1.0 [a.u.]
```

Available units:

- **Energy**: `Hartree` (`a.u.`), `eV`, `kJ/mol`, `kcal/mol`, `cm^-1`
- **Length**: `Bohr` (`a.u.`), `Angstrom` (`A`), `nm`, `pm`

CHAPTER

TWO

EXPLORING THE PES WITH AMS

AMS is the new driver program in the 2018 release of the ADF Modeling Suite. The job of AMS is to handle all changes in the simulated system's geometry, e.g. during a geometry optimization or molecular dynamics calculation, using energy and forces calculated by BAND.

The input options for these tasks (including the definition of the input geometry) are described in the AMS User Manual.

Important: We recommend you to read the General section of the AMS Manual

2.1 Input Geometry

The atom-types, atomic coordinates, lattice vectors and total charge are defined in the AMS part of the input. See the System definition section of the AMS manual

2.2 Single Point

See the Tasks section of the AMS manual

2.3 Geometry Optimization

See the Geometry Optimization section of the AMS manual

For optimizations **under pressure** (see AMS manual) we recommend to *disable symmetry* (page 111), use a smaller *frozen core* (page 32), more heavily *confine basis functions* (page 36) and use high *K-Space integration* (page 39).

2.4 Transition State Search

See the Transition State Search section of the AMS manual

2.5 Linear Transit and other PES Scan

See the Linear Transit and other PES Scan section of the AMS manual

2.6 Molecular Dynamics

See the Molecular Dynamics section of the AMS manual

2.7 Nuclear Gradients and Stress Tensor

See the Nuclear Gradients and Stress Tensor section of the AMS manual

CHAPTER

THREE

MODEL HAMILTONIANS

3.1 Density Functional (XC)

The starting point for the XC functional is usually the result for the homogeneous electron gas, after which the so called non-local or generalized gradient approximation (GGA) can be added.

3.1.1 XC functionals

The density functional approximation is controlled by the XC key.

Three classes of XC functionals are supported: LDA, GGA, meta-GGA, and range-separated hybrid functionals. There is also the option to add an empirical dispersion correction. The only ingredient of the LDA energy density is the (local) density, the GGA depends additionally on the gradient of the density, and the meta-GGA has an extra dependency on the kinetic energy density. The range-separated hybrids are explained below in the section *Range-Separated Hybrids* (page 18).

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. The *energy* functional is used for the nuclear gradients (geometry optimization), too. 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:

1. The evaluation of the GGA part (especially for meta-GGAs) in the *potential* is rather time-consuming. 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.
2. 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.

All subkeys of `XC` are optional and may occur twice in the data block: if one wants to specify different functionals for potential and energy evaluations respectively, see above.

```
XC
   {LDA  {Apply}   LDA {Stoll}}
   {GGA  {Apply}   GGA}
   {DiracGGA  GGA}
   {MetaGGA  {Apply}   GGA}
   {Dispersion {s6scaling} {RSCALE=r0scaling} {Grimme3} {BJDAMP} {PAR1=par1}
↪{PAR2=par2} {PAR3=par3} {PAR4=par4}}
   {Model [LB94|TB-mBJ|KTB-mBJ|JTS-MTB-MBJ|GLLB-SC|BGLLB-VWN|BGLLB-LYP]}
```

```
   {SpinOrbitMagnetization [None|NonCollinear|CollinearX|CollinearY|CollinearZ]}
   {LibXC {Functional}}
End
```

The common use is to specify either an LDA or a (meta)GGA line. (Technically it is possible to have an LDA line *and* a GGA line, in which case the LDA part of the GGA functional (if applicable) is replaced by what is specified by the LDA line.)

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 **POSTSCF**. (**default=SCF**)

LDA/GGA/metaGGA

LDA Defines the LDA part of the XC functional and can be any of the following:

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

Xalpha: the scaled (parameterized) 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 (**Default: 0.7**).

VWN: the parameterization of electron gas data given by Vosko, Wilk and Nusair (ref [*1* (page 225)], 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 or GL variety of the LDA form you may include Stoll's correction [*2* (page 225)] 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.

GGA Specifies the GGA part of the XC Functional. It uses derivatives (gradients) of the charge density. 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**: the gradient correction proposed in 1988 by Becke [*3* (page 225)]
- **PW86x**: the correction advocated in 1986 by Perdew-Wang [*4* (page 225)]
- **PW91x**: the exchange correction proposed in 1991 by Perdew-Wang [*5* (page 225)]
- **mPWx**: the modified PW91 exchange correction proposed in 1998 by Adamo-Barone [*27* (page 226)]
- **PBEx**: the exchange correction proposed in 1996 by Perdew-Burke-Ernzerhof [*12* (page 225)]
- **HTBSx**: the HTBS exchange functional [*43* (page 227)]
- **RPBEx**: the revised PBE exchange correction proposed in 1999 by Hammer-Hansen-Norskov [*13* (page 225)]
- **revPBEx**: the revised PBE exchange correction proposed in 1998 by Zhang-Yang [*28* (page 226)]
- **mPBEx**: the modified PBE exchange correction proposed in 2002 by Adamo-Barone [*29* (page 226)]
- **OPTX**: the OPTX exchange correction proposed in 2001 by Handy-Cohen [*30* (page 226)]

For the correlation part the options are:

- **Perdew**: the correlation term presented in 1986 by Perdew [*6* (page 225)]
- **PBEc**: the correlation term presented in 1996 by Perdew-Burke-Ernzerhof [*12* (page 225)]

- **PW91c**: the correlation correction of Perdew-Wang (1991), see [5 (page 225)], [8 (page 225)], [9 (page 225)]
- **LYP**: the Lee-Yang-Parr 1988 correlation correction [7 (page 225)]

Some GGA options define the exchange and correlation parts in one stroke. These are:

- **BP86**: this is equivalent to **Becke** + **Perdew** together
- **PW91**: this is equivalent to **pw91x** + **pw91c** together
- **mPW**: this is equivalent to **mPWx** + **pw91c** together
- **PBE**: this is equivalent to **PBEx** + **PBEc** together
- **HTBS**: this is equivalent to **HTBSx** + **PBEc** together
- **RPBE**: this is equivalent to **RPBEx** + **PBEc** together
- **revPBE**: this is equivalent to **revPBEx** + **PBEc** together
- **mPBE**: this is equivalent to **mPBEx** + **PBEc** together
- **BLYP**: this is equivalent to **Becke** (exchange) + **LYP** (correlation)
- **OLYP**: this is equivalent to **OPTX** (exchange) + **LYP** (correlation)
- **OPBE**: this is equivalent to **OPTX** (exchange) + **PBEc** (correlation) [31 (page 226)]

`DiracGGA` (Expert option!) This key handles which XC functional is used during the Dirac calculations of the reference atoms. A string is expected which is not restricted to names of GGAs but can be LDA-like functionals, too.

Note: In some cases using a GGA functional leads to slow convergence of matrix elements of the kinetic energy operator w. r. t. the `Accuracy` parameter. Then one can use the LDA potential for the calculation of the reference atom instead.

`MetaGGA` Key to select the evaluation of a meta-GGA. A byproduct of this option is that the bonding energies of all known functionals are printed (using the same density). Meta-GGA calculations can be time consuming, especially when active during the SCF.

Self consistency of the meta-GGA is implemented as suggested by Neuman, Nobes, and Handy.[11 (page 225)]

The available functionals of this type are:

- **TPSS**: The 2003 meta-GGA [15 (page 225)]
- **M06L**: The meta-GGA as developed by the Minesota group [16 (page 226)]
- **revTPSS**: The 2009 revised meta-GGA [26 (page 226)]

Note: For Meta-GGA XC functionals, it is recommended to use `small` or `none` *frozen core* (page 32) (the frozen orbitals are computed using LDA and not the selected Meta-GGA)

Dispersion Correction

In BAND parameters for *Grimme3* and *Grimme3 BJDAMP* can be used according to version 3.1 (Rev. 1) of the coefficients, published on the Bonn .

`DISPERSION Grimme3 BJDAMP {PAR1=par1 PAR2=par2 PAR3=par3 PAR4=par4}` If this key is present a dispersion correction (DFT-D3-BJ) by Grimme [42 (page 227)] will be added to the total bonding energy, gradient and second derivatives, where applicable. Parametrizations are implemented e.g. for B3LYP, TPSS, BP86, BLYP, PBE, PBEsol [14 (page 225)] , and RPBE. It has four parameters. One can override these 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
PAR1	s6
PAR2	a1
PAR3	s8
PAR4	a2

DISPERSION Grimme3 {PAR1=par1 PAR2=par2 PAR3=par3} If this key is present a dispersion correction (DFT-D3) by Grimme [*41* (page 227)] 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 [*14* (page 225)], and RPBE, and will be automatically set if one of these functionals is used. For all other functionals, PBE-D3 parameters are used as default. You can explicitly specify the three parameters.

variable	variable on Bonn
PAR1	s6
PAR2	sr,6
PAR3	s8

Dispersion {s6scaling RSCALE=r0scaling} If the DISPERSION keyword is present a dispersion correction will be added to the total bonding energy, where applicable. By default the correction of Grimme is applied.[*32* (page 227)] The term is added to the bonding energies of all printed functionals, here the LDA and a couple of GGAs are meant. The global scaling factor, with which the correction is added, depends on the XC functional used for SCF but it can be modified using the *s6scaling* parameter. The following scaling factors are used (with the XC functional in parentheses): 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 van der Waals radii, used in this implementation, are hard-coded. 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.[*32* (page 227)]

Model Potentials

Model Some functionals give only a potential and have no energy expression. We call such functionals model potentials. In BAND the following model potentials are available:

LB94 With this model the asymptotically correct potential of van Leeuwen and Baerends is invoked.[*10* (page 225)]

TB-mBJ This model potential can be used to correct for the band gap problem with GGAs for bulk systems.[*44* (page 227)] This potential depends on a c-factor for which there is a density dependent automatic expression. However you can override the automatic value by specifying XC%TB_mBJCFactor cfac. In principle: the bigger the value the larger the gap. **KTB-mBJ/JTS-mTB-mBJ** are variations of **TB-mBJ**. The formula for C contains three parameters: A,B, and E. The logic is as follows

potential	A	B	E
TB-mBJ[*44* (page 227)]	-0.012	1.023	0.5
KTB-mBJ[*54* (page 228)]	0.267	0.656	1.0
JTS-mTB-mBJ[*55* (page 228)]	0.4	1.0	0.5

The three parameters (A,B, and E) can be user-defined set as follows:

```
XC
   Model TB_mBJ
   TB_mBJAFactor valA
   TB_mBJBFactor valB
   TB_mBJEFactor valE
End
```

GLLB-SC This functional uses a model for the exchange response potential (based on J. Krieger, Y. Li and G. Iafrate response potential [*72* (page 229)]) from which the derivative discontinuity follows [*49* (page 228)].

This is an accurate functional for band gap predictions and Electric Field Gradient calcualtions. It is also a fast method and a very good compromise between accuracy and computational cost. This functional is composed of the GLLB exchange response potential and the PBESOL exchange hole and the correlation potential [*49* (page 228)].

BGLLB-VWN This functional is a variation of the GLLB-SC functional using the B88 exchange hole potential and the VWN correlation potential. This functional gives good results for Group I-VII and II-VI semi conductors.

BGLLB-LYP This functional is a variation of the GLLB-SC functional using the B88 exchange hole potential and the LYP correlation potential. This functional gives good results for large band gap insulators.

One can change the K parameter for the GLLB functionals with the `GLLBKParameter` key:

```
XC
   Model [GLLB-SC|BGLLB-VWN|BGLLB-LYP]
   GLLBKParameter val
End
```

The default value is K=0.382 (value obtained from the electron gas model in the original publication).

Non-Collinear Approach

`SpinOrbitMagnetization` (**Default=CollinearZ**) Most XC functionals have as one ingredient the spin polarization. Normally the direction of the spin quantization axis is arbitrary and conveniently chosen to be the z-axis. However, in a *spin-orbit* (page 22) calculation the direction matters, and it is arbitrary to put the z-component of the magnetization vector into the XC functional. It is also possible to plug the size of the magnetization vector into the XC functional. This is called the non-collinear approach. There is also the exotic option to choose the quantization axis along the x or y axis. To summarize, the value **NonCollinear** invokes the non-collinear method. The other three option **CollinearX**, **CollinearY** and **CollinearZ** causes either the x, y, or z component to be used as spin polarization for the XC functional.

LibXC Library Integration

`LibXC functional` LibXC is a library of approximate XC functionals, see Ref. [*63* (page 228)]. The development version 3 of LibXC is used. See the LibXC website for the complete list of functionals: http://www.tddft.org/programs/Libxc.

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, VV10, XLYP, XPBE
- **MetaGGA:** B97M-V, M06-L, M11-L, MN12-L, MS0, MS1, MS2, MVS, PKZB, TPSS
- **Hybrids** (**only for non-periodic systems**): 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** (**only for non-periodic systems**): 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** (**for periodic systems, only short range-separated functionals can be used**, see *Range-separated hybrid functionals* (page 18)): CAM-B3LYP, CAMY-B3LYP, HJS-PBE, HJS-PBESOL, HJS-B97X, HSE03, HSE06, LRC_WPBE, LRC_WPBEH, LC-VV10, LCY-BLYP, LCY-PBE, M11, MN12-SX, N12-SX, TUNED-CAM-B3LYP, WB97, WB97X, WB97X-V

Example usage for the MVS functional:

```
XC
  LibXC MVS
End
```

Notes:

- **All electron basis sets should be used** (see `CORE NONE` in section *Basis set* (page 31)).
- For periodic systems only short range-separated functionals can be used (see *Range-separated hybrid functionals* (page 18))
- In case of LibXC the output of the BAND 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 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.

Range-separated hybrid functionals

Short range-separated hybrid functionals, like the **HSE03** functional [*62* (page 228)], can be useful for prediction of more accurate band gaps compared to GGAs. These must be specified via the *LibXC* (page 17) key

```
XC
   LibXC functional {omega=value}
End
```

`functional` The functional to be used. (Incomplete) list of available functionals: **HSE06**, **HSE03**, **HJS-B97X**, **HJS-PBE** and **HJS-PBESOL** (See the LibXC website (http://www.tddft.org/programs/octopus/wiki/index.php/Libxc_functionals) for a complete list of available functionals).

`omega` *Optional*. You can optionally specify the switching parameter omega of the range-separated hybrid. Only possible for the **HSE03** and **HSE06** functionals (See [*62* (page 228)]).

Notes:

- Hybrid functionals can only be used in combination with all-electron basis sets (see `CORE NONE` in section *Basis set* (page 31)).
- The Hartree-Fock exchange matrix is calculated through a procedure known as Resolution of the Identity (RI). See *RIHartreeFock* (page 48) key.
- Regular hybrids (such as B3LYP) and long range-separated hybrids (such as CAM-B3LYP) **cannot** be used in periodic boundary conditions calculations (they can only be used for non-periodic systems).
- There is some confusion in the scientific literature about the value of the switching parameter ω for the HSE functionals. In LibXC, and therefore in BAND, the HSE03 functional uses $\omega = 0.106066$ while the HSE06 functional uses $\omega = 0.11$.

Usage example:

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

Defaults and special cases

- If the XC key is not used, the program will apply only the Local Density Approximation (no GGA terms). The chosen LDA form is then VWN.
- If only a GGA part is specified, omitting the *LDA* subkey, 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.
- Be aware that typing only the subkey *LDA*, without an argument, will activate the VWN form (also if LYP is specified in the GGA part).

3.1.2 GGA+U

A special way to treat correlation is with so-called LDA+U, or GGA+U calculations. It is intended to solve the band gap problem of traditional DFT, the problem being an underestimation of band gaps for transition-metal complexes. A Hubbard like term is added to the normal Hamiltonian, to model on-site interactions. In its very simplest form it depends on only one parameter, U, and this is the way it has been implemented in BAND. The energy expression is equation (11) in the work of Cococcioni.[*47* (page 227)] See also the review article [*46* (page 227)].

```
HubbardU
   Enabled [True | False]
   LValue string
   UValue string
   PrintOccupations [True | False]
End
```

HubbardU

Type Block

Description Options for Hubbard-corrected DFT calculations.

Enabled

Type Bool

Default value False

Description Whether or not to apply the Hubbard Hamiltonian

LValue

Type String

Default value

Description For each atom type specify the l value (0 - s orbitals, 1 - p orbitals, 2 - d orbitals). A negative value is interpreted as no l-value.

UValue

Type String

Default value

Description For each atom type specify the U value (in atomic units). A value of 0.0 is interpreted as no U.

PrintOccupations

Type Bool

Default value True

Description Whether or not to print the occupations during the SCF.

An example to apply LDA+U to the d-orbitals of NiO looks like:

```
...
Atoms
   Ni 0.000  0.000  0.000
   O 2.085  2.085  2.085
End
...

...
HubbardU
   printOccupations true
   Enabled          true
   uvalue           0.3 0.0
   lvalue           2   -1
End
...
```

3.1.3 OEP

(Expert options) When you are using a meta-GGA you are by default using a generalized Kohn-Sham method. However, it is possible to calculate a local potential, as is required for a strict Kohn-Sham calculation, via OEP, (see [*67* (page 229)]).

The main options are controlled with the `MetaGGA` subkey of the XC block if `OEP` is present.

```
XC
   [...]
   MetaGGA GGA OEP {approximation} {Fit} {Potential}
   [...]
End
```

GGA specifies the name of the used meta-GGA. In combination with OEP only **PBE**, **TPSS**, **MVS**, **MS0**, **MS1**, **MS2**, and **SCAN** can be used!

approximation (**Default: KLI**) There are three flavors to approximate the OEP: **KLI**, **Slater**, and **ELP**

Fit By adding the string **Fit** on this line, one uses the fitted density instead of the exact density for the evaluation.

Potential If not specified, only the tau-dependent part of the OEP is evaluated and used. By adding the string **Potential** in addition the tau-independent part is added to the XC potential. (This is needed e.g. for plotting the 'vxc')

With the following subkeys of the `XC` blockkey you have extra control over the iterative OEP evaluation:

MGGAOEPMaxIter (**Default: 30**) defines the maximum number of cycles for the iterative OEP evaluation.

MGGAOEPConvergence (**Default: 1E-6**) defines convergence criterion for OEP evaluation.

MGGAOEPWaitIter (**Default: 0**) defines the number of SCF cycles with the regular meta-GGA before switching to the OEP scheme.

MGGAOEPMaxAbortIter (**Default: 0**) defines number of cycles for which the error is allowed to increase before the calculation is aborted. Here, zero means: do never abort.

MGGAOEPMaxErrorIncrease (**Default: 0.0**) defines the maximum rate of increasing error before the calculation is aborted. Here, zero means: do never abort.

An example for an OEP metaGGA calculation

```
XC
   MetaGGA MVS OEP
End
```

Note that a very fine Becke grid is needed.

```
BeckeGrid
   Quality USER
   UserRadMulFactor 20.0
   UserCoreL 11
   UserInter1L 13
   UserInter2L 21
   UserExterL 31
   UserExterLBoost 35
 End
```

Note also: the gaps are typically not closer to experiment, and the calculations are more expensive. This option is mainly about academic interest.

3.2 Relativistic Effects and Spin

3.2.1 Spin polarization

By default Band calculations are spin-restricted. You can instruct Band to perform a spin-unrestricted via the `Unrestricted` key:

```
Unrestricted [True | False]
```

Unrestricted

Type Bool

Default value False

Description Controls wheather Band should perform a spin-unrestricted calculation. Spin-unrestricted calculations are computationally roughly twice as expensive as spin-restricted.

The orbitals are occupied according to the aufbau principle.

If you want to enforce a specific spin-polarization (instead of occupying according to the aufbau principle) you can use the `EnforcedSpinPolarization` key:

```
EnforcedSpinPolarization float
```

EnforcedSpinPolarization

Type Float

Description Enforce a specific spin-polarization instead of occupying according to the aufbau principle. The spin-polarization is the difference between the number of alpha and beta electron.Thus, a value of 1 means that there is one more alpha electron than beta electrons.The number may be anything, including zero, which may be of interest when searching for a spin-flipped pair, that may otherwise end up in the (more stable) parallel solution.

3.2.2 Relativistic Effects

Relativistic effects are treated with the accurate and efficient ZORA approach [*17* (page 226), *18* (page 226)], controlled by the `Relativistic` keyword. Relativistic effects are negligible for light atoms, but grow to dramatic changes for heavy elements. A rule of thumb is: Relativistic effects are quite small for elements of row 4, but very large for row 6 elements (and later).

```
Relativity
   Level [None | Scalar | Spin-Orbit]
End
```

`Relativity`

Type Block

Description Options for relativistic effects.

`Level`

Type Multiple Choice

Default value None

Options [None, Scalar, Spin-Orbit]

Description None: No relativistic effects.Scalar: Scalar relativistic ZORA.This option comes at very little cost.SpinOrbit: Spin-orbit coupled ZORA.This is the best level of theory, but it is (4-8 times) more expensive than a normal calculation. Spin-orbit effects are generally quite small, unless there are very heavy atoms in your system, especially with p valence electrons (like Pb).See also the SpinOrbitMagnetization key.

See also the *SpinOrbitMagnetization* (page 17) key.

3.3 Solvation

3.3.1 COSMO: Conductor like Screening Model and the Solvation-key

You can study chemistry in solution, as contrasted to the gas phase, with the implementation in BAND of the Conductor like Screening Model (COSMO) of solvation.[*36* (page 227)]

In the COSMO model all solvents are roughly the same, and approximated by an enveloping metal sheet. One explicit dependency on the solvent is that the solvation energy is scaled by

$$f(\epsilon) = \frac{\epsilon - 1}{\epsilon + \chi}$$

and this depends on the dielectric constant of the solvent, and an empirical factor χ. The other is that the shape of the surface is influenced by the *Rad* parameter, see below.

The solvent information is specified in the `solvent` key of the `solvation` block. The simplest option is to use one of the pre-defined solvents:

```
Solvation
   Enabled [True | False]
   Solvent
      Name [...]
   End
End
```

`Solvation`

Type Block

Description Options for the COSMO (Conductor like Screening Model) solvation model.

`Enabled`

Type Bool

Default value False

Description Use the Conductor like Screening Model (COSMO) to include solvent effects.

`Solvent`

Type Block

Description Solvent details

`Name`

Type Multiple Choice

Default value Water

Options [AceticAcid, Acetone, Acetonitrile, Ammonia, Aniline, Benzene, BenzylAlcohol, Bromoform, Butanol, isoButanol, tertButanol, CarbonDisulfide, CarbonTetrachloride, Chloroform, Cyclohexane, Cyclohexanone, Dichlorobenzene, DiethylEther, Dioxane, DMFA, DMSO, Ethanol, EthylAcetate, Dichloroethane, EthyleneGlycol, Formamide, FormicAcid, Glycerol, HexamethylPhosphoramide, Hexane, Hydrazine, Methanol, MethylEthylKetone, Dichloromethane, Methylformamide, Methypyrrolidinone, Nitrobenzene, Nitrogen, Nitromethane, PhosphorylChloride, IsoPropanol, Pyridine, Sulfolane, Tetrahydrofuran, Toluene, Triethylamine, TrifluoroaceticAcid, Water]

Description Name of a pre-defined solvent. A solvent is characterized by the dielectric constant (Eps) and the solvent radius (Rad).

This is the list of possible solvents and their corresponding `Eps` and `Rad` values:

Solvent 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
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

Continued on next page

Table 3.1 – continued from previous page

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

Several other options can be defined in the `Solvation` block:

```
Solvation
   CVec [EXACT | FITPOT]
   Charge
      Conv float
      Corr [True | False]
      Iter integer
      Method [CONJ | INVER]
   End
   Enabled [True | False]
   Radii # Non-standard block. See details.
      ...
```

```
   End
   SCF [VAR | PERT | NONE]
   Solvent
      Del float
      Emp float
      Eps float
      Name [...]
      Rad float
   End
   Surf [Delley | Wsurf | Asurf | Esurf | Klamt]
End
```

Solvation

Type Block

Description Options for the COSMO (Conductor like Screening Model) solvation model.

CVec

Type Multiple Choice

Default value EXACT

Options [EXACT, FITPOT]

Description Choose how to calculate the Coulomb interaction matrix between the molecule and the point charges on the surface:- EXACT: use exact density, and integrate against the potential of the point charges. This may have inaccuracies when integration points are close to the point charges.- FITPOT: evaluate the molecular potential at the positions of the point charges, and multiply with these charges.

Charge

Type Block

Description Select the algorithm to determine the charges.

Conv

Type Float

Default value 1e-08

Description Charge convergence threshold in iterative COSMO solution.

Corr

Type Bool

Default value True

Description Correct for outlying charge.

Iter

Type Integer

Default value 1000

Description Maximum number of iterations to solve COSMO equations.

Method

Type Multiple Choice

Default value CONJ

Options [CONJ, INVER]

Description INVER: matrix inversion, CONJ: biconjugate gradient method.The CONJ method is guaranteed to converge with small memory requirements and is normally the preferred method.

Enabled

Type Bool

Default value False

Description Use the Conductor like Screening Model (COSMO) to include solvent effects.

Radii

Type Non-standard block

Description The values are the radii of the atomic spheres. If not specified the default values are those by Allinge. Format: 'AtomType value'. e.g.: 'H 0.7'

SCF

Type Multiple Choice

Default value VAR

Options [VAR, PERT, NONE]

Description Determine the point charges either Variational (VAR) or after the SCF as a Perturbation (PERT).

Solvent

Type Block

Description Solvent details

Del

Type Float

Description Del is the value of Klamt's delta_sol parameter, only relevant in case of Klamt surface.

Emp

Type Float

Description Emp is the empirical scaling factor x for the energy scaling.

Eps

Type Float

Description User-defined dielectric constant of the solvent (overrides the Eps value of the solvent defined in 'Name')

Name

Type Multiple Choice

Default value Water

Options [AceticAcid, Acetone, Acetonitrile, Ammonia, Aniline, Benzene, BenzylAlcohol, Bromoform, Butanol, isoButanol, tertButanol, CarbonDisulfide, CarbonTetrachloride, Chloroform, Cyclohexane, Cyclohexanone, Dichlorobenzene, DiethylEther, Dioxane, DMFA, DMSO, Ethanol, EthylAcetate, Dichloroethane, EthyleneGlycol, Formamide,

FormicAcid, Glycerol, HexamethylPhosphoramide, Hexane, Hydrazine, Methanol, MethylEthylKetone, Dichloromethane, Methylformamide, Methypyrrolidinone, Nitrobenzene, Nitrogen, Nitromethane, PhosphorylChloride, IsoPropanol, Pyridine, Sulfolane, Tetrahydrofuran, Toluene, Triethylamine, TrifluoroaceticAcid, Water]

Description Name of a pre-defined solvent. A solvent is characterized by the dielectric constant (Eps) and the solvent radius (Rad).

Rad

Type Float

Unit Angstrom

Description User-defined radius of the solvent molecule (overrides the Rad value of the solvent defined in 'Name').

Surf

Type Multiple Choice

Default value Delley

Options [Delley, Wsurf, Asurf, Esurf, Klamt]

Description Within the COSMO model the molecule is contained in a molecule shaped cavity.Select one of the following surfaces to define the cavity:- Wsurf: Van der Waals surface- Asurf: solvent accessible surface- Esurf: solvent excluding surface- Klamt: Klamt surface- Delley: Delley surface.

3.3.2 Additional keys for periodic systems

For the simulation of periodic structures ICW solvation, you may specify the following options:

```
PeriodicSolvation
   RemovePointsWithNegativeZ [True | False]
   NStar integer
End
```

PeriodicSolvation

Type Block

Description Additional options for simulations of periodic structures with solvation.

RemovePointsWithNegativeZ

Type Bool

Default value False

Description Whether the COSMO surface is constructed on both sides of a surface.If one is only interested in the solvation effect on the upper side of a surface (in the Z direction), then this option should be set to 'True'

NStar

Type Integer

Default value 4

Description This option, expecting an integer number (>2), handles the accuracy for the construction of the COMSO surface. The larger the given number the more accurate the construction.

General remarks: The accuracy of the result and the calculation time is influenced by the screening radius `SCREENING%RMADEL` (see *Screening* (page 56) block). If the calculation does take too long, defining a smaller radius does help. **But:** too small radii, especially smaller than the lattice constants, will give unphysical results.

3.4 Electric and Magnetic Fields

3.4.1 Electric Field

```
EField
   Ez float
   unit [Volt/Angstrom | a.u. | Volt/Bohr | Volt/meter]
End
```

EField

Type Block

Description Include a homogeneous, static, electric field in the z-direction (only possible for 0D, 1D or 2D periodic systems)

Ez

Type Float

Default value 0.0

Description Strength of the electric field, in units as selected with the EField unit key.

unit

Type Multiple Choice

Default value Volt/Angstrom

Options [Volt/Angstrom, a.u., Volt/Bohr, Volt/meter]

Description Unit of the electric field Ez

The static field is explicitly handled in the determination of the orbital coefficients, energy, and gradients. If you apply it to any other property, such as the NMR shielding tensor, dielectric function, solvation energy, etc., the result is probably not entirely correct (that is, it might not include the effect of the external filed). In case of doubt contact the SCM staff.

The effect of a magnetic filed can be **approximated** by the following potential: $\mu_B \vec{\sigma}_i \vec{B}$, where μ_B is the Bohr magneton, $\vec{\sigma}_i$ are the Pauli matrices and $\vec{B}$ is the magnetic field. For *Spin-unrestricted collinear* (page 21) calculations, the spin is assumed to be aligned with the z-axis.

3.4.2 Magnetic Field

```
BField
   Bx float
   By float
   Bz float
   Dipole [True | False]
   DipoleAtom integer
   Method [NR_SDOTB | NR_LDOTB | NR_SDOTB_LDOTB]
   Unit [tesla | a.u.]
End
```

BField

Type Block

Description The effect of a magnetic filed can be approximated by the following potential: mu * sigma_i * B, where mu is the Bohr magneton, sigma_i are the Pauli matrices and B is the magnetic field

Bx

Type Float

Default value 0.0

Unit Tesla

Description Value of the x component of the BField

By

Type Float

Default value 0.0

Unit Tesla

Description Value of the y component of the BField

Bz

Type Float

Default value 0.0

Unit Tesla

Description Value of the z component of the BField

Dipole

Type Bool

Default value False

Description Use an atomic dipole as magnetic field instead of a uniform magnetic field.

DipoleAtom

Type Integer

Default value 1

Description Atom on which the magnetic dipole should be centered (if using the dipole option)

Method

Type Multiple Choice

Default value NR_SDOTB

Options [NR_SDOTB, NR_LDOTB, NR_SDOTB_LDOTB]

Description There are two terms coupling to an external magnetic field.One is the intrinsic spin of the electron, called S-dot-B, the other one is the orbital momentum call L-dot-B.The L.B is implemented non-relativistically, using GIAOs in the case of a homogeneous magnetic field (not for the dipole case).

Unit

Type Multiple Choice

Default value tesla

Options [tesla, a.u.]

Description Unit of magnetic filed. The a.u. is the SI version of a.u.

3.4.3 Atoim-wise fuzzy potential

```
FuzzyPotential # Non-standard block. See details.
   ...
End
```

FuzzyPotential

Type Non-standard block

Description Atomic (fuzzy cell) based, external, electric potential. See example.

Example:

```
FuzzyPotential
   scale $scale
   a1 v1   ! atom with index a1 gets potential coefficient v1 (a.u.)
   a2 v2   ! atom a2 gets potential v2
   ...
End
```

scale Overall scaling factor to be applied.

If an atom is not in the list it gets a coefficient of zero. The potential of an atom is its number (v_i) as specified on input times its fuzzy cell

$$V(r) = \sum_{i}^{\text{atoms}} v_i \mathcal{P}_{i,U}(r)$$

using the same partition function $\mathcal{P}$ as for the *BeckeGrid* (page 43). A partition function (or fuzzy cell) of an atom is close to one in the neighborhood of this atom.

The sign convention is: negative is favorable for electrons. (Unit: a.u.)

3.5 Nuclear Model

```
NuclearModel [PointCharge | Gaussian | Uniform]
```

NuclearModel

Type Multiple Choice

Default value PointCharge

Options [PointCharge, Gaussian, Uniform]

Description Specify what model to use for the nucleus.For the Gaussian model the nuclear radius is calculated according to the work of Visscher and Dyall (L. Visscher, and K.G. Dyall, Dirac-Fock atomic electronic structure calculations using different nuclear charge distributions, Atomic Data and Nuclear Data Tables 67, 207 (1997))

CHAPTER

FOUR

ACCURACY AND EFFICIENCY

Given a Model Hamiltonian, the most important aspects determining the accuracy of a Band calculation are:

- *Basis set* (page 31)
- *K-Space* (page 39)

Also important, but to a lesser degree, are the following aspects:

- *Numerical Integration (BeckeGrid)* (page 43)
- *Density fitting (ZlmFit)* (page 45)
- *Basis-set confinement (SoftConfinement)* (page 36)
- *SCF convergence (Convergence)* (page 51)
- *Hartree–Fock Resolution of the Identity (RIHartreeFock)* (page 48) (only for hybrid functionals)

The CPU time and memory requirements strongly depends on these options, as does the accuracy of the results.

General NumericalQuality

A simple way of tweaking the accuracy of the calculation is via the **NumericalQuality** key. This sets the quality of several technical aspects of a Band calculation (with the notable exception of the *basis set* (page 31))

```
NumericalQuality [Basic | Normal | Good | VeryGood | Excellent]
```

`NumericalQuality`

Type Multiple Choice

Default value Normal

Options [Basic, Normal, Good, VeryGood, Excellent]

Description Set the quality of several important technical aspects of a BAND calculation (with the notable exception of the basis set). It sets the quality of: BeckeGrid (numerical integration), ZlmFit (density fitting), KSpace (reciprocal space integration), and SoftConfinement (basis set confinement). Note: the quality defined in the block of a specific technical aspects supersedes the value defined in NumericalQuality (e.g. if I specify 'NumericalQuality Basic' and 'BeckeGrid%Quality Good', the quality of the BeckeGrid will be 'Good')

4.1 Basis set

Band represents the single-determinant electronic wave functionus as a linear combinations of atom-centered basis functions (the basis set). See also: Wikipedia page on Basis Sets (https://en.wikipedia.org/wiki/Basis_set_(chemistry)).

The basis sets in Band consists of **NAOs** (Numerical Atomic Orbitals, obtained by solving numerically the Kohn-Sham equations for the isolated spherical atoms) augmented by a set of **STOs** (Slater Type Orbitals).

The choice of basis set is very important, as it influences heavily the accuracy, the CPU time and the memory usage of the calculation. Band comes with 6 predefined types of basis sets: **SZ**, **DZ**, **DZP**, **TZP**, **TZ2P**, **QZ4P** (SZ: Single Zeta, DZ: Double Zeta, DZP: Double Zeta + Polarization, TZP: Triple Zeta + Polarization, TZ2P: Triple Zeta + Double Polarization, QZ4P: Quadruple Zeta + Quadruple Polarization). See the sections *Which basis set should I use?* (page 32) and *Available Basis Sets* (page 35) for more details.

To speed up the calculation, Band can use the **frozen core approximation** in which core orbitals are kept frozen during the SCF procedure (and the valence orbitals are orthogonalized against the frozen orbitals). One can run an **all electron** calculation by specifying `Core None` in the Basis input block. Note: some features, such as *Hybrid functionals* (page 18), are incompatible with the frozen-core approximation, and require an all electron (i.e. `Core None`) basis set.

4.1.1 Basis input block

You can specify which basis set to use in the `Basis` input block.

```
Basis
   Type [SZ | DZ | DZP | TZP | TZ2P | QZ4P]
   Core [None | Small | Medium | Large]
End
```

`Basis`

Type Block

Description Definition of the basis set

`Type`

Type Multiple Choice

Default value DZ

Options [SZ, DZ, DZP, TZP, TZ2P, QZ4P]

Description The basis sets to be used.

`Core`

Type Multiple Choice

Default value Large

Options [None, Small, Medium, Large]

Description Size of the frozen core.

4.1.2 Which basis set should I use?

The hierarchy of basis sets, from the smallest and least accurate (SZ) to the largest and most accurate (QZ4P), is **SZ < DZ < DZP < TZP < TZ2P < QZ4P**.

The choice of basis set is in general a trade off between accuracy and computation time: the more accurate the basis set, the more computationally demanding the calculation will be (both in term of CPU time and the memory usage).

As an example, in the following table we compare accuracy v.s. CPU time for a (24,24) carbon nanotube using different basis sets. "Energy error" is defined as the absolute error in the formation energy per atom using the QZ4P results as reference.[1]

Table 4.1: Accuracy and CPU time ratio for a (24,24) carbon nanotube using different basis sets

Basis Set	Energy Error [eV]	CPU time ratio (relative to SZ)
SZ	1.8	1
DZ	0.46	1.5
DZP	0.16	2.5
TZP	0.048	3.8
TZ2P	0.016	6.1
QZ4P	reference	14.3

It is worthwhile noting that the error in formation energies are to some extend systematic, and they partially cancel each other out when taking energy differences. For example, if one considers the difference in energy between two carbon nanotubes variants ((24,24) and (24-0)) with the same number of atoms, the basis set error is smaller than 1 milli-eV/atom already with a DZP basis set, which is much smaller than the absolute error in the individual energies. The same consideration holds for reactions barriers: the error in the energy difference between different conformations is much smaller than the error in the absolute energies themselves.

Band gaps:

The following figure shows the convergence WRT basis set of band gaps (XC:PBE). While DZ is often inaccurate (since DZ lacks any polarization function, the description of the virtual orbital space is very poor), TZP captures the trends very well.

In general, since the basis set might have different effects on different properties, it is advisable to run a few simple calculations to get an idea of the effect of the basis set with your property of interest.

A summary of the basis sets:

- **SZ**: Single Zeta, the minimal basis set (only NAOs), serves mostly a technical purpose. The results are rather inaccurate, but it's computationally efficient. It can be useful for running a very quick test calculation. See also Single-Zeta-SCF-Restart link xxx.
- **DZ**: The Double Zeta basis set is computationally very efficient. It can be used for pre-optimization of structures (that should then be further optimized with a better basis set). Since it has no polarization functions, properties depending on the virtual orbital space will be rather inaccurate.
- **DZP**: Double zeta plus polarization function. Only available for main group elements up to Krypton. For other elements a TZP basis set will be used **automatically**. This is a reasonably good basis set for geometry optimizations of organic systems.
- **TZP**: The Triple Zeta plus Polarization basis set offers the best balance between performance and accuracy. This is the basis set we would generally recommend.
- **TZ2P**: The Triple Zeta plus Double Polarization basis set is an accurate basis set. It is qualitatively similar to TZP but quantitatively better. It should be used when a good description of the virtual orbital space is needed.
- **QZ4P**: Quadruple zeta plus Quadruple Polarization. This is the biggest basis set available. It can be used for benchmarking.

Frozen core:

[1] Computational details: Single Point calculation, 'Good' NumericalQuality, no frozen core, 7 k-points, XC functional: GGA:PBE. Calculation performed on a 24 cores compute node. 96 atoms in the unit cell.

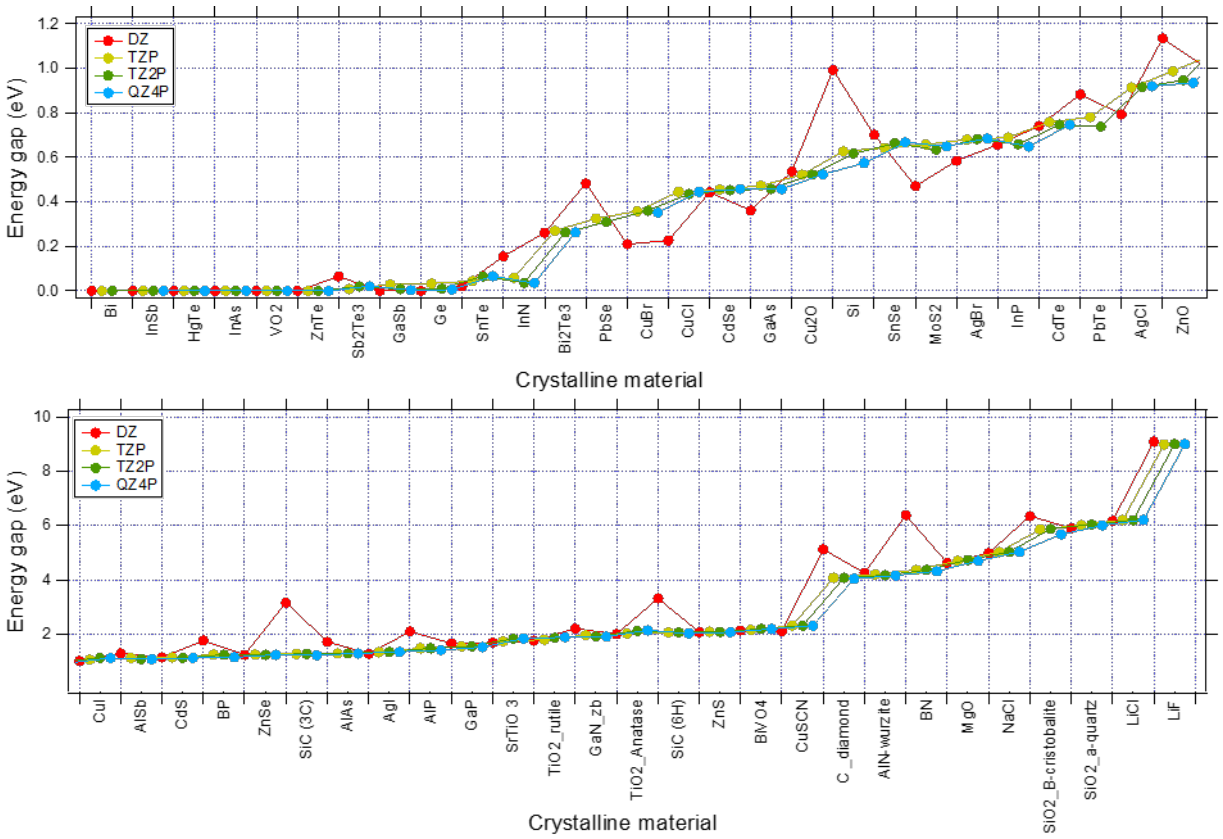


Fig. 4.1: Convergence of Band Gaps WRT basis set for various system. (XC:PBE, K-Space Quality: Good)

In general, the frozen core approximation does not influence the results significantly (especially if one uses a `small` frozen core). For accurate results on certain properties (like *Properties at Nuclei* (page 71)) all electron basis sets are needed on the atoms of interest.

- For Meta-GGA XC functionals, it is recommended to use `small` or `none` frozen core (the frozen orbitals are computed using LDA and not the selected Meta-GGA)
- For optimizations under pressure, use `small` or `none` frozen core

4.1.3 Available Basis Sets

The basis set files, containing the definition of the basis set, are located in `$ADFHOME/atomicdata/band`.

The next table gives an indication frozen core (*fc*) standard basis sets are available for the different elements in BAND. Note that all electron (*ae*) basis set are available for all basis sets types.

Table 4.2: Available standard basis sets for non-relativistic and ZORA calculations H-Uuo (Z=1-118)

Element	frozen core	SZ, DZ	DZP	TZP, TZ2P, QZ4P
H-He (Z=1-2)	ae	Yes	Yes	Yes
Li-Ne (Z=3-10)	ae .1s	Yes	Yes	Yes
Na-Mg (Z=11-12)	ae .1s .2p	Yes	Yes	Yes
Al-Ar (Z=13-18)	ae .2p	Yes	Yes	Yes
K-Ca (Z=19-20)	ae .2p .3p	Yes	Yes	Yes
Sc-Zn (Z=21-30)	ae .2p .3p	Yes		Yes
Ga-Kr (Z=31-36)	ae .3p .3d	Yes	Yes	Yes
Rb-Sr (Z=37-38)	ae .3p .3d .4p	Yes		Yes
Y-Cd (Z=39-48)	ae .3d .4p	Yes		Yes
In-Ba (Z=49-56)	ae .4p .4d	Yes		Yes
La-Lu (Z=57-71)	ae .4d .5p	Yes		Yes
Hf-Hg (Z=72-80)	ae .4d .4f	Yes		Yes
Tl (Z=81)	ae .4d .4f .5p	Yes		Yes
Pb-Rn (Z=82-86)	ae .4d .4f .5p .5d	Yes		Yes
Fr-Ra (Z=87-88)	ae .5p .5d	Yes		Yes
Ac-Lr (Z=89-103)	ae .5d .6p	Yes		Yes
Rf-Uuo(Z=104-118)	ae .5d .5f	Yes		Yes

- element name (without suffix): all electron (ae)
- .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

Note: Not all combinations of basis set `Type` and `Core` are available for all elements. If a specific combination is not available, Band will pick the first *better* basis set.

4.1.4 More Basis input options

```
Basis
   Folder string
   ByAtomType # Non-standard block. See details.
      ...
   End
End
```

`Basis`

Type Block

Description Definition of the basis set

`Folder`

Type String

Description Path to a folder containing the basis set files. This can be used for special use-defined basis sets. Cannot be used in combination with 'Type'

`ByAtomType`

Type Non-standard block

Description Definition of the basis set for specific atom types (one definition per line). Format: 'AtomType Type=Type Core=Core'. Example: 'C.large_basis Type=TZ2P Core=None'

See also: *Example: Multiresolution* (page 141).

4.1.5 Confinement of basis functions

It is possible to alter the radial part of the basis functions in order to make them more compact, which will in turn speeds up the calculation.

```
SoftConfinement
   Quality [Auto | Basic | Normal | Good | VeryGood | Excellent]
   Radius float
   Delta float
End
```

`SoftConfinement`

Type Block

Description In order to make the basis functions more compact, the radial part of the basis functions is multiplied by a Fermi-Dirac (FD) function (this 'confinement' is done for efficiency and numerical stability reasons). A FD function goes from one to zero, controlled by two parameters. It has a value 0.5 at Radius, and the decay width is Delta.

Quality

Type Multiple Choice

Default value Auto

Options [Auto, Basic, Normal, Good, VeryGood, Excellent]

Description In order to make the basis functions more compact, the radial part of the basis functions is multiplied by a Fermi-Dirac (FD) function (this 'confinement' is done for efficiency and numerical stability reasons). A FD function goes from one to zero, controlled by two parameters. It has a value 0.5 at Radius, and the decay width is Delta. This key sets the two parameters 'Radius' and 'Delta'.Basic: Radius=7.0, Delta=0.7;Normal: Radius=10.0, Delta=1.0;Good: Radius=20.0, Delta=2.0;VeryGood and Excellent: no confinement at all.If 'Auto', the quality defined in the 'NumericalQuality' will be used.

Radius

Type Float

Unit Bohr

Description Explicitely specify the radius parameter of the Fermi-Dirac function.

Delta

Type Float

Unit Bohr

Description Explicitely specify the delta parameter of the Fermi-Dirac function (if not specified, it will be 0.1*Radius).

- For geometry optimizations under pressure, `Basic` soft confinement is recommended.

4.1.6 Manually specifying AtomTypes (expert option)

AtomType (block-type) (*Expert Option*) Description of the atom type. Contains the block keys `Dirac`, `BasisFunctions` and `FitFunctions`. The key corresponds to one atom type. The ordering of the `AtomType` keys (in case of more than one atom type) is **NOT** arbitrary. It is interpreted as corresponding to the ordering of the `Atoms` keys. The n-th `AtomType` key supplies information for the numerical atom of the n^{th} type, which in turn has atoms at positions defined by the n^{th} `Atoms` key.

