Scientific Computing & Modelling

Examples

ADF Program System
Release 2009.01

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

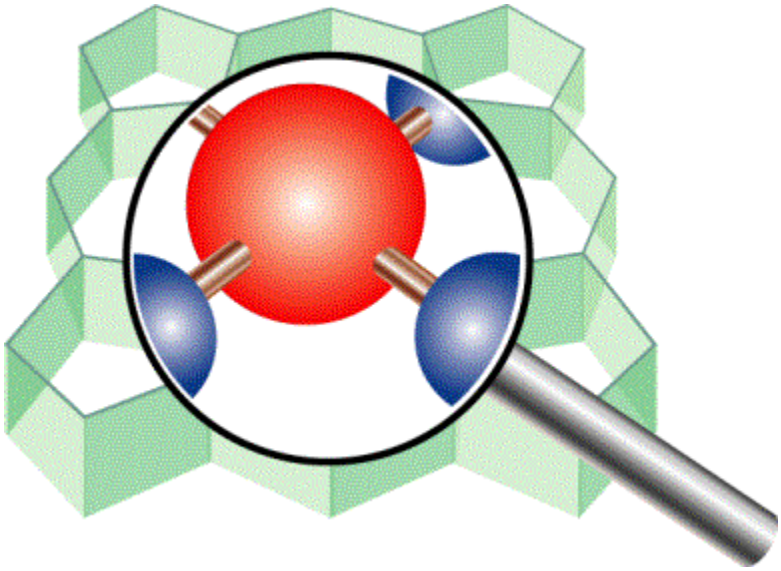


Table of Contents

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. In most directories, there are also files for ADFinput present.

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 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/adf/HCN/ contains an ADF calculation on the HCN molecule. Each sample subdirectory contains:

- A file TestName.run: the UNIX script to execute the calculation 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.
- Zero or more files with a .adf extension. These files, if present, are intended for ADFinput and demonstrate the same functionality as the two files above. However, there are also differences between the .adf and the TestName.run files so the results obtained with the .adf files **cannot be compared directly** with TestName_orig.out. Also, the TestName.run file usually contains more than one calculation, for which more than one .adf file is required. That's why in some directories you may find more than one .adf file.
  In some directories, there are no .adf files, which usually means the functionality demonstrated by the example is not supported by the GUI.

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, Analysis, 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.
- An ADF 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.

Survey of the Examples

The Survey of Applications follows a survey of the main application topics with references to related sample runs is given. A sample run usually involves several calculations, for instance a few CREATE runs (with ADF), then a molecular calculation (also ADF), and finally a NMR calculation (with the NMR program) to compute chemical shifts. The samples are identified in this documentation by the name of the directory they reside in. The samples are indicated by these directory names. For instance, GO_H2O refers to the directory GO_H2O/ (in $ADFHOME/adf/), where in this case GO stands for Geometry Optimization.

Model Hamiltonians

Special exchange-correlation functionals

CO: asymptotically correct XC potentials

Sample directory adf/CO_model

For property calculations, xc potentials with asymptotically correct (-1/r) behavior outside the molecule, the results tend to be superior to regular LDA or GGA calculations. This is especially true for small molecules and for properties that depend heavily on the proper description of the outer region of the molecule. In the example, all-electron basis sets are used. This is mandatory for the SAOP potential.

```
$ADFBIN/adf -n1 <<EOR
create C  $ADFRESOURCES/TZ2P/C
end input
EOR
mv TAPE21 t21.C

$ADFBIN/adf -n1 <<EOR
create O  $ADFRESOURCES/TZ2P/O
end input
EOR
mv TAPE21 t21.O
```

In the next example, excitation energies are calculated with the GRACLB potential. This potential requires one number as argument: the experimental ionization potential in atomic units. This number can be either based on an experimental value, or on previous GGA total energy calculations.

```
$ADFBIN/adf <<EOR
title CO excitations grac potential

INTEGRATION 6.0

XC
 Model GRACLB 0.515
End

Atoms
O  0             0   0
C  1.128205364   0   0
end

Excitation
 Lowest 10
 Onlysing
End

Fragments
 O t21.O
 C t21.C
```

```
End


end input
EOR

rm TAPE21 logfile
```

The same calculation with the SAOP xc potential would differ in the XC block only:

```
XC
 Model SAOP
End
```

SAOP depends on the orbitals which makes it more expensive to evaluate than GRAC for large molecules.

OH: Meta-GGA energy functionals

Sample directory adf/OH_MetaGGA

First two calculations on OH are performed which use, respectiveley, the hybrid meta-GGA TPSSh and the meta-GGA TPPS during the SCF. They require, respectively, the following XC input:

```
XC
 MetaHybrid TPSSh
END
XC
 MetaGGA TPSS
END
```

Next large even-tempered basis sets are used in the calculation of the atomization energy of OH using various modern GGA, meta-GGA and hybrid post-SCF energy expressions.

In the Create runs, a large even-tempered basis set is selected for O and H, which should give results closer to the basis set limit than the regular ADF basis sets. For both atoms, a second atomic calculation follows the Create run, in order to enable a comparison to the true atoms, rather than the artificial spherically symmetric atom from the Create run. This is achieved by specifying the keywords

```
unrestricted
charge 0 2
symmetry C(lin)
occupations
sigma  3 // 3
pi     2 // 0
end
```

in the case of oxygen. This fixes the proper occupations. The result files of both the Create runs and the atomic correction runs are stored.

In the molecular calculation, the symmetry of the molecule is explicitly broken and the occupations are specified in order to avoid the fractional occupations that ADF would otherwise choose. Although it is not

said that such a solution would be inferior, the integer occupation solution is the one which allows direct comparison to literature results obtained with other programs.

One of the new GGA potentials has been specified for the xc potential and the keyword METAGGA implies that a series of GGA and meta-GGA xc energies is to be calculated and compared to those energies from the atomic calculations. Specifying HARTREEFOCK also enables calculation of PostSCF energies using hybrid functionals.

```
METAGGA
symmetry C(lin)
xc
GGA PBE
end
HARTREEFOCK
```

A fairly high numerical integration has been specified. For meta-GGA calculations we do recommend this, at least 6 for the time being, as the numerical stability of the results tends to be somewhat lower than for regular GGA calculations.

The block key ENERGYFRAG

```
ENERGYFRAG
O  t21.unr.O
H  t21.unr.H
END
```

implies that the meta-GGA result must not only be compared to the spherically symmetric results from the Create runs, but also to the non-spherical atoms.

The molecular output file prints the PBE Total Bonding energy as usual (in various energy units).

Then a prints a list of 'Total Bonding Energies' for many different Exc functionals, including PBE. Because the numerical approach to obtain the two PBE results is somewhat different, small differences may occur between the two numbers. You now have an overview of the bonding energies of all (meta)GGA functionals currently implemented in ADF. This should give a good indication of the theoretical error bar or the uncertainty in the xc approximation.

```
  Total Bonding Energy:         -0.286127457276205         -7.7859          -179.55          -751.23

 TOTAL BONDING ENERGIES FROM VARIOUS XC FUNCTIONALS

 with respect to fragments in FRAGMENTS input block

                               hartree           eV             kcal/mol        kJ/mol

 Total Bonding Energy with respect to FRAGMENTS

XC Energy Functional
====================
FR: KCIS-modified  [1]  =    -0.2755742057   -7.4987587362 -172.9254430523 -723.5200549587
FR: KCIS-original  [2]  =    -0.2777894828   -7.5590395194 -174.3155506035 -729.3362649626
FR: PKZB           [3]  =    -0.2815570432   -7.6615600946 -176.6797306630 -739.2279943483
FR: VS98           [4]  =    -0.3017049511   -8.2098127875 -189.3227350810 -792.1263249228
FR: LDA(VWN)       [5]  =    -0.2887564297   -7.8574654492 -181.1974143810 -758.1299830563
FR: PW91           [6]  =    -0.2876922977   -7.8285089331 -180.5296614163 -755.3361046473
FR: BLYP           [7]  =    -0.2770745036   -7.5395839361 -173.8668943006 -727.4590869882
FR: BP             [8]  =    -0.2855241909   -7.7695117221 -179.1691537365 -749.6437405057
FR: PBE            [9]  =    -0.2858734106   -7.7790144775 -179.3882924288 -750.5606167957
.....
```

The same energy comparison is done with respect to the fragments (which most currently be atomic) in the ENERGYFRAG block. These are the numbers which should be comparable to experimental numbers.

Finally, the references for the various Exc functionals are printed in the output file.

```
XC Energy Functional
====================
EF: KCIS-modified  [1]  =    -0.1713622482   -4.6630059333 -107.5314455812 -449.9115690750
EF: KCIS-original  [2]  =    -0.1701706820   -4.6305817515 -106.7837263654 -446.7831118709
EF: PKZB           [3]  =    -0.1716508948   -4.6708604097 -107.7125740668 -450.6694106602
EF: VS98           [4]  =    -0.1712676117   -4.6604307410 -107.4720602503 -449.6631008503
EF: LDA(VWN)       [5]  =    -0.1980694328   -5.3897456994 -124.2904587006 -520.0312800855
EF: PW91           [6]  =    -0.1759694023   -4.7883730257 -110.4224787188 -462.0076517434
EF: BLYP           [7]  =    -0.1748768123   -4.7586421272 -109.7368680765 -459.1390568111
EF: BP             [8]  =    -0.1785853781   -4.8595573769 -112.0640284617 -468.8758958794
EF: PBE            [9]  =    -0.1751227104   -4.7653333576 -109.8911714787 -459.7846622469
....
```

Similar calculations can be done to obtain energy differences between different molecules. In that case the ENERGYFRAG keyword is not operational though. No detailed breakdown of the bonding energy is currently available for these new energy functionals. Experience shows that the energy values depend only mildly on the chosen xc functional for the xc potential.

H: SIC-VWN potential

Sample directories: adf/H_SICVWN/

Computation of the hydrogen atom with the SIC-VWN potential, should give the exact result (E=-0.5 a.u.).

Note: adf with the SIC-VWN only runs correctly serial, and symmetry NOSYM is required.

```
$ADFBIN/adf -n1 << eor
TITLE H atom, SIC-VWN (should be exact)
SYMMETRY NOSYM
UNRESTRICTED
CHARGE 0 1
ATOMS
  1  H  0.0000 0.0000  0.0000
END
INTEGRATION 6.0 6.0
FRAGMENTS
  H   t21.H
END
XC
  LDA VWN
END
SICOEP
  IPRINT 1
  SELF 35
END
DEPENDENCY fit=1e-10 bas=1e-8
SINGULARFIT FRUGAL
END INPUT
eor
```

HI: Hartree-Fock

Sample directory: adf/HI_EFG/

Example shows a Hartree-Fock calculation with a non-relativistic, scalar relativistic ZORA, and a spin-orbit coupled ZORA Hamiltonian. In this case ADF also calculates the electric field gradient (EFG) at the H and I nuclei (keyword QTENS).

First the non-relativistic calculation. Note that in this case the all-electron basis sets are obtained from the $ADFRESOURCES/ZORA directory.

```
$ADFBIN/adf << eor
Atoms
 H 0 0 0
 I 0 0 1.609
End
qtens
xc
 hartreefock
end
integration 5
Basis
 Type ZORA/TZ2P
 Core None
End
End input
eor
```

Next the scalar relativistic ZORA calculation. Note that in this case the all-electron basis sets are also obtained from the $ADFRESOURCES/ZORA directory, but this is default place where the key BASIS will search for basis sets in case of ZORA. ADF will also calculate the EFG including the small component density, also called SR ZORA-4.

```
$ADFBIN/adf << eor
Atoms
 H 0 0 0
 I 0 0 1.609
End
qtens
xc
 hartreefock
end
Relativistic  Scalar ZORA
integration 5
Basis
 Type TZ2P
 Core None
End
End input
eor
```

Next the spin-orbit coupled relativistic ZORA calculation. Note that in this case the all-electron basis sets are also obtained from the $ADFRESOURCES/ZORA directory, but again this is default place where the key BASIS will search for basis sets in case of ZORA. If one calculates this molecule with symmetry nosym, ADF will also calculate the EFG including the small component density, also called ZORA-4.

```
$ADFBIN/adf << eor
Atoms
```

```
 H 0 0 0
 I 0 0 1.609
End
qtens
xc
 hartreefock
end
Relativistic  Spinorbit ZORA
symmetry nosym
integration 5
Basis
 Type TZ2P
 Core None
End
End input
eor
```

H_2PO: B3LYP

Sample directory: adf/H2PO_B3LYP/

Example shows an unrestricted B3LYP calculation. In this case ADF also calculates the hyperfine interactions at H, P, and O nuclei (keyword ESR).

The 'DEPENDENCY' key is set to 1e-4. Note that for hybrids and Hartree-Fock the dependency key is always set. The default value in that case is 4e-3. By explicitely setting the 'DEPENDENCY' key we can use a lower value, which is possible in this case. One should check that the results remain reliable if one uses a smaller value for the 'DEPENDENCY' key.

```
$ADFBIN/adf << eor
Title hfs H2PO B3LYP TZ2P
Atoms
    O   1.492  0.000  0.000
    P   0.000  0.000  0.000
    H  -0.600 -0.650  1.100
    H  -0.600 -0.650 -1.100
End
xc
 hybrid B3LYP
end
Basis
 Type TZ2P
 Core None
End
dependency bas=1e-4
integration 5
esr
end
unrestricted
charge 0 1
```

```
end input
eor
```

For the hyperfine interactions it is important to use all-electron basis sets on the interesting nuclei. One can get more accurate results if one uses a larger basis set, like the QZ4P basis set, which is present in the $ADFRESOURCES/ZORA directory. The Basis key should then be:

```
Basis
 Type ZORA/QZ4P
 Core None
End
```

The QZ4P results for the isotropic value of the A-tensor are approximately: -24.77 MHz for ^{17}O, 962.02 MHz for ^{31}P, and 110.72 MHz for ^{1}H.

You may want to compare the results with previous B3LYP results by N. R. Brinkmann and I. Carmichael, J. Phys. Chem. A (2004), **108**, 9390-9399, which give for the Isotropic Fermi Contact Couplings (MHz) for the ^{2}A' State of H_2PO using B3LYP, with an aug-cc-pCVQZ basis set: -24.24 MHz for ^{17}O, 963.33 MHz for ^{31}P, and 111.51 MHz for ^{1}H.

MM Dispersion: Molecular Mechanics dispersion-corrected functionals

Sample directory: adf/MM_Dispersion/

Summary:

- MM dispersion (old implementation)
- Dispersion-corrected GGA-D functionals

MM dispersion (old implementation)

First example shows a geometry optimization of a van der Waals complex of two benzene molecules, connected to each other with a hydrogen molecule. With the MMDISPERSION keyword an extra empirical force (of similar form as in molecular mechanics) is added to the interaction between the three fragments, where one benzen molecule is fragment 1 (FD=1), the other benzene molecule is fragment 2 (FD=2), and the hydrogen molecule is fragment 3 (FD=3).

The atomic parameters are read from the file $ADFRESOURCES/MMDispersion/disp-param. The PBE functional and the TZP basis set are used, which is necessary if one wants to use the TZ parameters for the damping function, which are optimized for this combination of functional and basis set.

```
$ADFBIN/adf << eor
basis
  type TZP
  core small
End
XC
  GGA PBE
End
geometry
```

```
  converge grad=0.001
  iterations 5
end
Integration  4.5
SCF
  Iterations  60
  Converge  1.0E-06  1.0E-6
End
mmdispersion
  damping sigm
  damp_param tz
  combi s-k
  file_name $ADFRESOURCES/MMDispersion/disp-param
  nodefault
end
noprint sfo
Atoms   cartesians
C.ctr  0.000000000000     3.050000000000     1.391500000000  FD=1
H.h    0.000000000000     3.050000000000     2.471500000000  FD=1
C.ctr  1.205074349366     3.050000000000     0.695750000000  FD=1
H.h    2.140381785453     3.050000000000     1.235750000000  FD=1
C.ctr  1.205074349366     3.050000000000    -0.695750000000  FD=1
H.h    2.140381785453     3.050000000000    -1.235750000000  FD=1
C.ctr -0.000000000000     3.050000000000    -1.391500000000  FD=1
H.h   -0.000000000000     3.050000000000    -2.471500000000  FD=1
C.ctr -1.205074349366     3.050000000000    -0.695750000000  FD=1
H.h   -2.140381785453     3.050000000000    -1.235750000000  FD=1
C.ctr -1.205074349366     3.050000000000     0.695750000000  FD=1
H.h   -2.140381785453     3.050000000000     1.235750000000  FD=1
C.ctr -1.205074349366    -3.050000000000    -0.695750000000  FD=2
H.h   -2.140381785453    -3.050000000000    -1.235750000000  FD=2
C.ctr -0.000000000000    -3.050000000000    -1.391500000000  FD=2
H.h   -0.000000000000    -3.050000000000    -2.471500000000  FD=2
C.ctr  1.205074349366    -3.050000000000    -0.695750000000  FD=2
H.h    2.140381785453    -3.050000000000    -1.235750000000  FD=2
C.ctr  1.205074349366    -3.050000000000     0.695750000000  FD=2
H.h    2.140381785453    -3.050000000000     1.235750000000  FD=2
C.ctr -0.000000000000    -3.050000000000     1.391500000000  FD=2
H.h   -0.000000000000    -3.050000000000     2.471500000000  FD=2
C.ctr -1.205074349366    -3.050000000000     0.695750000000  FD=2
H.h   -2.140381785453    -3.050000000000     1.235750000000  FD=2
H.h    0.0                0.35              0.0             FD=3
H.h    0.0               -0.35              0.0             FD=3
End
End Input
```

The part of the bond energy that is due to the Grimme dispersion corrected functional is only inter-molecular (atom-atom contributions for which the fragment numbers FD are different).

Dispersion-corrected GGA-D functionals

In the second example a structure with 2 benzene molecules and a hydrogen molecule is optimized with the Grimme dispersion corrected PBE. Needed is the subkey DISPERSION in the key XC. If one starts with

atomic fragments the part of the bond energy that is due to the Grimme dispersion corrected functional is both inter-molecular as well as intra-molecular. In this case the subargument FD= in the ATOMS block key word is not used, which was only used in the old MM dispersion calculation.

```
$ADFBIN/adf << eor
Title Geometry optimization with Grimme dispersion correction for GGA
basis
  type TZP
  core small
End
XC
  GGA PBE
  DISPERSION
End
geometry
  converge grad=0.001
  Branch OLD
  iterations 50
end
Integration  4.5
Atoms   cartesians
C  0.000000000000     3.050000000000     1.391500000000
H  0.000000000000     3.050000000000     2.471500000000
C  1.205074349366     3.050000000000     0.695750000000
H  2.140381785453     3.050000000000     1.235750000000
C  1.205074349366     3.050000000000    -0.695750000000
H  2.140381785453     3.050000000000    -1.235750000000
C -0.000000000000     3.050000000000    -1.391500000000
H -0.000000000000     3.050000000000    -2.471500000000
C -1.205074349366     3.050000000000    -0.695750000000
H -2.140381785453     3.050000000000    -1.235750000000
C -1.205074349366     3.050000000000     0.695750000000
H -2.140381785453     3.050000000000     1.235750000000
C -1.205074349366    -3.050000000000    -0.695750000000
H -2.140381785453    -3.050000000000    -1.235750000000
C -0.000000000000    -3.050000000000    -1.391500000000
H -0.000000000000    -3.050000000000    -2.471500000000
C  1.205074349366    -3.050000000000    -0.695750000000
H  2.140381785453    -3.050000000000    -1.235750000000
C  1.205074349366    -3.050000000000     0.695750000000
H  2.140381785453    -3.050000000000     1.235750000000
C -0.000000000000    -3.050000000000     1.391500000000
H -0.000000000000    -3.050000000000     2.471500000000
C -1.205074349366    -3.050000000000     0.695750000000
H -2.140381785453    -3.050000000000     1.235750000000
H  0.0                0.35               0.0
H  0.0               -0.35               0.0
End
End Input
```

In the last example first three molecules (2 benzene molecules and a hydrogen molecule) are calculated with the Grimme dispersion corrected PBE. Needed again is the subkey DISPERSION in the key XC. The one for H_2 is given below:

```
$ADFBIN/adf << eor
Title Grimme dispersion-corrected GGA
basis
  type TZP
  core small
End
XC
  GGA PBE
  DISPERSION
End
SCF
  Iterations  60
  Converge  1.0E-06  1.0E-6
End
Atoms
H         0.000000    0.000000   -0.377906
H         0.000000    0.000000    0.377906
End
End Input
eor
mv TAPE21 h2.t21
```

Note that even for such a molecule there is a contribution from the so called Dispersion energy in the bonding energy (although it will be very small in this case).

Next a structure is calculated in which the three calculated molecules in it. If one starts with molecular fragments the part of the bond energy that is due to the Grimme dispersion corrected functional is only inter-molecular.

```
$ADFBIN/adf << eor
Title Grimme dispersion-corrected GGA
Fragments
  b1 benzene1.t21
  b2 benzene2.t21
  h2 h2.t21
End
XC
  GGA PBE
  DISPERSION
End
Atoms
C         0.000000    1.398973   -3.054539  f=b1
H         0.000000    2.490908   -3.049828  f=b1
C         1.211546    0.699486   -3.054539  f=b1
H         2.157190    1.245454   -3.049828  f=b1
C         1.211546   -0.699486   -3.054539  f=b1
H         2.157190   -1.245454   -3.049828  f=b1
C         0.000000   -1.398973   -3.054539  f=b1
H         0.000000   -2.490908   -3.049828  f=b1
C        -1.211546   -0.699486   -3.054539  f=b1
H        -2.157190   -1.245454   -3.049828  f=b1
C        -1.211546    0.699486   -3.054539  f=b1
H        -2.157190    1.245454   -3.049828  f=b1
C        -1.211546   -0.699486    3.054539  f=b2
```

```
H        -2.157190   -1.245454    3.049828  f=b2
C         0.000000   -1.398973    3.054539  f=b2
H         0.000000   -2.490908    3.049828  f=b2
C         1.211546   -0.699486    3.054539  f=b2
H         2.157190   -1.245454    3.049828  f=b2
C         1.211546    0.699486    3.054539  f=b2
H         2.157190    1.245454    3.049828  f=b2
C         0.000000    1.398973    3.054539  f=b2
H         0.000000    2.490908    3.049828  f=b2
C        -1.211546    0.699486    3.054539  f=b2
H        -2.157190    1.245454    3.049828  f=b2
H         0.000000    0.000000   -0.377906  f=h2
H         0.000000    0.000000    0.377906  f=h2
End
End Input
eor
```

ZORA and spin-orbit Relativistic Effects

Au_2: ZORA Relativistic Effects

Sample directory: adf/Au2_ZORA/

Another relativistic geometry optimization, now with the ZORA formalism. The build-up is quite similar to the RelGO_AuH case: DIRAC calculations for the involved atoms to get relativistic core potentials, Create runs and finally the molecular optimization run. In between the Create runs and the molecular optimization run, a single-atom Spin-Orbit calculation is carried out. The Spin-Orbit corrections are not available in optimization calculations, so in the final molecular run, the *scalar* (ZORA) relativistic terms are used.

```
$ADFBIN/adf << eor

Title  Au  relativistic spinorbit

Integration  6.5

Atoms
  Au  0 0 0
End

Fragments
  Au   t21.Au
End

XC
  GGA Becke Perdew
End

Relativistic SpinOrbit ZORA
Corepotentials  t12.rel

end input
```

```
eor
```

Since only one type of atom is used, the CorePotentials key can be used as simple key: the data block is not necessary since the program takes (by default) the first section on the TAPE12 file for the first (here: only) atom type in the calculation.

```
$ADFBIN/adf << eor
Title  Au2  relativistic optimization: scalar ZORA

Integration  6.5

Atoms  Zmat
  Au   0 0 0
  Au   1 0 0  2.5
End

Fragments
  Au   t21.Au
End

XC
 GGA Becke Perdew
End

Relativistic scalar ZORA
CorePotentials  t12.rel

Geometry
  convergence grad=1e-4
End

End Input
eor
```

Bi and Bi_2: Spin-Orbit

Sample directory: adf/SO_Bi2/

Application of the Spin-Orbit relativistic option (using double-group symmetry) to Bismuth (atom and dimer).

To prepare for the relativistic calculations, the *dirac* program is applied to generate the relativistic core potential for the Bismuth atom with a frozen core up to the 5p shell.

```
$ADFBIN/dirac -n1 < $ADFRESOURCES/Dirac/Bi.5p

mv TAPE12 t12rel
```

The next step is the creation of the restricted Bismuth atom (scalar relativistic).

The GGA (Becke-Perdew) facility is used for consistency with the calculations to follow, but is not necessary *per se* to carry out the subsequent calculations.

```
$ADFBIN/adf <<eor
create Bi  file=$ADFRESOURCES/TZP/Bi.5p
xc
  LDA vwn
  GGA  becke perdew
end
relativistic scalar
corepotentials  t12rel  &
Bi     1
end
end input
eor

mv TAPE21 t21Bi
```

Note that usage of the *block* form for the CorePotentials key would not have been necessary here. We could as well have used:

corepotentials t12rel

instead of

```
corepotentials  t12rel  &
Bi     1
end
```

Bi: single atom

For comparision with the full double-group calculation, the 'standard' unrestricted calculation on Bismuth is carried out, using the *scalar* relativistic option.

A net spin polarization of 3 electrons is applied (key charge).

```
$ADFBIN/adf <<eor

title Bi unrestricted

integration 4.0

xc
  LDA vwn
  GGA  becke perdew
end

relativistic scalar
corepotentials  t12rel  &
Bi    1
end

ATOMS
Bi   0.000000      0.000000      0.00000000
end

fragments
```

```
Bi t21Bi
end

unrestricted

charge 0 3

end input
eor
```

The CHARGE key, in conjunction with the UNRESTRICTED key is used to specify that 3 electrons must be unpaired (second value of the CHARGE key), while the system is neutral (first value of the CHARGE key).

Next we do a Spin-Orbit calculation on the Bismuth atom.

Note that it is a 'restricted' run (the key unrestricted is not used). The double-group symmetry orbitals are, like the single-group ones in a non-SpinOrbit calculation, degenerate, allowing 2 electrons in each spatial orbital. These are equally occupied (using fractional occupations if necessary) and the electronic charge density is not spin-polarized.

```
$ADFBIN/adf <<eor
title Bi spinorbit

integration 4.0

xc
  LDA vwn
  GGA  becke perdew
end

relativistic spinorbit
corepotentials  t12rel  &
Bi    1
end

ATOMS
Bi   0.000000      0.000000      0.00000000
end

fragments
Bi t21Bi
end

end input
eor
```

Comparison of the bonding energy (w.r.t. the create restricted atom) for the scalar relativistic and spin-orbit runs respectively show that application of the spin-orbit operator lowers the energy by approximately 1.1 eV.

In the previous run default occupations were used: the occupations were determined from the aufbau principle during the first few scf iterations.

The following is an excited state calculation: occupation numbers are specified in input and by comparison with the result from the previous run we see that one electron has been promoted from a *p*1/2 to a *p*3/2 orbital.

```
$ADFBIN/adf <<eor
title Bi spinorbit, specified occupations

PRINT SpinOrbit

integration 4.0

xc
  LDA vwn
  GGA  becke perdew
end

relativistic spinorbit
corepotentials  t12rel  &
Bi    1
end

ATOMS
Bi   0.000000      0.000000      0.00000000
end

fragments
Bi t21Bi
end

charge 0

occupations
 s1/2  2
 p1/2  1
 p3/2  2
 d3/2  4
 d5/2  6
end

end input
eor
```

The PRINT key (here with argument SPINORBIT) controls output printing. Here it induces the printing of some extra information about the relativistic double group symmetry orbitals.

Bi_2 dimer

Now we turn to the dimer Bi_2: a series of Single Point calculations, all with the same inter atomic distance.

First the scalar relativistic run.

```
$ADFBIN/adf <<eor
title   Bi2, scalar relativistic
```

```
integration 4.0

relativistic scalar
corepotentials  t12rel  &
Bi    1
end

ATOMS
Bi       0.0             0.0             1.33
Bi       0.0             0.0            -1.33
end

fragments
Bi     t21Bi
end

xc
  LDA vwn
  GGA  becke perdew
end

end input
eor

mv tape21 t21Bi2
```

The result file tape21 is used as reference in subsequent calculations: run the spin-orbit case starting from the just completed dimer calculation as a fragment. The resulting 'bonding energy', ie the energy w.r.t. the scalar relativistic dimer, gives directly the effect of the full-relativistic versus the scalar relativistic option: the energy is lowered by 2.3 eV.

```
$ADFBIN/adf <<eor
title   Bi2 from fragment Bi2,  with SpinOrbit coupling

PRINT SpinOrbit

integration 4.0

relativistic spinorbit
corepotentials  t12rel  &
Bi    1
end

ATOMS
Bi    0.0    0.0    1.33  f=Bi2
Bi    0.0    0.0   -1.33  f=Bi2
end

fragments
Bi2     t21Bi2
end

xc
```

```
  LDA vwn
  GGA  becke perdew
end

end input
eor

rm TAPE21 logfile
```

A final consistency check: run the spin-orbit dimer from single-atom fragments. The bonding energy should equal the sum of the bonding energies of the previous two runs: scalar relativistic dimer w.r.t. single atom fragments plus spin-orbit dimer w.r.t. the scalar relativistic dimer.

```
$ADFBIN/adf <<eor
title   Bi2 from atomic fragments, SpinOrbit coupling

PRINT SpinOrbit

integration 4.0

relativistic spinorbit
corepotentials  t12rel  &
Bi    1
end

ATOMS
Bi       0.0             0.0             1.33
Bi       0.0             0.0            -1.33
end

fragments
Bi     t21Bi
end

xc
  LDA vwn
  GGA  becke perdew
end

end input
eor
```

Tl: Spin-Orbit unrestricted non-collinear

Sample directory: adf/Tl_noncollinear/

Application of the Spin-Orbit relativistic option (using double-group symmetry, in this case NOSYM) to Tl using the collinear and non-collinear approximation for unrestricted Spin-Orbit calculations

Note: For the collinear and the non-collinear approximation one should use symmetry NOSYM and use the key UNRESTRICTED.

The non-collinear example:

```
$ADFBIN/adf  << eor
Title Tl spinorbit noncollinear
Atoms
 Tl 0 0 0
End

Relativistic  Spinorbit ZORA
COREPOTENTIALS t12.rel   &
 Tl 1
End

XC
 gradients becke perdew
end

symmetry nosym
unrestricted
noncollinear

Fragments
 Tl t21.Tl
End

End input
eor
```

If one replaces the key NONCOLLINEAR with COLLINEAR the collinear approximation will be used instead of the non-collinear approximation. In the case of the collinear approximation default the direction of the magnetization is in the direction of the z-axis. In the non-collinear approximation the magnetization can differ in each point in space.

