Scientific Computing & Modelling

Examples BAND

ADF Program System
Release 2010

Scientific Computing & Modelling NV
Vrije Universiteit, Theoretical Chemistry
De Boelelaan 1083; 1081 HV Amsterdam; The Netherlands
E-mail: support@scm.com

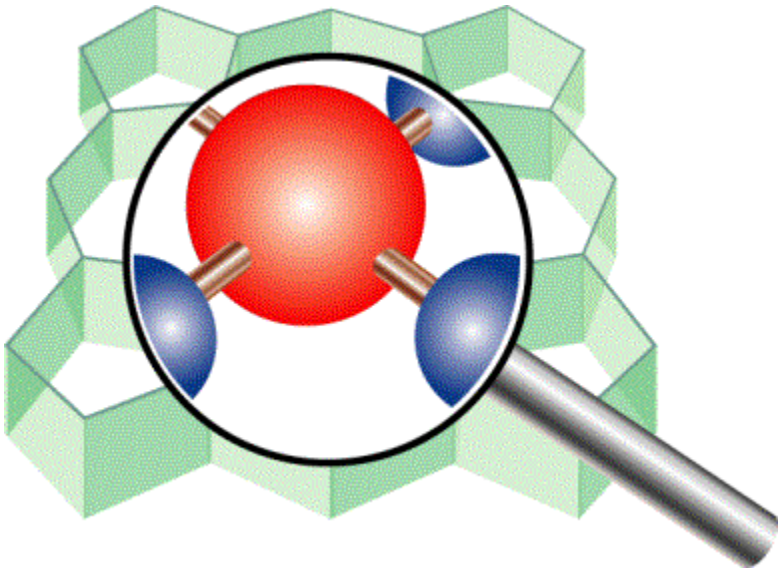


Table of Contents

General notes on the Examples

The ADF package contains a series of sample runs. 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 ADF and 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 a UNIX type OS is used and a hierarchical structure of *directories* is assumed.

All sample files are stored in subdirectories under $ADFHOME/examples/, where $ADFHOME is the main directory of the ADF package. There are two main subdirectories in examples/: adf/ for calculations with the molecular code ADF (and related utility programs) and band/ for calculations with the periodic structures code BAND. Each sample run has its own directory (under adf/ or band/ respectively). For instance, $ADFHOME/examples/band/NaCl/ contains an 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 Guides (ADF and BAND) 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.

Model Hamiltonians

NiCu surface alloy with GGA xc potential

Sample directory: band/NiCu_XC/

This is an important example featuring many important keywords.

The line:
Skip DOS

reduces the length of the output considerably.

A spin-unrestricted calculation can be done by using the keyword
Unrestricted

The Becke Perdew generalized gradient approximation (GGA) for the exchange-correlation functional is invoked through the XC block:

```
XC
 GGA Always Becke Perdew
End
```

The subkey ALWAYS indicates that the Becke-Perdew xc potential should be used during the SCF. If this is not specified, the GGA energy will be calculated post-SCF using the LDA density.

```
$ADFBIN/band << eor
Title Surface alloy: Cu slab with one surface Cu replaced by Ni (1:1)

Comment
 Technical
   Low quadratic K space integration
   Reasonable real space integration accuracy
 Features
   Lattice   : 2D
   Unit cell : 4 atoms, sqrt(2) x sqrt(2), 2 layers
   Basis     : NO+STO w/ core
   Options   : XC functional in SCF
End

Skip DOS

Convergence
 Degenerate default
End

DIIS
 NCycleDamp 15
 DiMix 0.25
End

SCF
 Mixing 0.10
End
```

```
Accuracy 3.5
Kspace 3

Unrestricted

XC
  GGA Always Becke Perdew
End

Units
 Length Angstrom
End

Define
latt=3.61 ! FCC lattice parameter
halflatt=latt/2
End

Lattice
latt 0    0
0    latt 0
End

Atoms Ni
0         0        0
End

Atoms Cu
:: layer 1
halflatt  halflatt 0
:: layer 2
0         halflatt -halflatt
halflatt  0        -halflatt
End

...
EndInput
eor
```

MetaGGA functionals

Sample directory: band/LiMetaGGA/

Example for a post SCF metaGGA calculation.

```
$ADFBIN/band << eor
Title Li bulk

KSpace 3

accuracy 5

DIIS
```

```
dimix 0.3
ncycledamp 0
end

scf
mixing 0.6
end

xc
metaGGA postscf
end

define
ha=6.60/2
end

Lattice
-ha ha ha
ha -ha ha
ha ha -ha
end

atoms Li
0.0
end

BasisDefaults
BasisType DZ
Core Large
end

end input
eor
```

Relativistic effects: Platinum slab

Sample directory: band/Pt_slab/

This example can of course be compared directly to the Cu slab. This example is important, as SCF convergence is frequently difficult in slab calculations. The specifications in the CONVERGENCE, SCF, and DIIS blocks are typical. Such settings are recommended in slab calculations with convergence problems.

The DEGENERATE subkey specifies that bands with the same energy should have the same occupation numbers. This helps SCF convergence. The same is true for the values for the MIXING subkey in the SCF block and the DIMIX subkey in the DIIS block. Please note that the recommended value for Mixing is approximately half of the value for Dimix.

Another important feature in BAND is that it enables relativistic treatments for heavy nuclei. Both the ZORA scalar relativistic option and spin-orbit effects have been implemented. The line

Relativistic ZORA SPIN

specifies that in this case both the scalar relativistic effects (ZORA) and spin-orbit effects (SPIN) will be taken into account. Whereas the ZORA keyword does not make the calculation much more time-consuming,

the same cannot be said for the spin-orbit option. Usually the ZORA keyword will give the most pronounced relativistic effects and the spin-orbit effects will be a fairly minor correction to that. We therefore recommend scalar ZORA as a good default method for treating heavy nuclei.

The DEPENDENCY keyword means that the calculation should continue even if the basis is nearly linearly dependent (as measured by the eigenvalues of the overlap matrix).

```
$ADFBIN/band << eor
Title Platinum slab

Comment
 Technical
   Low quadratic K space integration
   Low real space integration accuracy
 Features
   Lattice   : 2D
   Unit cell : 3 atoms, 1x1
   Basis     : NO+STO w/ core
   Options   : Spinorbit ZORA
End

Convergence
 Degenerate 1.0E-03
End

SCF
 Iterations 60
 Mixing 0.06
End

DIIS
 NCycleDamp 15
 DiMix 0.15
End

KSpace 3
Accuracy 3

Relativistic ZORA SPIN

Dependency Basis=1E-8

Define
latt=7.41
lvec=latt/SQRT(2.0)
ysh=lvec/SQRT(3.0)
dlay=latt/SQRT(3.0)
End

Lattice
 SQRT(3.0)*lvec/2.0  0.5*lvec
 SQRT(3.0)*lvec/2.0 -0.5*lvec
End

Atoms Pt
  0   0        0    :: layer 1
```

```
 -ysh 0.0     -dlay :: layer 2
  ysh 0.0 -2.0*dlay :: layer 3
End

...

END INPUT
eor
```

Spin polarization: antiferromagnetic iron

Sample directory: band/BetaIron/

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 antiferrmagnetic density.

For antiferromagnetic iron we need a larger unit cell of two atoms. Because these atoms appear to the program as symmetry equivalent we have to specify them as sepearate types.

```
$ADFBIN/band << eor
TITLE Beta iron

UNITS
    length Angstrom
    angle Degree
END

ATOMS Fe
0.0 0.0 0.0
end

! second iron as new type to break the symmetry
atoms Fe
-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

CONVERGENCE
CRITERION 1.0e-4
Degenerate default
SpinFlip 2  ! Flip (startup) spin density at second atom
END

BasisDefaults
BasisType DZ
Core Large
```

```
End

UNRESTRICTED

end input
eor
```

Graphene sheet with dispersion correction

Sample directory: band/Graphene_Dispersion/

A normal GGA would give no interaction between two graphene sheets.

Use the dispersion option in the XC keyblock.