```
AtomType ElementSymbol
   Dirac ChemSym
      {option}
      ...
      shells cores
      shell_specification {occupation_number}
      ...
   SubEnd
   {BasisFunctions
      shell_specification STO_exponent
      ...
   SubEnd}
   FitFunctions
      shell_specification STO_exponent
      ...
   SubEnd
END
```

The argument *ElementSymbol* to `AtomType` is the symbol of the element that is referred to in the `Atoms` key block.

`Dirac (block-type)` Specification of the numerical ('Herman-Skillman') free atom, which defines the initial guess for the SCF density, and which also (optionally) supplies Numerical Atomic Orbitals (NOs) as basis functions, and/or as STO fit functions for the crystal calculation. The argument *ChemSym* of this option is the symbol of the element of the atom type. The data records of the `Dirac` key are:

1. the number of atomic shells (1s,2s,2p,etc.) and the nr. of core-shells (two integers on one line).
2. specification of the shell and its electronic occupation.

This specification can be done via quantum numbers or using the standard designation (e.g. '1 0' is equivalent to '1s'). Optionally one may insert anywhere in the Dirac block a record *Valence*, which signifies that all numerical valence orbitals will be used as basis functions (NOs) in the crystal calculation. You can also insert *NumericalFit* followed by a number (max. l-value) in the key block, which causes the program to use numerical STO fit functions. For example NumericalFit 2 means that the squares of all s,p, and d NOs will be used as STO fit functions with $l = 0$, since the NOs are spherically symmetric. If you insert *Spinor*, a spin-orbit relativistic calculation for the single-atom will be carried out.

The Herman-Skillman program generates all its functions (atomic potential, charge density, one-electron states) as tables of values in a logarithmic radial grid. The number of points in the grid, and the min. and max. r-value are defaulted at 3000, 0.000001, and 100.0 (a.u.) respectively. These defaults can be overwritten by specifying anywhere in the Dirac block the (sub)keys *radial*, *rmin* and *rmax*.

The program will do a spin-unrestricted calculation for the atoms in addition to the restricted one. The occupation of the spin-orbitals will be of maximum spin-multiplicity and cannot be controlled in the Dirac key-block.

`BasisFunctions (block-type)` Slater-type orbitals, specified by quantum numbers n,:math:l or by the letter designation (e.g. 2p) and one real (alpha) per STO. One STO per record. Use of this key is optional in the sense that Slater-type functions are not needed if other basis functions have been specified (i.e. the numerical atomic orbitals, see key Dirac).

`FitFunctions (block-type)` Slater-type fit functions, described in the same way as in `BasisFunctions`. Each `FitFunctions` key corresponds to one atom type, the type being the one of the preceding `Dirac` key. The selection choice of a 'good' fit set is a matter of experience. Fair quality sets are included in the database of the molecular program ADF.

Example:

```
AtomType C :: Carbon atom
   Dirac C
      3 1
   VALENCE
      1s
      2s
      2p 2.0
   SubEnd
   BasisFunctions
      1s 1.7
      ...
   SubEnd
   FitFunctions
      1s 13.5
      2s 11.0
      ...
   SubEndEnd
```

TestFunctions (block-type) An optional subkey of the `AtomType` key block is `TestFunctions` which has the same format as the `BasisFunctions` and `FitFunctions` blocks. The `TestFunctions` block specifies STOs to be used as test functions in the numerical integration package. For the time being the l value is ignored. A possible application is to include a very tight function, to increase the accuracy near a nucleus.

4.1.7 Basis Set Superposition Error (BSSE)

The Ghost Atom feature enables the calculation of Basis Set Superposition Errors (BSSE). Normally, if you want to know the bonding energy of system A with system B you calculate three energies

1. $E(A+B)$
2. $E(A)$
3. $E(B)$

The bond energy is then $E(A+B) - E(A) - E(B)$

The BSSE correction is about the idea that we can also calculate E(A) including basis functions from molecule B.

You can make a ghost atom by simply adding "Gh." in front of the element name, for instance "Gh.H" for a ghost hydrogen , "Gh.C" for a ghost Carbon atom.

You will get a better bonding energy, closer to the basis set limit by calculating

$E(A+B) - E(\text{A with B as ghost}) - E(\text{B with A as ghost})$

The BSSE correction is

$E(A) - E(\text{A with B as ghost}) + E(B) - E(\text{B with A as ghost})$

See also:

Example: BSSE correction (page 143)

4.2 K-Space

The K-Space sampling (i.e., the k-points used to sample the Brillouin Zone) is an important technical aspect of Band, as it influences heavily the accuracy, the CPU time and the memory usage of the calculation (see section *Recommendations for k-space* (page 41)).

4.2.1 KSpace input block

The K-Space can be controlled via the `KSpace` input block. Two different k-space integration methods are available: the *Regular Grid* (**default**) and the *Symmetric Grid*.

```
KSpace
   Type [Regular | Symmetric]
   Quality [Auto | GammaOnly | Basic | Normal | Good | VeryGood | Excellent]
End
```

KSpace

Type Block

Description Options for the k-space integration (i.e. the grid used to sample the Brillouin zone)

Type

Type Multiple Choice

Default value Regular

Options [Regular, Symmetric]

Description The type of k-space integration grid used to sample the Brillouin zone (BZ) used.'Regular': simple regular grid.'Symmetric': symmetric grid for the irreducible wedge of the first BZ (useful when high-symmetry points in the BZ are needed to capture the correct physics of the system, graphene being a notable example).

`Quality`

Type Multiple Choice

Default value Auto

Options [Auto, GammaOnly, Basic, Normal, Good, VeryGood, Excellent]

Description Select the quality of the K-space grid used to sample the Brillouin Zone. If 'Auto', the quality defined in the 'NumericalQuality' will be used. If 'GammaOnly', only one point (the gamma point) will be used. The actual number of K points generated depends on this option and on the size of the unit cell. The larger the real space cell, the fewer K points will be generated.The CPU-time and accuracy strongly depend on this option.

Regular K-Space grid

By default, Band will look at the size of a lattice vectors and the KSpace quality to determine the number of k-points. The larger the lattice vector in real space, the smaller the reciprocal space vectors are, and as a result fewer k-points are needed. The following intervals will be distinguished: 0-5 Bohr, 5-10 Bohr, 10-20 Bohr, 20-50 Bohr, and beyond. Here is the table explaining how many k-points will be used along a lattice vector.

Lattice vector length	Basic	Normal	Good	VeryGood	Excellent
0-5 Bohr	5	9	13	17	21
5-10 Bohr	3	5	9	13	17
10-20 Bohr	1	3	5	9	13
20-50 Bohr	1	1	3	5	9
50- Bohr...	1	1	1	3	5

By preferring odd-numbered values we can use a quadratic interpolation method, and have the Γ point in the grid. It is then reasonable to assume a decaying error when going to a better quality setting.

It is also possible to manually specify the number of k-space points along each reciprocal lattice vector

```
KSpace
   Regular
      NumberOfPoints integer_list
   End
End
```

`KSpace`

Type Block

Description Options for the k-space integration (i.e. the grid used to sample the Brillouin zone)

`Regular`

Type Block

Description Options for the regular k-space integration grid.

NumberOfPoints

Type Integer List

Description Use a regular grid with the specified number of k-points along each reciprocal lattice vector.For 1D periodic systems you should specify only one number, for 2D systems two numbers, and for 3D systems three numbers.

Symmetric K-Space grid (tetrahedron method)

The tetrahedron method can be useful when high symmetry points in the BZ are needed to capture the correct physics of the system, graphene being a notable example.

The number of k-points in the symmetric grid depends on the KSpace quality and on the length of the shortest lattice vector.

It is also possible to manually specify the symmetric k-space integration parameter:

```
KSpace
   Symmetric
      KInteg integer
   End
End
```

KSpace

Type Block

Description Options for the k-space integration (i.e. the grid used to sample the Brillouin zone)

Symmetric

Type Block

Description Options for the symmetric k-space integration grid.

KInteg

Type Integer

Description Specify the accuracy for the Symmetric method.1: absolutely minimal (only the G-point is used)2: linear tetrahedron method, coarsest spacing3: quadratic tetrahedron method, coarsest spacing4,6,... (even): linear tetrahedron method5,7.... (odd): quadratic methodThe tetrahedron method is usually by far inferior.

General Remark: The tetrahedron method samples the irreducible wedge of the first BZ, whereas the regular grid samples the whole, first BZ. As a rule of thumb you need to choose roughly twice the value for the regular grid. For example kspace 2 compares to grid 4 4 4, kspace 3 to grid 5 5 5, etc.. Sticking to this rule the number of unique k-points will be roughly similar.

4.2.2 Recommendations for k-space

Which K-Space quality to use depends very much a) the system you are studying and b) the property you are interested in. We strongly recommend you to test the effect of different K-Space qualities on your system and properties of interest.

As an example, in the following table we list the errors on formation energy and band gap for diamond using regular k-space grids of different qualities (using Excellent kSpace quality as reference).

Table 4.3: Accuracy of formation energy for diamond (primitive unit cell) using different KSpace grids

KSpace quality	Energy error / atom [eV]	CPU time ratio
Gamma-Only	3.3	1
Basic	0.6	2
Normal	0.03	6
Good	0.002	16
VeryGood	0.0001	35
Excellent	reference	64

It is worthwhile noting that the errors due to finite k-space sampling in formation energies are to some extend systematic, and they partially cancel each other out when taking energy differences.

In general, metals (or narrow-gap semiconductor) require higher K-Space sampling than insulators. For insulators and wide-gap semiconductors, Normal K-Space quality often suffices. For Narrow-gap semiconductor and metals, Good K-Space quality is highly recommended. For geometry optimizations under pressure, Good K-Space quality is recommended.

Furthermore for certain properties, such as band gaps, Normal K-Space quality might not be enough to obtain reliable results. For example, in following figure we see how Normal K-Space quality is often not enough for computing band gaps (especially for the narrow-gap semiconductor of the top panel). For band gap prediction, it is recommended to use Good K-Space quality.

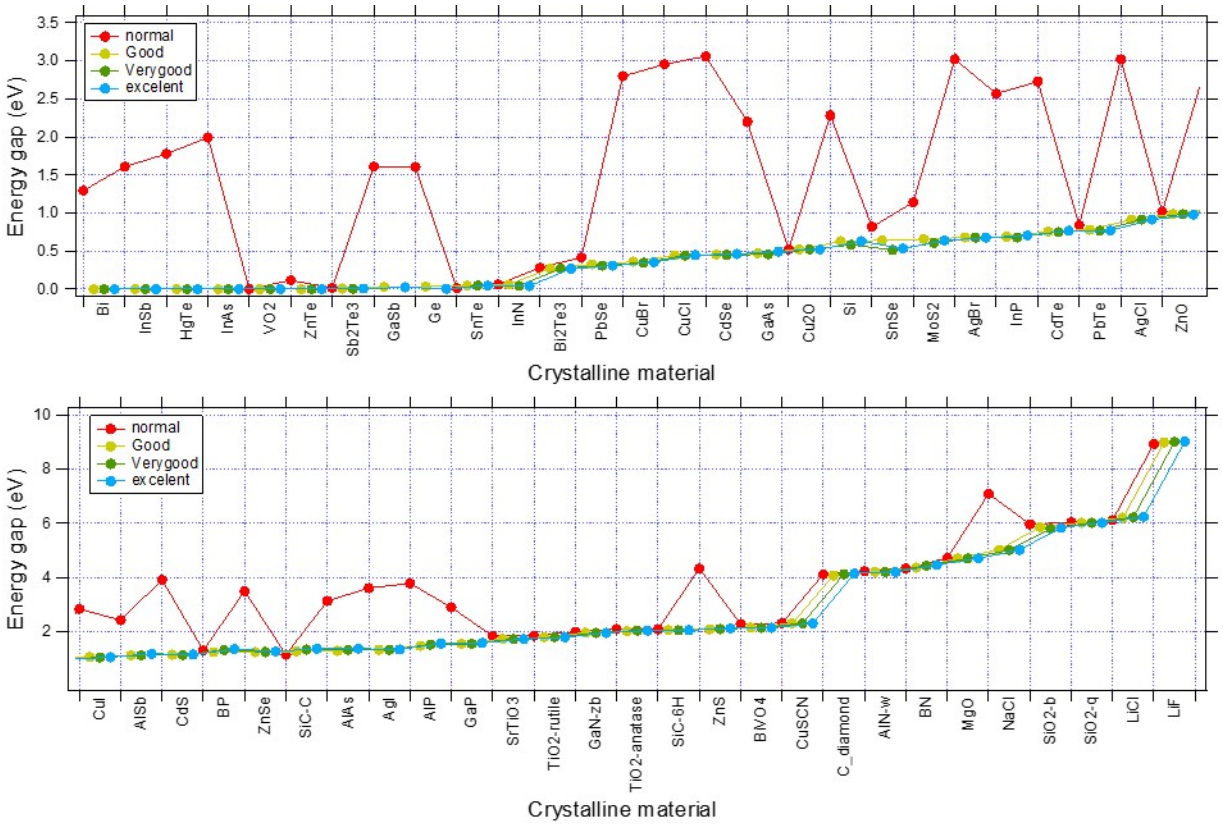


Fig. 4.2: Convergence of band gaps WRT the k-space quality set for various system. (XC:PBE, Basis:TZP)

4.3 Numerical Integration

Many of the integrals needed by Band are computed via numerical integration. See also: Wikipedia page on Numerical Integration (https://en.wikipedia.org/wiki/Numerical_integration).

4.3.1 Becke Grid

The numerical integration grid is a refined version of the fuzzy cells integration scheme developed by Becke.[*51* (page 228)] The implementation in BAND is described in Ref. [*52* (page 228)].

The quality of the Becke integration grid can be changed within the `BeckeGrid` block:

```
BeckeGrid
   Quality [Auto | Basic | Normal | Good | VeryGood | Excellent]
   RadialGridBoost float
   AtomDepQuality # Non-standard block. See details.
      ...
   End
End
```

`BeckeGrid`

Type Block

Description Options for the numerical integration grid, which is a refined version of the fuzzy cells integration scheme developed by Becke.

`Quality`

Type Multiple Choice

Default value Auto

Options [Auto, Basic, Normal, Good, VeryGood, Excellent]

Description Quality of the integration grid. For a description of the various qualities and the associated numerical accuracy see reference. If 'Auto', the quality defined in the 'NumericalQuality' will be used.

`RadialGridBoost`

Type Float

Default value 1.0

Description The number of radial grid points will be boosted by this factor. Some XC functionals require very accurate radial integration grids, so BAND will automatically boost the radial grid by a factor 3 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.

`AtomDepQuality`

Type Non-standard block

Description One can define a different grid quality for each atom (one definition per line). Line format: 'AtomIndex Quality', e.g. '3 Good' means that a grid of Good quality will be used

for the third atom in input order. If the index of an atom is not present in the AtomDepQuality section, the quality defined in the Quality key will be used

Example: Multiresolution (page 141) illustrates how to use the `AtomDepQuality` option.

Notes:

- The space-partition function used in BAND differs from the one described in Ref. [*52* (page 228)]. 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-Uuo):

$$\mathcal{P}_{i,U} = \begin{cases} 1 & \text{if } r_{i,U} < \Omega_I \\ 0 & \text{if } \exists j : r_{j,U} < \Omega_J \\ \eta_i \frac{e^{-2(r_{i,U}-\Omega_I)/a_0}}{(r_{i,U}-\Omega_I)^2} & \text{elsewhere} \end{cases}$$

- 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.

4.3.2 Radial grid of NAOs

```
RadialDefaults
   NR integer
   RMax float
   RMin float
End
```

`RadialDefaults`

Type Block

Description Options for the logarithmic radial grid of the basis functions used in the subprogram Dirac

`NR`

Type Integer

Default value 3000

Description Number of radial points. With very high values (like 30000) the Dirac subprogram may not converge.

`RMax`

Type Float

Default value 100.0

Unit Bohr

Description Upper bound of the logarithmic radial grid

`RMin`

Type Float

Default value 1e-06

Unit Bohr

Description Lower bound of the logarithmic radial grid

4.3.3 Voronoi grid (deprecated)

It is possible to use an alternative numerical integration scheme to the Becke Grid, namely the Voronoi Grid.

```
IntegrationMethod [Becke | Voronoi]
```

IntegrationMethod

Type Multiple Choice

Default value Becke

Options [Becke, Voronoi]

Description Choose the real-space numerical integration method. Note: the Voronoi integration scheme is deprecated.

The options for the Voronoi Grid are specified in the `Integration` block:

```
Integration
   AccInt float
End
```

Integration

Type Block

Description Options for the Voronoi numerical integration scheme. Deprecated. Use BeckeGrid instead.

AccInt

Type Float

Default value 3.5

Description General parameter controlling the accuracy of the Voronoi integration grid. A value of 3 would be basic quality and a value of 7 would be good quality.

4.4 Density Fitting

The Coulomb potential in Band is computed using a method called density fitting. The density fitting scheme in BAND is called **Zlm Fit**, and it is described in reference *53* (page 228). The ZlmFit is also used to compute (when needed) the gradient and hessian of the electron density.

4.4.1 Zlm Fit

The idea behind Zlm Fit can be summarized as follows: the total electron density is split into localized atomic densities (in a similar way as the volume is partitioned in the Becke grid). These atomic densities are then approximated by a combination of radial spline functions and real spherical harmonics (Zlm), for which the Coulomb potential can be easily computed.

```
ZlmFit
   Quality [Auto | Basic | Normal | Good | VeryGood | Excellent]
   AtomDepQuality # Non-standard block. See details.
      ...
   End
End
```

ZlmFit

Type Block

Description Options for the density fitting scheme 'ZlmFit'.

Quality

Type Multiple Choice

Default value Auto

Options [Auto, Basic, Normal, Good, VeryGood, Excellent]

Description Quality of the density-fitting approximation. For a description of the various qualities and the associated numerical accuracy see reference. If 'Auto', the quality defined in the 'NumericalQuality' will be used.

AtomDepQuality

Type Non-standard block

Description One can specify different ZlmFit-quality for different atoms, The syntax for this free block is 'iAtom quality', where iAtom is the index of the atom in input order. For the atoms that are not present in the AtomDepQuality sub-block, the quality defined in the Quality key will be used.

Example: Multiresolution (page 141) illustrates how to use the `AtomDepQuality` option.

Expert options

```
ZlmFit
   LMargin integer
   AllowBoost [True | False]
   DensityThreshold float
   PartitionFunThreshold float
   FGaussianW float
   FGridSpacing float
   FKSpaceCutOff float
   FirstTopoCell integer
   LastTopoCell integer
   OrderTopoTrick integer
   NumStarsPartitionFun integer
End
```

ZlmFit

Type Block

Description Options for the density fitting scheme 'ZlmFit'.

LMargin

Type Integer

Default value 4

Description User-defined l-margin, i.e., l_max for fitting is max(lMargin + l_max_basis_function, 2*l_max_basis_function)

AllowBoost

Type Bool

Default value True

Description Allow automatic atom-dependent tuning of maximum l of spherical harmonics expansion. Whether or not this boost is needed for a given atom is based on an heuristic estimate of how complex the density around that atom is.

DensityThreshold

Type Float

Default value 1e-07

Description Threshold below which the electron density is considered to be negligible.

PartitionFunThreshold

Type Float

Default value 0.0

Description Threshold for the partition functions: if an integration point has a partition function weight smaller than this threshold, it will be discarded.

FGaussianW

Type Float

Default value 1.0

Description Only for 3D periodic systems. Width of the Gaussian functions replacing the S and P Zlms for Fourier transform.

FGridSpacing

Type Float

Description Only for 3D periodic systems. Spacing for the Fourier grid. By default, this depends on the quality.

FKSpaceCutOff

Type Float

Description Only for 3D periodic systems. Cut-off of the grid in k-space for the Fourier transform.

FirstTopoCell

Type Integer

Default value 5

Description First cell for the topological extrapolation of the long range part of the Coulomb Potential.

LastTopoCell

Type Integer

Default value 10

Description Last cell for the topological extrapolation of the long range part of the Coulomb Potential.

OrderTopoTrick

Type Integer

Default value 3

Description Order of the topological extrapolation of the long range part of the Coulomb Potential.

NumStarsPartitionFun

Type Integer

Default value 5

Description Number of cell stars to consider when computing the partition function.

4.4.2 STO Fit (Deprecated)

In previous version of BAND (pre2014) this was the default option, which is now replaced by Zlm Fit. It is still used in the context of `NMR` and `OldResponse` calculations.

4.5 Hartree–Fock RI

The Hartree-Fock exchange matrix is calculated through a procedure known as Resolution of the Identity (RI). The implementation of the RI scheme in BAND is loosely based on work by Ren *et al.* [*57* (page 228)]. For more information on hybrid functionals in BAND, see the *XC section* (page 13).

Technical aspects of the RI scheme can be tweaked in the `RIHartreeFock` block:

```
RIHartreeFock
   Quality [VeryBasic | Basic | Normal | Good | VeryGood | Excellent]
   FitSetQuality [VeryBasic | Basic | Normal | Good | VeryGood | Excellent]
   DependencyThreshold float
   AtomDepQuality # Non-standard block. See details.
      ...
   End
End
```

RIHartreeFock

Type Block

Description The Hartree-Fock exchange matrix is calculated through a procedure known as Resolution of the Identity (RI). Here you can tweak various parameters of the procedure.

Quality

Type Multiple Choice

Default value Normal

Options [VeryBasic, Basic, Normal, Good, VeryGood, Excellent]

Description Accuracy of numerical integration and thresholds of the RI procedure.

FitSetQuality

Type Multiple Choice

Default value Normal

Options [VeryBasic, Basic, Normal, Good, VeryGood, Excellent]

Description The auxiliary fit set employed in the RI scheme. This is an important aspect of the procedure, significantly affecting both accuracy and computation time. For SZ and DZ

basis set a 'basic' FitSetQuality will suffice. For 'DZP' and 'TZP' a normal quality is recommended. For larger basis set, use either 'normal' or better FitSetQuality.

DependencyThreshold

Type Float

Default value 0.001

Description To improve numerical stability, almost linearly-dependent combination of basis functions are removed from the Hartree-Fock exchange matrix. If the SCF does not converge or 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

Type Non-standard block

Description One can define a different fit-set quality for each atom. The syntax for this free block is 'iAtom quality', where iAtom is the index of the atom in input order.

For efficiency and numerical stability reasons, it is advisable to include:

```
SoftConfinement
  Quality Basic
End
```

See the *Confinement of basis functions* (page 36) section for more info.

Notes: for periodic systems it is only possible to use short-range hybrid functionals (*e.g.* HSE06) and all-electron basis sets.

4.6 Self Consistent Field (SCF)

The SCF procedure searches for a self-consistent density. The self-consistent error is the square root of the integral of the squared difference between the input and output density of the cycle operator. When the SCF error is below a certain criterion, controlled by subkey `Criterion` of block key `Convergence`, convergence is reached. In case of bad convergence the SCF looks at the subkeys `Mixing`, and `Degenerate`, and the subkeys of block key `DIIS`.

See also:

Troubleshooting: *SCF does not converge* (page 117)

4.6.1 SCF block

```
SCF
   Eigenstates [True | False]
   Iterations integer
   Method [DIIS | MultiSecant]
   Mixing float
   PMatrix [True | False]
   Rate float
   VSplit float
End
```

SCF

Type Block

Description Controls technical SCF parameters.

Eigenstates

Type Bool

Description The program knows two alternative ways to evaluate the charge density iteratively in the SCF procedure: from the P-matrix, and directly from the squared occupied eigenstates. By default the program actually uses both at least one time and tries to take the most efficient. If present, Eigenstates turns off this comparison and lets the program stick to one method (from the eigenstates).

Iterations

Type Integer

Default value 100

Description The maximum number of SCF iterations to be performed.

Method

Type Multiple Choice

Default value DIIS

Options [DIIS, MultiSecant]

Description Choose the general scheme used to converge the density in the SCF. In case of scf problems one can try the MultiSecant alternative at no extra cost per SCF cycle. For more details see the DIIS and MultiSecantConfig block.

Mixing

Type Float

Default value 0.075

Description Initial 'damping' parameter in the SCF procedure, for the iterative update of the potential: new potential = old potential + mix (computed potential-old potential). Note: the program automatically adapts Mixing during the SCF iterations, in an attempt to find the optimal mixing value.

PMatrix

Type Bool

Description If present, evaluate the charge density from the P-matrix. See also the key Eigenstates.

Rate

Type Float

Default value 0.99

Description Minimum rate of convergence for the SCF procedure. If progress is too slow the program will take measures (such as smearing out occupations around the Fermi level, see key Degenerate of block Convergence) or, if everything seems to fail, it will stop

VSplit

Type Float

Default value 0.05

Description To disturb degeneracy of alpha and beta spin MOs the value of this key is added to the beta spin potential at the startup.

4.6.2 Convergence

All options and parameters related to the convergence behavior of the SCF procedure are defined in the `Convergence` block key. Also the finite temperature distribution is part of this

```
Convergence
   Criterion float
   Degenerate string
   ElectronicTemperature float
   InitialDensity [rho | psi]
   LessDegenerate [True | False]
   NoDegenerate [True | False]
   SpinFlip string
   startwithmaxspin [True | False]
End
```

`Convergence`

Type Block

Description Options and parameters related to the convergence behavior of the SCF procedure.

`Criterion`

Type Float

Description Criterion for termination of the SCF procedure. The default depends on the NumericalQuality and on the number of atoms in the system.

`Degenerate`

Type String

Default value default

Description Smooths (slightly) occupation numbers around the Fermi level, so as to insure that nearly-degenerate states get (nearly-) identical occupations. Be aware: In case of problematic SCF convergence the program will turn this key on automatically, unless the key 'Nodegenerate' is set in input. The smoothing depends on the argument to this key, which can be considered a 'degeneration width'. When the argument reads default, the program will use the value 1e-4 a.u. for the energy width.

`ElectronicTemperature`

Type Float

Default value 0.0

Unit a.u.

Description Simulates a finite-temperature electronic distribution using the defined energy. This may be used to achieve convergence in an otherwise problematically converging system. The energy of a finite-T distribution is different from the T=0 value, but for small T a fair approximation of the zero-T energy is obtained by extrapolation. The extrapolation energy correction term is printed with the survey of the bonding energy in the output file. Check that this value is not too large. Build experience yourself how different settings may affect the outcomes. Note: this key is meant to help you overcome convergence problems, not to do

finite-temperature research! Only the electronic distribution is computed T-dependent, other aspects are not accounted for!

InitialDensity

Type Multiple Choice

Default value rho

Options [rho, psi]

Description The SCF is started with a guess of the density. There are the following choices RHO: the sum of atomic density. PSI: construct an initial eigensystem by occupying the atomic orbitals. The guessed eigensystem is orthonormalized, and from this the density is calculated/

LessDegenerate

Type Bool

Default value False

Description If smoothing of occupations over nearly degenerate orbitals is applied (see Degenerate key), then, if this key is set in the input file, the program will limit the smoothing energy range to 1e-4 a.u. as soon as the SCF has converged 'halfway', i.e. when the SCF error has decreased to the square root of its convergence criterion.

NoDegenerate

Type Bool

Default value False

Description This key prevents any internal automatic setting of the key DEGENERATE.

SpinFlip

Type String

Default value

Description List here the atoms for which you want the initial spin polarization to be flipped. This way you can distinguish between ferromagnetic and anti ferromagnetic states. Currently, it is not allowed to give symmetry equivalent atoms a different spin orientation. To achieve that you have to break the symmetry.

startwithmaxspin

Type Bool

Default value True

Description To break the initial perfect symmetry of up and down densities there are two strategies. One is to occupy the numerical orbitals in a maximum spin configuration. The alternative is to add a constant to the potential. See also Vsplit key.

4.6.3 DIIS

The DIIS procedure to obtain the SCF solution depends on several parameters. Default values can be overruled with this block.

```
DIIS
   Adaptable [True | False]
   CHuge float
   CLarge float
   Condition float
   DiMix float
   NCycleDamp integer
   NVctrx integer
   Variant [DIIS | LISTi | LISTb | LISTd]
End
```

DIIS

Type Block

Description Parameters for the DIIS procedure to obtain the SCF solution

Adaptable

Type Bool

Default value True

Description Change automatically the value of dimix during the SCF.

CHuge

Type Float

Default value 20.0

Description When the largest coefficient in the DIIS expansion exceeds this value, damping is applied

CLarge

Type Float

Default value 20.0

Description When the largest DIIS coefficient exceeds this value, the oldest DIIS vector is removed and the procedure re-applied

Condition

Type Float

Default value 1000000.0

Description The condition number of the DIIS matrix, the largest eigenvalue divided by the smallest, must not exceed this value. If this value is exceeded, this vector will be removed.

DiMix

Type Float

Default value 0.2

Description Mixing parameter for the DIIS procedure

NCycleDamp

Type Integer

Default value 1

Description Number of initial iterations where damping is applied, before any DIIS is considered

NVctrx

Type Integer

Default value 20

Description Maximum number of DIIS expansion vectors

Variant

Type Multiple Choice

Default value DIIS

Options [DIIS, LISTi, LISTb, LISTd]

Description Which variant to use. In case of problematic SCF convergence, first try MultiSecant, and if that does not work the LISTi is the advised method. Note: LIST is computationally more expensive per SCF iteration than DIIS.

4.6.4 Multi secant

```
MultiSecantConfig
   CMax float
   InitialSigmaN float
   MaxSigmaN float
   MaxVectors integer
   MinSigmaN float
End
```

MultiSecantConfig

Type Block

Description Parameters for the Multi-secant SCF convergence method.

CMax

Type Float

Default value 20.0

Description Maximum coefficient allowed in expansion

InitialSigmaN

Type Float

Default value 0.1

Description This is a lot like a mix factor: bigger means bolder

MaxSigmaN

Type Float

Default value 0.3

Description Upper bound for the SigmaN parameter

MaxVectors

Type Integer

Default value 20

Description Maximum number of previous cycles to be used

MinSigmaN

Type Float

Default value 0.01

Description Lower bound for the SigmaN parameter

4.6.5 DIRIS

In the DIRIS block, which has the same options as the `DIIS` block, you can specify the DIIS options to be used in the Dirac subprogram, for numerical single atom calculations, which constructs the radial tables for the NAOs.

4.7 More Technical Settings

There are of course many other settings influencing the precision and performance. Usually the user does not need to care about them.

4.7.1 Linear Scaling

```
Tails
   Bas float
End
```

Tails

Type Block

Description Ignore function tails.

Bas

Type Float

Default value 1e-06

Description Cut off the basis functions when smaller than the specified threshold.

4.7.2 Dependency

```
Dependency
   Basis float
   Core float
   CoreValence float
   Fit float
End
```

Dependency

Type Block

Description Criteria for linear dependency of the basis and fit set

Basis

Type Float

Default value 1e-08

Description Criteria for linear dependency of the basis: smallest eigenvalue of the overlap matrix of normalized Bloch functions.

Core

Type Float

Default value 0.98

Description The program verifies that the frozen core approximation is reasonable, by checking the smallest value of the overlap matrix of the core (Bloch) orbitals against this criterion.

CoreValence

Type Float

Default value 1e-05

Description Criterion for dependency of the core functions on the valence basis. The maximum overlap between any two normalized functions in the two respective function spaces should not exceed 1.0-corevalence

Fit

Type Float

Default value 5e-06

Description Criterion for dependency of the total set of fit functions. The value monitored is the smallest eigenvalue of the overlap matrix of normalized Bloch sums of symmetrized fit functions.

4.7.3 Screening

Band performs many lattice summations which are in practice truncated. The two prime examples are the construction of the Bloch basis and the calculation of the solvation potential. The precision of the lattice summations is controlled by the `Screening` key

```
Screening
   CutOff float
   DMadel float
   NoDirectionalScreening [True | False]
   RCelx float
   RMadel float
End
```

Screening

Type Block

Description For the periodic solvation potential and for the old (not default anymore) fitting method, BAND performs lattice summations which are in practice truncated. The precision of the lattice summations is controlled by the options in this block.

CutOff

Type Float

Description Criterion for negligibility of tails in the construction of Bloch sums. Default depends on Accuracy.

DMadel

Type Float

Description One of the parameters that define the screening of Coulomb-potentials in lattice sums. Depends by default on Accuracy, rmadel, and rcelx. One should consult the literature for more information

NoDirectionalScreening

Type Bool

Description Real space lattice sums of slowly (or non-) convergent terms, such as the Coulomb potential, are computed by a screening technique. In previous releases, the screening was applied to all (long-range) Coulomb expressions. Screening is only applied in the periodicity directions. This key restores the original situation: screening in all directions

RCelx

Type Float

Description Max. distance of lattice site from which tails of atomic functions will be taken into account for the Bloch sums. Default depends on Accuracy.

RMadel

Type Float

Description One of the parameters that define screening of the Coulomb potentials in lattice summations. Depends by default on Accuracy, dmadel, rcelx. One should consult the literature for more information.

4.7.4 Direct (on the fly) calculation of basis and fit

BAND usually calculates basis functions and theirs derivatives on the fly. However, for small bulk systems it can be faster to write the information to disk. Then one can set the `DirectBas` key to false. (**Default = true**)

```
Programmer
   DirectBas bool
End
```

4.7.5 Fermi energy search

```
Fermi
   Delta float
   Eps float
   MaxTry integer
End
```

Fermi

Type Block

Description Technical parameter used in determining the Fermi energy, which is carried out at each cycle of the SCF procedure.

Delta

Type Float

Default value 0.0001

Description Convergence criterion: upper and lower bounds for the Fermi energy and the corresponding integrated charge volumes must be equal within delta.

Eps

Type Float

Default value 1e-10

Description After convergence of the Fermi energy search procedure, a final estimate is defined by interpolation and the corresponding integrated charge volume is tested. It should be exact, to machine precision. Tested is that it deviates not more than eps.

MaxTry

Type Integer

Default value 15

Description Maximum number of attempts to locate the Fermi energy. The procedure is iterative in nature, narrowing the energy band in which the Fermi energy must lie, between an upper and a lower bound. If the procedure has not converged sufficiently within MaxTry iterations, the program takes a reasonable value and constructs the charge density by interpolation between the functions corresponding to the last used upper and lower bounds for the Fermi energy.

4.7.6 Block size

```
CPVector integer
```

CPVector

Type Integer

Default value 128

Description The code is vectorized and this key can be used to set the vector length

```
KGrpX integer
```

KGrpX

Type Integer

Default value 5

Description Absolute upper bound on the number of k-points processed together. This only affects the computational performance.

CHAPTER

FIVE

SPECTROSCOPY AND PROPERTIES

5.1 Frequencies and Phonons

Frequencies and Phonons and can be computed via numerical differentiation by the AMS diver. See the Normal Modes section or the Phonon section of the AMS manual.

Several thermodynamic properties, such as Zero-point Energy, Internal Energy, Entropy, Free Energy and Specific Heat are computed by default when calculating Phonons.

5.2 Elastic Tensor

The elastic tensor (and related elastic properties such as Bulk modulus, Shear modulus and Young modulus) can be computed via numerical differentiation by AMS. See the Elastic Tensor section of the AMS manual.

When calculating the elastic tensors using Band one should disable *Symmetry* (page 111).

5.3 Optical Properties: Time-Dependent Current DFT

Time-Dependent Current Density Functional Theory (**TD-CDFT**) is a theoretical framework for computing optical response properties, such as the frequency-dependent dielectric function.

In this section, the TD-CDFT implementation for extended systems (1D, 2D and 3D) in BAND is described. The input keys are described in *NewResponse* (page 61) or in *OldResponse* (page 65).

Some examples are available in the `$ADFHOME/examples/band` directory and are discussed in the Examples section.

- Tutorial: Silicon (OldResponse)
- Tutorial: MoS2 Monolayer (NewResponse)
- *Example: TD-CDFT for bulk diamond (OldResponse)* (page 185)

5.3.1 Insulators, semiconductors and metals

The TD-CDFT module enables the calculation of real and imaginary parts of the material property tensor $\chi_e(\omega)$, called the **electric susceptibility**. The electric susceptibility is related to the macroscopic **dielectric function**, $\varepsilon_M(\omega)$.

For semi-conductors and insulator, for which the bands are either fully occupied or fully unoccupied, the dielectric function $\varepsilon_M(\omega)$ comprises only of the so called interband component:

$$\varepsilon_M(\omega) = 1 + 4\pi\chi_e(\omega)$$

In general $\chi_e(\omega)$ and $\varepsilon_M(\omega)$ are tensors. They, however, simplify to scalars in isotropic systems.

For metals, for which partially-occupied bands exist, there is a so called intraband component arising due to transitions within a partially-occupied band:

$$\varepsilon_M(\omega) = 1 + 4\pi\chi_e(\omega) - 4\pi i\sigma_e(\omega)/\omega$$

5.3.2 Frequency dependent kernel

It is known that the exact Vignale-Kohn (VK) kernel greatly improves the static polarizabilities of infinite polymers and nanotubes (see reference (https://doi.org/10.1063/1.2102899)), but gives bad results for the optical spectra of semiconductors and metals. For the low frequency part one needs a frequency dependent kernel, since Drude-like tails are completely absent in the adiabatic local density approximation (ALDA). With a modified VK kernel, which neglects μ_{xc} so that it reduces to the ALDA form in the static limit (see reference (https://doi.org/10.1103/PhysRevB.74.245117)), much better results can be obtained. BAND currently only supports the modified VK kernel in either the QV or CNT parametrization, and it should **only be used for metals**.

5.3.3 EELS

From the macroscopic dielectric function it is possible to calculate the electron energy loss function (EELS). In transmission EELS one studies the inelastic scattering of a beam of high energy electrons by a target. The scattering rates obtained in these experiments are related to the dynamical structure factor $S(q,\omega)$ [A1]. In the special case with wavevector $q = 0$, $S(q,\omega)$ is related to the longitudinal macroscopic dielectric function. This is the long-wave limit of EELS. For isotropic system the dielectric function is simply a scalar ($1/3\mathrm{Tr}(\varepsilon_M(\omega))$). In this case the long-wave limit of the electron energy loss function assumes the trivial form

$$\lim_{q\to 0} 2\pi \frac{S(q,\omega)}{q^2 V} = \frac{\varepsilon_2}{\varepsilon_1^2 + \varepsilon_2^2}$$

with ε_1 and ε_2 as real and imaginary part of the dielectric function.

References

The three related Ph.D. theses, due to F. Kootstra (on TD-DFT for insulators), P. Romaniello (on TD-CDFT for metals), and A. Berger (on the Vignale-Kohn functional in extended systems) contain much background information, and can be downloaded from the SCM website (http://www.scm.com).

The most relevant publications on this topic due to the former "Groningen" group of P.L. de Boeij are [*22* (page 226)], [*23* (page 226)], [*24* (page 226)], [*25* (page 226)].

[A1] S. E. Schnatterly, in Solid State Physics Vol.34, edited by H. Ehrenreich, F. Seitz, and D. Turnbull (Academic Press, Inc., New York, 1979).

5.3.4 Input Options

In the 2017 release of BAND there are two implementations of the TD-CDFT formalism. The original implementation, relying on obsolete algorithms of BAND, is accessible via the *OldResponse* (page 65) key block. The new code section, relying on more modern algorithms of BAND, is accessible via the *NewResponse* (page 61), *NewResponseSCF* (page 62) and *NewResponseKSpace* (page 64) key blocks. The differences between the two flavours are summarized in the following table:

	OldResponse	NewResponse
3D-systems	yes	yes
2D-systems	no	yes
1D-systems	(yes)	yes
Semiconductors	yes	yes
Metals	yes	(yes)
ALDA	yes	yes
Vignale-Kohn	yes	no
Berger2015 (3D)	yes	yes
Scalar ZORA	yes	yes
Spin Orbit ZORA	yes	no

Besides these differences, one should not expect both flavours to give the exact same result, if the reciprocal space limit is not reached! This can be explained by different approaches to evaluate the integration weights of single-particle transitions in recirpocal space.

Attention: Response properties **converge slowly** with respect to k-space sampling (number of k-points). **Always check the convergence of ε_M with respect to** *K-Space* (page 39) **options!!!**

NewResponse

The dielectric function is computed when the key block *NewResponse* (page 61) is present in the input. Several important settings can be defined in this key block.

Additional details can be specified via the *NewResponseKSpace* (page 64) and *NewResponseSCF* (page 62) blocks.

```
NewResponse
   NFreq integer
   FreqLow float
   FreqHigh float
   EShift float
   ActiveESpace float
   DensityCutOff float
   ActiveXYZ string
End
```

NewResponse

Type Block

Description The TD-CDFT calculation to obtain the dielectric function is computed when this block is present in the input. Several important settings can be defined here.

NFreq

Type Integer

Default value 5

Description Number of frequencies for which a linear response TD-CDFT calculation is performed.

FreqLow

Type Float

Default value 1.0

Unit eV

Description Lower limit of the frequency range for which response properties are calculated. (omega_{low})

FreqHigh

Type Float

Default value 3.0

Unit eV

Description Upper limit of the frequency range for which response properties are calculated (omega_{high}).

EShift

Type Float

Default value 0.0

Unit eV

Description Energy shift of the virtual crystal orbitals.

ActiveESpace

Type Float

Default value 5.0

Unit eV

Description Modifies the energy threshold (DeltaE^{max}_{thresh} = omega_{high} + ActiveESpace) for which single orbital transitions (DeltaEpsilon_{ia} = Epsilon_{a}^{virtual} - Epsilon_{i}^{occupied}) are taken into account.

DensityCutOff

Type Float

Default value 0.001

Description For 1D and 2D systems the unit cell volume is undefined. Here, the volume is calculated as the volume bordered by the isosurface for the value DensityCutoff of the total density.

ActiveXYZ

Type String

Default value t

Description Expects a string consisting of three letters of either 'T' (for true) or 'F' (for false) where the first is for the X-, the second for the Y- and the third for the Z-component of the response properties. If true, then the response properties for this component will be evaluated.

```
NewResponseSCF
   Bootstrap integer
   COApproach [True | False]
   COApproachBoost [True | False]
   Criterion float
   DIIS [True | False]
   LowFreqAlgo [True | False]
   Mixing float
   NCycle integer
```

```
   XC integer
End
```

NewResponseSCF

Type Block

Description Details for the linear-response self-consistent optimization cycle. Only influencing the NewResponse code.

Bootstrap

Type Integer

Default value 0

Description defines if the Berger2015 kernel (Bootstrap 1) is used or not (Bootstrap 0). If you chose the Berger2015 kernel, you have to set NewResponseSCF%XC to '0'. Since it shall be used in combination with the bare Coulomb response only. Note: The evaluation of response properties using the Berger2015 is recommend for 3D systems only!

COApproach

Type Bool

Default value True

Description The program automatically decides to calculate the integrals and induced densities via the Bloch expanded atomic orbitals (AO approach) or via the cyrstal orbitals (CO approach). The option COApproach overrules this decision.

COApproachBoost

Type Bool

Default value False

Description Keeps the grid data of the Crystal Orbitals in memory.Requires significantly more memory for a speedup of the calculation. One might have to use multiple computing nodes to not run into memory problems.

Criterion

Type Float

Default value 0.001

Description For the SCF convergence the RMS of the induced density change is tested. If this value is below the Criterion the SCF is finished.Furthermore, one can find the calculated electric susceptibility for each SCF step in the output and can therefore decide if the default value is too loose or too strict.

DIIS

Type Bool

Default value True

Description In case the DIIS method is not working, one can switch to plain mixing by setting DIIS to false.

LowFreqAlgo

Type Bool

Default value True

Description Numerically more stable results for frequencies lower than 1.0 eV. Note: for a graphene monolayer the conical intersection results in a very small band gap (zero band gap semi-conductor). This leads ta a failing low frequency algorithm. One can then chose to use the algoritm as originally proposed by Kootstra by setting the input value to *false*. But, this can result in unreliable results for frequencies lower than 1.0 eV!

Mixing

Type Float

Default value 0.2

Description Mixing value for the SCF optimization.

NCycle

Type Integer

Default value 20

Description Number of SCF cycles for each frequency to be evaluated.

XC

Type Integer

Default value 1

Description Influences if the bare induced Coulomb response (XC 0) is used for the effective, induced potential or the induced potential derived from the ALDA kernel as well (XC 1).

```
NewResponseKSpace
   Eta float
   SubSimp integer
End
```

NewResponseKSpace

Type Block

Description Modify the details for the integration weights evaluation in reciprocal space for each single-particle transition. Only influencing the NewResponse code.

Eta

Type Float

Default value 1e-05

Description Defines the small, finite imaginary number i*eta which is necessary in the context of integration weights for single-particle transitions in reciprocal space.

SubSimp

Type Integer

Default value 3

Description determines into how many sub-integrals each integration around a k point is split. This is only true for so-called quadratic integration grids. The larger the number the better the convergence behavior for the sampling in reciprocal space. Note: the computing time for the weights is linear for 1D, quadratic for 2D and cubic for 3D!

OldResponse

```
OldResponse
   Berger2015 [True | False]
   CNT [True | False]
   CNVI float
   CNVJ float
   Ebndtl float
   Enabled [True | False]
   Endfr float
   Isz integer
   Iyxc integer
   NewVK [True | False]
   Nfreq integer
   QV [True | False]
   Shift float
   Static [True | False]
   Strtfr float
End
```

OldResponse

Type Block

Description Options for the old TD-CDFT implementation.

Berger2015

Type Bool

Default value False

Description Use the parameter-free polarization functional by A. Berger (Phys. Rev. Lett. 115, 137402). This is possible for 3D insulators and metals. Note: The evaluation of response properties using the Berger2015 is recommend for 3D systems only!

CNT

Type Bool

Description Use the CNT parametrization for the longitudinal and transverse kernels of the XC kernel of the homogeneous electron gas. Use this in conjunction with the NewVK option.

CNVI

Type Float

Default value 0.001

Description The first convergence criterion for the change in the fit coefficients for the fit functions, when fitting the density.

CNVJ

Type Float

Default value 0.001

Description the second convergence criterion for the change in the fit coefficients for the fit functions, when fitting the density.

Ebndtl

Type Float

Default value 0.001

Unit Hartree

Description the energy band tolerance, for determination which routines to use for calculating the numerical integration weights, when the energy band posses no or to less dispersion.

Enabled

Type Bool

Default value False

Description If true, the response function will be calculated using the old TD-CDFT implementation

Endfr

Type Float

Default value 3.0

Unit eV

Description The upper bound frequency of the frequency range over which the dielectric function is calculated

Isz

Type Integer

Default value 0

Description Integer indicating whether or not scalar zeroth order relativistic effects are included in the TDCDFT calculation. 0 = relativistic effects are not included, 1 = relativistic effects are included. The current implementation does NOT work with the option XC%SpinOrbitMagnetization equal NonCollinear

Iyxc

Type Integer

Default value 0

Description integer for printing yxc-tensor (see http://aip.scitation.org/doi/10.1063/1.1385370). 0 = not printed, 1 = printed.

NewVK

Type Bool

Description Use the slightly modified version of the VK kernel (see https://aip.scitation.org/doi/10.1063/1.1385370). When using this option one uses effectively the static option, even for metals, so one should check carefully the convergence with the KSPACE parameter.

Nfreq

Type Integer

Default value 5

Description the number of frequencies for which a linear response TD-CDFT calculation is performed.

QV

Type Bool

Description Use the QV parametrization for the longitudinal and transverse kernels of the XC kernel of the homogeneous electron gas. Use this in conjunction with the NewVK option. (see reference).

Shift

Type Float

Default value 0.0

Unit eV

Description energy shift for the virtual crystal orbitals.

Static

Type Bool

Description An alternative method that allows an analytic evaluation of the static response (normally the static response is approximated by a finite small frequency value). This option should only be used for non-relativistic calculations on insulators, and it has no effect on metals. Note: experience shows that KSPACE convergence can be slower.

Strtfr

Type Float

Default value 1.0

Unit eV

Description is the lower bound frequency of the frequency range over which the dielectric function is calculated.

5.4 ESR/EPR

BAND is capable to calculate electron paramagnetic resonance (EPR) parameters for paramagnetic defects in solids: hyperfine A-tensor and the Zeeman g-tensor.

The implementation of EPR parameters in BAND is described in the publications by Kadantsev and co-workers [*20* (page 226)] and [*21* (page 226)].

Hyperfine A-tensor

The A-tensor is implemented within the non-relativistic and scalar relativistic, spin-polarized Kohn-Sham scheme.