AuH: excitation energies including spin-orbit coupling

Sample directory: adf/AuH_analyse_exciso/

Calculation of the excitation energies of AuH including spin-orbit coupling.

```
$ADFBIN/adf << eor
Title [AuH]
Atoms
 Au .0000  .0000  1.5238
  H .0000  .0000  0.0000
End
relativistic scalar zora
Basis
 Type TZ2P
 Core None
End
symmetry C(7v)
EPRINT
```

```
SFO eig ovl
END
integration 6.0
Excitations
 lowest 40
End
End input
eor

mv TAPE21 t21.frag
rm logfile

$ADFBIN/adf << eor
Title [AuH]
Atoms
 Au .0000  .0000  1.5238  f=Frag
  H .0000  .0000  0.0000  f=Frag
End
relativistic spinorbit zora
symmetry C(7v)
EPRINT
SFO eig ovl
END
integration 6.0
Excitations
 lowest 40
End
Fragments
 Frag t21.frag
End
STCONTRIB
End input
eor
```

ADF can not handle ATOM and linear symmetries in excitation calculations. Therefore a subsymmetry is used, in this case symmetry C(7v).

A relatively small TZ2P basis set is used, which is not sufficient for excitations to Rydberg-like orbitals, one needs more diffuse functions.

The key STCONTRIB is used, which will give a composition of the spin-orbit coupled excitation in terms of singlet-singlet and singlet-triplet scalar relativistic excitations. In order to use the key STCONTRIB the scalar relativistic fragment should be the complete molecule.

Starting from ADF2008.01 one needs to include the subkey SFO of the key EPRINT with arguments eig and ovl in order to get the SFO MO coefficients and SFO overlap matrix printed on standard output.

Solvents, other environments

HCl: COSMO

Sample directory: adf/Solv_HCl/

Computing solvent effects, with the COSMO model, is illustrated in the HCl example.

After a non-solvent (reference) calculation, which is omitted here, two solvent runs are presented, with somewhat different settings for a few input parameters. The block key Solvation controls all solvent-related input.

All subkeys in the SOLVATION block are discussed in the User's Guide. Most of them are rather technical and should not severely affect the outcome. Physically relevant is the specification of the solute properties, by the SOLVENT subkey: the dielectric constant and the effective radius of the solvent molecule.

A rather strong impact on the computation times has the method of treating the 'C-matrix'. There are 3 options (see the User's Guide): EXACT is the most expensive, but presumably most accurate. POTENTIAL is the cheapest alternative and is usually quite adequate. EXACT uses the exact charge density for the Coulomb interaction between the molecular charge distribution and the point charges (on the Van der Waals type molecular surface) which model the effects of the solvent. The alternatives, notably 'POTENTIAL', use the *fitted* charge density instead. Assuming that the fit is a fairly accurate approximation to the exact charge density, the difference in outcome should be marginal.

```
$ADFBIN/adf << eor
TITLE  HCl(1) Solv-excl surfac; Gauss-Seidel (old std options)

SYMMETRY  NOSYM

ATOMS Cartesian
  H    0.000000     0.000000    0.000000    R=1.18
  Cl   1.304188     0.000000    0.000000    R=1.75
END

Fragments
 H  t21.H
 Cl t21.Cl
End

SOLVATION
   Solvent        epsilon=78.8 radius=1.4
   SurfaceType    esurf
   DivisionLevel  ND=4  min=0.5  Ofac=0.8
   ChargeUpdate   Method=Gauss-Seidel
   DiscAttributes SCale=0.01  LEGendre=10 TOLerance=1.0d-2
   SCF            Variational
   C-Matrix       Exact
END

NOPRINT Bas EigSFO EKin SFO, frag, functions
EPRINT
SCF NoEigvec
END
END INPUT
eor

rm TAPE21 logfile
```

In the second solvent run, another (technical) method is used for determining the charge distribution on the cavity surface (conjugate-gradient versus Gauss-Seidel in the previous calculation), and the POTENTIAL variety is used for the C-matrix handling. The results show that it makes little difference in outcome, but quite a bit in computation times.

```
$ADFBIN/adf << eor
TITLE  HCl(9) NoDisk and Cmatrix potential

FRAGMENTS
   H    t21.H
   Cl   t21.Cl
END

ATOMS Cartesian
  H    0.000000     0.000000    0.000000    R=1.18
  Cl   1.304188     0.000000    0.000000    R=1.75
END

SOLVATION
   Solvent        epsilon=78.8 radius=1.4
   SurfaceType    esurf
   DivisionLevel  ND=4  min=0.5  Ofac=0.8
   ChargeUpdate   Method=conjugate-gradient
   SCF            Variational
   C-Matrix       POTENTIAL
END

NOPRINT Bas EigSFO EKin SFO, frag, functions
EPRINT
SCF NoEigvec
END
END INPUT
eor
```

N_2 and PtCO: Electric Field, Point Charge(s), use of Basis keyword

*Sample directories:*adf/Efield.PntQ_N2/ and adf/Field_PtCO

Two illustrations of applying the very useful BASIS keyword and of application of an Electric Field.

For N2, three calculations are provided: 1) a normal N_2 run as a reference with the BASIS keyword, 2) with a homogeneous electric field, 3) with a point charge.

In this example, no Create run is needed in the input file, because the first molecular calculation uses the BASIS keyword. If the $ADFBIN/adf script finds this keyword, it will first generate a new input file which will then be executed. The new input file will contain the required Create run for the N atom in this case. The proper xc functional and relativistic options will automatically be selected by the BASIS keyword. This includes Dirac calculations in case of relativistic runs. The output files is identical to what would have appeared if one would provide the Create runs explicitly in the input file. It also copies the atomic input, so that everything can be checked.

```
$ADFBIN/adf -n1 << eor
title N2  reference for comparison with E-Field runs

atoms
 N  0 0 -.55
 N  0 0 +.55
end

Basis
 Type DZP
 Core Small
End

 end input
eor

rm TAPE21 logfile

$ADFBIN/adf << eor
scf
 conv 1e-8
end

title N2 in a homogeneous electric field

atoms
 N  0 0 -.55
 N  0 0 +.55
end
fragments
 N  t21.N
end

EField    0 0 0.01

end input
eor

rm TAPE21 logfile

$ADFBIN/adf << eor
title  N2 polarized by a point charge on the axis

EField
 0 0 3.0   1.0
end

atoms
 N 0 0 -.55
 N 0 0 .55
end

Fragments
 N  t21.N
```

```
end

endinput
eor
```

In the second n_2 run the homogeneous field is supplied with the key efield, used as simple key: one record, data on the same line as the keyword. The field strength is specified in atomic units.

Homogeneous electric fields can be used to study the polarizability: for sufficiently small fields the dipole moment should respond linearly.

For point charges, the third calculation, the block form of the key efield must be used. The program first tries to find data on the same line as the keyword (defining a homogeneous field). If this is absent, a data block is expected with point-charge specifications: x, y, z and q.

The coordinates are in the same units as in the atoms block (angstrom by default) (but always Cartesian). Q is the charge in elementary units (+1 for a proton).

Point charges can be used for instance to simulate crystal fields (Madelung potential).

Note: the symmetry will be determined automatically by the program as C(lin), rather than D(lin), in the two runs that involve an electric field: the fields break the symmetry.

For PtCO, a fairly large electric field is applied in combination with a tight SCF convergence criterion.

The BASIS keyword in this example illustrates how different choices can be made for different atoms (in this case a frozen core for Pt).

```
Basis
 Type DZ
 Core None
 Pt Pt.4d
END
```

FDE: Frozen Density Embedding

H_2O in water: FDE

Sample directory: adf/FDE_H2O_128/

This example demonstrates how to use FDE in combination with a large environment, that is modeled as a superposition of the densities of isolated molecules. Here, the excitation energies of a water molecule surrounded by an environment of 127 water molecules. For details, see C.R. Jacob, J. Neugebauer, L. Jensen, L. Visscher, Phys. Chem. Chem. Phys., 2006 **8**: 2349.

This calculation consists of two steps:

- First a prototype water molecule is calculated.
- Next the embedding calculation of water in water is performed.

To reduce the amount of output the next lines are included in the adf calculations:

```
EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END
NOPRINT BAS FUNCTIONS
```

First, a prototype water molecule is calculated. The density of this isolated water molecules will afterwards be used to model the environment. Since this molecule will be used as a frozen fragment that is rotated and translated, the option NOSYMFIT has to be included.

```
$ADFBIN/adf << eor
Title Input generated by modco

UNITS
  length bohr
  angle degree
END

XC
LDA
END

SYMMETRY NOSYM

GEOMETRY
  sp
END

SCF
  iterations 50
  converge 1.0e-6 1.0e-6
  mixing 0.2
  lshift 0.0
  diis n=10 ok=0.5 cyc=5 cx=5.0 cxx=10.0
END
INTEGRATION 5.0 5.0

FRAGMENTS
  O  t21.DZP.O
  H  t21.DZP.H
END

ATOMS
 O       -11.38048700000000    -11.81055300000000     -4.51522600000000
 H       -13.10476265095705    -11.83766918322447     -3.96954531282721
 H       -10.51089289290947    -12.85330720999229     -3.32020577897331
END

ENDINPUT
eor

mv TAPE21 t21.mol_1
```

Afterwards, the FDE calculation is performed. In this FDE calculation, there is one nonfrozen water molecule and the previously prepared water molecule is included as a frozen fragment that is duplicated 127 times. For this frozen fragment, the more efficient fitted density is used.

```
$ADFBIN/adf << eor
Title Input generated by modco

UNITS
  length bohr
  angle degree
END

XC
MODEL SAOP
END

SYMMETRY NOSYM

SCF
  iterations 50
  converge 1.0e-6 1.0e-6
  mixing 0.2
  lshift 0.0
  diis n=10 ok=0.5 cyc=5 cx=5.0 cxx=10.0
END

EXCITATION
  ONLYSING
  LOWEST  5
END

INTEGRATION 4.0 4.0

FRAGMENTS
  O      t21.DZP.O
  H      t21.DZP.H
  frag1  t21.mol_1 type=fde &
    fdedenstype SCFfitted
  SubEnd
END

ATOMS
 O   0.00000000000000   0.00000000000000   0.00000000000000
 H  -1.43014300000000   0.00000000000000   1.10739300000000
 H   1.43014300000000   0.00000000000000   1.10739300000000
 O -11.38048700000000 -11.81055300000000  -4.51522600000000  f=frag1/1
 H -13.10476265095705 -11.83766918322447  -3.96954531282721  f=frag1/1
 H -10.51089289290947 -12.85330720999229  -3.32020577897331  f=frag1/1
 O  -1.11635000000000   9.11918600000000  -3.23094800000000  f=frag1/2
 H  -2.82271357869859   9.71703285239153  -3.18063201242303  f=frag1/2
 H  -0.12378551814273  10.53819303003839  -2.70860866559857  f=frag1/2
...
 O   5.96480100000000   4.51370300000000   3.70332800000000  f=frag1/127
 H   5.24291272273548   3.06620845434369   2.89384293177905  f=frag1/127
```

```
 H    4.73614594944492   5.00201400735317   4.93765482424434  f=frag1/127
END

FDE
  PW91K
END


ENDINPUT
eor
```

$HeCO_2$: FDE freeze-and-thaw

Sample directory: adf/FDE_HeCO2_freezeandthaw/

This example demonstrates how a freeze-and-thaw FDE calculation can be performed. As test system, a He-CO_2van der Waals complex is used. It will further be shown how different exchange-correlation potential can be used for different subsystems, and how different basis set expansions can be employed. For details, see C.R. Jacob, T.A. Wesolowski, L. Visscher, J. Chem. Phys. 123 (2005), 174104. It should be stressed that the basis set and integration grid used in this example are too small to obtain good results.

Summary:

- PW91 everywhere
- SAOP for He; PW91 for CO_2
- FDE(s) calculation with PW91 everywhere

PW91 everywhere

In the first part, the PW91 functional will be used for both the He and the CO_2 subsystems. In this part, the FDE(m) basis set expansion is used, i.e., basis functions of the frozen subsystem are not included in the calculation of the nonfrozen subsystem.

First, the CO_2 molecule is prepared. In this calculation, the C2v symmetry of the final complex is used, and the NOSYMFIT option has to be included because this molecule will be rotated as a frozen fragment.

```
$ADFBIN/adf << eor
Title TEST 1 -- Preparation of frozen CO2

Units
 Length Bohr
end

Atoms
C          0.000000  0.000000  0.000000
O         -2.192000  0.000000  0.000000
O          2.192000  0.000000  0.000000
end

Symmetry C(2V)
NOSYMFIT
```

```
Fragments
 C  t21.C
 O  t21.O
End

integration 5.0

xc
 GGA pw91
end

End Input
eor

mv TAPE21 t21.co2.0
```

Afterwards, the FDE calculation is performed. In this calculation, the He atom is the nonfrozen system, and the previously prepared CO_2 molecule is used as frozen fragment. For this frozen fragment the RELAX option is specified, so that the density of this fragment is updated in freeze-and-thaw iteration (a maximum number of three iteration is specified).

```
$ADFBIN/adf << eor
Title TEST 1 -- Embedding calulation: He + frozen CO2 density -- freeze-and-thaw

Units
 Length Bohr
end

Atoms
 He   0.000000  0.000000  6.019000 f=He
 C    0.000000  0.000000  0.000000 f=co2
 O   -2.192000  0.000000  0.000000 f=co2
 O    2.192000  0.000000  0.000000 f=co2
end

Fragments
 He   t21.He
 co2  t21.co2.0  type=fde   &
   fdeoptions RELAX
 SubEnd
End

NOSYMFIT

integration 5.0

xc
 GGA pw91
end

FDE
 PW91K
 FULLGRID
```

```
 RELAXCYCLES 3
end

End Input
eor
```

SAOP for He; PW91 for CO_2

In this second part, the above example is modified such that PW91 is employed for the CO_2 subsystem, while the SAOP potential is used for He. This can be achieved by choosing SAOP in the XC key (this sets the functional that will be used for the nonfrozen subsystem). Additionally, for the frozen fragment the XC option is used to chose the PW91 functional for relaxing this fragment. Furthermore, the PW91 functional is chosen for the nonadditive exchange-correlation functional that is used in the embedding potential with the GGAPOTXFD and GGAPOTCFD options in the FDE key.

```
$ADFBIN/adf << eor
Title TEST 2 -- Embedding calulation: He + frozen CO2 density -- freeze-and-thaw

Units
 Length Bohr
end

Atoms
 He   0.000000  0.000000  6.019000 f=He
 C    0.000000  0.000000  0.000000 f=co2
 O   -2.192000  0.000000  0.000000 f=co2
 O    2.192000  0.000000  0.000000 f=co2
end

Fragments
 He   t21.He
 co2  t21.co2.0  type=fde   &
   fdeoptions RELAX
   XC         GGA PW91
 SubEnd
End

NOSYMFIT

integration 5.0

xc
 MODEL SAOP
end

FDE
 PW91K
 FULLGRID
 GGAPOTXFD PW91x
 GGAPOTCFD PW91c
 RELAXCYCLES 3
end
```

```
End Input
eor
```

FDE(s) calculation with PW91 everywhere

In this third part, the PW91 functional is applied for both subsystems again, but in contrast to part 1, now the FDE(s) basis set expansion is used, i.e., the basis functions of the frozen subsystem are included in the calculation of the nonfrozen subsystem. This can be achieved by employing the USEBASIS option. This option can be combined with the RELAX option.

```
$ADFBIN/adf << eor
Title TEST 3 -- Embedding calulation: He + frozen CO2 density -- freeze-and-thaw

Units
 Length Bohr
end

Atoms
 He   0.000000  0.000000  6.019000 f=He
 C    0.000000  0.000000  0.000000 f=co2
 O   -2.192000  0.000000  0.000000 f=co2
 O    2.192000  0.000000  0.000000 f=co2
end

Fragments
 He   t21.He
 co2  t21.co2.0  type=fde   &
   fdeoptions RELAX USEBASIS
 SubEnd
End

NOSYMFIT

integration 5.0

xc
 GGA pw91
end

FDE
 PW91K
 FULLGRID
 RELAXCYCLES 3
end

End Input
eor
eor
```

The example continues with the same calculation where partly the SAOP potential is used.

NH_3-H_2O: FDE energy

Sample directory: adf/FDE_Energy_NH3-H2O/

This is example for a calculation of FDE interaction energies in ADF in case of closed shell fragments.

It performs single point runs for H_2O and NH_3 with LDA/DZ (all-electron) and uses these fragments in:

- an FDE energy embedding calculation calculation in which the energy of water in presence of a frozen ammonia is computed This requires a supermolecular integration grid
- a fully variational FDE energy calculation (with freeze-and-thaw)

Integration accuracy is 6.0 which should give total energies for the fragments accurate at least up to 10**(-4) atomic units.

```
$ADFBIN/adf << EOF
Title H2O LDA/DZ single point
 ATOMS
         O        1.45838        0.10183        0.00276
         H        0.48989       -0.04206        0.00012
         H        1.84938       -0.78409       -0.00279
 END
 SYMMETRY tol=1e-2
 BASIS
   Type DZ
   Core None
 END
 XC
   LDA
 END
 INTEGRATION
  accint  6.0
 END
 NOSYMFIT
EOF
rm logfile
mv TAPE21 t21.water
EOF
```

In a similar way the N_3 fragment is calculated. Next the FDE calculation is performed. The subkey ENERGY of the key FDE is used, such that the total FDE energy and FDE interaction energy is calculated. First an FDE energy embedding calculation calculation in which the energy of water in presence of a frozen ammonia is computed. This requires a supermolecular integration grid.

```
$ADFBIN/adf << EOF
Title NH3-H2O LDA/Thomas-Fermi/DZ FDE single point with interaction energy
 ATOMS
         O        1.45838        0.10183        0.00276    f=frag1
         H        0.48989       -0.04206        0.00012    f=frag1
         H        1.84938       -0.78409       -0.00279    f=frag1
         N       -1.51248       -0.03714       -0.00081    f=frag2
         H       -1.71021        0.95994       -0.11003    f=frag2
```

```
        H       -1.96356       -0.53831       -0.76844    f=frag2
        H       -1.92899       -0.35123        0.87792    f=frag2
 END
 SYMMETRY tol=1e-2
 FRAGMENTS
  frag1  t21.water
  frag2  t21.ammonia type=FDE
 END
 XC
   LDA
 END
 INTEGRATION
  accint  6.0
 END
 EXACTDENSITY
 FDE
   THOMASFERMI
   FULLGRID
   ENERGY
 END
EOF
```

Next a fully variational FDE energy calculation (with freeze-and-thaw) is performed.

```
$ADFBIN/adf << EOF
Title NH3-H2O LDA/Thomas-Fermi/DZ FDE single point with interaction energy
 ATOMS
        O        1.45838        0.10183        0.00276    f=frag1
        H        0.48989       -0.04206        0.00012    f=frag1
        H        1.84938       -0.78409       -0.00279    f=frag1
        N       -1.51248       -0.03714       -0.00081    f=frag2
        H       -1.71021        0.95994       -0.11003    f=frag2
        H       -1.96356       -0.53831       -0.76844    f=frag2
        H       -1.92899       -0.35123        0.87792    f=frag2
 END
 SYMMETRY tol=1e-2
 FRAGMENTS
  frag1  t21.water
  frag2  t21.ammonia type=FDE  &
     fdeoptions RELAX
  SubEnd
 END
 XC
   LDA
 END
 INTEGRATION
  accint  6.0
 END
 EXACTDENSITY
 SAVE TAPE21
 FDE
   THOMASFERMI
   RELAXCYCLES 3
   ENERGY
```

```
 END
EOF
```

Ne-H_2O: FDE energy, unrestricted fragments

Sample directory: adf/FDE_Energy_H2O-Ne_unrestricted/

This is example for a calculation of FDE interaction energies in ADF for an open-shell frozen fragment.

It performs single point runs for H2O and Ne, the latter unrestricted with LDA/DZ (all-electron) and uses these fragments in an FDE energy embedding calculation in which the energy of water in presence of a frozen (open-shell) neon atom is computed. This is a bit of an artificial example but it serves its purpose.

No freeze-thaw is done, this is at present not possible with unrestricted (open shell) fragments, but has to be done manually.

Integration accuracy is 6.0 which should give total energies for the fragments accurate at least up to 10**(-4) atomic units.

This test has been checked to yield the same energy as a run with a closed- shell (restricted) Ne atom (just comment UNRESTRICTED in the input below). First the Ne and H_2O fragments are calculated.

```
$ADFBIN/adf << EOF
Title Ne LDA/DZ single point, unrestricted
 ATOMS
         Ne      -1.51248       -0.03714       -0.00081
 END
 UNRESTRICTED
 BASIS
   Type DZ
   Core None
 END
 INTEGRATION
  accint  6.0
 END
 SCF
   iterations  100
   converge 1.0e-06 1.0e-06
 END
 EXACTDENSITY
 NOSYMFIT
EOF

rm logfile
mv TAPE21 t21.ne
EOF
```

In a similar way the H_2O fragment is calculated. Next the FDE calculation is performed. The subkey ENERGY of the key FDE is used, such that the total FDE energy and FDE interaction energy is calculated.

```
$ADFBIN/adf << EOF
Title Ne-H2O LDA/Thomas-Fermi/DZ FDE single point with interaction energy
```

```
 ATOMS
         O        1.45838        0.10183        0.00276    f=frag1
         H        0.48989       -0.04206        0.00012    f=frag1
         H        1.84938       -0.78409       -0.00279    f=frag1
         Ne      -1.51248       -0.03714       -0.00081    f=frag2
 END

 SYMMETRY tol=1e-2

 FRAGMENTS
  frag1  t21.water
  frag2  t21.ne type=FDE
 END

 INTEGRATION
  accint  6.0
 END

 SCF
   iterations  100
   converge 1.0e-06 1.0e-06
 END

 EXACTDENSITY

 FDE
   THOMASFERMI
   FULLGRID
   ENERGY
 END
EOF
```

H_2O-Li(+): FDE geometry optimization

Sample directory: adf/GO_FDE_H2O-Li/

This examples checks the gradient implementation for FDE. It performs a structure optimization H2O-Li(+) with LDA/DZP.

First, the fragments are made, Li^+, and water. Next the FDE geometry optimization is performed with:

```
$ADFBIN/adf << eor
TITLE H2O-Li(+) FDE/LDA/DZP GO New Optimizer starting at too short Li-O distance
ATOMS
 Li      0.000000000000       0.000000000000      -0.054032208082
 O       0.000000000000       0.000000000000      -1.534032208080  f=water
 H      -0.778216093965       0.000000000000      -2.135966332900  f=water
 H       0.778216093965       0.000000000000      -2.135966332900  f=water
END
CHARGE 1.0
```

```
FRAGMENTS
 Li    t21.Li.LDA.DZP
 water t21.water.LDA.DZP type=fde
END
XC
 LDA VWN
END
FDE
 ThomasFermi
END
GEOMETRY
 Optim Delocalized
 iterations 15
 Converge e=1.0e-3 grad=1.0e-3
END
GEOSTEP GradientTerms
INTEGRATION 5.0 5.0 5.0
eor
```

NH_3-H_2O: FDE geometry optimization

Sample directory: adf/GO_FDE_NH3-H2O/

This examples performs a structure optimization of H_2O in presence of frozen NH_3 (via optimization of selected coordinates) with LDA and DZ basis. We need a high accint of 6.0 here because the potential energy surface is rather flat and small errors might lead to discrepancies in final structures. It uses (at present) the old branch optimizer for this purpose.

First, the NH_3 fragment is made. Next the FDE geometry optimization is performed with:

```
$ADFBIN/adf << eor
TITLE NH3-H2O dimer FDE LDA DZ structure optimization of H2O
ATOMS
      N     -1.01393958      -0.15260815       0.00000000    f=nh3
      H     -1.16290010      -1.15738765       0.00000000    f=nh3
      H     -1.49925696       0.21074929       0.81414267    f=nh3
      H     -1.49925696       0.21074929      -0.81414267    f=nh3
      O     Ox  Oy  Oz
      H     H1x H1y H1z
      H     H2x H2y H2z
END
GEOVAR
 Ox    2.25288687
 Oy   -0.00423586
 Oz    0.00000000
 H1x   1.28270504
 H1y   0.05211069
 H1z   0.00000000
 H2x   2.54788803
 H2y   0.90516678
 H2z   0.00000000
```

```
END
FRAGMENTS
 O t21.O_LDA_DZ
 H t21.H_LDA_DZ
 nh3 t21.NH3_LDA_DZ type=fde
END
FDE
 THOMASFERMI
END
XC
 LDA VWN
END
GEOMETRY
 Branch Old
 Optim Selected
 Iterations 100 ! (default is 30)
END
INTEGRATION 6.0 6.0 6.0
eor
```

Acetonitrile in water: FDE NMR shielding

Sample directory: adf/FDE_NMR_relax/

This examples demonstrates both the calculation of NMR shieldings using FDE, and how the approximate environment density can be improved by partial relaxation of individual solvent molecules. The test system is a cluster of acetonitrile and 12 solvent water molecules, of which for two the densities are relaxed, while for the remaining 10 the frozen density of the isolated water is used. For details, see Refs.
C. R. Jacob, J. Neugebauer, and L. Visscher, A flexible implementation of frozendensity embedding for use in multilevel simulation, submitted, 2007.
R. E. Bulo, Ch. R. Jacob, and L. Visscher, NMR Solvent Shifts of Acetonitrile from Frozen-Density Embedding Calculation, to be submitted, 2007

First, the isolated solvent water molecule is prepared. Again, because this will be rotated and translated afterwards, the option NOSYMFIT has to be included.

```
$ADFBIN/adf << eor

UNITS
  Length Angstrom
  Angle Degree
END

ATOMS
     O       -1.46800        2.60500        1.37700
     H       -0.95200        3.29800        0.96500
     H       -1.16100        1.79900        0.96100
END

FRAGMENTS
  H        t21.H.DZP
```

```
  O        t21.O.DZP
END

XC
  LDA
END

INTEGRATION
 accint  4.0
END

end input
eor

mv TAPE21 t21.h2o
```

Afterwards, the FDE calculation is performed. In addition to the nonfrozen acetonitrile molecule, three different fragments are used for the solvent water molecules. The first two fragments frag1 and frag2 are relaxed (in up to two freeze-and-thaw cycles), while the third fragment is used for the remaining 10 solvent molecules. Since a calculation of the shielding is performed afterwards, the option has to be included.

```
$ADFBIN/adf << eor
Title Input generated by PyADF

UNITS
  Length Angstrom
  Angle Degree
END

ATOMS
     C        0.83000        0.66100       -0.44400
     N        0.00000        0.00000        0.00000
     C        1.87800        1.55900       -0.81900
     H        1.78500        2.40300       -0.13500
     H        1.76200        1.94900       -1.83000
     H        2.82900        1.12200       -0.51300
     O       -1.46800        2.60500        1.37700    f=frag1/1
     H       -0.95200        3.29800        0.96500    f=frag1/1
     H       -1.16100        1.79900        0.96100    f=frag1/1
     O        2.40400       -2.51000       -0.36200    f=frag2/1
     H        2.70000       -3.41900       -0.40900    f=frag2/1
     H        1.77500       -2.50000        0.35900    f=frag2/1
 ...
     O       -3.44400        2.36700        3.13700    f=frag3/10
     H       -2.70200        2.29200        2.53700    f=frag3/10
     H       -3.47300        3.29500        3.36800    f=frag3/10
END


FRAGMENTS
  H        t21.H.DZP
  C        t21.C.DZP
  N        t21.N.DZP
  frag1  t21.h2o   type=FDE &
```

```
    fdeoptions RELAX
    RELAXCYCLES 2
  SubEnd
  frag2  t21.h2o   type=FDE &
    fdeoptions RELAX
    RELAXCYCLES 2
  SubEnd
  frag3  t21.h2o   type=FDE &
    FDEDENSTYPE SCFexact
  SubEnd
END

XC
  GGA BP86
END

INTEGRATION
 accint  4.0
END

SAVE TAPE10

FDE
  PW91k
END

End Input
eor
```

Finally, the calculation of the NMR shielding of the nitrogen atom is performed using the NMR program.

```
$ADFBIN/adf << eor
NMR
 out tens iso
 nuc 3
END
eor
```

QM/MM calculations

pdb2adf: transforms a PDB file in a QM/MM adf-input file

Sample directory: adf/pdb2adf/

This example shows how to use the utiliy pdb2adf, which creates an ADF input file from a PDB file, for a subsequent QM/MM calculation using ADF. See also the Utilities document for more examples.

First create the PDB file that can be used in this example.

```
cat << eor > chymotrypsin.pdb
HEADER    COMPLEX (SERINE PROTEASE/INHIBITOR)     12-MAR-97   1AFQ
TITLE     CRYSTAL STRUCTURE OF BOVINE GAMMA-CHYMOTRYPSIN COMPLEXED
TITLE    2 WITH A SYNTHETIC INHIBITOR
```

```
REMARK
REMARK    Adaptation of original PDB file by M. Swart, March 2005
REMARK    only coordinates of GAMMA-CHYMOTRYPSIN are kept;
REMARK    rest has been deleted.
REMARK
ATOM      1  N   CYS A   1      13.717  20.021  22.754  1.00 13.46      PROA N
ATOM      2  CA  CYS A   1      14.211  18.932  23.617  1.00 13.34      PROA C
ATOM      3  C   CYS A   1      13.597  19.033  25.005  1.00 13.34      PROA C
...
ATOM     68  CD2 LEU A  10       9.768  11.681  39.555  1.00 27.46      PROA C
ATOM     69  OXT LEU A  10       6.329  11.066  42.743  1.00 27.55      PROA O
TER      70      LEU A  10
END
eor
```

Then run the pdf2adf program to create ADF inputfile

```
$ADFBIN/pdb2adf << eor

chymotrypsin.pdb



4 5
c


Y
1
1
17.5
eor
```

The program works interactively. The input described here are answers to the questions that were asked interactively. In cases where the user agrees with the suggestion given by the program, the user can press the `Enter` key, which is shown here with an empty line.