In the first run we use BP86-D, and in the second BLYP-D3.

```
$ADFBIN/band << eor
TITLE Dispersion energy with two parallel graphene sheets

kspace 3
accuracy 4.5

XC
gga scf bp86
dispersion default
end

dependency basis=1e-6

UNITS
    length Angstrom
    angle Degree
END

ATOMS C
0.0 0.0 0.0
0.0 0.0 -3.355
1.23 0.7101408312 0.0
-1.23 -0.7101408311 -3.355
END

Lattice
 2.46 0.000000 0
 1.23 2.130422493 0
End

CONVERGENCE
CRITERION 1.0e-4
Degenerate default
END

DIIS
```

```
CLARGE 10
CHUGE 30
DIMIX 0.3
NVCTRX 10
NCYCLEDAMP 5
END

SCF
iterations 50
mixing 0.2
END

BasisDefaults
BasisType TZP
Core Large
End
end input
eor

rm RUNKF

$ADFBIN/band << eor
TITLE Grimme3 dispersion energy with two parallel graphene sheets

kspace 3
accuracy 4.5

XC
gga scf blyp
dispersion Grimme3
end

dependency basis=1e-6

Gradients
End

UNITS
    length Angstrom
    angle Degree
END

ATOMS C
0.0 0.0 0.0
0.0 0.0 -3.355
1.23 0.7101408312 0.0
-1.23 -0.7101408311 -3.355
END

Lattice
 2.46 0.000000 0
 1.23 2.130422493 0
End

CONVERGENCE
CRITERION 1.0e-4
Degenerate default
```

```
END

DIIS
CLARGE 10
CHUGE 30
DIMIX 0.3
NVCTRX 10
NCYCLEDAMP 5
END

SCF
iterations 50
mixing 0.2
END

BasisDefaults
BasisType TZP
Core Large
End

end input
eor
```

H on perovskite with the COSMO solvation model

Sample directory: band/HonPerovskite_Solvation/

We want to model H adsorption on a Perovskite surface in a solution, modelled by a COSMO surface.

We create only the COSMO surface above the slab with the RemovePointsWithNegativeZ option.

```
$ADFBIN/band << eor
TITLE Hydrogen on Perovksite wit solvation

PeriodicSolvation
nstar 3
SymmetrizeSurfacePoints true
RemovePointsWithNegativeZ true
end

screening
rmadel 30 ! to speed up the calculation
end

Solvation
Surf Esurf
charge method=inver
end

UNITS
    length Angstrom
    angle Degree
END
```

```
atoms H
0 0 0.9
end

ATOMS Ca
0.0 0.0 0.0
0.0 3.535533906 -3.535533906
END

ATOMS Ti
-2.5 -3.535533906 0.0
-2.5 4.440892099e-16 -3.535533906
END

ATOMS O
0.0 -3.535533906 0.0
2.5 1.767766953 -1.767766953
2.5 -1.767766953 -1.767766953
END

diis
dimix 0.3
nvctrx 20
end

Lattice
 5.0 0.000000 0
 0.000000 7.071067812 0
End

CONVERGENCE
CRITERION 1.0e-4
Degenerate default
END

BasisDefaults
BasisType SZ
Core Large
End

scf
iterations 30
end

XC
LDA SCF VWN
END

Dependency basis=1e-6
end input
eor
```

Precision and performance

Convenient way to specify a basis set

Sample directory: band/BasisDefaults/

This example shows some of the flexibility of the `BasisDefaults` key. The defaults are set to a DZ basis and a Large core. As the example shows, it is possible to override the defaults per atom type by specifying subkeys in `Atoms` blocks.

```
$ADFBIN/band << eor
Title CO + H2: fine tuning the basis defaults

KSpace 1

Accuracy 4

Define
far=20
End

Lattice
far
End

! So we have  cheap defaults
BasisDefaults
BasisType DZ
Core Large
End

Atoms C ! This C has no frozen core
0
Core None
End

Atoms O ! This O with a larger basis
0 2.13
BasisType TZ2P
End

Atoms H ! This one also with a larger basis
4 0
BasisType V
End

Atoms H ! Let us use the default settings for this atom
4 1.43
End

END INPUT
eor
```

Confinement and tails options for copper dimer

Sample directory: band/Cu2_Confine/

This example illustrates two keywords to speed up BAND calculations and reduce the required disk space to some extent, without significant loss of accuracy.

The TAILS keyword implies that cutoffs will be applied to the (basis) functions. Basis functions will be assumed to be zero outside the radius where the remaining relative norm of the function is smaller than 0.00001, in this example. The shape of the function is slightly modified, in a 'soft' way, beyond the radius where the relative norm of the function outside the radius is 0.01. The Coulomb option is always recommended in this line as it gives more accurate results for core functions.

The CONFINEMENT option can be specified for each atom type separately. It is based on a soft cut-off related to a distance criterion. In this sense it is quite different from the tails option. The CONFINEMENT option can be useful in itself, for avoiding near linear dependency in the basis. However, its use in making BAND calculations more efficient is primarily in combination with the tails option.

```
$ADFBIN/band << eor

Title Copper dimer - confinement and tails options

KSpace 1

Accuracy 4.5

Dependency Basis=1E-10 Fit=1E-7

Units
 Angle Radian
End

SCF
 Mixing 0.05
 Iterations 20
End

Convergence
 Degenerate default
End

DIIS
 NCycleDamp 0
 DiMix 0.15
End

Define
bbb=20
ccc=4.2
End

Lattice
  bbb bbb 0
 -bbb bbb 0
```

```
End

Atoms Cu
 0 0 0
 0 0 ccc
End

....
END INPUT
eor
```

The output file prints some information on how much was gained because of the tails and confinement options in various routines, for example:

Compression due to negligible tails in baspnt

max. percentage stored per node 98.23%

min. percentage stored per node 95.45%

Restarts

Restart the SCF

Sample directory: band/RestartSCF/

This very simple 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. You can, however, change more, as long as the basis set remains the same.

```
# ---------------------------- first run -------------------------

$ADFBIN/band << eor
Title B chain

KSpace 3

Accuracy 4

skip dos

XC
GGA Becke Perdew
END

UNRESTRICTED

SCF
 Mixing 0.4
 Iterations 40
End

Convergence
 Degenerate default
End

DIIS
 NCycleDamp 0
 DiMix 0.5
End

Define
ddd=4.0
ccc=1.5
End

Lattice
ddd
End

Atoms B
0
```

```
End

BasisDefaults
BasisType TZ2P
Core Large
End

END INPUT
eor

mv RUNKF BChain.runkf
rm Points

# ----------------------------- second run --------------------------

$ADFBIN/band << eor
Title B chain restart

KSpace 3

Accuracy 4

XC
GGA Becke Perdew
END

UNRESTRICTED

Restart BChain.runkf&
scf
end


SCF
 Mixing 0.4
 Iterations 40
End

Convergence
 Degenerate default
End

DIIS
 NCycleDamp 0
 DiMix 0.5
End

Define
ddd=4.0
zzz=3
End

Lattice
ddd
End

Atoms B
```

```
0
End

BasisDefaults
BasisType TZ2P
Core Large
End

END INPUT
eor
```

Properties on a grid

Sample directory: band/BeO_tape41/

If we save the RUNKF file of a calculation we can restart it to calculate things on a grid.

In the second run we restart from the file BeO.runkf. We specify to use a regular grid and ask the program to calculate a bunch of properties on that grid.

```
$ADFBIN/band << eor

Title BeO

Comment
in the "ideal" most symmetric configuration: u=3/8, and c=sqrt(8/3)*a
End

KSpace 3

accuracy 4

Dependency basis=1e-9 fit=1e-8

DIIS
dimix 0.2
ncycledamp 0
end

scf
mixing 0.4
end

xc
gga scf bp86
end

Define
uuu=3/8
aaa=5.10
ccc=sqrt(8/3)*aaa
End
```

```
Coordinates Natural

ATOMS Be
0 0 0
1/3 1/3 1/2
END

ATOMS O
0 0 uuu
1/3 1/3 uuu+1/2
END

Lattice
aaa 0 0
0.5*aaa 0.5*sqrt(3)*aaa 0
0 0 ccc
End

BasisDefaults
BasisType DZ
Core large
end
end input
eor

mv RUNKF BeO.runkf

$ADFBIN/band << eor
Title BeO

Comment
in the "ideal" most symmetric configuration: u=3/8, and c=sqrt(8/3)*a
End

Restart BeO.runkf &
DensityPlot
End

Grid
Type Coarse
End

DensityPlot
FITDENSITY_deformation_scf
FITDENSITY_total_scf
ATOMIC_density
ATOMIC_coulombPot
COULOMBPOTENTIAL_multipole_deformation_scf
COULOMBPOTENTIAL_scf
COULOMBPOTENTIAL_shortrange_deformation_scf
XCPOTENTIAL_scf
End

KSpace 3

accuracy 4
```

```
Dependency basis=1e-9 fit=1e-8

DIIS
dimix 0.2
ncycledamp 0
end

scf
mixing 0.4
end

xc
gga scf bp86
end

Define
uuu=3/8
aaa=5.10
ccc=sqrt(8/3)*aaa
End

Coordinates Natural

ATOMS Be
0 0 0
1/3 1/3 1/2
END

ATOMS O
0 0 uuu
1/3 1/3 uuu+1/2
END

Lattice
aaa 0 0
0.5*aaa 0.5*sqrt(3)*aaa 0
0 0 ccc
End

BasisDefaults
BasisType DZ
Core large
end

end input
eor
```

Strucure and Reactivity

NaCl: Bulk Crystal

Sample directory: band/NaCl/

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.

Calculations on periodic systems are carried out with the BAND program. Its input format has recently been changed substantially. It is now more similar in style to ADF. Old BAND input files are no longer compatible with the new version however.