```
ATensor
   Enabled [True | False]
End
```

ATensor

Type Block

Description Hyperfine A-tensor.

Enabled

Type Bool

Default value False

Description Compute the hyperfine A-tensor.Note: Unrestricted calculation is required.

Two methods are used for A-tensor calculation:

- Method 1: involves the gradient of the spin-polarized density and integration by parts. The isotropic component of the A-tensor obtained through integration, in a "non-local fashion".
- Method 2: the A-tensor is computed from spin-polarized density and does not relies on the integration by parts. The isotropic component is obtained in a "local fashion" from the value of the spin-polarized density on the grid points near the nuclei.

The user should be aware that numerical integration in A- and g-tensor routines is carried out over the Wigner-Seitz (WS) cell, and, therefore, to obtain a meaningful result, the defect in question should lie at or very close to the WS cell origin. This might require, on the user's part, some modification of the input geometry.

It also might happen that the size of the WS cell is not large enough for the adequate description of the paramagnetic defect in question. In this case, Method 1 will fails, since it relies on the integration by parts and assumes that the spin-polarized density is localized inside the WS cell. For the same reason, we recommend that the user removes diffuse basis set functions that describe the defect subsystem.

Finally, we note that the final result for A-tensor as presented by BAND is not scaled by the nuclear spin (as it is done in ADF) and the user is responsible for making necessary adjustments.

g-tensor

The calculation of the Zeeman g-tensor is invoked within the ESR block:

```
ESR
   Enabled [True | False]
End
```

ESR

Type Block

Description Zeeman g-tensor. The Zeeman g-tensor is implemented using two-component approach of Van Lenthe and co-workers in which the g-tensor is computed from a pair of spinors related to each other by time-reversal symmetry. Note: the following options are necessary for ESR: 'Relativistic zora spin' and 'Kspace 1'

Enabled

Type Bool

Default value False

Description Compute Zeeman g-tensor.The Zeeman g-tensor is implemented using two-component approach of Van Lenthe and co-workers in which the g-tensor is computed from a pair of spinors related to each other by time-reversal symmetry.Note: the following options are necessary for ESR: 'Relativistic zora spin' and 'Kspace 1'

(Γ-only calculation). The g-tensor is then computed from the HOMO spinor at the Γ point. In the output, the user can find two-contributions to the g-tensor: one that stems from the K_σ operator and a second one, that stems from the orbital angular momentum. By default, GIAO and spin-Zeeman corrections are **not** included. From our experience, these corrections are quite small.

5.5 Nuclear Quadrupole Interaction (EFG)

```
EFG
   Enabled [True | False]
End
```

EFG

Type Block

Description The electronic charge density causes an electric field, and the gradient of this field couples with the nuclear quadrupole moment, that some (non-spherical) nuclei have and can be measured by several spectroscopic techniques. The EFG tensor is the second derivative of the Coulomb potential at the nuclei. For each atom it is a 3x3 symmetric and traceless matrix. Diagonalization of this matrix gives three eigenvalues, which are usually ordered by their decreasing absolute size and denoted as V_{xx}, V_{yy}, V_{zz}. The result is summarized by the largest eigenvalue and the asymmetry parameter.

Enabled

Type Bool

Default value False

Description Compute the EFG tensor (for nuclear quadrupole interaction).

This option honors the `SelectedAtoms` key, in which case only the EFG will be calculated for the selected atoms.

5.6 NMR

Warning: The calculations of NMR shielding with BAND has not been thorougly tested. One should be extra careful when running NMR calculation, and validate the results by using different super-cells and different technical parameters.

With the NMR option the *shielding tensor* is calculated. There are two methods implemented: the super cell method and the single-dipole method.

1. The super cell method is according to the implementation by Skachkov *et al.* [*37* (page 227)] The symmetry will be automatically disabled. The unit cell should not be chosen too small.
2. The other method is the single-dipole method. In principle one can now use the primitive cell.[*48* (page 227)] In practice also this method needs to be converged with super cell size. However, depending on the system the required super cell may be much smaller. At a given super cell size this method is more expensive than the super cell method.

```
NMR
   Enabled [True | False]
   SuperCell [True | False]
End
```

NMR

Type Block

Description Options for the calculations of the NMR shielding tensor.

Enabled

Type Bool

Default value False

Description Compute NMR shielding.

SuperCell

Type Bool

Default value True

Description This is the switch between the two methods, either the super cell (true), or the single-dipole method (false)

This option honors the `SelectedAtoms` key, in which case only the NMR properties will be calculated for the selected atoms only.

5.7 Effective Mass

```
EffectiveMass
   Enabled [True | False]
   KPointCoord float_list
   NumAbove integer
   NumBelow integer
   StepSize float
End
```

`EffectiveMass`

Type Block

Description In a semi-conductor, the mobility of electrons and holes is related to the curvature of the bands at the top of the valence band and the bottom of the conduction band.With the effective mass option, this curvature is obtained by numerical differentiation.The estimation is done with the specified step size, and twice the specified step size, and both results are printed to give a hint on the accuracy. The easiest way to use this key is to enabled it without specifying any extra options.

`Enabled`

Type Bool

Default value False

Description Compute the EffectiveMass.

`KPointCoord`

Type Float List

Unit 1/Bohr

Recurring True

Description Coordinate of the k-points for which you would like to compute the effective mass.

`NumAbove`

Type Integer

Default value 1

Description Number of bands to take into account above the Fermi level.

`NumBelow`

Type Integer

Default value 1

Description Number of bands to take into account below the Fermi level.

StepSize

Type Float

Default value 0.001

Description Size of the step taken in reciprocal space to perform the numerical differentiation

5.8 Properties at Nuclei

PropertiesAtNuclei (block-type) A number of properties can be obtained near the nucleus. An average is taken over a tiny sphere around the nucleus. The following properties are available.

```
PropertiesAtNuclei
  vxc[rho(fit)]
  rho(fit)
  rho(scf)
  v(coulomb/scf)
  rho(deformation/fit)
  rho(deformation/scf)
End
```

The electron density, `rho(scf)`, is physically the most relevant one.

5.9 X-Ray Form Factors

X-ray structure factors (Fourier analysis of the charge density) are computed by default after termination of the SCF procedure.

Form factors options:

```
FormFactors integer
```

FormFactors

Type Integer

Default value 2

Description Number of stars of K-vectors for which the form factors are computed

CHAPTER

SIX

ANALYSIS

6.1 Density of States (DOS)

```
DOS
   DeltaE float
   Enabled [True | False]
   Energies integer
   File string
   IntegrateDeltaE [True | False]
   Max float
   Min float
   StoreCoopPerBasPair [True | False]
End
```

DOS

Type Block

Description Density-Of-States (DOS) options

DeltaE

Type Float

Default value 0.005

Unit Hartree

Description Energy step for the DOS grid. Using a smaller value (e.g. half the default value) will result in a finer sampling of the DOS.

Enabled

Type Bool

Default value False

Description Whether or not to calculate the density of states.

Energies

Type Integer

Description Number of equidistant energy-values for the DOS grid. This keyword supersedes the 'DeltaE' keyword.

File

Type String

Description Write the DOS (plain text format) to the specified file instead of writing it to the standard output.

IntegrateDeltaE

Type Bool

Default value True

Description This subkey handles which algorithm is used to calculate the data-points in the plotted DOS. If true, the data-points represent an integral over the states in an energy interval. Here, the energy interval depends on the number of Energies and the user-defined upper and lower energy for the calculation of the DOS. The result has as unit [number of states / (energy interval * unit cell)]. If false, the data-points do represent the number of states for a specific energy and the resulting plot is equal to the DOS per unit cell (unit: [1/energy]). Since the resulting plot can be a wild function and one might miss features of the DOS due to the step length between the energies, the default is set to the integration algorithm.

Max

Type Float

Default value 0.75

Unit Hartree

Description Upper bound energy (with respect to the Fermi energy)

Min

Type Float

Default value -0.75

Unit Hartree

Description Lower bound energy (with respect to the Fermi energy)

StoreCoopPerBasPair

Type Bool

Default value False

Description Calculate the COOP (crystal orbital overlap population).

An example input:

```
DOS
   Enabled     True
   Energies    500
   Min        -0.35
   Max         1.05
   File        plotfile
End
```

According to this example, DOS values will be generated in an equidistant mesh of 500 energy values, ranging from 0.35 a.u. below the Fermi level to 1.05 a.u. above it. All information will be written to a file plotfile. The information on the plot file is a long list of pairs of values (energy and DOS), with some informative text-headers and general information. DOS values are generated for the total DOS and optionally also for some partial DOS (see the keys *GrossPopulations* (page 75) and *OverlapPopulations* (page 75)).

In the **DOS** and **Band Structure GUI modules**, it is possible to visualize partial density of states (**p-DOS**). The partial contributions are obtained from the total DOS by following the **Mulliken population analysis** partitioning prescription (see wikipedia (https://en.wikipedia.org/wiki/Mulliken_population_analysis)).

6.1.1 Gross populations

```
GrossPopulations # Non-standard block. See details.
   ...
End
```

GrossPopulations

Type Non-standard block

Description Partial DOS (pDOS) are generated for the gross populations listed under this key. See example.

Syntax:

```
GrossPopulations
   {iat lq}
   {FragFun jat ifun}
   {Frag kat}
   {Sum
    ...
    EndSum}
End
```

iat pDOS is generated for atom *lq*.

FragFun pDOS is generated for atom *jat* with all real spherical harmonics belonging to *l*-value *ifun*.

Frag pDOS of the functions belonging to atom *kat* will be calculated.

Sum sum all pDOS, specified in this block.

Example:

```
GrossPopulations
   FragFun 1 2:: Second function of first atom
   Frag 2 :: Sum of all functions from second atom
   SUM:: sum following PDOSes
      Frag 1::Atom nr.1
      FragFun 2 1::First function of second atom
      5 1:: All pfunctions of fifth atom
   EndSum
End
```

6.1.2 Overlap populations

```
OverlapPopulations # Non-standard block. See details.
   ...
End
```

OverlapPopulations

Type Non-standard block

Description Overlap population weighted DOS (OPWDOS), also known as the crystal orbital overlap population (COOP).

Overlap population weighted DOS are generated for the overlap populations listed:

```
OVERLAPPOPULATIONS
   Left
      { iat lq }
      { FragFun jat ifun }
      { Frag kat }
   Right
      ...
End
```

You can use this to get the OPWDOS of two functions, or, if you like, one bunch of functions with another bunch of functions. The key-block should consist of left-right pairs. After a line with left you enter lines that specify one or more functions (according to *GrossPopulations* (page 75)), followed by a similar structure beginning with right, which will produce the OPWDOS of the left functions with the right functions.

Example:

```
OVERLAPPOPULATIONS
   LEFT::First OPWDOS
      Frag 1
   RIGHT
      Frag 2
   LEFT:: Next OPWDOS
      FragFun 1 1
   RIGHT
      2 1
      FragFun 3 5
End
```

6.2 Band Structure

BAND can calculate the band structure for the standard k-path in the Brillouin zone [*65* (page 228)] and saves the corresponding data to the binary file RUNKF.

The band structure is best examined with the GUI module **BandStructure** (see BAND-GUI tutorial Getting Started with BAND).

```
BandStructure
   Enabled [True | False]
   Automatic [True | False]
   DeltaK float
   FatBands [True | False]
   UseSymmetry [True | False]
   EnergyAboveFermi float
   EnergyBelowFermi float
End
```

`BandStructure`

Type Block

Description Options for the calculation of the band structure.

`Enabled`

Type Bool

Default value False

Description If True, Band will calculate the band structure and save it to file for visualization.

Automatic

Type Bool

Default value True

Description If True, BAND will automatically generate the standard path through the Brillouin zone.If False BAND will use the user-defined path in BZPath.

DeltaK

Type Float

Default value 0.1

Unit 1/Bohr

Description Step (in reciprocal space) for band structure interpolation.Using a smaller number (e.g. 0.03) will result in smoother band curves at the cost of an increased computation time.

FatBands

Type Bool

Default value True

Description If True, BAND will compute the fat bands (only if BandStructure%Enabled is True).The Fat Bands are the periodic equivalent of the Mulliken population analysis.

UseSymmetry

Type Bool

Default value True

Description If True, only the irreducible wedge of the Wigner-Seitz cell is sampled.If False, the whole (inversion-unique) Wigner-Seitz cell is sampled.Note: The Symmetry key does not influence the symmetry of the band structure sampling.

EnergyAboveFermi

Type Float

Default value 0.75

Unit Hartree

Description Bands with minimum energy larger then FermiEnergy + EnergyAboveFermi are not saved to file. Increasing the value of EnergyAboveFermi will result in more unoccupied bands to be saved to file for visualization.

EnergyBelowFermi

Type Float

Default value 10.0

Unit Hartree

Description Bands with maximum energy smaller then FermiEnergy - EnergyBelowFermi are not saved to file. Increasing the value of EnergyBelowFermi will result in more occupied core bands to be saved to file for visualization. Note: EnergyBelowFermi should be a positive number!

Information on the k-path used for band structure plotting (including the fractional coordinates of high-symmetry k-points) can be found in the section `KPath` of the output file.

6.2.1 User-defined path in the Brillouin zone

If `BZStruct%Automatic` is `False`, BAND will compute the band structure for the user-defined path in the `BZPath` block.

```
BZPath
   path # Non-standard block. See details.
      ...
   End
End
```

BZPath

Type Block

Description Definition of the user-defined path in the Brillouin zone for band structure plotting.

path

Type Non-standard block

Recurring True

Description Definition of the k-points in a path. The vertices of your path should be defined in fractional coordinates (wrt the reciprocal lattice vectors)

You should define the vertices of your path in fractional coordinates (wrt the reciprocal lattice vectors) in the `Path` sub-block. If you want to make a *jump* in your path, you need to specify a new `Path` sub-block.

In the following example we define the path `Gamma-X-W-K|U-X` for a FCC lattice:

```
BZPath
   Path
      0.000   0.000   0.000
      0.500   0.000   0.500
      0.500   0.250   0.750
      0.375   0.375   0.750
   SubEnd
   Path
      0.625   0.250   0.625
      0.500   0.000   0.500
   SubEnd
End
```

6.2.2 Definition of the Fat Bands

The *fat bands* (page 76) $F_{i,n,\sigma,\vec{k}}$ are the periodic equivalent of the Mulliken population. They are defined as:

$$F_{i,n,\sigma,\vec{k}} = \sum_j C_{i,n,\sigma,\vec{k}} C_{j,n,\sigma,\vec{k}} S_{i,j,\vec{k}}$$

where $C_{i,n,\sigma,\vec{k}}$ and $S_{i,j,\vec{k}}$ are the orbital coefficients and the overlap matrix elements respectively. The indices i and j denote basis functions, n is the band index, σ is the spin index and $\vec{k}$ is a reciprocal vector in the Brillouin zone.

6.2.3 Band Gap

The band gap (if any) is printed in the output. Here is an example for the NaCl crystal:

```
---------------------------------------
Band gap information
---------------------------------------
Number of valence electrons                  16
Valence Band index                            8
Top of valence Band (a.u.)                -0.192
Bottom of conduction Band (a.u.)          -0.039
Band gap (a.u.)                            0.153
Band gap (eV)                              4.173
Band gap (kcal)                           96.235
```

6.3 Charges

6.3.1 Default Atomic Charge Analysis

By default BAND computes the following atomic charge analyses:

- **Hirshfeld Charges** [*69* (page 229), *70* (page 229)]
- **Voronoi Deformation Charges** (VDD, Voronoi Deformation Density)
- **Mulliken Charges** (note: not calculated for *Spin-Orbit* (page 22) calculations)
- **CM5** (Charge Model 5) [*71* (page 229)]

These atomic charges are printed to the output file and can be visualized using the ADFView GUI module.

A more detailed output of the atomic charges can be printed by specifying following print option (note: in Band 2017 and previous versions this detailed output was printed by default):

```
Print AtomicChargesDetails
```

6.3.2 Bader Analysis (AIM)

The QTAIM (Quantum Theory of Atoms in Molecules), also known as Bader Analysis can be enabled in the GridBasedAIM input block:

```
GridBasedAIM
   Enabled [True | False]
   Iterations integer
   SmallDensity float
   UseStartDensity [True | False]
End
```

GridBasedAIM

Type Block

Description Invoke the ultra fast grid based Bader analysis.

Enabled

Type Bool

Default value False

Description Invoke the ultra fast grid based Bader analysis.

Iterations

Type Integer

Default value 40

Description The maximum number of steps that may be taken to find the nuclear attractor for a grid point.

SmallDensity

Type Float

Default value 1e-06

Description Value below which the density is ignored. This should not be chosen too small because it may lead to unassignable grid points.

UseStartDensity

Type Bool

Default value False

Description Whether the analysis is performed on the startup density (True) or on the final density (False).

```
AIMCriticalPoints
   Enabled [True | False]
   EqvPointsTol float
   GridPadding float
   GridSpacing float
End
```

AIMCriticalPoints

Type Block

Description Compute the critical points of the density (Atoms In Molecules). The algorithm starts from a regular mesh of points, and from each of these it walks towards its corresponding critical point.

Enabled

Type Bool

Default value False

Description Compute the critical points of the density (Atoms In Molecules). The algorithm starts from a regular mesh of points, and from each of these it walks towards its corresponding critical point.

EqvPointsTol

Type Float

Default value 0.27

Unit Bohr

Description If the distance between two critical points is smaller than this value, the two critical points are considered to be the same point.

GridPadding

Type Float

Default value 0.7

Unit Bohr

Description How much extra space is added to the starting guess domain in the search for the critical points

GridSpacing

Type Float

Default value 0.5

Unit Bohr

Description The distance between the initial trial points.

Note: The Bader (AIM) analysis is performed on the fitted density (see *ZlmFit* (page 45)). We advise to use a Good (or better) ZlmFit quality.

6.4 Fragments

A fragment feature is available albeit rather primitive. It allows for the analysis of the DOS in a fragment basis and for the calculation of the deformation density with respect to fragment densities. A typical application is the periodical adsorption of one or more molecules on a surface. For instance, consider periodic adsorption of hydrogen molecules over a surface. First you calculate the free molecule in the same orientation as when adsorbed to the substrate. Since you would like to use a molecular fragment, it makes sense to put the molecules far apart (large lattice spacing) and force dispersion to be neglected (KSPACE 1). To use the fragment in the next run you need to rename the result file ("rkf"), to something like "frag.rkf", see the example script discussed below or the *example* (page 192) covering this topic.

```
Fragment
   AtomMapping # Non-standard block. See details.
      ...
   End
   FileName string
   Labels # Non-standard block. See details.
      ...
   End
End
```

Fragment

Type Block

Recurring True

Description Defines a fragment. You can define several fragments for a calculation.

AtomMapping

Type Non-standard block

Description Format 'indexFragAt indexCurrentAt'. One has to associate the atoms of the fragment to the atoms of the current calculation. So, for each atom of the fragment the indexFragAt has to be associated uniquely to the indexCurrentAt for the current calculation.

FileName

Type String

Description File name of the fragment. Absolute path or path relative to the executing directory.

Labels

Type Non-standard block

Description This gives the possibility to introduce labels for the fragment orbitals. See examples.

Example:

```
Fragment
  filename test.rkf
  AtomMapping
    1 3 !  atom 1 of this fragment is assigned to third atom
    2 4 !  atom 2 of this fragment is assigned to fourth atom
  End
  Labels
   Sigma
   Sigma*
   Pi_x
   Pi_y
   Pi_x*
   Pi_y*
  Subend
End
```

In this example the first six fragment orbitals will be labeled as stated in the body of this key. The remaining orbitals are labeled by the default labeling system (e.g. 1/FO/5, etc.). The labels are used in combination with options like `Print Eigens` and `Print OrbPop`. (See also `Print OrbLabels`). This key can be given once for each fragment.

Tip: Specifying:

```
Print Eigens
```

for a calculation produces output concerning the eigen states, thereby providing a means to identify the eigen states (e.g. to be sigma, pi, et cetera). So, one can label the orbitals of a fragment according to this information.

6.5 Energy Decomposition Analysis

In BAND there are two fragment-based energy decomposition methods available: the periodic energy decomposition analysis (PEDA)[*56* (page 228)] and the periodic energy decomposition analysis combined with the natural orbitals of chemical valency method (PEDA-NOCV)[*56* (page 228)].

6.5.1 Periodic Energy Decomposition Analysis (PEDA)

```
PEDA [True | False]
```

PEDA

Type Bool

Default value False

Description If present in combination with the fragment block, the decomposition of the interaction energy between fragments is invoked.

If used in combination with the `fragment` keyblocks the decomposition of the interaction energy between fragments is invoked and the resulting energy terms (ΔE_{int}, ΔE_{disp}, ΔE_{Pauli}, ΔE_{elstat}, ΔE_{orb}) presented in the output file. (See the *example* (page 203) or the tutorial)

Attention: In case of the error message "Fragments cannot be assigned by a simple translation!", BAND does only allow for fragments which can be transformed to the structure in the PEDA calculation by a simple translation. So, a rotation is not allowed.

6.5.2 Periodic Energy Decomposition Analysis and natural orbitals of chemical valency (PEDA-NOCV)

`PEDANOCV (block-type)`

If present in combination with the `fragment` keyblocks and the `PEDA` key the decomposition of the orbital relaxation term is performed. The binary result file will contain the information to *plot NOCV Orbitals and NOCV deformation densities* (page 108). (See the *example* (page 206) or the tutorial)

```
PEDANOCV
   EigvalThresh float
   Enabled [True | False]
End
```

PEDANOCV

Type Block

Description If present in combination with the fragment blocks and the PEDA key, the decomposition of the orbital relaxation term is performed.

EigvalThresh

Type Float

Default value 0.001

Description The threshold controls that for all NOCV deformation densities with NOCV eigenvalues larger than EigvalThresh the energy contribution will be calculated and the respective pEDA-NOCV results will be printed in the output

Enabled

Type Bool

Default value False

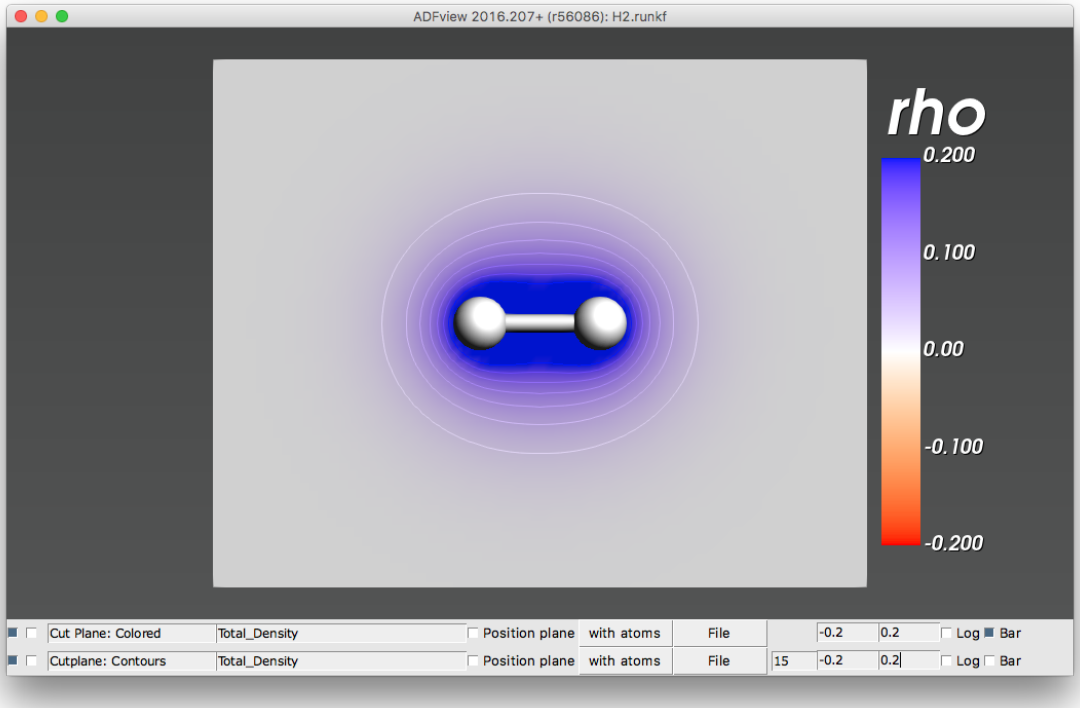
Description If true in combination with the fragment blocks and the PEDA key, the decomposition of the orbital relaxation term is performed.

6.6 3D field visualization with BAND

With ADFView you can visualize three-dimensional fields from the results of a BAND Calculation (**runkf** file).

Following is a list of relevant fields, with a short explanation and illustrative pictures (from a simple non-periodic H_2 calculation). All fields are in atomic units (a.u.) (https://en.wikipedia.org/wiki/Atomic_units).

Total_Density (rho) The electronic density $\rho(r)$. The integral of the electronic density over the whole space (or, for periodic systems, over the unit cell) equals the total number of electrons (valence + core).

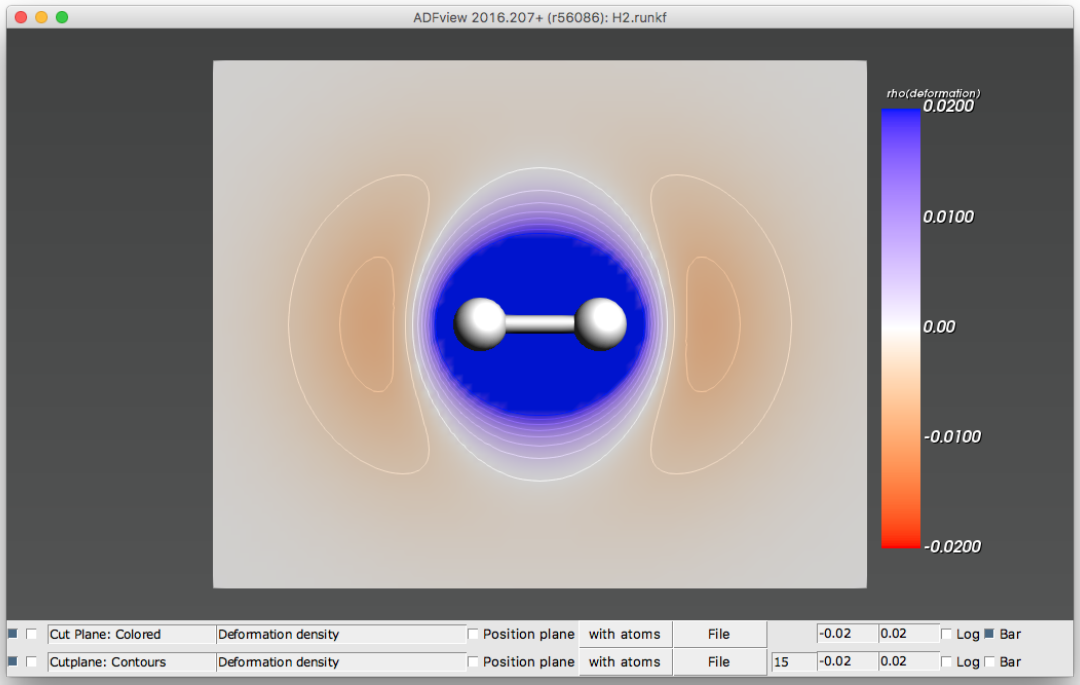


Deformation density (rho(deformation)) The deformation density is the difference between total density $\rho(r)$ and reference density $\rho_{\text{reference}}(r)$

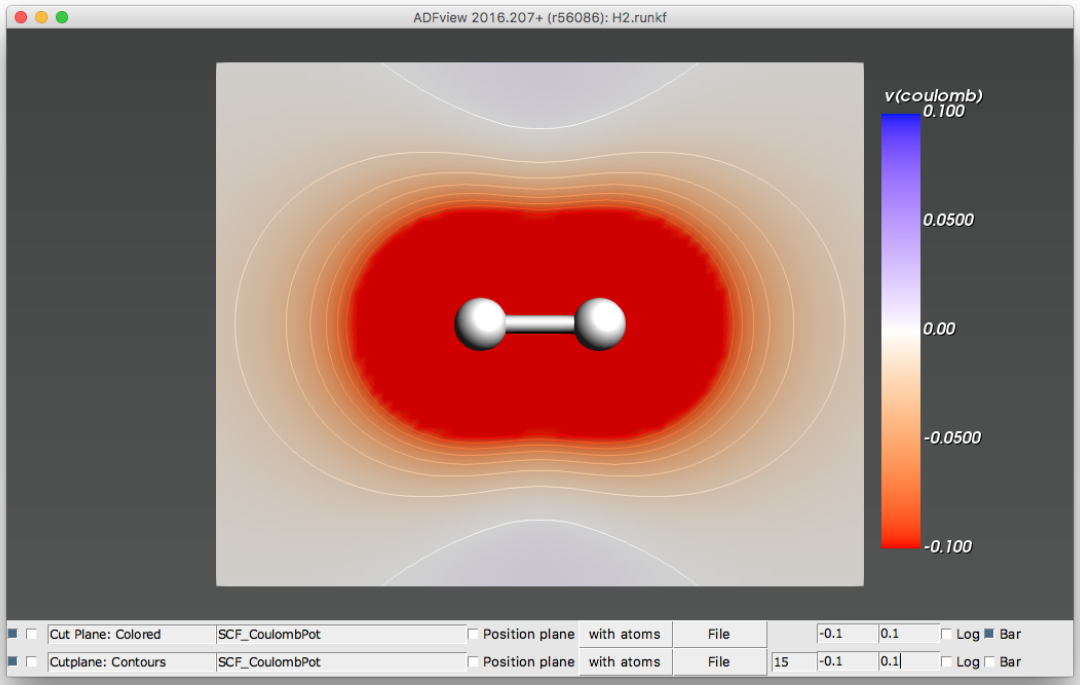
$$\rho_{\text{deformation}}(r) = \rho(r) - \rho_{\text{reference}}(r)$$

The reference density $\rho_{\text{reference}}(r)$ is defined as the sum of densities of spherical spin-unrestricted isolated atoms.

The deformation density is electrically neutral, i.e. its integral over the whole space (or, for periodic systems, over the unit cell) is zero. Positive values of deformation density indicate density accumulation wrt isolated atoms; negative values represent density depletion. In our H_2 example, the deformation density shows how there is electron accumulation in the bonding region between the two hydrogen atoms.

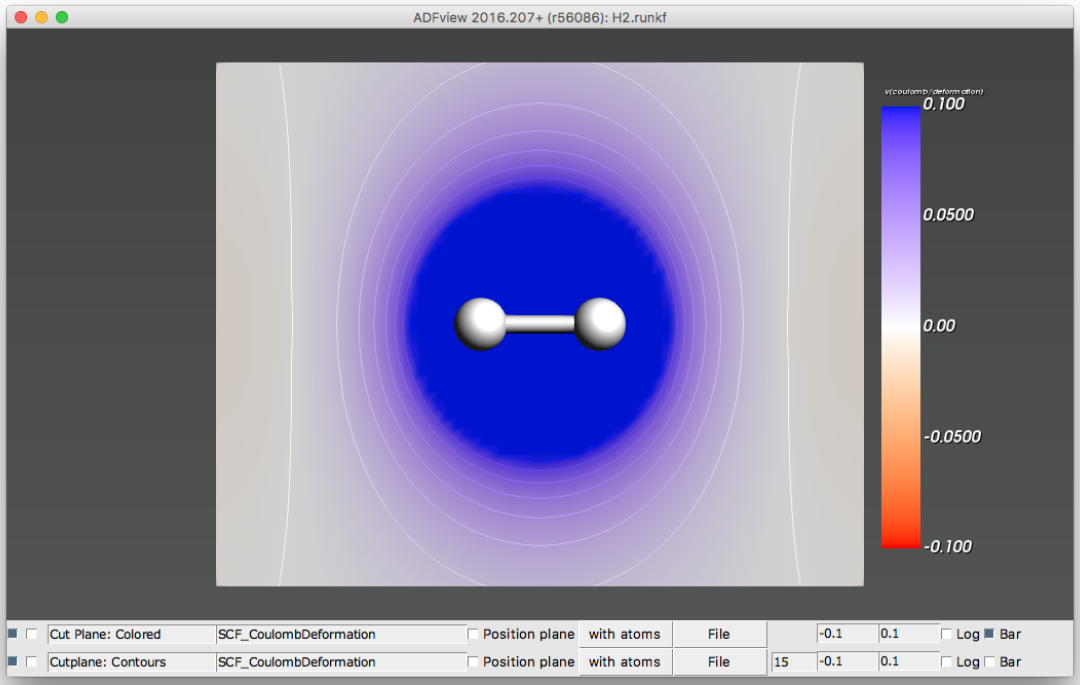


SCF_CoulombPot (v(coulomb)) The total Coulomb potential (nuclear + electronic potentials). BANDs convention for the Coulomb potential: the potential of positive charges (like nuclei) is **negative**, while the potential of negative charges (like electrons) is **positive**. In our example, the nuclear potential (negative) is larger than the electronic potential (positive) in the region of space near the H_2 molecule.

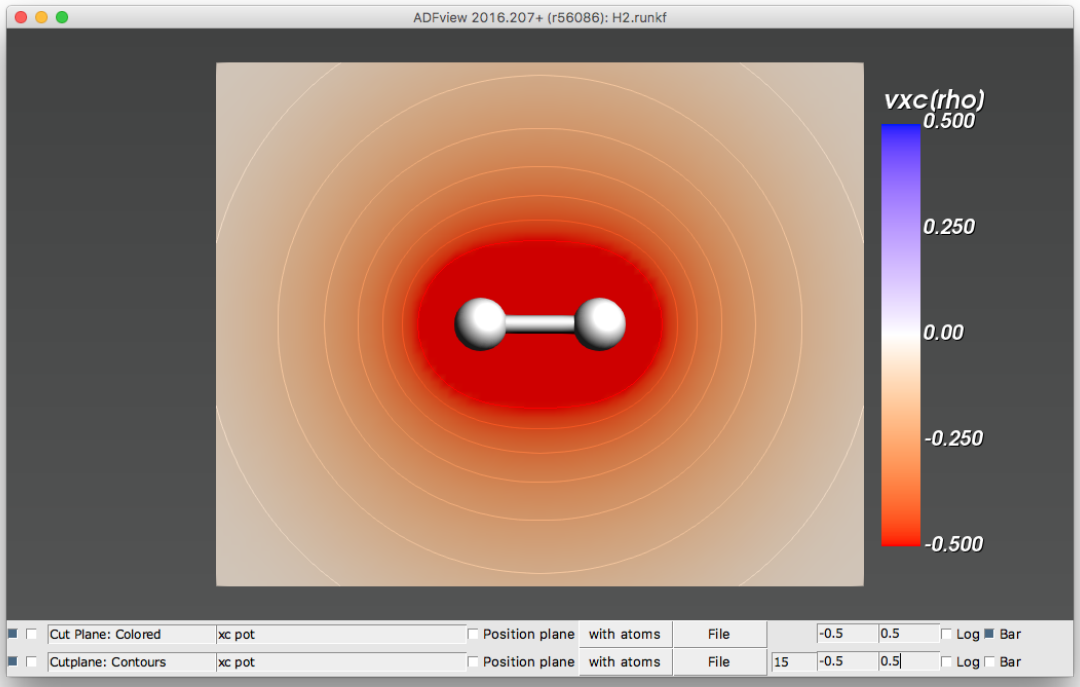


Note: The sign convention for potentials in **BAND** is the opposite to the **ADF** sign convention.

SCF_CoulombDeformation (v(coulomb/deformation)) The Coulomb potential originating from the (overall neutral) deformation density.



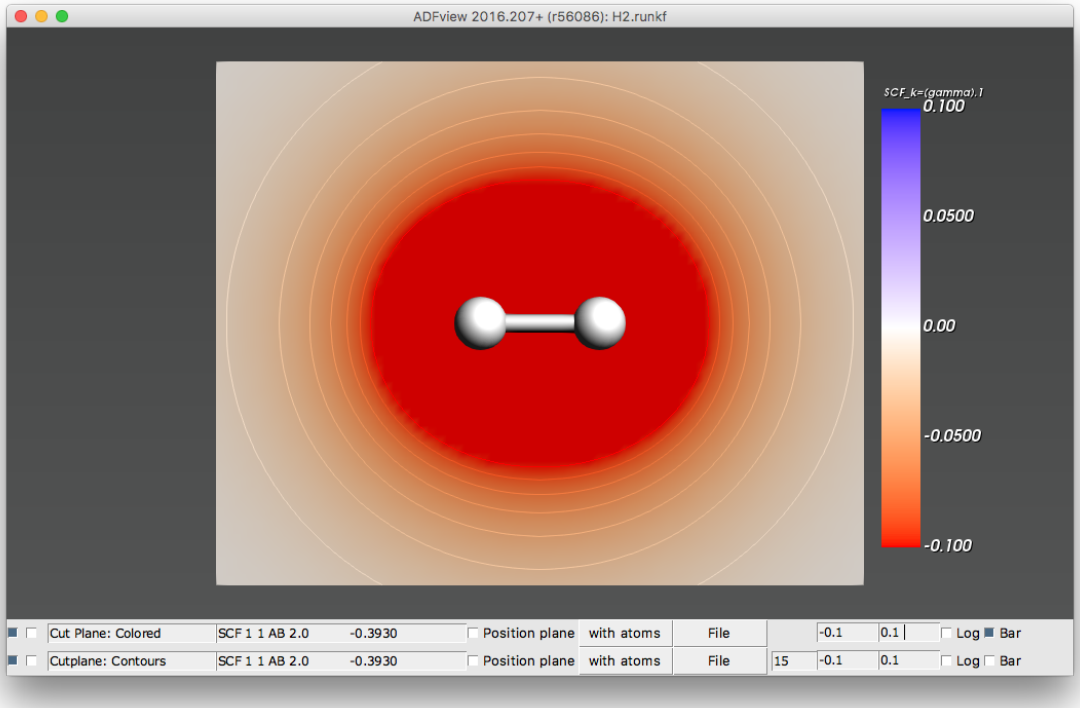
xc pot (vxc(rho)) The Exchange Correlation (XC) potential. Electrons are *attracted* by negative XC potentials (just like they are *attracted* by the negative nuclear Coulomb potential)

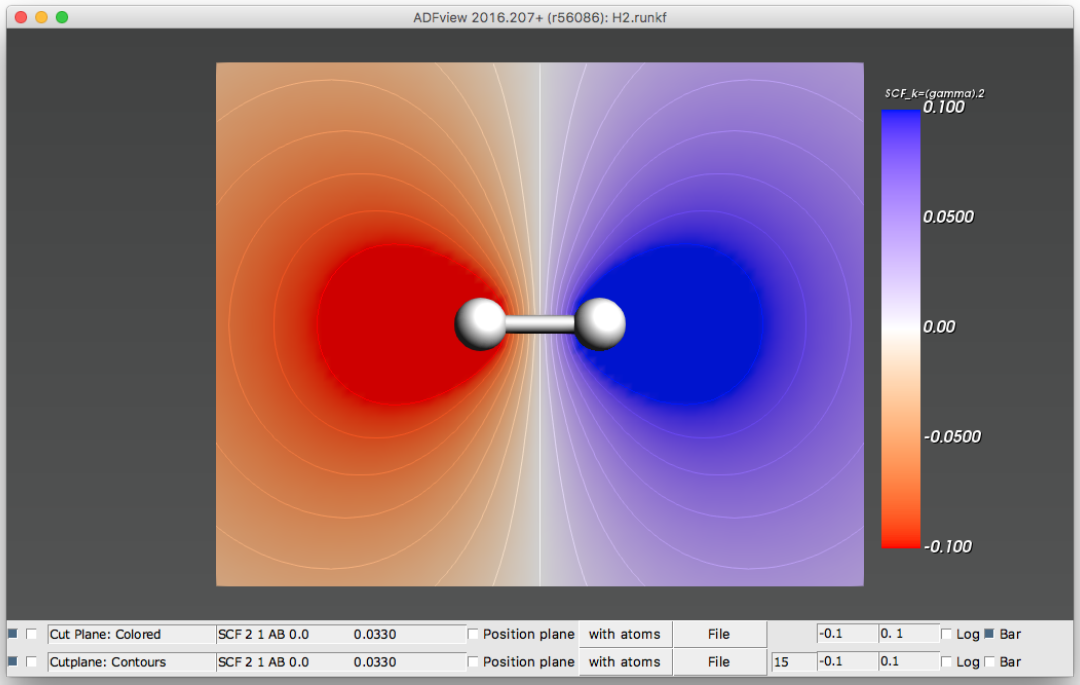


Orbitals (occupied/virtual) The Kohn-Sham orbitals.

Note: Be aware that there is an over-all arbitrariness in the sign of the orbitals

Here we show the occupied and first virtual orbital of H_2.




ADFview 2016.207+ (r56086): H2.runkf
SCF_k=(gamma).2
0.100
0.0500
0.00
-0.0500
-0.100
Cut Plane: Colored
SCF 2 1 AB 0.0
0.0330
Position plane
with atoms
File
-0.1
0. 1
Log
Bar
Cutplane: Contours
SCF 2 1 AB 0.0
0.0330
Position plane
with atoms
File
15
-0.1
0.1
Log
Bar

CHAPTER

SEVEN

ELECTRONIC TRANSPORT (NEGF)

See also:

BAND-NEGF GUI tutorial

Some examples are available in the `$ADFHOME/examples/band` directory and are discussed in the Examples section.

Example: Main NEGF flavors (page 154)
Example: NEGF with bias (page 161)

Note: In the BAND-GUI it is possible to choose between three NEGF methods (*flavors*):

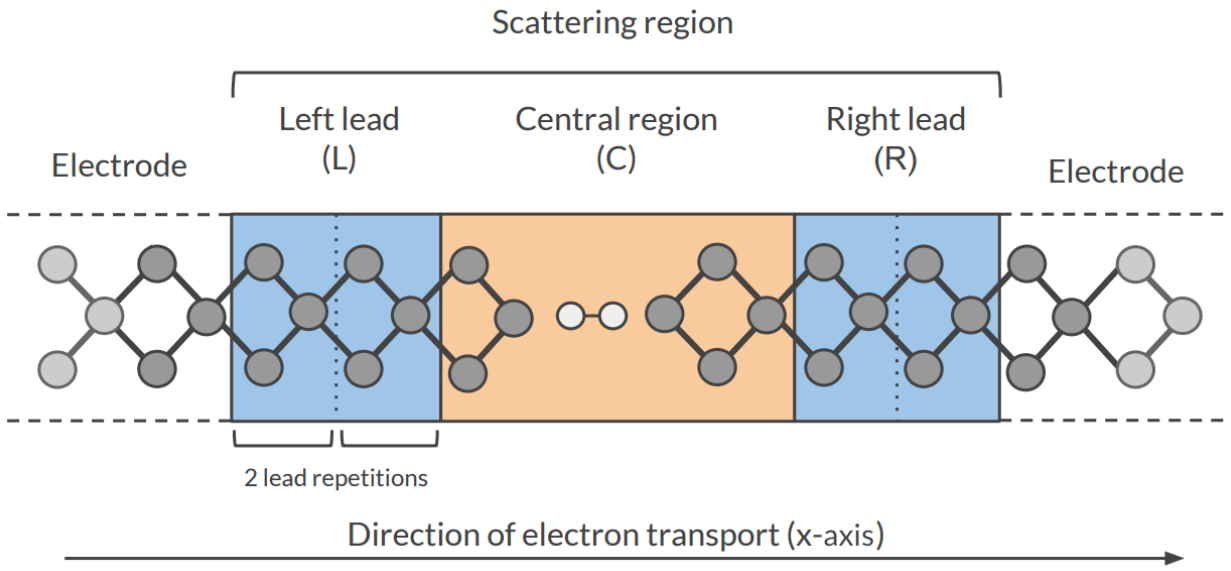
Self consistent This is the internal BAND-NEGF implementation, which is described in this page.

Self consistent + align This is the internal BAND-NEGF implementation with an extra alignment-run (workflow step 3a)

Non self consistent Computationally cheap method, equivalent to the DFTB-NEGF approach with H and S matrix elements computed by BAND (instead of DFTB).

7.1 Transport with NEGF in a nutshell

The **Non-Equilibrium Green's Functions** formalism (**NEGF**) is a theoretical framework for modeling electron transport through nano-scale devices. Electron transport is treated as a one-dimensional coherent scattering process in the "scattering region" for electrons coming in from the electrodes:



Our goal is to compute the **transmission function** $T(E)$, which describes the rate at which electrons of energy E are transferred from the left electrode to the right electrode by propagating through the scattering region. From the transmission function we can calculate the electric current for given **Bias Voltage** V applied between the electrodes:

$$I(V) = \frac{2e}{h} \int_{-\infty}^{\infty} T(E,V) \left(f(E-\mu_L) - f(E-\mu_R) \right) dE$$

where $f(E)$ is the Fermi-Dirac distribution function for a given temperature, and μ_L (μ_R) is $\epsilon_F + eV/2$ ($\epsilon_F - eV/2$), ϵ_F being the Fermi energy of the electrodes.

The transmission function $T(E)$ can be computed from the **Green's function** of our system.

The Green's function $G(E)$ of the scattering region is obtained solving the following equation:

$$(ES - H)G(E) = I$$

where S is the overlap matrix, H is the Hamiltonian and I is the identity matrix. The Hamiltonian is composed as follows (**L**, **C** and **R** denote the **left lead**, the **central region** and the **right lead** respectively):

$$H = \begin{pmatrix} H_L + \Sigma_L & H_{LC} & 0 \\ H_{LC} & H_C & H_{RC} \\ 0 & H_{RC} & H_R + \Sigma_R \end{pmatrix}$$

The two *self-energies* Σ_L and Σ_R model the two semi-infinite electrodes.

The transmission function $T(E)$ can be calculated from the Green's function $G(E)$ and the so-called *broadening matrices* $\Gamma_L(E)$ and $\Gamma_R(E)$:

$$T(E) = Tr[G(E)\Gamma_R(E)G(E)\Gamma_L(E)]$$

The broadening matrix being

$$\Gamma_L(E) = -2\Im\Sigma_L(E)$$

7.1.1 Self consistency

The density matrix is determined self consistently [*68* (page 229)]:

$$P_{\text{in}} \rightarrow H_{KS} \xrightarrow{\text{shifts}} H_{\text{aligned}} + \Sigma_L(E) + \Sigma_R(E) \rightarrow G(E) \xrightarrow{\int de} P_{\text{out}}$$

From a guess of the density matrix the corresponding KS Hamiltonian is calculated. This Hamiltonian is aligned, and then the NEGF Hamiltonian in the complex plane is constructed by adding the self energies, representing the influence of the electrodes. From the resulting Green's function a new density matrix follows.

From the difference between input and output density a next input is guessed. This is repeated until the input and output densities converge.

For the alignment of the Hamiltonian there are two shifts. The first shift aligns the potential in the leads to the electrodes.

$$\text{shift } 1 = \frac{1}{n} \sum_{\text{i in lead}}^{n} \frac{H_{ii}^{TB} - H_{ii}^{KS}}{S_{ii}}$$

The second and usually smaller shift results from the alignment run. A shift Δ is applied globally

$$H_{ij}^{\text{aligned}} = H_{ij} + \Delta S_{ij}$$

7.1.2 Contour integral

Without bias the density matrix follows from

$$P(\mu) = -\frac{1}{\pi} \int_{-\infty}^{\infty} de\ f(e,\mu)\ \Im G(e)$$

As the Green's function is singular on the real axis we add a small imaginary value (**eta**) to the energy. Still, the integrand will be very wild function, and it is numerically better to do a contour integral instead.

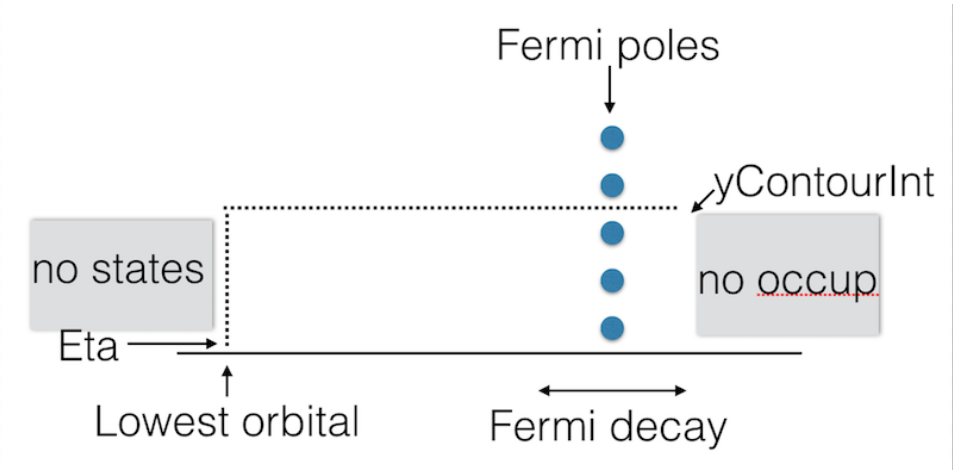
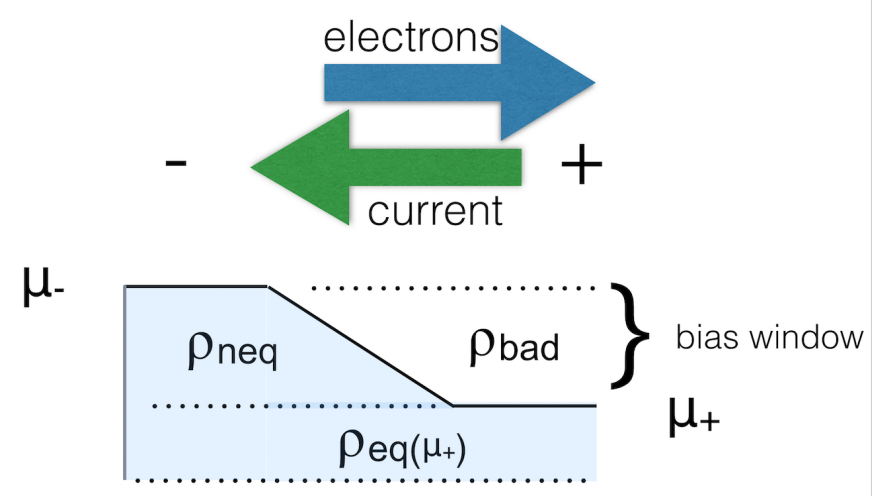


Fig. 7.1: Figure: BAND uses a rectangular contour in the complex energy plane to integrate the (integrand of the) density matrix. The integrand also needs to be evaluated in the enclosed FD poles (three in this picture).

7.1.3 Gate potential

There is no direct key for the gate potential. You can model this with the *FuzzyPotential* (page 30) key. Setting up the gate potential for NEGF is most conveniently done with the GUI.



7.1.4 Bias potential

When there is a bias specified there are two important things to keep in mind.

First of all you need to define a ramp potential. In the negative lead this should have the value +V/2 and in the negative lead -V/2. The ramp should smoothly go from one to the other value. For metals one could start the ramp at the surface atoms of the lead material. For semi-conductors it is less clear. The ramp potential can be specified with the *FuzzyPotential* (page 30) key. The GUI can be helpful here.

Secondly, the expression for the density is different from the zero-bias case:

$$\rho = \rho_{\mathrm{eq}}(\mu_+) + \rho_{\mathrm{neq}}$$

The first (equilibrium) term is calculated with a contour integral as before, the second (non-equilibrium) part cannot be calculated with a contour integral. Instead, an integral in the complex plane (close to the real axis) is performed, the range covering the bias energy window.

See also:

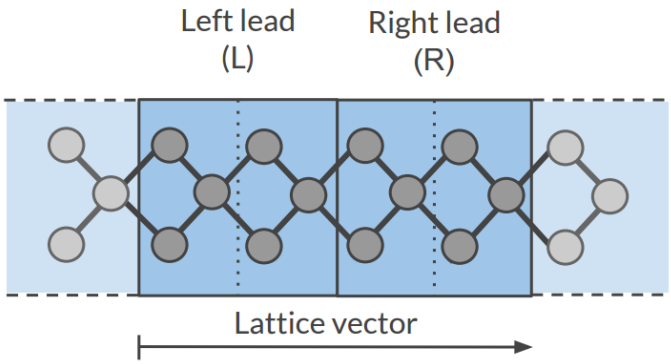
PhD Thesis (https://www.scm.com/wp-content/uploads/Verzijl2012.pdf) of C. Verzijl (BAND-NEGF developer)

7.2 Workflow

The computation of the transmission function $T(E)$ within the BAND-NEGF [*68* (page 229)] formalisms requires three or four individual simulations.

Tip: Use ADFInput (GUI) to set up your BAND-NEGF calculation (see the BAND-NEGF GUI tutorial)

1): Lead calculation A 1D-periodic BAND calculation of the lead (including *StoreHamiltonian2* (page 101)):



A tight binding (TB) representation is calculated for the overlap ($S(R=0)$ and $S(R=a)$) and Fock matrix ($H(R=0)$ and $H(R=a)$). This is not an approximation provided that the functions do not extend beyond the neighboring cells. You should choose a sufficiently large super cell for this to be true. For this reason we recommend setting the *SoftConfinement* (page 36) Quality to Basic, thus reducing the range of the functions.

2): SGF calculation A small program that determines the fermi energy ϵ_F corresponding to the TB representation, and the specified temperature. This fermi energy is typically a bit higher than the one from the lead calculation. This also tests the contour integration.

3a): Alignment run (optional) The idea is to fill the central region with bulk material. Then one expects to have zero charge in the central region. In practice this is not exactly true. In the alignment run the shift is determined that makes the central region neutral. This global shift is to be used in the next run.

3b): Transport calculation Computes the NEGF transmission function $T(E)$. The density matrix is determined fully self-consistently. Without alignment (3a) one should set `NEGF%ApplyShift2` to `False`.

To get the current as a function of bias potential you need to repeat calculation 3b for a various bias potentials.

7.3 Input options

7.3.1 SGF Input options

SGF is a small separate program. An input looks like:

```
$ADFBIN/sgf   << eor
TITLE Test for NEGF inputs
SAVE SIGMA
SURFACEGF
   SCMCode True
   KT 0.001
   ContourQuality normal
END
eor
```

It looks for a file `RUNKF` and the output is a file named `SigmaSCM`. The only important parameter is KT which is the Boltzmann constant times the temperature in Hartree. The other parameter of interest is the ContourQuality, which can be set to Basic,Normal,Good,VeryGood, or Excellent.

7.3.2 NEGF Input options (no bias)

The NEGF functionality is controlled by the NEGF block key.