The questions asked can be found in the output file, and are repeated here. The `Enter` key or empty line is indicated here with **Enter**.

```
Do you want a logfile to be written (Y/n) ?
```

Enter

```
Please give name of PDB-file
```

chymotrypsin.pdb

```
Found the following terminal amino acid residues : (C-term)    10 (N-term)     1
Do you want to use these as terminal residues (Y/n) ?
```

Enter

```
Multiple AMBER options for CYS :
  0         Decide every time differently
  1   CYS   Cysteine (SH)
  2   CYM   Deprotonated Cysteine (S-)
  3   CYX   Cystine (S-S bridge)

Suggested option: 0
```

Enter

```
Multiple AMBER options for CYS    1 (    1) :
  1   CYS   Cysteine (SH)
  2   CYX   Cystine (S-S bridge)

  Connections and Nearest Atoms for SG CYS    1 SG  ( P2A #    8 PDB#    6 )
      Dist  P2A Nr  PDB Nr  Label                Near     Dist  P2A Nr  PDB Nr  Label
 1    1.83       5       5  CB  CYS    1  CB       1     5.58      19       0  H1  GLY    2
                                                    2     6.06      36       0  HC  VAL    3
                                                    3     6.09      26       0  H   VAL    3
                                                    4     6.47      25      11  N   VAL    3  N
                                                    5     7.15      35      17  CT  VAL    3  CG2
Suggestion: 1
```

Enter

```
Option  Molecule    Option  Molecule    Option  Molecule    Option  Molecule    Option  Molecule
     1: CYS    1         4: PRO    4         7: GLN    7        10: LEU   10
     2: GLY    2         5: ALA    5         8: PRO    8
     3: VAL    3         6: ILE    6         9: VAL    9

Give option number of molecules to be put in QM region (or 'c' to continue):
Note: by specifying a negative number a molecule is removed from the QM region
```

4 5

```
Give option number of molecules to be put in QM region (or 'c' to continue):
Note: by specifying a negative number a molecule is removed from the QM region
```

c

```
Make a choice for the QM/MM treatment of PRO    4
 0:  Put completely in QM region
 1:  Cut off at C-alpha (put NH in QM region, CO in MM region)
 2:  Cut off at C-alpha (put NH in MM region, CO in QM region)
 3:  Cut off at C-alpha (put NH and CO in MM region)
 4:  Cut off at C-alpha (put NH and CO in QM region,  sidechain in MM region)
 5:  Put only part of sidechain in QM region

Suggestion: 2
...
Give choice:
```

Enter

```
Make a choice for the QM/MM treatment of ALA    5
 0:  Put completely in QM region
 1:  Cut off at C-alpha (put NH in QM region, CO in MM region)
 2:  Cut off at C-alpha (put NH in MM region, CO in QM region)
 3:  Cut off at C-alpha (put NH and CO in MM region)
 4:  Cut off at C-alpha (put NH and CO in QM region,  sidechain in MM region)
 5:  Put only part of sidechain in QM region

Suggestion: 1
Give choice:
```

Enter

```
Do you want to add solvent to your system (Y/n) ?
```

Y

```
Solvent (box) available:
   1:  HOH     HOH  Water molecule
   2:  MOH     MOH  Methanol molecule
   3:  CHL     CHL  Chloroform molecule
```

1

```
Make a choice:
```

1

```
Give boxsize (def.:    16.71 Angs)
```

17.5

QMMM_Butane: Basic QMMM Illustration

Sample directory: adf/QMMM_Butane/

This example is a simple illustration of the QMMM functionality: half of the butane molecule is treated quantum-mechanically, the other half by molecular mechanics.

```
$ADFBIN/adf << eor
Title  BUTANE in Z-matrix input
```

(Omitted in this printout: the usual specifications of fragments, symmetry, integration accuracy, -)

```
QMMM
   FORCEFIELD_FILE  $ADFRESOURCES/ForceFields/amber95.ff
   RESTART_FILE mm.restart
   OUTPUT_LEVEL=2
   WARNING_LEVEL=2
   ELSTAT_COUPLING_MODEL=0

   LINKS
     1 - 4       1.38000      H
   SUBEND

   MM_CONNECTION_TABLE
     1 CT  QM     2   3   4   5
     2 HC  QM   1
     3 HC  QM   1
     4 CT  LI   1   9  13  14
     5 CT  QM   1   6   7   8
     6 HC  QM   5
     7 HC  QM   5
     8 HC  QM   5
     9 CT  MM   4  10  11  12
    10 HC  MM   9
    11 HC  MM   9
    12 HC  MM   9
    13 HC  MM   4
```

```
    14 HC  MM   4
   SUBEND

End

Atoms   Internal
 C   0   0   0            0             0             0
 H   1   0   0           B1             0             0
 H   1   2   0           B2            A1             0
 C   1   2   3           B3            A2            D1
 C   1   2   3           B4            A3            D2
 H   5   1   2           B5            A4            D3
 H   5   1   6           B6            A5            D4
 H   5   1   6           B7            A6            D5
 C   4   1   2           B8            A7            D6
 H   9   4   1           B9            A8            D7
 H   9   4  10          B10            A9            D8
 H   9   4  10          B11           A10            D9
 H   4   1   9          B12           A11           D10
 H   4   1   9          B13           A12           D11
End

GeoVar
....
```

In the QMMM key block, the MM connection table identifies the atoms as belonging to either the QM (quantum mechanics) part, or the MM (molecular mechanics) part, or to the set of LI (link) atoms, which define the connection between the QM and the MM regions. Order and numbering are one-to-one with the list under the Atoms key.

The Link atom, part of the MM section of the system, is associated with a *capping atom*, in the QM part of the system. The Links subkey block specifies for each LI atom defined under the MM_Connection_Table subkey block the chemical type of the replacing capping atom (here: H). On the same line we find the ratio of the QM atom LI atom distance to the QM atom capping atom distance (here: 1.38), and the numbers (1 and 4) of the involved QM atom and LI atom.

The other subkeys in the QM key block are simple subkeys. The specify the file with the force field parameters to be used in the MM subsystem, the (restart) file to write MM data to, print and warning levels and a code for the electrostatic coupling model to use. See the QMMM manual for a detailed discussion of all options.

The calculation is a simple geometry optimization (the Geometry key is not displayed here, but is contained in the full input). This consists of a repeated two-step process. At the first step, the MM system is kept frozen, the SCF equations are solved for the QM system, where potentials resulting from the MM system are included, and gradients on the QM atoms are computed from the SCF solution. At the second step, the QM system's geometry is updated and then kept frozen while the MM system's geometry is optimized (converged) for that particular QM configuration. And so on, until the whole combined system is self-consistently converged.

QMMM_CYT

Sample directory: adf/QMMM_CYT/

See the QMMM manual , where this case is used as a 'walk through' for the QMMM feature.

It is a more or less straightforward application of QMMM to geometry optimization (Cytocine). In the Atoms block all atoms are listed (QM as well as MM). All QMMM aspects, such as which atoms belong to the QM core and which are to be treated by the approximate MM method, are found in the QMMM key block, and its various subkey blocks. The remainder of the input file is not different from what it would be in a non-QMMM run.

The standard amber95 force field is used, which is located in the database of the ADF distribution.

```
$ADFBIN/adf << eor
Title  CYT amber95 - Cartesian Geometry Optimization

Fragments
  C   t21.C
  H   t21.H
End

Charge  0  0

Atoms Cartesian
   1 C       1.94807   3.58290  -0.58162
   2 C       1.94191   3.61595   1.09448
   3 H       1.69949   4.49893  -1.05273
   4 H       2.99455   3.17964  -0.86304
   5 C       0.94659   2.40054  -0.92364
   6 N      -1.74397  -3.46417   0.31178
   7 C      -1.00720  -2.20758   0.33536
   8 C      -1.66928  -1.00652   0.31001
   9 C      -0.92847   0.25653   0.34895
  10 N       0.43971   0.26735   0.38232
  11 N       0.36409  -2.20477   0.28992
  12 C       1.09714  -0.95413   0.22469
  13 H      -2.89781  -3.50815   0.31746
  14 H      -1.21484  -4.49217   0.31721
  15 H      -2.80940  -0.93497   0.30550
  16 H      -1.55324   1.21497   0.33885
  17 C       1.23309   1.44017   0.30994
  18 O       2.58277  -1.01636   0.23914
  19 H       2.37276   1.25557   0.29984
  20 O       1.02358   2.43085   1.50880
  21 H       1.17136   1.95097  -1.87367
  22 H      -0.10600   2.77333  -0.80348
  23 H       1.62170   4.54039   1.51392
  24 H       2.99608   3.28749   1.41345
End

QMMM
   FORCEFIELD_FILE  $ADFRESOURCES/ForceFields/amber95.ff
   RESTART_FILE mm.restart
   OUTPUT_LEVEL=1
   WARNING_LEVEL=2
   ELSTAT_COUPLING_MODEL=1
```

```
  LINK_BONDS
    1 - 5       1.38000      H
    1 - 2       1.38030      H
  SUBEND

  MM_CONNECTION_TABLE
    1    CT    QM     2   3   4   5
    2    CT    LI     1  20  23  24
    3    HC    QM     1
    4    HC    QM     1
    5    CT    LI     1  17  21  22
    6    N2    MM     7  13  14
    7    CA    MM     6   8  11
    8    CM    MM     7   9  15
    9    CM    MM     8  10  16
   10    N*    MM     9  12  17
   11    NC    MM     7  12
   12    C     MM    10  11  18
   13    H     MM     6
   14    H     MM     6
   15    HA    MM     8
   16    H4    MM     9
   17    CT    MM     5  10  19  20
   18    O     MM    12
   19    H2    MM    17
   20    OS    MM     2  17
   21    HC    MM     5
   22    HC    MM     5
   23    H1    MM     2
   24    H1    MM     2
SUBEND

CHARGES
  1  0.0  CT
  2  0.0  CT
  3  0.0  HC
  4  0.0  HC
  5  0.0  CT
  6 -0.9530  N2
  7  0.8185  CA
  8 -0.5215  CM
  9  0.0053  CM
 10 -0.0484  N*
 11 -0.7584  NC
 12  0.7538  C
 13  0.4234  H
 14  0.4234  H
 15  0.1928  HA
 16  0.1958  H4
 17  0.0066  CT
 18 -0.6252  O
 19  0.2902  H2
 20 -0.2033  OS
 21  0.0000  HC
```

```
  22  0.0000  HC
  23  0.0000  H1
  24  0.0000  H1
 SUBEND

END

Geometry
  Iterations   20
  Converge   E=1.0E-3  Grad=0.0005
  Step       Rad=0.3   Angle=5.0
End

XC
  LDA  VWN
  GGA  PostSCF Becke Perdew
End

Integration  3.0

SCF
  Iterations  60
  Converge  1.0E-06  1.0E-6
  Mixing  0.20
  DIIS  N=10  OK=0.500  CX=5.00  CXX=25.00  BFAC=0.00
End

End Input
eor
```

QMMM_Surface: Ziegler-Natta catalysis

Sample directory: adf/QMMM_Surface/

This is an example of a Ziegler-Natta type catalytic system: a TiCl complex embedded in a MgCl surface with two organic substrates also attached to the surface. To make the computation faster, the QMMM approach is applied. The QM part includes only the active site and a piece of the MgCl surface.

The computation is formally a geometry optimization, but to keep the sample doable in a reasonable time the sample performs only one geometry update step. In the optimization, all of the MgCl surface atoms are frozen.

The standard force field has been modified to accommodate this calculation. The modified force field file is part of the sample run script. In this modified file, bonds are defined between Mg-Cl atoms in the MM connection table. This results in some torsions where the atoms are collinear. To rectify this problem, the torsional potentials for these atoms are set to potential type '0' (no potential).

There are no capping atoms mediating the bonds between the QM and MM regions because the boundary goes through the MgCl surface, which is ionically bound.

```
cat << eor > champ_de_force.ff
YBYL/TRIPOS FORCE FIELD FILE FOR ADF QM/MM
```

```
MODIFIED WITH UFF1.01 FOR Si Mg Ti Cl
L. Petitjean 15.11.1999
**************************************************************************
```

(Most of the contents of the modified force field file is omitted here. You quickly get the difference with the standard sybyl force field file in the ADF database by running a UNIX *diff* on the two files.

```
====================================
eor


$ADFBIN/adf  << eor
Title  ADF-QMMM in a surface study
NoPrint SFO, Frag, Functions

! keywords for calculation methods and optimization
XC
 GGA    BLYP
End

Geometry
 Optim          Cartesian Selected
 Iterations     1
 HessUpd        BFGS
 Converge       e=1e-4 grad=1e-3 rad=1e-2
 Step           rad=0.15
END
```

The 'Iterations 1' subkey specification in the Geometry block specifies that only one step in the optimization is carried out.

```

Integration   3.0 3.0

SCF
 Iterations  250
 Converge    1E-6 1E-6
 Mixing      0.2
 DIIS        N=10 OK=0.5 cyc=5 CX=5.0 BFAC=0
End


! keywords for molecule specification
Charge 0 0

Atoms Cartesian
    1 Mg    x1   y1   z1
```

(all other atoms in the Atoms block omitted here)

```

End
```

```
GeoVar
   x1=.00000 F
   y1=.00000 F
   z1=.00000 F
   x2=.00000 F
   y2=1.72129 F
   z2=1.82068 F
   x3=.00000 F
   y3=.00000 F
   z3=-3.64100 F
   x4=.00000 F
   y4=-1.72130 F
   z4=-1.82068 F
   x5=.00000 F
   y5=1.72130 F
   z5=-1.82032 F
   x6=.00000 F
   y6=1.72130 F
   z6=-5.46132 F
   x7=2.53903
   y7=.03004
   z7=-3.50645
   x8=2.50628
   y8=-.07048
   z8=-.10022
   x9=2.63009
   y9=3.50093
   z9=-3.02634
...
```

Many of the coordinates have a 'F' after their initial value specification under Geovar, indicating that these coordinates will be kept frozen during optimization.

The remaining initial value specifications are omitted here.

```
END
QMMM
   OPTIMIZE
     MAX_STEPS 3000
     MAX_GRADIENT 0.01
     METHOD BFGS
     PRINT_CYCLES 100
   SUBEND

   FORCE_FIELD_FILE champ_de_force.ff
```

The local file 'champ_de_force.ff' is used as force field file. Of course, this is the file we've just set up in the run script.

```
   OUTPUT_LEVEL=1
   WARNING_LEVEL=1
   ELSTAT_COUPLING_MODEL=1
```

```
   MM_CONNECTION_TABLE
       1   Mg  QM    2    4    5    8   58   60
...
```

Contents of the MM_Connection_Table block is omitted.

```
   SUBEND
   CHARGES
     1    .957
     2    -.608
     3    1.017
     4    -.411
     5    -.561
...
```

Initial charges are specified for (all) the atoms. Whether or not the charges on the QM (and LI) atoms are used depends on the type of electrostatic coupling between the QM and MM system. See the QMMM manual for details.

```
   SUBEND
END

Fragments
 Ti t21.Ti
 Cl t21.Cl
 Mg t21.Mg
 C  t21.C
 H  t21.H
End


End Input
eor
```

Structure and Reactivity

Geometry Optimizations

H_2O: Geometry Optimization

Sample directory: adf/GO_H2O/

Summary:

- geometry optimization in delocalized coordinates
- geometry optimization in internal coordinates
- start-up Hessian defined in the input file
- geometry optimization in Cartesian coordinates

Geometry optimization of the water molecule, using the (default) *local* density functional approximation (LDA)

Fair quality basis set: triple zeta with polarization. Four equivalent computations are carried out. The first optimization is done in delocalized coordinates, which requires that atomic coordinates in the input are Cartesian. In the three other optimizations the atomic coordinates are input in Z-matrix format. The optimization is carried out by optimization of the *internal* coordinates in the second and third calculations, and by optimizing the *Cartesian* coordinates in the fourth one. In calculation #3 the start-up Hessian is defined in the input file; in #1,2, and 4 the default start-up Hessian (from a force-field approximation) is applied.

As expected all final results are the same, within the range that might be expected from the convergence thresholds (here: the default values).

The '-n' flag, with value one (1) in the commands $ADFBIN/adf is used to control parallelization. 'adf' and other program names that you may find in $ADFBIN are not the executables themselves but (UNIX) scripts to control running the corresponding programs. If a parallel version has been installed and the machine configuration is right, you can carry calculations out in parallel by supplying a suitable value to the -n flag. Omitting the -n flag invokes the default value, which is given by the environment variable $NSCM. Finally, depending on the type of parallel platform, a file $ADFBIN/nodeinfo may define an upper bound on the parallel execution. See the Installation manual for details on the parallel installation.

In all subsequent examples, the set-shell and remove-file commands will be omitted, as well as any -n flags. Also any inputs for create runs will not be shown in other examples except when a special feature is involved or when it may help to clarify the example at hand.

Optimization in delocalized coordinates

```
$ADFBIN/adf <<eor
Title WATER Geometry Optimization with Delocalized Coordinates

Atoms
    O             0.000000     0.000000     0.000000
    H             0.000000    -0.689440    -0.578509
    H             0.000000     0.689440    -0.578509
```

```
End

Basis
 Type TZP
 Core Small
End

Geometry
  Optim Deloc
End

End Input
eor
```

A title is supplied. This title is printed in the output header. It is also written to any result files from the calculation and will be printed out when such a file is attached to another calculation, for instance as a fragment file. In addition, adf constructs a 'jobidentification' string that contains the adf release number and the date and time. The jobidentification is also printed in the output header and dumped on any result files.

The atomic positions are given with the key atoms. The Cartesian atomic coordinates are in Angstrom. The structure used here does not necessary imply that the two HO bonds must remain equal in the optimization. The symmetry will keep them equal.

The key geometry *must* be supplied to let the program do an optimization: otherwise a single point calculation would be carried out. The geometry data block is empty here, meaning that no default values are reset for the options that are controlled with this key.

No symmetry is specified by a Schönfliess type symbol (key symmetry). The program will use the true symmetry of the nuclear frame (accounting for any fields, if present). In this case that is C(2v). If such symmetry would not be acceptable for adf (not all pointgroups are supported!) or when you want to run in a lower symmetry, the symmetry to be used must be specified.

The fragment files are defined implicitly with the Basis keyword. In this case (as well as in most other samples) the fragment files reside in the local directory since they were created there in the same job. If they would have been located elsewhere you could specify a full path for each of the files, or alternatively (if all fragmentfiles are in one single directory) write the directory after the keyword fragments (on the same line).

The precision of numerical integration, to evaluate Hamiltonian matrix elements etc., is not specified and attains therefore the default value (4.0 in an optimization run).

Z-matrix Optimization

```
$ADFBIN/adf <<eor
Title WATER Geometry Optimization with Internal Coordinates

Atoms    Z-Matrix
 1. O   0 0 0
 2. H   1 0 0   rOH
 3. H   1 2 0   rOH  theta
End

Basis
 Type TZP
 Core Small
```

```
End

GeoVar
 rOH=0.9
 theta=100
End

Geometry
End

End Input
eor
```

The atomic positions are given with the key atoms. Bond lengths are in Angstrom and angles in degrees. The key geometry assigns numerical starting values to the two variables. We could also have written numerical values directly in the atoms block. The structure used here implies that the two HO bonds are equal and must remain equal: they are associated with the same variable; this constraint would not have applied if numerical data had been put in the atoms section, although the symmetry would have kept them equal anyway.

Definition of (diagonal) start-up Hessian

```
$ADFBIN/adf <<eor
Title WATER   optimization with (partial) specification of Hessian

Atoms    Z-Matrix
 1. O   0 0 0
 2. H   1 0 0   rOH
 3. H   1 2 0   rOH  theta
End

GeoVar
 rOH=0.9
 theta=100
End

HessDiag  rad=1.0  ang=0.1

Fragments
 H   t21.H
 O   t21.O
End

Geometry
End

End Input
eor
```

All input is identical to the previous case, except for the key HessDiag. This defines here the start-up Hessian to be diagonal with values 1.0 and 0.1 for the entries related to bondlengths and angles respectively.

Optimization in Cartesian coordinates

```
$ADFBIN/adf <<eor
Title WATER Geometry Optimization in Cartesians

Geometry
  Optim Cartesian
End

Define
 rOH=0.9
 theta=100
End

Atoms    Z-Matrix
 1. O   0 0 0
 2. H   1 0 0   rOH
 3. H   1 2 0   rOH theta
End

Fragments
 H   t21.H
 O   t21.O
End

End Input
eor
```

In the last calculation the atomic coordinates are input in the same way as before, but the geometry block now specifies, with the subkey *optim*, that the *cartesian* coordinates are to be varied and monitored for convergence.

If different coordinates are specified in the *optim* instruction than were used for the input in the atoms block, no constraints can be used. The variable 'rOH' cannot be placed in the geovar block therefore, since that would imply a constraint: keep the two OH distances equal.

The placement of rOH (and theta) in the define block has a completely different meaning. define merely associates a numerical value with an identifier. Wherever the identifier occurs in input (not only in the gatoms block) it will be replaced by the numerical value. This means that there are now nine (9) variables: the x,y,z coordinates of the three atoms.

Pure translations and rotations will be filtered out by the program and the symmetry (explicitly specified or internally computed), C(2v) here, will be enfored on all developments so that the situation is equivalent to the previous calculation as regards the degrees of freedom of the system.

Remark: the define block must occur in input *before* the variables defined in it are used. This is one of the few cases where the relative position of keys in the input stream is relevant. The same does not hold for the geovar key used in the earlier example: geovar may be placed anywhere in the input, irrespective of the locations of atoms.

Formaldehyde: another Optimization

Sample directory: adf/GO_Formaldehyde/

In the input for the optimization run the atomic coordinates are in Z-matrix format while the optimization variables are the *Cartesian* coordinates. This is achieved with the *optim* subkey in the geometry block.

A single geovar variable is used for different coordinates. However, since the type of optimization variables (Cartesian) is not the same as the type of input coordinates (Z-matrix), no constraints are implied by this. In fact, the related coordinates do remain equal, but this is because they are symmetry related and the program preserves symmetry anyway.

NonLocal gradient corrections (gga: Generalized Gradient Approximation) according to the approach known as 'Becke' (for exchange) and 'Perdew' (correlation) are included self-consistently with the key xc.

```
$ADFBIN/adf << eor
Title  formaldehyde

Geometry
 Optim  cartes
End

XC
  GGA Becke Perdew
END

Symmetry  C(2v)

Atoms  Z-matrix
  1 O   0 0 0  0.0  0.0  0.0
  2 C   1 0 0  r2   0.0  0.0
  3 H   2 1 0  r3   a3   0.0
  4 H   2 1 3  r3   a3   t4
End

Fragments
  C  t21.C
  O  t21.O
  H  t21.H
End

Geovar
 r2     1.94
 r3     0.95
 a3   120
 t4  -180
End

integration 4.5

End Input
eor
```

Aspirin: an optimization in delocalized coordinates

Sample directory: adf/DelocalGO_aspirin/

Geometry optimization of the aspirin molecule, using the delocalized coordinates.

```
$ADFBIN/adf <<eor
Title geometry optimization of aspirin in delocalized coordinates

Basis
Type DZ
End

GEOMETRY
  OPTIM DELOCAL
END

ATOMS
    C         0.000000  0.000000  0.000000
    C         1.402231  0.000000  0.000000
    C         2.091015  1.220378  0.000000
    C         1.373539  2.425321  0.004387
    C        -0.034554  2.451759  0.016301
    C        -0.711248  1.213529  0.005497
    O        -0.709522  3.637718  0.019949
    C        -2.141910  1.166077 -0.004384
    O        -2.727881  2.161939 -0.690916
    C        -0.730162  4.530447  1.037168
    C        -0.066705  4.031914  2.307663
    H        -0.531323 -0.967191 -0.007490
    H         1.959047 -0.952181 -0.004252
    H         3.194073  1.231720 -0.005862
    H         1.933090  3.376356 -0.002746
    O        -2.795018  0.309504  0.548870
    H        -2.174822  2.832497 -1.125018
    O        -1.263773  5.613383  0.944221
    H        -0.337334  4.693941  3.161150
    H         1.041646  4.053111  2.214199
    H        -0.405932  3.005321  2.572927
END

END INPUT
```

AuH: Scalar-Relativistic Optimization

Sample directory: adf/RelGO_AuH/

A simple geometry optimization using the scalar relativistic option, implying that relativistic core potentials must be generated first (*dirac*).

```
$ADFBIN/dirac -n1 < $ADFRESOURCES/Dirac/H
$ADFBIN/dirac -n1 < $ADFRESOURCES/Dirac/Au.4d

mv TAPE12 t12.rel
```

The optimization run is now straightforward (after having created the relativistic basic atoms, not shown here).

```
$ADFBIN/adf << eor
title  AuH  relativistic optimization

integration  5.5

atoms  Zmat
Au  0 0 0
H   1 0 0  1.5
end

fragments
Au   t21.Au
H    t21.H
end

xc
  GGA Becke Perdew
end

relativistic
CorePotentials  t12.rel &
H 1
Au 2
end

geometry
convergence grad=1e-4
end

end input
eor
```

The key COREPOTENTIALS is used as block key *and* it has an argument ('t12.rel'). Consequently the continuation character (&) is used. Note that the *order* of DIRAC runs, to create the relativistic corepotentials file TAPE12, determines that in the key block to the CorePotentials key, the H atom must relate to the first section on TAPE12 and the Gold atom to the second section.

H_2O: restraint Geometry Optimization

Sample directory: adf/GO_restraint/

The restraint does not have to be satisfied at the start of the geometry optimization. An extra force is added to restrain the bond length, angle, or dihedral angle to a certain value.

Example for angle restraint

```
$ADFBIN/adf << eor
title WATER geometry optimization with angle restraint

ATOMS
    1.O         0.001356    0.000999    0.000000
    2.H         0.994442   -0.037855    0.000000
    3.H        -0.298554    0.948531    0.000000
END

BASIS
 Type DZP
END

INTEGRATION 4 4

RESTRAINT
  ANGLE  3 1 2 125.0
END

GEOMETRY
END

endinput
eor
```

Example for bond length restraint

```
$ADFBIN/adf << eor
title WATER Geometry Optimization with bond length restraint

ATOMS
    1.O         0.001356    0.000999    0.000000
    2.H         0.994442   -0.037855    0.000000
    3.H        -0.298554    0.948531    0.000000
END

BASIS
 Type DZP
END

INTEGRATION 4 4

RESTRAINT
  DIST  1 2 1.03
  DIST  1 3 1.03
END

GEOMETRY
END

endinput
eor
```

Example for dihedral angle restraint

```
$ADFBIN/adf << eor
Title Restraining dihedral of ethane

SYMMETRY NOSYM

ATOMS
  1.C        -0.004115   -0.000021    0.000023
  2.C         1.535711    0.000022    0.000008
  3.H        -0.399693    1.027812   -0.000082
  4.H        -0.399745   -0.513934    0.890139
  5.H        -0.399612   -0.513952   -0.890156
  6.H         1.931188    0.514066    0.890140
  7.H         1.931432    0.513819   -0.890121
  8.H         1.931281   -1.027824    0.000244
END

INTEGRATION 4 4

RESTRAINT
  DIHED   6    2    1    3     20.00
END

BASIS
  type DZP
END

GEOMETRY
END

endinput
eor
```

H_2O: constraint Geometry Optimization

Sample directory: adf/GO_constraints/

The key CONSTRAINTS can only be used in case of the New branch for optimization of coordinates. The input for this key is very similar to that of the RESTRAINT keyword. The key CONSTRAINTS can, however, also be used to constrain Cartesian coordinates. Note that the key RESTRAINT and freezing of coordinates with the GEOVAR key can also be used in the New branch for optimization of coordinates. In ADF2007 the New branch for optimization can only be used in geometry optimizations and transition state searches. Note that the before ADF2008.01 the key CONSTRAINTS was called NEWCONSTRAINTS.

The constraints do not have to be satisfied at the start of the geometry optimization.

Example for angle restraint

```
$ADFBIN/adf << eor
title WATER geometry optimization with angle constraint
ATOMS
```

```
    1.O         0.001356    0.000999    0.000000
    2.H         0.994442   -0.037855    0.000000
    3.H        -0.298554    0.948531    0.000000
END
BASIS
 Type DZP
END
INTEGRATION 4 4
CONSTRAINTS
  ANGLE  3 1 2 125.0
END
GEOMETRY
  OPTIM DELOCAL
END
endinput
eor
```

Example for fixed-atom constraint. Note that the optimization should be done in Cartesian.

```
$ADFBIN/adf << eor
title WATER geometry optimization with fixed-atom constraint
ATOMS
    1.O         0.001356    0.000999    0.000000
    2.H         0.994442   -0.037855    0.000000
    3.H        -0.298554    0.948531    0.000000
END
BASIS
 Type DZP
END
INTEGRATION 4 4
SYMMETRY NOSYM
CONSTRAINTS
  ATOM 1 0.0 0.0 0.0
  ATOM 2 1.0 0.0 0.0
END
GEOMETRY
  OPTIM CARTESIAN
  BRANCH NEW
END
endinput
eor
```

Example for bond length restraint.

```
$ADFBIN/adf << eor
title WATER Geometry Optimization with bond length constraint
ATOMS
    1.O         0.001356    0.000999    0.000000
    2.H         0.994442   -0.037855    0.000000
    3.H        -0.298554    0.948531    0.000000
END
BASIS
 Type DZP
END
```

```
INTEGRATION 4 4
CONSTRAINTS
  DIST  1 2 1.03
  DIST  1 3 1.03
END
GEOMETRY
  OPTIM CARTESIAN
  BRANCH NEW
END
endinput
eor
```

Example for dihedral angle restraint

```
$ADFBIN/adf << eor
Title Constraining dihedral of ethane
SYMMETRY NOSYM
ATOMS
  1.C        -0.004115   -0.000021    0.000023
  2.C         1.535711    0.000022    0.000008
  3.H        -0.399693    1.027812   -0.000082
  4.H        -0.399745   -0.513934    0.890139
  5.H        -0.399612   -0.513952   -0.890156
  6.H         1.931188    0.514066    0.890140
  7.H         1.931432    0.513819   -0.890121
  8.H         1.931281   -1.027824    0.000244
END
INTEGRATION 4 4
CONSTRAINTS
  DIHED   6    2    1    3     20.00
END
BASIS
  type DZP
END
GEOMETRY
  OPTIM DELOCAL
END
endinput
eor
```

Example for Block constraint (with a dihedral constraint).

```
$ADFBIN/adf << eor
Title Block constraints (with a dihedral constraint)
SYMMETRY NOSYM
ATOMS
  1.C        -0.004115   -0.000021    0.000023 b=b1
  2.C         1.535711    0.000022    0.000008 b=b2
  3.H        -0.399693    1.027812   -0.000082 b=b1
  4.H        -0.399745   -0.513934    0.890139 b=b1
  5.H        -0.399612   -0.513952   -0.890156 b=b1
  6.H         1.931188    0.514066    0.890140 b=b2
  7.H         1.931432    0.513819   -0.890121 b=b2
  8.H         1.931281   -1.027824    0.000244 b=b2
```

```
END
INTEGRATION 4 4
CONSTRAINTS
  DIHED   6    2    1    3     20.00
  BLOCK b1
  BLOCK b2
END
BASIS
  type DZP
END
GEOMETRY
  OPTIM DELOCAL
END
endinput
eor
```

LiF: optimization with an external electric field or point charges

Sample directory: adf/GO_LiF_Efield/

In the first example a geometry optimization is performed with an external homogeneous electric field. In the second example a geometry optimization is performed with an external point charges

Note that SYMMETRY NOSYM should be used. In case of point charges it is important to use the QPNEAR subkeyword of the INTEGRATION key with a large enough value that would include some of the point charges.

```
$ADFBIN/adf << eor
Title LiF Cartesian Geometry Optimization in the presence of electric field
Symmetry NOSYM
Atoms
    F             0.000000     0.800000     0.000000
    Li            0.000000    -0.800000     0.000000
End
Basis
 Type TZP
 Core Small
End
Geometry
  Optim Cartesian
  Branch New
  Converge 0.0000001
  Iterations 100
End
Efield 0.0 0.0 0.01

End Input
eor
$ADFBIN/adf << eor
Title LiF Cartesian Geometry Optimization in the presence of point charges
Symmetry NOSYM
```

```
Atoms
    F             0.000000     0.800000     0.000000
    Li            0.000000    -0.800000     0.000000
End
Basis
 Type TZP
 Core Small
End
Geometry
  Optim Cartesian
  Branch New
  Converge 0.001
  Iterations 100
End
Efield &
0.0 0.0  5.3  0.5
0.0 0.0 -5.3 -0.5
End
integration
   qpnear 20
end
End Input
eor
```

Transition States, Linear Transits, Intrinsic Reaction Coordinates

HCN: LT, Frequencies, TS, and IRC

Sample directory: adf/HCN/

Summary

- For a sequence of intermediates, each defined by a fixed angle H-C-N between the linear extremes
  HCN and CNH, the remaining geometrical parameters are optimized, giving a Linear Transit point-by-point scan of the energy curve of the Hydrogen atom travelling from one end of the CN fragment to the other. This is a useful way to get a reasonable first guess of the Transition State.
- At the approximate TS a Frequencies calculation is performed to obtain a fairly accurate Hessian for the next calculation.
- A TS search is carried out, using the computed Hessian. As variation, the TS search is repeated, first
  with the automatic (internal) Hessian (based on force fields) and then also with a constraint applied.
- A full IRC scan of the full path, starting from the TS, down to the two minima.