The BAND input still follows slightly different conventions from the ADF input, for historical reasons.

The COMMENT keyword allows users to provide some information about the run which may be of use later. Usually a brief summary of the run is given here.

Numerical integration precision is controlled with the key Accuracy (in ADF: Integration)

The accuracy for integrals over the Brillouin Zone is set by the Kspace key. The latter should, generally, take as value an *odd* number (3, 5...) to invoke the accurate *quadratic* tetrahedron integration procedure. For *even* values it will revert to the *linear* tetrahedron method, which is almost always inferior in accuracy.

The key Angstroms specifies that geometric data, such as lattice constants are in angstrom units.

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 Coordinates NATURAL means that atomic positions are input as coefficients in terms of the lattice vectors, rather than as absolute (Cartesian) coordinate values.

For each of the atoms in the calculations, Na and Cl here, there must be data blocks to specify various items. First, their positions in the crystal unit cell (key Atoms). Second, the single isolated atom computation that will serve as start-up (Dirac). Third, any Slater-type orbital basis functions (BasisFunctions) for that atom. Fourth, the fit functions (FitFunctions) for the calculation of the Coulomb potential and. The third item (BasisFunctions) is optional and not present in this example.

It is recommended to include the numerical atomic orbitals that are computed by the Herman-Skillman type subprogram DIRAC as basis functions for the periodic structure calculation. This is effectuated by putting the word VALENCE in the Dirac data blocks. If that is done, additional STO basis functions (key BasisFunctions) are optional and are used to increase the basis set flexibility. In absence of the numerical (DIRAC/ VALENCE) orbitals, a minimal STO set is necessary of course, lest we wouldn't have any basis set at all.

In an equivalent ADF calculation, basis and fit functions would be provided through the Create runs, which pick up the basis and fit functions from a database file. The Create runs would also serve to provide the start-up density, as the DIRAC runs do in BAND.

The basis and fit sets that one has to insert into the BAND input files can be taken from the corresponding ADF database. Note, however, that ADF does not use any numerical orbitals. Since it is recommended to include such numerical orbitals in a BAND calculation, one has to adjust the STO-type basis set for BAND, in comparison with ADF, so as to avoid linear dependency with the numerical orbitals. As a general

guideline: for each of the included numerical orbitals (the occupied valence orbitals of the DIRAC calculation), one should remove one STO of the appropriate (n,l)-value. This keeps the overall size and flexibility of the basis at the same level and is usually sufficient to avoid dependency troubles.

The RUNKF key, early in the input, specifies that this standard result file from BAND must be saved under the name 't21.NaCl'. This file will be used in the follow-up calculation of Density-of-States properties.

Note, finally, that the data blocks of block type keys in the input for BAND end with a record 'END', as in ADF, whereas previously '**' was used in BAND to end a record.

```
$ADFBIN/band << eor
Title Title NaCl (from neutral atoms)

Comment
 Technical
   Hybrid K space integration (3D)
   Low real space integration accuracy
   Natural coordinates
   Lengths in Angstrom
   Parameters Dirac procedure
 Features
   Lattice   : 3D
   Unit cell : 2 atoms
   Basis     : NO w/ core
   Options   : Save restart file
End

Accuracy 3.5
Kspace 3

Units
 Length Angstrom
End

Lattice
0    2.75   2.75
2.75  0     2.75
2.75  2.75   0
End

ATOMS NA
0
End

Coordinates Natural
Atoms Cl
  .5  .5  .5
End

AtomType Na
Dirac   Na
  4 1
  Radial  2000
  RMin    1E-4
  RMax    60
  VALENCE
  1 0
```

```
  2 0
  2 1
  3 0 1.0
SubEnd


FitFunctions
  1 0 18.9
  2 0 30.3
  2 0 15.5
  3 0 14.9
  3 0  8.9
  4 0  7.8
  4 0  5.1
  4 0  3.3
  5 0  2.8
  5 0  1.9
  5 0  1.3
  2 1 14.3
  3 1  9.9
  4 1  6.7
  4 1  3.4
  5 1  2.4
  5 1  1.3
  3 2 10.5
  4 2  5.4
  5 2  3.0
  5 2  1.3
  4 3  5.8
  5 3  1.7
  5 4  2.0
SubEnd
End

AtomType Cl
Dirac  Cl
  5 3
  VALENCE
  1 0
  2 0
  2 1
  3 0
  3 1  5.0
SubEnd

FitFunctions
  1 0 29.1
  2 0 49.5
  2 0 26.1
  3 0 25.8
  3 0 15.8
  4 0 14.2
  4 0  9.4
  4 0  6.2
  5 0  5.4
  5 0  3.8
  5 0  2.6
```

```
  2 1 21.2
  3 1 16.5
  4 1 12.4
  4 1  6.8
  5 1  5.1
  5 1  3.1

  3 2 16.6
  4 2  9.4
  5 2  5.5
  5 2  2.6

  4 3  8.7
  5 3  3.3

  5 4  4.0
SubEnd
End

End Input
eor

mv RUNKF t21.NaCl

rm Points
```

The next run has largely the same input and provides a restart of the previous run.

The key RESTARTDOS 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).

```
$ADFBIN/band << eor
Title Title NaCl (from neutral atoms)   DOS analysis (restart)

Comment
 Technical
   Hybrid K space integration (3D)
   Low real space integration accuracy
   Natural coordinates
   Lengths in Angstrom
   Parameters Dirac procedure
 Features
   Lattice   : 3D
   Unit cell : 2 atoms
   Basis     : NO w/ core
   Options   : Use restart file for DOS
               Analysis: DOS, PDOS, COOP
End

Restart t21.NaCl &
DOS
End
```

```
Accuracy 3.5
Kspace 3

SCF
 Iterations 15
End

Units
 Length Angstrom
End

Lattice
  0    2.75   2.75
  2.75  0     2.75
  2.75  2.75   0
End

DOS
  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

Atoms NA
  0
End

Coordinates Natural

Atoms Cl
  .5  .5  .5
End
```

```
AtomType Na
Dirac   Na
  4 1
  Radial  2000
  RMin    1E-4
  RMax    60
  VALENCE
  1 0
  2 0
  2 1
  3 0 1.0
SubEnd


FitFunctions
  1 0 18.9
  2 0 30.3
  2 0 15.5
  3 0 14.9
  3 0  8.9
  4 0  7.8
  4 0  5.1
  4 0  3.3
  5 0  2.8
  5 0  1.9
  5 0  1.3
  2 1 14.3
  3 1  9.9
  4 1  6.7
  4 1  3.4
  5 1  2.4
  5 1  1.3
  3 2 10.5
  4 2  5.4
  5 2  3.0
  5 2  1.3
  4 3  5.8
  5 3  1.7
  5 4  2.0
SubEnd
End

AtomType Cl
Dirac  Cl
  5 3
  VALENCE
  1 0
  2 0
  2 1
  3 0
  3 1  5.0
SubEnd

FitFunctions
  1 0 29.1
  2 0 49.5
  2 0 26.1
```

```
   3 0 25.8
   3 0 15.8
   4 0 14.2
   4 0  9.4
   4 0  6.2
   5 0  5.4
   5 0  3.8
   5 0  2.6

   2 1 21.2
   3 1 16.5
   4 1 12.4
   4 1  6.8
   5 1  5.1
   5 1  3.1

   3 2 16.6
   4 2  9.4
   5 2  5.5
   5 2  2.6

   4 3  8.7
   5 3  3.3

   5 4  4.0
SubEnd
End

End Input
eor
```

Finally, we copy the contents of the DOS result file to standard output

```
echo Contents of DOS file
cat NaCl.dos
```

Polyacetylene polymer calculation

Sample directory: band/CnHn/

This example illustrates how a one-dimensional periodic system can be treated by specifying only one lattice vector. It further shows how variables can be defined with the DEFINE keyword. The rest more or less speaks for itself. The Kspace integration is taken very accurate, whereas real space integration (ACCURACY keyword) is not so accurate.