```
NEGF
   LeadFile string
   SGFFile string
   ContourQuality [basic | normal | good | verygood]
   EMin float
   EMax float
   NE integer
End
```

NEGF

Type Block

Description Options for the NEGF (non-equilibrium green function) transport calculation.

LeadFile

Type String

Default value

Description File containing the tight binding representation of the lead.

SGFFile

Type String

Default value

Description The result from the SGF program. Contains the Fermi energy of the lead.

ContourQuality

Type Multiple Choice

Default value good

Options [basic, normal, good, verygood]

Description The density matrix is calculated numerically via a contour integral. Changing the quality influences the number of points. This influences a lot the performance.

EMin

Type Float

Default value -5.0

Unit eV

Description The minimum energy for the transmission grid (with respect to the Fermi level of the lead)

EMax

Type Float

Default value 5.0

Unit eV

Description The maximum energy for the transmission grid (with respect to the Fermi level of the lead)

NE

Type Integer

Default value 100

Description The number of energies for the transmission energy grid.

The following are expert / technical options:

```
NEGF
   CheckOverlapTol float
   Eta float
   ApplyShift1 [True | False]
   ApplyShift2 [True | False]
   YContourInt float
   DEContourInt float
End
```

NEGF

Type Block

Description Options for the NEGF (non-equilibrium green function) transport calculation.

CheckOverlapTol

Type Float

Default value 0.01

Description BAND checks how well the TB overlap matrix S(R=0) represents the overlap matrix in the lead region. Elements corresponding to the outer layer are neglected, because when using a frozen core they have bigger errors.

Eta

Type Float

Default value 1e-05

Description Small value used for the contour integral: stay at least this much above the real axis. This value is also used for the evaluation of the Transmission and dos.

ApplyShift1

Type Bool

Default value True

Description Apply the main shift, obtained from comparing matrix elements in the leads with those from the tight-binding run. Strongly recommended.

ApplyShift2

Type Bool

Default value True

Description Apply the smaller alignment shift. This requires an extra alignment run. Usually this shift is smaller.

YContourInt

Type Float

Default value 0.3

Description The density is calculated via a contour integral. This value specifies how far above the real axis the (horizontal part of the) contour runs. The value is rounded in such a way that it goes exactly halfway between two Fermi poles. There is a trade off: making it bigger

makes the integrand more smooth, but the number of enclosed poles increases. For low temperatures it makes sense to lower this value, and use a smaller deContourInt.

DEContourInt

Type Float

Default value -1.0

Description The energy interval for the contour grid. Defaults depends on the contour quality

7.3.3 NEGF Input options (with bias)

With a bias potential there are some extra keys.

```
NEGF
   BiasPotential float
   NonEqDensityMethod integer
   BoundOccupationMethod integer
   YRealaxisInt float
   DERealAxisInt float
End
```

NEGF

Type Block

Description Options for the NEGF (non-equilibrium green function) transport calculation.

BiasPotential

Type Float

Default value 0.0

Description Apply a bias potential (atomic units). Can be negative. One has to specify the ramp potential with the FuzzyPotential key. This is mostly conveniently done with the GUI.

NonEqDensityMethod

Type Integer

Default value 1

Description See text.

BoundOccupationMethod

Type Integer

Default value 1

Description See text. Only relevant with NonEqDensityMethod equal 2 or 3.

YRealaxisInt

Type Float

Default value 1e-05

Description The non-Equilibrium density is calculated near the real axis.

DERealAxisInt

Type Float

Default value -1.0

Description The energy interval for the real axis grid. Defaults depends on the contour quality.

NonEqDensityMethod Let us introduce some terms [66 (page 228)]. First of all the total density in the bias window (ignoring occupation)

$$D = \frac{1}{2\pi} \int A \, (f_- - f_+)$$

And then there are the side resolved densities

$$D_{+/-} = \frac{1}{2\pi} \int A_{+/-} \, (f_- - f_+)$$

The issue here is that the side resolved densities do not sum to the total one

$$D = D_+ + D_- + D_{\text{bound states}}$$

The NonEqDensityMethod is about how these integrals are calculated . With option 1, or 2 a contour integral is used for D: they are essentially the same. However, when choosing option 2, you can choose a BoundOccupationMethod, leading to other physics. If set to 3, the total density in the bias window (D) will be calculated near the real axis: this way one avoids the possibility of a negative nr. of bound states (deviating from [66 (page 228)]).

BoundOccupationMethod Only relevant with NonEqDensityMethod equal 2 or 3. If set to one, the density of bound states (ignoring occupation) is simply multiplied by a half. If set to two, atoms closer to the negative lead will get a higher occupation [66 (page 228)]. Atoms coupled to the right lead will have a low occupation. For this we recommend setting NonEqDensityMethod to 3, to avoid a possible negative number of bound states.

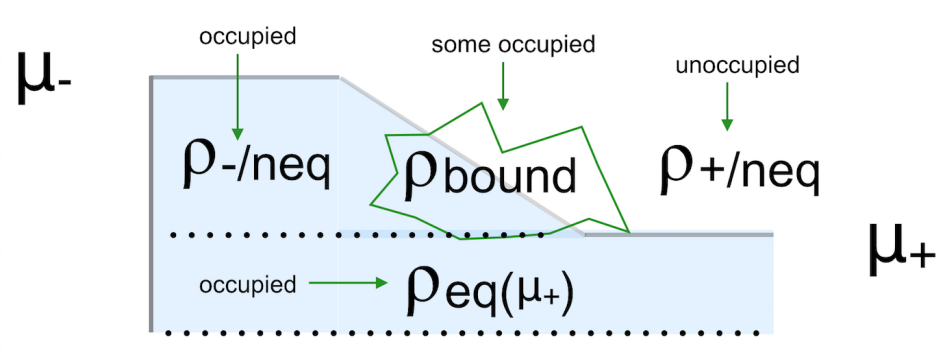


Fig. 7.2: Figure: The non-equilibrium density consists of three parts: the left and right parts (ρ-/neq and ρ+/neq) and the bound states (ρbound). We want to know the occupied part.

Setting the method BoundOccupationMethod to 1, leads to

$$\rho = \rho(\mu_+) + \rho_{\text{-/neq}} + \frac{1}{2}\rho_{\text{bound}}$$

By setting the method to 2, each atom gets its own weight in the density matrix

$$\rho_{ij} = \rho_{ij}(\mu_+) + \rho_{\text{-/neq}} + \sqrt{w_i w_j}\rho_{ij}^{\text{bound}}$$

with [66 (page 228)]

$$w_i = \frac{\text{Tr}[D_-]_i}{\text{Tr}[D_-]_i + \text{Tr}[D_+]_i}$$

These weights are the same for all functions on an atom. The intended effect is: bound states that are coupled more strongly to the negative electrode get a higher occupation than the ones that are coupled more strongly to the positive electrode.

To summarize here are three reasonable settings

NonEqDensityMethod	BoundOccupationMethod	intention
1	1	Multiply the bound states with a half
2	2	Occupy bound states with atom-resolved w_i
3	2	... and prevent a negative nr. of bound states

To get the current from a calculation you can use adfprep:

```
$ADFBIN/adfreport RUNKF 'NEGF%current'
```

7.3.4 NEGF Input options (alignment)

For the (optional) alignment run there are some extra keys.

```
NEGF
   DoAlignment [True | False]
   Alpha float
   AlignChargeTol float
   CDIIS [True | False]
End
```

NEGF

Type Block

Description Options for the NEGF (non-equilibrium green function) transport calculation.

DoAlignment

Type Bool

Default value False

Description Set this to True if you want to do an align run. Between the leads there should be lead material. The GUI can be of help here.

Alpha

Type Float

Default value 1e-05

Description A charge error needs to be translated in a potential shift. DeltaV = alpha * DeltaQ

AlignChargeTol

Type Float

Default value 0.1

Description In an alignment run you want to get the number of electrons in the center right. This number specifies the criterion for that.

CDIIS

Type Bool

Default value False

Description Make the normal DIIS procedure aware of the align charge error

7.4 Troubleshooting

The self consistent approach, unique to BAND, may be difficult to converge. If this is true for the alignment run it can be decided to skip this run. For the final transport run, here are some tips / considerations.

- Use a SZ basis for the metal atoms
- Restart (the density matrix) from the result of a smaller (such as the SZ) basis. (See "*Save* (page 110) Density-Matrix" and the *Restart* (page 103) key)
- Restart (the density matrix) from the result obtained with a smaller bias (only relevant for calculations with bias potential).
- Setting NEGF%BoundOccupationMethod to 2 (and NEGF%NonEqDensityMethod to 3) might help. Note that this affects the physics: you are differently occupying the bound states.
- Use a better NEGF%ContourQuality (there comes a computational price tag with this).

If everything fails it is possible to use BAND in a **non-self consistent way**, similar to the way DFTB-NEGF works. This option is available via the GUI.

7.5 Miscellaneous remarks on BAND-NEGF

- You should make sure that your results are converged with respect to the number of lead repetitions; the results should not change significantly if you increase the number of lead repetitions.
- It's good practice to include at least one lead repetition in the central region.

7.5.1 Store tight-binding Hamiltonian

Let us consider a Fourier transformation of a 1D Bloch matrix

$$S(R = na) = \int_k e^{-ikR} S(k)$$

In (the tight-binding) case that the functions do not extend beyond the neighboring cells only S(R=0) and S(R=a) are nonzero. (And S(R=-a) is equivalent to S(R=a))

```
StoreHamiltonian2 [True | False]
```

StoreHamiltonian2

Type Bool

Default value False

Description determine the tight-binding representation of the overlap an fock matrix. Used for (at least) NEGF.

Adding StoreHamiltonian2 to the input cause band to determine the tight-binding representation of the overlap an fock matrix. Currently this only works for 1D periodic systems. For the overlap matrix you will get two parts. The first $S(R = 0)$ is the (symmetric) overlap matrix of atoms in the unit cell. The second $S(R = a)$ is a non symmetric matrix describing the coupling of functions in the central cell with functions in its right neighboring cell. On the RUNKF file you will find the TB representations of the overlap and Hamiltonian stored in the 'Matrices' section as "S(R)" and "H(R)", being dimensioned (nBas,nBas,2).

CHAPTER

EIGHT

EXPERT OPTIONS

8.1 Restarts

The main results of a BAND calculation are stored in the rkf file. If you save this file you can use it to restart your calculation. The input for the restart calculation is essentially the same, except for some extra keys, like `Restart`, `Grid`, and `DensityPlot`.

Plots of the density (and many other symmetric properties) can can be obtained with the key `DensityPlot`. Density and orbital plot restarts require the specification of the `Grid` key. With the subkey `SCF` you can start the SCF procedure with the last solution from the restart file. This can be useful if the SCF did not converge or if you want to compute some post-SCF properties (e.g. the *DOS* (page 73) or the *band structure* (page 76)). Similarly, a geometry optimization can be restarted with the subkey `GeometryOptimization` You can use the geometry of a previous calculation.

Usually the input for a restarted job is the same as for the original calculation, with some extra options, described in this section.

Some examples are available in the `$ADFHOME/examples/band` directory and are discussed in the Examples section.

Example: Restart SCF for properties calculation (page 150)
Example: Restart the SCF (page 146)
Example: Properties on a grid (page 152)

8.1.1 Restart key

```
Restart
   File string
   SCF [True | False]
   DensityPlot [True | False]
   OrbitalPlot [True | False]
   NOCVdRhoPlot [True | False]
   NOCVOrbitalPlot [True | False]
   UseDensityMatrix [True | False]
End
```

`Restart`

Type Block

Description Tells the program that it should restart with the restart file, and what to restart.

`File`

Type String

Default value

Description Name of the restart file.

SCF

Type Bool

Default value False

Description Restart the SCF procedure.

DensityPlot

Type Bool

Default value False

Description Goes together with the DensityPlot block and Grid blocks

OrbitalPlot

Type Bool

Default value False

Description Goes together with the OrbitalPlot and Grid

NOCVdRhoPlot

Type Bool

Default value False

Description Goes together with the NOCVdRhoPlot and Grid blocks.

NOCVOrbitalPlot

Type Bool

Default value False

Description Goes together with the NOCVOrbitalPlot and Grid blocks.

UseDensityMatrix

Type Bool

Default value False

Description If set to True: For restarting the SCF the density matrix will be used. Requires you to set 'Save DensityMatrix' in the previous run.

8.1.2 Grid

The Grid block is used for restart options `OrbitalPlot`, `DensityPlot`, `NOCVOrbitalPlot` and `NOCVdRhoPlot`. There are two ways to define your grid. The most easy way is to use the Type key, which automatically generates a grid around the atoms in the unit cell:

```
Grid
   Type [coarse | medium | fine]
End
```

Grid

Type Block

Description Options for the regular grid used for plotting (e.g. density plot). Used ICW the restart option.

Type

Type Multiple Choice

Default value coarse

Options [coarse, medium, fine]

Description The default regular grids.

One alternative is to specify everything by hand via the 'UserDefined' sub-block.

```
Grid
   UserDefined # Non-standard block. See details.
      ...
   End
End
```

Grid

Type Block

Description Options for the regular grid used for plotting (e.g. density plot). Used ICW the restart option.

UserDefined

Type Non-standard block

Description Once can define the regular grid specification in this block. See example.

The following input would create a cube from (-1,-1,-1) to (1,1,1):

```
Grid
  UserDefined
     -1 -1 -1 ! Starting point
     1 0 0 0.1   ! vec1 and dvec1
     0 1 0 0.1   ! vec2 and dvec2
     0 0 1 0.1   ! vec3 and dvec3
     20 20 20    ! nr. of steps along three directions
  End
End
```

One can also specify a text file from which the grid is imported:

```
Grid
   FileName string
End
```

Grid

Type Block

Description Options for the regular grid used for plotting (e.g. density plot). Used ICW the restart option.

FileName

Type String

Default value

Description Read in the grid from a file. The file format of the grid is: three numbers per line (defining the x, y and z coordinates of the points).

8.1.3 Plots of the density, potential, and many more properties

```
DensityPlot # Non-standard block. See details.
   ...
End
```

DensityPlot

Type Non-standard block

Description Plots of the density. Goes together with the Restart%DensityPlot and Grid keys.

The DensityPlot block goes together with the `Restart%DensityPlot` and `Grid` keys. Example input:

```
...
Restart
   File my_file.rkf
   DensityPlot
End

Grid
   Type Coarse
End

DensityPlot
   rho(fit)
   vxc[rho]
End
...
```

After such a run you get a TAPE41 file that you should rename to my.t41, and view with ADFview. The most common properties to plot are:

- `rho(fit)` The fitted density.
- `v(coulomb)` The Coulomb potential.
- `vxc[rho(fit)]` the XC potential (using the fitted density)
- `vxc[rho]` XC potential of the exact density
- `rho` The density
- `|gradRho|` The norm of the gradient of the density
- `tau` The symmetric kinetic energy density
- `LDOS` The local density of states. (See *LDOS key* (page 109))
- `elf[rho]` The electron localization function

Some more specialized options are:

- `rho(deformation/fit)` the fitted deformation density
- `rho(atoms)` The density of the startup atoms
- `v(coulomb/atoms)` The Coulomb potential of the start density
- `s[rho]` Reduced density gradient. Common ingredient for XC functionals

- `s[rho(fit)]` Same as above, now for the fit density
- `alpha[rho]` Ingredient for some meta-GGAs

In the BAND example directory there is the *Frags_COCu* (page 192) example which shows how this can be used in combination with the `Fragment` key.

8.1.4 Orbital plots

```
OrbitalPlot # Non-standard block. See details.
   ...
End
```

`OrbitalPlot`

Type Non-standard block

Description Goes together with the Restart%OrbitalPlot and Grid keys. See Example.

The OrbitalPlot block goes together with the `Restart%OrbitalPlot` and `Grid` keys. Example input:

```
...
Restart
   File my_file.rkf
   OrbitalPlot
End

Grid
   Type Coarse
End

OrbitalPlot
   1 Band 5 8  ! for k-point 1 plot bands 5 to 8
   5 Band 6    ! for k-point 5 plot band 6
   6 -0.2 +0.3 ! for k-point 6 plot bands between -0.2 and +0.3 a.u. w.r.t Fermi level
End
...
```

After such a run you get a TAPE41 file that you should rename to my.t41, and view with ADFview.

8.1.5 Induced Density Plots of Response Calculations

```
ResponseInducedDensityPlot # Non-standard block. See details.
   ...
End
```

`ResponseInducedDensityPlot`

Type Non-standard block

Description Goes together with Restart%ResponseInducedDensityPlot and Grid.

`ResponseInducedDensityPlot (block-type)` The ResponseInducedDensityPlot block goes together with the `Restart%ResponseInducedDensityPlot` and `Grid` keys. In the BAND example directory there is the *TD-CDFT for MoS2 Monolayer* (page 179) example that shows how this can be used. Example input:

```
...
Restart
   File my_file.rkf
   ResponseInducedDensityPlot
End

Grid
   Type Coarse
End

ResponseInducedDensityPlot
   XCOMPONENT 5 8  ! plot x component of induced densities
                   ! for frequencies number 5 to 8
   YCOMPONENT 6    ! plot y component of induced densities
                   ! for frequency number 6
   ZCOMPONENT 1    ! plot z component of induced densities
                   ! for frequency number 1
End
...
```

After such a run you get a TAPE41 file that you should rename to my.t41, and view with ADFview.

Attention: The plotting capability works only with response calculation RUNKF files based on the *NewResponse* (page 61) method!

8.1.6 NOCV Orbital Plots

```
NOCVOrbitalPlot # Non-standard block. See details.
   ...
End
```

NOCVOrbitalPlot

Type Non-standard block

Description Goes together with the Restart%NOCVOrbitalPlot and Grid keys. See example.

The NOCVOrbitalPlot blockg oes together with the `Restart%NOCVOrbitalPlot` and `Grid` keys. See example *PEDANOCV_MgO+CO* (page 206). Example input:

```
...
Restart
   File my_file.rkf
   NOCVOrbitalPlot
End

Grid
   Type Coarse
End

NOCVOrbitalPlot
   1 Band 5 8 ! for k-point 1 plot NOCV Orbitals 5 to 8
End
...
```

After such a run you get a TAPE41 file that you should rename to my.t41, and view with ADFview.

8.1.7 NOCV Deformation Density Plots

```
NOCVdRhoPlot # Non-standard block. See details.
   ...
End
```

NOCVdRhoPlot

Type Non-standard block

Description Goes together with the Restart%NOCVdRhoPlot and Grid keys. See example.

The NOCVdRhoPlot blockg oes together with the `Restart%NOCVdRhoPlot` and `Grid` keys. See example *PEDANOCV_MgO+CO* (page 206). Example input:

```
...
Restart
   File my_file.rkf
   NOCVdRhoPlot
End

Grid
   Type Coarse
End

NOCVdRhoPlot
   1 Band 5 8 ! for k-point 1 plot NOCV deformation densities 5 to 8
End
...
```

After such a run you get a TAPE41 file that you should rename to my.t41, and view with ADFview.

8.1.8 LDOS (STM)

```
LDOS
   DeltaNeg float
   DeltaPos float
   Shift float
End
```

LDOS

Type Block

Description Local Density-Of-States information. This can be used to generate STM images in the Tersoff-Hamann approximation (see https://doi.org/10.1103/PhysRevB.31.805)

DeltaNeg

Type Float

Default value 0.0001

Unit Hartree

Description Lower bound energy (Shift-DeltaNeg)

DeltaPos

Type Float

Default value 0.0001

Unit Hartree

Description Upper bound energy (Shift+DeltaPos)

Shift

Type Float

Default value 0.0

Unit Hartree

Description The energy bias with respect to the Fermi level.

The local density of states is integrated over the resulting interval. Example of an LDOS restart:

```
Restart
   File my_file.rkf
   DensityPlot
End

Grid
   Type Coarse
End

DensityPlot
   LDOS
   Shift      0.1
   DeltaNeg .001
   DeltaPos 0
End
```

According to this example, we restart from the result file of a previous calculation. The calculation will generate a file TAPE41 which can be viewed with ADFview. (Rename the file to my.t41)

See also *Restart* (page 103), and *DensityPlot* (page 106).

8.1.9 Save

```
Save string
```

Save

Type String

Recurring True

Description Save scratch files or extra data that would be otherwise deleted at the end of the calculation. e.g. 'TAPE10' (containing the integration grid) or 'DensityMatrix'

8.2 Symmetry

The symmetry of the system is automatically detected. Normally the symmetry of the initial system is maintained. One can lower the symmetry with the `Symmetry` key. In such cases the keyword `POTENTIALNOISE` can force the

solution away from the initial symmetry.

Whether or not symmetry should be used can be controlled vial the `UseSymmetry` key

```
UseSymmetry [True | False]
```

`UseSymmetry`

Type Bool

Default value True

Description Whether or not to exploit symmetry during the calculation.

One can also select a sub set of symmetry operators:

```
SubSymmetry integer_list
```

`SubSymmetry`

Type Integer List

Description The indices of the symmetry operators to maintain.

To get the indices of the symmetry operators, you should first run the calculation with the following options added to your input:

```
 print symmetry
 stopafter gemtry

and then you look in the output for (here the first four operators are listed)


::

    64    SYMMETRY OPERATORS:

 NO               MATRIX               TRANSL        AXIS    DET   ROTATION
 ------------------------------------------------------------------------------

 1)         1.000   0.000   0.000       0.000       0.000    1.0       1
            0.000   1.000   0.000       0.000       0.000
            0.000   0.000   1.000       0.000       1.000

 2)         1.000   0.000   0.000       0.000       0.000    1.0       1
            0.000   1.000   0.000       5.400       0.000
            0.000   0.000   1.000       0.000       1.000

 3)         1.000   0.000   0.000       5.400       0.000    1.0       1
            0.000   1.000   0.000       0.000       0.000
            0.000   0.000   1.000       0.000       1.000

 4)         1.000   0.000   0.000       5.400       0.000    1.0       1
            0.000   1.000   0.000       5.400       0.000
            0.000   0.000   1.000       0.000       1.000
```

from this list you should select the desired operators and use that in your final calculation, for example:

```
SubSymmetry 1 7 21 31
```

8.2.1 Symmetry breaking for SCF

```
PotentialNoise float
```

PotentialNoise

Type Float

Default value 0.0001

Description The initial potential for the SCF procedure is constructed from a sum-of-atoms density. Added to this is some small noise in the numerical values of the potential in the points of the integration grid. The purpose of the noise is to help the program break the initial symmetry, if that would lower the energy, by effectively inducing small differences between (initially) degenerate orbitals.

8.3 Advanced Occupation Options

By default the levels are occupied according to the aufbau principle. In some cases it is possible to create holes below the Fermi level or uneven occupation for alpha and beta electrons with the `Occupations` (Γ-only) and alternatively the `EnforcedSpinPolarization` (for an arbitrary number of k-points) key.

```
Occupations # Non-standard block. See details.
   ...
End
```

Occupations

Type Non-standard block

Description Allows one to input specific occupations numbers. Applies only for calculations that use only one k-point (i.e. pseudo-molecule calculations). See example.

Example:

```
OCCUPATIONS
   1 occupations_alpha {// occupations_beta}
End
```

- `occupations_beta` and the separating double slash (//) must not be used in a spin-restricted calculation.
- `occupations_alpha/beta` is a sequence of values assigned to the states ('bands') in energy ordering.

```
ElectronHole
   BandIndex integer
   SpinIndex integer
End
```

ElectronHole

Type Block

Description Allows one to specify an occupied band which shall be depopulated, where the electrons are then moved to the Fermi level. For a spin-restricted calculation 2 electrons are shifted and for a spin-unrestricted calculation only one electron is shifted.

BandIndex

Type Integer

Description Which occupied band shall be depopulated.

SpinIndex

Type Integer

Description Defines the spin of the shifted electron (1 or 2).

See the example *Si_ElectronHole* (page 218)

CHAPTER

NINE

TROUBLESHOOTING

9.1 Recommendations

9.1.1 Model Hamiltonian

Relativistic model

By default we do not use relativistic effects. The best approximation is to use spin-orbit coupling, however that is computationally very expensive. The scalar relativistic option comes for free, and for light elements will give very similar results as non-relativistic theory, and for heavy ones better results w. r. t. experiment. We recommend to always use this (scalar ZORA). To go beyond to the spin-orbit level can be important when there are heavy elements with *p* valence electrons. Also the band gap appears quite sensitive for the spin-orbit effect.

XC functional

The default functional is the LDA, that gives quite good geometries but terrible bonding energies. GGA functionals are usually better at bonding energies, and among all possibilities the PBE is a common choice. Using a GGA is not a lot more expensive than using plain LDA. For the special problem of band gaps there are a number *Model Hamiltonians* (page 13) available (eg. TB-mBJ and GLLC-SC). The *Unrestricted* (page 21) option will be needed when the system is not closed shell. For systems interacting through dispersion interactions it is advised to use the *Grimme corrections* (page 15). Unfortunately there is no clear-cut answer to this problem, and one has to try in practice what works best.

9.1.2 Technical Precision

See also:

- *Which basis set should I use?* (page 32)
- *Recommendations for k-space* (page 41)

The easiest way to control the technical precision is via the *NumericalQuality* (page 31) key. One can also independently tweak the precision of specific technical aspects, e.g.:

```
BeckeGrid
  Quality Good ! tweak the grid
End
KSpace
  Quality Good  ! tweak the k-space grid
End
ZlmFit
  Quality Normal   ! tweak the density fit
```

```
End
SoftConfiment
  Quality Basic   ! tweak the radial confinement of basis functions
End
```

Here are per issue hints for when to go for a better quality (but it is by no means complete)

- **BeckeGrid**: Increase quality if there are geometry convergence problems. Also negative frequencies can be caused by an inaccurate grid.
- **KSpace**: Increase quality for metals
- **ZlmFit**: Increase quality if the SCF does not converge.
- **SoftConfinment**: Increase quality for weakly bonded systems, such as layered materials

9.1.3 Performance

The performance is influenced by the model Hamiltonian and basis set, discussed above. Here follow more technical tips.

Reduced precision

One of the simplest things to try is to run your job with NumericalQuality Basic. For many systems this will work well, and it can be used for instance to pre-optimize a geometry. However, it can also cause problems such as problematic SCF convergence, geometry optimization, or simply bad results. See above how to tweak more finely the *Technical Precision* (page 115).

Memory usage

Another issue that is the choice CPVector (say the vector length of you machine) and the number of k-points processed together during the calculation of the parameters. In the output you see the used value

```
                 =========================
                 = Numerical Integration =
                 =========================

TOTAL NR. OF POINTS                                4738
BLOCK LENGTH                                        256
NR. OF BLOCKS                                        20
MAX. NR. OF SYMMETRY UNIQUE POINTS PER BLOCK         35
NR. OF K-POINTS PROCESSED TOGETHER IN BASPNT          5
NR. OF SYMMETRY OPERATORS (REAL SPACE)               48
SYMMETRY OPERATORS IN K-SPACE                        48
```

If you want to change the default settings you can specify the `CPVector` and `KGRPX` keywords. The optimal combination depends on the calculation, on the machine. Example

```
CPVector 512
KGRPX 3
```

Note: bigger is not necessarily better.

Reduced basis set

When starting work on a large unit cell it is wise to start with a DZ basis. With such a basis, one can test for instance the quality of the k-space integration. However, for most properties, the DZ basis is probably not very accurate. You can next go for the DZP (if available) or TZP basis set, but that may be a bit of overkill.

Frozen core for 5d elements

The standard basis sets TZ2P are not optimal for third-row transition elements. Sometimes you need to relax the frozen core dependency criterion

```
Dependency Core=0.8 ! The frozen core overlap may not be exactly 1
```

9.2 Troubleshooting

9.2.1 SCF does not converge

Some systems are more difficult to converge than others. A Pd slab for instance is easier to converge than an Fe slab. Generally, what you do in a problematic case is to go for more conservative settings. The two main option are to decrease `SCF%Mixing` and/or `DIIS%Dimix`.

```
SCF
   Mixing 0.05 ! more conservative mixing
End

Diis
   DiMix 0.1 ! also more conservative strategy for DIIS procedure
   Adaptable false ! disable automatic changing of dimix
End

Convergence
   Degenerate Default ! For most calculations this is quite a good idea anyway
End
```

An other option is to first run the system with a SZ basis, which may be easier to converge. Then you can *Restart* (page 103) the SCF with a larger basis set from this result.

Sometimes SCF convergence problems are caused by bad precision. An indication of this is when there are many iteration after the HALFWAY message. The simplest thing to try is to see whether increasing the `NumericalAccuracy` helps. Specifically an insufficient quality of the **density fit** may cause problems. For systems with heavy elements the quality of the **Becke grid** may also play a role. Another potential problem is using **only one k-point**.

Next thing to try is the MultiSecant method. This one comes at no extra cost per SCF cycle compared to the DIIS method.

```
SCF
  Method MultiSecant
End

MultiSecantConfig
   ! put here optional keywords to tweak the MultiSecant method
End
```

An alternative is to try a **LIST** method. For sure the cost of a single SCF iteration will increase, but it may reduce the number of SCF cycles, see *Diis%Variant* (page 52).

```
Diis
  Variant LISTi ! invoke the LISTi method
End
```

For heavy elements the use of a small or no frozen core may complicate the SCF convergence.

9.2.2 Geometry does not converge

One thing that you should make sure is that at least the **SCF converges**. If that is so, then maybe the **gradients are not accurate enough**. Here are some settings to improve the accuracy of the gradients

```
RadialDefaults
  NR 10000   ! more radial points
End

NumericalQuality Good
```

9.2.3 Negative frequencies in phonon spectra

When doing a phonon calculation one sometimes encounter unphysical negative frequencies. There are two likely causes: either the **geometry was not in the minimum geometry**, or the **step size** used in the Phonon run is too large. Also **general accuracy** issues may be the cause, such as numerical integration, k-space integration and fit error.

9.2.4 Basis set dependency

A calculation aborts with the message: dependent basis. It means that for at least one k-point in the BZ the set of Bloch functions, constructed from the elementary basis functions is so close to linear dependency that the numerical accuracy of results is in danger. To check this, the program computes, for each k-point separately, the overlap matrix of the Bloch basis (normalized functions) and diagonalizes it. If the smallest eigenvalue is zero, the basis is linearly dependent. (Negative values should not occur at all!). Given the limited precision of numerical integrals and other aspects in the calculation, you are bound for trouble already if the smallest eigenvalue is very small, even if not exactly zero. The program compares it against a criterion that can be set in input (key `Dependency` option *Bas*).

If you encounter such an error abort, you are strongly advised not to adjust the criterion so as to pass the internal test: there were good reasons to implement the test and to set the default criterion at its current value. Rather, you should adjust your basis set. There are two ways out: using confinement or removing basis functions.

Using confinement

Usually the dependency problem is due to the diffuse basis functions. This is especially so for highly coordinated atoms. One way to reduce the range of the functions is to use the Confinement key. In a slab you could consider to use confinement only in the inner layers, and to use the normal basis to the surface layers. The idea is that basis functions of the surface atoms can describe the decay into the vacuum properly, and that inside the slab the diffuseness of the functions is not needed. If all the atoms of the slab are of the same type, you should make a special type for the inner layers: simply put them in a separate Atoms block. The confinement can be specified per type.

Removing basis functions

You can remove one or more basis functions and maybe modify some of the (other) STO basis functions. The program prints information that helps you determine which basis functions should be modified/removed. Another way to modify your basis set, is to use the confinement keyword. This has the effect of making the diffuse basis functions more localized, thus reducing problematically large overlap with similar functions on neighboring atoms.

In the standard output file, after the error message, you will find a list of eigenvalues of the overlap matrix. If only the first is smaller than the threshold, you should remove one basis function. If more eigenvalues are very small, it is likely that you have to remove more than one function, although you can of course try how far you can get by eliminating just one.

Next the program prints the so-called Dependency Coefficients: a list of numbers, one for each basis function. Those with a large value are the suspicious ones. If you find two coefficients that are significantly larger than the others, you should replace the two corresponding functions by one. Easiest is to remove one of them (take the one with the bigger coefficient). If one of them is a numerical orbital from Dirac and the other an STO, remove the STO. If both are STOs, remove one and replace the other by some kind of average (regarding the radial characteristic: exponential factor and power of radial coordinate).

To identify how the functions in your input correspond to the list the underlies the series of Dependency Coefficients, you have to set up the list of basis functions as follows:

- Consider an outer loop over all atom TYPES. These correspond, in order as well as in number, to the sequence of AtomType keys in your input file.
- For each type, consider a loop over all atoms of that type, i.e. the atoms in the ATOM block corresponding to the AtomType key at hand.
- For each atom (each AtomType key), first write down all DIRAC basis functions, then all STOs. When writing down the functions, be aware that each entry in your input file specifies a function *set*, by the quantum number L and hence corresponds to 2L+1 actual basis functions.
- Regarding the DIRAC basis functions: they belong to the list of basis functions only if the key Valence occurs in the pertaining DIRAC input block. If not, no DIRAC functions of that type are included in the basis. *If* the Dirac functions are included, you must omit the Core functions and include only the Valence functions from that DIRAC block. The first record in your DIRAC block with two numbers defines (by the first number) the total number of function sets in the DIRAC block (which you can verify by simple counting) and (by the second number) the number of Core function sets among them. The Core function sets, if any, are always the first so many in the list in the DIRAC block.

The program stops as soon as it encounters a dependency problem. This may happen for the first k-point. After you have adjusted the basis set following the above guidelines, you will have solved it. However, it may easily happen that the problem shows up again, but now for another (later) k-point, where other entries in the basis set may cause trouble. Do not think you have repaired the first problem incorrectly. Just repeat the procedure until you pass all k-points in the basis set construction without errors. Typically (as a last remark), although not necessarily, the first k-point may have a dependency problem from too many *s*-type functions, while other k-points may be more sensitive to the series of *p*-functions in your basis.

9.2.5 Frozen core too large

BAND calculates the overlap matrix of the core functions, and this should approximate the unit matrix. When the deviation is larger then the frozen-core overlap criterion the program stops. The default criterion (0.98) is fairly strict. The safest solution is to choose a smaller frozen core. For performance reasons, however, this may not be the preferred option. In practice you might still get reliable results by setting the criterion to 0.8, see the :ref:Dependency <key-Dependency>‘ block. For the *5d* transition metals, for instance, you can often freeze the 4f orbital, thus reducing the basis set considerably. We strongly advise you to compare these results to a calculation with a smaller core. Such tests can be performed with a smaller unit cell or with a lower quality for the `KSPACE` block key.

9.3 Various issues

9.3.1 Understanding the logfile

In practice you will look often at the logfile to see whether the calculation is going fine. Here is a logfile for a single point calculation.

```
<Jul10-2018> <18:24:55>  AMS development version  RunTime: Jul10-2018 18:24:55 ␣
↪Nodes: 1  Procs: 8
<Jul10-2018> <18:24:56>  BAND development version  RunTime: Jul10-2018 18:24:56 ␣
↪Nodes: 1  Procs: 8
<Jul10-2018> <18:24:56>  custom tolerance for symmetry detection
<Jul10-2018> <18:24:56>  All basis functions smoothly confined at radius:  4.8
<Jul10-2018> <18:24:56>  CalcAtomicProperties
<Jul10-2018> <18:24:56>  ------ K ..  14
<Jul10-2018> <18:24:56>  ------ K ..  14
<Jul10-2018> <18:24:57>  start of SCF loop
<Jul10-2018> <18:24:57>  initial density from psi
<Jul10-2018> <18:24:57>  cyc=  0 err=0.00E+00 cpu=   1s ela=   1s
<Jul10-2018> <18:24:58>  cyc=  1 err=5.72E-01 meth=m nvec= 1 mix=0.0750 cpu=   1s␣
↪ela=   1s fit=4.01E-02
<Jul10-2018> <18:24:59>  cyc=  2 err=5.22E-01 meth=d nvec= 2 mix=0.2000 cpu=   1s␣
↪ela=   1s fit=1.46E-02
<Jul10-2018> <18:24:59>  cyc=  3 err=4.35E-02 meth=d nvec= 2 mix=0.2200 cpu=   1s␣
↪ela=   1s fit=1.60E-02
<Jul10-2018> <18:25:00>  cyc=  4 err=3.29E-02 meth=d nvec= 2 mix=0.2420 cpu=   1s␣
↪ela=   1s fit=2.05E-02
<Jul10-2018> <18:25:00>  cyc=  5 err=6.22E-03 meth=d nvec= 2 mix=0.2662 cpu=   1s␣
↪ela=   1s fit=2.07E-02
<Jul10-2018> <18:25:01>  cyc=  6 err=4.26E-03 meth=d nvec= 2 mix=0.2928 cpu=   1s␣
↪ela=   1s fit=2.10E-02
<Jul10-2018> <18:25:02>  HALFWAY
<Jul10-2018> <18:25:02>  cyc=  7 err=6.23E-04 meth=d nvec= 2 mix=0.3221 cpu=   1s␣
↪ela=   1s fit=2.10E-02
<Jul10-2018> <18:25:02>  cyc=  8 err=3.82E-04 meth=d nvec= 2 mix=0.3543 cpu=   1s␣
↪ela=   1s fit=2.09E-02
<Jul10-2018> <18:25:03>  cyc=  9 err=6.13E-05 meth=d nvec= 2 mix=0.3897 cpu=   1s␣
↪ela=   1s fit=2.09E-02
<Jul10-2018> <18:25:04>  cyc= 10 err=3.49E-05 meth=d nvec= 2 mix=0.4287 cpu=   1s␣
↪ela=   1s fit=2.09E-02
<Jul10-2018> <18:25:04>  SCF CONVERGENCE
<Jul10-2018> <18:25:04>  cyc= 11 err=2.91E-06 meth=d nvec= 2 mix=0.4716 cpu=   1s␣
↪ela=   1s fit=2.09E-02
<Jul10-2018> <18:25:05>  cyc= 12 err=2.91E-06 meth=d nvec= 1 mix=1.0000 cpu=   1s␣
↪ela=   1s fit=2.09E-02
<Jul10-2018> <18:25:05>  ENERGY OF FORMATION:   -0.7677 A.U.
<Jul10-2018> <18:25:05>                        -20.8894 E.V.
<Jul10-2018> <18:25:05>                       -481.7210 KCAL/MOL
<Jul10-2018> <18:25:05>  FERMI ENERGY:          -0.1642 A.U.
<Jul10-2018> <18:25:05>                         -4.4677 E.V
<Jul10-2018> <18:25:05>  Band gap:               0.1614 A.U.
<Jul10-2018> <18:25:05>                          4.3911 E.V
<Jul10-2018> <18:25:05>  Storing all partial DOS
<Jul10-2018> <18:25:05>  Integrate over delta E
<Jul10-2018> <18:25:05>  partial dos
<Jul10-2018> <18:25:05>  copy T(V/VOC)
<Jul10-2018> <18:25:05>  copy eigensystem
```

```
<Jul10-2018> <18:25:05>  NORMAL TERMINATION
```

There are three different phases. The first phase is the preparation phase. The second phase is the SCF procedure. The third part is the properties phase. Particularly important are the SCF CONVERGENCE and NORMAL TERMINATION messages.

Let us take a closer look at a line during the SCF.

```
<Jul10-2018> <18:24:59>  cyc=  3 err=4.35E-02 meth=d nvec= 2 mix=0.2200 cpu=   1s␣
↪ela=   1s fit=1.60E-02
```

The meaning of cyc is the iteration number, so it is the third iteration. The self consistent error (err) is 4.35E-02. The method (meth) to guess the density for the next cycle is d, meaning DIIS, being a linear combination (nvec) of two vectors. The density is biased (mix) by 0.2 towards output densities. The SCF cycle took 1 second of cpu time (per core), and needed 1 seconds of real time. Finally the error of the density fitting was 1.60E-02

9.3.2 Breaking the symmetry

In some cases you want to break the symmetry. An example of this is when you want to get the antiferromagnetic state of Fe. Another common example is when you want to apply geometry constraints on atoms.

The easiest way to do this is of course to disable all symmetry, see *UseSymmetry key* (page 111), but this might make your calculation more expensive than is needed. A bit more elegant way is to define separate types for the equivalent atoms. Here follows an example input for antiferromagnetic iron

```
! The two iron atoms have different "types" to break the symmetry
System
    Atoms
      Fe.a   0.0    0.0    0.0
      Fe.b  -1.435 -1.435  1.435
    End
End

Lattice
   -1.435  1.435  1.435
    1.435 -1.435  1.435
    2.87   2.87  -2.87
End

...
...