LT

The first calculation is a Linear Transit where the Hydrogen atom moves from one side of CN to the other by a parameterized step-by-step change of the angle H-C-N. The other coordinates of the system are optimized along the path.

In the atoms block, one coordinate value is represented by an identifier (th). In the geovar block this is asssigned two values, implying that it is a Linear Transit parameter. The initial and final values for the parameter are given.

Since the geometry block does not have OPTIM SELECTED, all other coordinates are optimized for each of the 10 Linear Transit points.

The subkey *iterations* in the geometry block carries *two* arguments: the first is the maximum number of optimization steps (per LT point). The second is the number of LT points to compute in this run: 4. This implies that only a part of the 10-point path defined by the LT parameter(s) will be scanned. The remainder will be done in a follow-up run to illustrate usage of the restart facility.

```
$ADFBIN/adf <<eor
Title    HCN Linear Transit, first part
NoPrint  SFO, Frag, Functions, Computation

Atoms      Internal
  1 C  0 0 0       0    0    0
  2 N  1 0 0       1.3  0    0
  3 H  1 2 0       1.0  th   0
End

Basis
 Type DZP
End

Symmetry NOSYM

Integration 6.0 6.0

Geometry
  Branch Old
  LinearTransit  10
  Iterations     30  4
  Converge   Grad=3e-2,  Rad=3e-2,  Angle=2
END

Geovar
  th   180    0
End

End Input
eor

mv TAPE21 t21.LT
rm logfile
```

The NoPRINT key turns off a lot of default output. There are several PRINT and NOPRINT options; see the User's Guides for details.

Since the geometry changes from linear to planar (and finally back to linear again), the symmetry must be given explicitly in the input file. Otherwise the program would find a C(lin) symmetry for the initial geometry and assume that this symmetry is preserved throughout. This would of course result in an error abort when the first LT step is carried out, breaking the linear symmetry.

The here specified symmetry (NOSYM: no symmetry at all) is not the true symmetry of the complete path C(s) but a subgroup. It is always allowed to specify a lower symmetry than the actually present symmetry. Such may be necessary (for instance when the true symmetry cannot be handled by adf) or in special cases required for reasons of analysis. Generally speaking, however, we recommend to use the highest symmetry possible (given the case at hand and taking into account the symmetries recognizable by ADF) to boost performance.

Convergence thresholds in the geometry block are set less tight than the defaults: we need only a reasonable estimate of the path, but no highly converged geometries.

At the end of the run the tape21 result file is saved and renamed *t21.LT* to serve as restart file for the follow-up calculation.

LT continuation

```
$ADFBIN/adf <<eor
Title    HCN Linear Transit
NoPrint  SFO,Frag,Functions,Computation

Restart   t21.LT

Fragments
  N   t21.N
  C   t21.C
  H   t21.H
End

Atoms       Internal
  1 C  0 0 0       0    0    0
  2 N  1 0 0       1.3  0    0
  3 H  1 2 0       1.0  th   0
End

symmetry NOSYM

Integration 6.0 6.0

Geometry
  Branch Old
  LinearTransit   10
  Converge        Grad=3e-2, Rad=3e-2, Angle=2
END

Geovar
  th   180    0
```

```
End

End Input
eor

rm TAPE21 logfile
```

From the restart file, supplied with the key restart, the program reads off that the first 4 points of the LT path have been done already and the scan is continued with LT point #5. The same path definition is supplied again, including the *original* starting values for the coordinates. The actual starting coordinates (for LT point #5) are read from the restart file. The input values, however, serve to define and verify consistency of the defined LT path and must therefore be supplied correctly.

The key noprint is used to suppress major parts of standard output: all information pertaining to the sfo analysis, all build-from-fragments information, and the lists of elementary functions in the basis sets and fit sets.

Frequencies at the estimated Transition State

From the results of the Linear Transit run we can sketch the energy barrier that H passes over when going from one side of the molecule to the other. This yields a reasonable guess for the Transition State.

To check that the so-obtained estimate is adequate we compute the frequencies in that geometry: one of them should be imaginary.

Apart from serving as a check that the TS estimate is not too bad, the computed Hessian will also serve in the follow-up calculation to obtain the true TS.

```
$ADFBIN/adf <<eor
Title    HCN Frequencies in LT max (approx), moderate precision
NoPrint  SFO,Frag,Functions,Computation

Integration 6.0 6.0

Fragments
  N   t21.N
  C   t21.C
  H   t21.H
End

Atoms      Internal
  1 C  0 0 0      0       0   0
  2 N  1 0 0      1.186   0   0
  3 H  1 2 0      1.223  70   0
End

Geometry
  Frequencies
End

End Input
eor
```

```
mv TAPE21 t21.Freq
```

Inspection of the output file shows that one of the frequencies is imaginary, as expected (printed as negative), signalling the proximity of the Transition State.

The TAPE21 result file of the calculation is renamed and saved. Later we will use it as a 'restart' file for a TS search, namely to supply the computed Hessian as the initial 'guess' of the Hessian in the (TS) optimization run.

TS search

Now carry out the Transition State search, starting from the lt-derived guess.

In this first attempt to find the TS, no use is made of the tape21 result file from the Frequencies run. That will be done in the next calculation.

```
$ADFBIN/adf <<eor
Title    HCN Transition State, automatic initial Hessian
NoPrint  SFO,Frag,Functions,Computation

Integration 6.0 6.0

Atoms      Internal
  1 C  0 0 0      0       0   0
  2 N  1 0 0      1.186   0   0
  3 H  1 2 0      1.223  70   0
End

Fragments
  N   t21.N
  C   t21.C
  H   t21.H
End

Geometry
  TransitionState
End

End Input
eor

rm TAPE21 logfile
```

The TS-search run type is specified in the geometryblock.

No symmetry is specified; the program determines the symmetry to be C(s) and consequently carries out the ts search in that symmetry.

TS search, using the Hessian from the Frequencies run

```
$ADFBIN/adf <<eor
Title    HCN Transition State,  initial Hessian from Freq run
```

```
NoPrint  SFO,Frag,Functions,Computation

Restart  t21.Freq
Save     TAPE13

Integration  6.0 6.0

Atoms          Internal
  1 C  0 0 0      0       0   0
  2 N  1 0 0      1.186   0   0
  3 H  1 2 0      1.223  70   0
End

Fragments
  N   t21.N
  C   t21.C
  H   t21.H
End

Geometry
  TransitionState
End

End Input
eor

mv TAPE13  t13.TS
rm TAPE21 logfile
```

The CheckPoint file TAPE13, at normal termination automatically deleted by the program, is here saved, using the SAVE key. TAPE13 is as good a restart file as TAPE21 is, but it is a lot smaller. TAPE21 contains a large amount of information for analysis purposes, while TAPE13 contains essentially *only* restart-type data.

The input is identical to the previous one, except for the restart file. This is used here to provide the Hessian computed in the Frequencies run as the start-up Hessian for the ts optimization. At the same time the atomic coordinates are read off from the restart file and override the values in the input file. This latter aspect could have been suppressed; see the User's Guide for using the restart key.

Constrained TS search

Finally the ts search where one coordinate is kept frozen, to illustrate a constrained optimization.

```
$ADFBIN/adf <<eor
Title    HCN  constrained TS search
NoPrint  SFO,Frag,Functions,Computation

Restart t21.Freq

Integration  6.0 6.0

Atoms     Internal
  1 C  0 0 0      0       0   0
```

```
  2 N  1 0 0      rNC     0   0
  3 H  1 2 0      1.223  70   0
End

GeoVar
  rNC=1.186 F
End

Fragments
  N   t21.N
  C   t21.C
  H   t21.H
End

Geometry
  TransitionState
End

End Input
eor

rm TAPE21 logfile
rm t21.Freq
```

The geovar key specifies that the nc distance, rNC has the initial value 1.15 and remains frozen ('F').

The fact that the optimization is now carried out in a different subspace of atomic coordinates does not prevent us from using the *t21.Freq* restart file to supply the initial Hessian.

IRC scan of the reaction path

The IRC calculation is split in three steps, to illustrate the Restart facility applied to the IRC functionality.

In the first only a few points are computed, along one of the two paths leading from the TS to the adjacent minima. Since no explicit directives are given in the input to specify the *direction* of the first path, the so-called 'forward' path is taken. The definition of which is 'forward' and which is 'backward' is in fact quite arbitrary and is determined by the program. See the User's Guide for details.

The saved TAPE13 file from one of the TS calculations is used as restart file. This provides (a) the optimized coordinates of the TS as starting point, (b) the initial Hessian to guide the point-by-point optimizations along the IRC path, and (c) the eigenvector of the lowest Hessian eigenvalue to define the initial direction of the IRC path.

The TAPE13 file from this partial IRC scan is saved to serve as start-up file for the next calculations, which will continue the IRC scan.

In the Geometry key block, the run type is set to IRC and the 'Points' option is used to limit the number of IRC points to compute.

```
$ADFBIN/adf << eor
Title   HCN  IRC partial path (forward)
NoPrint SFO,Frag,Functions, Computation
```

```
Integration 6.0 6.0

Restart  t13.TS
Save     TAPE13

Atoms          Internal
  1 C  0 0 0      0       0   0
  2 N  1 0 0      1.186   0   0
  3 H  1 2 0      1.223  70   0
End

Fragments
  N   t21.N
  C   t21.C
  H   t21.H
End

Geometry
  IRC  Points=5
End

End Input
eor

mv TAPE13  t13.IRC_1
rm TAPE21 logfile
```

The IRC is continued in the next calculation, using the TAPE13 file from the previous one as restart file. From this file, the program reads the IRC path information computed sofar. By default, it would continue on the 'forward' path, since that was not yet finished. However, in the Geometry key block, we now specify not only that a limited number of points is to be computed in this run (5 again), but we instruct the program also to compute only points on the 'backward' path.

```
$ADFBIN/adf << eor
Title   HCN  IRC partial part (backward)
NoPrint SFO,Frag,Functions, Computation

Restart  t13.IRC_1
Save     TAPE13

Integration 6.0 6.0

Atoms          Internal
  1 C  0 0 0      0       0   0
  2 N  1 0 0      1.186   0   0
  3 H  1 2 0      1.223  70   0
END

Fragments
  N   t21.N
  C   t21.C
  H   t21.H
End
```

```
Geometry
  IRC  Points=5  Backward
End

End Input
eor

mv TAPE13  t13.IRC_2
rm TAPE21 logfile
```

In the third IRC run, the IRC scan is finished. We start with the TAPE13 file from the previous run and set a maximum of 70 IRC points to compute (which turns out to be sufficient for the complete IRC scan). The program starts on the forward path, continuing where the first (not the previous) had stopped after 5 points, completes the forward path, and then continues on the backward path, starting where the second IRC run had stopped. Both paths are finished and a summary of the path characteristics is printed in the final part of the output.

```
$ADFBIN/adf << eor
Title    HCN  IRC completion
NoPrint  SFO,Frag,Functions, Computation

Restart t13.IRC_2

Integration 6.0 6.0

Atoms          Internal
  1 C  0 0 0      0       0   0
  2 N  1 0 0      1.186   0   0
  3 H  1 2 0      1.223  70   0
End

Fragments
  N   t21.N
  C   t21.C
  H   t21.H
End

Geometry
  IRC  Points=70
End

End Input
eor
```

HCN: transition state search with the CINEB method

Sample directory: adf/HCN_CINEB/

This example demonstrates the use of the Nudged Elastic Band method in ADF for finding a transition state of the HCN isomerisation reaction. A shell script used to run the example calculation is shown below:

```
$ADFBIN/adf <<eor
TITLE Test of the CI-NEB method

SYMMETRY C(S)

NOPRINT SCF SFO
UNITS
    length Angstrom
    angle Degree
END

ATOMS
    1.C         0.000000    0.000000    0.000000
    2.N         XN          0.000000    0.000000
    3.H         XH          YH          0.000000
END

GEOVAR
XN      1.180  1.163
XH      2.196  1.831 1.006 0.105 -0.718 -1.078
YH      0.000  0.799 1.122 1.163  0.813  0.000
END

BASIS
END

GEOMETRY
  CINEB      9
  iterations 150
  OPTIM selected
  converge grad=0.001
  nebspring 1 0.06
END

integration 4.0

SCF
  Convergence 0.00000001
END

eor
```

A few important points to note:

- Symmetry is set to C_s explicitly because all images must have the same symmetry but symmetry of the reaction products is higher, C_∞. Thus, it is necessary to lower the overall symmetry to match the lowest.
- In the GEOVAR section, there are six values specified for the coordinates of the hydrogen atom. Thisi is necessary in order to bring the initial guess for the reaction path closer to the final result. Moreover, if only two values for YH were specified (0.0 and 0.0) then the hydrogen atom would be "dragged" along the C-N bond leading to unrealistic geometries and, eventually, to a failure.
- In the GEOMETRY section of the input, the number of NEB images is set to 9; the convergence criterion is lowered to 0.001; optimization of only the coordinates specified in the GEOVAR block is requested;

- the NEB spring parameter is set to a constant (energy-independent) value of 0.06 a.u.. The choice of the parameter depends on the stiffness of the bonds involved in the reaction and the value of the parameter should, in principle, be of about the same magnitude as the Hessian eigenvalue for the coordinates participating in the reaction.

C2H6 internal rotation: TS search using partial Hessian

Sample directory: adf/TS_C2H6/

Frequently when searching for a transition state, one needs an accurate second derivatives matrix, a Hessian. An exact Hessian may be obtained analytically but this may be very expensive for large molecules. In such cases it may be beneficial to calculate Hessian matrix elements only for atoms directly involved in the reaction for which a transition state is sought for. The rest of the Hessian can then be approximated using a cheaper method.

In this example, a saddle point of the ethane internal rotation around C-C bond is found. In principle, only hydrogen atoms contribute to the normal mode we are interested in. Therefore we calculate a partial Hessian matrix including hydrogen atoms only. For this purpose, the AnalyticalFreq block key is used. In this block, a NUC keyword is added specifying that the second derivatives are calculated for atom 3 (and its symmetry-equivalents) only. Note that the Hessian matrix elements between symmetry-equivalent atoms, for example between 3,H and 4.H are also calculated. The rest of the matrix is estimated using the default method.

```
$ADFBIN/adf <<eor
TITLE Ethane transition state search using partial Hessian

ATOMS
1 C       0.000000000000       0.000000000000       0.767685465031
2 C       0.000000000000       0.000000000000      -0.767685465031
3 H       0.964354016767       0.347635559279       1.177128271450
4 H      -0.181115782790      -1.008972856410       1.177128271450
5 H      -0.783238233981       0.661337297125       1.177128271450
6 H      -0.500471876676       0.894626767091      -1.177128271450
7 H      -0.524533568868      -0.880734742626      -1.177128271450
8 H       1.025005445540      -0.013892024465      -1.177128271450
END

BASIS
type DZ
core Large
END

AnalyticalFreq
 NUC 3
End

INTEGRATION 5.0 5.0 5.0
eor
```

After the Hessian is calculated, the resulting TAPE21 file is used for a subsequent transition state search:

```
mv TAPE21 ethane-frq.t21

$ADFBIN/adf <<eor
TITLE Ethane transition state search using partial Hessian

ATOMS
1 C       0.000000000000       0.000000000000       0.767685465031
2 C       0.000000000000       0.000000000000      -0.767685465031
3 H       0.964354016767       0.347635559279       1.177128271450
4 H      -0.181115782790      -1.008972856410       1.177128271450
5 H      -0.783238233981       0.661337297125       1.177128271450
6 H      -0.500471876676       0.894626767091      -1.177128271450
7 H      -0.524533568868      -0.880734742626      -1.177128271450
8 H       1.025005445540      -0.013892024465      -1.177128271450
END

Fragments
  H t21.H
  C t21.C
END

GEOMETRY
    smooth conservepoints
    TransitionState mode=1
    optim All Cartesian
    iterations 30
    step rad=0.15
    hessupd BOFILL
    converge e=1.0e-4 grad=1.0e-3 rad=1.0e-3
END

RESTART ethane-frq.t21

INTEGRATION 5.0 5.0 5.0
eor
```

Important note: care should be taken to specify correct mode in the TransitionState keyword. Because a significant part of the Hessian will not be calculated exactly, it is possible that it will have more than one negative eigenvalue, in which case the one we are interested in may not be the first one. In such a case, one needs to specify the correct mode number in the TransitionState keyword.

$CH_4+HgCl_2 \Leftrightarrow CH_3HgCl+HCl$: a TS search

Sample directory: adf/RelTS_CH4_HgCl2/

A Transition State calculation, including scalar relativistic terms in the Hamiltonian.

First the relativistic core potentials are generated.

```
$ADFBIN/dirac -n1 < $ADFRESOURCES/Dirac/Hg.4d
$ADFBIN/dirac -n1 < $ADFRESOURCES/Dirac/Cl.2p
```

```
$ADFBIN/dirac -n1 < $ADFRESOURCES/Dirac/C.1s
$ADFBIN/dirac -n1 < $ADFRESOURCES/Dirac/H

mv TAPE12 t12.rel
```

Then the (relativistic) Create runs.

```
$ADFBIN/adf << eor
create Hg $ADFRESOURCES/TZP/Hg.4d
relativistic
corepotentials t12.rel &
Hg 1
end
end input
eor

mv TAPE21 t21.Hg


$ADFBIN/adf << eor
create Cl $ADFRESOURCES/TZP/Cl.2p
relativistic
corepotentials t12.rel &
Cl 2
end
end input
eor

mv TAPE21 t21.Cl

$ADFBIN/adf << eor
create C $ADFRESOURCES/TZP/C.1s
relativistic
corepotentials t12.rel &
C  3
end
end input
eor

mv TAPE21 t21.C


$ADFBIN/adf << eor
create H $ADFRESOURCES/TZP/H
relativistic
corepotentials t12.rel &
H  4
end
end input
eor

mv TAPE21 t21.H
```

In the first Create run (Hg) the CorePotentials key could have been used in its simple form, but in the second (and third and fourth, omitted here) the block form is required to identify the appropriate section on TAPE12 for the atom at hand. In the first case we could have relied on the default: the first section on TAPE12 for the first (in this case: only) atom type.

Note that even for H, which obviously has no frozen core at all, we specify the TAPE12 corepotentials file and indicate the appropriate section for H. The reason is that TAPE12 contains not only the (frozen) *core*, but also the *total* atomic (relativistic) potential.

Finally the TS run.

```
$ADFBIN/adf << eor
TITLE  Transition State: CH4 + HgCl2  ↔  CH3HgCl + HCl

noprint sfo,frag
print atdist

GEOMETRY
  TransitionState
END

relativistic
corepotentials  t12.rel &
C   3
Hg  1
Cl  2
H   4
end

XC
lda  VWN Stoll
END

DEFINE
  rHg = 2.30
  rX1 = 2.35
  rX2 = 2.90
  rH1 = 1.10
  rH2 = 1.10
  rH3 = 1.40

  aX1 = 160
  aX2 = 70
  aH1 = 100
  aH2 = 140
  aH3 = 65

  dH  = 60
END

ATOMS     Z-Matrix
  1.  C     0  0  0   0.        0.        0.
  2.  Hg    1  0  0   rHg       0.        0.
  3.  Cl    2  1  0   rX1      aX1        0.
```

```
  4.  H     1  2  3   rH1      aH1       dH
  5.  H     1  2  3   rH1      aH1      -dH
  6.  H     1  2  3   rH2      aH2       180.
  7.  H     1  2  3   rH3      aH3       180.
  8.  Cl    2  1  3   rX2      aX2       180.
END

FRAGMENTS
 Hg   t21.Hg
 C    t21.C
 H    t21.H
 Cl   t21.Cl
END

endinput
eor
```

For the density-functional the Local Density approximation is used (no GGA corrections), with a correlation correction term due to Stoll (see the User's Guide).

At each geometry cycle the interatomic distance matrix is printed (print atdist).

The initial geometry is a reasonable but not very accuracte estimate of the Transition State. The program needs quite a few cycles to converge, which is rather typical for ts searches: they are a lot more tricky and fail more often than a simple minimization.

H_2O: constraint Linear Transit

Sample directory: adf/LT_constraint/

The LINEARCONSTRAINTS keyword allows geometry optimizations (old branch) with constraints defined by arbitrary linear combinations of (internal) coordinates. The constraint has to be satisfied at the start of the geometry optimization. Note that the before ADF2008.01 the key LINEARCONSTRAINTS was called CONSTRAINT.

Example for bond length constraint, where at the start of the linear transit rOH1=R1=1.0, and rOH2=R2=1.5, such that (-1.0)*R1+(1.0)*R2=0.5, and in the final geometry -R1+R2=0.0 (Reactcoord 0.5 0.0)

```
$ADFBIN/adf << eor
title constraint keyword
XC
  GGA Becke Perdew
END

Geometry
 Branch Old
 LinearTransit  6
End

Integration 5.0

Atoms Internal
```

```
O  0 0 0
H  1 0 0    R1
H  1 2 0    R2  109.9
End

GeoVar
 R1 =1.
 R2 =1.5
End

LinearConstraints
 ReactCoord 0.5 0.0
  R1  -1.0
  R2   1.0
 SubEnd
End

Basis
 Type DZP
End

end input
eor
```

H_2O: (non-)Linear Transit

Sample directory: adf/Transit_H2O/

In ADF2008.01 a transit calculation option has been added in the new optimization branch. This is capable of performing both linear transits, and non-linear transits, and is the default when the LINEARTRANSIT or TRANSIT sub-block is included in the 'Geometry' block.

The new transit code works differently to the old: the transit is represented as a sequence of constrained optimizations. A 'Constraints' block is used to delineate the constraints applied at each stage of the transit.

Non-linear transits are possible, and can even be combined with linear transits in other coordinates. To perform a non-linear transit in a particular coordinate, explicit values must be given.

```
$ADFBIN/adf << eor
Title WATER Transit (non-linear), with the new optimizer branch
Atoms
    O             0.000000     0.000000     0.000000
    H             0.000000    -0.689440    -0.578509
    H             0.000000     0.689440    -0.578509
End
Symmetry NOSYM
Constraints
  dist 1 2 0.8 1.0 1.25 1.5
  angle 2 1 3 start=100.0 end=120.0
End
Basis
 Type SZ
```

```
 Core Large
End
Integration 4.0 2.0
Geometry
  Transit 4
  Optim Deloc
  Converge 0.0001
End
End Input
```

In the example above, 4 values are given for the distance between atoms 1 and 2. This distance constraint will be applied simultaneously with the linear transit constraints for the angle, with other degrees of freedom optimized at each stage of the transit.

Finally, it should be pointed out that 'partial constraints' are used by default in the transit calculations. These constraints are not required to be fully met at each intermediate geometry, but are fully met at the converged geometries. You can use fully converged constraints by supplying the FULLCONVERGE option to the 'Constraints' subblock of the 'Geometry' block (not to be confused with the 'Constraints' block at root level).

Quild

CO: Quild B3LYP geometry optimization

Sample directory: quild/quild_b3lyp_opt/

This example shows how to use the program Quild for a B3LYP geometry optimization, where only the bond energy of the ADF calculation is used.

B3LYP post-SCF

The subkey `NUMGRAD` of the block key `QUILD` is 1, which means that the ADF bond energy is read from the post-SCF METAGGA scheme. The subkey `SMETAGGA` of the block key `QUILD` has as argument a string for the functional (B3LYP(VWN5)), which should be exactly the same as is written on output if the METAGGA keyword is used.

```
$ADFBIN/quild << eor
title Geometry optimization
EPRINT
 SFO NOEIG NOOVL
END
XC
 GGA BLYP
END
ATOMS
O      .000000     .000000     .000000
C      .000000     .000000    1.128100
END
BASIS
 type DZ
 core NONE
```

```
END
GEOMETRY
END
SCF
 diis ok=0.01
 converge 1.0e-5 1.0e-5
END
QUILD
  cvg_grd 1.0e-4
  numgrad 1
  SMETAGGA B3LYP(VWN5)
END
METAGGA
HFEXCHANGE
INTEGRATION 5.0 5.0
endinput
eor
```

B3LYP SCF

In the next calculation the optimization is done with B3LYP as SCF functional. The difference with the previous calculation is is in blue. The subkey `NUMGRAD` of the block key `QUILD` is 2, which means that the standard ADF bond energy is used.

```
$ADFBIN/quild << eor
title Geometry optimization
EPRINT
 SFO NOEIG NOOVL
END
XC
 HYBRID B3LYP
END
ATOMS
O      .000000     .000000     .000000
C      .000000     .000000    1.128100
END
BASIS
 type DZ
 core NONE
END
GEOMETRY
END
SCF
 diis ok=0.01
 converge 1.0e-5 1.0e-5
END
QUILD
  cvg_grd 1.0e-4
  numgrad 2
END
INTEGRATION 5.0 5.0
endinput
eor
```

H_2O dimer: Quild QM/MM geometry optimization

Sample directory: quild/quild_qmmm_water2/

This example shows how to use the program Quild for a QM/MM geometry optimization of the water dimer.

```
$ADFBIN/quild << eor
TITLE QM/MM calculation setup by pdb2adf: M.Swart, 2005

GEOMETRY
END

ATOMS
O      0.0000     0.0000     0.0000
H     -0.5220     0.2660    -0.7570
H     -0.5220     0.2660     0.7570
O      0.0000    -3.2000     0.0000
H      0.0570    -2.2440     0.0000
H      0.9110    -3.4950     0.0000
END

QUILD
  NR_REGIONS=2

  INTERACTIONS
    TOTAL     description 1
    REPLACE region 1   description 3 for description 2
  SUBEND

  REGION 1
    1-3
  SUBEND
  REGION 2
    4-6
  SUBEND

  DESCRIPTION 1 NEWMM
    QMMM
      FORCE_FIELD_FILE $ADFRESOURCES/ForceFields/amber95.ff
      MM_CONNECTION_TABLE
        1 OW  QM     2     3
        2 HW  QM     1
        3 HW  QM     1
        4 OW  MM     5     6
        5 HW  MM     4
        6 HW  MM     4
      SUBEND
      CHARGES
        1  -0.8340
        2   0.4170
        3   0.4170
        4  -0.8340
```

```
        5   0.4170
        6   0.4170
      SUBEND
    END
  SUBEND

  DESCRIPTION 2 NEWMM
    QMMM
      FORCE_FIELD_FILE $ADFRESOURCES/ForceFields/amber95.ff
      MM_CONNECTION_TABLE
        1 OW  QM     2     3
        2 HW  QM     1
        3 HW  QM     1
      SUBEND
      CHARGES
        1  -0.8340
        2   0.4170
        3   0.4170
      SUBEND
    END
  SUBEND

  DESCRIPTION 3
    EPRINT
      SFO NOEIG NOOVL
    END
    XC
     GGA Becke-Perdew
    END
    BASIS
     type TZP
     core small
    END
    SCF
     Converge 1.0e-5 1.0e-5
     Iterations 99
    END
    INTEGRATION 5.0 5.0 5.0
    CHARGE   0.0
  SUBEND

END
ENDINPUT
eor
```

F^- + CH_3Cl: Quild transition state search

Sample directory: quild/quild_qmmm_water2/

This example shows how to use the program Quild for a transition state search of the S_N2 reaction of F^- + CH_3Cl

```
$ADFBIN/quild << eor
Title TransitionState search for Sn2 reaction of F- + CH3Cl
ANALYTICALFREQ
END
XC
 GGA OPBE
END

QUILD
  TSRC
    dist 1 5
    dist 1 6
  SUBEND
END

ATOMS
   C             0.000000     0.000000        0.000000
   H            -0.530807     0.919384693     0.112892
   H            -0.530807    -0.919384693     0.112892
   H             1.061614     0.000000        0.112892
   Cl            0.000000     0.000000       -2.124300
   F             0.000000     0.000000        2.019100
END
Geometry
 TS
End
BASIS
 type TZ2P
 core NONE
END
INTEGRATION 6.0 6.0
SCF
 converge 1.0e-6 1.0e-6
 diis ok=0.01
 iterations 99
END
Charge -1
EPRINT
 SFO noeig noovl
END
endinput
eor
```

DFTB

Aspirin: DFTB geometry optimization

Sample directory: dftb/GO_aspirin/

This example shows how to use the program DFTB for a density functional tight binding (DFTB) geometry optimization.

The directory where the parameter DFTB files will be read can be changed with the RESOURCESDIR keyword. In this case it is assumed that Slater-Koster parameter files have been downloaded from the DFTB.org web site and put in the directory $ADFHOME/atomicdata/DFTB/DFTB.org.

Note that the input for DFTB is not so different from that for ADF.

```
$ADFBIN/dftb << eor

ResourcesDir DFTB.org

Geometry
  RunType GO
  Optim Delocal
  Converge Grad=0.0001
End

Atoms
    C          0.000000  0.000000  0.000000
    C          1.402231  0.000000  0.000000
    C          2.091015  1.220378  0.000000
    C          1.373539  2.425321  0.004387
    C         -0.034554  2.451759  0.016301
    C         -0.711248  1.213529  0.005497
    O         -0.709522  3.637718  0.019949
    C         -2.141910  1.166077 -0.004384
    O         -2.727881  2.161939 -0.690916
    C         -0.730162  4.530447  1.037168
    C         -0.066705  4.031914  2.307663
    H         -0.531323 -0.967191 -0.007490
    H          1.959047 -0.952181 -0.004252
    H          3.194073  1.231720 -0.005862
    H          1.933090  3.376356 -0.002746
    O         -2.795018  0.309504  0.548870
    H         -2.174822  2.832497 -1.125018
    O         -1.263773  5.613383  0.944221
    H         -0.337334  4.693941  3.161150
    H          1.041646  4.053111  2.214199
    H         -0.405932  3.005321  2.572927
End
eor
```

CH_3CN_3H2O: DFTB frequency calculation

Sample directory: dftb/DFTB_Freq_CH3CN_3H2O/

This example shows how to use the program DFTB for a density functional tight binding (DFTB) frequency calculation.

The directory where the parameter DFTB files will be read can be changed with the RESOURCESDIR keyword. In this case it is assumed that Slater-Koster parameter files have been downloaded from the DFTB.org web site and put in the directory $ADFHOME/atomicdata/DFTB/DFTB.org.

Note that the input for DFTB is not so different from that for ADF.