Here and in the following BAND examples, we will leave out some space consuming parts of the input file which have been discussed already. Please check the actual input files if you wish to repeat one of the calculations.

```
$ADFBIN/band << eor

Title Polymer

Comment
```

```
 Technical
   Quadratic k space integration (1D)
   Low real space integration accuracy
   Definitions of variables
 Features
   Lattice   : 1D, polymer
   Unit cell : 4 atoms
   Basis     : NO+STO w/ core
End

Kspace 5
Accuracy 3

Units
 Length Angstrom
 Angle  Radian
End

Define
  dCCd=1.3386
  dCCs=1.4510
  dCH=1.0770
  aCCC=124.5/180*pi
  arC=aCCC-pi/2
  aCCH=119.2/180*pi ! double bonded CC
  arH=aCCH-pi/2
End

Lattice
dCCd+sin(arC)*dCCs cos(arC)*dCCs 0.0
End


Atoms C
  dCCd/2 0.0 0.0
 -dCCd/2 0.0 0.0
End

Atoms H
  dCCd/2+sin(arH)*dCH -cos(arH)*dCH 0.0
 -dCCd/2-sin(arH)*dCH  cos(arH)*dCH 0.0
End
```

A larger unit cell can of course be specified as well. In the second part of the example a supercell of 5 units is used. Another new feature introduced in this example is the TAILS keyword, which similar to ADF implies that distance cut-offs are applied to make the calculation cheaper. At the moment no big gains are yet to be expected from this, but this situation is expected to change in future versions of the code.

in the BASIS key. This subkey is actually mandatory at the moment if the TAILS keyword is used.

```
$ADFBIN/band << eor
Title Polymer with big unit cell (5 units)

Comment
 Technical
   Low quadratic k space integration (1D)
   Low real space integration accuracy
```

```
   Definitions of variables
 Features
   Lattice   : 1D, polymer
   Unit cell : 4 atoms
   Basis     : NO+STO w/ core
End

Kspace 3
Accuracy 3

Units
 Length Angstrom
 Angle  Radian
End

Tails Bas=1E-2

Define
  dCCd=1.3386
  dCCs=1.4510
  dCH=1.0770
  aCCC=124.5/180*pi
  arC=aCCC-pi/2
  aCCH=119.2/180*pi ! double bonded CC
  arH=aCCH-pi/2
  Latx(nlatt)=nlatt*(dCCd+sin(arC)*dCCs)
  Laty(nlatt)=nlatt*(cos(arC)*dCCs)
  Laty(nlatt)=nlatt*(cos(arC)*dCCs)
End

Lattice
Latx(5) Laty(5) 0.0
End

Atoms C
  dCCd/2 0.0 0.0
 -dCCd/2 0.0 0.0
  dCCd/2+Latx(1) Laty(1) 0.0
 -dCCd/2+Latx(1) Laty(1) 0.0
  dCCd/2+Latx(2) Laty(2) 0.0
 -dCCd/2+Latx(2) Laty(2) 0.0
  dCCd/2+Latx(3) Laty(3) 0.0
 -dCCd/2+Latx(3) Laty(3) 0.0
  dCCd/2+Latx(4) Laty(4) 0.0
 -dCCd/2+Latx(4) Laty(4) 0.0
End

Atoms H
  dCCd/2+sin(arH)*dCH -cos(arH)*dCH 0.0
 -dCCd/2-sin(arH)*dCH  cos(arH)*dCH 0.0
  dCCd/2+sin(arH)*dCH+Latx(1.0) -cos(arH)*dCH+Laty(1.0) 0.0
 -dCCd/2-sin(arH)*dCH+Latx(1.0)  cos(arH)*dCH+Laty(1.0) 0.0
  dCCd/2+sin(arH)*dCH+Latx(2.0) -cos(arH)*dCH+Laty(2.0) 0.0
 -dCCd/2-sin(arH)*dCH+Latx(2.0)  cos(arH)*dCH+Laty(2.0) 0.0
  dCCd/2+sin(arH)*dCH+Latx(3.0) -cos(arH)*dCH+Laty(3.0) 0.0
 -dCCd/2-sin(arH)*dCH+Latx(3.0)  cos(arH)*dCH+Laty(3.0) 0.0
  dCCd/2+sin(arH)*dCH+Latx(4.0) -cos(arH)*dCH+Laty(4.0) 0.0
```

```
 -dCCd/2-sin(arH)*dCH+Latx(4.0)  cos(arH)*dCH+Laty(4.0) 0.0
End
```

Pd bulk

Sample directory: band/Pd/

The main things to note are the specification of a scalar relativistic ZORA calculation, and the fact that symmetry information will be printed on output.

This example also shows incidentally how to override optimization memory parameters KGRPX and CPVector.

```
$ADFBIN/band << eor
Title Pd bulk

Comment
 Technical
   Hybrid k space integration (3D)
   Reasonable real space integration accuracy
   Definitions of variables
 Features
   Lattice   : 3D
   Unit cell : 1 atom
   Basis     : NO+STO w/ core
   Options   : spin restricted, scalar relativistic,
               numerical fit functions
               Full symmetry
End

KGRPX 2
CPVector 256
Kspace 5
Accuracy 4.0

Print SYMMETRY

Relativistic ZORA

Define
 halflatt=7.35/2
End

Lattice :: FCC
  0        halflatt halflatt
  halflatt 0        halflatt
  halflatt halflatt 0
End

Atoms Pd
  0.0
End
```

```
...
END INPUT
eor
```

Ne bulk

Sample directory: band/Ne/

```
$ADFBIN/band << eor
Title Ne matrix 4.2 K

Comment
 Technical
   Hybrid k space integration (3D) (high)
   Reasonable real space integration accuracy
   Definitions of variables
 Features
   Lattice   : 3D
   Unit cell : 1 atom
   Basis     : NO+STO w/o core
End

Kspace 9
Accuracy 4.0

Units
 Length Angstrom
End

Define
 halflatt=4.43/2
End

Lattice :: FCC
  0        halflatt halflatt
  halflatt 0        halflatt
  halflatt halflatt 0
End

Atoms Ne
  0.0 0.0 0.0
End

AtomType Ne
Dirac Ne
 3 0
 Valence
 1S
 2S
 2P
SubEnd

BasisFunctions
```

```
 1S  12.45
 2S   3.65
 2S   1.55
 2P   1.45
 2P   5.50

 3D   2.0
 4F   3.0
SubEnd

FitFunctions
 3S         2.9000
 3S         4.0900
 3S         5.7700
 3S         8.1400
 2S         8.3300
 2S        12.5200
 2S        18.8100
 1S        17.2000
 3P         3.1000
 3P         4.8000
 3P         7.4200
 2P         8.4100
 2P        14.1000
 3D         2.9000
 3D         5.6700
 3D        11.1000
 4F         3.9500
 4F         8.0000
 5G         5.0000
SubEnd
End

END INPUT
eor
```

Atomic energies

Sample directory: band/H_ref/

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

Only the first run is given here:

```
$ADFBIN/band << eor
Title Formation energy of the H-atom w.r.t. spherical atom
```

```
Comment
 Technical
   Quadratic K space integration
   High real space integration accuracy
 Features
   Lattice   : 2D, quasi atom
   Unit cell : 1 atom, 1x1
   Basis     : NO+STO
End

Kspace 5
Accuracy 5.0

Convergence
 Criterion 1E-6
End

Define
 far=20.0
End

Lattice
 far 0.0 0.0
 0.0 far 0.0
End

Atoms H
 0.0 0.0 0.0
End

AtomType H
Dirac H
1  0
VALENCE
1S 1
SubEnd

BasisFunctions
1S   1.58
2P   1.0
SubEnd

FitFunctions
 1S   3.16
 1S   2.09
 1S   1.38
 2S   1.50
 2P   4.00
 2P   2.65
 2P   1.75
 3D   4.00
 3D   2.50
 4F   3.00
 5G   4.00
SubEnd

End
```

```
END INPUT
eor
```

Hydrogen on Pt surface

Sample directory: band/H_on_Pt/

This example is in many ways similar to the Pt slab example. We refer to the explanations in that example for details. Suffice it to say here that the convergence criteria have again been chosen such that also difficult slab calculations have a good chance to converge. Further, this calculation is once again at the spin-orbit relativistic level. The geometry is such that two layers of Pt are covered by a hydrogen layer.

```
$ADFBIN/band << eor
Title Hydrogen on platinum

Comment
 Technical
   Low quadratic K space integration
   Low real space integration accuracy
 Features
   Lattice   : 2D
   Unit cell : 4 atoms, 1x1
   Basis     : NO+STO
   Options   : Spinorbit ZORA
End

SCF
 Mixing 0.1
 Iterations 100
End

Convergence
 Degenerate default
 Criterion 1e-6
End

DIIS
 NCycleDamp 0
 DiMix 0.15
End

KSpace 3
Accuracy 3

Relativistic ZORA SPIN

Dependency Basis=1E-8
Define
latt=7.41
lvec=latt/SQRT(2.0)
ysh=lvec/SQRT(3.0)
dlay=latt/SQRT(3.0)
height=3.0
```

```
End

Lattice
 SQRT(3.0)*lvec/2.0  0.5*lvec
 SQRT(3.0)*lvec/2.0 -0.5*lvec
End

Atoms Pt
  0   0        0    :: layer 1
 -ysh 0.0     -dlay :: layer 2
End

Atoms   H
 0.0 0.0 height
End
...

END INPUT
eor
```