Band Engine

  ...
  CONVERGENCE
      CRITERION 1.0e-4
      Degenerate default
      SpinFlip 2  ! Flip (startup) spin density at second atom
  END
  ...
EndEngine
```

Another solution is to use the expert SYMMETRY keyword.

9.3.3 Labels for the basis functions

You see the labels for the basis functions in for instance the DOS section of the output. The labels are also used in combination with options like `Print Eigens` and `Print OrbPop`.

What do the labels look like? A normal atomic basis function, i.e. a numerical orbital or a Slater type orbital, gets a label like <atom number>/<element>/<orbital type>/<quantum numbers description>/<exp in sto>

Example with a Li and a H atom:

```
1/LI/NO/1s
1/LI/NO/2s
1/LI/STO/2s/1.4
1/LI/STO/2p_y/1.3
1/LI/STO/2p_z/1.3
1/LI/STO/2p_x/1.3
2/H/NO/1s
2/H/STO/1s/1.9
...
```

Core states will just get simple numbers as labels:

```
CORE STATE 1
CORE STATE 2
```

With the `Fragment` key you can give meaningful names to the fragment option, see `Fragment%Labels` and `DosBas`.

9.3.4 Reference and Startup Atoms

The formation energy of the crystal is calculated with respect to the reference atoms. BAND gives you the formation energy with respect to the spherically symmetric spin-*restricted* LDA atoms. If you want the program to do the spin-unrestricted calculation for the atoms you can give key Unrestricted the extra option Reference. We do not recommend this as it would give you the false (except in special cases) feeling that you've applied the right atomic correction energy so as to obtain the 'true' bonding energy with respect to isolated atoms. The true atomic correction energy is the difference in energy between the used artificial object, i.e. the spherically symmetric, spin-restricted atom with possibly fractional occupation numbers, and the appropriate multiplet state. The spin-*un*restricted reference atom would still be spherically symmetric, with possibly fractional occupations: it would only have the probably correct (Hund's rule) net spin polarization.

The startup density is normally the sum of the restricted atoms. In case you do an unrestricted calculation you may want to get the sum of the unrestricted atoms as startup density by giving key Unrestricted the extra option StartUp. This does not always provide a better startup density since all atoms will have their net-spins pointing up. If a frozen core is used this option can sometimes lead to a negative valence density, because the frozen core is derived from the restricted atom. The program will stop in such a case.

No matter what reference or startup atoms you use, core orbitals and NOs originate always from the restricted free-atom calculation, because we don't want a spatial dependence of the *basis functions* on spin.

9.3.5 Numerical Atoms and Basis functions

The program starts with a calculation of the free atoms, assuming spherical symmetry. The formation energy is calculated w.r.t such atoms. You have to specify the configuration (i.e. which orbitals are occupied) in the Dirac subkey of the block key AtomType, and you can for instance use the experimental configuration. Keep in mind, however, that this is not necessarily the optimal configuration for your density functional. For instance, Ni has experimentally two

electrons in the 4s shell, but with LDA you will find that it is energetically more profitable to move one electron from the 4s to the 3d. The configuration of the reference atoms does not (i.e. should not) affect the final (SCF) density.

Besides the available basis sets in $ADFHOME/atomicdata/band, you could in principle use the basis functions from the database of the molecular ADF program (see the documentation of ADF for how this database is organized). The functions you will find there are STOs, which is not optimal since BAND offers you the option to use NOs from the numerical atom. The most efficient approach is to use the NOs and remove from the ADF basis set those STOs that are already well described by the NOs.

As an example we will construct a basis for the Ni atom with orbitals frozen up to the *2p* shell, derived from a triple-zeta ADF basis. In the Dirac subkey of the block key AtomType you specify that the NOs up to *2p* should be kept frozen and that the 3d and *4s* NOs be included in the valence basis. Copy from the ADF database all *3d*, *4s* and the polarization functions into the BasisFunctions subkey of the block key AtomType and remove the middle STOs of the *3d* and the *4s*.

Usually it is already quite adequate for a good-quality basis to augment each NO with one STO. You could then take a double zeta ADF basis and remove one of the 3d and one of the 4s STOs. We often find that such a basis, with one STO added per NO, has a quality that is comparable to *triple* zeta STO sets. We strongly recommend that you use combined NO/STO bases. Of course, you may want to verify the quality of the basis set by calculations on a few simple systems.

9.4 Warnings

9.4.1 Warnings specific to periodic codes (BAND, DFTB)

```
WARNING: GUESS: INPUT LATTICE MAY BE INACCURATE

WARNING: KSPACE ONLY GAMMA POINT.

WARNING: LINEAR INTEGRATION IN K-SPACE

WARNING: Problem with Brillouin zone k-path generation (see output file)

WARNING: problematic mapping to central cell

WARNING: QUAD2:  electronic temperature problem
```

CHAPTER

TEN

EXAMPLES

10.1 Introduction

The ADF package contains a series of sample runs for the BAND program. Provided are UNIX scripts to run the calculations and the resulting output files.

The examples serve:

- To check that the program has been installed correctly: run the sample inputs and compare the results with the provided outputs. *Read the remarks below about such comparisons.*
- To demonstrate how to do calculations: an illustration to the User manuals. The number of options available in BAND 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.

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.

All sample files are stored in subdirectories under $ADFHOME/examples/, where $ADFHOME is the main directory of the ADF package. The main subdirectory for the BAND examples is $ADFHOME/examples/band. Each sample run has its own directory. For instance, $ADFHOME/examples/band/NaCl/ contains a BAND calculation on the NaCl bulk crystal. Each sample subdirectory contains:

- A file TestName.run: the UNIX script to execute the calculation or sequence of calculations of the example
- A file TestName_orig.out: the resulting output(s) against which you can compare the outcome of your own calculation.

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 database. 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', 'band', and so on. When you inspect your $ADFBIN directory, however, you may find that the program executables have names 'adf.exe', 'band.exe'. There are also files in $ADFBIN with names 'adf', 'band', 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

Many 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 Guide and the Utilities and Property Programs documents (NMR, DIRAC, and other utility programs).

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, 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.
- A BAND run may generate, apart from result files that you may want to save, a few scratch files. The UNIX scripts that run the samples take care of removing these files after the calculations have finished, to avoid that the program aborts in the next run by attempting to open a 'new' file that is found to exist already.
- A sample calculation may use one or more data files, in particular *fragment* files. The samples are self-contained: they first run the necessary pre-calculations to produce the fragment files. In 'normal' research work you may have libraries of fragments available, first for the 'basic atoms', and later, as projects are developing, also for larger fragments so that you can start immediately on the actual system by attaching the appropriate fragment files.

Default settings of print options result in a considerable amount of output. This is also the case in some of the sample runs, although in many of them quite a bit of 'standard' output is suppressed by inserting applicable print control keys in the input file. Consult the User's Guide about how to regulate input with keys in the input file.

10.2 Model Hamiltonians

10.2.1 Example: Spin polarization: antiferromagnetic iron

Download BetaIron.run

```
#! /bin/sh

# With the UNRESTRICTED keyword we do a spin polarized calculation. Normally
# this would converge to the ferromagnetic solution.

# With the SpinFlip keyword we make sure that we start with an antiferromagnetic
# density.

# For antiferromagnetic iron we need a larger unit cell of two atoms. Since
# these atoms appear to the program as symmetry equivalent we have to specify
# them as separate types.

$ADFBIN/ams <<eor

Task SinglePoint

System
  ! The two iron atoms have different "types" to break the symmetry
  ATOMS
    Fe.a   0.0    0.0    0.0
    Fe.b  -1.435 -1.435  1.435
  end

  Lattice
     -1.435  1.435  1.435
      1.435 -1.435  1.435
      2.87   2.87  -2.87
  End
End

Engine Band
  TITLE Beta iron

  CONVERGENCE
     CRITERION 1.0e-4
     Degenerate default
     SpinFlip 2  ! Flip (startup) spin density at second atom
  END

  Basis
     Type DZ
     Core Large
  End

  UNRESTRICTED

  Print AtomicChargesDetails
EndEngine
eor
```

10.2.2 Example: Applying a Magnetic Field

Download BFieldLdotB.run

```
#!/bin/sh

STRUCTDIR=$ADFHOME/atomicdata/Molecules/TestMols

class=Molecules
#STRUCTDIR=$HOME/projects/Structures/$class
system=Methane

bas=QZ4P
core=None

component=bz

bz=0.001

export AMS_JOBNAME=run1

$ADFBIN/ams <<EOF
Task SinglePoint

System
   GeometryFile $STRUCTDIR/$system.xyz
End

Engine band
   print timing

   BField
    Unit A.U.
    $component $bz
    Method NR_LDOTB
   end


   Basis
     Type $bas
     Core $core
   End

EndEngine

EOF

echo "Begin of shielding row for all atoms, unit-=ppm"
$ADFBIN/adfreport $AMS_JOBNAME'.results/band.rkf' -k 'Magnetic properties
↪%ShieldingRowAtNuclei(ppm)#12.5f##3'
echo "End of shielding row"


export AMS_JOBNAME=run2

$ADFBIN/ams <<EOF
Task SinglePoint
```

```
System
   GeometryFile $STRUCTDIR/$system.xyz
End

Engine band
   print timing


   BField
    Unit A.U.
    $component $bz
    Method NR_LDOTB
    Dipole true
    DipoleAtom 1
   end


   Basis
     Type $bas
     Core $core
   End

EndEngine

EOF

echo "Begin of shielding row for all atoms, unit-=ppm"
$ADFBIN/adfreport $AMS_JOBNAME'.results/band.rkf' -k 'Magnetic properties
↪%ShieldingRowAtNuclei(ppm)#12.5f##3'
echo "End of shielding row"
```

10.2.3 Example: Graphene sheet with dispersion correction

Download Graphene_Dispersion.run

```
#! /bin/sh

# A normal GGA would give only negligible interaction between two graphene
# sheets.

# Use the dispersion option in the XC key block.

# In the first run we use BP86-D, in the second BLYP-D3 and in the third run
# BLYP-D3(BJ).

# == First run: dispersion default ==

AMS_JOBNAME=default $ADFBIN/ams <<eor

Task SinglePoint

System
   GeometryFile $ADFHOME/examples/band/Graphene_Dispersion/Graphene_double_layer.xyz
End

Engine Band
```

```
   XC
      gga scf bp86
      dispersion default
   End

   NumericalQuality Basic

   Basis
      Type TZP
      Core Large
   End
EndEngine
eor

# == Second run: dispersion Grimme3 ==

AMS_JOBNAME=grimme3 $ADFBIN/ams <<eor

Task SinglePoint

System
   GeometryFile $ADFHOME/examples/band/Graphene_Dispersion/Graphene_double_layer.xyz
End

Properties
  Gradients True
End

Engine Band
   XC
      gga scf blyp
      dispersion Grimme3
   end

   NumericalQuality Basic

   Basis
      Type TZP
      Core Large
   End
EndEngine
eor

# == Third run: dispersion Grimme3 bjdamp ==

AMS_JOBNAME=grimme3_bjdamp $ADFBIN/ams <<eor

Task SinglePoint

System
   GeometryFile $ADFHOME/examples/band/Graphene_Dispersion/Graphene_double_layer.xyz
End

Properties
  Gradients True
End

Engine Band
```

```
   XC
      gga scf blyp
      dispersion Grimme3 bjdamp
   end

   NumericalQuality Basic

   Basis
      Type TZP
      Core Large
   End
EndEngine
eor
```

10.2.4 Example: H on perovskite with the COSMO solvation model

Download HonPerovskite_Solvation.run

```
#! /bin/sh

# We want to model H adsorption on a Perovskite surface in a solution, modeled
# by a COSMO surface.

# We create only the COSMO surface above the slab with the
# RemovePointsWithNegativeZ option.

$ADFBIN/ams <<eor

Task SinglePoint

System
   Atoms
      H    0.0  0.000000000  0.900000000
      Ca   0.0  0.000000000  0.000000000
      Ca   0.0  3.535533906 -3.535533906
      Ti  -2.5 -3.535533906  0.000000000
      Ti  -2.5  0.000000000 -3.535533906
      O    0.0 -3.535533906  0.000000000
      O    2.5  1.767766953 -1.767766953
      O    2.5 -1.767766953 -1.767766953
   End
   Lattice
      5.0 0.000000000 0.0
      0.0 7.071067812 0.0
   End
End

Properties
   Gradients True
End

Engine Band
   TITLE Hydrogen on Perovksite wit solvation

   Solvation
      Enabled True
```

```
      Surf Delley
      charge method=inver
      Solvent
         Eps 78.4
         Rad 1.4
      End
   End

   PeriodicSolvation
      nstar 3
      SymmetrizeSurfacePoints true
      RemovePointsWithNegativeZ true
   End

   Screening
      rmadel 30 ! to speed up the calculation
   End

   Convergence
      Criterion 1.0e-4
   End

   Basis
      Type SZ
      Core Large
   End
EndEngine
eor
```

10.2.5 Example: Applying a homogeneous electric field

Download EField.run

```
#!/bin/sh

# With the EFIELD keyword you can specify a static electric field in the
# z-direction.

# == first run: field specified in a.u. ==

$ADFBIN/ams <<eor

Task SinglePoint

System
   lattice [Bohr]
      15.0 0.0  0.0
      0.0  15.0 0.0
   End
   Atoms [Bohr]
      H   0.0 0.0 0.0
      Li  0.0 1.0 3.0
   End
End

Properties
```

```
  Gradients True
End

Engine Band
   Title Electric Field (field in a.u.)

   EField
      Unit a.u.
      eZ 0.03
   end

   KSpace
      Quality GammaOnly
   End

   Basis
      Type TZP
      Core Large
   End
EndEngine
eor

rm -r ams.results

# == second run: field specified in Volt/Angstrom ==

$ADFBIN/ams <<eor

Task SinglePoint

System
   lattice [Bohr]
      15.0 0.0  0.0
      0.0  15.0 0.0
   End
   Atoms [Bohr]
      H   0.0 0.0 0.0
      Li  0.0 1.0 3.0
   End
End

Properties
  Gradients True
End

Engine Band
   Title Electric Field (field in Volt/Angstrom)

   EField
      Unit Volt/Angstrom
      eZ -1
   End

   KSpace
      Quality GammaOnly
   End

   Basis
```

```
      Type TZP
      Core Large
   End
EndEngine

eor
```

10.2.6 Example: Finite nucleus

Download FiniteNucleus.run

```
#! /bin/sh

# Normally the nucleus is approximated as a point charge. However we can change
# this to a finite size. Properties that might be affected are EFG, and the
# A-tensor. For such calculations one needs to crank up the precision and also
# use a relativistic Hamiltonian.

# == First run: NuclarModel PointCharge ==

AMS_JOBNAME=PointCharge $ADFBIN/ams <<eor

Task SinglePoint

System
  lattice
     30.0 0.0 0.0
  End

  Atoms
     Au  0.000000      0.000000      0.000000
  End
End

Engine Band
  NuclearModel PointCharge

  Efg
    Enabled True
  End

  Atensor
    Enabled True
  End

  Unrestricted
  Relativity
    Level Scalar
  End

  PropertiesAtNuclei
    rho
    rho(deformation/scf)
    vxc[rho(fit)]
    rho(fit)
    v(coulomb)
```

```
  End

  RadialDefaults
     nr 10000
  End

  NumericalQuality Good

  Basis
     Type TZ2P
     Core None
  End

  XC
     gga PBE
  END
EndEngine
eor

# == Second run: NuclearModel Gaussian ==

AMS_JOBNAME=Gaussian $ADFBIN/ams <<eor

Task SinglePoint

System
  lattice
     30.0 0.0 0.0
  End

  Atoms
     Au  0.000000      0.000000      0.000000
  End
End

Engine Band
  NuclearModel Gaussian

  Efg
    Enabled True
  End

  Atensor
    Enabled True
  End

  Unrestricted
  Relativity
    Level Scalar
  End

  PropertiesAtNuclei
    rho
    rho(deformation/scf)
    vxc[rho(fit)]
    rho(fit)
    v(coulomb)
  End
```

```
  RadialDefaults
     nr 10000
  End

  NumericalQuality Good

  Basis
     Type TZ2P
     Core None
  End

  XC
     gga PBE
  END
EndEngine

eor
```

10.2.7 Example: Fixing the Band gap of NiO with GGA+U

Download NiO_Hubbard.run

```
#! /bin/sh

# With the UNRESTRICTED keyword we do a spin polarized calculation.

# With the HubbardU key block we set up the GGA+U calculation. You need to
# specify per atom type (only two here, Ni, and O) the U and the l-value to
# which it should be applied.


$ADFBIN/ams <<eor

Task SinglePoint

System
   Atoms
      Ni 0.0   0.0   0.0
      O  2.085 2.085 2.085
   End

   Lattice
      0.000 2.085 2.085
      2.085 0.000 2.085
      2.085 2.085 0.000
   End
End

Engine Band
   Title NiO GGA+U (Hubbard)

   Unrestricted

   HubbardU
      Enabled True
```

```
      PrintOccupations True
      uvalue 0.3 0.0
      lvalue 2 -1
   End

   KSpace
      Symmetric KInteg=3
      Type Symmetric
   End

   Basis
      Type TZP
      Core Large
   End

   XC
      GGA Becke Perdew
   End

   Print AtomicChargesDetails
EndEngine

eor
```

10.2.8 Example: Fixing the band gap of ZnS with the TB-mBJ model potential

Download ZnS_ModelPotential.run

```
#! /bin/sh

# With the XC subkey model we invoke the so-called TB-mBJ model potential, which
# increases band gaps for solids.



AMS_JOBNAME=TB-mBJ $ADFBIN/ams <<eor

Task SinglePoint

System
   ATOMS
      Zn  0.0000  0.0000  0.0000
      S   1.3525  1.3525  1.3525
   END

   Lattice
      0.000  2.705  2.705
      2.705  0.000  2.705
      2.705  2.705  0.000
   End
End

Engine Band
   TITLE ZnS pot=TB-mBJ

   XC
```

```
      model TB-mBJ
   END

   Basis
      Type DZ
      Core Large
   End
EndEngine
eor


AMS_JOBNAME=GLLB-SC $ADFBIN/ams <<eor

Task SinglePoint

System
   ATOMS
      Zn  0.0000  0.0000  0.0000
      S   1.3525  1.3525  1.3525
   END
   Lattice
      0.000  2.705  2.705
      2.705  0.000  2.705
      2.705  2.705  0.000
   End
End

Engine Band
   TITLE ZnS pot=GLLB-SC

   XC
      model GLLB-SC
   END

   Basis
      Type DZ
      Core Large
   End
EndEngine
eor


AMS_JOBNAME=lb94 $ADFBIN/ams <<eor

Task SinglePoint

System
   Atoms
   H       0.000000000     0.000000000    -0.370500000
   H       0.000000000     0.000000000     0.370500000
   End
End

Engine Band
   Title H2 pot=lb94

   XC
    model lb94
```

```
   end

   Basis
      Type TZP
      Core Large
   End
EndEngine
eor
```

10.3 Precision and performance

10.3.1 Example: Convenient way to specify a basis set

Download BasisDefaults.run

```
#! /bin/sh

# This example shows some of the flexibility of the Basis key. The
# defaults are set to a DZ basis set with a Large frozen core. As the example
# shows, it is possible to override the defaults per atom type:

$ADFBIN/ams <<eor

Task SinglePoint

System
   Atoms [Bohr]
      C              0.0   0.0   0.0
      O.large_basis  0.0   2.13  0.0
      H.large_basis  4.0   0.0   0.0
      H              4.0   1.43  0.0
   End
End

Engine Band
   Title CO + H2: fine tuning the basis defaults

   NumericalQuality Basic

   Basis
      ! Cheap defaults
      Type DZ
      Core Large
      ByAtomType
         C              Core=None    ! This C has no frozen core
         O.large_basis  Type=TZ2P    ! This O with a larger basis
         H.large_basis  Type=V       ! This one also with a larger basis
      End
   End
EndEngine
eor
```

10.3.2 Example: Tuning precision and performance

Download Peptide_NumericalQuality.run

```
#! /bin/sh

# This example shows how to tune the numerical quality of the calculation. This
# will influence both efficiency and accuracy of the calculation.

$ADFBIN/ams <<eor

Task SinglePoint

System
   ATOMS
      C  -2.543276676    0.646016253   -0.226282061
      C  -1.380007216   -0.349821933   -0.099968062
      C   1.066549862   -0.581911934   -0.064823014
      C   2.223931363    0.423839954   -0.118070453
      N  -0.149937993    0.193000383   -0.179010633
      N   3.452833267   -0.128914507   -0.101813389
      O  -1.589886979   -1.564606357    0.062390357
      O   2.010772661    1.647347397   -0.186192833
      H  -2.480330907    1.422845016    0.554868474
      H   3.629655835   -1.142731500   -0.018098016
      H  -2.511564496    1.180719545   -1.193540463
      H   0.024515371    1.206808884   -0.244500253
      H   1.160598100   -1.320381370   -0.884522980
      H   1.071343640   -1.136930542    0.888913220
   END
   Lattice
      7.211585775    0.000000000    0.000000000
   End
End

Engine Band
   TITLE Quality

   NumericalQuality Normal

   ZlmFit
      Quality Normal
   End

   BeckeGrid
      Quality Basic
   End

   KSpace
      Quality Basic
   End

   SoftConfinement
      Quality VeryGood
   End

   Basis
      Type DZ
```

```
      Core Large
   End

   XC
      GGA PBE
   END
EndEngine
eor
```

10.3.3 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/ams <<eor

Task SinglePoint

System
  Atoms
     O.Accurate    0.00000000       0.00000000       0.00000000    ! 1
     H.Accurate    0.40399229      -0.65578342       0.63241539    ! 2
     H.Accurate    0.81410032       0.55624569      -0.23129097    ! 3
     O.Near       -3.02535626      -0.08473104      -0.47678489    ! 4
     H.Near       -2.56531481       0.62644005       0.07759386    ! 5
     H.Near       -2.25289644      -0.61700366      -0.80790649    ! 6
     O.Near        2.95394790      -0.54939973      -0.38206034    ! 7
     H.Near        3.91427727      -0.21304908      -0.44738291    ! 8
     H.Near        2.87780992      -1.13241278      -1.20853726    ! 9
     O.Far        -5.95425742      -0.56764616      -0.02016682    ! 10
     H.Far        -5.26308282      -0.46969096       0.69255963    ! 11
     H.Far        -5.42117992      -0.54361203      -0.86443121    ! 12
     O.Far         6.25171470      -0.62004899      -0.03702467    ! 13
     H.Far         6.16508647      -1.38696453       0.58541903    ! 14
     H.Far         7.09161199      -0.16700550       0.23679419    ! 15
  End
```

```
End

Engine Band
   XC
      LibXC B3LYP
   End

   ! ============================================================
   ! Set the NumerialQuality to be used for the atoms that are not
   ! explicitely defined in AtomDepQuality
   ! ============================================================

   NumericalQuality Normal

   ! ==========================================================
   ! Set different basis sets for atoms in different regions:
   ! ==========================================================

   Basis
      Type DZ
      ByAtomType
        O.Accurate Type=TZP Core=None
        H.Accurate Type=TZP Core=None
        O.Near     Type=DZ  Core=None
        H.Near     Type=DZ  Core=None
        O.Far      Type=SZ  Core=None
        H.Far      Type=SZ  Core=None
      End
   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
EndEngine

eor
```

10.3.4 Example: BSSE correction

Download BSSE.run

```
#!/bin/bash

# This example shows how to calculate the basis set superposition error for the
# interaction of CO with H2. In this shell script we loop over progressively
# better basis sets.

for bas in DZ TZ2P QZ4P
do

core=Large

$ADFBIN/ams <<eor
Task SinglePoint
System
  Atoms [Bohr]
    C 0 0 0
    O 2.13 0 0
    H 0 0 6
    H 1.5 0 6
  End
End

Engine Band
```

```
  Basis
    Type $bas
    Core $core
  End
EndEngine
eor

ECOH2=`$ADFBIN/adfreport  ams.results/band.rkf -r 'Bond energies%final bond energy'`
rm -r ams.results

$ADFBIN/ams <<eor
Task SinglePoint
System
  Atoms [Bohr]
    H 0 0 6
    H 1.5 0 6
  End
End

Engine Band
  Basis
    Type $bas
    Core $core
  End
EndEngine

eor

EH2=`$ADFBIN/adfreport  ams.results/band.rkf -r 'Bond energies%final bond energy'`
rm -r ams.results


$ADFBIN/ams <<eor
Task SinglePoint
System
  Atoms [Bohr]
    Gh.C 0 0 0
    Gh.O 2.13 0 0
    H 0 0 6
    H 1.5 0 6
  End
End

Engine Band
  Basis
    Type $bas
    Core $core
  End
EndEngine

eor

EH2_GHOST_CO=`$ADFBIN/adfreport  ams.results/band.rkf -r 'Bond energies%final bond␣
↪energy'`
rm -r ams.results


$ADFBIN/ams <<eor
```

```
Task SinglePoint
System
  Atoms [Bohr]
    C 0 0 0
    O 2.13 0 0
  End
End

Engine Band
  Basis
    Type $bas
    Core $core
  End
EndEngine

eor

ECO=`$ADFBIN/adfreport  ams.results/band.rkf -r 'Bond energies%final bond energy'`
rm -r ams.results


$ADFBIN/ams <<eor
Task SinglePoint
System
  Atoms [Bohr]
    C 0 0 0
    O 2.13 0 0
    Gh.H 0 0 6
    Gh.H 1.5 0 6
  End
End

Engine Band
  Basis
    Type $bas
    Core $core
  End
EndEngine

eor

ECO_GHOST_H2=`$ADFBIN/adfreport  ams.results/band.rkf -r 'Bond energies%final bond␣
↪energy'`
rm -r ams.results


EV=27.212
echo "Start report"
echo "basis set: $bas"
echo "H2 + CO : $ECOH2"
echo "H2 : $EH2"
echo "H2 (with ghost CO) : $EH2_GHOST_CO"
echo "CO : $ECO"
echo "CO (with ghost H2) : $ECO_GHOST_H2"
BSSEEV=`$ADFBIN/startpython -c "print (( $EH2 - $EH2_GHOST_CO + $ECO - $ECO_GHOST_H2␣
↪) *$EV)"`
echo "BSSE correction: $BSSEEV (eV)"
BOND1EV=`$ADFBIN/startpython -c "print (( $ECOH2 - $EH2 - $ECO ) *$EV)"`
```

```
BOND2EV=`$ADFBIN/startpython -c "print ($BOND1EV + $BSSEEV)"`
echo "Bond energy: $BOND1EV (eV)"
echo "Bond energy + BSSE: $BOND2EV (eV)"
echo "End report"

done
```

10.4 Restarts

10.4.1 Example: Restart the SCF

Download RestartSCF.run

```
#! /bin/sh

# This example shows how you can continue with an unfinished calculation. It
# consists of two runs. After the first run the RUNKF file is saved, and the
# renamed file is used in the second run. The second run is almost a copy for
# the first, except for the Restart key. It is also possible to restart from a
# smaller basis set (provided that the functions are contained in the bigger
# basis set). Finally you can also restart from a density matrix, but this
# should be explicitly saved (unlike the orbitals).

# ----------------------------- first run --------------------------

AMS_JOBNAME=BChain $ADFBIN/ams <<eor

Task SinglePoint

System
   Lattice [Bohr]
      4.0 0.0 0.0
   End
   Atoms [Bohr]
      B 0.0 0.0 0.0
   End
End

Engine Band
   Title B chain

   NumericalQuality Good

   skip dos

   XC
      GGA Becke Perdew
   END

   UNRESTRICTED

   DIIS
      NCycleDamp 0
      DiMix 0.5
      Adaptable false ! Otherwise it converges to a spin-restricted solution
```

```
   End

   Basis
      Type TZ2P
      Core Large
   End
EndEngine

eor

# ------------------------------ second run --------------------------

AMS_JOBNAME=restart_1 $ADFBIN/ams <<eor

Task SinglePoint

System
   Lattice [Bohr]
      4.0 0.0 0.0
   End
   Atoms [Bohr]
      B 0.0 0.0 0.0
   End
End

Engine Band
   Title B chain restart

   NumericalQuality Good

   XC
      GGA Becke Perdew
   END

   UNRESTRICTED

   Restart
      File BChain.results/band.rkf
      scf
   end

   Basis
      Type TZ2P
      Core Large
   End
EndEngine

eor

# ------------------------------ third run --------------------------

AMS_JOBNAME=BChain_SZ $ADFBIN/ams <<eor

Task SinglePoint

System
   Lattice [Bohr]
      4.0 0.0 0.0
```

```
   End
   Atoms [Bohr]
      B 0.0 0.0 0.0
   End
End

Engine Band
   Title B chain bas_SZ

   NumericalQuality Good

   Save DensityMatrix

   skip dos

   XC
      GGA Becke Perdew
   END

   UNRESTRICTED

   DIIS
      NCycleDamp 0
      DiMix 0.3
      Adaptable false ! Otherwise it converges to a spin-restricted solution
   End

   Basis
      Type SZ
      Core Large
   End
EndEngine

eor

# ---------------------------- fourth run -------------------------


AMS_JOBNAME=restart_2 $ADFBIN/ams <<eor

Task SinglePoint

System
   Lattice [Bohr]
      4.0 0.0 0.0
   End
   Atoms [Bohr]
      B 0.0 0.0 0.0
   End
End

Engine Band
   Title B chain restart bas_SZ from density matrix

   NumericalQuality Good

   XC
      GGA Becke Perdew
```

```
   END

   UNRESTRICTED

   Restart
      File BChain_SZ.results/band.rkf
      scf
      useDensityMatrix true
   end

   Basis
      Type SZ
      Core Large
   End
EndEngine
eor


# ---------------------------- fifth run --------------------------

AMS_JOBNAME=BChain_TZ2P $ADFBIN/ams <<eor

Task SinglePoint

System
   Lattice [Bohr]
      4.0 0.0 0.0
   End
   Atoms [Bohr]
      B 0.0 0.0 0.0
   End
End

Engine Band
   Title B chain restart bas=TZ2P from orbitals

   NumericalQuality Good

   XC
      GGA Becke Perdew
   END

   UNRESTRICTED

   Restart
      File BChain_SZ.results/band.rkf
      scf
      useDensityMatrix false
   end

   Basis
      Type TZ2P
      Core Large
   End
EndEngine

eor
```

```
# ---------------------------- sixth run -------------------------

$ADFBIN/ams <<eor

Task SinglePoint

System
   Lattice [Bohr]
      4.0 0.0 0.0
   End
   Atoms [Bohr]
      B 0.0 0.0 0.0
   End
End

Engine Band
   Title B chain restart bas=TZ2P from density matrix (bas_SZ)

   NumericalQuality Good

   XC
      GGA Becke Perdew
   END

   UNRESTRICTED

   Restart
      File BChain_SZ.results/band.rkf
      scf
      useDensityMatrix true
   end

   Basis
      Type TZ2P
      Core Large
   End
EndEngine
eor
```

10.4.2 Example: Restart SCF for properties calculation

Download RestartProperties.run

```
#!/bin/sh

# This example shows how to restart the SCF and compute various properties, like
# a density of states, and a band structure plot, or the effective mass.

# ======================
# polyethylene .xyz file:
# ======================

cat <<eor > polyethylene.xyz
6

C      -0.623348981    -0.055000000     0.425969423
```

```
C       0.633348981     0.015000000    -0.422636089
H      -0.633348981     0.964974570     1.055290696
H      -0.623348981    -0.914974570     1.055290696
H       0.633348981     0.904974570    -1.051957363
H       0.613348981    -0.914974570    -1.061957363
VEC1    2.553395923     0.000000000     0.000000000
eor

# ================================================
# Simple single point calculation (no properties)
# ================================================

AMS_JOBNAME=ToBeRestarted $ADFBIN/ams <<eor

Task SinglePoint

System
  GeometryFile polyethylene.xyz
End

Engine Band
  Unrestricted True
EndEngine
eor

# ===================================
# Restart and compute some properties
# ===================================

AMS_JOBNAME=prop $ADFBIN/ams <<eor

Task SinglePoint

System
  GeometryFile polyethylene.xyz
End

Engine Band
  Unrestricted True

  Restart
    SCF
    File ToBeRestarted.results/band.rkf
  End

  DOS
    Enabled True
  End

  BandStructure
    Enabled True
    DeltaK 0.3
    EnergyAboveFermi 10.0
  End

  EffectiveMass
    Enabled True
  End
```

```
EndEngine
eor


echo 'Extract some properties from the rkf file:'

echo "Density of States:"
$ADFBIN/adfreport prop.results/band.rkf -r 'DOS%Total DOS##1'

echo "Band curve:"
$ADFBIN/adfreport prop.results/band.rkf -r 'band_curves%Edge_1_bands##1'

echo "Fab bands:"
$ADFBIN/adfreport prop.results/band.rkf -r 'band_curves%Edge_1_fatBands##1'

echo "Effective Mass:"
$ADFBIN/adfreport prop.results/band.rkf -r 'EffectiveMass%EffectiveMasses##1'

echo 'Done extracting properties'
```

10.4.3 Example: Properties on a grid

Download BeO_tape41.run

```
#!/bin/sh

# Saving the RUNKF file of a calculation gives rise to the opportunity to
# restart from it to calculate properties on a grid, like densities, potentials,
# or crystal orbitals. Find more details in the user documentation (Restarts).

# Regarding the following example, in the first run we perform a single-point
# calculation for a bulk BeO system. After the calculation finished the RUNKF
# file shall be renamed to BeO.kf. In the second run we restart from this
# file. We specify to use a regular grid and ask the program to calculate a
# bunch of properties on that grid.

# == First Job: ==

AMS_JOBNAME=First $ADFBIN/ams <<eor

Task SinglePoint

System
  FractionalCoords True

  Atoms
   Be  0.              0.              0.
   Be  0.333333333333 0.333333333333 0.5
   O   0.              0.              0.375
   O   0.333333333333 0.333333333333 0.875
  END

  Lattice [Bohr]
    5.10 0              0
    2.55 4.416729559300 0
    0    0              8.328265125462
```

```
  End
End

Engine Band
  Title BeO

  NumericalQuality Basic

  xc
    GGA BP86
  end

  Basis
    Type DZ
    Core large
  end
EndEngine
eor

# == Second Job: ==

AMS_JOBNAME=Second $ADFBIN/ams <<eor

Task SinglePoint

System
  FractionalCoords True

  Atoms
   Be  0.              0.              0.
   Be  0.333333333333 0.333333333333 0.5
   O   0.              0.              0.375
   O   0.333333333333 0.333333333333 0.875
  END

  Lattice [Bohr]
    5.10 0              0
    2.55 4.416729559300 0
    0    0              8.328265125462
  End
End

Engine Band
  Title BeO_restart

  Restart
    File First.results/band.rkf
    DensityPlot
  End

  Grid
     Type Coarse
  End

  DensityPlot
     rho(deformation/fit) ! FITDENSITY_deformation_scf
     rho(fit)             ! FITDENSITY_total_scf
     rho(atoms)           ! ATOMIC_density
```

```
     v(coulomb/atoms)      ! ATOMIC_coulombPot
     v(coulomb)            ! COULOMBPOTENTIAL_scf
     vxc[rho(fit)]         ! XCPOTENTIAL_scf
  End

  NumericalQuality Basic

  xc
    GGA BP86
  end

  Basis
    Type DZ
    Core large
  end
EndEngine
eor


NSCM=1
export NSCM
echo ""
echo "Begin TOC of tape41"

$ADFBIN/dmpkf -n 1 Second.results/TAPE41 --toc

echo "End TOC of tape41"
```

10.5 NEGF

10.5.1 Example: Main NEGF flavors

Download NEGF_Cr_wire.run

```
#! /bin/sh

# This example shows how to use the NEGF functionality.
# Note: Setting up a NEGF calculation is quite hard without the GUI.

# It starts of with Method 1: the non-self consistent approach. Here, BAND
# merely serves to provide matrix elements, being unaware of the electrodes.

# Then follows Method 2: here the NEGF density is really used to calculate the
# matrix elements.

# Method 3 is a variation on Method 2, and includes an extra alignment run.

# ============================================================================
#  Method #1: non-self consistent NEGF (uses the conductance program, like DFTB)
# ============================================================================


# ========================   Method #1. Run #1   ==============================

AMS_JOBNAME=lead_1 $ADFBIN/ams <<eor
```

```
Task SinglePoint

System
  ATOMS
     Cr.1 0.0 0.0 0.0
     Cr.2 2.5 0.0 0.0
  END

  Lattice
     5.0 0.0 0.0
  End
End

Engine Band
  Title method_1_run_1

  KSpace
      Quality Good
  End

  NumericalQuality Basic

  Basis
     Type DZ
     Core Large
  End

  Unrestricted

  StoreHamiltonian2
EndEngine
eor

# ========================   Method #1. Run #2   ================================

AMS_JOBNAME=scattering_1 $ADFBIN/ams <<eor

Task SinglePoint

System
  ATOMS
      Cr.1L -10.0 0.0 0.0
      Cr.2L  -7.5 0.0 0.0
      Cr.C   -5.0 0.0 0.0
      Cr.C   -2.5 0.0 0.0
      Cr.C    0.0 0.0 0.0
      Cr.C    2.5 0.0 0.0
      Cr.C    5.0 0.0 0.0
      Cr.1R   7.5 0.0 0.0
      Cr.2R  10.0 0.0 0.0
  END

  Lattice
     22.5 0.0 0.0
  End
End
```

```
Engine Band
  Title method_1_run_2

  NumericalQuality Basic

  Basis
     Type DZ
     Core Large
  End

  Unrestricted

  StoreHamiltonian2
  StoreHamAsMol
EndEngine

eor

# ==========================   Method #1. Run #3   ==============================

$ADFBIN/conductance <<EOF
EnergyGrid min=-5 max=5 num=200
Files
   Leads      lead_1.results/band.rkf
   Scattering scattering_1.results/band.rkf
End

EOF

# Compy the content of the "reseults" section from ConductanceResults.kf to band.rkf␣
↪and rename the section to NEGF
$ADFBIN/cpkf "ConductanceResults.kf" "scattering_1.results/band.rkf" "results --
↪rename NEGF"


echo "Extract transmisstion from rkf file (Method 1)"
$ADFBIN/adfreport scattering_1.results/band.rkf -r "NEGF%transmission#12.5f##1"

# ============================================================================
#               Method #2: self consistent NEGF without alignment
# ============================================================================


# ==========================   Method #2. Run #1   ==============================

AMS_JOBNAME=lead_2 $ADFBIN/ams <<eor

Task SinglePoint

System
  ATOMS
      Cr.1 0.0 0.0 0.0
      Cr.2 2.5 0.0 0.0
  END

  Lattice
   5.0 0.0 0.0
  End
```

```
End

Engine Band
  Title method_2_run_1

  KSpace
      Quality Good
  End

  NumericalQuality Basic

  Basis
    Type DZ
    Core Large
  End

  Unrestricted
  StoreHamiltonian2
EndEngine

eor

# ============================   Method #2. Run #2   ========================


$ADFBIN/sgf   <<eor
TITLE Test for NEGF inputs
SAVE SIGMA
SURFACEGF
   RKFFileName lead_2.results/band.rkf
   SCMCode
   KT 0.001
   ContourQuality normal
END
eor

mv SigmaSCM Sigma.kf

# ============================   Method #2. Run #3   ========================

AMS_JOBNAME=scattering_2 $ADFBIN/ams <<eor

Task SinglePoint

System
  ATOMS
      Cr.1L -10.0 0.0 0.0
      Cr.2L -7.5 0.0 0.0
      Cr.C -5.0 0.0 0.0
      Cr.C -2.5 0.0 0.0
      Cr.C 0.0 0.0 0.0
      Cr.C 2.5 0.0 0.0
      Cr.C 5.0 0.0 0.0
      Cr.1R 7.5 0.0 0.0
      Cr.2R 10.0 0.0 0.0
  END
End
```

```
Engine Band
  Title method_2_run_3

  NumericalQuality Basic

  Basis
    Type DZ
    Core Large
  End

  Unrestricted
  NEGF
    LeadFile        lead_2.results/band.rkf
    SGFFile         Sigma.kf
    ApplyShift2     False
    ContourQuality  normal
    EMin           -5
    EMax            5
    NE 200
  End
EndEngine
eor

echo "Extract transmisstion from rkf file (Method 2)"
$ADFBIN/adfreport scattering_2.results/band.rkf -r "NEGF%transmission#12.5f##1"

# ============================================================================
#                Method #3: self consistent NEGF wit alignment run
# ============================================================================

# ======================   Method #3. Run #1   ================================


AMS_JOBNAME=lead_3 $ADFBIN/ams <<eor

Task SinglePoint

System
  ATOMS
      Cr.1 0.0 0.0 0.0
      Cr.2 2.5 0.0 0.0
  END
  Lattice
   5.0 0.0 0.0
  End
End

Engine Band
  Title method_3_run_1

  KSpace
      Quality Good
  End

  NumericalQuality Basic

  Basis
    Type DZ
```

```
    Core Large
  End

  Unrestricted
  StoreHamiltonian2
EndEngine
eor

# ============================   Method #3. Run #2   ============================


$ADFBIN/sgf   <<eor
TITLE Test for NEGF inputs
SAVE SIGMA
SURFACEGF
   RKFFileName lead_3.results/band.rkf
   SCMCode
   KT 0.001
   ContourQuality normal
END
eor

mv SigmaSCM Sigma.kf

# ============================   Method #3. Run #3   ============================

AMS_JOBNAME=align $ADFBIN/ams <<eor

Task SinglePoint

System
  ATOMS
      Cr.1L 0.0 0.0 0.0
      Cr.2L 2.5 0.0 0.0
      Cr.C 5.0 0.0 0.0
      Cr.C 7.5 0.0 0.0
      Cr.C 10.0 0.0 0.0
      Cr.C 12.5 0.0 0.0
      Cr.1R 15.0 0.0 0.0
      Cr.2R 17.5 0.0 0.0
  END
End

Engine Band
  Title method_3_run_3

  NumericalQuality Basic

  Basis
    Type DZ
    Core Large
  End

  Unrestricted
  NEGF
    DoAlignment True
    LeadFile lead_3.results/band.rkf
    SGFFile Sigma.kf
```

```
    ContourQuality normal
    EMin -5.0
    EMax 5.0
    NE 200
    AlignChargeTol 0.0001
  End
EndEngine
eor

# ============================   Method #3. Run #4  ==========================

AMS_JOBNAME=scattering_3 $ADFBIN/ams <<eor

Task SinglePoint

System
  ATOMS
      Cr.1L -10.0 0.0 0.0
      Cr.2L -7.5 0.0 0.0
      Cr.C -5.0 0.0 0.0
      Cr.C -2.5 0.0 0.0
      Cr.C 0.0 0.0 0.0
      Cr.C 2.5 0.0 0.0
      Cr.C 5.0 0.0 0.0
      Cr.1R 7.5 0.0 0.0
      Cr.2R 10.0 0.0 0.0
  END
End

engine Band
  Title method_3_run_4

  NumericalQuality Basic

  Basis
    Type DZ
    Core Large
  End

  Unrestricted

  NEGF
     LeadFile lead_3.results/band.rkf
     SGFFile Sigma.kf
     AlignmentFile align.results/band.rkf
     ContourQuality normal
     EMin -5.0
     EMax 5.0
     NE 200
  End
EndEngine
eor

echo "Extract transmisstion from rkf file (Method 2)"
$ADFBIN/adfreport scattering_3.results/band.rkf -r "NEGF%transmission#12.5f##1"
```

10.5.2 Example: NEGF with bias

Download NEGF_bias.run

```
#! /bin/sh

# This example shows how to use the NEGF key when including a bias potential
# between the electrodes. It starts of with the usual tight-binding run,
# followed by an SGF one. The alignment run is omitted. Finally, there is a loop
# over bias potentials. Here the scale feature of the FuzzyPotential is used.
# The current is appended to a text file, which one could plot eg. with gnuplot.

# Note: Setting up a NEGF calculation is quite hard without the GUI.

AMS_JOBNAME=tight-binding $ADFBIN/ams <<eor

Task SinglePoint

System
  Atoms
      Li.1 0.0 0.0 0.0
      Li.2 2.876 0.0 0.0
      Li.3 5.752 0.0 0.0
  End
  Lattice
   8.628 0.0 0.0
  End
End

Engine Band
  TITLE tight-binding

  KSpace
    Quality VeryGood
  End

  SoftConfinement
    Quality Basic
  End

  Basis
    Type DZ
    Core Large
  End

  StoreHamiltonian2
EndEngine
eor


$ADFBIN/sgf   <<eor
TITLE Test for NEGF inputs
SAVE SIGMA
SURFACEGF
   RKFFileName tight-binding.results/band.rkf
   SCMCode
   KT 0.001
   ContourQuality normal
```

```
END
eor

mv SigmaSCM Sigma.kf

REPORT=Li-CuAg.report
touch $REPORT

for bias in -0.01 0.01
do


AMS_JOBNAME=negf $ADFBIN/ams <<eor

Task SinglePoint

System
  ATOMS
       Li.1L -15.818 0.0 0.0
       Li.2L -12.942 0.0 0.0
       Li.3L -10.066 0.0 0.0
       Li.1C -7.19 0.0 0.0
       Li.2C -4.314 0.0 0.0
       Cu.C -0.7 -1.0 0
       Ag.C 0.7 1.0 0
       Li.3C 4.314 0.0 0.0
       Li.4C 7.19 0.0 0.0
       Li.1R 10.066 0.0 0.0
       Li.2R 12.942 0.0 0.0
       Li.3R 15.818 0.0 0.0
  END
End

Engine Band
  TITLE bias=$bias

  SoftConfinement
    Quality Basic
  End

  Basis
    Type DZ
    Core Large
  End

  NEGF
    LeadFile tight-binding.results/band.rkf
    SGFFile Sigma.kf
    EMin -5.0
    EMax 5.0
    NE 200
    ApplyShift2 False
    BiasPotential $bias
  End

  FuzzyPotential
  scale $bias
  1   0.5
```

```
  2    0.5
  3    0.5
  4    0.5
  5    0.5
  6    0.2
  7   -0.2
  8   -0.5
  9   -0.5
  10  -0.5
  11  -0.5
  12  -0.5
  end
EndEngine

eor

current=`$ADFBIN/adfreport negf.results/band.rkf 'NEGF%current'`
echo "NEGFREPORT: Bias=$bias, Current=$current" >> $REPORT

fname=`ls Transmission_*.plt`

echo "start of transmission (bias=$bias)"
cat $fname
echo "end of transmission"

rm Transmission_*.plt

rm -r negf.results

done

echo "Start of report"
cat $REPORT
echo "End of report"
```