This example calculation is on a CH_3CN molecule surrounded by 3 water molecules. The structure is not optimized, thus there may be imaginary frequencies. In the output the imaginary frequencies have a negative sign.

```
$ADFBIN/dftb << eor

ResourcesDir DFTB.org

ATOMS
 C     1.11496173     0.31854733     0.53703775
 N     0.59792937     0.63879019    -0.43665677
 C     1.77005290    -0.09099876     1.76456883
 H     1.01869497    -0.35318808     2.52004768
 H     2.39216439     0.72757119     2.14786361
 H     2.40589405    -0.96472653     1.57420526
 O    -2.09278652     1.16966361    -1.33023377
 H    -1.59202088     1.50609280    -2.09426593
 H    -1.43148181     0.79477801    -0.72038096
 O     0.90768880    -0.74489752    -3.05331980
 H     0.82923598    -0.51524973    -2.11150163
 H     0.68273250     0.06615005    -3.54260509
 H     0.64422673     1.90218300    -2.26586945
 O     0.02801757     1.96024309    -3.01706698
 H    -0.85740391     1.75708207    -2.66132398
END

Geometry
  RunType Frequencies
End
eor
```

Total energy, Multiplet States, S^2, Localized hole, CEBE

H_2O: Total Energy calculation

Sample directory: adf/Energy_H2O/

If the TOTALENERGY is included the total energy will be calculated.

This example performs single point runs for H_2O with PBE/DZP with frozen cores and all-electron and B3LYP/DZP with all-electron and HARTREEFOCK/DZP with all-electron The tests run in C(2v) symmetry. Integration accuracy is 6.0 which should give total energies accurate at least up to 10^{-4} atomic units. The key EXACTDENSITY is used for higher accuracy of the results.

First example:

```
$ADFBIN/adf <<eor
Title H2O PBE/DZP (frozen core) single point calculation
ATOMS
```

```
  O        0.00000        0.00000        0.00000
  H        0.00000       -0.68944       -0.57851
  H        0.00000        0.68944       -0.57851
END
BASIS
  Type DZP
  Core Small
END
XC
  GGA PBE
END
INTEGRATION 6.0
EXACTDENSITY
TOTALENERGY
eor
```

Note that only energy difference comparisons are meaningful. These are the only energies that play a role in chemistry of course, and for this one does not need total energies.

$Cr(NH_3)_6$: Multiplet States

Sample directory: adf/SD_CrNH3_6/

The computation of multiplet states corresponding to an open-shell system can be carried out with ADF by first computing the 'Average-of-Configuration' (aoc) state, where all orbitals in the open shell are degenerate and equally occupied. This computation is spin-restricted and serves as a fragment file for the multiplet run, where then different occupation numbers are assigned to the various orbitals in the open shell. The corresponding energies are computed in the field of the aoc, which is achieved by *not* iterating the self-consistency equations to convergence but only computing the orbitals in the initial field.

Since ADF requires that all symmetry-partners in an irreducible representation (irrep) have equal occupations, the multiplet calculation, where such orbitals are *not* equally occupied, must be carried out in a formally lower pointgroup symmetry. The pointgroup to select and the appropriate occupation numbers to apply must be worked out by the user 'on paper' in advance. An auxiliary program asf, developed by the group of Claude Daul in Fribourg can be used to determine which calculations are needed, and how to compute the multiplet energies from the results. See the discussion of Multiplet energies in the Theory document.

The script starts with the 'creation' of the required basic atoms, N, H, Cr using a fair basis set quality.

The next step is the computation of the ammonia fragment NH_3. This is not a crucial step here: the multiplet state computation can equally well be carried out by not using any intermediate compound fragments. However, it illustrates once more how a bigger molecule can be built up from smaller, but not trivial fragments.

```
$ADFBIN/adf <<eor
title AMMONIA
NOPRINT sfo,frag,functions

define
xH=0.95522523
yH=xH*sqrt(3)/2
```

```
zH=0.3711068
end

atoms
N    0       0    0
H   -xH      0    zH
H    xH/2  -yH   zH
H    xH/2   yH   zH
end

Basis
 Type TZP
 Core Small
End

symmetry  C(3V)

endinput
eor

mv TAPE21 t21.NH3
```

The input of the atomic coordinates uses expressions, in this case to enforce exact symmetry relations that would otherwise require 14-digit input values or some inaccuracy. The symmetry specification is redundant: the program would also find it by itself.

Average-of-Configuration

The next step is to compute the reference state, with respect to which we will later compute the multiplet states. The reference state is the so-called 'Average-of-configuration' (aoc) state. The result file (TAPE21) of this calculation will be used as a fragment file.

```
$ADFBIN/adf <<eor
title Cr(NH3)6 : Average-of-Configuration run

COMMENT
using NH3-fragments
END

symmetry D(3d)

scf
iterations  25
mix   0.15
end

atoms
 Cr      0.000000     0.000000     0.000000
  N      0.000000     1.714643     1.212436  f=NH3/1
  H      0.000000     1.466154     2.206635  f=NH3/1
  H     -0.827250     2.293404     1.036727  f=NH3/1
  H      0.827250     2.293404     1.036727  f=NH3/1
  N     -1.484924    -0.857321     1.212436  f=NH3/2
```

```
  H     -1.269726    -0.733077     2.206635  f=NH3/2
  H     -1.572521    -1.863121     1.036727  f=NH3/2
  H     -2.399771    -0.430282     1.036727  f=NH3/2
  N      1.484924    -0.857321     1.212436  f=NH3/3
  H      1.269726    -0.733077     2.206635  f=NH3/3
  H      2.399771    -0.430282     1.036727  f=NH3/3
  H      1.572521    -1.863121     1.036727  f=NH3/3
  N      0.000000    -1.714643    -1.212436  f=NH3/4
  H      0.000000    -1.466154    -2.206635  f=NH3/4
  H      0.827250    -2.293404    -1.036727  f=NH3/4
  H     -0.827250    -2.293404    -1.036727  f=NH3/4
  N      1.484924     0.857321    -1.212436  f=NH3/5
  H      1.269726     0.733077    -2.206635  f=NH3/5
  H      1.572521     1.863121    -1.036727  f=NH3/5
  H      2.399771     0.430282    -1.036727  f=NH3/5
  N     -1.484924     0.857321    -1.212436  f=NH3/6
  H     -1.269726     0.733077    -2.206635  f=NH3/6
  H     -2.399771     0.430282    -1.036727  f=NH3/6
  H     -1.572521     1.863121    -1.036727  f=NH3/6
  H     -1.572521     1.863121    -1.036727  f=NH3/6
end

fragments
Cr  t21.Cr
NH3 t21.NH3
end

occupations
 A1.G 8.75
 A2.G 2
 E1.G 16 1.5 0.75
 A1.U 2
 A2.U 8
 E1.U 20
END

end input
eor

mv TAPE21 t21.CrA6ES
```

Occupation numbers are specified, to make certain what the reference state is that we will start from in the subsequent calculations. The result file TAPE21 is saved to serve as fragment file in the subsequent calculations.

One-determinant states

Now, we proceed with the multiplet calculations. In the example they are combined in one single run, but they could also be evaluated in separate runs. For each calculation it is required to:

a) Use the aoc TAPE21 file as fragment file

b) Choose which molecular orbitals in the open shell to occupy: select the appropriate pointgroup symmetry and the UnRestricted key if necessary and specify the occupation numbers, using the irreducible representations of the selected point group.

The results are one-determinant calculations, which must then, later, be combined analytically to obtain the required multiplet energy values.

```
$ADFBIN/adf <<eor
title Cr(NH3)6 : SlaterDeterminants run
NOPRINT frag

symmetry C(I)  !  lower symmetry

scf
iterations 0
end

atoms
   Cr        0.000000       0.000000       0.000000  f=CrA6
    N        0.000000       1.714643       1.212436  f=CrA6
    H        0.000000       1.466154       2.206635  f=CrA6
    H       -0.827250       2.293404       1.036727  f=CrA6
    H        0.827250       2.293404       1.036727  f=CrA6
    N       -1.484924      -0.857321       1.212436  f=CrA6
    H       -1.269726      -0.733077       2.206635  f=CrA6
    H       -1.572521      -1.863121       1.036727  f=CrA6
    H       -2.399771      -0.430282       1.036727  f=CrA6
    N        1.484924      -0.857321       1.212436  f=CrA6
    H        1.269726      -0.733077       2.206635  f=CrA6
    H        2.399771      -0.430282       1.036727  f=CrA6
    H        1.572521      -1.863121       1.036727  f=CrA6
    N        0.000000      -1.714643      -1.212436  f=CrA6
    H        0.000000      -1.466154      -2.206635  f=CrA6
    H        0.827250      -2.293404      -1.036727  f=CrA6
    H       -0.827250      -2.293404      -1.036727  f=CrA6
    N        1.484924       0.857321      -1.212436  f=CrA6
    H        1.269726       0.733077      -2.206635  f=CrA6
    H        1.572521       1.863121      -1.036727  f=CrA6
    H        2.399771       0.430282      -1.036727  f=CrA6
    N       -1.484924       0.857321      -1.212436  f=CrA6
    H       -1.269726       0.733077      -2.206635  f=CrA6
    H       -2.399771       0.430282      -1.036727  f=CrA6
    H       -1.572521       1.863121      -1.036727  f=CrA6
end

fragments
  CrA6  t21.CrA6ES
end

UnRestricted

SlaterDeterminants
  Check AOC
    A1.g   4 0.375         // 4 0.375
```

```
    A2.g   1                // 1
    E1.g:1 4 0.375 0.1875   // 4 0.375  0.1875
    E1.g:2 4 0.375 0.1875   // 4 0.375  0.1875
    A1.u   1//1
    A2.u   4//4
    E1.u:1 5//5
    E1.u:2 5//5
  SUBEND
  State1
    A1.g   4 1              // 4 1
    A2.g   1                // 1
    E1.g:1 4 0      0       // 4 0      1
    E1.g:2 4 0      0       // 4 0      0
    A1.u   1//1
    A2.u   4//4
    E1.u:1 5//5
    E1.u:2 5//5
  SUBEND
  State2
    A1.g   4 1              // 4 1
    A2.g   1                // 1
    E1.g:1 4 0      0       // 4 1      0
    E1.g:2 4 0      0       // 4 0      0
    A1.u   1//1
    A2.u   4//4
    E1.u:1 5//5
    E1.u:2 5//5
  SUBEND
  State3
    A1.g   4 1              // 4 1
    A2.g   1                // 1
    E1.g:1 4 0      1       // 4 0      0
    E1.g:2 4 0      0       // 4 0      0
    A1.u   1//1
    A2.u   4//4
    E1.u:1 5//5
    E1.u:2 5//5
  SUBEND
end

end input
eor
```

The SlaterDeterminants block may contain any number of sub blocks, each starting with an (arbitrary) title record, followed by a set of occupation numbers and closed by a SubEnd record. Each such subkey block specifies a single one-determinant-state calculation. All occupation numbers must reference the irreps of the specified pointgroup symmetry, C(I) in the example, and must be just a reassignment of the electrons that are equally distributed over the corresponding degenerate irreps in the reference aoc calculation.

The so-obtained energies of the one-determinant states can now be combined to calculate the desired multiplet energies. See the Theory document and the adf User's Guide.

Note carefully that in the calculation of the SingleDeterminants, the scf procedure is prevented to cycle to convergence by setting the subkey Iterations to zero in the SCF data block.

CuH+: calculation of S^2

Sample directory: adf/CuH+_S-squared/

Example calculates expectation value of S^2 (< S^2 >) of CuH+ in various symmetries, using unrestricted density functional theory. Last example in this example file calculates this value in the case there are more beta electrons than alpha electrons.

```
$ADFBIN/adf << eor
Title calculate expectation value of S-squared

ATOMS Z-Matrix
 Cu   0 0 0
 H    1 0 0  1.463
END

CHARGE 1.0 -1.0
Unrestricted

FRAGMENTS
H  t21.H
Cu t21.Cu
END

endinput
eor
```

N_2^+: Localized Hole

Sample directory: adf/ModStPot_N2+/

This calculation illustrates:

- a) How to specify the net total charge on a molecule
- b) How to enforce breaking the symmetry that is present in the start-up situation, in this case to localize a hole in the electron density on one of the two equivalent atoms.
- c) How to prevent the scf from oscillating back and forth between the two equivalent situations or from even restoring the unwanted symmetry

```
$ADFBIN/adf <<eor
title N2+  hole localization

atoms
N 0 0  -2.0
N 0 0   2.0
end

Basis
 Type DZP
 Core Small
```

```
End

symmetry C(lin)  ! allow symmetry breaking

unrestricted

Occupations   keeporbitals=3   &
  ! keeporbitals: let the density relax a bit, then fix the MO occups
  sigma 3 // 1 0 1
  pi    2 // 2
end

CHARGE  1  1   ! this duplicates info from "OCCUPATIONS" (check)

modifystartpotential  ! to break the symmetry in the start-up potential
N/1    0.5   0.5
N/2      4   1
end

end input
eor
```

The purpose of this run is to compute the N_2^+ ion, with the hole localized on one of the atoms. In a very small system like N_2^+ this is a tricky thing to do. The program has a tendency towards the symmetric solution, with the hole delocalized. A few trial runs, just putting a net +1 charge into the system, will reveal that clearly.

To achieve the desired situation we apply the key modifystartpotential to break the symmetry of the initial potential. A potential is generated as if the electronic cloud in the second N fragment is spin-polarized in a ratio 4:1 (this precise value is not very relevant), which achieves that *initially* a non-symmetric solution is obtained. The symmetry must be specified, lest the program determine and use the higher symmetry from the nuclear frame. This would prevent any symmetry breaking altogether.

Next, in order to prevent that the system relaxes to the symmetric situation, we apply the *keeporbitals* option of the occupations key. This fixes the occupied orbitals in the sense that in each scf cycle the program will try to keep the electrons in orbitals that resemble the previously occupied orbitals as much as possible.

The key modifystartpotential here demonstrated has a more relevant and less unstable application in larger systems. See the User's Guide for references.

Fe_4S_4: broken spin-symmetry

Sample directory: adf/Fe4S4_BrokenSymm/

This calculation shows a spin-flip restart feature that allows to exchange alpha and beta fit coefficients for selected atoms upon restart. First the high spin configuration with 8 more α-electrons than beta-electrons is calculated (S_Z=4). Next the broken spin-symmetry configuration is calculated (S_Z=0), using the subkey spinflip in the restart key. In this case the spin will be flipped for iron atoms 1 and 2.

```
$ADFBIN/adf <<eor
TITLE Fe4S4 High-spin configuration
ATOMS
```

```
 Fe      -0.000000000000      -1.256142548900       0.888226914500
 Fe       0.000000000000       1.256142548900       0.888226914500
 Fe      -1.256142548900       0.000000000000      -0.888226914500
 Fe       1.256142548900      -0.000000000000      -0.888226914500
  S      -1.845393493800       0.000000000000       1.304890253400
  S       1.845393493800      -0.000000000000       1.304890253400
  S      -0.000000000000      -1.845393493800      -1.304890253400
  S       0.000000000000       1.845393493800      -1.304890253400
END

Symmetry C(2v)

CHARGE 0.0 8.0
UNRESTRICTED

BASIS
type DZ
core Large
createoutput None
END

XC
GGA OPBE
END
end input
eor

mv TAPE21 Fe4S4-high-spin.t21

$ADFBIN/adf <<eor
TITLE Fe4S4 LOW-spin configuration

Restart Fe4S4-high-spin.t21 &
! Make sure atoms specified in the SpinFlip keyword are symmetry-equivalent
  SpinFlip 1 2
End

ATOMS
 Fe      -0.000000000000      -1.256142548900       0.888226914500
 Fe       0.000000000000       1.256142548900       0.888226914500
 Fe      -1.256142548900       0.000000000000      -0.888226914500
 Fe       1.256142548900      -0.000000000000      -0.888226914500
  S      -1.845393493800       0.000000000000       1.304890253400
  S       1.845393493800      -0.000000000000       1.304890253400
  S      -0.000000000000      -1.845393493800      -1.304890253400
  S       0.000000000000       1.845393493800      -1.304890253400
END

Symmetry C(2v)

CHARGE 0.0 0.0

UNRESTRICTED
```

```
BASIS
type DZ
core Large
createoutput None
END

XC
GGA OPBE
END

eor
```

NNO: Core-electron binding energies

Sample directory: adf/CEBE_NNO/

ADF is well suited for calculating Core Electron Binding Energies (CEBEs). In this example it is shown how one can differentiate between the 1s CEBEs of the two non-equivalent nitrogen atoms in N_2O, using a delta-SCF technique. It starts with a regular calculation that has the purpose of preparing a reference TAPE21 file for the NNO molecule, which will later be useful in the energy analysis. The result file is saved to t21.NNO.

The same GGA functional is specified throughout the run. The amount of output is reduced by using some print keys.

```
$ADFBIN/adf -n1  << eor
create N $ADFRESOURCES/TZ2P/N
xc
gradients pw86x pw91c
end
end input
eor

mv TAPE21 t21.N

$ADFBIN/adf -n1  << eor
create O $ADFRESOURCES/TZ2P/O
xc
gradients pw86x pw91c
end
end input
eor

mv TAPE21 t21.O
```

The prepare the nitrogen atom with a core hole (restricted) will be used as a fragment later. This enables selection of where the core hole should be.

```
$ADFBIN/adf -n1  << eor
title   N atom core hole

ATOMS
```

```
N   0.0    0.0     0.0
end

fragments
N      t21.N
end

xc
gradients pw86x pw91c
end

eprint
SFO noeig noovl
end

occupations
s    1 2
p    3
end

end input
eor

mv TAPE21 t21.N_ch
```

Now perform the restricted ground state molecule for analysis later. The TAPE21 result file is saved.

```
$ADFBIN/adf << eor
title   NNO

noprint sfofragpop fragsfo

xc
gradients pw86x pw91c
end

ATOMS
N   0.0    0.0    -1.1284
N   0.0    0.0     0.0
O   0.0    0.0     1.1841
end

fragments
N      t21.N
O      t21.O
end

eprint
SFO noeig noovl
end

end input
eor
```

```
mv TAPE21 t21.NNO
```

Next follow two sets of almost identical calculations in which a 1s electron is removed from one or the other N atom (please note that the deepest s level is associated with the 1s of the oxygen atom). The molecular NNO result file is used as fragment. An unrestricted calculation is done and a positive charge is specified. The final result file for the molecule with the core hole is saved. Then another calculation is done to conveniently obtain the energy with respect to the normal molecule. This is repeated for a core hole on the other N atom.

```
$ADFBIN/adf <<eor
title NNO unrestricted core hole

noprint sfofragpop fragsfo

ATOMS
N   0.0    0.0    -1.1284       f=N_ch
N   0.0    0.0     0.0          f=N
O   0.0    0.0     1.1841       f=O
end

xc
gradients pw86x pw91c
end

fragments
N_ch  t21.N_ch
N     t21.N
O     t21.O
end

eprint
SFO noeig noovl
end

unrestricted
charge 1 1

occupation
sigma  1 1 1 4 // 1 0 1 4
pi     4       // 4
end

end input
eor

mv TAPE21 t21.NNO.unr1
```

In the second calculation the result file of one of the unrestricted NNO calculations is used as restart file, which ensures that the hole stays at its place, because the starting density is already correct. The result file t21.NNO for the normal NNO calculation is specified as fragment to serve as an energy reference. The final Bonding Energy printed by ADF indicates what the CEBE is. However, please check Refs.[Chong, 2002

#1239][Chong, 2002 #1238] for more detailed information on Core-Electron Binding Energies. These references also contain infomration on empirical corrections that may have to be made on the final numbers.

```
$ADFBIN/adf <<eor
title   NNO unr. core hole

noprint sfofragpop fragsfo

xc
gradients pw86x pw91c
end

restart  t21.NNO.unr1

ATOMS
N   0.0    0.0    -1.1284       f=NNO
N   0.0    0.0     0.0          f=NNO
O   0.0    0.0     1.1841       f=NNO
end

eprint
SFO noeig noovl
end

fragments
NNO      t21.NNO
end

unrestricted
charge 1 1

occupation
  sigma  1 1 1 4 // 1 0 1 4
  pi     4       // 4
end

end input
eor
```

Similarly, one could easily have prepared an oxygen with a core hole and determined the CEBE of the oxygen 1s atom.

Spectroscopic Properties

IR Frequencies, (resonance) Raman, VCD, Franck-Condon factors

NH_3: Numerical Frequencies

Sample directory: adf/Freq_NH3/

Summary:

- Frequencies with symmetric displacements
- Frequencies with Cartesian displacements
- Isotope effects in the frequencies

Frequencies with symmetric displacements

Computation of frequencies by symmetric displacements. The assumed equilibrium input structure should be given in Cartesian coordinates.

The symmetry is determined automatically by the program as C(3v), from the input coordinates. During the calculation first symmetric atomic displacements are constructed. The number of such displacements in each irreducible representation corresponds to the number of frequencies with the corresponding symmetry. All displaced geometries within one representation have the same symmetry, which enables us to use it to speed up the computation significantly. Another advantage of having the same symmetry is that the numerical integration data can be reused efficiently (see SMOOTH option) thus reducing the level of numerical noise in gradients and force constant matrix.

```
$ADFBIN/adf <<eor
title NH3 frequencies in symmetric displacements

atoms
   N               0.0000    0.0000    0.0000
   H               0.4729    0.8190    0.3821
   H              -0.9457    0.0000    0.3821
   H               0.4729   -0.8190    0.3821
end

Basis
 Type TZP
 Core Small
End

geometry
  frequencies Symm
end

thermo  T=300,400

integration  5.0
```

```
end input
eor
```

Frequencies with Cartesian displacements

Computation of frequencies by Cartesian displacements. The assumed equilibrium input structure is given in internal coordinates. A dummy atom is used for a convenient definition of the Z-matrix such that it reflects the pointgroup symmetry C(3v).

```
$ADFBIN/adf <<eor
title NH3 frequencies

define
  rNH=1.02
  theta=112
  phi=120
end

atoms   Z-matrix
  XX   0 0 0
  N    1 0 0   1.0
  H    2 1 0   rNH   theta
  H    2 1 3   rNH   theta   phi
  H    2 1 4   rNH   theta   phi
end

Basis
 Type TZP
 Core Small
End

geometry
  optim  cartesian
  frequencies
end

thermo  T=300,400

integration  5.0

end input
eor
```

The symmetry is determined automatically by the program as C(3v), from the input coordinates. In a Frequencies calculation the symmetry (specified on input or computed internally) is used for analysis and in some cases to speed up the calculation.

The equilibrium coordinate values are supplied as identifiers that are associated with values in the define block.

Unlike using the geovar key, applying the define key does not mean anything in the sense that the various coordinates that refer to the same identifier would be forced to remain equal; it is just a way to display (to the

human reader) symmetry in the equilibrium values, to avoid typing errors and to allow an easy adjustment of starting coordinates for another calculation.

Since the atomic coordinates are input in Z-matrix format, the program would by default carry out displacements in internal coordinates to scan the energy surface and hence compute force constants and frequencies. This is overriden by specifying in the geometry block optim cartesian: carry out *cartesian* displacements.

The key thermo addresses the thermodynamical analysis (only available in a Frequencies calculation, otherwise ignored). The specification 'T=300,400' means that the thermodynamic properties are printed for the temperature range 300-400K, in steps of 10K (default) and for a pressure of 1.0 atmosphere (default).

Frequencies calculations suffer easily from numerical inaccuracies. Therefore, the default numerical integration precision in a Frequencies calculation is much higher than in an ordinary single-point or minimization run. Here we specify the INTEGRATION level to be 5.0 (quite high, but the default for Frequencies is even 6.0).

Isotope effects in the frequencies

Rename the TAPE21 result file of the previous calculation so we can restart with other masses. Calculate a different isotope of H, in this case deuterium. It will differ from the original one only in the mass of the nucleus. Repeat the frequency calculation with different fragments. It is important to preserve symmetry at this step so we replace fragment files for ALL H atoms. If you want to replace only one fragment then the original calculation must be performed the same way, with different fragment names.

```
mv TAPE21 restart.t21

$ADFBIN/adf -n1 <<eor
create H M=2.014101779 $ADFRESOURCES/TZP/H
eor

mv TAPE21 t21.D

$ADFBIN/adf <<eor
title NH3 frequencies

define
  rNH=1.02
  theta=112
  phi=120
end

atoms   Z-matrix
  XX   0 0 0
  N    1 0 0   1.0
  H    2 1 0   rNH   theta
  H    2 1 3   rNH   theta   phi
  H    2 1 4   rNH   theta   phi
end

Fragments
  N t21.N
! The different isotope mass sits in the next line.
```

```
  H t21.D
End

geometry
  optim  cartesian
  frequencies
end

! Restart the frequency calculation.
! In fact ADF should perform only one geometry cycle
restart restart.t21

thermo  T=300,400

integration  5.0

end input
eor
```

UF_6: Numerical Frequencies, spin-orbit coupled ZORA

Sample directory: adf/Freq_UF6/

Calculation of spin-orbit coupled ZORA gradients for the LDA and GGA functionals is possible since the ADF2007.01 version.
Summary:

- Geometry optimization
- Frequencies with symmetric displacements
- Both for scalar relativistic ZORA and spin-orbit coupled ZORA

Geometry optimization

Here only the spin-orbit coupled input file for ADF is given (in the scalar relativistic case change "spinorbit" in "scalar"). The resulting TAPE21 is saved such that it can be used in the frequency calculation.

```
$ADFBIN/adf <<eor
Title  UF6 geometry optimization: scalar ZORA
integration 5 5
Geometry
 conv grad=1e-4
End
relativistic spinorbit zora
Basis
 Type TZP
end
ATOMS cartesian
1 U   .00000  .00000   .00000
2 F  2.00000  .00000   .00000
3 F -2.00000  .00000   .00000
4 F   .00000  2.0000   .00000
```

```
5 F   .00000 -2.0000   .00000
6 F   .00000   .0000  2.00000
7 F   .00000   .0000 -2.00000
END
end input
eor
mv TAPE21 UF6.t21
```

Frequencies with symmetric displacements

Computation of frequencies by symmetric displacements. The assumed equilibrium input structure should be given in Cartesian coordinates. The calculation starts with the optimized structure read from UF6.t21 (restart file). Again only the spin-orbit coupled input file for ADF is given.

The symmetry is determined automatically by the program as O(H), from the input coordinates. During the calculation first symmetric atomic displacements are constructed. The number of such displacements in each irreducible representation corresponds to the number of frequencies with the corresponding symmetry. All displaced geometries within one representation have the same symmetry, which enables us to use it to speed up the computation significantly. Another advantage of having the same symmetry is that the numerical integration data can be reused efficiently (see SMOOTH option) thus reducing the level of numerical noise in gradients and force constant matrix.

In case of spin-orbit coupling the frequencies can not (yet) be calculated with analytical second derivatives.

```
$ADFBIN/adf <<eor
Title  UF6 frequencies and IR intensities: spinorbit ZORA
Restart  UF6.t21
Geometry
  Frequencies Symm
End
integration 5 5
relativistic spinorbit zora
Fragments
 U t21.U
 F t21.F
end
ATOMS cartesian
1 U   .00000  .00000   .00000
2 F  2.00000  .00000   .00000
3 F -2.00000  .00000   .00000
4 F   .00000  2.0000   .00000
5 F   .00000 -2.0000   .00000
6 F   .00000   .0000  2.00000
7 F   .00000   .0000 -2.00000
END
end input
eor
```

CN: Analytic Frequencies

Sample directory: adf/CN_SecDeriv/

The ADF2002.01 version featured analytic second derivatives (SD) for the first time. This initial implementation had severe limitations, both in terms of speed, as in terms of user-friendliness of the output and the number of available options. As of ADF2006.01, the implementation has been significantly improved. More specifically:

- what was formerly a separate SD program is now part of the main ADF executable;
- most current GGA's have been implemented, with the exeption of PW91;
- the speed of the code, serial as well as parallel, has been improved;
- full support for symmetry has been implemented, including linear molecules.

Calculation of analytical second derivatives is requested by specifying

```
AnalyticalFreq
End
```

in the main ADF input.

A high accuracy is specified for the numerical integration to be sure of reliable results.i In general, it seems advisable to use high accuracy for heavy nuclei at the moment, whereas default integration accuracy is usually sufficient for light atoms. Further, high integration accuracy is more needed in the atomic spheres than in the rest of the molecule. A cost-effective solution may therefore be to specify a higher integration accuracy in the spheres only (using the accsph subkey of the INTEGRATION keyword).

```
$ADFBIN/adf << eor
title CN

atoms
 N  -1.3  0.0   0.0
 C   0.0  0.0   0.0
end

Basis
 Type DZ
 Core None
End

charge -1

XC
 LDA Xonly
End

integration 6.0

AnalyticalFreq
End

End input
eor
```

After SCF is completed, the energy second derivatives matrix is calculated and analysed, which yields in this case one frequency.

CH_4: Analytic Frequencies

Sample directory: adf/CH4_SecDeriv/

In this example, we use a new feature of adf2006.01: geometry optimization immediately followed by calculation of frequencies. This is done by specifying the Geometry and AnalyticalFreq input blocks in one file.

Note: when using this feature, one should generally set the integration accuracy to a value appropriate for the frequencies calculation, which is about 5.0. In order to save time we neglect the recommendation in this example.

```
$ADFBIN/adf << eor
title CH4 LDA potential

Define
 ZERO = 0.0
 RCH = 1.0850
 DCH = sqrt(3)*(RCH/3)
End

Atoms
 C  0.0  0.0  0.0
 H  DCH -DCH  DCH
 H  DCH  DCH -DCH
 H -DCH  DCH  DCH
 H -DCH -DCH -DCH
End

Basis
 Type TZP
 Core None
End

integration 4.0

Geometry
  Optim all
  converge grad=0.0001
End

AnalyticalFreq
End

End input
eor
```

HI: Analytic Frequencies, scalar ZORA

Sample directory: adf/HI_SecDer_ZORA/

The main difference of this example to the previous examples is that a ZORA Hessian is calculated in this example, through the line:

RELATIVISTIC scalar ZORA

Furthermore, the suggestion to use high integration accuracy in the atomic spheres only is shown explicitly here.

```
$ADFBIN/adf << eor
TITLE HI scalar, ZORA,
DEFINE
 ZERO = 0.0
 R = 1.6090
 X1 = ZERO
 Y1 = ZERO
 Z1 = R
 X2 = ZERO
 Y2 = ZERO
 Z2 = ZERO
END
ATOMS
 I X2 Y2 Z2
 H X1 Y1 Z1
END
XC
 LDA Xonly
END
RELATIVISTIC scalar ZORA
COREPOTENTIALS t12.rel &
 H 1
 I 2
END
FRAGMENTS
 H t21H
 I t21I
END
integration
  accint  4.0
  accsph  6.0
end

AnalyticalFreq
End

end input
eor
```

Ethanol: mobile block Hessian

Sample directory: adf/MBH_Ethanol/

A frequency calculation is performed using the mobile block Hessian (MBH) method. The coordinates in the ATOMS section should be the partially optimized coordinates (or the fully optimized coordinates would work too). The next input for ADF shows how to perform a frequency calculation with MBH. The flag b=b1 in the ATOMS section adds the label 'b1' to some of the atoms. Only the four atoms labeled 'b1' (CH_3) will be considered as a block with fixed internal geometry.

```
$ADFBIN/adf <<eor
TITLE ethanol: second derivatives with MBH approach. CH3 is treated as a rigid block

ATOMS
1 C      -0.029587   -0.006554    0.008124     b=b1
2 H      -0.087498   -0.025163    1.109913     b=b1
3 H       1.027473   -0.056237   -0.302751     b=b1
4 H      -0.565305   -0.891154   -0.376242     b=b1
5 C      -0.694908    1.238909   -0.501807     b=b2
6 H      -0.670258    1.265092   -1.608847     b=b2
7 O      -2.069894    1.175059   -0.017251
8 H      -0.182335    2.138977   -0.109315     b=b2
9 H      -2.586972    1.972802   -0.317216
END

SYMMETRY nosym

BASIS
  type DZ
  core Large
  CreateOutput None
END

XC
LDA SCF VWN
END

GEOMETRY
   frequencies
   mbh b1
   branch new
END

INTEGRATION 6.0
End input
eor
```

For comparison in this example also a calculation is performed without any restrictions.

```
$ADFBIN/adf <<eor
TITLE ethanol: complete vibrational spectrum, compare with MBH above

ATOMS
1 C      -0.029587   -0.006554    0.008124
2 H      -0.087498   -0.025163    1.109913
3 H       1.027473   -0.056237   -0.302751
4 H      -0.565305   -0.891154   -0.376242
5 C      -0.694908    1.238909   -0.501807
```

```
6 H      -0.670258    1.265092   -1.608847
7 O      -2.069894    1.175059   -0.017251
8 H      -0.182335    2.138977   -0.109315
9 H      -2.586972    1.972802   -0.317216
END

BASIS
  type DZ
  core Large
  CreateOutput None
END

XC
LDA SCF VWN
END

AnalyticalFreq
End

INTEGRATION 5.0
End input
eor
```

NH_3: Raman

Sample directory: adf/MBH_CH4/

A frequency calculation is performed using the mobile block Hessian (MBH)method. The coordinates in the ATOMS section should be the partially optimized coordinates (or the fully optimized coordinates would work too).