Calculating the atomic forces

Sample directory: band/BNForce/

This example shows how to calculate the gradient of the energy with respect to atomic displacements, using the GRADIENTS key block.

```
$ADFBIN/band << eor
Title BN zincblende structure (force calculation)

comment
  Calculate Atomic Forces (energy gradient)
  Frozen core approximation
  TZ2P basis
  A/K -> 5/3
  2 atoms per unit cell
end

dependency basis=1e-8

Kspace 3
Accuracy 5.0

Units
  Length Angstrom
End

Define
   a = 3.60
End

GRADIENTS
END
```

```
Lattice
  0.0   0.5*a 0.5*a
  0.5*a 0.0   0.5*a
  0.5*a 0.5*a 0.0
End

Atoms B
  0.0 0.0 0.0
end

Atoms N
  0.2404*a 0.2404*a 0.2404*a
end

BasisDefaults
BasisType TZ2P
Core Large
End

END INPUT
eor
```

In the output you will find the result

```
 FINAL GRADIENTS
    1  B  0.019676  0.019676  0.019676
    2  N -0.019677 -0.019677 -0.019677
```

Optimizing the geometry

Sample directory: band/H2BulkGeo/

This example shows how to optimize the geometry, using the GEOOPT key block, including a restart of a previous optimization. It consists of two runs (The example is designed for speed and not for accuracy.)

```
$ADFBIN/band << eor
Title H2 bulk geometry optimization
Comment
    Optimized for speed / Accuracy low
    Basis very small
    Demonstrates the GeoOpt block
    SCF iterations limited to make it faster
End

KSpace 2

Accuracy 3.1

screening
RMadel 9
end

Dependency Basis=1e-8
```

```
GeoOpt
Iterations 5
Converge Grad=1e-6
RestartSCF true
End

SCF
 Mixing 0.2
 Iterations 2
End

Convergence
 Degenerate default
End

DIIS
 NCycleDamp 0
 DiMix 0.5
End

Units
 Angle Radian
End

Define
aaa=1
End

ATOMS H
0
aaa
End

Lattice
 5.0*aaa  0 0
0 5.0*aaa 0
0 0 5.0*aaa
End

AtomType H

DIRAC H
 1  0
 VALENCE
 1S 1
SubEnd

BasisFunctions
 1S   0.69
 2P   1.25
SubEnd

FitFunctions
 1S   3.16
 1S   2.09
 1S   1.38
```

```
 2S   1.50
 2P   4.00
 2P   2.65
 2P   1.75
 3D   4.00
 3D   2.50
:: 4F   3.00
:: 5G   4.00
SubEnd

End

END INPUT
eor

mv RUNKF H2BulkGeo.runkf

rm Points
```

Note that the RUNKF file is saved. In the next run we use the result file to continue the geometry optimization

```
# ----------------------------- second run -------------------------

$ADFBIN/band << eor
Title H2 bulk geometry optimization
Comment
    Restarting the geometry optimization
    Optimized for speed / Accuracy low
    Basis very small
    Demonstrates the GeoOpt block
    SCF iterations limited to make it faster
End

KSpace 2

Accuracy 3.1

screening
RMadel 9
end

Dependency Basis=1e-8

Restart H2BulkGeo.runkf&
GeometryOptimization
end

GeoOpt
Iterations 5
Converge Grad=1e-6
RestartSCF true
End

SCF
 Mixing 0.2
:: Iterations 2
```

```
End

Convergence
 Degenerate default
End

DIIS
 NCycleDamp 0
 DiMix 0.5
End

Units
 Angle Radian
End

Define
aaa=1
End

ATOMS H
0
aaa
End

Lattice
 5.0*aaa  0 0
0 5.0*aaa 0
0 0 5.0*aaa
End

AtomType H

DIRAC H
 1  0
 VALENCE
 1S 1
SubEnd

BasisFunctions
 1S   0.69

2P   1.25
SubEnd

FitFunctions
 1S   3.16
 1S   2.09
 1S   1.38
 2S   1.50
 2P   4.00
 2P   2.65
 2P   1.75
 3D   4.00
 3D   2.50
:: 4F   3.00
:: 5G   4.00
SubEnd
```

```
End

END INPUT
eor
```

Nuclear gradient with a MetaGGA functional

Sample directory: band/CO_MetaGGA_Force/

This example shows how to calculate the gradient of the energy with respect to atomic displacements for a metaGGA functional, using the GRADIENTS and XC key block.

```
$ADFBIN/band << eor
Title CO chain

KSpace 3

accuracy 5

DIIS
dimix 0.2
ncycledamp 0
end

scf
mixing 0.3
end

xc
metaGGA scf force tpss
end

gradients
end

define
ha=6.60/2
end

Lattice
6.0 0 0
end

atoms C
0.0
end

atoms O
2.20
end
```

```
BasisDefaults
BasisType DZ
Core Large
end

end input
eor
```

Frequency run and transition state search

Sample directory: band/H2SlabFreqTS/

First run: H_2 slab frequency calculation near minimum. Second run: transition state search, using as restart the results of the first run.

```
$ADFBIN/band << eor
Title H2 slab frequency calculation near minimum
Comment
    Optimized for speed / Accuracy low
    Basis very small
    Demonstrates the frequencies runtype
End

KSpace 2

Accuracy 4

screening
RMadel 9
end

Dependency Basis=1e-8

RunType
 Frequencies
End

GeoOpt
RestartSCF true
End

SCF
 Mixing 0.2
End

Convergence
 Degenerate default
End

DIIS
 NCycleDamp 0
 DiMix 0.5
End
```

```
Units
 Length Angstrom
 Angle Radian
End

ATOMS H
:: TS coords
:: 0.408818   -0.059284    0.374229
:: 0.305357    0.059284   -0.374229
:: Local minimum coords:
-0.4 0 0.1
0.4 0 -0.1
End

Lattice
 2.645886    0      0
     0   2.645886   0
End

AtomType H

DIRAC H
 1  0
 VALENCE
 1S 1
SubEnd

BasisFunctions
 1S   0.69

2P   1.25
SubEnd

FitFunctions
 1S   3.16
 1S   2.09
 1S   1.38
 2S   1.50
 2P   4.00
 2P   2.65
 2P   1.75
 3D   4.00
 3D   2.50
SubEnd

End


END INPUT
eor
mv RUNKF H2SlabFreq.runkf

rm Points
```

Here comes the second run where the Hessian (from the frequency run) is used to find the transtition state

```
$ADFBIN/band << eor
Title H2 slab TS search
Comment
    Optimized for speed / Accuracy low
    Basis very small
    Demonstrates the GeoOpt block
End

KSpace 2

Accuracy 5.0

screening
RMadel 9
end

Dependency Basis=1e-8

RunType
 TS
End

GeoOpt
Iterations 50
Converge Grad=1e-3
RestartSCF true
InitialHessian H2SlabFreq.runkf
End

SCF
 Mixing 0.2
End

Convergence
 Degenerate default
End

DIIS
 NCycleDamp 0
 DiMix 0.5
End

Units
 Length Angstrom
 Angle Radian
End

ATOMS H
-0.4 0 0.1
0.4 0 -0.1
End

Lattice
 2.645886    0      0
     0   2.645886   0
```

```
End

AtomType H

DIRAC H
 1  0
 VALENCE
 1S 1
SubEnd

BasisFunctions
 1S   0.69
2P   1.25
SubEnd

FitFunctions
 1S   3.16
 1S   2.09
 1S   1.38
 2S   1.50
 2P   4.00
 2P   2.65
 2P   1.75
 3D   4.00
 3D   2.50
SubEnd

End


END INPUT
eor

mv RUNKF H2SlabTS.runkf

rm Points
```

Time dependent DFT

Time-dependent DFT calculations for bulk silicon

Sample directory: band/Silicon/

The time-dependent DFT functionality is an important functionality. It enables you to calculate frequency-dependent dielectric functions for 1-dimensional and 3-dimensional periodic systems. In the present example, a standard geometry for bulk Silicon is given. The accuracy and kspace variables can keep their normal values. The important part in this example is of course the RESPONSE block. It specifies that 7 frequencies should be treated, with an even spacing between 0.0 hartree and 0.25 hartree. In this example scalar ZORA relativistic effects are switched on with the isz line in the RESPONSE block.

```
$ADFBIN/band << eor
TITLE Silicon

ACCURACY 5
KSPACE 2

DEPENDENCY BASIS 1e-10

UNITS
 LENGTH ANGSTROM
END

RESPONSE
nfreq   7
strtfr  0d0
endfr   25d-2
isz     1
END


DEFINE
AAA=5.43
HA=AAA/2
END

LATTICE
0   HA  HA
HA  0   HA
HA  HA  0
END


ATOMS Si
0.0   0.0   0.0
HA/2  HA/2  HA/2
END
...
END INPUT
eor
```

For Silicon the real and imaginary parts of the dielectric function: epsilon(omega) = 1 + 4 Pi Chi(omega) are calculated.

In the output file, the results will look something like the fragment below. The output specifies for which frequency the dielectric function is determined, and then proceeds to print the values for the 3x3 tensors.