10.5.3 Example: NEGF using the non-self consistent method

Download NEGF_Conductance.run

```
#! /bin/sh

# In this example we demonstrate how to run a Band-NEGF calculation using the non
# self consistent approach (using the conductance program). In the first example
# we study the conductivity of a mono-atomic gold chain with a CO molecule
# adsorbed on top. Such calculation consists of three separate runs. See the
# documentation for more details.

# ============================================================================
#                              CO on gold chain
# ============================================================================

# =======
# Au lead
# =======

AMS_JOBNAME=Au_lead $ADFBIN/ams <<eor
```

```
Task SinglePoint

System
    ATOMS
        Au.1 0.0       0.0  0.0
        Au.2 2.884996  0.0  0.0
        Au.3 5.769992  0.0  0.0
    END

    Lattice
        8.654988  0.0  0.0
    End
End

Engine Band
    TITLE Au_lead

    KSpace
        Quality VeryGood
    End

    SoftConfinement
        Quality Basic
    End

    Relativity
        Level Scalar
    End

    Basis
        Type DZ
        Core Large
    End

    StoreHamiltonian2
EndEngine
eor

# =============
# Au scattering
# =============

AMS_JOBNAME=Au_scattering $ADFBIN/ams <<eor

Task SinglePoint

System
    ATOMS
        Au.1L -20.194972  0.0  0.0
        Au.2L -17.309976  0.0  0.0
        Au.3L -14.42498   0.0  0.0
        Au.C  -11.539984  0.0  0.0
        Au.C  -8.654988   0.0  0.0
        Au.C  -5.769992   0.0  0.0
        Au.C  -2.884996   0.0  0.0
        Au.C   0.0        0.0  0.20
        Au.C   2.884996   0.0  0.0
```

```
        Au.C   5.769992   0.0  0.0
        Au.C   8.654988   0.0  0.0
        Au.C   11.539984  0.0  0.0
        O.C    0.0        0.0  3.12
        C.C    0.0        0.0  1.96
        Au.1R  14.42498   0.0  0.0
        Au.2R  17.309976  0.0  0.0
        Au.3R  20.194972  0.0  0.0
    END

    Lattice
        43.27494  0.0  0.0
    End
End

Engine Band
    TITLE Au_scattering

    SoftConfinement
        Quality Basic
    End

    Relativity
        Level Scalar
    End

    Basis
        Type DZ
        Core Large
    End

    StoreHamiltonian2
    StoreHamAsMol
EndEngine

eor

# ==============
# Au Conductance
# ==============

$ADFBIN/conductance <<EOF
EnergyGrid min=-3.5 max=3 num=200
Files
  Leads         Au_lead.results/band.rkf
  Scattering    Au_scattering.results/band.rkf
End
EOF

mv ConductanceResults.kf Au_ConductanceResults.kf


echo "Extract DOS from the kf file (AuCO):"
$ADFBIN/adfreport Au_ConductanceResults.kf -r "results%dos#12.5f##1"

echo "Extract the transmission from the kf file (AuCO):"
$ADFBIN/adfreport Au_ConductanceResults.kf -r "results%transmission#12.5f##1"
```

```
# ============================================================================
#                          Spin-unrestricted Cr chain
# ============================================================================

# =======
# Cr Lead
# =======

AMS_JOBNAME=Cr_lead $ADFBIN/ams <<eor

Task SinglePoint

System
    ATOMS
        Cr.1 1.18995235  0.0  0.0
        Cr.2 4.00745359  0.0  0.0
        Cr.3 6.82495483  0.0  0.0
    END

    Lattice
        8.45250372  0.0  0.0
    End
End

Engine Band
    TITLE Cr_lead

    KSpace
        Quality VeryGood
    End

    SoftConfinement
        Quality Basic
    End

    Relativity
        Level Scalar
    End

    Basis
        Type DZ
        Core Large
    End

    UNRESTRICTED

    StoreHamiltonian2
EndEngine

eor

# =============
# Cr Scattering
# =============

AMS_JOBNAME=Cr_scattering $ADFBIN/ams <<eor
```

```
Task SinglePoint

System
    ATOMS
        Cr.1L -10.08005261  0.0  0.0
        Cr.2L  -7.26255137  0.0  0.0
        Cr.3L  -4.44505013  0.0  0.0
        Cr.C   -1.62754889  0.0  0.0
        Cr.C    1.18995235  0.0  0.0
        Cr.C    4.00745359  0.0  0.0
        Cr.1R   6.82495483  0.0  0.0
        Cr.2R   9.64245607  0.0  0.0
        Cr.3R  12.45995731  0.0  0.0
    END

    Lattice
        25.35751116  0.0  0.0
    End
End

Engine Band
    TITLE Cr_scattering

    KSpace
        Quality Good
    End

    SoftConfinement
        Quality Basic
    End

    Relativity
        Level Scalar
    End

    Basis
        Type DZ
        Core Large
    End

    UNRESTRICTED
    StoreHamiltonian2
    StoreHamAsMol
EndEngine

eor

# ==============
# Cr Conductance
# ==============

$ADFBIN/conductance <<EOF
EnergyGrid min=-4 max=4 num=200
Files
  Leads      Cr_lead.results/band.rkf
  Scattering Cr_scattering.results/band.rkf
End
```

```
EOF

mv ConductanceResults.kf Cr_ConductanceResults.kf

echo "Extract DOS from the kf file (Cr):"
$ADFBIN/adfreport Cr_ConductanceResults.kf -r "results%dos#12.5f##1"

echo "Extract the transmission from the kf file (Cr):"
$ADFBIN/adfreport Cr_ConductanceResults.kf -r "results%transmission#12.5f##1"
```

10.6 Structure and Reactivity

10.6.1 Example: NaCl: Bulk Crystal

Download NaCl.run

```
#! /bin/sh

# A bulk crystal computation for Sodium Chloride (common salt), with a
# subsequent DOS analysis, using a Restart facility to use the results from a
# preceding calculation.

# The BAND input follows slightly different conventions from the ADF input, for
# historical reasons.

# Since there are 3 data records in the Lattice block, the calculation will
# assume 3-dimensional periodicity, with lattice vectors as indicated. Note that
# lattice vectors are undefined up to linear combinations among themselves.
# Internally, the program will recombine the input vectors so as to minimize the
# size of the actually used vectors.

# The input line FractionalCoords True means that atomic positions are input as
# coefficients in terms of the lattice vectors, rather than as absolute
# (Cartesian) coordinate values.

AMS_JOBNAME=NaCl $ADFBIN/ams <<eor

Task SinglePoint

System
  FractionalCoords True

  Atoms
    Na 0.0 0.0 0.0
    Cl 0.5 0.5 0.5
  End

  Lattice
    0.0   2.75   2.75
    2.75  0.0    2.75
    2.75  2.75   0.0
  End
End

Engine Band
```

```
  Title NaCl

  Kspace
    Symmetric KInteg=3
  End

  Basis
    Type SZ
    Core None
  End

  Print AtomicChargesDetails

EndEngine

eor

# The next run has largely the same input and provides a restart of the previous
# run.

# The key DOS in the block Restart tells the program to pick up the indicated
# file as restart file and to use it for DOS analysis purposes.

# The DOS key block details the energy grid (and range) and the file to write
# the data to. The optional keys GROSSPOPULATIONS and OverlapPopulations invoke
# the computation of, respectively, gross populations and overlap populations
# (i.e. for each of these the density-of-states values in the user-defined
# energy grid).

AMS_JOBNAME=NaCl-restart $ADFBIN/ams <<eor

Task SinglePoint

LoadSystem
  File NaCl.results/ams.rkf
  Section InputMolecule
End

Engine Band
  Title NaCl DOS analysis (restart)

  Kspace
    Symmetric KInteg=3
  End

  Basis
    Type SZ
    Core None
  End

  Restart
    File NaCl.results/band.rkf
    SCF
  End

  DOS
    Enabled   True
    File      NaCl.dos
```

```
    Energies  1000
    Min      -0.5
    Max       0.5
  End

  GrossPopulations
   FRAG 1
   FRAG 2
   SUM
     1 0
     2 0
   ENDSUM
  End

  OverlapPopulations
  Left
   FRAG 1
  Right
   FRAG 2
  Left
    1 0
    1 1
  Right
    2 0
    2 1
  End

  Print AtomicChargesDetails
EndEngine
eor

# Finally, we copy the contents of the DOS result file to standard output

echo ""
echo Contents of DOS file
cat NaCl.dos
```

10.6.2 Example: Transition-State search using initial Hessian

Download COChainFreqTS.run

```
#! /bin/sh

# This example demonstrates in the first step how to calculate the Hessian.
# The second run uses the pre-calculated Hessian and performs a transition
# state search along the frequency mode with the smallest frequency.


# First run: Calculate Hessian
# ============================

AMS_JOBNAME=hessian $ADFBIN/ams << EOF

   Task SinglePoint

   Properties
```

```
      Hessian True
   End

   System
      Atoms
         C  0.0  0.0  0.0
         O  1.5  0.0  0.0
      End
      Lattice
         3.2  0.0  0.0
      End
   End

   Engine Band
      Basis Type=DZP
      KSpace Quality=Good
   EndEngine

EOF


# Second run: TS search with initial Hessian
# ==========================================

AMS_JOBNAME=TS $ADFBIN/ams << EOF

   Task TransitionStateSearch

   System
      Atoms
         C  0.0  0.0  0.0
         O  1.5  0.0  0.0
      End
      Lattice
         3.2  0.0  0.0
      End
   End

   GeometryOptimization
      Convergence Gradients=1.0e-4
      InitialHessian
         # Load the pre-calculated Hessian as the initial Hessian for the
         # transition state search using the Quasi-Newton based optimizer.
         Type FromFile
         File hessian.results/band.rkf
      End
   End

   Properties
      # Also calculate normal modes in the end, so we can see if we actually
      # found a transition state.
      NormalModes True
   End

   Engine Band
      Basis Type=DZP
      KSpace Quality=Good
   EndEngine
```

```
EOF
```

10.6.3 Example: Atomic energies

Download H_ref.run

```
#! /bin/sh

# This example consists of several atomic energy calculations:

# - Formation energy of the H-atom w.r.t. spherical atom
# - Formation energy of the H-atom w.r.t. spherical atom
# - Spin polarization energy of the H-atom w.r.t. spherical atom
# - Spin polarization (relativistic) energy of the H-atom w.r.t. spherical atom
# - Spin polarization energy of the H-atom w.r.t. spin unrestricted atom
# - Spin polarization (relativistic) energy of the H-atom w.r.t. spin
#   unrestricted atom

# XYZ file of H atom with large 2 lattice
cat << eor > H.xyz
1

H 0.0 0.0 0.0
VEC1 10.583544212 0.0          0.0
VEC2 0.0          10.583544212 0.0
eor


$ADFBIN/ams <<eor

Task SinglePoint

System
   GeometryFile H.xyz
End

Engine Band
   Title Formation energy of the H-atom w.r.t. spherical atom

   Print AtomicChargesDetails

   Kspace
      Symmetric KInteg=5
   End
   Integration
      Accint 5.0
   End

   Convergence
      Criterion 1E-6
   End

   AtomType H
      Dirac H
        1  0
```

```
      VALENCE
         1S 1
      SubEnd

      BasisFunctions
         1S   1.58
         2P   1.0
      SubEnd

      FitFunctions
      SubEnd
   End
EndEngine
eor

rm -r ams.results

$ADFBIN/ams <<eor

Task SinglePoint

System
   GeometryFile H.xyz
End

Engine Band
   Title Spin polarization energy of the H-atom w.r.t. spherical atom

   Print AtomicChargesDetails

   Kspace
      Symmetric KInteg=5
   End
   Integration
      Accint 5.0
   End

   Convergence
      Criterion 1E-6
   End

   Unrestricted

   AtomType H
      Dirac H
        1  0
      VALENCE
         1S 1
      SubEnd

      BasisFunctions
         1S   1.58
         2P   1.0
      SubEnd

      FitFunctions
      SubEnd
   End
```

```
EndEngine
eor

rm -r ams.results


$ADFBIN/ams <<eor

Task SinglePoint

System
   GeometryFile H.xyz
End

Engine Band
   Title Spin polarization (relativistic) energy of the H-atom w.r.t. spherical atom

   Print AtomicChargesDetails

   Kspace
      Symmetric KInteg=5
   End
   Integration
      Accint 5.0
   End

   Convergence
      Criterion 1E-6
   End

   Unrestricted

   Relativity
      Level Scalar
   End

   AtomType H
      Dirac H
        1  0
      VALENCE
         1S 1
      SubEnd

      BasisFunctions
         1S   1.58
         2P   1.0
      SubEnd

      FitFunctions
      SubEnd
   End
EndEngine
eor

rm -r ams.results


$ADFBIN/ams <<eor
```

```
Task SinglePoint

System
   GeometryFile H.xyz
End

Engine Band
   Title Spin polarization energy of the H-atom w.r.t. spin unrestricted atom

   Print AtomicChargesDetails

   Kspace
      Symmetric KInteg=5
   End
   Integration
      Accint 5.0
   End

   Convergence
      Criterion 1E-6
   End

   Unrestricted
   UnrestrictedReference

   AtomType H
      Dirac H
        1  0
      VALENCE
         1S 1
      SubEnd

      BasisFunctions
         1S   1.58
         2P   1.0
      SubEnd

      FitFunctions
      SubEnd
   End
EndEngine
eor

rm -r ams.results


$ADFBIN/ams <<eor

Task SinglePoint

System
   GeometryFile H.xyz
End

Engine Band
   Title Spin polarization (relativistic) energy of the H-atom w.r.t. spin␣
↪unrestricted atom
```

```
   Print AtomicChargesDetails

   Kspace
      Symmetric KInteg=5
   End
   Integration
      Accint 5.0
   End

   Convergence
      Criterion 1E-6
   End

   UnrestrictedReference
   Unrestricted

   Relativity
      Level Scalar
   End

   AtomType H
      Dirac H
        1  0
      VALENCE
         1S 1
      SubEnd

      BasisFunctions
         1S   1.58
         2P   1.0
      SubEnd

      FitFunctions
      SubEnd
   End
EndEngine
eor
```

10.6.4 Example: Calculating the atomic forces

Download BNForce.run

```
#!/bin/sh

# This example shows how to calculate the gradient of the energy with respect to
# atomic displacements, using the GRADIENTS key block. For more details on
# consult the documentation for Gradients.

$ADFBIN/ams <<eor

Task SinglePoint

System
   Atoms
    B   0.0 0.0 0.0
```

```
    N   0.86544 0.86544 0.86544
   end
   Lattice
     0.0 1.8 1.8
     1.8 0.0 1.8
     1.8 1.8 0.0
   End
End

Properties
  Gradients True
End

Engine Band
   Title BN zincblende structure (force calculation)

   NumericalQuality Basic  ! for speed, not very accurate

   Basis
      Type TZ2P
      Core Large
   End
EndEngine

eor

# In the output you will find the result
# FINAL GRADIENTS
#     1  B  0.017517  0.017517  0.017517
#     2  N -0.017517 -0.017517 -0.017517
```

10.6.5 Example: Optimizing the geometry

Download H2BulkGeo.run

```
#! /bin/sh

# This example shows how to optimize the geometry.

# This example consists of two runs. The first run performs 5 iterations
# regarding the geometry optimization. And the second run exploits the
# possibility to restart a geometry optimization based on the rkf of a
# previous, presumably non-converged run.

# ----------------------------- first run -------------------------

AMS_JOBNAME=First $ADFBIN/ams <<eor

Task GeometryOptimization

System
  ATOMS [Bohr]
    H   0.0 0.0 0.0
    H   1.0 0.0 0.0
  End
```

```
  Lattice [Bohr]
    5.0      0        0
    0        5.0      0
    0        0        5.0
  End
End

GeometryOptimization
  MaxIterations 5
  Convergence Gradients=1e-6
End

Engine Band
  Basis
    Type DZP
  End
EndEngine

eor


# In the next run we use the result file to continue the geometry optimization.

# ---------------------------- second run -------------------------

$ADFBIN/ams <<eor

Task GeometryOptimization

LoadSystem
  File First.results/ams.rkf
End

GeometryOptimization
  MaxIterations 5
  Convergence Gradients=1e-6
End

Engine Band
  Basis
     Type DZP
  End
EndEngine
eor


echo 'Extract optimized geometry from the rkf file'
$ADFBIN/adfreport ams.results/ams.rkf -r 'Molecule%Coords##3'

echo 'Extract number of steps from the rkf file'
$ADFBIN/adfreport ams.results/ams.rkf -r 'History%nEntries'
```

10.7 Time dependent DFT

10.7.1 Example: TD-CDFT for MoS2 Monolayer (NewResponse)

Download NewResp_2DMoS2Restart.run

```
#!/bin/sh

# This example demonstrates how to calculate the frequency-dependent dielectric
# function with the help of the NewResponse implementation for a two-dimensional
# system. (see NewResponse) Furthermore, the general setup to run the TD-CDFT
# section as a restart calculation is presented as well. This allows for
# splitting of the frequency range into several parts, which can then be
# calculated in separate calculation without the overhead of evaluating the
# groundstate properties for each of them! Hence, it is a trivial
# parallelization possibility.

# =========================
# MoS2 Monolayer .xyz file:
# =========================

cat << eor > MoS2_2D_1L.xyz
3

S          0.00000000       0.00300000      -7.76123300
S          0.00000000       0.00000000      -4.53876700
Mo         1.58000000       0.91221300      -6.15000000
VEC1       3.16000000       0.00000000       0.00000000
VEC2       1.58000000       2.73664028       0.00000000

eor

# ================================================
# Simple single point calculation (no properties)
# ================================================

AMS_JOBNAME=MoS2 $ADFBIN/ams <<eor

Task SinglePoint

System
  GeometryFile MoS2_2D_1L.xyz
End

Engine Band
  UseSymmetry False

  NumericalQuality good
  Relativity
    Level Scalar
  End

  DEPENDENCY BASIS=1e-10

  Tails bas=1e-10

  KSpace
```

```
    Regular
      NumberOfPoints 5 5
    End
  End

  Basis
    Type DZP
    Core Large
  End

  Convergence
   Criterion 1E-8
  End
EndEngine

eor

# ==================================
# Restart and compute some properties
# ==================================

# Caution!
# One has to make sure to use the same
# Symmetry/NumericalQuality/KSpace/Basis/ZORA/... options for the
# ground state calculation and for the restart calculation! Otherwise a normal
# ground state SCF optimization will be performed in the restart calculation.


AMS_JOBNAME=MoS2_restart $ADFBIN/ams <<eor

  Task SinglePoint

  System
    GeometryFile MoS2_2D_1L.xyz
  End

  Engine Band

    UseSymmetry False

    NumericalQuality good
    Relativity
      Level Scalar
    End

    DEPENDENCY BASIS=1e-10

    Tails bas=1e-10

    KSpace
      Regular
        NumberOfPoints 5 5
      End
    End

    Basis
      Type DZP
      Core Large
```

```
    End

    Convergence
     Criterion 1E-8
    End

    Restart
      File MoS2.results/band.rkf
      SCF
    End

    NewResponse
      nFreq        4
      FreqLow      2.0
      FreqHigh     2.7
      ActiveESpace 10.0
      ActiveXYZ    T T F
    End

    NewResponseSCF
      nCycle       50
      Criterion    1E-3
    End

    NewResponseKSPACE
      subsimp 10
      eta     1e-6
    End
  EndEngine

eor

# ============
# Extract info
# ============

$ADFBIN/adfreport MoS2_restart.results/band.rkf RESPDIELRE
$ADFBIN/adfreport MoS2_restart.results/band.rkf RESPDIELIM

# The results are accessible via the standard output or via the prop.kf file.
# For the latter, one can use the ADFreport command $ADFBIN/adfreport prop.kf
# RESPDIELRE and $ADFBIN/adfreport prop.kf RESPDIELIM to print the components
# of the dielectric function for the real (RESPDIELRE) and imaginary
# (RESPDIELIM) part separately. In the following tables, only the diagonal
# components are presented:

# Real part
# Frequency (au)  epsilon_1(XX)  epsilon_1(YY)  epsilon_1(ZZ)
# 0.0735           8.1622063      8.1788067      1.8845925
# 0.0772           8.7718566      8.7960299      1.8891231
# 0.0808           9.6251443      9.6631930      1.8941277
# 0.0845          10.9457271     11.0126367      1.8996502
# 0.0882          13.4618956     13.6001321      1.9057858
# 0.0919          26.5135344     25.9300685      1.9126665
# 0.0955           6.1134118      4.1756368      1.9204849
# 0.0992           6.2789015      4.6880515      1.9295347
# 0.1029          13.7665058     11.5484340      1.9403044
# 0.1066          -7.2575153     -5.8285172      1.9537079
```

```
# 0.1102          -0.7937277      1.2661253      1.9718981

# Imaginary part
# Frequency (au)  epsilon_2(XX) epsilon_2(YY)  epsilon_2(ZZ)
# 0.0735           0.0015601     0.0015758      0.0000213
# 0.0772           0.0020566     0.0020839      0.0000200
# 0.0808           0.0029274     0.0029798      0.0000216
# 0.0845           0.0047632     0.0048794      0.0000231
# 0.0882           0.0104743     0.0107877      0.0000246
# 0.0919           0.2658531     0.1942899      0.0000264
# 0.0955          12.8856772    14.5286319      0.0000294
# 0.0992          9.7571573     10.1567455      0.0000338
# 0.1029           7.5936072     6.7674596      0.0000399
# 0.1066           13.0264038    9.5897946      0.0000487
# 0.1102           0.2483041     0.3222301      0.0000676

# The more convenient option is to plot the spectral data directly with the help
# of ADFspectra. Just type: $ADFBIN/adfspectra prop.kf
```

10.7.2 Example: TD-CDFT for Copper (NewResponse)

Download NewResp_3DCopper.run

```
#! /bin/sh

$ADFBIN/ams <<eor

Task SinglePoint

System
   Lattice :: FCC
      0     1.805 1.805
      1.805 0     1.805
      1.805 1.805 0
   End

   Atoms
      Cu  0.00  0.00  0.00
   End
End

Engine Band
   Title NewResponse of Cu within ALDA

   NumericalQuality basic

   KSpace
     Regular
       NumberOfPoints 5 5 5
     End
   End

   NewResponse
      nfreq         10
      freqLow       0.1
      freqHigh      10.0
```

```
      activeEspace  10
   END

   NewResponseSCF
      Criterion       0.1
      LowFreqAlgo     true
      COApproach      true
      COApproachBoost true
   End

   NewResponseKSPACE
      subsimp       5
   End

   Basis
      Type TZ2P
      Core Large
   End
EndEngine

eor
```

10.7.3 Example: TDCDFT: Plot induced density (NewResponse)

Download NewResp_PlotInducedDensity.run

```
#!/bin/sh

AMS_JOBNAME=polyethylene $ADFBIN/ams <<eor

  Task SinglePoint

  System
    Lattice
      2.553395923     0.000000000     0.000000000
    end

    Atoms
       C      -0.623348981    -0.055000000     0.425969423
       C       0.633348981     0.015000000    -0.422636089
       H      -0.633348981     0.964974570     1.055290696
       H      -0.623348981    -0.914974570     1.055290696
       H       0.633348981     0.904974570    -1.051957363
       H       0.613348981    -0.914974570    -1.061957363
    end
  End

  Engine Band

    Title Polyethylene

    KSPACE
       Regular
        NumberOfPoints 11
      End
    End
```

```
    NumericalQuality basic

    DEPENDENCY BASIS=1e-10

    Tails bas=1e-10

    NEWRESPONSE
       nFreq        10
       FreqLow      6.0
       FreqHigh     8.0
       ActiveXYZ    T F F
       ActiveESpace 2.0
    END

    NEWRESPONSESCF
       nCycle     50
       Mixing     0.075
       Criterion  0.01
    End

    Basis
      Type  TZP
      Core       small
    End
  EndEngine
eor

# ======================================
# Restart and compute Induced Densities
# ======================================

export NSCM=1
$ADFBIN/ams -n 1 <<EOF

  Task SinglePoint

  LoadSystem
    File polyethylene.results/ams.rkf
    Section InputMolecule
  End

  Engine Band
    Title Polyethylene Plot Induced Response Density

    UseSymmetry False

    NumericalQuality basic

    DEPENDENCY BASIS=1e-10

    Tails bas=1e-10

    KSpace
      Regular
        NumberOfPoints 11
      End
    End
```

```
    Basis
      Type TZP
      Core Small
    End

    Restart
      File polyethylene.results/band.rkf
      ResponseInducedDensityPlot
    End

    ResponseInducedDensityPlot
     xcomponent 1 2
     xcomponent 5
    End

    Grid
    End

    debug BlockPropertyModule
  EndEngine
EOF

echo ""
echo "Begin TOC of tape41"
export NSCM=1
$ADFBIN/pkf -n 1 ams.results/FILE_BLOCKPROPERTIES
echo "End TOC of tape41"
```

10.7.4 Example: TD-CDFT for bulk diamond (OldResponse)

Download OldResp_Diamond.run

```
#! /bin/sh

# Response calculation for diamond

$ADFBIN/ams <<eor

Task SinglePoint

System
   LATTICE
      0      1.785  1.785
      1.785  0      1.785
      1.785  1.785  0
   END
   ATOMS
      C  0.0   0.0   0.0
      C  0.8925  0.8925  0.8925
   END
End

Engine Band
   TITLE DIAMOND
```

```
   Integration
      Accint 5
   End

   KSPACE
      Symmetric KInteg=2
   End

   Dependency  Basis=1.e-6

   OLDRESPONSE
      Enabled  True
      nfreq    7
      strtfr   0.0
      endfr    19.0480
   END

   Basis
      Type DZ
   End
EndEngine

eor
```

10.8 Spectroscopy

10.8.1 Example: Hyperfine A-tensor

Download TiF3a.run

```
#!/bin/sh

# Example for an ESR A-tensor calculation.

# Be aware that the calculation must be spin unrestricted and the ATENSOR
# keyword must be present, too.

$ADFBIN/ams <<eor

Task SinglePoint

System
   Atoms
      Ti  0.0     0.0               0.0
      F   1.78    0.0               0.0
      F  -0.89    1.541525218736    0.0
      F  -0.89   -1.541525218736    0.0
   end
End

Engine Band
   Title TiF3

   Unrestricted True
```

```
   EFG
      Enabled True
   End

   ATensor
      Enabled True
   End

   Basis
      Type TZP
      Core None
   End
EndEngine
eor
```

10.8.2 Example: Zeeman g-tensor

Download TiF3g.run

```
#!/bin/sh

# Example for an ESR g-tensor calculation. More information in the documentation
# of ESR

# Be aware that this calculation must include the spin-orbit, relativistic
# approximation (Relativistic ZORA Spin)!

$ADFBIN/ams <<eor

Task SinglePoint

System
   Atoms
      Ti  0.0               0.0   0.0
      F   0.0               1.78  0.0
      F  -1.541525218736  -0.89  0.0
      F   1.541525218736  -0.89  0.0
   end
End

Engine Band
   Title TiF3

   Relativity
      Level Spin-Orbit
   End

   ESR
      Enabled True
   end

   Basis
      Type DZ
      Core None
   End
EndEngine
```

```
eor
```

10.8.3 Example: NMR

Download PE-NMR.run

```
#! /bin/sh

# With the NMR key block you can specify for which atom you want the shielding
# tensor.

# Be aware that the NMR option is not implemented for the frozen core
# approximation, hence one must set option core of the Basis key block
# to NONE.

$ADFBIN/ams <<eor

Task SinglePoint

System
   Atoms
      C      -0.638348981     0.000000000     0.424302756
      C       0.638348981     0.000000000    -0.424302756
      H      -0.638348981     0.889974570     1.053624029
      H      -0.638348981    -0.889974570     1.053624029
      H       0.638348981     0.889974570    -1.053624029
      H       0.638348981    -0.889974570    -1.053624029
   End

   Lattice
      2.553395923     0.000000000     0.000000000
   end
End

Engine Band
   NMR
      Enabled True
      nmratom 1
      ms0 1.
   end

   XC
      GGA Always Becke Perdew
   end

   Dependency
      Basis 1e-10
   End

   Kspace
      Symmetric KInteg=3
   End
   Integration
      Accint 5
   End
```

```
   Basis
      Type TZ2P
      Core NONE
   End
EndEngine

eor
```

10.8.4 Example: EFG

Download SnO_EFG.run

```
#!/bin/sh

# The calculation of the electric field gradient is invoked by the EFG key
# block.

# Since Sn is quite an heavy atom we use the scalar relativistic option.

$ADFBIN/ams <<eor

Task SinglePoint

System
   FractionalCoords True

   Lattice
      3.8029  0.0  0.0
      0.0  3.8029  0.0
      0.0  0.0  4.8382
   End

   Atoms
      O   0.0  0.0  0.0
      O   0.5  0.5  0.0
      Sn  0.0  0.5  0.2369
      Sn  0.5  0.0 -0.2369
   End
End

Engine Band
   Title SnO EFG

   NumericalQuality Basic ! Only for speed

   Tails bas=1e-8 ! Only for reproducibility with nr. of cores

   Relativity
      Level Scalar
   End

   ! useful for Moessbauer spectroscopy: density and coulomb pot. at nuclei
   PropertiesAtNuclei
   End
```

```
   EFG
      Enabled True
   End

   Basis
      Type DZ
      Core none
   End
EndEngine
eor
```

10.8.5 Example: Phonons

Download GraphenePhonons.run

```
#!/bin/sh

# A phonon calculation should be performed at the equilibrium geometry.

# In the first calculation we optimize the geometry, including the lattice
# vectors. We also set the criteria a bit more strict.

echo "Geometry optimization"

AMS_JOBNAME=GO $ADFBIN/ams <<eor

System
  Atoms
    C   0.0 0.0 0.0
    C   1.23 0.7101408312 0.0
  END
  Lattice
    2.46 0.000000 0
    1.23 2.130422493 0
  End
End

Task GeometryOptimization

GeometryOptimization
  OptimizeLattice true
  Convergence Gradients=1e-5
End

Engine Band
  Title Graphene geometry optimization

  ! For Graphene we need to use a symmetryc grid
  KSpace
    Symmetric KInteg=5
    Type Symmetric
  End

  StrainDerivatives
    Analytical false
  End
```

```
  Basis
    Type DZ
  end
EndEngine
eor

# In the second calculation we use the pre-optimized geometry. (See details of
# the Restart key block) Then we define a supercell and perform a phonon run by
# using Task and Phonons keys. Note that KSpace can be chosen
# a bit lower, since we now have a bigger unit cell.

echo "Phonon calculation"

AMS_JOBNAME=Phonons $ADFBIN/ams <<eor

LoadSystem
  File GO.results/ams.rkf
End

Task SinglePoint

Properties
  Phonons True
End

NumericalPhonons
  stepSize 0.0913
  superCell
    2 0
    0 2
  subend
  Parallel
    nGroups 4
  End
end

Engine Band
  Title Graphene phonon calc

  KSpace
    Symmetric KInteg=3
    Type Symmetric
  End

  Basis
    Type DZ
  end
EndEngine

eor

NSCM=1
export NSCM
echo ""
echo "Begin TOC"

$ADFBIN/dmpkf -n 1 Phonons.results/band.rkf --toc
```

```
echo "End TOC"
```

10.9 Analysis

10.9.1 Example: CO absorption on a Cu slab: fragment option and densityplot

Download Frags_COCu.run

```
#! /bin/sh

# This example illustrates the usage of fragments in a BAND calculation for
# analysis purposes. It takes two runs to do the DOS analysis in a fragment
# basis, and an extra two runs to get the deformation density with respect to
# the fragment densities.

# The setup involves first the computation of the free CO overlayer, which is to
# be adsorbed on a Cu surface. To suppress (most of the) interactions between
# the CO molecules, i.e. to effectively get the molecular CO, the KSpace
# parameter is set to 1 (= no dispersion), and the lattice parameters are set so
# large that the CO molecules are far apart. The standard result file RUNKF is
# saved under the name 'CO.results/band.rkf'.


# ----------------------------- CO molecule --------------------------

AMS_JOBNAME=CO $ADFBIN/ams <<eor

Task SinglePoint

System
  ! CO molecules far apart

  Atoms [Bohr]
    C   0 0 0
    O   0 0 2.18
  End

  Lattice [Bohr]
    25.0  0.0   0.0
    0.0   25.0  0.0
  End
End

Engine Band
  Title The CO fragment

  Print AtomicChargesDetails

  Comment
   Technical
     Zero order k space integration
   Features
     Lattice   : 2D, large lattice vectors
     Unit cell : 2 atoms, 1x1, quasi molecular
```

```
     Basis     : NO+STO w/ core
  End

  Print Eigens

  Kspace
    Quality GammaOnly  ! neglect dispersion
  End

  Basis
    Type DZ
    Core Large
  End

  DOS
     Enabled True
     Energies 300
  End
EndEngine
eor


# Now we can use the result file to do a DOS analysis for CO on a copper surface
# treating the molecule as a fragment. With Fragment%Labels we assign names to
# the different symmetry orbitals. The Density-of-States analysis details are
# given with the keys DOS (energy grid, result file with DOS data) and,
# optionally, GrossPopulations and OverlapPopulations.

# ----------------------------- CO + Cu slab --------------------------

AMS_JOBNAME=COCu $ADFBIN/ams <<eor

Task SinglePoint

System
  Lattice [Bohr]
    4.822 0.0   0.0
    0.0   4.822 0.0
  End
  Atoms [Bohr]
    C   0     0     3.44
    O   0     0     5.62
    Cu  0.0   0.0   0.0
  End
End

Engine Band
  Title Cu slab with CO adsorbed

  Print AtomicChargesDetails

  Comment
   Technical
     Quadratic K space integration (low)
   Features
     Lattice   : 2D
     Unit cell : 3 atoms, 1x1
     Basis     : NO+STO w/ core
```

```
    Options    : Molecular fragment
                 Analysis: DOS, PDOS, COOP
End

KSpace
  Symmetric KInteg=3
End

! fragment specification

Fragment
  filename CO.results/band.rkf
  atommapping
    1 1
    2 2
  SubEnd
  Labels ! let us give them some labels
    2Sigma
    2Sigma*
    1Pi_x
    1Pi_y
    3Sigma
    1Pi_x*
    1Pi_y*
    3Sigma*
  SubEnd
End

! use fragment basis in dos
DosBas
  Fragment 1
End

DOS   ! Analysis
  Enabled   True
  File      pdos.CO_Cu
  Energies  500
  Min      -0.750
  Max       0.300
End

GrossPopulations
  3 2    ! All metal d states
  Sum     ! ALl metal sp states
    3 0
    3 1
  EndSum

  Frag 1   ! All CO states
  Sum   ! CO 1pi
    FragFun 1 5
    FragFun 1 6
  EndSum
  FragFun 1 7  ! CO 5-sigma
End

OverlapPopulations
  Left   ! Metal d with CO
```

```
      3 2
    Right
      Frag 1
  End

  Basis
    Type DZ
    Core Large
  End
EndEngine
eor

# After this run we copy the computed DOS data from the DOS result file to
# standard output. We also save the restart file for later use.

echo ""
echo "Contents of DOS file"
cat pdos.CO_Cu

# Next we want to know the deformation density with respect to the two
# fragments: 1) The CO molecule and 2) the bare Cu surface. We haven't done the
# bare Cu surface yet, so that is what happens next.

# ---------------------------- Cu slab -------------------------

AMS_JOBNAME=Cu $ADFBIN/ams <<eor

Task SinglePoint

System
  Lattice [Bohr]
    4.822 0.0 0.0
    0.0   4.822 0.0
  End
  Atoms [Bohr]
    Cu  0.0   0.0   0.0
  End
End


Engine Band
  Title Cu slab

  Print AtomicChargesDetails

  Comment
   Technical
     Quadratic K space integration (low)
   Features
     Lattice   : 2D
     Unit cell : 3 atoms, 1x1
     Basis     : NO+STO w/ core
     Options   :
  End

  Kspace
    Symmetric KInteg=3
  End
```

```
  Basis
    Type DZ
    Core Large
  End

  DOS
     Enabled True
     Energies 300
  End
EndEngine
eor

# Now we are all set to do our final calculation. We have the two fragment files
# CO.results/band.rkf and Cu.results/band.rkf, and the restart file COCu.results/band.
↪rkf. Next we want to know
# the deformation density with respect to the two fragments: 1) The CO molecule
# and 2) the bare Cu surface. The visualization options like OrbitalPlot and
# Densityplot require a regular set of points (a grid). Here is how it works

# ----------------------------- CO + Cu slab restart --------------------------

NSCM=1
export NSCM

AMS_JOBNAME=Final $ADFBIN/ams <<eor

Task SinglePoint

System
  Lattice [Bohr]
    4.822 0.0 0.0
    0.0   4.822 0.0
  End

  Atoms [Bohr]
    C   0     0     3.44
    O   0     0     5.62
    Cu  0.0   0.0   0.0
  End
End

Engine Band
  Title Cu slab with CO adsorbed (restart density plot)

  Print AtomicChargesDetails

  debug BlockPropertyModule

  Kspace
    Symmetric KInteg=3
  End

  Restart
    File COCu.results/band.rkf
    DensityPlot
  End
```

```
  Grid
    Type Coarse
  End

  DensityPlot
    rho(deformation/fit) !FITDENSITY_deformation_scf
  End

  ! fragment specification

  Fragment
    filename CO.results/band.rkf
    atommapping
    1 1
    2 2
    subend
  End

  Fragment
    filename Cu.results/band.rkf
    atommapping
      1 3
    subend
  End

  Basis
    Type DZ
    Core Large
  End

  DOS
     Enabled True
     Energies 300
  End
EndEngine
eor

# This particular restart options does not work in parallel, hence the '-n 1' on
# the first line.The result of the last run is a file named TAPE41. Normally you
# would save that to COCu.t41

# mv TAPE41 COCu.t41 and view it with ADFview. On the TAPE41 file are now three
# fields shown in ADFview as

# FITDENSITY_deformation_scf FITDENSITY_deformation_scf_frag1
# FITDENSITY_deformation_scf_frag2 being the deformation density of CO+Cu with
# respect to the atoms, and the same for the two fragments CO and the Cu slab.
# In ADFview you can add the fields of the two fragments, and then create
# another field that holds the difference.

NSCM=1
export NSCM
echo ""
echo "Begin TOC of tape41"

$ADFBIN/dmpkf -n 1 Final.results/FILE_BLOCKPROPERTIES --toc

echo "End TOC of tape41"
```

10.9.2 Example: Grid key for plotting results

Download GridKey.run

```
#!/bin/sh

SYSTEM=$ADFHOME/atomicdata/Molecules/TestMols/Methane.xyz

# Initial run

AMS_JOBNAME=methane $ADFBIN/ams <<eor

Task SinglePoint

System
  GeometryFile $SYSTEM
End

Engine Band
  Basis
    Type TZP
  End
EndEngine
eor


# Use the grid

AMS_JOBNAME=auto_grid $ADFBIN/ams <<eor

Task SinglePoint

System
  GeometryFile $SYSTEM
End

Engine Band
  Restart
    File methane.results/band.rkf
    DensityPlot
  End

  Grid
   Type Coarse
   ExtendX 21.1671 [Angstrom]
  End

  DensityPlot
    rho(fit)
  End

  Basis
    Type TZP
  End
EndEngine
eor
```

```
echo ""
echo "Begin TOC of tape41"

$ADFBIN/dmpkf -n 1 auto_grid.results/TAPE41 --toc

echo "End TOC of tape41"



# Use a completely user specified regular grid

AMS_JOBNAME=user_grid $ADFBIN/ams <<eor

Task SinglePoint

System
  GeometryFile $SYSTEM
End

Engine Band
  Restart
    File methane.results/band.rkf
    DensityPlot
  End

  Grid
  UserDefined
    -2.0 -1.3 -2.5
    1.0 0.0 0.0 0.02
    0 1 0.0 0.02
    0.0 0.0 1.0 0.02
    20 30 40
  SubEnd
  End

  DensityPlot
    rho(fit)
  End

  Basis
    Type TZP
  End
EndEngine
eor

echo ""
echo "Begin TOC of tape41"

$ADFBIN/dmpkf -n 1 user_grid.results/TAPE41 --toc

echo "End TOC of tape41"


# Use a text file to import the (arbitrary grid)

cat << eor > coords.txt
-3.0 0.0 0.0
```

```
-2.0 0.1 0.0
 0.0 0.2 0.0
eor

AMS_JOBNAME=file_grid $ADFBIN/ams <<eor

Task SinglePoint

System
  GeometryFile $SYSTEM
End

Engine Band
  Restart
    File methane.results/band.rkf
    DensityPlot
    vtkFile result.txt
  End

  Grid
    Filename coords.txt
  End

  DensityPlot
    rho(fit)
  End

  Basis
    Type TZP
  End
EndEngine

eor

echo ""
echo "Begin of result.txt"
cat result.txt
echo "End of result.txt"
```

10.9.3 Example: H2 on [PtCl4]2-: charged molecules and PEDA

Download PEDA_0D_PtCl4H2.run

```
#! /bin/sh

# This example shows that the pEDA formalism can be applied to
# molecules. Here, there is no periodic boundary condition
# necessary. Hence, charged fragments or final molecules can be
# investigated!

#
#
# Fragment 1 is the [PtCl4]2- fragment
#
#
```

```
AMS_JOBNAME=Frag1 $ADFBIN/ams <<eor

Task SinglePoint

System
    ATOMS
        Pt 0.0 0.0 0.0
        Cl 0.0 -2.308048739 0.0
        Cl 0.0 2.308048739 0.0
        Cl -2.308048739 0.0 0.0
        Cl 2.308048739 0.0 0.0
    END

    Charge -2
End

Engine Band
    TITLE PtCl4 2- fragment
    Relativity
        Level Scalar
    End

    Basis
    Type DZP
    Core Large
    End

    XC
    GGA Becke Perdew
    END

    UseSymmetry False
EndEngine
eor

#
#
# Fragment 2 is the H2 fragment
#
#

AMS_JOBNAME=Frag2 $ADFBIN/ams <<eor

Task SinglePoint

System
    ATOMS
        H 0.0 0.0 3.84182655
        H 0.0 0.0 2.952808836
    END
End

Engine Band
    TITLE H2 fragment


    Relativity
```

```
        Level Scalar
    End

    Basis
    Type DZP
    Core Large
    End

    XC
    GGA Becke Perdew
    END

    UseSymmetry False
EndEngine
eor

#
#
# The energy decomposition run for the complex ([PtCl4]H2)2- complex
#
#

$ADFBIN/ams <<eor

Task SinglePoint

System
    ATOMS
        Pt 0.0 0.0 0.0
        Cl 0.0 -2.308048739 0.0
        Cl 0.0 2.308048739 0.0
        Cl -2.308048739 0.0 0.0
        Cl 2.308048739 0.0 0.0
        H 0.0 0.0 2.952808836
        H 0.0 0.0 3.84182655
    END
    Charge -2
End

Engine Band
    Relativity
        Level Scalar
    End

    Basis
    Type DZP
    Core Large
    End

    XC
    GGA Becke Perdew
    END

    fragment
        filename Frag1.results/band.rkf
        AtomMapping
            1 1
            2 2
```

```
            3 3
            4 4
            5 5
        SubEnd
    end

    fragment
        filename Frag2.results/band.rkf
        AtomMapping
            1 7
            2 6
        SubEnd
    end

    PEDA

    UseSymmetry False
EndEngine

eor
```

10.9.4 Example: CO absorption on a MgO slab: fragment option and PEDA

Download PEDA_MgO+CO.run

```
#! /bin/sh

# This example shall illustrate the use of the Fragment keywords in combination
# with the PEDA keyword to perform the PEDA. For this example two fragment
# calculations are necessary to calculate the unperturbed eigensystems of the
# MgO slab and CO fragment.

# == Fragment calculations ==

# ------------------------------ MgO slab -------------------------

AMS_JOBNAME=MgO $ADFBIN/ams <<eor

Task SinglePoint

System
  Atoms
     Mg  0.00000000       0.00000000       0.00000000
     Mg  1.50260191      -1.50260191      -2.12400000
     Mg  0.00000000       0.00000000      -4.24800000
     Mg  3.00520382       0.00000000       0.00000000
     Mg  1.50260191       1.50260191      -2.12400000
     Mg  3.00520382       0.00000000      -4.24800000
     O   1.50260191      -1.50260191       0.00200000
     O   0.00000000       0.00000000      -2.12400000
     O   1.50260191      -1.50260191      -4.25000000
     O   1.50260191       1.50260191       0.00200000
     O   3.00520382       0.00000000      -2.12400000
     O   1.50260191       1.50260191      -4.25000000
  End
```

```
  Lattice
         3.00520382      -3.00520382       0.00000000
         3.00520382       3.00520382       0.00000000
  End
End

Engine Band
  Title MgO surface

  skip dos

  KSpace
    Regular
      NumberOfPoints 3 3
    End
  End

  BeckeGrid
     quality basic
  End

  XC
     GGA PBE
  End

  Basis
     Type TZP
     Core small
  End
EndEngine

eor

#------------------------------ CO fragment -------------------------

AMS_JOBNAME=CO $ADFBIN/ams <<eor

Task SinglePoint

System
  Atoms
     C   0.00000000       0.00000000       2.61000000
     O   0.00000000       0.00000000       3.737000000
  End

  Lattice
         3.00520382      -3.00520382       0.00000000
         3.00520382       3.00520382       0.00000000
  End
End

Engine Band
  Title CO fragment

  KSpace
    Regular
      NumberOfPoints 3 3
    End
```

```
  End

  BeckeGrid
     quality basic
  End

  XC
     GGA PBE
  End

  Basis
     Type TZP
     Core small
  End
EndEngine
eor

# == PEDA calculation ==

# The two result files, MgO.kf and CO.kf, can now be used to perform the
# PEDA. Here, the mapping of the atoms of the PEDA calculation and the fragment
# calculations is necessary. And the used grid points in reciprocal space have
# to be identical in all three calculations.

# ---------------------------- PEDA calculation -----------------------

$ADFBIN/ams <<eor

Task SinglePoint

System
  Atoms
     Mg.frag_MgO  0.00000000       0.00000000       0.00000000
     Mg.frag_MgO  1.50260191      -1.50260191      -2.12400000
     Mg.frag_MgO  0.00000000       0.00000000      -4.24800000
     Mg.frag_MgO  3.00520382       0.00000000       0.00000000
     Mg.frag_MgO  1.50260191       1.50260191      -2.12400000
     Mg.frag_MgO  3.00520382       0.00000000      -4.24800000
     O.frag_MgO   1.50260191      -1.50260191       0.00200000
     O.frag_MgO   0.00000000       0.00000000      -2.12400000
     O.frag_MgO   1.50260191      -1.50260191      -4.25000000
     O.frag_MgO   1.50260191       1.50260191       0.00200000
     O.frag_MgO   3.00520382       0.00000000      -2.12400000
     O.frag_MgO   1.50260191       1.50260191      -4.25000000
     O.frag_CO    0.00000000       0.00000000       3.73700000
     C.frag_CO    0.00000000       0.00000000       2.61000000
  End

  Lattice
         3.00520382      -3.00520382       0.00000000
         3.00520382       3.00520382       0.00000000
  End
End

Engine Band
  Title PEDA

  KSpace
```

```
    Regular
      NumberOfPoints 3 3
    End
  End

  BeckeGrid
     quality basic
  End

  XC
     GGA PBE
  End

  fragment
    filename MgO.results/band.rkf
    AtomMapping
       1  1
       2  2
       3  3
       4  4
       5  5
       6  6
       7  7
       8  8
       9  9
       10 10
       11 11
       12 12
     SubEnd
  end

  fragment
    filename CO.results/band.rkf
    AtomMapping
       2  13
       1  14
     SubEnd
  end

  PEDA

  Basis
     Type TZP
     Core small
  End
EndEngine

eor

# In the output file the results can be found in the PEDA block after the Energy
# Analysis.
```