Example input how to do a block constraint:

```
ATOMS
    C         0.000000    0.000000    0.000000 b=b1
    H         0.634671    0.634671    0.634671 b=b1
    H        -0.634671   -0.634671    0.634671 b=b1
    H        -0.634671    0.634671   -0.634671 b=b1
    H         0.634671   -0.634671   -0.634671
END
CONSTRAINTS
 block b1
END
```

Such geometry optimization will not be discussed here any further. The next input for ADF shows how to perform a frequency calculation with MBH.

```
$ADFBIN/adf <<eor
TITLE Methane

BASIS
 Type DZ
```

```
 Core None
END

ATOMS
    C         0.000000    0.000000    0.000000 b=b1
    H         0.634671    0.634671    0.634671 b=b1
    H        -0.634671   -0.634671    0.634671 b=b1
    H        -0.634671    0.634671   -0.634671 b=b1
    H         0.634671   -0.634671   -0.634671 b=b2
END

integration 8 8 8
SYMMETRY nosym

GEOMETRY
   frequencies disrad=0.001
   mbh b1
   branch new
END
End input
eor
```

The flag b=b1 in the ATOMS section adds the label 'b1' to some of the atoms. The four atoms labeled 'b1' will be considered as a block with fixed internal geometry.

In the GEOMETRY section, a Mobile Block Hessian calculation is requested by using the FREQUENCIES and MBH keywords. Here the atoms with label 'b1' are selected to be in the same mobile block. The position/orientation of the block are supposed to be optimized in a preceding partial optimization run. In the vibrational analyis, the block 'b1' is only allowed to vibrate as a whole. The number of resulting modes/ frequencies is 3 for the fifth atom plus 6 for the block 'b1' (3 position/3 orientation), resulting in 9 frequencies in total. Since 6 of those frequencies are zero due to translational and rotational invariance of the system, one will find 3 non-zero characteristic frequencies in the output. In practice with ADF not exactly 6 zero's are found, but they are close to zero.

The quality of the frequencies/modes depends largely on the block choice. Best results are obtained when grouping atoms in a block if those atoms are known to form rather rigid structures. For instance, grouping the 11 atoms of benzene side group into a block, will usually result in representative frequencies. In this example the block choice is only illustrative for the methodology.

NH_3: Raman

Sample directory: adf/Freq_NH3_RAMAN/

Summary:

- Analytical frequencies with subsequent calculation of Raman intensities in a range
- Numerical frequencies including all Raman intensities

Raman Intensities for Selected Frequencies

The *RamanRange* keyword (available since ADF2007.01) can be used to calculate Raman intensities for a range of frequencies only. Using this option is a fast alternative for the existing method of calculating Raman intensities, which is described in the second part of this example.

Two values defining an interval of frequencies to calculate the Raman intensities for. The Raman intensities are calculated by numerical differentiation of the polatizability tensor. Only frequencies frequencies withing the interval that are known to be Raman-active will be included.

```
$ADFBIN/adf <<eor
title NH3 frequencies and calculation of Raman intensities in the range 0-2000 cm-1
atoms
   N               0.0000    0.0000    0.0000
   H               0.4729    0.8190    0.3821
   H              -0.9457    0.0000    0.3821
   H               0.4729   -0.8190    0.3821
end
Basis
 Type TZP
 Core Small
End
AnalyticalFreq
end
thermo  T=300,400
integration  5.0
end input
eor

mv TAPE21 NH3_freqs.t21

$ADFBIN/adf <<eor
title NH3 Raman intensities in the range 0-2000 cm-1
atoms
   N               0.0000    0.0000    0.0000
   H               0.4729    0.8190    0.3821
   H              -0.9457    0.0000    0.3821
   H               0.4729   -0.8190    0.3821
end
Restart NH3_freqs.t21
Fragments
 H t21.H
 N t21.N
End
RamanRange 0.0 2000.0
thermo  T=300,400
integration  5.0
end input
eor
```

Raman Intensities for All Frequencies

Raman scattering intensities and depolarization ratios for all molecular vibrations at a certain laser frequency can be calculated in a single run. The run type must be Frequencies and the RESPONSE key is used to specify that Raman intensities are computed.

In this example the static Raman scattering is calculated (ω = 0). This type of calculation is very similar to an IR intensity calculation. In fact, all IR output is automatically generated as well. At all distorted geometries the dipole polarizability tensor is calculated. This is very time-consuming and is only feasible for small molecules.

```
$ADFBIN/adf <<eor
title NH3 frequencies with Raman intensities
atoms
   N               0.0000    0.0000    0.0000
   H               0.4729    0.8190    0.3821
   H              -0.9457    0.0000    0.3821
   H               0.4729   -0.8190    0.3821
end
Fragments
 H t21.H
 N t21.N
End
geometry
  frequencies
end
response
  raman
end
thermo  T=300,400
integration  5.0
end input
eor
```

HF: Resonance Raman, excited state finite lifetime

Sample directory: adf/HF_ResonanceRaman/

Example shows a calculation of the Resonance Raman spectrum (RRS) of HF. In this example the RRS is calculated from the geometrical derivatives of the frequency-dependent polarizability, including a finite lifetime.

In the ADF input one then needs to include the subkey FREQUENCIES of the key GEOMETRY (numerical frequencies) and include the subkeys RAMAN and LIFETIME of the key AORESPONSE.

```
$ADFBIN/adf << eor
title HF ao-raman

BASIS
 F DZP/F
 H DZP/H
```

```
END

GEOMETRY
 Frequencies
END

Symmetry NOSYM

Atoms
 H 0.0000 0.0000 0.0000
 F 0.0000 0.0000 0.9170
End

aoresponse
 frequency 1 0.52362 Hartree
 lifetime 0.0034
 raman
end

NOPRINT SFO

END INPUT
eor
```

Note that used basis set is too small to get accurate results.

Uracil: Resonance Raman, excited state gradient

Sample directory: adf/Vibron_RR_uracil/

Example shows a calculation of the Resonance Raman spectrum (RRS) of uracil. In this example the RRS is calculated using the excited-state gradient.

A frequency restart file 'restart.freq' is used as input in the resonance Raman calculation. This restart file is the TAPE21 of a frequency calculation of the runfile 'restart.freq.run'.

First the to ASCII dumped TAPE21 'restart.freq.ascii' is undumped again to make a binary file.

```
cp $ADFHOME/examples/Test/e_Vibron_RR_uracil/restart.freq.ascii .
$ADFBIN/udmpkf < restart.freq.ascii restart.freq
```

Next the resonance Raman calculation is performed by setting the 'VIBRON' subkey in the 'GEOMETRY' block key, including both the 'EXCITATION' block key and the 'VIBRON' block key. These are the only differences with the frequency run where only the 'FREQUENCIES' subkey was set in the 'GEOMETRY' block key., and the 'EXCITATION' and 'VIBRON' block key were not set.

```
$ADFBIN/adf << eor
Title Input generated by modco
EPRINT
  SFO NOEIG NOOVL NOORBPOP
  SCF NOPOP
END
```

```
NOPRINT BAS FUNCTIONS
UNITS
  length angstrom
  angle degree
END
ATOMS
  N        -0.0147481688       -0.0251586720        0.0000000000
  C        -0.0263429706        1.3809974655        0.0000000000
  N         1.2556533768        1.9305098959        0.0000000000
  C         2.5041083561        1.2440596334        0.0000000000
  C         2.3755611578       -0.2074475201        0.0000000000
  C         1.1446314693       -0.7882184482        0.0000000000
  H        -0.9346804118       -0.4675883900        0.0000000000
  O        -1.0845317554        2.0515533614        0.0000000000
  H         1.3029888073        2.9549419374        0.0000000000
  O         3.5819185026        1.8899458170        0.0000000000
  H         3.2859343437       -0.7987226158        0.0000000000
  H         0.9976482662       -1.8650665505        0.0000000000
END
BASIS
  type DZ
  core NONE
END
XC
  GGA Becke88 Perdew86
END
SYMMETRY tol=0.001
GEOMETRY
  VIBRON
END
SCF
  iterations 50
  converge 1.0e-6 1.0e-6
  mixing 0.2
  lshift 0.0
  diis n=10 ok=0.5 cyc=5 cx=5.0 cxx=10.0
END
EXCITATION
  ONLYSING
  LOWEST 5
END
MBLOCKBIG
VIBRON
  NMTAPE restart.freq
  RESRAMAN
  STPSIZ 0.1
  DOMODES 11 13 16 17
  DSCHEME ELCHAR
END
INTEGRATION 4.0 4.0
END INPUT
```

Note that used basis set is too small to get accurate results.

NHDT: Vibrational Circular Dichroism

Sample directory: adf/VCD_COG_NHDT/

Analytical frequencies with subsequent calculation of vibrational circular dichroism (VCD)

The *VCD* keyword (available since ADF2007.01) can be used to calculate VCD spectra. It is important to note that the VCD keyword only works in combination with the keys AnalyticalFreq and symmetry NOSYM.

Recomended is use to use high accuracy for the geometry optimization which one needs to do before the frequency calculation. This simple example is an NHDT molecule, which is NH_3 where one hydrogen atom is replaced with deuterium and another with tritium.

First the atoms are created, next the molecule is calculated.

```
$ADFBIN/adf -n1 <<eor
create H q=1 m=2.014101778 file=$ADFRESOURCES/TZP/H
XC
 gga Becke Perdew
end
end input
eor
mv TAPE21 t21.D

$ADFBIN/adf -n1 <<eor
create H q=1 m=3.01604927 file=$ADFRESOURCES/TZP/H
XC
 gga Becke Perdew
end
end input
eor
mv TAPE21 t21.T

$ADFBIN/adf -n1 <<eor
create N file=$ADFRESOURCES/TZP/N
XC
 gga Becke Perdew
end
end input
eor
mv TAPE21 t21.H

$ADFBIN/adf -n1 <<eor
create N file=$ADFRESOURCES/TZP/N
XC
 gga Becke Perdew
end
end input
eor
mv TAPE21 t21.N
```

Next the molecule is calculated.

```
$ADFBIN/adf <<eor
Title Single Point calc.

Atoms
    N         0.000000    0.000000    0.010272
    H        -0.471582   -0.816803    0.407861
    H.D       0.943163    0.000000    0.407861
    H.T      -0.471582    0.816803    0.407861
End

Symmetry NOSYM

xc
 GGA BP86
end

Fragments
N     t21.N
H     t21.H
H.D   t21.D
H.T   t21.T
End

Integration 7.0

AnalyticalFreq
End

VCD

end input
eor
```

NO_2: Franck-Condon Factors

Sample directory: adf/FranckCondon_NO2/

As an example of a Franck-Condon calculation, lets look at the transition of NO_2 to NO_2^-. NO_2 is a small molecule with only three vibrational modes. Putting an extra electron on the molecule will cause a big displacement, resulting in large electron-phonon couplings.

In general, the larger the molecule, the smaller the displacement and hence the electron-phonon couplings and Franck-Condon factors. Moreover, larger molecules have more vibrational modes, meaning that the already smaller displacement will generally be smeared out over more modes, resulting in an additional decrease in electron-phonon couplings. This is fortunate, since the number of Franck-Condon factors increases factorially with the number of vibrational modes, making it prohibitively expensive to take more than a few vibrational quanta into account for most molecules.

In order to calculate the Franck-Condon factors for Nitrite and Nitrogen dioxide, the equilibrium positions of the nuclei and the vibrational modes have to be obtained (the geometry optimizations are not shown here):

```
$ADFBIN/adf << eor

TITLE Nitrogen dioxide

ATOMS
    N         0.000000    0.000000   -0.016179
    O         0.000000    1.098646   -0.492918
    O         0.000000   -1.098646   -0.492918
END

BASIS
    CORE NONE
    TYPE DZP
END

XC
    LDA SCF VWN
END

ANALYTICALFREQ
END

eor

mv TAPE21 NO2.t21
rm t21.* logfile

$ADFBIN/adf << eor

TITLE Nitrite

ATOMS
    N         0.000000    0.000000    0.093662
    O         0.000000    1.120366   -0.540999
    O         0.000000   -1.120366   -0.540999
END

CHARGE -1.0 1.0
UNRESTRICTED

BASIS
    CORE NONE
    TYPE DZP
END

XC
    LDA SCF VWN
END

ANALYTICALFREQ
END

eor
```

```
mv TAPE21 NO2-.t21
rm t21.* logfile
```

This runscript produces two TAPE21 files containing the frequencies and the normal modes for both charge states. Lets first look at the ground state to ground state overlap:

```
$ADFBIN/fcf << eor

STATES NO2.t21 NO2-.t21

QUANTA 0 0

TRANSLATE
ROTATE

eor
```

Here, zero vibrational quanta are specified for both charge states, which corresponds to the vibrational ground state. Looking at the standard output, we see for NO_2:

Frequency (cm^{-1})	***λ (dimensionless)***
1072.490460	1.216108
1434.990571	1.873915
1875.876562	0.000000

And for NO_2^-:

Frequency (cm^{-1})	***λ (dimensionless)***
816.952242	0.594643
1264.390562	2.071319
1314.362101	0.000000

Both states have two vibrational modes with a significant electron-phonon coupling. The ground state to ground state Franck-Condon factor is therefore expected to be quite small. And indeed, looking at the output, we see that it is $0.7944250686*10^{-2}$, less than one percent of the total.

Since NO_2 has only three vibrational modes, many quanta can be included, and this indeed turns out to be necessary. Setting the maximum number of quanta at 20 results in 1771 permutations for both states and a total of 3136441 Franck-Condon factors. Even with so many factors, the average sum is still only 0.5196635779. Including one extra vibrational quanta results in an additional 960135 Franck-Condon factors, but an average sum of only 0.5280010614, i.e. less than a percent more. This one percent is smeared out over so many factors that their individual contributions become negligible.

Time-dependent DFT applications

Au_2: Response Properties

Sample directory: adf/Au2_Resp/

A calculation of response properties of the Au_2 dimer, with ZORA relativistic corrections

```
$ADFBIN/adf << eor
Title Au2, Response Properties

XC
  GGA  LB94
End

Relativistic Scalar ZORA
CorePotentials t12.Au

Atoms
  Au   0.0 0.0 1.236
  Au   0.0 0.0 -1.236
End

Fragments
  Au  t21.Au
End

Symmetry D(8h)

Excitations
  Lowest 10
  TOLERANCE 1d-10
End

Response
  AllComponents
End

End Input
eor
```

In the response module infinite symmetries cannot be handled (see the User's Guide), so we specify a lower subgroup in the input file, here D(8h).

In this sample run the LB94 potential is used. The implementation implies that the XC potential is evaluated from the exact charge density, rather than the cheaper and faster fitted density (as is the case for other XC functionals). This means that the computation times are longer. In a small molecule like Au_2 this hardly shows, but in larger molecules the differences may be more significant.

Excitation energies are computed, in principle the lowest 10 in each irrep of the symmetry (see, however, the User's Guide).

CN: excitation energies open shell molecule

Sample directory: adf/CN_unr_exci/

Calculation of the excitation energies of the open shell molecule CN

```
$ADFBIN/adf << eor
Title excitation energies of CN

Atoms
 C  .0000  .0000  .0000
 N  .0000  .0000 1.1718
End

unrestricted
charge 0 1

excitations
 lowest 20
end

Basis
 Type AUG/ADZP
End

End input
eor
```

In this example, the lowest 20 excitation energies of CN are calculated in a spin-unrestricted TDDFT calculation. In the MO → MO transitions part for the excitations of the output file, the spin of each molecular orbitals are also specified to help assign the spin state of the excited states. The transitions are always from α spin-orbital to α spin-orbital or from β spin-orbital to β spin-orbital.

Next the same example for CN is given with the Tamm-Dancoff approximation (TDA) approximation (including TDA in the input). Due to this approximation the calculated excitation energies will not be exactly the same as in the first example.

The third calculation is the calculation of spin-flip excitation energies for CN. Again these energies will not be exactly the same as in the first example. For open-shell molecules, spin-flip transition can result in transition to the ground state with a different S_z value, while the symmetry of the transition density is A1 (Σ^+ for linear molecules). The excitation energy of this transition should be zero and this can be used to test the reliability of spin-flip TDDFT. Indeed the calculation of the spin-flip excition energies of CN shows one value which is close to zero and has a transition density of Σ^+ symmetry.

```
$ADFBIN/adf << eor
Title spin-flip excitation energies (TDA) of CN

Atoms
 C  .0000  .0000  .0000
 N  .0000  .0000 1.1718
End

unrestricted
charge 0 1

excitations
 lowest 20
end

SFTDDFT
```

```
TDA
FORCELDA

Basis
 Type AUG/ADZP
End

End input
eor
```

SiH_2: spin-flip excitation energies

Sample directory: adf/SiH2_spinflip/

Calculation of the spin-flip excitation energies of the open shell molecule SiH_2

```
$ADFBIN/adf << eor
Title spin-flip excitation energies of SiH2
Atoms Zmatrix
 Si 0  0  0
 H  1  0  0  1.5145
 H  1  2  0  1.5145  92.68
End

excitations
  lowest 20
end

unrestricted
charge 0 2

SFTDDFT
FORCEALDA
TDA

Basis
 Type TZ2P
End

End input
eor
```

In this example, the lowest 20 spin-flip excitation energies of SiH_2 are calculated in a spin-unrestricted TDDFT calculation.

In this case an excited state is used as reference, which means that there can also be a negative excitation energy, which is indeed the case. The electron configuration used in the SCF is $(a_1)^1(b_1)^1$, with S_z=1, thus a $^3B_1$ state, which is an excited state. The $^1A_1$ state with electron configuration $(a_1)^2$ is lower in energy, and is the ground state.

There is also an excited $^1A_1$ state with electron configuration $(b_1)^2$. The transition from the ground $^1A_1$ state to the excited $^1A_1$ state is an excitation from the electron configuration $(a_1)^2$ to $(b_1)^2$. This transition is

actually a double excitation, which means that some double excitations can be reached using spin-flip TDDFT with carefully selected reference states.

In the MO → MO transitions part for the excitations of the output file, the spin of each molecular orbitals are also specified to help assign the spin state of the excited states. Note that in these spin-flip calculations the transitions are always from α spin-orbital to β or from β spin-orbital to α spin-orbital.

For open-shell molecules, spin-flip transition can result in transition to the ground state with a different S_z value, while the symmetry of the transition density is A1. The excitation energy of this transition should be zero and this can be used to test the reliability of spin-flip TDDFT. Indeed the calculation of the spin-flip excition energies of SiH_2 shows one value which is close to zero and has a transition density of A1 symmetry.

The $^1A_1$ state with electron configuration $(a_1)^2$ can also be used in the calculation of the excitation energies. This is a closed shell configuration, in which case we do not need the spin-flip method.

```
$ADFBIN/adf << eor
Title excitation energies of SiH2
Atoms Zmatrix
 Si 0  0  0
 H  1  0  0  1.5145
 H  1  2  0  1.5145  92.68
End

excitations
  lowest 20
end

Basis
 Type TZ2P
End

End input
eor
```

The transition from the ground $^1A_1$ state to the excited $^1A_1$ state, which is an excitation from the electron configuration $(a_1)^2$ to $(b_1)^2$, can not be reached in this calculation, since it has mainly double excitation character. Of course, other excited $^1A_1$ states can be reached.

N_2: TDHF excitation energies

Sample directory: adf/N2_TDHF/

Calculation of the excitation energies of N_2 using time-dependent Hartree-Fock (TDHF). It also shows the possibility to use the Tamm-Dancoff approximation (TDA). This examples consists of 4 calculations:

- non-relativistic TDHF
- spin-orbit coupled TDHF
- non-relativistic TDHF, TDA
- spin-orbit coupled TDHF, TDA

The results will be inaccurate due to small basis set. The key ADDDIFFUSEFIT is required for a more accurate fit of the density.

```
$ADFBIN/adf << eor
Atoms
N 0 0 0
N 0 0 1.0977
End

XC
 hartreefock
end

dependency bas=1e-4
adddiffusefit

Basis
 Type DZ
 Core None
End

integration 10

excitations
 lowest 5
end
End Input
eor
```

In case of spin-orbit coupling one needs to include the key RELATIVISTIC with argument ZORA SPINORBIT. In practice one needs to calculate more excitations if one includes spin-orbit coupling, since the singlet-singlet and singlet-triplet excitations are not calculated separately, but will be treated simultaneously, since they may mix.

```
excitations
 lowest 20
end
relativistic spinorbit zora
```

The Tamm-Dancoff approximation (TDA) will be used if one includes the key TDA. The calculation is then effectively a CIS calculation.

```
TDA
```

$TiCl_4$: core excitation energies

Sample directory: adf/TiCl4_CoreExci/

Calculation of the 2p Ti and 2p Cl core excitation energies of $TiCl_4$

```
$ADFBIN/adf << eor
Title TiCl4 TD-DFT scalar relativistic 2p Ti core excitations
```

```
Units
 LENGTH BOHR
End

Atoms
 Ti 0.       0.        0.
 Cl    2.36754     2.36754     2.36754
 Cl   -2.36754    -2.36754     2.36754
 Cl    2.36754    -2.36754    -2.36754
 Cl   -2.36754     2.36754    -2.36754
End
SYMMETRY T(D)

EPRINT
 eigval 1000 1000
End

XC
 GGA LB94
End

relativistic scalar zora

ModifyExcitation
 UseOccupied
  T2 2
 SubEnd
 UseScaledZORA
END

Excitation
 ONLYSING
 Davidson &
  T2 12
 SubEnd
End

Basis
 Type DZ
 Core None
End

end input
eor
```

In this example, the 12 lowest singlet-singlet excitation energies of T_2 symmetry are calculated, the dipole allowed excitations. This can also be achieved using the ALLOWED subkey in the key Excitation.

In this example only excitations from the $2t_2$-orbital are included (see the key MODIFYEXCITATION), an almost pure 2p core orbital of titanium. The orbital energies of the uninteresting other occupied orbitals are artificially changed to a large negative value (-1d6 hartree).

In the second example the 2p Cl core excitation energies of $TiCl_4$ are calculated. The difference between the first example in this one is mainly the use of the key MODIFYEXCITATION:

```
ModifyExcitation
 UseOccRange -8.0 -6.0
 UseScaledZORA
END
```

In this example only excitations from occupied orbitals are considered which have orbital energies between -8 and -6 hartree, namely the $5a_1$, 1e, $1t_1$, $4t_2$, and $5t_2$ orbitals, which are almost pure 2p core orbitals of chlorine. The orbital energies of the uninteresting other occupied orbitals are again artificially changed to a large negative value (-1d6 hartree).

Ne: (core) excitation energies including spin-orbit coupling

Sample directories: adf/Ne_exciso/ and adf/Ne_CoreExci/

Calculation of the (core) excitation energies of Ne including spin-orbit coupling.

```
$ADFBIN/adf << eor
Title Ne
Atoms
  Ne .0000  .0000  0.0000
End
Basis
 Type QZ4P
End
relativistic scalar zora
symmetry d(8h)
integration 6.0
xc
 model SAOP
end
Excitations
 lowest 10
End

 ModifyExcitation
   UseOccupied
    A1.g 1
   SubEnd
   UseScaledZORA
  END

End input
eor

mv TAPE21 Frag.t21
rm logfile TAPE21

$ADFBIN/adf << eor
Title Ne spin-orbit
```

```
Atoms
  Ne .0000  .0000  0.0000  f=Frag
End
relativistic spinorbit zora
symmetry d(8h)
xc
 model SAOP
end
integration 6.0
Excitations
 lowest 12
End

  ModifyExcitation
   UseOccupied
    E1/2.g 1
   SubEnd
   UseScaledZORA
  END

Fragments
 Frag Frag.t21
End
STCONTRIB
End input
eor
```

The difference between the core excitation calculation and the standard excitation is the extra subkey MODIFYEXCITATION in the core excitation calculation (in italic).

ADF can not handle ATOM and linear symmetries in excitation calculations. Therefore a subsymmetry is used, in this case symmetry d(8h).

A relatively large QZ4P basis set is used, which is still insufficient for excitations to Rydberg-like orbitals, one needs more diffuse functions.

The key STCONTRIB is used, which will give a composition of the spin-orbit coupled excitation in terms of singlet-singlet and singlet-triplet scalar relativistic excitations. In order to use the key STCONTRIB the scalar relativistic fragment should be the complete molecule.

In this case the key MODIFYEXCITATION takes care that only excitations from the occupied 1s-orbital (spinor) are included. In symmetry d(8H) the 1s-orbital (spinor) is of A1.g (E1/2.g) symmetry.

AgI: excitation energies including spin-orbit coupling perturbatively

Sample directory: adf/AgI_asoexcit/

Calculation of the excitation energies of AgI including spin-orbit coupling in a perturbative way.

```
$ADFBIN/adf << eor
Title [AgI]
```

```
Atoms
 Ag .0000  .0000  2.5446
  I .0000  .0000  0.0000
End
relativistic scalar zora
symmetry C(7v)
integration 6.0
Excitations
 lowest 60
End
SOPERT
Basis
 Type TZ2P
 Core None
End
eor
```

ADF can not handle ATOM and linear symmetries in excitation calculations. Therefore a subsymmetry is used, in this case symmetry C(7v).

A relatively small TZ2P basis set is used, which is not sufficient for excitations to Rydberg-like orbitals, one needs more diffuse functions.

The key SOPERT is included in scalar relativistic ZORA calculations of excitation energies. First scalar relativistic TDDFT calculations are performed to determine the lowest 60 singlet-singlet and singlet-triplet excited states and the spin-orbit coupling operator is applied to these single-group excited states to obtain the excitation energies with spin-orbit coupling effects included.

Hyperpol: Hyperpolarizabilities of He and H_2

Sample directory: adf/Hyperpol/

This sample illustrates the computation of (hyper) polarizability tensors for the He atom and the H_2 molecule.

The symmetry is specified, because the Response module in ADF cannot yet handle the infinite symmetries ATOM, C(lin), D(lin).

```
$ADFBIN/adf <<EOR
Title expt geometrie H2(VII),VWN
noprint sfo,frag,functions

Symmetry C(8v)

Atoms
  H 0 0 -0.37305
  H 0 0  0.37305
End

Fragments
  H t21.H7
End
```

```
Response
  HyperPol 0.03
  DynaHyp
  AllComponents
End

EField 0 0 0.001

end input
EOR
```

The Response data block specifies (AllComponents) that not only the (default) zz-dipole polarizability is to be computed, but the complete tensor. The subkey HyperPol instructs the program to compute *hyper*polarizabilities and not only polarizabilities. The DynaHyp subkey implies that the *frequency-dependent* (hyper)polarizability is calculated. In that case the main laser frequency has to be specified, in hartree units, after the HyperPol subkey.

Only the first hyperpolarizability has been implemented in ADF. Some information on second hyperpolarizabilities can be obtained from the calculation of the first one in a finite field (EFIELD).

In similar fashion the frequency-dependent hyperpolarizability is computed for He, but only the zzz-component because now the AllComponents subkey is omitted.

```
$ADFBIN/adf <<EOR
Title hyperpolarizability He with the LB94 potential
noprint sfo,frag,functions

Atoms
  He 0 0 0
End

XC
  GGA LB94
END

Fragments
  He t21.He8
End

Response
  HyperPol 0.07
  DynaHyp
End

integration 5.0

EField 0 0 0.001

Symmetry C(8v)

end input
EOR
```

HF: Dispersion Coefficients

Sample directory: adf/Disper_HF/

General dispersion coefficients (beyond de dipole-dipole C6 interaction coefficient) are computed with the auxiliary program DISPER. It uses two output files from previous ADF Response calculations. In the example, the two ADF runs are one and the same and the relevant TENSOR output file is used twice.

```
$ADFBIN/adf <<EOR

title Van der Waals coefficients HF

atoms
H  0 0  -0.8708056087
F  0 0   0.04619439132
end

Basis
 Type DZP
 Core Small
End

symmetry C(8v)

RESPONSE
  MAXWAALS     8
  VANDERWAALS  7
  ALLTENSOR
  ALLCOMPONENTS
END

end input
EOR
```

Polarizabilities are computed at 7 (imaginary) frequencies between 0 and infinity. The program determines internally the actual frequency *values* in this range to use. The user only specifies the number of them, thereby determining the precision of, in fact, a numerical integration over the zero-infinity frequency range. A value of 7 is rather low.

MaxWaals determines that not only the C_6 but also C_7 and C_8 coefficients are computed. A value higher than 8 would not be recommended, because the available basis sets would be inadequate for higher coefficients.

In DISPER calculations the preparatory Response calculation *must* use the AllTensor and AllComponents subkeys.

The calculation produces a file TENSOR. The subsequent DISPER run uses two such files. In this example, both are taken from the same ADF run, copying the TENSOR file to, respectively, tensorA and tensorB. These names are prescribed for a DISPER calculation.

```
cp TENSOR tensorA
cp TENSOR tensorB
```

```
$ADFBIN/disper -n1  << eor
eor
```

The DISPER program needs no other input than just the files tensorA and tensorB, which must both be present as local files. Note the '-n1' flag: this enforces that a single-node (non-parallel) run is performed. The current implementation does not support parallelization of DISPER, because the kid processes may not have the (local to the master!) files tensorA and tensorB.

DMO: Circular Dichroism spectrum

Sample directory: adf/DMO_CD/

If the subkey CDSPECTRUM is included in the key EXCITATIONS, the rotatory strength is calculated for the calculated excitations, in order to calculate the CD (Circular Dichroism) spectrum. Only useful for chiral molecules.

With the VELOCITY keyword also the dipole-velocity representation of the rotatory strength is calculated.

Note: results will be physically meaningless due to small basis set. purpose of this job is to provide a test case for the CD implementation

Do not use less strict convergence criteria than default, better to use tighter criteria. The approximations in the evaluation of the integrals one makes with the linear scaling techniques are effectively switched off by setting LINEARSCALING 100 (recommended to use this).