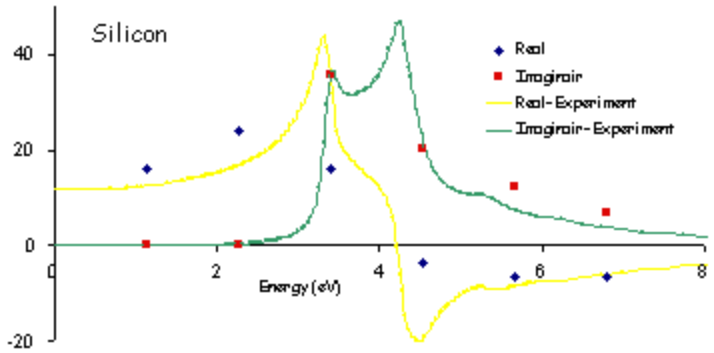
The real and imaginary parts are printed separately. At this frequency, the imaginary part is still zero. Because of the high symmetry of the system, the real part is a constant times the unit matrix except for numerical noise.

```
 ** Frequency **   0.833333E-01 au    2.26756     eV **
 ** Start the SCF procedure **
 ***** Real ******
 ** Chi_jj   X  **   -12.8363       0.142802E-18   0.547977E-17
 ** Chi_jj   Y  **   0.202883E-17   -12.8363       0.121052E-17
 ** Chi_jj   Z  **   0.124042E-16   0.215311E-17   -12.8363
 ***** Imag ******
 ** Chi_jj   X  **   0.000000E+00   0.000000E+00   0.000000E+00
 ** Chi_jj   Y  **   0.000000E+00   0.000000E+00   0.000000E+00
 ** Chi_jj   Z  **   0.000000E+00   0.000000E+00   0.000000E+00
 *****************
```

After each frequency has been treated, the results are summarized in a table, for each component separately. For the xx-component, this looks as in the table below. The frequency/energy is again printed in two different units, the Dielectric Function is printed in a.u. The values for Chi, which are trivially related to those printed here, are summarized in a separate table.

```
=================================================================
==          Frequency          ===       Dielectric Function     ==
==     a.u.    ==      e.V.    ===         Re    ==         Im     ==
============XX-dir================================================
    0.416667E-01     1.13378          16.1119         0.000000E+00
    0.833333E-01     2.26756          23.7904         0.000000E+00
    0.125000         3.40134          15.8529          35.8574
    0.166667         4.53512         -3.49949          20.2221
    0.208333         5.66890         -6.60897          12.3661
    0.250000         6.80268         -6.42943          6.87957
============YY-dir================================================
    0.416667E-01     1.13378          16.1119         0.000000E+00
    0.833333E-01     2.26756          23.7904         0.000000E+00
    0.125000         3.40134          15.8529          35.8574
    0.166667         4.53512         -3.49949          20.2221
    0.208333         5.66890         -6.60897          12.3661
    0.250000         6.80268         -6.42943          6.87957
============ZZ-dir================================================
    0.416667E-01     1.13378          16.1119         0.000000E+00
    0.833333E-01     2.26756          23.7904         0.000000E+00
    0.125000         3.40134          15.8529          35.8574
    0.166667         4.53512         -3.49949          20.2221
    0.208333         5.66890         -6.60897          12.3661
    0.250000         6.80268         -6.42943          6.87957
```

Results of the test calculation (red/blue) are plotted in next Figure together with experimental data (yellow/ green). The results for the seven specified frequencies are given. It should be obvious that more frequencies are needed (resulting in longer run times) to obtain a smooth curve in which peaks cannot be missed because of too coarse interpolation.



Time-dependent DFT calculations for bulk helium

Sample directory: band/He_crystal/

```
$ADFBIN/band << eor
TITLE DIAMOND

ACCURACY 5
KSPACE 2

Dependency  Basis=1.e-6

UNITS
 LENGTH ANGSTROM
END

RESPONSE
nfreq    7
strtfr   0d0
endfr    7d-1
END

DEFINE
AAA=3.57
HA=AAA/2
END

LATTICE
0   HA  HA
HA  0   HA
HA  HA  0
END


ATOMS C
0.0   0.0   0.0
HA/2  HA/2  HA/2
END

BasisDefaults
```

```
BasisType DZ
End

END INPUT
eor
```

Time-dependent DFT calculations for bulk diamond

Sample directory: band/Diamond/

```
$ADFBIN/band << eor
TITLE DIAMOND

ACCURACY 5
KSPACE 2

Dependency  Basis=1.e-6

UNITS
 LENGTH ANGSTROM
END

RESPONSE
nfreq    7
strtfr   0d0
endfr    7d-1
END

DEFINE
AAA=3.57
HA=AAA/2
END

LATTICE
0   HA  HA
HA  0   HA
HA  HA  0
END


ATOMS C
0.0   0.0   0.0
HA/2  HA/2  HA/2
END

BasisDefaults
BasisType DZ
End

END INPUT
eor
```

Time-dependent DFT calculations for hydrogen chain

Sample directory: band/H_chain

The input for a one-dimensional system is no different from that for a three-dimensional system. No tests have been performed on two-dimensional systems yet in combination with the TDDFT code. It is therefore not expected to work. Besides the number of frequencies, and the beginning and end frequency of the frequency range, the RESPONSE block also contains stricter than default convergence criteria (cnvi and cnvj) for the fit coefficients describing the density change.

```
$ADFBIN/band << eor
Title H2-chain

ACCURACY 5
KSPACE 3

DEPENDENCY BASIS 1e-10

RESPONSE
nfreq  10
strtfr 0d0
endfr  15d-3
cnvi   1d-5
cnvj   1d-5
END

DEFINE
HX = 4.5
END

LATTICE
HX
END

ATOMS H
 1.0  0.001 0.0
-1.0 -0.001 0.0
END

...

END INPUT
eor
```

The output for this calculation will give rise to a table like the following one:

```
 ================================================================
 ==         Frequency         ===       Polarizability(xx)      ==
 ==     a.u.   ==      e.V.   ===        Re    ==        Im     ==
 ================================================================
    0.166667E-02   0.453512E-01    127.119        0.00000
    0.333333E-02   0.907023E-01    127.201        0.00000
    0.500000E-02   0.136054        127.338        0.00000
    0.666667E-02   0.181405        127.530        0.00000
```

```
    0.833333E-02   0.226756        127.778        0.00000
    0.100000E-01   0.272107        128.083        0.00000
    0.116667E-01   0.317458        128.446        0.00000
    0.133333E-01   0.362809        128.868        0.00000
    0.150000E-01   0.408161        129.351        0.00000
 ================================================================
 ==          Frequency         ===            Chi_JJ (xx)        ==
 ==     a.u.   ==      e.V.   ===        Re    ==        Im     ==
 ================================================================
     0.00000        0.00000       -2.74118        0.00000
    0.166667E-02   0.453512E-01   -2.74153        0.00000
    0.333333E-02   0.907023E-01   -2.74259        0.00000
    0.500000E-02   0.136054       -2.74436        0.00000
    0.666667E-02   0.181405       -2.74685        0.00000
    0.833333E-02   0.226756       -2.75005        0.00000
    0.100000E-01   0.272107       -2.75399        0.00000
    0.116667E-01   0.317458       -2.75866        0.00000
    0.133333E-01   0.362809       -2.76409        0.00000
    0.150000E-01   0.408161       -2.77028        0.00000
 ================================================================
```

Time-dependent current DFT calculations for a metal

Sample directory: band/Cu_resp_NR_ALDA/

This example shows how to calculate the dielectric function for a metal. The RESPONSE keyblock is as usual. In the manual it is explained that the KSPACE parameter should be chosen with care.

```
$ADFBIN/band << eor
Title: response of Cu within ALDA

Comment
 Technical
   Hybrid k space integration (3D)
   Reasonable real space integration accuracy
   Definitions of variables
 Features
   Lattice   : 3D
   Unit cell : 1 atom
   Basis     : NO+STO w/ core (DZ/Cu.2p)
   Options   : ALDA
End

kspace  7
Accuracy 4.0

RESPONSE
 nfreq   2
 strtfr  0.10d0
 endfr   0.25d0
END

Units
```

```
 Length Angstrom
End

Define
 halflatt=3.61/2
End

Lattice :: FCC
  0        halflatt halflatt
  halflatt 0        halflatt
  halflatt halflatt 0
End

Atoms Cu
  0.0
End

BasisDefaults
BasisType DZ
Core Large
End

END INPUT
eor
```

Spectroscopy

hyperfine A-tensor

Sample directory: band/TiF3a/

Sample calculation for an ESR A-tensor calculation. The calculation should be spin unrestricted and have the ATENSOR keyword.

```
$ADFBIN/band << eor
Title tif3

comment
end

Kspace 1
Accuracy 5.0


Units
  Length Angstrom
End

Define
   a = 25
   RTIF = 1.78
   RY  = RTIF*SQRT(3)/2
End
```