10.9.5 Example: CO absorption on a MgO slab: fragment option, PEDA and PEDANOCV

Download PEDANOCV_MgO+CO.run

```
#! /bin/sh

# This example shall illustrate the use of the Fragment keywords in combination
# with the PEDA and PEDANOCV keywords to perform the PEDANOCV calculation. For
# this example two fragment calculations are necessary to calculate the
# unperturbed eigensystems of the MgO slab and CO fragment. Here, the sampling
# of the reciprocal space is restricted to gamma point

# == Fragment calculations ==

# ------------------------------ MgO slab -------------------------

AMS_JOBNAME=MgO $ADFBIN/ams <<eor

Task SinglePoint

System
   Atoms
      Mg  0.00000000       0.00000000       0.00000000
      Mg  1.50260191      -1.50260191      -2.12400000
      Mg  0.00000000       0.00000000      -4.24800000
      Mg  3.00520382       0.00000000       0.00000000
      Mg  1.50260191       1.50260191      -2.12400000
      Mg  3.00520382       0.00000000      -4.24800000
      O   1.50260191      -1.50260191       0.00200000
      O   0.00000000       0.00000000      -2.12400000
      O   1.50260191      -1.50260191      -4.25000000
      O   1.50260191       1.50260191       0.00200000
      O   3.00520382       0.00000000      -2.12400000
      O   1.50260191       1.50260191      -4.25000000
   End

   Lattice
          3.00520382      -3.00520382       0.00000000
          3.00520382       3.00520382       0.00000000
   End
End

Engine Band
   Title MgO fragment

   skip dos

    KSpace
      Regular
        NumberOfPoints 1 1
      End
    End

   BeckeGrid
      quality basic
   End

   Relativity
      Level Scalar
   End

   XC
```

```
      GGA PBE
   End

   Basis
      Type TZP
      Core none
   End
EndEngine
eor

#------------------------------ CO fragment -------------------------

AMS_JOBNAME=CO $ADFBIN/ams <<eor

Task SinglePoint

System
   Atoms
      C   0.00000000       0.00000000       2.61000000
      O   0.00000000       0.00000000       3.737000000
   End

   Lattice
         3.00520382      -3.00520382       0.00000000
         3.00520382       3.00520382       0.00000000
   End
End

Engine Band
   Title CO fragment

   KSpace
      Regular
         NumberOfPoints 1 1
      End
   End

   BeckeGrid
      quality basic
   End

   Relativity
      Level Scalar
   End

   XC
      GGA PBE
   End

   Basis
      Type TZP
      Core none
   End
EndEngine
eor

# == PEDANOCV calculation ==
```

```
# The two result files, MgO.kf and CO.kf, can now be used to perform the
# PEDANOCV. Here, the mapping of the atoms of the PEDA calculation and the
# fragment calculations is necessary. And the used grid points in reciprocal
# space have to be identical in all three calculations - in this case the gamma
# point for all calculations.

#------------------------------ PEDANOCV calculation ------------------------


AMS_JOBNAME=decomp $ADFBIN/ams <<eor

Task SinglePoint

System
   Atoms
      Mg.frag_MgO  0.00000000       0.00000000       0.00000000
      Mg.frag_MgO  1.50260191      -1.50260191      -2.12400000
      Mg.frag_MgO  0.00000000       0.00000000      -4.24800000
      Mg.frag_MgO  3.00520382       0.00000000       0.00000000
      Mg.frag_MgO  1.50260191       1.50260191      -2.12400000
      Mg.frag_MgO  3.00520382       0.00000000      -4.24800000
      O.frag_MgO   1.50260191      -1.50260191       0.00200000
      O.frag_MgO   0.00000000       0.00000000      -2.12400000
      O.frag_MgO   1.50260191      -1.50260191      -4.25000000
      O.frag_MgO   1.50260191       1.50260191       0.00200000
      O.frag_MgO   3.00520382       0.00000000      -2.12400000
      O.frag_MgO   1.50260191       1.50260191      -4.25000000
      C.frag_CO    0.00000000       0.00000000       2.61000000
      O.frag_CO    0.00000000       0.00000000       3.737000000
   End

   Lattice
          3.00520382      -3.00520382       0.00000000
          3.00520382       3.00520382       0.00000000
   End
End

Engine Band
   Title Mg+CO

   KSpace
      Regular
         NumberOfPoints 1 1
      End
   End

   BeckeGrid
      quality basic
   End

   Relativity
      Level Scalar
   End

   XC
      GGA PBE
   End
```

```
   fragment
      filename MgO.results/band.rkf
      AtomMapping
         1  1
         2  2
         3  3
         4  4
         5  5
         6  6
         7  7
         8  8
         9  9
         10 10
         11 11
         12 12
      SubEnd
   end

   fragment
      filename CO.results/band.rkf
      AtomMapping
         1  13
         2  14
      SubEnd
   end

   PEDA

   PEDANOCV
      EigvalThresh 0.001
   End

   Basis
      Type TZP
      Core none
   End
EndEngine
eor

# In the output file the results can be found in the PEDANOCV block after the
# Energy Analysis and PEDA block.

# The NOCV orbitals and NOCV deformation densities can be visualized using
# ADFview or by a restart calculation. In the latter case, one adds the Restart
# block key with the options File decomp.kf and the NOCVdRhoPlot and
# NOCVOrbitalPlot keys. These will trigger the calculation of the plot
# properties. To specify which NOCV deformation densities and NOCV orbitals are
# plotted, one adds the NOCVdRhoPlot and NOCVOrbitalPlot block key. In both
# blocks the line 1 Band 1 5 means, that for k-point 1 the densities/orbitals 1
# to 5 are calculated.



export NSCM=1
$ADFBIN/ams <<eor

Task SinglePoint
```

```
System
   Atoms
      Mg  0.00000000       0.00000000       0.00000000
      Mg  1.50260191      -1.50260191      -2.12400000
      Mg  0.00000000       0.00000000      -4.24800000
      Mg  3.00520382       0.00000000       0.00000000
      Mg  1.50260191       1.50260191      -2.12400000
      Mg  3.00520382       0.00000000      -4.24800000
      O   1.50260191      -1.50260191       0.00200000
      O   0.00000000       0.00000000      -2.12400000
      O   1.50260191      -1.50260191      -4.25000000
      O   1.50260191       1.50260191       0.00200000
      O   3.00520382       0.00000000      -2.12400000
      O   1.50260191       1.50260191      -4.25000000
      C   0.00000000       0.00000000       2.61000000
      O   0.00000000       0.00000000       3.737000000
   End

   Lattice
          3.00520382      -3.00520382       0.00000000
          3.00520382       3.00520382       0.00000000
   End
End

Engine Band
   Title Restart Calculation

   Restart
      File decomp.results/band.rkf
      NOCVdRhoPlot
   End

   NOCVdRhoPlot
      1 Band 1
   End

   Grid
      Type coarse
   End

   KSpace
      Regular
         NumberOfPoints 1 1
      End
   End

   BeckeGrid
      quality basic
   End

   Relativity
      Level Scalar
   End

   XC
      GGA PBE
   End
```

```
   Basis
      Type TZP
      Core none
   End

   debug BlockPropertyModule
EndEngine
eor

echo ""
echo "Begin TOC of tape41"
export NSCM=1
$ADFBIN/pkf -n 1 ams.results/FILE_BLOCKPROPERTIES

echo "End TOC of tape41"

# The important output of this calculation is the TAPE41 file. Renaming it to
# foobar.t41 will allow ADFview to read and interpret the data stored on this
# file.
```

10.9.6 Example: Bader analysis

Download Li2O_Bader.run

```
#! /bin/sh

# To get the Quantum Theory of Atoms In Molecules and Crystals (QT-AIMAC)
# analysis use the GridBasedAIM block key.

# The grid-based AIM method is very fast, but a bit inaccurate. Hence, on has to
# make sure that the results are converged w.r.t. the real-space integration
# grid.

$ADFBIN/ams <<eor

Task SinglePoint

System
   Lattice [Bohr]
      0.0   4.365 4.365
      4.365 0.0   4.365
      4.365 4.365 0.0
   end

   Atoms [Bohr]
      O   0.0     0.0      0.0
      Li  2.1825  2.1825   2.1825
      Li  6.5475  2.1825   2.1825
   end
End

Engine Band
   Title Li2O bulk (fluorite structure)

   KSpace
      Symmetric KInteg=3
```

```
   End

   IntegrationMethod Voronoi

   Integration
      Accint 4
      accsph 6
      accpyr 6
   end

   GridBasedAIM
      Enabled Yes
   End

   Dependency basis=1e-9 fit=1e-8

   DIIS
      dimix 0.2
      ncycledamp 0
   end

   scf
      mixing 0.4
   end

   xc
      gga scf bp86
   end

   Basis
      Type TZ2P
      Core small
   end
EndEngine
eor
```

10.9.7 Example: Properties at nuclei

Download PropertiesAtNuclei.run

```
#!/bin/sh

# One can obtain the values of some properties near the nucleus. (see
# PropertiesAtNuclei)

# Note: Instead of calculating the properties at a point in space an average is
# taken over a tiny sphere around this point.

$ADFBIN/ams <<eor

Task SinglePoint

System
   Atoms
      O   0.000  0.000  0.000
      O   0.000  0.000  1.208
```

```
   end
End

Engine Band
   Title Properties at nuclei for O2

   Unrestricted

   PropertiesAtNuclei
      vxc[rho(fit)]
      rho(fit)
      rho
      v(coulomb)
      rho(deformation/fit)
      rho(deformation/scf)
   End

   Basis
      Type DZ
      Core None
   End

   XC
      gga always pbe
   END
EndEngine
eor
```

10.9.8 Example: Band structure plot

Download Li_BZPlot.run

```
#! /bin/sh

# In the first example we use the automatic k-path through the Brillouin zone
# (see BandStructure key-block). The results can be visualized with the BandStructure
# Gui Module.


AMS_JOBNAME=auto $ADFBIN/ams <<eor

Task SinglePoint

System
  Atoms
     Li 0.0 0.0 0.0
  END
  Lattice
    -1.745  1.745  1.745
     1.745 -1.745  1.745
     1.745  1.745 -1.745
  End
End

Engine Band
  NumericalQuality Basic
```

```
  BandStructure
     Enabled   true
     Automatic true
     FatBands  false
     EnergyAboveFermi 2.0
  end
EndEngine
eor

# In the second example we specify the path through the Brillouin zone by hand.
# We set automatic to false and then specify the path with the BZPath key block,
# using one or more path subkeys. Here, the second run will produce exactly the
# same path as the automatic one.

AMS_JOBNAME=user $ADFBIN/ams <<eor

Task SinglePoint

System
  Atoms
     Li 0.0 0.0 0.0
  END
  Lattice
    -1.745  1.745  1.745
     1.745 -1.745  1.745
     1.745  1.745 -1.745
  End
End

Engine Band
  NumericalQuality Basic

  BandStructure
     Enabled   true
     Automatic false
     FatBands  false
     EnergyAboveFermi 2.0
  end

  bzpath
     path
        0.00  0.00  0.00  G
        0.50 -0.50  0.50  H
        0.00  0.00  0.50  N
        0.00  0.00  0.00  G
        0.25  0.25  0.25  P
        0.50 -0.50  0.50  H
     subend
     path
        0.25  0.25  0.25  P
        0.00  0.00  0.50  N
     subend
  end
EndEngine
eor

export NSCM=1
```

```
# The band structure is best visualized using the BandStructure GUI module.

echo 'Extract the band_curves section from the rkf files:'
$ADFBIN/dmpkf auto.results/band.rkf 'band_curves'
$ADFBIN/dmpkf user.results/band.rkf 'band_curves'
```

10.9.9 Example: Effective Mass (electron mobility)

Download EffectiveMass.run

```
#! /bin/sh

# An effective mass calculation is about the curvature of band at the top of the
# valence band and the bottom of the conduction band. This is obtained by
# numerical differentiation.

# It can be done for systems with 1D, 2D, or 3D translational symmetry.

# The easiest way to use this feature is to specify an empty EffectiveMass key
# block (so leave out the NumAbove, NumBelow, and UniqueKPoints).

# == Example 1D ==

echo "example 1D"

AMS_JOBNAME=EffectiveMass1D $ADFBIN/ams <<eor

Task SinglePoint

System
  Atoms
     Al 0.0 0.0 0.0
  END
  Lattice
     2.12440502 0.0 0.0
  End
End

Engine Band
  TITLE 1D Al Chain

  EffectiveMass
    Enabled True
    KPointCoord -0.783
    StepSize 0.001
    NumAbove 4
    NumBelow 2
  End

  Basis
     Type DZ
     Core Large
  End
EndEngine
eor
```

```
# == Example 2D ==

echo "example 2D"

AMS_JOBNAME=EffectiveMass2D $ADFBIN/ams <<eor

Task SinglePoint

System
  Atoms
     Mo     -1.626960686     0.313108730     0.000000000
     S       0.000000000     1.252434919     1.547040825
     S       0.000000000     1.252434919    -1.547040825
  End

  Lattice
     1.626960686    -2.817978569     0.000000000
     1.626960686     2.817978569     0.000000000
  End
End

Engine Band
  TITLE MoS2Slab

  Relativity
    Level Scalar
  End

  EffectiveMass
    Enabled True
  End

  Basis
     Type DZ
     Core Large
  End
EndEngine

eor

# == Example 3D ==

echo "example 3D"

AMS_JOBNAME=EffectiveMass3D $ADFBIN/ams <<eor

Task SinglePoint
System
  Atoms
     Zn  1.625 0.9381941876 0.0
     Zn  1.625 -0.9381941878 2.615
     O   1.625 0.9381941876 1.96125
     O   1.625 -0.9381941878 4.57625
  END
  Lattice
     1.625 -2.814582562 0.000000
```

```
     1.625 2.814582562 0.000000
     0.000000 0.000000 5.23
  End
End

Engine Band
  TITLE ZnO

  NumericalQuality Basic

  KSpace
    Quality Normal
  End
  tails bas=1e-8

  EffectiveMass
    Enabled True
    NumAbove 1
    NumBelow 1
  End

  Basis
     Type DZ
     Core Large
  End
EndEngine

eor
```

10.9.10 Example: Generating an Excited State with and Electron Hole

Download Si_ElectronHole.run

```
#! /bin/sh

# There is the possibility define the excitation of an electron from a low
# lying, localized band to a virtual band. The ElectronHole key does allow the
# specification of the original band and the spin of the electron. The
# EnforcedSpinPolarization key allows to restrict the spin polarization of the
# whole system.

$ADFBIN/ams <<eor

Task SinglePoint

System
  Atoms
    Si.frozen_core -0.67875 -0.67875 -0.67875
    Si              0.67875  0.67875  0.67875
  End
  Lattice
      0.000  2.715  2.715
      2.715  0.000  2.715
      2.715  2.715  0.000
  End
End
```

```
Engine Band

  TITLE Untitled

  Basis
    Type DZP
    core none
    ByAtomType
      Si.frozen_core  Type=DZP core=Large
    End
  End


  XC
     LDA SCF VWN
  END

  UNRESTRICTED

  ElectronHole
      BandIndex 1
      SpinIndex 1
  End

  EnforcedSpinPolarization 0
EndEngine
eor
```

10.10 List of Examples

- *BasisDefaults* (page 139)
- *BeO_tape41* (page 152)
- *BetaIron* (page 127)
- *BFieldLdotB* (page 128)
- *BNForce* (page 176)
- *BSSE* (page 143)
- *COChainFreqTS* (page 170)
- *EffectiveMass* (page 216)
- *EField* (page 132)
- *FiniteNucleus* (page 134)
- *Frags_COCu* (page 192)
- *Graphene_Dispersion* (page 129)
- *GraphenePhonons* (page 190)
- *GridKey* (page 198)
- *H2BulkGeo* (page 177)
- *H_ref* (page 172)
- *HonPerovskite_Solvation* (page 131)
- *Li2O_Bader* (page 212)
- *Li_BZPlot* (page 214)
- *Multiresolution_H2O* (page 141)
- *NaCl* (page 168)
- *NEGF* (page 154)
- *NEGF with bias* (page 161)

- *NEGF_Conductance* (page 163)
- *NewResp_3DCopper* (page 182)
- *NewResp_PlotInducedDensity* (page 183)
- *NewResponse for 2D Slab* (page 179)
- *NiO_Hubbard* (page 136)
- *OldResp_Diamond* (page 185)
- *PE-NMR* (page 188)
- *PEDA* (page 203)
- *PEDANOCV* (page 206)
- *Peptide_NumericalQuality* (page 140)
- *PropertiesAtNuclei* (page 213)
- *Restart a SCF* (page 146)
- *Restart for Properties* (page 150)
- *Si_ElectronHole* (page 218)
- *SnO_EFG* (page 189)
- *TiF3a* (page 186)
- *TiF3g* (page 187)
- *ZnS_ModelPotential* (page 137)

CHAPTER

ELEVEN

REQUIRED CITATIONS

When you publish results in the scientific literature which 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 references to special features are mandatory, in case you have used them.

11.1 General References

For calculations with the periodic structures BAND program, version 2018:

1. G. te Velde and E.J. Baerends, *Precise density-functional method for periodic structures*, Physical Review B 44, 7888 (1991) (https://doi.org/10.1103/PhysRevB.44.7888).
2. G. Wiesenekker and E.J. Baerends, *Quadratic integration over the three-dimensional Brillouin zone*, Journal of Physics: Condensed Matter 3, 6721 (1991) (https://doi.org/10.1088/0953-8984/3/35/005).
3. 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).
4. 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).
5. BAND2018, SCM, Theoretical Chemistry, Vrije Universiteit, Amsterdam, The Netherlands, http://www.scm.com Optionally, you may add the following list of authors and contributors: P.H.T. Philipsen, G. te Velde, E.J. Baerends, J.A. Berger, P.L. de Boeij, M. Franchini, J.A. Groeneveld, E.S. Kadantsev, R. Klooster, F. Kootstra, P. Romaniello, M. Raupach, D.G. Skachkov, J.G. Snijders, C.J.O. Verzijl, J.A. Celis Gil, J. M. Thijssen, G. Wiesenekker, T. Ziegler.

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.

11.2 Feature References

Lead See key references above, for all work with BAND

Suggested G. Wiesenekker, G. te Velde and E.J. Baerends, *Analytic quadratic integration over the two-dimensional Brillouin zone*, Journal of Physics C: Solid State Physics 21, 4263 (1988) (https://doi.org/10.1088/0022-3719/21/23/012).

11.2.1 Geometry optimization

Lead E.S. Kadantsev, R. Klooster. P.L. de Boeij and T. Ziegler, *The Formulation and Implementation of Analytic Energy Gradients for Periodic Density Functional Calculations with STO/NAO Bloch Basis Set*, Molecular Physics 105, 2583 (2007) (https://doi.org/10.1080/00268970701598063).

11.2.2 TDDFT

Lead F. Kootstra, P.L. de Boeij and J.G. Snijders, *Efficient real-space approach to time-dependent density functional theory for the dielectric response of nonmetallic crystals*, Journal of Chemical Physics 112, 6517 (2000) (https://doi.org/10.1063/1.481315).

P. Romaniello and P.L. de Boeij, *Time-dependent current-density-functional theory for the metallic response of solids*, Physical Review B 71, 155108 (2005) (https://doi.org/10.1103/PhysRevB.71.155108).

Main applications F. Kootstra, P.L. de Boeij, and J.G. Snijders, *Application of time-dependent density-functional theory to the dielectric function of various nonmetallic crystals*, Physical Review B 62, 7071 (2000) (https://doi.org/10.1103/PhysRevB.62.7071).

P. Romaniello, P.L. de Boeij, F. Carbone, and D. van der Marel, *Optical properties of bcc transition metals in the range 0 - 40 eV*, Physical Review B 73, 075115 (2006) (https://doi.org/10.1103/PhysRevB.73.075115).

Suggested book references F. Kootstra, Ph.D. thesis (http://www.scm.com/Doc/ft439.pdf), Rijksuniversiteit Groningen, Groningen (2001).

P. Romaniello, Ph.D. thesis (http://www.scm.com/Doc/Thesis_Pina.pdf), Rijksuniversiteit Groningen, Groningen (2006).

A. Berger, Ph.D. thesis (http://www.scm.com/Doc/Thesis_Arjan.pdf), Rijksuniversiteit Groningen, Groningen (2006).

11.2.3 Relativistic TDDFT

Lead P. Romaniello and P.L. de Boeij, *Relativistic two-component formulation of time-dependent current-density functional theory: Application to the linear response of solids*, Journal of Chemical Physics 127, 174111 (2007) (https://doi.org/10.1063/1.2780146).

11.2.4 Vignale Kohn

Lead J.A. Berger, P.L. de Boeij and R. van Leeuwen, *Analysis of the viscoelastic coefficients in the Vignale-Kohn functional: The cases of one- and three-dimensional polyacetylene*, Physical Review B 71, 155104 (2005) (https://doi.org/10.1103/PhysRevB.71.155104).

Applications J.A. Berger, P. Romaniello, R. van Leeuwen and P.L. de Boeij, *Performance of the Vignale-Kohn functional in the linear response of metals*, Physical Review B 74, 245117 (2006) (https://doi.org/10.1103/PhysRevB.74.245117).

J.A. Berger, P.L. de Boeij, and R. van Leeuwen, *Analysis of the Vignale-Kohn current functional in the calculation of the optical spectra of semiconductors*, Physical Review B 75, 35116 (2007) (https://doi.org/10.1103/PhysRevB.75.035116).

11.2.5 NMR

Lead D. Skachkov, M. Krykunov, E. Kadantsev, and T. Ziegler, *The Calculation of NMR Chemical Shifts in Periodic Systems Based on Gauge Including Atomic Orbitals and Density Functional Theory*, Journal of Chemical Theory and Computation 6, 1650 (2010) (https://doi.org/10.1021/ct100046a)

D. Skachkov, M. Krykunov, and T. Ziegler, *An improved scheme for the calculation of NMR chemical shifts in periodic systems based on gauge including atomic orbitals and density functional theory*, Canadian Journal of Chemistry 89, 1150 (2011) (https://doi.org/10.1139/v11-050).

11.2.6 ESR

A-tensor: Nuclear magnetic dipole hyperfine interaction E.S. Kadantsev and T. Ziegler, *Implementation of a Density Functional Theory-Based Method for the Calculation of the Hyperfine A-tensor in Periodic Systems with the Use of Numerical and Slater Type Atomic Orbitals: Application to Paramagnetic Defects*, Journal of Physical Chemistry A 112, 4521 (2008) (https://doi.org/10.1021/jp800494m).

G-tensor: Zeeman interaction E.S. Kadantsev and T. Ziegler, *Implementation of a DFT Based Method for the Calculation of Zeeman g-tensor in Periodic Systems with the use of Numerical and Slater Type Atomic Orbitals*, Journal of Physical Chemistry A 113, 1327 (2009) (https://doi.org/10.1021/jp805466c).

11.2.7 NEGF

Lead

3. (a) O. Verzijl and J. M. Thijssen *DFT-Based Molecular Transport Implementation in ADF/BAND*, J. Phys. Chem. C, 2012, 116 (46), pp 24393–24412 (https://doi.org/10.1021/jp3044225).

11.3 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.

CHAPTER

TWELVE

REFERENCES

1. 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, 1200 (1980) (https://doi.org/10.1139/p80-159).

2. 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/PL00020511).

3. 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).

4. J.P. Perdew and Y. Wang, *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).

5. J.P. Perdew, J.A. Chevary, S.H. Vosko, K.A. Jackson, M.R. Pederson, D.J. Singh 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).

6. J.P. Perdew, *Density-functional approximation for the correlation energy of the inhomogeneous electron gas.* Physical Review B 33, 8822 (1986) (https://doi.org/10.1103/PhysRevB.33.8822).

7. 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).

8. 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).

9. 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).

10. 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).

11. R. Neumann, R.H. Nobes and N.C. Handy, *Exchange functionals and potentials.* Molecular Physics 87, 1 (1996) (https://doi.org/10.1080/00268979600100011).

12. 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).

13. B. Hammer, L.B. Hansen, and J.K.Nørskov, *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).

14. J.P. Perdew, A. Ruzsinszky, G.I. Csonka, O.A. Vydrov, G.E. Scuseria, L.A. Constantin, X. Zhou and K. Burke, *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).

15. J. Tao, J.P. Perdew, V.N. Staroverov and G.E. Scuseria, *Climbing the Density Functional Ladder: Nonempirical Meta-Generalized Gradient Approximation Designed for Molecules and Solids.* Physical Review Letters 91, 146401 (2003) (https://doi.org/10.1103/PhysRevLett.91.146401).

16. Y. Zhao, 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).

17. 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).

18. P.H.T. Philipsen, and E.J. Baerends, *Relativistic calculations to assess the ability of the generalized gradient approximation to reproduce trends in cohesive properties of solids.* Physical Review B 61, 1773 (2000) (https://doi.org/10.1103/PhysRevB.61.1773).

19. E.S. Kadantsev, R. Klooster. P.L. de Boeij and T. Ziegler, *The Formulation and Implementation of Analytic Energy Gradients for Periodic Density Functional Calculations with STO/NAO Bloch Basis Set.* Molecular Physics 105, 2583 (2007) (https://doi.org/10.1080/00268970701598063).

20. E.S. Kadantsev and T. Ziegler, *Implementation of a Density Functional Theory-Based Method for the Calculation of the Hyperfine A-tensor in Periodic Systems with the Use of Numerical and Slater Type Atomic Orbitals: Application to Paramagnetic Defects.* Journal of Physical Chemistry A 112, 4521 (2008) (https://doi.org/10.1021/jp800494m).

21. E.S. Kadantsev and T. Ziegler, *Implementation of a DFT Based Method for the Calculation of Zeeman g-tensor in Periodic Systems with the use of Numerical and Slater Type Atomic Orbitals.* Journal of Physical Chemistry A 113, 1327 (2009) (https://doi.org/10.1021/jp805466c).

22. F. Kootstra, P.L. de Boeij and J.G. Snijders, *Efficient real-space approach to time-dependent density functional theory for the dielectric response of nonmetallic crystals.* Journal of Chemical Physics 112, 6517 (2000). (https://doi.org/10.1063/1.481315)

23. P. Romaniello and P.L. de Boeij, *Time-dependent current-density-functional theory for the metallic response of solids.* Physical Review B 71, 155108 (2005) (https://doi.org/10.1103/PhysRevB.71.155108).

24. J.A. Berger, P.L. de Boeij and R. van Leeuwen, *Analysis of the viscoelastic coefficients in the Vignale-Kohn functional: The cases of one- and three-dimensional polyacetylene.*, Physical Review B 71, 155104 (2005) (https://doi.org/10.1103/PhysRevB.71.155104).

25. P. Romaniello and P.L. de Boeij, *Relativistic two-component formulation of time-dependent current-density functional theory: application to the linear response of solids.*, Journal of Chemical Physics 127, 174111 (2007) (https://doi.org/10.1063/1.2780146).

26. 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).

27. 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).

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. C. Adamo and V. Barone, *Physically motivated density functionals with improved performances: The modified Perdew.Burke.Ernzerhof model.* Journal of Chemical Physics 116, 5933 (2002) (https://doi.org/10.1063/1.1458927).

30. N.C. Handy and A.J. Cohen, *Left-right correlation energy.* Molecular Physics 99, 403 (2001) (https://doi.org/10.1080/00268970010018431).

31. M. Swart, A.W. Ehlers and K. Lammertsma, *Performance of the OPBE exchange-correlation functional.* Molecular Physics 2004 102, 2467 (2004) (https://doi.org/10.1080/00268970420002750 17).

32. 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).

33. J.I. Rodriguez, 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 Compututational Chemistry 30, 1082 (2009) (https://doi.org/10.1002/jcc.21134).

34. J.I. Rodriguez, R.F.W. Bader, P.W. Ayers, C. Michel, A.W. Gotz 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).

35. J. Tersoff and D. R. Hamann, *Theory of the scanning tunneling microscope.* Physical Review B 31, 505 (1985) (https://doi.org/10.1103/PhysRevB.31.805).

36. 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).

37. D. Skachkov, M. Krykunov, E. Kadantsev, and T. Ziegler, *The Calculation of NMR Chemical Shifts in Periodic Systems Based on Gauge Including Atomic Orbitals and Density Functional Theory.* Journal of Chemical Theory and Computation 6, 1650 (2010) (https://doi.org/10.1021/ct100046a).

38. 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).

39. B. Delley, *The conductor-like screening model for polymers and surfaces.* Molecular Simulation 32, 117 (2006) (https://doi.org/10.1080/08927020600589684).

40. 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).

41. 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*, The Journal of Chemical Physics 132, 154104 (2010) (https://doi.org/10.1063/1.3382344).

42. S. Grimme, S. Ehrlich, and L. Goerigk, *Effect of the Damping Function in Dispersion Corrected Density Functional Theory*, Journal of Computational Chemistry 32, 1456 (2011) (https://doi.org/10.1002/jcc.21759).

43. P. Haas, F. Tran, P. Blaha, and K. H. Schwarz, *Construction of an optimal GGA functional for molecules and solids*, Physical Review B 83, 205117 (2011) (https://doi.org/10.1103/PhysRevB.83.205117).

44. F. Tran, and P. Blaha, *Accurate Band Gaps of Semiconductors and Insulators with a Semilocal Exchange-Correlation Potential*, Physical Review Letters 102, 226401 (2009) (https://doi.org/10.1103/PhysRevLett.102.226401).

45. D. Alfè, *PHON: A program to calculate phonons using the small displacement method*, Computer Physics Communications 180, 2622 (2009) (https://doi.org/10.1016/j.cpc.2009.03.010).

46. V.I. Anisimov, F. Aryasetiawan, and A.I. Lichtenstein, *First-principles calculations of the electronic structure and spectra of strongly correlated systems: the LDA + U method*, Journal Physics: Condensed Matter 9, 767 (1997) (https://doi.org/10.1088/0953-8984/9/4/002).

47. M. Cococcioni, and S. de Gironcoli, *Linear response approach to the calculation of the effective interaction parameters in the LDA+U method*, Physical Review B 71, 035105 (2005) (https://doi.org/10.1103/PhysRevB.71.035105).

48. D. Skachkov, M. Krykunov, and T. Ziegler, *An improved scheme for the calculation of NMR chemical shifts in periodic systems based on gauge including atomic orbitals and density functional theory*, Canadian Journal of Chemistry 89, 1150 (2011) (https://doi.org/10.1139/V11-050).

49. M. Kuisma, J. Ojanen, J. Enkovaara, and T.T. Rantala, *Kohn-Sham potential with discontinuity for Band gap materials*, Physical review B 82, 115106 (2010) (https://doi.org/10.1103/PhysRevB.82.115106).

50. L. Visscher, and K.G. Dyall, *Dirac-Fock atomic electronic structure calculations using different nuclear charge distributions*, Atomic Data and Nuclear Data Tables 67, 207 (1997) (https://doi.org/10.1006/adnd.1997.0751).

51. 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).

52. 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).

53. 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).

54. D. Koller, F. Tran, and P. Blaha, *Improving the Modified Becke-Johnson Exchange Potential.*, Physical Review B 83, 155109 (2012) (https://doi.org/10.1103/PhysRevB.85.155109).

55. R. A.Jishi, O. B. Ta, and A. Sharif, *Modeling of Lead Halide Perovskites for Photovoltaic Applications.*, Archive (http://arxiv.org/abs/1405.1706).

56. M. Raupach and R. Tonner, *A periodic energy decomposition analysis method for the investigation of chemical bonding in extended systems*, The Journal of Chemical Physics 142, 194105 (2015) (https://doi.org/10.1063/1.4919943).

57. X. Ren, P. Rinke, V. Blum, J. Wieferink, A. Tkatchenko, A. Sanfilippo, K. Reuter and M. Scheffler, *Resolution-of-identity approach to Hartree–Fock, hybrid density functionals, RPA, MP2 and GW with numeric atom-centered orbital basis functions*, New J. Phys. 14 053020 (https://doi.org/10.1088/1367-2630/14/5/053020).

58. J.A. Berger, P. Romaniello, R. van Leeuwen and P.L. de Boeij, *Performance of the Vignale-Kohn functional in the linear response of metals*, Phys. Rev. B 74, 245117 (https://doi.org/10.1103/PhysRevB.74.245117).

59. J.A. Berger, *Fully Parameter-Free Calculation of Optical Spectra for Insulators, Semiconductors, and Metals from a Simple Polarization Functional*, Phys. Rev. Lett. 115, 137402 (https://doi.org/10.1103/PhysRevLett.115.137402).

60. Z. Qian and G. Vignale, *Dynamical exchange-correlation potentials for an electron liquid*, Phys. Rev. B 65, 235121 (https://doi.org/10.1103/PhysRevB.65.235121).

61. S. Conti, R. Nifosì and M.P. Tosi, *The exchange - correlation potential for current-density functional theory of frequency-dependent linear response*, J. Phys. Condens. Matter 9, L475 (https://doi.org/10.1088/0953-8984/9/34/004).

62. J. Heyd, G.E. Scuseria and M. Ernzerhof, *Hybrid functionals based on a screened Coulomb potential*, J. Chem. Phys. 118, 8207 (2003) (https://doi.org/10.1063/1.1564060).

63. 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).

64. (a) Raupach and R. Tonner, unpublished. Please contact the authors of reference 56.

65. W. Setyawan and S. Curtarolo, *High-throughput electronic band structure calculations: Challenges and tools*, Computational Materials Science 49 (2010) 299–312 (https://doi.org/10.1016/j.commatsci.2010.05.010).

66. Rui Li, Jiaxing Zhang, Shimin Hou, Zekan Qian, Ziyong Shen, Xingyu Zhao, Zengquan Xue, *A corrected NEGF + DFT approach for calculating electronic transport through molecular devices: Filling bound states and patching the non-equilibrium integration*, Chemical Physics 336 (2007) 127-135 (https://doi.org/10.1016/j.chemphys.2007.06.011).

67. Zeng-hui Yang, Haowei Peng, Jianwei Sun, and John P. Perdew, *More realistic band gaps from meta-generalized gradient approximations: Only in a generalized Kohn-Sham scheme*, Physical Review B 93, 205205 (2016) (https://doi.org/10.1103/PhysRevB.93.205205).

68. C. J. O. Verzijl and J. M. Thijssen *DFT-Based Molecular Transport Implementation in ADF/BAND*, J. Phys. Chem. C, 2012, 116 (46), pp 24393–24412 (https://doi.org/10.1021/jp3044225).

69. F.L. Hirshfeld, *Bonded-atom fragments for describing molecular charge densities*, Theoretica Chimica Acta 44, 129 (1977) (https://doi.org/10.1007/BF00549096)

70. 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)

71. 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)

72. J.B. Krieger, Yan Li, G.J. Iafrate, *Derivation and application of an accurate Kohn-Sham potential with integer discontinuity*, Physics Letters A 8, 146 (1990) (https://doi.org/10.1016/0375-9601(90)90975-T)

CHAPTER

THIRTEEN

KEYWORDS

13.1 Links to manual entries

- *AIMCriticalPoints* (page 80)
- *ATensor* (page 67)
- *BField* (page 28)
- *BZPath* (page 78)
- *BandStructure* (page 76)
- *Basis* (page 32)
- *BeckeGrid* (page 43)
- *CPVector* (page 58)
- *Convergence* (page 51)
- *DIIS* (page 52)
- *DOS* (page 73)
- *DensityPlot* (page 106)
- *Dependency* (page 55)
- *EFG* (page 68)
- *EField* (page 28)
- *ESR* (page 68)
- *EffectiveMass* (page 70)
- *ElectronHole* (page 112)
- *EnforcedSpinPolarization* (page 21)
- *Fermi* (page 57)
- *FormFactors* (page 71)
- *Fragment* (page 81)
- *FuzzyPotential* (page 30)
- *Grid* (page 104)
- *GridBasedAIM* (page 79)
- *GrossPopulations* (page 75)
- *HubbardU* (page 19)
- *Integration* (page 45)
- *IntegrationMethod* (page 45)
- *KGrpX* (page 58)
- *KSpace* (page 39)
- *LDOS* (page 109)
- *MultiSecantConfig* (page 54)
- *NEGF* (page 96)
- *NMR* (page 69)
- *NOCVOrbitalPlot* (page 108)
- *NOCVdRhoPlot* (page 109)
- *NewResponse* (page 61)

- *NewResponseKSpace* (page 64)
- *NewResponseSCF* (page 62)
- *NuclearModel* (page 30)
- *NumericalQuality* (page 31)
- *Occupations* (page 112)
- *OldResponse* (page 65)
- *OrbitalPlot* (page 107)
- *OverlapPopulations* (page 75)
- *PEDA* (page 82)
- *PEDANOCV* (page 83)
- *PeriodicSolvation* (page 27)
- *PotentialNoise* (page 112)
- *RIHartreeFock* (page 48)
- *RadialDefaults* (page 44)
- *Relativity* (page 22)
- *ResponseInducedDensityPlot* (page 107)
- *Restart* (page 103)
- *SCF* (page 49)
- *Save* (page 110)
- *Screening* (page 56)
- *SoftConfinement* (page 36)
- *Solvation* (page 24)
- *StoreHamiltonian2* (page 101)
- *SubSymmetry* (page 111)
- *Tails* (page 55)
- *Unrestricted* (page 21)
- *UseSymmetry* (page 111)
- *ZlmFit* (page 45)

13.2 Summary of all keywords

`AIMCriticalPoints`

Type Block

Description Compute the critical points of the density (Atoms In Molecules). The algorithm starts from a regular mesh of points, and from each of these it walks towards its corresponding critical point.

`Enabled`

Type Bool

Default value False

Description Compute the critical points of the density (Atoms In Molecules). The algorithm starts from a regular mesh of points, and from each of these it walks towards its corresponding critical point.

`EqvPointsTol`

Type Float

Default value 0.27

Unit Bohr

Description If the distance between two critical points is smaller than this value, the two critical points are considered to be the same point.

GridPadding

Type Float

Default value 0.7

Unit Bohr

Description How much extra space is added to the starting guess domain in the search for the critical points

GridSpacing

Type Float

Default value 0.5

Unit Bohr

Description The distance between the initial trial points.

Allow

Type String

Recurring True

Description Debugging feature to let the program continue even when intermediate results seem to be wrong or very inaccurate

ATensor

Type Block

Description Hyperfine A-tensor.

Enabled

Type Bool

Default value False

Description Compute the hyperfine A-tensor.Note: Unrestricted calculation is required.

AtomType

Type Block

Recurring True

Description Explicit basis set definition for given atom type.

AutomaticGaussians

Type Non-standard block

Description Definition of the automatic gaussians

BasisFunctions

Type Non-standard block

Description Definition of the extra Slater-type orbitals

Dirac

Type Non-standard block

Description Specification of the numerical ('Herman-Skillman') free atom, which defines the initial guess for the SCF density, and which also (optionally) supplies Numerical Atomic Orbitals (NOs) as basis functions

FitFunctions

Type Non-standard block

Description Slater-type fit functions. Obsolete feature.

BandStructure

Type Block

Description Options for the calculation of the band structure.

Automatic

Type Bool

Default value True

Description If True, BAND will automatically generate the standard path through the Brillouin zone.If False BAND will use the user-defined path in BZPath.

DeltaK

Type Float

Default value 0.1

Unit 1/Bohr

Description Step (in reciprocal space) for band structure interpolation.Using a smaller number (e.g. 0.03) will result in smoother band curves at the cost of an increased computation time.

Enabled

Type Bool

Default value False

Description If True, Band will calculate the band structure and save it to file for visualization.

EnergyAboveFermi

Type Float

Default value 0.75

Unit Hartree

Description Bands with minimum energy larger then FermiEnergy + EnergyAboveFermi are not saved to file. Increasing the value of EnergyAboveFermi will result in more unoccupied bands to be saved to file for visualization.

EnergyBelowFermi

Type Float

Default value 10.0

Unit Hartree

Description Bands with maximum energy smaller then FermiEnergy - EnergyBelowFermi are not saved to file. Increasing the value of EnergyBelowFermi will result in more occupied core bands to be saved to file for visualization. Note: EnergyBelowFermi should be a positive number!

FatBands

Type Bool

Default value True

Description If True, BAND will compute the fat bands (only if BandStructure%Enabled is True).The Fat Bands are the periodic equivalent of the Mulliken population analysis.

UseSymmetry

Type Bool

Default value True

Description If True, only the irreducible wedge of the Wigner-Seitz cell is sampled.If False, the whole (inversion-unique) Wigner-Seitz cell is sampled.Note: The Symmetry key does not influence the symmetry of the band structure sampling.

Basis

Type Block

Description Definition of the basis set

ByAtomType

Type Non-standard block

Description Definition of the basis set for specific atom types (one definition per line). Format: 'AtomType Type=Type Core=Core'. Example: 'C.large_basis Type=TZ2P Core=None'

Core

Type Multiple Choice

Default value Large

Options [None, Small, Medium, Large]

Description Size of the frozen core.

Folder

Type String

Description Path to a folder containing the basis set files. This can be used for special user-defined basis sets. Cannot be used in combination with 'Type'

Type

Type Multiple Choice

Default value DZ

Options [SZ, DZ, DZP, TZP, TZ2P, QZ4P]

Description The basis sets to be used.

BeckeGrid

Type Block

Description Options for the numerical integration grid, which is a refined version of the fuzzy cells integration scheme developed by Becke.

AtomDepQuality

Type Non-standard block

Description One can define a different grid quality for each atom (one definition per line). Line format: 'AtomIndex Quality', e.g. '3 Good' means that a grid of Good quality will be used for the third atom in input order. If the index of an atom is not present in the AtomDepQuality section, the quality defined in the Quality key will be used

Quality

Type Multiple Choice

Default value Auto

Options [Auto, Basic, Normal, Good, VeryGood, Excellent]

Description Quality of the integration grid. For a description of the various qualities and the associated numerical accuracy see reference. If 'Auto', the quality defined in the 'NumericalQuality' will be used.

RadialGridBoost

Type Float

Default value 1.0

Description The number of radial grid points will be boosted by this factor. Some XC functionals require very accurate radial integration grids, so BAND will automatically boost the radial grid by a factor 3 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.

BField

Type Block

Description The effect of a magnetic filed can be approximated by the following potential: mu * sigma_i * B, where mu is the Bohr magneton, sigma_i are the Pauli matrices and B is the magnetic field

Bx

Type Float

Default value 0.0

Unit Tesla

Description Value of the x component of the BField

By

Type Float

Default value 0.0

Unit Tesla

Description Value of the y component of the BField

Bz

Type Float

Default value 0.0

Unit Tesla

Description Value of the z component of the BField

Dipole

Type Bool

Default value False

Description Use an atomic dipole as magnetic field instead of a uniform magnetic field.

DipoleAtom

Type Integer

Default value 1

Description Atom on which the magnetic dipole should be centered (if using the dipole option)

Method

Type Multiple Choice

Default value NR_SDOTB

Options [NR_SDOTB, NR_LDOTB, NR_SDOTB_LDOTB]

Description There are two terms coupling to an external magnetic field.One is the intrinsic spin of the electron, called S-dot-B, the other one is the orbital momentum call L-dot-B.The L.B is implemented non-relativistically, using GIAOs in the case of a homogeneous magnetic field (not for the dipole case).

Unit

Type Multiple Choice

Default value tesla

Options [tesla, a.u.]

Description Unit of magnetic filed. The a.u. is the SI version of a.u.

BZPath

Type Block

Description Definition of the user-defined path in the Brillouin zone for band structure plotting.

path

Type Non-standard block

Recurring True

Description Definition of the k-points in a path. The vertices of your path should be defined in fractional coordinates (wrt the reciprocal lattice vectors)

Comment

Type Non-standard block

Description The content of this block will be copied to the output header as a comment to the calculation.

Convergence

Type Block

Description Options and parameters related to the convergence behavior of the SCF procedure.

Criterion

Type Float

Description Criterion for termination of the SCF procedure. The default depends on the NumericalQuality and on the number of atoms in the system.

Degenerate

Type String

Default value default

Description Smooths (slightly) occupation numbers around the Fermi level, so as to insure that nearly-degenerate states get (nearly-) identical occupations. Be aware: In case of problematic SCF convergence the program will turn this key on automatically, unless the key 'Nodegenerate' is set in input. The smoothing depends on the argument to this key, which can be considered a 'degeneration width'. When the argument reads default, the program will use the value 1e-4 a.u. for the energy width.

ElectronicTemperature

Type Float

Default value 0.0

Unit a.u.

Description Simulates a finite-temperature electronic distribution using the defined energy. This may be used to achieve convergence in an otherwise problematically converging system. The energy of a finite-T distribution is different from the T=0 value, but for small T a fair approximation of the zero-T energy is obtained by extrapolation. The extrapolation energy correction term is printed with the survey of the bonding energy in the output file. Check that this value is not too large. Build experience yourself how different settings may affect the outcomes. Note: this key is meant to help you overcome convergence problems, not to do finite-temperature research! Only the electronic distribution is computed T-dependent, other aspects are not accounted for!

InitialDensity

Type Multiple Choice

Default value rho

Options [rho, psi]

Description The SCF is started with a guess of the density. There are the following choices RHO: the sum of atomic density. PSI: construct an initial eigensystem by occupying the atomic orbitals. The guessed eigensystem is orthonormalized, and from this the density is calculated/

LessDegenerate

Type Bool

Default value False

Description If smoothing of occupations over nearly degenerate orbitals is applied (see Degenerate key), then, if this key is set in the input file, the program will limit the smoothing energy range to 1e-4 a.u. as soon as the SCF has converged 'halfway', i.e. when the SCF error has decreased to the square root of its convergence criterion.

NoDegenerate

Type Bool

Default value False

Description This key prevents any internal automatic setting of the key DEGENERATE.

SpinFlip

Type String

Default value

Description List here the atoms for which you want the initial spin polarization to be flipped. This way you can distinguish between ferromagnetic and anti ferromagnetic states. Currently, it is not allowed to give symmetry equivalent atoms a different spin orientation. To achieve that you have to break the symmetry.

startwithmaxspin

Type Bool

Default value True

Description To break the initial perfect symmetry of up and down densities there are two strategies. One is to occupy the numerical orbitals in a maximum spin configuration. The alternative is to add a constant to the potential. See also Vsplit key.

CPVector

Type Integer

Default value 128

Description The code is vectorized and this key can be used to set the vector length

DensityPlot

Type Non-standard block

Description Plots of the density. Goes together with the Restart%DensityPlot and Grid keys.

Dependency

Type Block

Description Criteria for linear dependency of the basis and fit set

Basis

Type Float

Default value 1e-08

Description Criteria for linear dependency of the basis: smallest eigenvalue of the overlap matrix of normalized Bloch functions.

Core

Type Float

Default value 0.98

Description The program verifies that the frozen core approximation is reasonable, by checking the smallest value of the overlap matrix of the core (Bloch) orbitals against this criterion.

CoreValence

Type Float

Default value 1e-05

Description Criterion for dependency of the core functions on the valence basis. The maximum overlap between any two normalized functions in the two respective function spaces should not exceed 1.0-corevalence

Fit

Type Float

Default value 5e-06

Description Criterion for dependency of the total set of fit functions. The value monitored is the smallest eigenvalue of the overlap matrix of normalized Bloch sums of symmetrized fit functions.

DIIS

Type Block

Description Parameters for the DIIS procedure to obtain the SCF solution

Adaptable

Type Bool

Default value True

Description Change automatically the value of dimix during the SCF.

CHuge

Type Float

Default value 20.0

Description When the largest coefficient in the DIIS expansion exceeds this value, damping is applied

CLarge

Type Float

Default value 20.0

Description When the largest DIIS coefficient exceeds this value, the oldest DIIS vector is removed and the procedure re-applied

Condition

Type Float

Default value 1000000.0

Description The condition number of the DIIS matrix, the largest eigenvalue divided by the smallest, must not exceed this value. If this value is exceeded, this vector will be removed.

DiMix

Type Float

Default value 0.2

Description Mixing parameter for the DIIS procedure

NCycleDamp

Type Integer

Default value 1

Description Number of initial iterations where damping is applied, before any DIIS is considered

NVctrx

Type Integer

Default value 20

Description Maximum number of DIIS expansion vectors

Variant

Type Multiple Choice

Default value DIIS

Options [DIIS, LISTi, LISTb, LISTd]

Description Which variant to use. In case of problematic SCF convergence, first try MultiSecant, and if that does not work the LISTi is the advised method. Note: LIST is computationally more expensive per SCF iteration than DIIS.

DOS

Type Block

Description Density-Of-States (DOS) options

DeltaE

Type Float

Default value 0.005

Unit Hartree

Description Energy step for the DOS grid. Using a smaller value (e.g. half the default value) will result in a finer sampling of the DOS.

Enabled

Type Bool

Default value False

Description Whether or not to calculate the density of states.

Energies

Type Integer

Description Number of equidistant energy-values for the DOS grid. This keyword supersedes the 'DeltaE' keyword.

File

Type String

Description Write the DOS (plain text format) to the specified file instead of writing it to the standard output.

IntegrateDeltaE

Type Bool

Default value True

Description This subkey handles which algorithm is used to calculate the data-points in the plotted DOS. If true, the data-points represent an integral over the states in an energy interval. Here, the energy interval depends on the number of Energies and the user-defined upper and lower energy for the calculation of the DOS. The result has as unit [number of states / (energy interval * unit cell)]. If false, the data-points do represent the number of states for a specific energy and the resulting plot is equal to the DOS per unit cell (unit: [1/energy]). Since the resulting plot can be a wild function and one might miss features of the DOS due to the step length between the energies, the default is set to the integration algorithm.

Max

Type Float

Default value 0.75

Unit Hartree

Description Upper bound energy (with respect to the Fermi energy)

Min

Type Float

Default value -0.75

Unit Hartree

Description Lower bound energy (with respect to the Fermi energy)

StoreCoopPerBasPair

Type Bool

Default value False

Description Calculate the COOP (crystal orbital overlap population).

DosBas

Type Non-standard block

Description Used to specify the fragment basis for the DOS.

EffectiveMass

Type Block

Description In a semi-conductor, the mobility of electrons and holes is related to the curvature of the bands at the top of the valence band and the bottom of the conduction band.With the effective mass option, this curvature is obtained by numerical differentiation.The estimation is done with the specified step size, and twice the specified step size, and both results are printed to give a hint on the accuracy. The easiest way to use this key is to enabled it without specifying any extra options.

Enabled

Type Bool

Default value False

Description Compute the EffectiveMass.

KPointCoord

Type Float List

Unit 1/Bohr

Recurring True

Description Coordinate of the k-points for which you would like to compute the effective mass.