Usage:

```
$ADFBIN/adf <<eor
TITLE dimethyloxirane excitations + CD

COMMENT
 results will be physically  meaningless due to small basis set.
 purpose of this job is to provide a test case for the CD implementation
END

XC
  gga becke perdew
END

Basis
 Type DZP
 Core Small
End

ATOMS
O       0.000129    1.141417    0.000023
C      -0.597040   -0.094320    0.428262
C       0.596952   -0.094328   -0.428223
H      -0.442927   -0.302650    1.487698
H       0.442944   -0.302474   -1.487720
C      -1.978779   -0.386617   -0.093924
```

```
H      -2.723275    0.220579    0.429114
H      -2.043506   -0.157697   -1.159810
H      -2.236045   -1.439970    0.055144
C       1.978716   -0.386693    0.093893
H       2.236030   -1.439985   -0.055498
H       2.723156    0.220701   -0.429005
H       2.043497   -0.158088    1.159845
END

linearscaling 100
excitations
 cdspectrum
 onlysinglet
 velocity
 lowest 10
end

END INPUT
eor
```

DMO: Optical Rotation Dispersion

Sample directory: adf/DMO_ORD/

If the subkey OPTICALROTATION is included in the key RESPONSE, the (frequency dependent) optical rotation is calculated.

Note: results will be physically meaningless due to small basis set. purpose of this job is to provide a test case for the ORD implementation

Do not use less strict convergence criteria than default, better to use tighter criteria. The approximations in the evaluation of the integrals one makes with the linear scaling techniques are effectively switched off by setting LINEARSCALING 100 (recommended to use this).

Usage:

```
$ADFBIN/adf <<eor
TITLE dimethyloxirane excitations + ORD

COMMENT
 results will be physically  meaningless due to small basis set.
 purpose of this job is to provide a test case for the ORD implementation
END

XC
  gga becke perdew
END

Basis
 Type DZP
 Core Small
End
```

```
ATOMS
O       0.000129    1.141417    0.000023
C      -0.597040   -0.094320    0.428262
C       0.596952   -0.094328   -0.428223
H      -0.442927   -0.302650    1.487698
H       0.442944   -0.302474   -1.487720
C      -1.978779   -0.386617   -0.093924
H      -2.723275    0.220579    0.429114
H      -2.043506   -0.157697   -1.159810
H      -2.236045   -1.439970    0.055144
C       1.978716   -0.386693    0.093893
H       2.236030   -1.439985   -0.055498
H       2.723156    0.220701   -0.429005
H       2.043497   -0.158088    1.159845
END

linearscaling 100
response
 allcomponents
 opticalrotation
end
END INPUT
eor
```

DMO: Optical Rotation Dispersion, lifetime effects (key AORESPONSE)

Sample directory: adf/DMO_ORD_aoresponse/

If the subkey OPTICALROTATION is included in the key AORESPONSE, the (frequency dependent) optical rotation is calculated. In this example lifetime effects are included. This test example consists of two ORD calculations: one with and one without the velocity gauge.

Note: results will be physically meaningless due to small basis set. purpose of this job is to provide a test case for the ORD implementation

Usage:

```
$ADFBIN/adf <<eor
TITLE dimethyloxirane, ORD
COMMENT
 results will be physically  meaningless due to small basis set.
 purpose of this job is to provide a test case for the ORD implementation
END
XC
  gga becke perdew
END
Basis
 Type DZP
 Core Small
```

```
End
ATOMS
O       0.000129    1.141417    0.000023
C      -0.597040   -0.094320    0.428262
C       0.596952   -0.094328   -0.428223
H      -0.442927   -0.302650    1.487698
H       0.442944   -0.302474   -1.487720
C      -1.978779   -0.386617   -0.093924
H      -2.723275    0.220579    0.429114
H      -2.043506   -0.157697   -1.159810
H      -2.236045   -1.439970    0.055144
C       1.978716   -0.386693    0.093893
H       2.236030   -1.439985   -0.055498
H       2.723156    0.220701   -0.429005
H       2.043497   -0.158088    1.159845
END
allpoints
aoresponse
 ALDA
 opticalrotation
 frequency 1 5893 angstrom
 scf iter 20
 lifetime 0.007
end
END INPUT
eor
```

In the second example the subkey OPTICALROTATION of the key AORESPONSE is changed into VELOCIYORD:

```
aoresponse
 ALDA
 VelocityORD
 frequency 1 5893 angstrom
 scf iter 20
 lifetime 0.007
end
```

Propene: damped Verdet constants

Sample directory: adf/DampedVerdet/

Specify the subkey MAGOPTROT in the AORESPONSE key to calculate the Verdet constant. Here it is specified together with the LIFETIME key, such that the real and imaginary part of the damped Verdet constant will be calculated.

```
$ADFBIN/adf <<eor
title Propene

ATOMS
C   0.867000     1.441800     3.000000
C   0.849400     2.777300     3.000000
```

```
C   2.115500     0.591200     3.000000
H  -0.088300     0.909000     3.000000
H  -0.085900     3.336500     3.000000
H   1.772400     3.363200     3.000000
H   2.737100     0.793300     2.115200
H   1.876900    -0.479100     3.000000
H   2.737100     0.793300     3.884800
END

basis
 type DZP
 core None
end

SCF
iterations 50
converge 1.0e-6 1.0e-3
mixing 0.2
lshift 0.0
diis n=10 ok=0.5 cyc=5 cx=5.0 cxx=10.0
END

symmetry nosym
allpoints

INTEGRATION 5.0
linearscaling 9

XC
 Model SAOP
END

noprint sfo

aoresponse
 scf  converge 1d-5 iterations 25
 frequency 1 0.2 Hartree
 ALDA
 giao
 magoptrot
 lifetime 0.007
end

end input
eor
```

H_2O: Verdet constants

Sample directory: adf/H2O_Verdet/

Specify the subkey MAGOPTROT in the AORESPONSE key to calculate the Verdet constant.

```
$ADFBIN/adf <<eor
title water

basis
 type TZP
 core None
end

atoms
O           0.000000    0.134692    0.000000
H           0.869763   -0.538741    0.000000
H          -0.869763   -0.538794    0.000000
end

symmetry nosym
allpoints

integration 6.0
linearscaling 99

xc
 lda vwn
 gga revPBE
end

aoresponse
 scf  converge 1d-6 iterations 25
 frequency 1 0.088558 Hartree
 ALDA
 giao
 magoptrot
end

end input
eor
```

H_2O: MCD

Sample directory: adf/H2O_MCD/

Example for the calculation of magnetic circular dichroism (MCD). If the subkey MCD is included in the key EXCITATIONS the MCD parameters of the calculated excitations are calculated (A and B terms). The keys RELATIVISTIC ZORA and SOMCD are required for a calculation of temperature-dependent C terms. The key ALLPOINTS is required for an MCD calculation (if the molecule has symmetry).

```
$ADFBIN/adf <<eor
title water MCD

atoms
O 0 0 0
H 0 0 1
```

```
H 0 1 0
end

BASIS
TYPE DZP
end

ALLPOINTS

SOMCD

UNRESTRICTED

CHARGE 1 1

RELATIVISTIC ZORA

excitations
lowest 20
onlysinglet
mcd NMCDTERM=5
end

end input
eor
```

H_2O: static magnetizability

Sample directory: adf/H2O_magnet/

Basic example for a magnetizability calculation.

One should set iterations=0 for STATIC magnetizability. If one does not use SYMMETRY NOSYM, one should set use ALLPOINTS for correct results in AORESPONSE. If a line starts with :: it will be skipped during the reading of the input.

```
$ADFBIN/adf <<eor
title H2O magnetizability test
basis
 type DZP
 core None
end
units
 length bohr
 angle degree
end
atoms zmatrix
O 0 0 0 0.       0.     0.
H 1 0 0 1.808846 0.     0.
H 1 2 0 1.808846 104.52 0.
end
```

```
linearscaling 99
tails
xc
 lda
 gga revPBE
end
Comment
 New optiond fro AOResponse below
End
:: symmetry nosym
allpoints :: needed for correct results in AOResponse
AOResponse
ALDA
magneticpert :: needed for magnetizability
scf iterations 0  converge 1e-3 :: set iterations=0 for STATIC magnetizability
End
end input
eor
```

H_2O: dynamic magnetizability

Sample directory: adf/H2O_TD_magnet/

Example for time-dependent magnetizability with GIAOs (Gauge including atomic orbitals).

```
$ADFBIN/adf <<eor
basis
 type TZP
 core None
end
ATOMS
O           0.000000    0.134692    0.000000
H           0.869763   -0.538741    0.000000
H          -0.869763   -0.538794    0.000000
END
symmetry nosym
allpoints
integration 6.0
linearscaling 99
xc
 lda vwn
 gga revPBE
end
aoresponse
 scf  conv 1d-6 iter 25
 frequency 1 5893 Angstrom
 giao
 ALDA
 magneticpert
 FitAOderiv
end
```

```
END INPUT
eor
```

C_2H_4: Time-dependent current-density-functional theory

Sample directory: adf/C2H4_TDCDFT/

Calculation of excitation energies and response properties of C_2H_4, with the VK functional, thus using time-dependent current-density-functional theory.

```
$ADFBIN/adf << eor
title C2H4 excitation energy calculation with the VK functional

ATOMS
1. C 0.000000    0.000000    0.666318
2. C 0.000000    0.000000   -0.666318
3. H 0.000000    0.928431    1.239388
4. H 0.000000   -0.928431    1.239388
5. H 0.000000    0.928431   -1.239388
6. H 0.000000   -0.928431   -1.239388
END

BASIS
C $ADFHOME/atomicdata/ET/ET-pVQZ/C
H $ADFHOME/atomicdata/ET/ET-pVQZ/H
END

EXCITATIONS
END

CURRENTRESPONSE
END

endinput
eor
$ADFBIN/adf << eor
title C2H4 response calculation with the VK functional

ATOMS
1. C 0.000000    0.000000    0.666318
2. C 0.000000    0.000000   -0.666318
3. H 0.000000    0.928431    1.239388
4. H 0.000000   -0.928431    1.239388
5. H 0.000000    0.928431   -1.239388
6. H 0.000000   -0.928431   -1.239388
END

BASIS
TYPE TZ2P
END
```

```
RESPONSE
ALLCOMPONENTS
END

CURRENTRESPONSE
END

endinput
eor
```

NMR chemical shifts and spin-spin coupling constants

HBr: NMR Chemical Shifts

Sample directories: adf/HBr/ and adf/HBr_SO/

Computation of the NMR chemical shifts for HBr. The second sample uses spin-orbit relativistic corrections.

```
$ADFBIN/adf << eor
TITLE HBr non-relativistic

ATOMS
 1. H   .0000  .0000  .0000
 2. Br  .0000  .0000 1.4140
End

Basis
 Type DZ
 Core Large
End

XC
 GGA Becke Perdew
End

End input
eor
```

The TAPE21 result file of ADF must be present under that name for the NMR calculation

mv t21.nmr TAPE21

The NMR program uses only one input (block) key NMR, currently. The subkeys specify what output is produced (OUT) and for which Nuclei the NMR data are computed and printed (NUC). See the User's Guide and the Utilities document for more details.

```
$ADFBIN/nmr << eor
NMR
  Out  TENS
  Nuc  1  2
```

```
End
eor
```

The second run is like the first, except that it uses relativistic corrections, including Spin-Orbit terms. This implies that NOSYM symmetry *must* be used in the ADF calculation: the NMR program cannot handle symmetry calculations in combination with spin-orbit terms and will stop with an error message if you try to do so.

```
$ADFBIN/adf << eor
TITLE HBr relativistic spinorbit Pauli

Atoms
 1. H   .0000  .0000  .0000
 2. Br  .0000  .0000 1.4140
End

Basis
 Type DZ
 Core Large
End

Symmetry NoSYM

XC
 GGA Becke Perdew
End

Relativistic SpinOrbit Pauli

End input
eor

rm t12.rel

$ADFBIN/nmr << eor
 NMR
   OUT TENS
   NUC  1 2
 End
eor
```

HgMeBr: NMR Chemical Shifts

Sample directories: adf/HgMeBr_pnr/ (non-relativistic), adf/HgMeBr_psc/ (Pauli scalar relativistic), adf/HgMeBr_zso/ (ZORA relativistic *and* Spin-Orbit terms included)

NMR data are computed for the 1st and 3rd nucleus only. The UIK subkey is used to indicate that certain terms are to be included in the 'U-matrix', which goes into the first-order change of the MO's due to the applied magnetic field. See the documentation (Utilities) for more information.

The 'BEST' specification means that the mass-velocity and Darwin terms are included for a scalar relativistic calculation. In a non-relativistic run it has no meaning. In a spin-orbit run it would include the Fermi-contact term for the Pauli formalism, and the ZORA Spin-Orbit terms for a ZORA calculation.

```
$ADFBIN/nmr << eor
NMR
  U1K BEST
  NUC 1 3
END
eor
```

The other two calculations are similar, apart from the specification of the applicable relativistic features.

CH_4: NMR Chemical Shifts, SAOP potential

Sample directory: adf/CH4_SAOP/

Computation of the NMR chemical shifts for CH_4, with the model potential SAOP.

Important: use SAVE TAPE10. This is necessary for SAOP, since the nmr program does not know about SAOP or other model potentials. On TAPE10 the SCF potential is written, which is read in by the nmr program.

Note: For SAOP one needs an all-electron basis set

```
$ADFBIN/adf << eor
xc
 model saop
end

Define
 RCH = 1.085
 XCH = sqrt(3)*(RCH/3)
End

Atoms
 C  0 0 0
 H  XCH  XCH  XCH
 H  XCH -XCH -XCH
 H -XCH  XCH -XCH
 H -XCH -XCH  XCH
End

Basis
 Type TZ2P
 Core None
End

save TAPE10
End Input
eor
```

```
$ADFBIN/nmr << eor
NMR
  Out  TENS
  Nuc  1  2
End
eor
```

CO: NMR Chemical Shifts, SIC-VWN potential

Sample directory: adf/CO_fc_SICVWN/

Computation of the NMR chemical shifts for CO, with the SIC-VWN potential.

Important: use SAVE TAPE10. This is necessary for SIC-VWN or SAOP, since the epr or nmr program does not know about SIC-VWN or other model potentials. On TAPE10 the SCF potential is written, which is read in by the epr or nmr program.

Note: adf with the SIC-VWN only runs serial correctly, and symmetry NOSYM is required.

Note: Both epr and nmr change TAPE10, TAPE21. Therefore use original TAPEs from adf.

```
$ADFBIN/adf -n1 << eor
TITLE CO, SIC-VWN, Basis set TZ2P
Basis
 Type TZ2P
 Core Small
End
RELATIVISTIC Scalar Pauli
SYMMETRY NOSYM
ATOMS Z-mat
  1  C 0 0 0
  2  O 1 0 0  RCO
END
GEOVAR
  RCO    1.139719
END
INTEGRATION 6.0
XC
  LDA VWN
END
SICOEP
  IPRINT 1
  SELF 35
END
SAVE TAPE10
END INPUT
eor

cp TAPE21 t21
cp TAPE10 t10

$ADFBIN/epr -n1 << eor
```

```
NMRSHIELDING
  NUCLEI ALL
  OUTPUT
    SIZE LARGE
  SUBEND
END
SICOEP
  IPRINT 1
END
eor

rm TAPE10 TAPE15 TAPE21
cp t21 TAPE21
cp t10 TAPE10

$ADFBIN/nmr << eor
NMR
  U1K BEST
END
eor
```

PF_3: NMR Properties, Nucleus-independent chemical shifts

Sample directory: adf/Nmr_PF3/

Both the NMR program and the EPR/NMR program enables the calculation of so-called nucleus-independent chemical shifts. More details are available in the Properties Programs User's Guide.

In the ADF run, the Efield key is used to define points charges with zero charge. The GHOSTS key in the nmr or epr program then basically copies this block. For the interpretation of the results we refer to the literature.

```
...
Efield
   3.0 4.0 5.0 0.0
   1.0 2.0 3.0 0.0
End
...
eor
```

Example input for the EPR/NMR program.

```
$ADFBIN/epr -n1 << eor
CLGEPR
  NUCLEI ALL
  GHOSTS
     3.0 4.0 5.0
     1.0 2.0 3.0
  SUBEND
  OUTPUT
    SIZE LARGE
  SUBEND
```

```
END

END INPUT
eor
```

Example input for the NMR program.

```
$ADFBIN/nmr << eor
NMR
  Out Iso Tens
  GHOSTS
     3.0 4.0 5.0
     1.0 2.0 3.0
  SUBEND
END
END INPUT
eor
```

PF_3: Comparison of NMR with EPR/NMR

Sample directory: adf/PF3_nmr/

This example uses both the NMR program and the EPR/NMR program, which is somewhat different from the NMR program used in the examples above. Please check the Property Programs User's Guide for a discussion on the advantages and disadvantages of the two implementations. This example explicitly compares the two.

```
$ADFBIN/nmr << eor
NMR
  U1K BEST
END
eor
```

The NMR program can currently formally be run in parallel. Due to certain single-CPU bottlenecks, this is hardly noticeable at the moment though. For this reason, there is currently limited advantage in using more than one CPU for the nmr program. Most other property programs can currently not be run in parallel at all and require the -n1 flag.

The output of the two NMR calculations should be virtually identical.

```
$ADFBIN/epr -n1 << eor
CLGEPR
  NUCLEI ALL
  OUTPUT
    SIZE LARGE
  SUBEND
END

END INPUT
eor
```

PF_3: NMR with B3LYP

Sample directory: adf/NMR_B3LYP/

This example shows how to do hybrid calculation of NMR chemical shifts.

One needs of course a hybrid functional in the XC block key in ADF. One should also SAVE TAPE10, such that it is an input file in the *nmr* module.

```
$ADFBIN/adf << eor
title PF3-NMR-B3LYP
basis
 type DZP
 core None
end
Define
  RPF  =   1.641314
  AXPF = 119.702107
End
Atoms
P       0.00000000       0.00000000       1.00000000
F      -0.71283358       1.23466398       1.81325568
F      -0.71283358      -1.23466398       1.81325568
F       1.42566716       0.00000000       1.81325568
End
integration 6.0
noprint sfo
xc
 hybrid B3LYP
end
save TAPE10
end input
eor
rm logfile

$ADFBIN/nmr << eor
NMR
  Out  TENS
  Nuc  1  2
  SCF 1.d-4
End
eor
```

Next the same calculation is performed with the scalar relativistic ZORA Hamiltonian. In that case one should include in the ADF calculation.

```
RELATIVISTIC SCALAR ZORA
```

In the last example spin-orbit coupling is included. Symmetry should be NOSYM.

```
symmetry nosym
RELATIVISTIC SPINORBIT ZORA
```

In the input for the *nmr* module one can add the key ZSOAO2007 to approximate the effect of spin on the nucleus in the spin-orbit coupled calculations.

$VOCl_3$: NMR Chemical shifts

Sample directory: adf/Nmr_VOCl3/

After a scalar relativistic Pauli calculation in ADF, using NOSYM, the EPR/NMR program is invoked.The EPR/NMR program does not support ZORA at the moment.

```
$ADFBIN/adf << eor
TITLE VOCl3

CHARGE 0

ATOMS Z-mat
  1  V  0 0 0
  2  O  1 0 0  RVO
  3  Cl 1 2 0  RVCl AOVCl
  4  Cl 1 2 3  RVCl AOVCl  120.
  5  Cl 1 2 3  RVCl AOVCl -120.
END

INTEGRATION 5.0 5.0

GEOVAR
  RVO       1.584452
  RVCl      2.164767
  AOVCl   108.614124
END

SYMMETRY NOSYM

RELATIVISTIC Scalar Pauli

XC
   LDA VWN
END

FRAGMENTS
  V  t21.V
  O  t21.O
  Cl t21.Cl
END

COREPOTENTIALS t12.rel &
  V  1
  Cl 2
  O  3
END
```

```
END INPUT
eor
```

The NUCLEI key now specifies that all atoms are to be treated for NMR. Much output is demanded.

```
$ADFBIN/epr -n1 << eor
CLGEPR
  NUCLEI ALL
  OUTPUT
    SIZE LARGE
  SUBEND
END
END INPUT
eor
```

C_2H_2: NMR Spin-spin coupling constants

Sample directory adf/CPL_C2H2

Nonrelativistic calculation

A calculation of NMR nuclear spin-spin coupling constants (NSCCs).

As explained in the 'ADF Properties Programs' documentation, the quality of a calculation for spin-spin coupling constants, using the program 'CPL', depends largely on the preceding ADF calculation, which produces the Kohn-Sham orbitals and orbital energies, used as a starting point.

One of the quality-determining factors is the chosen basis set. It should be sufficiently flexible near the nucleus. Although the all-electron basis TZ2P is chosen in this example, it is recommendable to add more functions to the basis and fit sets near the nucleus in case of heavy elements. One could start from a ZORA/QZ basis for example.

The integration accuracy in the ADF calculation is chosen such that the region near the nuclei is described relatively more accurately than the rest of the molecule.

```
INTEGRATION
accint 5
accsph 6
end
```

The NOSYM symmetry currently needs to be specified in ADF to enable the CPL program to work correctly.

The first call to cpl is as follows:

```
$ADFBIN/cpl <<eor
nmrcoupling
dso
pso
sd
scf convergence 1e-7
nuclei 1 2 3 4
nuclei 3 4
```

```
end
endinput
eor
```

The CPL program can run in parallel.

The specification of what needs to be calculated is given in the nmrcoupling block key.

In this first example, the SD subkey is left out, as this would lead to a very strong increase in the required CPU time. The SD subkey is included in the second CPL run. That subkey controls the calculation of the so-called spin-dipole term.

The subkeys dso and pso specify that, respectively, the diamagnetic and paramagnetic orbital terms will be calculated. The often dominant Fermi contact term (FC) is calculated by default and therefore does not have to be specified explicitly.

The scf convergence subkey, in this context, refers to the convergence for the solution of the coupled-perturbed Kohn-sham equations which need to be solved to obtain to spin-spin couplings.

The lines

```
nuclei 1 2 3 4
nuclei 3 4
```

that one coupled-perturbed Kohn-Sham calculation is performed where nucleus number 1 (according to the ordering in the ADF output) is the perturbing nucleus, and nuclei 2, 3, and 4 are the perturbed nuclei, and another coupled-perturbed Kohn-Sham calculation is performed where nucleus 3 is the perturbing nucleus and nucleus 4 is the perturbed nucleus.

The second CPL run also includes the spin-dipole (SD) term, through the SD subkey.

The output of the CPL program first contains a lot of general information, a summary of the specified input, and then produces the desired numbers:

It prints separately the different contributions (FC, DSO, PSO, SD) if specified in input and sums them up to a total number. Experimental NSCCs between two nuclei A and B are usually reported as J(A,B) in Hertz. From a computational point of view, the so-called reduced NSCCs K(A,B) are more convenient for comparisons. CPL outputs both. In this example, the Fermi-contact term is indeed dominant.

The first part of the output refers to the line

```
nuclei 1 2 3 4
```

then the same thing is done for the second similar line where nucleus 3 is the perturbing nucleus.

The output for the second CPL run looks very similar, but now the SD term is added to the Fermi contact term, resulting in much longer execution times.

Scalar relativistic and spin-orbit calculations

The CPL program also enables calculations using scalar relativistic effects (ZORA) and/or spin-orbit effects.

Schematically, this requires the following changes to the input file with respect to a regular spin-orbit calculation and a nonrelativistic CPL calculation:

- steep (1s) functions may need to be added to the standard basis sets.
- the full-potential option for ZORA is needed in the create runs and all further runs:
  relativistic zora scalar full
- the molecular ADF calculation should contain the line
  relativistic zora full spinorbit
- the CPL input is unmodified with respect to the example given here. Please check the 'ADF Property Programs' document for details on relativistic input options.

HF: NMR Spin-spin coupling constants, hybrid PBE0

Sample directory adf/CPL_HF_hybrid

A calculation of NMR nuclear spin-spin coupling constants (NSCCs) for the hybrid PBE0.

The hybrid PBE0 is chosen as exchange-correlation potential in the ADF calculation. The key 'usespcode' is required for consistency reasons of the PBE0 implementation in ADF and the kernel that is used in the 'CPL' program, that calculates NMR spin-spin coupling constants. Symmetry should be NOSYM. The basis sets used are specially optimized all-electron basis sets for NMR spin-spin coupling calculations (in the directory $ADFHOME/atomicdata/ZORA/jcpl), which have extra tight functions, compared to a default ADF basis set. The integration accuracy is extra high in the core region.

```
$ADFBIN/adf <<eor
UNITS
    length Angstrom
    angle Degree
END
:: experimental bond length
ATOMS
F 0.0000 0.0000 0.0000
H 0.0000 0.0000 0.9170
END
BASIS
Type ZORA/jcpl
Core None
END
usespcode
XC
 hybrid PBE0
END
SYMMETRY nosym
SCF
converge 1.0e-8 1.0e-6
END
INTEGRATION
 accint 6.0
 accsph 7.5
end
scf
 converge 1e-8 1e-7
end
```

```
end input
eor
```

The first call to cpl is as follows:

```
$ADFBIN/cpl <<eor
gga
nmrcoupling
dso
pso
scf convergence 1e-6 iterations 20
 nuclei 1 2
end
endinput
eor
```

The key 'gga' is included to use the first-order GGA potential instead of the first-order VWN potential. The Hartree-Fock part of the kernel is included automatically if a hybrid potential is used in the ADF calculation.

The second CPL run also includes the spin-dipole (SD) term, through the SD subkey, which is much more time-consuming.

```
sd
```

ESR / EPR properties

TiF_3: ESR g-tensor, A-tensor, Q-tensor

Sample directory: adf/ESR_TiF3/

You calculate Electron Spin Resonance properties with the keywords ESR and QTENS. ESR is a block-type key and is used to compute the G-tensor or the Nuclear Magnetic Dipole Hyperfine interaction. QTENS is a simple key and invokes the computation of the Nuclear Electric Quadrupole Hyperfine interaction.

Proper usage of the key ESR requires that you do one of the following:

(a) A Spin-Orbit calculation, spin-restricted, with exactly one unpaired electron, or
(b) A Spin-Orbit calculation, spin-unrestricted in the collinear approximation, or
(c) No Spin-Orbit terms and spin-*un*restricted.

In case (a) and (b) you obtain the G-tensor. In case (b) and (c) you get the Magnetic Dipole Hyperfine interaction.

Note: in case (a) the program also prints a Magnetic Dipole Hyperfine interaction data, but these have then been computed without the terms from the spin-density at the nucleus.
Note: in case (b) and (c) one can have more than one unpaired electron.
Note: in case (b) one has to use symmetry NOSYM.

Five calculations are performed:

- Scalar relativistic spin-unrestricted
- Spin-Orbit relativistic spin-restricted

- Scalar relativistic spin-restricted
- Scalar relativistic open shell spin-restricted
- Spin-Orbit relativistic spin-unrestricted collinear

After the preliminary calculations (DIRAC, to get the relativistic TAPE12 file with relativistic potentials, and the Create runs), we first calculate the Dipole Hyperfine interaction: a spin-unrestricted calculation without Spin-Orbit coupling.

```
$ADFBIN/adf << eor
title  TiF3  relativistic open shell unrestricted
noprint sfo,frag,functions

DEFINE
 RTIF = 1.780
 RY  = RTIF*SQRT(3)/2
END

esr
end

qtens

atoms
  Ti    0    0   0
  F   RTIF   0   0
  F  -RTIF/2  RY 0
  F  -RTIF/2 -RY 0
end

fragments
  Ti   t21.ti
  F    t21.f
end

xc
  GGA Becke Perdew
end

charge 0 1
unrestricted

relativistic scalar zora
Corepotentials  t12.rel &
  Ti 1
  F 2
end

end input
eor
```

Then, for the same molecule, we compute the G-tensor in a Spin-Orbit run (spin-restricted).

The here-computed and printed Dipole Hyperfine interaction misses the terms from the spin-density at the nucleus: compare with the outcomes from the first calculation.

In each of the calculations, the QTENS key invokes the computation of the Electric Quadrupole Hyperfine interaction.

Note that an *all-electron* calculation is carried out. This is relevant for the computation of the A-tensor, the nuclear magnetic dipole hyperfine interaction, where an accurate value of the spin-polarization density at the nucleus is important. For the G-tensor (and also for the Q-tensor) this plays a minor role, but for reasons of consistency both calculations use the same basis set and (absence of) frozen core.

```
$ADFBIN/adf << eor
title  TiF3  relativistic spinorbit open shell restricted
noprint sfo,frag,functions

DEFINE
  RTIF = 1.780
  RY  = RTIF*SQRT(3)/2
END

esr
end

qtens

atoms
  Ti    0    0   0
  F   RTIF   0   0
  F  -RTIF/2  RY 0
  F  -RTIF/2 -RY 0
end

fragments
  Ti   t21.ti
  F    t21.f
end

xc
  GGA Becke Perdew
end

relativistic spinorbit zora
Corepotentials  t12.rel &
  Ti 1
  F 2
end

end input
eor
```

Next a scalar relativistic spin-restricted calculation is performed. The TAPE21 of this calculation is saved as a fragment in the next spin-unrestricted calculation, using only 1 SCF iteration, which is a way to get the scalar relativistic spin-restricted open shell result for the magnetic dipole hyperfine interaction.

```
$ADFBIN/adf << eor
title  TiF3  scalar relativistic restricted
noprint sfo,frag,functions
```

```
DEFINE
 RTIF = 1.780
 RY  = RTIF*SQRT(3)/2
END

atoms
  Ti    0    0   0
  F   RTIF   0   0
  F  -RTIF/2  RY 0
  F  -RTIF/2 -RY 0
end

fragments
  Ti   t21.ti
  F    t21.f
end

xc
  GGA Becke Perdew
end

relativistic scalar zora
Corepotentials  t12.rel &
  Ti 1
  F 2
end

end input
eor

mv TAPE21 t21.TiF3
rm logfile

$ADFBIN/adf << eor
title  TiF3  scalar relativistic open shell restricted
noprint sfo,frag,functions

DEFINE
 RTIF = 1.780
 RY  = RTIF*SQRT(3)/2
END

esr
end

qtens

atoms
  Ti    0    0   0 f=TiF3
  F   RTIF   0   0 f=TiF3
  F  -RTIF/2  RY 0 f=TiF3
  F  -RTIF/2 -RY 0 f=TiF3
end
```

```
fragments
  TiF3   t21.TiF3
end

xc
  GGA Becke Perdew
end

charge 0 1
unrestricted

scf
 iter 0
end

relativistic scalar zora
Corepotentials  t12.rel &
  Ti 1
  F 2
end

end input
eor
```

Finally a spin-orbit coupled spin-unrestricted calculation is performed using the collinear approximation. Note that symmetry NOSYM is used.

```
$ADFBIN/adf << eor
title  TiF3  relativistic spinorbit open shell unrestricted collinear
noprint sfo,frag,functions

DEFINE
  RTIF = 1.780
  RY  = RTIF*SQRT(3)/2
END

esr
end

qtens

symmetry nosym
unrestricted
collinear

atoms
  Ti    0    0   0
  F   RTIF   0   0
  F  -RTIF/2  RY 0
  F  -RTIF/2 -RY 0
end

fragments
```

```
  Ti   t21.ti
  F    t21.f
end

xc
  GGA Becke Perdew
end

relativistic spinorbit zora
Corepotentials  t12.rel &
  Ti 1
  F 2
end

end input
eor
```

VO: collinear approximation, ESR g-tensor, A-tensor, Q-tensor

Sample directory: adf/VO_collinear/

The ESR parameters of VO are calculated with the collinear approximation for unrestricted Spin-Orbit coupled calculations. In this example the VO-molecule has three unpaired electrons.