Unrestricted

ATENSOR
END

```
Lattice
  a   0.0 0.0
  0.0 a   0.0
  0.0 0.0 a
End

Atoms Ti
  0.0 0.0 0.0
end

Atoms F
     RTIF   0   0
    -RTIF/2  RY 0
    -RTIF/2 -RY 0
end

AtomType Ti
DIRAC Ti
 7  0
```

```
VALENCE
1S
2S
2P
3S
3P
4S
3D  2
SubEnd
BASISFUNCTIONS
  1S    26.300000
  2S     5.600000
  2P    12.000000
  3S     4.650000
  3P     5.400000
  3P     2.300000
  3D     4.950000
  3D     1.040000
  4S     1.900000
  4S     0.800000
  4P     1.200000
  4F     2.300000
SubEnd
FITFUNCTIONS
  1S    52.600000
  2S    58.320000
  2S    39.240000
  3S    38.750000
  3S    27.730000
  4S    26.240000
  4S    19.530000
  4S    14.540000
  5S    13.490000
  5S    10.330000
  5S     7.910000
  6S     7.260000
  6S     5.680000
  6S     4.450000
  6S     3.480000
  7S     3.170000
  7S     2.530000
  7S     2.010000
  7S     1.600000
  2P    38.300000
  3P    25.820000
  4P    17.610000
  5P    12.200000
  5P     7.070000
  6P     4.970000
  6P     3.010000
  7P     2.150000
  3D    28.600000
  4D    19.530000
  5D    13.550000
  5D     7.860000
  6D     5.540000
  6D     3.360000
```

```
  7D     2.400000
  4F    13.900000
  5F     7.920000
  5F     3.800000
  6F     2.240000
  5G     8.400000
  5G     4.390000
  5G     2.290000
SubEnd
End

AtomType F
DIRAC F
 3  0
VALENCE
1S
2S
2P  5
SubEnd
BASISFUNCTIONS
  1S    10.880000
  2S     3.240000
  2S     0.740000
  2P     4.540000
  2P     1.240000
  3D     2.000000
  4F     3.000000
SubEnd
FITFUNCTIONS
  1S    21.760000
  2S    24.390000
  2S    16.560000
  2S    11.240000
  2S     7.630000
  3S     7.600000
  3S     5.480000
  3S     3.950000
  3S     2.850000
  3S     2.050000
  3S     1.480000
  2P    14.400000
  2P     7.530000
  3P     5.900000
  3P     3.420000
  3P     1.980000
  3D     9.700000
  3D     6.160000
  3D     3.910000
  3D     2.480000
  4F     6.500000
  4F     3.750000
  5G     4.500000
SubEnd
End

END INPUT
eor
```

Zeeman g-tensor

Sample directory: band/TiF3g/

Sample calculation for an ESR g-tensor calculation. This makes only sense for a relativistic spin orbit calculation.

```
$ADFBIN/band << eor
Title tif3

comment
  1d structure
end

Kspace 1
Accuracy 5.0

Relativistic ZORA SPIN

esr
end

Units
  Length Angstrom
End

Define
   a = 25
   RTIF = 1.78
   RY  = RTIF*SQRT(3)/2
End

Lattice
  a
End

Atoms Ti
  0.0 0.0 0.0
end

Atoms F
     0   RTIF   0
    -RY -RTIF/2 0
     RY -RTIF/2 0
end

AtomType Ti
DIRAC Ti
 7  0
VALENCE
1S
2S
2P
3S
```

```
3P
4S
3D  2
SubEnd
BASISFUNCTIONS
  1S    26.300000
  2S     5.600000
  2P    12.000000
  3S     4.650000
  3P     2.450000
  3D     4.950000
  3D     1.040000
  4S     1.650000
  4P     1.200000
SubEnd
FITFUNCTIONS
  1S    52.600000
  2S    58.320000
  2S    39.240000
  3S    38.750000
  3S    27.730000
  4S    26.240000
  4S    19.530000
  4S    14.540000
  5S    13.490000
  5S    10.330000
  5S     7.910000
  6S     7.260000
  6S     5.680000
  6S     4.450000
  6S     3.480000
  7S     3.170000
  7S     2.530000
  7S     2.010000
  7S     1.600000
  2P    38.300000
  3P    25.820000
  4P    17.610000
  5P    12.200000
  5P     7.070000
  6P     4.970000
  6P     3.010000
  7P     2.150000
  3D    28.600000
  4D    19.530000
  5D    13.550000
  5D     7.860000
  6D     5.540000
  6D     3.360000
  7D     2.400000
  4F    13.900000
  5F     7.920000
  5F     3.800000
  6F     2.240000
  5G     8.400000
  5G     4.390000
  5G     2.290000
```

```
SubEnd
End

AtomType F
DIRAC F
 3  0
VALENCE
1S
2S
2P  5
SubEnd
BASISFUNCTIONS
  1S    10.880000
  2S     3.220000
  2P     3.520000
SubEnd
FITFUNCTIONS
  1S    21.760000
  2S    24.390000
  2S    16.560000
  2S    11.240000
  2S     7.630000
  3S     7.600000
  3S     5.480000
  3S     3.950000
  3S     2.850000
  3S     2.050000
  3S     1.480000
  2P    14.400000
  2P     7.530000
  3P     5.900000
  3P     3.420000
  3P     1.980000
  3D     9.700000
  3D     6.160000
  3D     3.910000
  3D     2.480000
  4F     6.500000
  4F     3.750000
  5G     4.500000
SubEnd
End

END INPUT
eor
```

NMR

Sample directory: band/PE-NMR/

With the NMR keyblock you can specify for which atom you want the shielding tensor.

The NMR option is not implemented with the frozen core approximation, hence we set core to NONE.

```
$ADFBIN/band << eor
TITLE PE

NMR
 nmratom=1
 ms0=1.
end

Units
  Length Angstrom
  Angle  Degree
end

xc
 GGA Always Becke Perdew
end

DEPENDENCY BASIS 1e-10

KSPACE 3
Accuracy 5

define
  aCCC = 112.777
  aHCH = 109.47
  alf2 = aCCC/2
  bet  = aHCH/2.
  rcc=1.533
  rch=1.09
  z3 = rch*cos(bet)
  y3 = rch*sin(bet)
  T = 2*rcc*sin(alf2)
  C2x =  rcc*sin(alf2)
  C2z = -rcc*cos(alf2)
end

Lattice
   T          0.        0.
end

Atoms C
   0.         0.        0.
   C2x        0.        C2z
end

Atoms H
   0.          y3       z3
   0.         -y3       z3
   C2x         y3       C2z-z3
   C2x        -y3       C2z-z3
end

BasisDefaults
 BasisType TZ2P
 Core NONE
End
```

```
end input
eor
```

EFG

Sample directory: band/SnO_EFG/

The calculation of the electric field gradient is invoked by the EFG keyblock.

Because Sn is quite a heavy atom we use the scalar relativistic option.

```
$ADFBIN/band << eor
Title SnO EFG

kspace 3
accuracy 5

relativistic ZORA

programmer
FlioSinglePrecision false
end

EFG
End

Units
Length Angstrom
End

coordinates natural

dependency basis=1e-8

diis
ncycledamp 0
dimix 0.2
end

define
aaa=3.8029
ccc=4.8382
vvv=0.2369
end


lattice
aaa 0 0
0 aaa 0
0 0 ccc
end

atoms O
```

```
0 0 0
0.5 0.5 0
end

atoms Sn
0 0.5 vvv
0.5 0 -vvv
end

BasisDefaults
BasisType DZ
Core none
End


end input
eor
```

Analysis

CO absorption on a Cu slab: fragment option and densityplot

Sample directory: band/Frags_COCu/

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 absorbed 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.runkf'.

```
# ---------------------------- CO molecule -------------------------

$ADFBIN/band << eor
Title The CO fragment

Comment
 Technical
   Zero order k space integration
   Good real space integration accuracy
   Definitions of variables
 Features
   Lattice   : 2D, large lattice vectors
   Unit cell : 2 atoms, 1x1, quasi molecular
   Basis     : NO+STO w/ core
End


PRINT EIGENS

Kspace   1  ! neglect dispersion
Accuracy   4

Define
  bond=2.18
  far=25
End

Lattice     ! CO molecules far apart
  far  0.0
  0.0  far
End

Atoms   C
  0 0 0
End

Atoms   O
  0 0 bond
```

```
End

BasisDefaults
BasisType DZ
Core Large
End

End Input
eor

mv RUNKF CO.runkf
```

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 -------------------------


$ADFBIN/band << eor
Title Cu slab with CO adsorbed

Comment
 Technical
   Quadratic K space integration (low)
   Good real space integration accuracy
   Definitions of variables
 Features
   Lattice   : 2D
   Unit cell : 3 atoms, 1x1
   Basis     : NO+STO w/ core
   Options   : Molecular fragment
               Analysis: DOS, PDOS, COOP
End