NumAbove

Type Integer

Default value 1

Description Number of bands to take into account above the Fermi level.

NumBelow

Type Integer

Default value 1

Description Number of bands to take into account below the Fermi level.

StepSize

Type Float

Default value 0.001

Description Size of the step taken in reciprocal space to perform the numerical differentiation

EFG

Type Block

Description The electronic charge density causes an electric field, and the gradient of this field couples with the nuclear quadrupole moment, that some (non-spherical) nuclei have and can be measured by several spectroscopic techniques. The EFG tensor is the second derivative of the Coulomb potential at the nuclei. For each atom it is a 3x3 symmetric and traceless matrix. Diagonalization of this matrix gives three eigenvalues, which are usually ordered by their decreasing absolute size and denoted as V_{xx}, V_{yy}, V_{zz}. The result is summarized by the largest eigenvalue and the asymmetry parameter.

Enabled

Type Bool

Default value False

Description Compute the EFG tensor (for nuclear quadrupole interaction).

EField

Type Block

Description Include a homogeneous, static, electric field in the z-direction (only possible for 0D, 1D or 2D periodic systems)

Ez

Type Float

Default value 0.0

Description Strength of the electric field, in units as selected with the EField unit key.

unit

Type Multiple Choice

Default value Volt/Angstrom

Options [Volt/Angstrom, a.u., Volt/Bohr, Volt/meter]

Description Unit of the electric field Ez

EigThreshold

Type Float

Default value 0.01

Description Threshold for printing the eigenvectors coefficients (Print Eigens)

ElectronHole

Type Block

Description Allows one to specify an occupied band which shall be depopulated, where the electrons are then moved to the Fermi level. For a spin-restricted calculation 2 electrons are shifted and for a spin-unrestricted calculation only one electron is shifted.

BandIndex

Type Integer

Description Which occupied band shall be depopulated.

SpinIndex

Type Integer

Description Defines the spin of the shifted electron (1 or 2).

EmbeddingPotential

Type Block

Description An external potential can be read in and will be added to the effective Kohn-Sham potential. It has to be on the becke grid

Filename

Type String

Default value

Description Name of the file containing the embedding potential.

PotentialName

Type String

Default value

Description Name of variable containing the potential.

EnforcedSpinPolarization

Type Float

Description Enforce a specific spin-polarization instead of occupying according to the aufbau principle. The spin-polarization is the difference between the number of alpha and beta electron.Thus, a value of 1 means that there is one more alpha electron than beta electrons.The number may be anything, including zero, which may be of interest when searching for a spin-flipped pair, that may otherwise end up in the (more stable) parallel solution.

ESR

Type Block

Description Zeeman g-tensor. The Zeeman g-tensor is implemented using two-component approach of Van Lenthe and co-workers in which the g-tensor is computed from a pair of spinors related to each other by time-reversal symmetry. Note: the following options are necessary for ESR: 'Relativistic zora spin' and 'Kspace 1'

Enabled

Type Bool

Default value False

Description Compute Zeeman g-tensor.The Zeeman g-tensor is implemented using two-component approach of Van Lenthe and co-workers in which the g-tensor is computed from a pair of spinors related to each other by time-reversal symmetry.Note: the following options are necessary for ESR: 'Relativistic zora spin' and 'Kspace 1'

Fermi

Type Block

Description Technical parameter used in determining the Fermi energy, which is carried out at each cycle of the SCF procedure.

Delta

Type Float

Default value 0.0001

Description Convergence criterion: upper and lower bounds for the Fermi energy and the corresponding integrated charge volumes must be equal within delta.

Eps

Type Float

Default value 1e-10

Description After convergence of the Fermi energy search procedure, a final estimate is defined by interpolation and the corresponding integrated charge volume is tested. It should be exact, to machine precision. Tested is that it deviates not more than eps.

MaxTry

Type Integer

Default value 15

Description Maximum number of attempts to locate the Fermi energy. The procedure is iterative in nature, narrowing the energy band in which the Fermi energy must lie, between an upper and a lower bound. If the procedure has not converged sufficiently within MaxTry iterations, the program takes a reasonable value and constructs the charge density by interpolation between the functions corresponding to the last used upper and lower bounds for the Fermi energy.

FormFactors

Type Integer

Default value 2

Description Number of stars of K-vectors for which the form factors are computed

Fragment

Type Block

Recurring True

Description Defines a fragment. You can define several fragments for a calculation.

AtomMapping

Type Non-standard block

Description Format 'indexFragAt indexCurrentAt'. One has to associate the atoms of the fragment to the atoms of the current calculation. So, for each atom of the fragment the indexFragAt has to be associated uniquely to the indexCurrentAt for the current calculation.

FileName

Type String

Description File name of the fragment. Absolute path or path relative to the executing directory.

Labels

Type Non-standard block

Description This gives the possibility to introduce labels for the fragment orbitals. See examples.

FuzzyPotential

Type Non-standard block

Description Atomic (fuzzy cell) based, external, electric potential. See example.

Grid

Type Block

Description Options for the regular grid used for plotting (e.g. density plot). Used ICW the restart option.

ExtendX

Type Float

Default value 0.0

Unit Bohr

Description Extend the default regular grid along the x-direction by the specified amount: [x_min, x_max] => [x_min - ExtendX/2, x_max + ExtendX/2].

ExtendY

Type Float

Default value 0.0

Unit Bohr

Description Extend the default regular grid along the y-direction by the specified amount: [y_min, y_max] => [y_min - ExtendY/2, y_max + ExtendY/2].

ExtendZ

Type Float

Default value 0.0

Unit Bohr

Description Extend the default regular grid along the z-direction by the specified amount: [z_min, z_max] => [z_min - ExtendZ/2, z_max + ExtendZ/2].

FileName

Type String

Default value

Description Read in the grid from a file. The file format of the grid is: three numbers per line (defining the x, y and z coordinates of the points).

Type

Type Multiple Choice

Default value coarse

Options [coarse, medium, fine]

Description The default regular grids.

UserDefined

Type Non-standard block

Description Once can define the regular grid specification in this block. See example.

GridBasedAIM

Type Block

Description Invoke the ultra fast grid based Bader analysis.

Enabled

Type Bool

Default value False

Description Invoke the ultra fast grid based Bader analysis.

Iterations

Type Integer

Default value 40

Description The maximum number of steps that may be taken to find the nuclear attractor for a grid point.

SmallDensity

Type Float

Default value 1e-06

Description Value below which the density is ignored. This should not be chosen too small because it may lead to unassignable grid points.

UseStartDensity

Type Bool

Default value False

Description Whether the analysis is performed on the startup density (True) or on the final density (False).

GrossPopulations

Type Non-standard block

Description Partial DOS (pDOS) are generated for the gross populations listed under this key. See example.

HubbardU

Type Block

Description Options for Hubbard-corrected DFT calculations.

Enabled

Type Bool

Default value False

Description Whether or not to apply the Hubbard Hamiltonian

LValue

Type String

Default value

Description For each atom type specify the l value (0 - s orbitals, 1 - p orbitals, 2 - d orbitals). A negative value is interpreted as no l-value.

PrintOccupations

Type Bool

Default value True

Description Whether or not to print the occupations during the SCF.

UValue

Type String

Default value

Description For each atom type specify the U value (in atomic units). A value of 0.0 is interpreted as no U.

Integration

Type Block

Description Options for the Voronoi numerical integration scheme. Deprecated. Use BeckeGrid instead.

AccInt

Type Float

Default value 3.5

Description General parameter controlling the accuracy of the Voronoi integration grid. A value of 3 would be basic quality and a value of 7 would be good quality.

IntegrationMethod

Type Multiple Choice

Default value Becke

Options [Becke, Voronoi]

Description Choose the real-space numerical integration method. Note: the Voronoi integration scheme is deprecated.

KGrpX

Type Integer

Default value 5

Description Absolute upper bound on the number of k-points processed together. This only affects the computational performance.

KSpace

Type Block

Description Options for the k-space integration (i.e. the grid used to sample the Brillouin zone)

Quality

Type Multiple Choice

Default value Auto

Options [Auto, GammaOnly, Basic, Normal, Good, VeryGood, Excellent]

Description Select the quality of the K-space grid used to sample the Brillouin Zone. If 'Auto', the quality defined in the 'NumericalQuality' will be used. If 'GammaOnly', only one point (the gamma point) will be used. The actual number of K points generated depends on this option and on the size of the unit cell. The larger the real space cell, the fewer K points will be generated.The CPU-time and accuracy strongly depend on this option.

Regular

Type Block

Description Options for the regular k-space integration grid.

NumberOfPoints

Type Integer List

Description Use a regular grid with the specified number of k-points along each reciprocal lattice vector.For 1D periodic systems you should specify only one number, for 2D systems two numbers, and for 3D systems three numbers.

Symmetric

Type Block

Description Options for the symmetric k-space integration grid.

KInteg

Type Integer

Description Specify the accuracy for the Symmetric method.1: absolutely minimal (only the G-point is used)2: linear tetrahedron method, coarsest spacing3: quadratic tetrahedron method, coarsest spacing4,6,... (even): linear tetrahedron method5,7.... (odd): quadratic methodThe tetrahedron method is usually by far inferior.

Type

Type Multiple Choice

Default value Regular

Options [Regular, Symmetric]

Description The type of k-space integration grid used to sample the Brillouin zone (BZ) used.'Regular': simple regular grid.'Symmetric': symmetric grid for the irreducible wedge of the first BZ (useful when high-symmetry points in the BZ are needed to capture the correct physics of the system, graphene being a notable example).

LDOS

Type Block

Description Local Density-Of-States information. This can be used to generate STM images in the Tersoff-Hamann approximation (see https://doi.org/10.1103/PhysRevB.31.805)

DeltaNeg

Type Float

Default value 0.0001

Unit Hartree

Description Lower bound energy (Shift-DeltaNeg)

DeltaPos

Type Float

Default value 0.0001

Unit Hartree

Description Upper bound energy (Shift+DeltaPos)

Shift

Type Float

Default value 0.0

Unit Hartree

Description The energy bias with respect to the Fermi level.

MolecularNMR

Type Block

Description Options for the calculations of the NMR shielding tensor for molecules, excluding periodic systems. Implements the Schreckenbach method like ADF.

Enabled

Type Bool

Default value False

Description Compute NMR shielding.

MultiSecantConfig

Type Block

Description Parameters for the Multi-secant SCF convergence method.

CMax

Type Float

Default value 20.0

Description Maximum coefficient allowed in expansion

InitialSigmaN

Type Float

Default value 0.1

Description This is a lot like a mix factor: bigger means bolder

MaxSigmaN

Type Float

Default value 0.3

Description Upper bound for the SigmaN parameter

MaxVectors

Type Integer

Default value 20

Description Maximum number of previous cycles to be used

MinSigmaN

Type Float

Default value 0.01

Description Lower bound for the SigmaN parameter

NEGF

Type Block

Description Options for the NEGF (non-equilibrium green function) transport calculation.

AlignChargeTol

Type Float

Default value 0.1

Description In an alignment run you want to get the number of electrons in the center right. This number specifies the criterion for that.

AlignmentFile

Type String

Default value

Description Band result file (.rkf) corresponding to the alignment calculation.

Alpha

Type Float

Default value 1e-05

Description A charge error needs to be translated in a potential shift. DeltaV = alpha * DeltaQ

ApplyShift1

Type Bool

Default value True

Description Apply the main shift, obtained from comparing matrix elements in the leads with those from the tight-binding run. Strongly recommended.

ApplyShift2

Type Bool

Default value True

Description Apply the smaller alignment shift. This requires an extra alignment run. Usually this shift is smaller.

AutoContour

Type Bool

Default value True

Description Use automatic contour integral.

BiasPotential

Type Float

Default value 0.0

Description Apply a bias potential (atomic units). Can be negative. One has to specify the ramp potential with the FuzzyPotential key. This is mostly conveniently done with the GUI.

BoundOccupationMethod

Type Integer

Default value 1

Description See text. Only relevant with NonEqDensityMethod equal 2 or 3.

CDIIS

Type Bool

Default value False

Description Make the normal DIIS procedure aware of the align charge error

CheckOverlapTol

Type Float

Default value 0.01

Description BAND checks how well the TB overlap matrix S(R=0) represents the overlap matrix in the lead region. Elements corresponding to the outer layer are neglected, because when using a frozen core they have bigger errors.

ContourQuality

Type Multiple Choice

Default value good

Options [basic, normal, good, verygood]

Description The density matrix is calculated numerically via a contour integral. Changing the quality influences the number of points. This influences a lot the performance.

DEContourInt

Type Float

Default value -1.0

Description The energy interval for the contour grid. Defaults depends on the contour quality

DERealAxisInt

Type Float

Default value -1.0

Description The energy interval for the real axis grid. Defaults depends on the contour quality.

DeltaPhi0

Type Float

Default value 0.0

Description Undocumented.

DeltaPhi1

Type Float

Default value 0.0

Description Undocumented.

DoAlignment

Type Bool

Default value False

Description Set this to True if you want to do an align run. Between the leads there should be lead material. The GUI can be of help here.

EMax

Type Float

Default value 5.0

Unit eV

Description The maximum energy for the transmission grid (with respect to the Fermi level of the lead)

EMin

Type Float

Default value -5.0

Unit eV

Description The minimum energy for the transmission grid (with respect to the Fermi level of the lead)

Eta

Type Float

Default value 1e-05

Description Small value used for the contour integral: stay at least this much above the real axis. This value is also used for the evaluation of the Transmission and dos.

IgnoreOuterLayer

Type Bool

Default value True

Description Whether or not to ignore the outer layer.

KT

Type Float

Default value 0.001

Description k-Boltzman times temperature.

LeadFile

Type String

Default value

Description File containing the tight binding representation of the lead.

NE

Type Integer

Default value 100

Description The number of energies for the transmission energy grid.

NonEqDensityMethod

Type Integer

Default value 1

Description See text.

SGFFile

Type String

Default value

Description The result from the SGF program. Contains the Fermi energy of the lead.

YContourInt

Type Float

Default value 0.3

Description The density is calculated via a contour integral. This value specifies how far above the real axis the (horizontal part of the) contour runs. The value is rounded in such a way that it goes exactly halfway between two Fermi poles. There is a trade off: making it bigger makes the integrand more smooth, but the number of enclosed poles increases. For low temperatures it makes sense to lower this value, and use a smaller deContourInt.

YRealaxisInt

Type Float

Default value 1e-05

Description The non-Equilibrium density is calculated near the real axis.

NewResponse

Type Block

Description The TD-CDFT calculation to obtain the dielectric function is computed when this block is present in the input. Several important settings can be defined here.

ActiveESpace

Type Float

Default value 5.0

Unit eV

Description Modifies the energy threshold (DeltaE^{max}_{thresh} = omega_{high} + ActiveESpace) for which single orbital transitions (DeltaEpsilon_{ia} = Epsilon_{a}^{virtual} - Epsilon_{i}^{occupied}) are taken into account.

ActiveXYZ

Type String

Default value t

Description Expects a string consisting of three letters of either 'T' (for true) or 'F' (for false) where the first is for the X-, the second for the Y- and the third for the Z-component of the response properties. If true, then the response properties for this component will be evaluated.

DensityCutOff

Type Float

Default value 0.001

Description For 1D and 2D systems the unit cell volume is undefined. Here, the volume is calculated as the volume bordered by the isosurface for the value DensityCutoff of the total density.

EShift

Type Float

Default value 0.0

Unit eV

Description Energy shift of the virtual crystal orbitals.

FreqHigh

Type Float

Default value 3.0

Unit eV

Description Upper limit of the frequency range for which response properties are calculated (omega_{high}).

FreqLow

Type Float

Default value 1.0

Unit eV

Description Lower limit of the frequency range for which response properties are calculated. (omega_{low})

NFreq

Type Integer

Default value 5

Description Number of frequencies for which a linear response TD-CDFT calculation is performed.

NewResponseKSpace

Type Block

Description Modify the details for the integration weights evaluation in reciprocal space for each single-particle transition. Only influencing the NewResponse code.

Eta

Type Float

Default value 1e-05

Description Defines the small, finite imaginary number i*eta which is necessary in the context of integration weights for single-particle transitions in reciprocal space.

SubSimp

Type Integer

Default value 3

Description determines into how many sub-integrals each integration around a k point is split. This is only true for so-called quadratic integration grids. The larger the number the better the convergence behavior for the sampling in reciprocal space. Note: the computing time for the weights is linear for 1D, quadratic for 2D and cubic for 3D!

NewResponseSCF

Type Block

Description Details for the linear-response self-consistent optimization cycle. Only influencing the NewResponse code.

Bootstrap

Type Integer

Default value 0

Description defines if the Berger2015 kernel (Bootstrap 1) is used or not (Bootstrap 0). If you chose the Berger2015 kernel, you have to set NewResponseSCF%XC to '0'. Since it shall be used in combination with the bare Coulomb response only. Note: The evaluation of response properties using the Berger2015 is recommend for 3D systems only!

COApproach

Type Bool

Default value True

Description The program automatically decides to calculate the integrals and induced densities via the Bloch expanded atomic orbitals (AO approach) or via the cyrstal orbitals (CO approach). The option COApproach overrules this decision.

COApproachBoost

Type Bool

Default value False

Description Keeps the grid data of the Crystal Orbitals in memory.Requires significantly more memory for a speedup of the calculation. One might have to use multiple computing nodes to not run into memory problems.

Criterion

Type Float

Default value 0.001

Description For the SCF convergence the RMS of the induced density change is tested. If this value is below the Criterion the SCF is finished.Furthermore, one can find the calculated electric susceptibility for each SCF step in the output and can therefore decide if the default value is too loose or too strict.

DIIS

Type Bool

Default value True

Description In case the DIIS method is not working, one can switch to plain mixing by setting DIIS to false.

LowFreqAlgo

Type Bool

Default value True

Description Numerically more stable results for frequencies lower than 1.0 eV. Note: for a graphene monolayer the conical intersection results in a very small band gap (zero band gap semi-conductor). This leads ta a failing low frequency algorithm. One can then chose to use the algoritm as originally proposed by Kootstra by setting the input value to *false*. But, this can result in unreliable results for frequencies lower than 1.0 eV!

Mixing

Type Float

Default value 0.2

Description Mixing value for the SCF optimization.

NCycle

Type Integer

Default value 20

Description Number of SCF cycles for each frequency to be evaluated.

XC

Type Integer

Default value 1

Description Influences if the bare induced Coulomb response (XC 0) is used for the effective, induced potential or the induced potential derived from the ALDA kernel as well (XC 1).

NMR

Type Block

Description Options for the calculations of the NMR shielding tensor.

Correction_r

Type Bool

Default value True

Description Undocumented.

Enabled

Type Bool

Default value False

Description Compute NMR shielding.

MS0

Type Float

Default value 0.01

Description Undocumented.

NMRAtom

Type Integer

Default value 0

Description The index of the atom atom (in input order) for which NMR should be computed.

Numeric

Type Bool

Default value False

Description Undocumented.

Original

Type Bool

Default value False

Description Undocumented.

Print_jp

Type Bool

Description Print paramagnetic current.

SuperCell

Type Bool

Default value True

Description This is the switch between the two methods, either the super cell (true), or the single-dipole method (false)

Test

Type Bool

Description Key for printing all intrinsic tensors.

Test_E

Type Bool

Description Test of energy levels.

Test_S

Type Bool

Description Test of overlap matrix.

UseSharedMemory

Type Bool

Default value True

Description Whether or not to use shared memory in the NMR calculation.

NOCVdRhoPlot

Type Non-standard block

Description Goes together with the Restart%NOCVdRhoPlot and Grid keys. See example.

NOCVOrbitalPlot

Type Non-standard block

Description Goes together with the Restart%NOCVOrbitalPlot and Grid keys. See example.

NuclearModel

Type Multiple Choice

Default value PointCharge

Options [PointCharge, Gaussian, Uniform]

Description Specify what model to use for the nucleus.For the Gaussian model the nuclear radius is calculated according to the work of Visscher and Dyall (L. Visscher, and K.G. Dyall, Dirac-Fock atomic electronic structure calculations using different nuclear charge distributions, Atomic Data and Nuclear Data Tables 67, 207 (1997))

NUElstat

Type Integer

Default value 50

Description Number of outward (parabolic) integration points (for elliptical integration of the electrostatic interaction)

NumericalQuality

Type Multiple Choice

Default value Normal

Options [Basic, Normal, Good, VeryGood, Excellent]

Description Set the quality of several important technical aspects of a BAND calculation (with the notable exception of the basis set). It sets the quality of: BeckeGrid (numerical integration), ZlmFit (density fitting), KSpace (reciprocal space integration), and SoftConfinement (basis set confinement). Note: the quality defined in the block of a specific technical aspects supersedes the value defined in NumericalQuality (e.g. if I specify 'NumericalQuality Basic' and 'BeckeGrid%Quality Good', the quality of the BeckeGrid will be 'Good')

NVElstat

Type Integer

Default value 80

Description Number of angular (elliptic) integration points (for elliptical integration of the electrostatic interaction)

Occupations

Type Non-standard block

Description Allows one to input specific occupations numbers. Applies only for calculations that use only one k-point (i.e. pseudo-molecule calculations). See example.

OldResponse

Type Block

Description Options for the old TD-CDFT implementation.

Berger2015

Type Bool

Default value False

Description Use the parameter-free polarization functional by A. Berger (Phys. Rev. Lett. 115, 137402). This is possible for 3D insulators and metals. Note: The evaluation of response properties using the Berger2015 is recommend for 3D systems only!

CNT

Type Bool

Description Use the CNT parametrization for the longitudinal and transverse kernels of the XC kernel of the homogeneous electron gas. Use this in conjunction with the NewVK option.

CNVI

Type Float

Default value 0.001

Description The first convergence criterion for the change in the fit coefficients for the fit functions, when fitting the density.

CNVJ

Type Float

Default value 0.001

Description the second convergence criterion for the change in the fit coefficients for the fit functions, when fitting the density.

Ebndtl

Type Float

Default value 0.001

Unit Hartree

Description the energy band tolerance, for determination which routines to use for calculating the numerical integration weights, when the energy band posses no or to less dispersion.

Enabled

Type Bool

Default value False

Description If true, the response function will be calculated using the old TD-CDFT implementation

Endfr

Type Float

Default value 3.0

Unit eV

Description The upper bound frequency of the frequency range over which the dielectric function is calculated

Isz

Type Integer

Default value 0

Description Integer indicating whether or not scalar zeroth order relativistic effects are included in the TDCDFT calculation. 0 = relativistic effects are not included, 1 = relativistic effects are included. The current implementation does NOT work with the option XC%SpinOrbitMagnetization equal NonCollinear

Iyxc

Type Integer

Default value 0

Description integer for printing yxc-tensor (see http://aip.scitation.org/doi/10.1063/1.1385370). 0 = not printed, 1 = printed.

NewVK

Type Bool

Description Use the slightly modified version of the VK kernel (see https://aip.scitation.org/doi/10.1063/1.1385370). When using this option one uses effectively the static option, even for metals, so one should check carefully the convergence with the KSPACE parameter.

Nfreq

Type Integer

Default value 5

Description the number of frequencies for which a linear response TD-CDFT calculation is performed.

QV

Type Bool

Description Use the QV parametrization for the longitudinal and transverse kernels of the XC kernel of the homogeneous electron gas. Use this in conjunction with the NewVK option. (see reference).

Shift

Type Float

Default value 0.0

Unit eV

Description energy shift for the virtual crystal orbitals.

Static

Type Bool

Description An alternative method that allows an analytic evaluation of the static response (normally the static response is approximated by a finite small frequency value). This option should only be used for non-relativistic calculations on insulators, and it has no effect on metals. Note: experience shows that KSPACE convergence can be slower.

Strtfr

Type Float

Default value 1.0

Unit eV

Description is the lower bound frequency of the frequency range over which the dielectric function is calculated.

OrbitalPlot

Type Non-standard block

Description Goes together with the Restart%OrbitalPlot and Grid keys. See Example.

OverlapPopulations

Type Non-standard block

Description Overlap population weighted DOS (OPWDOS), also known as the crystal orbital overlap population (COOP).

PEDA

Type Bool

Default value False

Description If present in combination with the fragment block, the decomposition of the interaction energy between fragments is invoked.

PEDANOCV

Type Block

Description If present in combination with the fragment blocks and the PEDA key, the decomposition of the orbital relaxation term is performed.

EigvalThresh

Type Float

Default value 0.001

Description The threshold controls that for all NOCV deformation densities with NOCV eigenvalues larger than EigvalThresh the energy contribution will be calculated and the respective pEDA-NOCV results will be printed in the output

Enabled

Type Bool

Default value False

Description If true in combination with the fragment blocks and the PEDA key, the decomposition of the orbital relaxation term is performed.

PeriodicSolvation

Type Block

Description Additional options for simulations of periodic structures with solvation.

NStar

Type Integer

Default value 4

Description This option, expecting an integer number (>2), handles the accuracy for the construction of the COMSO surface. The larger the given number the more accurate the construction.

RemovePointsWithNegativeZ

Type Bool

Default value False

Description Whether the COSMO surface is constructed on both sides of a surface.If one is only interested in the solvation effect on the upper side of a surface (in the Z direction), then this option should be set to 'True'

SymmetrizeSurfacePoints

Type Bool

Default value True

Description Whether or not the COSMO point should be symmetrized

PopThreshold

Type Float

Default value 0.01

Description Threshold for printing Mulliken population terms. Works with 'Print orbpop'

PotentialNoise

Type Float

Default value 0.0001

Description The initial potential for the SCF procedure is constructed from a sum-of-atoms density. Added to this is some small noise in the numerical values of the potential in the points of the integration grid. The purpose of the noise is to help the program break the initial symmetry, if that would lower the energy, by effectively inducing small differences between (initially) degenerate orbitals.

Print

Type String

Recurring True

Description One or more strings (separated by blanks) from a pre-defined set may be typed after the key. This induces printing of various kinds of information, usually only used for debugging and checking. The set of recognized strings frequently changes (mainly expands) in the course of software-developments. Useful arguments may be symmetry, and fit.

PropertiesAtNuclei

Type Non-standard block

Description A number of properties can be obtained near the nucleus. An average is taken over a tiny sphere around the nucleus. The following properties are available: vxc[rho(fit)], rho(fit), rho(scf), v(coulomb/scf), rho(deformation/fit), rho(deformation/scf).

RadialDefaults

Type Block

Description Options for the logarithmic radial grid of the basis functions used in the subprogram Dirac

NR

Type Integer

Default value 3000

Description Number of radial points. With very high values (like 30000) the Dirac subprogram may not converge.

RMax

Type Float

Default value 100.0

Unit Bohr

Description Upper bound of the logarithmic radial grid

RMin

Type Float

Default value 1e-06

Unit Bohr

Description Lower bound of the logarithmic radial grid

Relativity

Type Block

Description Options for relativistic effects.

Level

Type Multiple Choice

Default value None

Options [None, Scalar, Spin-Orbit]

Description None: No relativistic effects.Scalar: Scalar relativistic ZORA.This option comes at very little cost.SpinOrbit: Spin-orbit coupled ZORA.This is the best level of theory, but it is (4-8 times) more expensive than a normal calculation. Spin-orbit effects are generally quite small, unless there are very heavy atoms in your system, especially with p valence electrons (like Pb).See also the SpinOrbitMagnetization key.

ResponseInducedDensityPlot

Type Non-standard block

Description Goes together with Restart%ResponseInducedDensityPlot and Grid.

Restart

Type Block

Description Tells the program that it should restart with the restart file, and what to restart.

DensityPlot

Type Bool

Default value False

Description Goes together with the DensityPlot block and Grid blocks

File

Type String

Default value

Description Name of the restart file.

NOCVOrbitalPlot

Type Bool

Default value False

Description Goes together with the NOCVOrbitalPlot and Grid blocks.

NOCVdRhoPlot

Type Bool

Default value False

Description Goes together with the NOCVdRhoPlot and Grid blocks.

OrbitalPlot

Type Bool

Default value False

Description Goes together with the OrbitalPlot and Grid

ResponseInducedDensityPlot

Type Bool

Default value False

Description Goes together with the ResponseInducedDensityPlot and Grid blocks.

SCF

Type Bool

Default value False

Description Restart the SCF procedure.

UseDensityMatrix

Type Bool

Default value False

Description If set to True: For restarting the SCF the density matrix will be used. Requires you to set 'Save DensityMatrix' in the previous run.

RIHartreeFock

Type Block

Description The Hartree-Fock exchange matrix is calculated through a procedure known as Resolution of the Identity (RI). Here you can tweak various parameters of the procedure.

AtomDepQuality

Type Non-standard block

Description One can define a different fit-set quality for each atom. The syntax for this free block is 'iAtom quality', where iAtom is the index of the atom in input order.

DependencyThreshold

Type Float

Default value 0.001

Description To improve numerical stability, almost linearly-dependent combination of basis functions are removed from the Hartree-Fock exchange matrix. If the SCF does not converge or 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).

FitSetQuality

Type Multiple Choice

Default value Normal

Options [VeryBasic, Basic, Normal, Good, VeryGood, Excellent]

Description The auxiliary fit set employed in the RI scheme. This is an important aspect of the procedure, significantly affecting both accuracy and computation time. For SZ and DZ basis set a 'basic' FitSetQuality will suffice. For 'DZP' and 'TZP' a normal quality is recommended. For larger basis set, use either 'normal' or better FitSetQuality.

Quality

Type Multiple Choice

Default value Normal

Options [VeryBasic, Basic, Normal, Good, VeryGood, Excellent]

Description Accuracy of numerical integration and thresholds of the RI procedure.

Save

Type String

Recurring True

Description Save scratch files or extra data that would be otherwise deleted at the end of the calculation. e.g. 'TAPE10' (containing the integration grid) or 'DensityMatrix'

SCF

Type Block

Description Controls technical SCF parameters.

Eigenstates

Type Bool

Description The program knows two alternative ways to evaluate the charge density iteratively in the SCF procedure: from the P-matrix, and directly from the squared occupied eigenstates. By default the program actually uses both at least one time and tries to take the most efficient.

If present, Eigenstates turns off this comparison and lets the program stick to one method (from the eigenstates).

Iterations

Type Integer

Default value 100

Description The maximum number of SCF iterations to be performed.

Method

Type Multiple Choice

Default value DIIS

Options [DIIS, MultiSecant]

Description Choose the general scheme used to converge the density in the SCF. In case of scf problems one can try the MultiSecant alternative at no extra cost per SCF cycle. For more details see the DIIS and MultiSecantConfig block.

Mixing

Type Float

Default value 0.075

Description Initial 'damping' parameter in the SCF procedure, for the iterative update of the potential: new potential = old potential + mix (computed potential-old potential). Note: the program automatically adapts Mixing during the SCF iterations, in an attempt to find the optimal mixing value.

PMatrix

Type Bool

Description If present, evaluate the charge density from the P-matrix. See also the key Eigenstates.

Rate

Type Float

Default value 0.99

Description Minimum rate of convergence for the SCF procedure. If progress is too slow the program will take measures (such as smearing out occupations around the Fermi level, see key Degenerate of block Convergence) or, if everything seems to fail, it will stop

VSplit

Type Float

Default value 0.05

Description To disturb degeneracy of alpha and beta spin MOs the value of this key is added to the beta spin potential at the startup.

Screening

Type Block

Description For the periodic solvation potential and for the old (not default anymore) fitting method, BAND performs lattice summations which are in practice truncated. The precision of the lattice summations is controlled by the options in this block.

CutOff

Type Float

Description Criterion for negligibility of tails in the construction of Bloch sums. Default depends on Accuracy.

DMadel

Type Float

Description One of the parameters that define the screening of Coulomb-potentials in lattice sums. Depends by default on Accuracy, rmadel, and rcelx. One should consult the literature for more information

NoDirectionalScreening

Type Bool

Description Real space lattice sums of slowly (or non-) convergent terms, such as the Coulomb potential, are computed by a screening technique. In previous releases, the screening was applied to all (long-range) Coulomb expressions. Screening is only applied in the periodicity directions. This key restores the original situation: screening in all directions

RCelx

Type Float

Description Max. distance of lattice site from which tails of atomic functions will be taken into account for the Bloch sums. Default depends on Accuracy.

RMadel

Type Float

Description One of the parameters that define screening of the Coulomb potentials in lattice summations. Depends by default on Accuracy, dmadel, rcelx. One should consult the literature for more information.

SelectedAtoms

Type Integer List

Description With this key you can select atoms. This has an effect on a few of options, like NMR and EFG.

Skip

Type String

Recurring True

Description Skip the specified part of the Band calculation (expert/debug option).

SoftConfinement

Type Block

Description In order to make the basis functions more compact, the radial part of the basis functions is multiplied by a Fermi-Dirac (FD) function (this 'confinement' is done for efficiency and numerical stability reasons). A FD function goes from one to zero, controlled by two parameters. It has a value 0.5 at Radius, and the decay width is Delta.

Delta

Type Float

Unit Bohr

Description Explicitely specify the delta parameter of the Fermi-Dirac function (if not specified, it will be 0.1*Radius).

`Quality`

Type Multiple Choice

Default value Auto

Options [Auto, Basic, Normal, Good, VeryGood, Excellent]

Description In order to make the basis functions more compact, the radial part of the basis functions is multiplied by a Fermi-Dirac (FD) function (this 'confinement' is done for efficiency and numerical stability reasons). A FD function goes from one to zero, controlled by two parameters. It has a value 0.5 at Radius, and the decay width is Delta. This key sets the two parameters 'Radius' and 'Delta'.Basic: Radius=7.0, Delta=0.7;Normal: Radius=10.0, Delta=1.0;Good: Radius=20.0, Delta=2.0;VeryGood and Excellent: no confinement at all.If 'Auto', the quality defined in the 'NumericalQuality' will be used.

`Radius`

Type Float

Unit Bohr

Description Explicitely specify the radius parameter of the Fermi-Dirac function.

`Solvation`

Type Block

Description Options for the COSMO (Conductor like Screening Model) solvation model.

`CVec`

Type Multiple Choice

Default value EXACT

Options [EXACT, FITPOT]

Description Choose how to calculate the Coulomb interaction matrix between the molecule and the point charges on the surface:- EXACT: use exact density, and integrate against the potential of the point charges. This may have inaccuracies when integration points are close to the point charges.- FITPOT: evaluate the molecular potential at the positions of the point charges, and multiply with these charges.

`Charge`

Type Block

Description Select the algorithm to determine the charges.

`Conv`

Type Float

Default value 1e-08

Description Charge convergence threshold in iterative COSMO solution.

`Corr`

Type Bool

Default value True

Description Correct for outlying charge.

Iter

Type Integer

Default value 1000

Description Maximum number of iterations to solve COSMO equations.

Method

Type Multiple Choice

Default value CONJ

Options [CONJ, INVER]

Description INVER: matrix inversion, CONJ: biconjugate gradient method.The CONJ method is guaranteed to converge with small memory requirements and is normally the preferred method.

Enabled

Type Bool

Default value False

Description Use the Conductor like Screening Model (COSMO) to include solvent effects.

Radii

Type Non-standard block

Description The values are the radii of the atomic spheres. If not specified the default values are those by Allinge. Format: 'AtomType value'. e.g.: 'H 0.7'

SCF

Type Multiple Choice

Default value VAR

Options [VAR, PERT, NONE]

Description Determine the point charges either Variational (VAR) or after the SCF as a Perturbation (PERT).

Solvent

Type Block

Description Solvent details

Del

Type Float

Description Del is the value of Klamt's delta_sol parameter, only relevant in case of Klamt surface.

Emp

Type Float

Description Emp is the empirical scaling factor x for the energy scaling.

Eps

Type Float

Description User-defined dielectric constant of the solvent (overrides the Eps value of the solvent defined in 'Name')

Name

Type Multiple Choice

Default value Water

Options [AceticAcid, Acetone, Acetonitrile, Ammonia, Aniline, Benzene, BenzylAlcohol, Bromoform, Butanol, isoButanol, tertButanol, CarbonDisulfide, CarbonTetrachloride, Chloroform, Cyclohexane, Cyclohexanone, Dichlorobenzene, DiethylEther, Dioxane, DMFA, DMSO, Ethanol, EthylAcetate, Dichloroethane, EthyleneGlycol, Formamide, FormicAcid, Glycerol, HexamethylPhosphoramide, Hexane, Hydrazine, Methanol, MethylEthylKetone, Dichloromethane, Methylformamide, Methypyrrolidinone, Nitrobenzene, Nitrogen, Nitromethane, PhosphorylChloride, IsoPropanol, Pyridine, Sulfolane, Tetrahydrofuran, Toluene, Triethylamine, TrifluoroaceticAcid, Water]

Description Name of a pre-defined solvent. A solvent is characterized by the dielectric constant (Eps) and the solvent radius (Rad).

Rad

Type Float

Unit Angstrom

Description User-defined radius of the solvent molecule (overrides the Rad value of the solvent defined in 'Name').

Surf

Type Multiple Choice

Default value Delley

Options [Delley, Wsurf, Asurf, Esurf, Klamt]

Description Within the COSMO model the molecule is contained in a molecule shaped cavity.Select one of the following surfaces to define the cavity:- Wsurf: Van der Waals surface- Asurf: solvent accessible surface- Esurf: solvent excluding surface- Klamt: Klamt surface- Delley: Delley surface.

StopAfter

Type String

Default value BAND

Description Specifies that the program is stopped after execution of a specified program-part (subroutine).

StoreHamAsMol

Type Bool

Default value False

Description Undocumented, used for (at least) NEGF.

StoreHamiltonian

Type Bool

Description Undocumented.

StoreHamiltonian2

Type Bool

Default value False

Description determine the tight-binding representation of the overlap an fock matrix. Used for (at least) NEGF.

StrainDerivatives

Type Block

Description Undocumented.

Analytical

Type Bool

Default value False

Description Undocumented.

AnalyticalElectrostatic

Type Bool

Default value False

Description Undocumented.

Analyticalkinetic

Type Bool

Default value False

Description Undocumented.

Analyticalpulay

Type Bool

Default value False

Description Undocumented.

Analyticalxc

Type Bool

Default value False

Description Undocumented.

Cellpartitiondelta

Type Float

Default value 4.0

Description Undocumented.

Cellpartitioninterpolationcubic

Type Bool

Default value False

Description Undocumented.

Cellpartitioninterpolationmesh

Type Integer

Default value 100

Description Undocumented.

Cellpartitionversion

Type Integer

Default value 2

Description Undocumented.

Celltopoorder

Type Integer

Default value 20

Description Undocumented.

Centralizenaturallsg

Type Bool

Default value False

Description Undocumented.

Coreorthoption

Type Integer

Default value 2

Description Undocumented.

Fitrho0numintextrarad

Type Integer

Default value 0

Description Undocumented.

Fitrho0prune

Type Bool

Default value True

Description Undocumented.

Interpolatecellpartition

Type Bool

Default value False

Description Undocumented.

Kinviadagger

Type Bool

Default value False

Description Undocumented.

Lmaxmultipoleexpansion

Type Integer

Default value 4

Description Undocumented.

Naiveelstat

Type Bool

Default value False

Description Undocumented.

Numericaldefdef

Type Bool

Default value True

Description Undocumented.

Numericaldefdeflong

Type Bool

Default value False

Description Undocumented.

Numintextral

Type Integer

Default value 0

Description Undocumented.

Numintextrarad

Type Integer

Default value 0

Description Undocumented.

Pairgridlowerangularorder

Type Integer

Default value 5

Description Undocumented.

Pairgridradpointsincrease

Type Integer

Default value 0

Description Undocumented.

Partitionfunctiontol

Type Float

Default value 1e-08

Description Undocumented.

Prunelatticesummedgrid

Type Bool

Default value True

Description Undocumented.

Reduceaccuracylsg

Type Bool

Default value False

Description Undocumented.

Renormalizechargefitrho0

Type Bool

Default value False

Description Undocumented.

Shiftmultipoleorigin

Type Bool

Default value True

Description Undocumented.

Simplelatticesummedgrid

Type Bool

Default value False

Description Undocumented.

Skipinlgwsmodule

Type Bool

Default value True

Description Undocumented.

Subtractatomicxc

Type Bool

Default value False

Description Undocumented.

Usesymmetry

Type Bool

Default value False

Description Undocumented.

Usevstrainderrho

Type Bool

Default value False

Description Undocumented.

atomradiuslsg

Type Float

Default value 0.0

Description Undocumented.

fitrho0numintextral

Type Integer

Default value 0

Description Undocumented.

SubSymmetry

Type Integer List

Description The indices of the symmetry operators to maintain.

Tails

Type Block

Description Ignore function tails.

Bas

Type Float

Default value 1e-06

Description Cut off the basis functions when smaller than the specified threshold.

Title

Type String

Default value

Description Title of the calculation, which will be printed in the output file.

Unrestricted

Type Bool

Default value False

Description Controls wheather Band should perform a spin-unrestricted calculation. Spin-unrestricted calculations are computationally roughly twice as expensive as spin-restricted.

UnrestrictedOnlyReference

Type Bool

Default value False

Description Undocumented.

UnrestrictedReference

Type Bool

Default value False

Description Undocumented.

UnrestrictedStartup

Type Bool

Default value False

Description Undocumented.

UseSymmetry

Type Bool

Default value True

Description Whether or not to exploit symmetry during the calculation.

XC

Type Block

Description Exchange Correlation functionals

GLLBKParameter

Type Float

Default value 0.382

Description K parameter for the GLLB functionals. See equation (20) of the paper.

diracgga

Type String

Default value

Description GGA for the dirac .

dispersion

Type String

Default value DEFAULT

Description The dispersion correction model to be used.

gga

Type String

Default value NONE

Description GGA XC functional.

lda

Type String

Default value VWN

Description LDA XC functional.

libxc

Type String

Default value NONE

Description Functional using the LicXC library.

libxcdensitythreshold

Type Float

Default value 1e-10

Description Density threshold for LibXC functionals.

metagga

Type String

Default value NONE

Description MetaGG XC functional.

model

Type String

Default value LB94

Description Model potential.

spinorbitmagnetization

Type String

Default value collinearz

Description Type of Spin-Orbit magnetization.

tb_mbjafactor

Type Float

Default value -1.23456789

Description a parameter for the TB-MBJ model potential.

tb_mbjbfactor

Type Float

Default value -1.23456789

Description b parameter for the TB-MBJ model potential..

tb_mbjcfactor

Type Float

Default value -1.23456789

Description c parameter for the TB-MBJ model potential..

tb_mbjefactor

Type Float

Default value -1.23456789

Description e parameter for the TB-MBJ model potential..

usexcfun

Type Bool

Default value False

Description Whether ot not the XCFun library should be used.

xcfun

Type Bool

Default value False

Description Functional for the XCFun library.

ZlmFit

Type Block

Description Options for the density fitting scheme 'ZlmFit'.

AllowBoost

Type Bool

Default value True

Description Allow automatic atom-dependent tuning of maximum l of spherical harmonics expansion. Whether or not this boost is needed for a given atom is based on an heuristic estimate of how complex the density around that atom is.

AtomDepQuality

Type Non-standard block

Description One can specify different ZlmFit-quality for different atoms, The syntax for this free block is 'iAtom quality', where iAtom is the index of the atom in input order. For the atoms that are not present in the AtomDepQuality sub-block, the quality defined in the Quality key will be used.

DensityThreshold

Type Float

Default value 1e-07

Description Threshold below which the electron density is considered to be negligible.

FGaussianW

Type Float

Default value 1.0

Description Only for 3D periodic systems. Width of the Gaussian functions replacing the S and P Zlms for Fourier transform.

FGridSpacing

Type Float

Description Only for 3D periodic systems. Spacing for the Fourier grid. By default, this depends on the quality.

FKSpaceCutOff

Type Float

Description Only for 3D periodic systems. Cut-off of the grid in k-space for the Fourier transform.

FirstTopoCell

Type Integer

Default value 5

Description First cell for the topological extrapolation of the long range part of the Coulomb Potential.

LMargin

Type Integer

Default value 4

Description User-defined l-margin, i.e., l_max for fitting is max(lMargin + l_max_basis_function, 2*l_max_basis_function)

LastTopoCell

Type Integer

Default value 10

Description Last cell for the topological extrapolation of the long range part of the Coulomb Potential.

NumStarsPartitionFun

Type Integer

Default value 5

Description Number of cell stars to consider when computing the partition function.

OrderTopoTrick

Type Integer

Default value 3

Description Order of the topological extrapolation of the long range part of the Coulomb Potential.

PartitionFunThreshold

Type Float

Default value 0.0

Description Threshold for the partition functions: if an integration point has a partition function weight smaller than this threshold, it will be discarded.

Quality

Type Multiple Choice

Default value Auto

Options [Auto, Basic, Normal, Good, VeryGood, Excellent]

Description Quality of the density-fitting approximation. For a description of the various qualities and the associated numerical accuracy see reference. If 'Auto', the quality defined in the 'NumericalQuality' will be used.

INDEX

A

A-Tensor, 67
Accuracy, 30
AMS, 9
Analysis, 71
Atomic Charges, 79
Atoms, 11

B

Bader Analysis, 79
Band Gap, 78
Band Structure, 76
Basis Set, 31
basis set superposition error, 39
Becke Grid, 43
Brillouin zone (BZ), 39
Broken Symmetry, 121
BSSE, 39
Bulk Modulus, 59

C

Charge, 11
Charges, 79
CM5 (charge model 5), 79
Collinear, 13
COOP, 75
Coordinates, 11
COSMO, 22

D

Density Fitting, 45
Direct Method, 57
DOS, 73

E

EELS, 60
Effective Mass, 70
EFG, 68
Elastic Tensor, 59
Electric Field, 28
Electronic Transport, 89
ELF, 106
Entropy, 59
EPR, 67
ESR, 67
Exchange-Correlation Functionals, 13
Excited States, 112

F

Fat Bands, 76
Form Factors, 71
Fragments, 81
Free Energy, 59
Frequencies, 59
Frozen Core, 31

G

G-Tensor, 68
Geometry, 11
Geometry Optimization, 11
Geometry Relaxation, 11
GGA+U, 19

H

Hartree-Fock Exchange, 48
Hessian, 59
Hirshfeld Charges, 79
HSE, 18
HubbardU, 19

I

Internal Energy, 59

K

K-Space, 39

L

Lattice Vectors, 11
LDOS, 109
LIBXC, 17
Linear Transit, 11

M

Magnetic Field, 28

Magnetization, 13
Mobility, 70
Model Hamiltonian, 12
Molecular Dynamics, 11
Mulliken Analysis, 79

N

NEGF, 89
NMR, 69
Non-Collinear, 13
Normal Modes, 59
NQCC, 68
Nuclear Gradients, 12
Nuclear model, 30
Numerical Integration, 42

O

OEP, 20
OPWDOS, 75

P

PDOS, 74
PEDA, 82
PEDA-NOCV, 83
PES, 9
PES Scan, 11
Phonons, 59
Plotting Crystal Orbitals, 107
Plotting Densities, 106
Plotting NOCV Deformation Densities, 109
Plotting NOCV Orbitals, 108
Potential Energy Surface, 9
Properties, 58
Properties at Nuclei, 71

Q

Q-Tensor, 68
QTAIM, 79

R

Radial Grid, 44
Range-Separated Hybrids, 18
Reciprocal Space, 39
Recommendations, 113
Relativistic Effects, 22
Resolution of the Identity, 48
Restarts, 103

S

SCF Options, 49
Shear Modulus, 59
Shielding Tensor, 69
Single Point, 11
Solvation, 22
Solvent Effects, 22
Specific Heat, 59
Spin-orbit, 22
STM, 109
Stress Tensor, 12
Structure Relaxation, 11
Symmetry, 110

T

Task, 9
TDCDFT, 59
Technical precision, 115
Thermodynamic, 59
Tight binding, 101
Transition State, 11

U

Unrestricted calculation, 21

V

Voronoi Charges (VDD), 79
Voronoi Grid, 44

X

X-Ray, 71

Y

Young Modulus, 59

Z

Zero-point Energy, 59
ZlmFit, 45
ZORA, 22