KSpace 3
Accuracy 4

! fragment specification

Fragment CO.runkf
  1 1
  2 2
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
  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

Define
  dist=3.44
  bond=2.18
End

Lattice
  4.822 0.0
  0.0   4.822
End

Atoms   C
  0 0 dist
End

Atoms   O
  0 0 dist+bond
End

Atoms   CU
  0.0   0.0   0.0
End
```

```
BasisDefaults
BasisType DZ
Core Large
End

End Input
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 Contents of DOS file
cat pdos.CO_Cu

mv RUNKF COCu.runkf
```

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 -------------------------

$ADFBIN/band << eor
Title Cu slab

Comment
 Technical
   Quadratic K space integration (low)
   Good real space integration accuracy
   Definitions of variables
 Features
   Lattice   : 2D
   Unit cell : 3 atoms, 1x1
   Basis     : NO+STO w/ core
   Options   :
End


Kspace 3
Accuracy 4

Define
  dist=3.44
  bond=2.18
End

Lattice
  4.822 0.0
  0.0   4.822
End

Atoms   CU
  0.0   0.0   0.0
End

BasisDefaults
BasisType DZ
Core Large
End
```

```
End Input
eor

mv RUNKF Cu.runkf
```

Now we are all set to do our final calculation. We have the two fragment files CO.runkf and Cu.runkf, and the restart file COCu.runkf. 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
--------------------------


$ADFBIN/band -n 1 << eor
Title Cu slab with CO adsorbed (restart density plot)

Kspace 3
Accuracy 4

Restart COCu.runkf &
DensityPlot
End

Grid
Type Coarse
End

DensityPlot
scf
End

! fragment specification

Fragment CO.runkf
  1 1
  2 2
End

Fragment Cu.runkf
  1 3
End

Define
  dist=3.44
  bond=2.18
End

Lattice
  4.822 0.0
  0.0   4.822
End

Atoms   C
  0 0 dist
End
```

```
Atoms   O
  0 0 dist+bond
End

Atoms   CU
  0.0   0.0   0.0
End

BasisDefaults
BasisType DZ
Core Large
End

End Input
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.

Cu_slab: 2-dim. Finite temperature and orbital plot

Sample directory: band/Cu_slab/

A two-dimensional infinite (periodic boundary conditions) slab calculation is performed for Cu. The dimensionality is simply defined by the number of records in the Lattice data block. In a 2-dimensional calculation the lattice vectors are put in the xy-plane. (In a one-dimensional calculation (polymer), the lattice vector is taken along the x-axis in the program.)

Slab calculations for metals frequently suffer from SCF convergence problems, as a result of the open valence band(s). To help the program converge it is often useful or even necessary to use some special features, such as the ElectronicTemperature key. This particular key requires a numerical value (0.025 in the example) and implies that a finite-temperature electronic distribution is used, rather than a sharp cut-off at the Fermi level. The numerical value is the applied energy width, in hartree units.

The so-modified electronic distribution also affects the energy. The 'true' zero-T energy is computed, approximately, by an interpolation formula. The interpolation is not very accurate and one should try to use as small as possible values for the ElectronicTemperature key so as to avoid increasing uncertainty in the results. The program prints, in the energy section of the output file, the finite-T correction term that has been applied through the interpolation formula. This gives at least an indication of any remaining uncorrected deviation of the outcome from a true zero-T calculation.

In the second run the runkf file of the first run is used to do an orbital plot restart. Normally you would rename the resulting TAPE41 to "myslab.t41" and watch the orbitals with adfview.

```
# ----------------------------- first run --------------------------

$ADFBIN/band << eor
Title Cu slab

Comment
 Technical
   Quadratic K space integration
   Good real space integration accuracy
 Features
   Lattice   : 2D
   Unit cell : 1 atom, 1x1
   Basis     : NO+STO w/ core
   Options   : ElectronicTemperature (temperature effect)
End

Kspace 5
Accuracy 4

Convergence
 ElectronicTemperature  0.025
End

Lattice
  4.822 0.0
  0.0   4.822
End

Atoms Cu
  0.0   0.0   0.0
End

BasisDefaults
  BasisType DZ
End

EndInput
eor

mv RUNKF CuSlab.runkf
rm Points

# ----------------------------- orbital plot -------------------------
export NSCM=1

$ADFBIN/band -n 1 << eor
Title Cu slab orbital plot

Comment
 Technical
   Quadratic K space integration
   Good real space integration accuracy
 Features
   Lattice   : 2D
```

```
   Unit cell : 1 atom, 1x1
   Basis     : NO+STO w/ core
   Options   : ElectronicTemperature (temperature effect)
End

Kspace 5
Accuracy 4

Restart CuSlab.runkf&
OrbitalPlot
End

Grid
Type Coarse
End

OrbitalPlot
1 Band 2 4 ! k-point 1, bands 2 to 4
3  -0.1 0.1 ! k-point 3 orbitals within 0.1 Hartree from Fermi Level
End

Convergence
 ElectronicTemperature  0.025
End

Lattice
  4.822 0.0
  0.0   4.822
End

Atoms Cu
  0.0   0.0   0.0
End

BasisDefaults
  BasisType DZ
End

EndInput
eor

echo "Begin TOC of tape41"

$ADFBIN/dmpkf -n 1 TAPE41 --toc

echo "End TOC of tape41"
```

NaCl: ionic fragments

Sample directory: band/NaCl_ionic/

A bulk crystal computation for Sodium Chloride (common salt), starting with charged atoms.

```
$ADFBIN/band << eor
Title NaCl (from charged atoms)

Comment
 Technical
   Hybrid K space integration (3D)
   Low real space integration accuracy
   Natural coordinates
   Lengths in Angstrom
   Parameters Dirac procedure
 Features
   Lattice   : 3D
   Unit cell : 2 atoms
   Basis     : NO w/ core
   Options   : Ionic atoms in startup
 Note:
   Seems very unstable, highly sensitive to DIIS parameters
   (and simple damping) and well parameters.
End

Accuracy 3.5
Kspace 3

Diris
 NCycleDamp 15
End

Units
 Length Angstrom
End

Lattice
0    2.75   2.75
2.75  0     2.75
2.75  2.75   0
End

ATOMS Na
0
End

Coordinates Natural
Atoms Cl
  .5  .5  .5
End

AtomType Na
Dirac   Na
  4 1
  RADIAL  2000
  RMin    1E-8
  RMax    200.0
  VALENCE
  1 0
  2 0
  2 1
  3 0 0
```

```
SubEnd

BasisFunctions
3S   1.75
3P   1.75
SubEnd

FitFunctions
  1 0 18.9
  2 0 30.3
  2 0 15.5
  3 0 14.9
  3 0  8.9
  4 0  7.8
  4 0  5.1
  4 0  3.3
  5 0  2.8
  5 0  1.9
  5 0  1.3
  2 1 14.3
  3 1  9.9
  4 1  6.7
  4 1  3.4
  5 1  2.4
  5 1  1.3
  3 2 10.5
  4 2  5.4
  5 2  3.0
  5 2  1.3
  4 3  5.8
  5 3  1.7
  5 4  2.0
SubEnd
End

AtomType Cl
Dirac  Cl
  5 3
  RADIAL  2000
  RMin    1E-8
  RMax    450.0
  RWell   150.0
  VWell   -2.0
  VALENCE
  1 0
  2 0
  2 1
  3 0
  3 1
SubEnd

FitFunctions
  1 0 29.1
  2 0 49.5
  2 0 26.1
  3 0 25.8
  3 0 15.8
```

```
  4 0 14.2
  4 0  9.4
  4 0  6.2
  5 0  5.4
  5 0  3.8
  5 0  2.6

  2 1 21.2
  3 1 16.5
  4 1 12.4
  4 1  6.8
  5 1  5.1
  5 1  3.1

  3 2 16.6
  4 2  9.4
  5 2  5.5
  5 2  2.6

  4 3  8.7
  5 3  3.3

  5 4  4.0
SubEnd
End

End Input
eor
```

Bader analysis

Sample directory: band/Li2O_Bader/

To get the Atoms In Molecules (or Bader) analysis use the GridBasedAIM block key.

The grid-based AIM method is very fast, but a bit inaccurate. Therefore we force extra integration points.

```
Title Li2O bulk (fluorite structure)

KSpace 3

accuracy 4

Integration
accsph 6
accpyr 6
end

GridBasedAIM
End


Dependency basis=1e-9 fit=1e-8
```

```
tails bas=0 core=0 fit=0

DIIS
dimix 0.2
ncycledamp 0
end

scf
mixing 0.4
end

xc
gga scf bp86
end

define
ha=8.73/2
qa=8.73/4
end

Lattice
0 ha ha
ha 0 ha
ha ha 0
end

atoms O
0.0
end

atoms Li
qa qa qa
3*qa qa qa
end

BasisDefaults
BasisType TZ2P
Core small
end

end input
eor
```

List of examples