You calculate Electron Spin Resonance properties with the keywords ESR and QTENS. ESR is a block-type key and is used to compute the G-tensor or the Nuclear Magnetic Dipole Hyperfine interaction. QTENS is a simple key and invokes the computation of the Nuclear Electric Quadrupole Hyperfine interaction.

Proper usage of the key ESR requires that you do one of the following:

(a) A Spin-Orbit calculation, spin-restricted, with exactly one unpaired electron, or
(b) A Spin-Orbit calculation, spin-unrestricted in the collinear approximation, or
(c) No Spin-Orbit terms and spin-*un*restricted.

In case (a) and (b) you obtain the G-tensor. In case (b) and (c) you get the Magnetic Dipole Hyperfine interaction.

Note: in case (a) the program also prints a Magnetic Dipole Hyperfine interaction data, but these have then been computed without the terms from the spin-density at the nucleus.
Note: in case (b) and (c) one can have more than one unpaired electron.
Note: in case (b) one has to use symmetry NOSYM.

Two calculations are performed:

- Scalar relativistic spin-unrestricted
- Spin-Orbit relativistic spin-unrestricted collinear

After the preliminary calculations (DIRAC, to get the relativistic TAPE12 file with relativistic potentials, and the Create runs), we first calculate the Dipole Hyperfine interaction: a spin-unrestricted calculation without Spin-Orbit coupling.

Note that one has to use ALLPOINTS in the calculation for a linear molecule to get results for the nuclear magnetic dipole hyperfine interaction.

```
$ADFBIN/adf << eor
Atoms
 V 0 0 0
 O 0 0 1.589
End

XC
 GGA Becke Perdew
End

esr
end
qtens

allpoints
unrestricted
charge 0 3

Relativistic  Scalar ZORA
CorePotentials t12.rel  &
 V 1
 O 2
End

integration 5

Fragments
 V t21.V
 O t21.O
End
End input
eor
```

Then a spin-orbit coupled spin-unrestricted calculation is performed using the collinear approximation. Note that symmetry NOSYM is used.

```
$ADFBIN/adf << eor
Atoms
 V 0 0 0
 O 0 0 1.589
End

XC
 GGA Becke Perdew
End

esr
end
qtens

symmetry nosym
```

```
unrestricted
collinear

Relativistic  Spinorbit ZORA
CorePotentials t12.rel  &
 V 1
 O 2
End

integration 5

Fragments
 V t21.V
 O t21.O
End
End input
eor
```

Ge^{+} and H_2^{+}: ESR g-tensor (epr program)

Sample directories: adf/Epr_Ge2+ and adf/Epr_H2+

The NMR/EPR program gives functionality that partially overlaps and partially differs from the ESR keyword inside ADF. Please check the ADF and Property Programs User's Guides for details.

In this example a scalar relativistic Pauli calculation is performed (ZORA is not implemented in this program).

The preparatory ADF calculation can run parallel, the EPR module should run serial. The ADF calculation should not use symmetry. It uses a high numerical integration accuracy. The revised PBE functional is invoked. The implementation allows spin-unrestricted high-spin inputs and as does the ESR implementation within ADF itself.

The H2+ example is very similar and in fact a but simpler, so it will not be discussed separately here.

```
$ADFBIN/adf << eor

TITLE Ge2+, scf

SYMMETRY NOSYM

UNRESTRICTED

CHARGE +1 3

ATOMS Cart
  1  Ge  0.0000 0.0000 -1.2344
  2  Ge  0.0000 0.0000  1.2344
END

INTEGRATION 6.0
```

```
FRAGMENTS
  Ge  t21.Ge
END

COREPOTENTIALS t12.rel &
  Ge 1
END

XC
  GGA revPBE
END

RELATIVISTIC Scalar Pauli

END INPUT
eor
```

The EPR calculation itself then has a fairly simple input. It uses the TAPE21 file generated by ADF. The full EPR G-tensor is printed, including an extensive analysis for the contribution of different terms.

The input line

NUCLEI NONE

implies that no NMR calculation is requested.

```
$ADFBIN/epr -n1 << eor
CLGEPR
  EPRGT
  SUBEND
  NUCLEI NONE
  OUTPUT
    EPRSIZE LARGE
  SUBEND
END
END INPUT
eor
```

NF_2: spin-other-orbit contribution g-tensor

Sample directory: adf/EPR_SOO

In this example the spin-other-orbit contribution to the g-tensor G(SOO) is calculated with the EPR module.

The NMR/EPR program gives functionality that partially overlaps and partially differs from the ESR keyword inside ADF. Please check the ADF and Property Programs User's Guides for details. The spin-other-orbit contribution to the g-tensor is not something that can be calculated with the ESR option inside ADF.

The preparatory ADF calculation can run parallel, the EPR module should run serial. The ADF calculation should use symmetry NOSYM, and an all electron basis set is required for the calculation of the G(SOO) term. TAPE10 is saved, which is necessary if special XC-potentials are used, like SIC or SAOP.

```
$ADFBIN/adf << eor
Unrestricted
Charge 0 1
Atoms
  N       0.000000      0.000000      0.611280
  F       0.000000     -1.090100     -0.237720
  F       0.000000      1.090100     -0.237720
END
XC
  GGA Becke Perdew
END
Basis
 Type DZ
 Core None
End
Symmetry NOSYM
SAVE TAPE10, TAPE15
eor
```

The EPR calculation itself then has a fairly simple input. It uses the TAPE21, TAPE10, and TAPE15 files generated by ADF. The full EPR G-tensor is printed, including an extensive analysis for the contribution of different terms, including the (small) spin-other-orbit contribution to the g-tensor G(SOO).

The spin-other-orbit contribution to the g-tensor G(SOO) is calculated by including the subsubkey SOO in the subkey EPRGT of the key CLGEPR. Note that the calculation can take very long, if one uses a larger basis set, and a better integration grid. Both are in fact necessary for reliable results.

```
$ADFBIN/epr -n1 << eor
CLGEPR
  EPRGT
    SOO
  SUBEND
  NUCLEI NONE
  OUTPUT
    EPRSIZE LARGE
  SUBEND
END
END INPUT
eor
```

EFG, Mössbauer

Ferrocene: Mössbauer spectroscopy

Sample directory: adf/Mossbauer/

By default in ADF the electron density at the nuclei is calculated, no input key is required. The electron density at the nuclei could be used for the interpretation of isomer shifts in Mössbauer spectroscopy. The absolute electron density at a nucleus heavily depends on the accuracy of the basis set in the core region of this nucleus, especially if relativistic effects are included. Important is to use the same basis set, same exchange correlation functional, same integration accuracy, if electron densities at nuclei in different

molecules are compared. For the calculation of Mössbauer quadrupole splittings the key QTENS is required. For ^{57}Fe quadrupole splittings will be written in units of mm/s, used in Mössbauer spectroscopy Example shows a calculation on ferrocene with a non-relativistic, and two with a scalar relativistic ZORA Hamiltonian using a different all electron basis set.

First a non-relativistic calculation.

```
$ADFBIN/adf << eor
title ferrocene

Atoms
    FE        0.000000    0.000000    0.000000
    C         1.215650    0.000000    1.600813
    C         0.375656   -1.156152    1.600813
    C        -0.983481   -0.714541    1.600813
    C        -0.983481    0.714541    1.600813
    C         0.375656    1.156152    1.600813
    C         1.215650    0.000000   -1.600813
    C         0.375656    1.156152   -1.600813
    C        -0.983481    0.714541   -1.600813
    C        -0.983481   -0.714541   -1.600813
    C         0.375656   -1.156152   -1.600813
    H         2.310827    0.000000    1.629796
    H         0.714085   -2.197727    1.629796
    H        -1.869498   -1.358270    1.629796
    H        -1.869498    1.358270    1.629796
    H         0.714085    2.197727    1.629796
    H         2.310827    0.000000   -1.629796
    H         0.714085    2.197727   -1.629796
    H        -1.869498    1.358270   -1.629796
    H        -1.869498   -1.358270   -1.629796
    H         0.714085   -2.197727   -1.629796
End

xc
 gga blyp
end

Basis
     Type TZP
     Core none
End

qtens

Integration 6
exactdensity


End Input
eor
```

Next the scalar relativistic ZORA calculations. ADF will also calculate the quadrupole splittings including the small component density, also called SR ZORA-4. The only difference is the RELATIVISTIC keyword:

```
relativistic scalar zora
```

Next a scalar relativistic calculation is performed with a much larger basis set (QZ4P) in the core region. Changing the basis set will have a large effect on the electron density at the nucleus and a noticeable effect on the calculated quadrupole splittings.

```
Basis
     Type QZ4P
     Core none
End
relativistic scalar zora
```

Analysis

Fragment orbitals and bond energy decomposition

$Ni(CO)_4$: Compound Fragments

Sample directory: adf/Frags_NiCO4/

An illustration of the fragment feature of adf

A transition metal complex is built from a Nickel atom and four CO fragments. The outcomes allows for an analysis (of molecular orbitals and the Bonding energy) in terms of the fragment properties. It is a Single Point calculation. Geometry optimization would not have been possible in this set-up because an optimization requires that only single-atom fragments are used.

The three atoms are created first: C, O, and Ni. For Carbon and Oxygen a type-DZ basis set is used (double-zeta) using the Basis key, while Ni gets a type-TZP basis (triple-zeta plus polarization).

CO

The CO molecule, to serve as a fragment template in $Ni(CO)_4$, is computed from the atomic fragments C and O. The coordinate values (atoms) are in bohr, rather than in Angstrom because the unit-of-length is redefined by the key units with subkey length.

The key scf is used to specify a somewhat tighter convergence criterion than the default, just to illustrate how to do this (normal settings are quite adequate).

The TAPE21 result file is renamed *t21.CO*.

```
$ADFBIN/adf <<eor
title CO (as fragment for NiCO4)

SCF
converge  1e-8
end

EPRINT
SFO eig ovl
END

units
length  bohr
end

atoms
C   0  0  0
O   0  0  2.15617844
end

Basis
```

```
  Type DZ
  Core Small
End

endinput
eor

mv TAPE21 t21.CO
```

Starting from ADF2008.01 one needs to include the subkey SFO of the key EPRINT with arguments eig and ovl in order to get the SFO MO coefficients and SFO overlap matrix printed on standard output.

Main calculation

Apart from the title, the input contains comment. This does not specify computational parameters but is only echoed in the output header, similar to the title. Contrary to the title, however, such comments are not preserved, apart from their echo in output and they are not written to TAPE21 or any other result file.

The atomic coordinates (atoms) are given in bohr (Units). To supply the numerical values use is made of user-defined constants (define): xyzC and xyzOx. This is convenient and it prevents typing errors when several coordinate values are identical, in particular when they carry a lot of decimal places.

The Atoms records contain also a specification of the fragments to which the respective atoms belong: four different CO fragments. No fragment is specified for the Ni atom, which implies that it is a fragment on its own.

The numbers at the very left of the records (1 through 9, with (optionally) a period after them), have no relevance. You can set them for ease of reference or counting.

```
$ADFBIN/adf <<eor
title Ni(CO)4,  from fragments Ni and CO

COMMENT
No geometry optimization possible, because not all fragments
are single atoms
END

units
 length  bohr
end

EPRINT
SFO eig ovl
END

DEFINE
xyzC=2.0053211
xyzOx=3.2501913
END

atoms
1. Ni   0       0        0
2. C    xyzC    xyzC     xyzC     f=CO/1
3. C   -xyzC   -xyzC     xyzC     f=CO/2
```

```
4. C    xyzC   -xyzC    -xyzC     f=CO/3
5. C   -xyzC    xyzC    -xyzC     f=CO/4
6. O    xyzOx   xyzOx    xyzOx     f=CO/1
7. O   -xyzOx  -xyzOx    xyzOx     f=CO/2
8. O    xyzOx  -xyzOx   -xyzOx     f=CO/3
9. O   -xyzOx   xyzOx   -xyzOx     f=CO/4
end

fragments
CO  t21.CO
Ni  t21.Ni
end

endinput
eor
```

$PtCl_4H_2^2$: Fragments again

Sample directory: adf/Frags_PtCl4H2

The (scalar) relativistic option (Pauli formalism) is used because of the presence of the relativistic Pt atom. The complex is built from fragments H_2 and $PtCl_4^2$.

Dirac: relativistic core potentials

The program *dirac* is applied to generate the corepotentials file for all involved atom types, including Hydrogen. The latter has no frozen core, let alone a relativistic one, but the corepotentials file also contains the total (relativistic) atomic potential. The (relativistic) atomic total potential is used in some types of relativistic options only, but it is a good idea to simply always run DIRAC for all the atoms whenever you do a relativistic calculation.

```
$ADFBIN/dirac <$adfresources/Dirac/Cl.2p
$ADFBIN/dirac <$adfresources/Dirac/Pt.4d
$ADFBIN/dirac <$adfresources/Dirac/H

mv TAPE12 t12.rel
```

The script above generates *one* TAPE12 file. The second and third dirac runs recognize the presence of the TAPE12 file (with the standard name 'TAPE12') that resulted from the earlier ones and they write their resulting data to the tail of it.

Basic atoms, non-default settings

```
$ADFBIN/adf <<eor
Create H file=$ADFRESOURCES/DZP/H
XC
  LDA  vwn
  GGA  becke perdew
End
```

```
Relativistic Scalar
CorePotentials  t12.rel &
  H   3
End
End Input
eor

mv TAPE21 t21H
```

The final calculations of the molecule and larger fragments are performed with gga ('NonLocal') xc corrections. Although it is not necessary to use the same settings in the Create runs, applying them looks 'nicer' and gives a better approximation of the bond energy of the molecule with respect to the basic atoms. Here it serves to show that also in a Create run various options can be used.

```
$ADFBIN/adf <<eor
create Cl file=$ADFRESOURCES/DZP/Cl.2p
xc
  lda vwn
  GGA  becke perdew
end

relativistic scalar
corepotentials  t12.rel  &
  Cl     1
end

end input
eor

mv TAPE21 t21Cl


$ADFBIN/adf <<eor
Create Pt file=$ADFRESOURCES/DZ/Pt.4d
XC
  lda vwn
  GGA  becke perdew
End

Relativistic scalar
CorePotentials  t12.rel  &
  Pt     2
End

End Input
eor

mv TAPE21 t21Pt
```

It is important to use the relativistic option in the creation of the fragments if the final molecule will use it as well. The corepotentials file is attached and the input indicates that the section on that file for Cl is #1, while the Pt data are in section #2.

Fragments H_2 and $PtCl_4^{2-}$

Now, all basic atoms have been generated. We go on to generate the two larger fragments H_2 and $PtCl_4^{2-}$ from which we are going to build the final complex.

```
$ADFBIN/adf <<eor
Title   H2  R=1.68a.u.
NoPrint sfo,frag,functions

Units
  length bohr
End

Atoms
H       0.0             0.0              0.84
H       0.0             0.0             -0.84
End

Fragments
H         t21H
End

XC
  lda vwn
  GGA  becke perdew
End

Relativistic Scalar
CorePotentials  t12.rel &
  H  3
End

End Input
eor

mv TAPE21 t21H2
```

The result file TAPE21 is renamed and saved to serve as fragment file.

```
$adf <<eor

title   PtCl4 (2-)
noprint sfo,frag,functions

units
length   bohr
end

ATOMS
Pt    0           0          0
Cl    4.361580    0.000000   0
Cl    0.000000    4.361580   0
Cl   -4.361580    0.000000   0
Cl    0.000000   -4.361580   0
```

```
end

fragments
Pt     t21Pt
Cl     t21Cl
end

xc
lda vwn
  GGA  becke perdew
end

relativistic scalar
corepotentials  t12.rel  &
Cl    1
Pt    2
end

charge  -2

end input
eor

mv TAPE21 t21PtCl4
```

The key charge is used to specify the net total charge. The default for the net total charge is the sum-of-fragment-charges. The fragments (Pt and Cl atoms) have been computed neutrally, but we want to calculate the $PtCl_4$ complex as a 2- ion.

Main calculation

Finally we compute $PtCl_4H_2^{2-}$ from the fragments $PtCl_4^{2-}$ and H_2.

```
$ADFBIN/adf <<eor
title   PtCl4 H2

units
length bohr
end

EPRINT
SFO eig ovl
END

integration 4.0

xc
lda vwn
  GGA  becke perdew
end

relativistic scalar
corepotentials  t12.rel  &
```

```
H     3
Cl    1
Pt    2
end

ATOMS
Pt   0             0             0             f=PtCl4
Cl   4.361580      0.000000      0.00000000    f=PtCl4
Cl   0.000000      4.361580      0.00000000    f=PtCl4
Cl  -4.361580      0.000000      0.00000000    f=PtCl4
Cl   0.000000     -4.361580      0.00000000    f=PtCl4
H    0.0           0.0           5.58          f=H2
H    0.0           0.0           7.26          f=H2
end

fragments
PtCl4     t21PtCl4
H2        t21H2
end

end input
eor
```

Note that, although the key charge is not supplied, the molecule is *not* neutral: the default charge (that is, omitting the keys charge, occupations) is the *sum-of-fragments*: the fragments here are H_2 and $PtCl_4^{2-}$, yielding a net charge for the molecule of minus two.

Note the f= fragment specification in the Atoms block. No fragment-numbering suffix (/n) is required because there is only one fragment of each fragment *type*.

H_2: Spin-unrestricted Fragments

Sample directory: adf/UnrFrag_H2

This is a small but important example to illustrate what goes into an accurate calculation of the 'true' bond energy of a molecule. The (ADF-specific) problem is that in a straightforward molecular calculation, the bond energy is computed as the energy difference between at the one hand the molecule, and at the other hand the isolated *spherically symmetric spin-restricted* atoms. The *italic*-typed features imply that the reference (comparison) state is usually not the physical ground state of the reference system (isolated atoms) and hence the computed energy difference has no direct relation to experimental data. To account for the true atomic ground states, one has to add correction terms. Study this sample carefully to make sure that you fully understand the steps to take and consult the User's Guide for details. See also the Theory document for a discussion of multiplet states.

See also the example, SD_Cr(NH_3)$_6$.

The H_2 case consists of a sequence of simple calculations to demonstrate the Unrestricted Fragments option. The energy difference between an unrestricted fragment as it is used in adf and a *self-consistent* unrestricted fragment is also computed. This turns out to be quite small, confirming that the adf approach, although not formally exact, is adequate for practical purposes.

```
$ADFBIN/adf <<eor
create H    file=$ADFRESOURCES/DZP/H
end input
eor

mv TAPE21 t21H

$ADFBIN/adf <<eor
title H unrestr., not self-consistent (as used in unr.frag. calcs)

scf
iterations 0  ! prohibit relaxation
end

unrestricted
charge 0  1   !  if not specified up and down electrons
!           will both get 0.5 electron: in fact restricted

fragments
H  t21H
end

atoms
H 0 0 0
end

endinput
eor

rm TAPE21 logfile
```

By setting the scf iterations to zero (a value of one (1) would give the same result) we prevent cycling to self-consistency. The energy of the 'final' one-electron orbitals is consequently computed in the start-up potential, i.e. the field of the restricted (basic) atom, where spin-α and spin-β are equally occupied, in this case by 0.5 electron each. The not-self-consistent, unrestricted H atom is precisely the 'unrestricted' fragment as it can be used in an adf calculation with unrestricted fragments. The fragment file must be the TAPE21 result file from a *restricted* run, but at start-up you can specify that the Fragment Orbitals are, for purposes of reference and comparison, occupied in an unrestricted way in the final molecule.

A calculation that uses *restricted* fragments right away computes the bonding energy relative to the restricted fragments. The difference between using restricted and unrestriced fragments is the 'bonding' energy computed in the run above.

```
$ADFBIN/adf <<eor
title  H  unr. self-consistent from unr.0

unrestricted
charge 0  1

fragoccupations
H
s  1 // 0
subend
end
```

```
Atoms
H  0 0 0
end

fragments
H         t21H
end

end input
eor

rm TAPE21 logfile
```

Here we start with the unrestricted fragment and relax to self-consistency. The 'bonding energy', i.e. the relaxation energy, is very small, demonstrating that using non-self-consistent unrestricted fragments involves only a small error (which, moreover, can be computed as shown here).

The key UnRestricted sets the spin-unrestricted mode. The key Charge is used to specify a net total charge of zero and a net total spin polarization by an excess of 1.0 spin-α electrons over spin-β.

```
$ADFBIN/adf <<eor
title   H2 restricted, from restricted fragments

ATOMS
H   0  0   0.375
H   0  0  -0.375
end

fragments
H   t21H
end

end input
eor

rm TAPE21 logfile
```

This is the simplest approach, using *restricted* fragments. The bonding energy must be corrected because the reference (restricted H atoms, with 0.5 electrons in spin-α and 0.5 in spin-β) is far from the true H-atom ground state: see the previous runs on the single H atom.

```
$ADFBIN/adf <<eor
title   H2 from unrestricted fragments

ATOMS
H.1   0   0   0.375
H.2   0   0  -0.375
end

fragments !  two different fragment types are necessary
!          because the two atoms get different FragOccupations
!          (see below), while the key FragOc.. addresses
!         only fragmentTYPES
```

```
H.1  t21H
H.2  t21H
end

charge  0

occupations
  sigma   2  !  specify the state (not always
             !  necessary)
end

fragoccupations
  H.1
    s  1 // 0
  subend
  H.2
    s  0 // 1
  subend
end

modifystartpotential
H.1  1 // 0    !  this helps SCF start-up
H.2  0 // 1    !  but is here not necessary
end

end input
eor

rm TAPE21 logfile
```

This should be a fair approximation (in the lda model) to the bonding energy of H_2 with respect to the unrestricted H atoms. The difference between the bonding energies of this and the previous run should be very close to the energy of the not-self-consistent unrestricted H-atom with respect to the restricted basic atom (calculation #2).

Excited state

```
$ADFBIN/adf <<eor
title   H2  excited

ATOMS
H    0 .0     0.375
H    0 .0    -0.375
end

fragments
H    t21H
end

fragoccupations
H
s  1 // 0
subend
```

```
end

unrestricted

charge 0  2

occupations
sigma.g   1 // 0
sigma.u   1 // 0
end

end input
eor
```

Finally the calculation of an excited state, with respect to unrestricted fragments. The excitation energy is obtained by comparing the energy with the energy of the ground state calculation. This difference compares reasonably, but not accurately, to the difference in one-electron ground state energies of the involved orbitals (Koopman's theorem).

Note that excitation energies can also be calculated with Time-Dependent DFT, using the RESPONSE module of ADF. See related sample runs.

PCCP: Bond Energy analysis open-shell fragments

Sample directory: adf/PCCP_Unr_BondEnergy/

This example illustrates advanced usage of the bond energy decomposition scheme used in ADF.

A proper decomposition of an electron-pair bond energy requires specifying opposite spins for the unpaired electrons of the respective radical fragments, which can be done with the input key FragOccupations. The specified alpha- and beta-spin configurations of the radical fragments are shown in the output section B U I L D.

Please note that if one neglects explicitly specifying opposite spins for the unpaired electrons of the fragments, each of them is treated as being half an alpha and half a beta electron and consequently, they enter into a spurious Pauli repulsive interaction. This results, amongst others, into the Pauli repulsion term being too repulsive and the orbital interaction term being too much stabilizing.

The example consists of an analysis of the C-C single bond between two CP radicals in the four-atomic molecule PCCP. The CP fragment calculations used to provide the TAPE21 for the overall PCCP calculation must be done, for technical reasons, in the restricted mode ("cp_fpccp_asr"). The proper spins are then specified in the calculation of the overall molecule using the FragOccupations key ("pccp_fa1_as"). Note that this implies a slight approximation because the bond energy computed in this way refers to the energy difference between closed-shell PCCP and two CP radicals that are described by orbitals from a spin-restricted SCF calculation, which have been given an unrestricted occupation. In other words, the set of alpha- and beta-spin orbitals are identical and the effect of spin polarization is missing. In practice, this leads to minor energy differences with respect to the correct bond energy, that is, the energy difference between closed-shell PCCP and two CP radicals treated in the unrestricted mode, i.e., for which the set of alpha- and beta-spin orbitals are allowed to relax toward different solutions in the SCF procedure. This correction term can be computed directly by carrying out

An unrestricted computation of the CP radical ("cp_fpccp_asu") using the restricted CP radical ("cp_fpccp_asr") as a fragment.

```
$ADFBIN/adf<<eor
TITLE cp_fpccp_asr

EPRINT
SFO eig ovl
END

XC
GRADIENTS  BECKE PERDEW
END

ATOMS
  C         .0000    .0000    .6681
  P         .0000    .0000   2.2555
END

FRAGMENTS
C   t21.C
P   t21.P
END

integration 5.0

END INPUT
eor

mv TAPE21 t21cp_fpccp

$ADFBIN/adf<<eor
TITLE cp_fpccp_asu

EPRINT
SFO eig ovl
END

XC
GRADIENTS  BECKE PERDEW
END

ATOMS
  C         .0000    .0000    .6681  f=CP
  P         .0000    .0000   2.2555  f=CP
END

FRAGMENTS
CP   t21cp_fpccp
END

UNRESTRICTED
CHARGE     0   1
```

```
integration 5.0

END INPUT
eor

rm TAPE21 logfile


$ADFBIN/adf<<eor
TITLE pccp_fa1_as

EPRINT
SFO eig ovl
ORBPOP  20  20
SUBEND
END

XC
GRADIENTS  BECKE PERDEW
END

ATOMS
  P         .0000    .0000   2.2555  f=CP_A
  C         .0000    .0000    .6681  f=CP_A
  C         .0000    .0000   -.6681  f=CP_B
  P         .0000    .0000  -2.2555  f=CP_B
END

integration 5.0

FRAGMENTS
CP_A   t21cp_fpccp
CP_B   t21cp_fpccp
END

SYMMETRY   C(LIN)

FRAGOCCUPATIONS
CP_A
  SIGMA 3//2
  PI    2//2
SUBEND
CP_B
  SIGMA 2//3
  PI    2//2
SUBEND
END

END INPUT
eor
```

TlH: Spin-Orbit SFO analysis

Sample directory: adf/TlH_SO_analysis/

Application of the Spin-Orbit relativistic option (using double-group symmetry) to TlH with a detailed analysis of the spinors in terms of SFOs (Symmetrized Fragment Orbitals).

In order to get the population analysis, one should have one scalar relativistic fragment, which is the whole molecule. The SFOs in this case are the scalar relativistic orbitals, which are already orthonormal, because one has only one fragment which is the whole molecule.

First the relativistic fragment is made, including the create of the atoms:

```
$ADFBIN/dirac -n1  < $ADFRESOURCES/Dirac/Tl
$ADFBIN/dirac -n1  < $ADFRESOURCES/Dirac/H
mv TAPE12 t12.rel

$ADFBIN/adf  <<eor
create Tl  file=$ADFRESOURCES/ZORA/TZ2P/Tl
xc
  LDA vwn
  GGA  becke perdew
end
relativistic scalar zora
corepotentials  t12.rel  &
Tl     1
H      2
end
end input
eor
mv TAPE21 t21.Tl

$ADFBIN/adf  <<eor
create H  file=$ADFRESOURCES/ZORA/TZ2P/H
xc
  LDA vwn
  GGA  becke perdew
end
relativistic scalar zora
corepotentials  t12.rel  &
Tl     1
H      2
end
end input
eor
mv TAPE21 t21.H


$ADFBIN/adf  <<eor
title   TlH, scalar relativistic zora

integration 6.0
```

```
relativistic scalar zora
corepotentials  t12.rel  &
Tl    1
H     2
end

ATOMS
Tl       0.0             0.0             0.0
H        0.0             0.0             1.870
end

fragments
Tl t21.Tl
H  t21.H
end

xc
  LDA vwn
  GGA  becke perdew
end

EPRINT
SFO eig ovl
END

end input
eor

mv TAPE21 t21.TlH
```

In order to get the population analysis, one should have one scalar relativistic fragment, which is the whole molecule, which is TlH in this case.

```
$ADFBIN/adf  <<eor
title   TlH from fragment TlH,  with SpinOrbit coupling

integration 6.0

relativistic spinorbit zora
corepotentials  t12.rel  &
Tl    1
H     2
end

ATOMS
Tl       0.0             0.0             0.0    f=TlH
H        0.0             0.0             1.870  f=TlH
end

fragments
TlH     t21.TlH
end
```

```
xc
  LDA vwn
  GGA  becke perdew
end

EPRINT
SFO eig ovl
END

end input
eor
```

The output gives something like:

```
 ========================
 Double group symmetry :  ***   J1/2  ***
 ========================

                                        ===  J1/2:1 ===

 Spinors expanded in SFOs
....
Spinor:             21              22              23              24
 occup:            1.00            1.00            1.00            0.00
 ------            ----            ----            ----            ----
 SFO SIGMA
   13.alpha:   0.7614+0.0000i   0.0096+0.0000i   0.0052+0.0000i  -0.0006+0.0000i
   14.alpha:   0.0154+0.0000i  -0.9996+0.0000i   0.0208+0.0000i  -0.0077+0.0000i
   15.alpha:  -0.0146+0.0000i   0.0185+0.0000i   0.9849+0.0000i   0.1625+0.0000i
 SFO PI:x
    8.beta :   0.4578+0.0000i   0.0091+0.0000i   0.0112+0.0000i   0.0030+0.0000i
    9.beta :   0.0005+0.0000i  -0.0074+0.0000i  -0.1119+0.0000i   0.6910+0.0000i
 SFO PI:y
    8.beta :   0.0000+0.4578i   0.0000+0.0091i   0.0000+0.0112i   0.0000+0.0030i
    9.beta :   0.0000+0.0005i   0.0000-0.0074i   0.0000-0.1119i   0.0000+0.6910i
....
```

Left out are a lot of small numbers. The meaning is that a spinor of J_z=1/2 symmetry can have SIGMA and PI character, for example, the 21st spinor with occupation number 1.0, is approximately
(21 J_z=1/2) = 0.76 (13 SIGMA alpha) + 0.46 (8 PI:x beta) + i 0.46 (8 PI:y beta)

Next in the SFO contributions per spinor the real and imaginary spin alpha part and real and imaginary spin beta part are all summed together to give a percentage of a certain SFO. are summed. For example the 21st spinor has almost 60% (13 SIGMA) character.

```
 SFO contributions (%) per spinor
 Spinor:      21     22     23     24
 occup:      1.00   1.00   1.00   0.00
 ------      ----   ----   ----   ----
 SFO SIGMA
     13:    57.97   0.01   0.00   0.00
     14:     0.02  99.92   0.04   0.01
     15:     0.02   0.03  97.01   2.64
 SFO PI:x
```

```
      8:    20.96   0.01   0.01   0.00
      9:     0.00   0.01   1.25  47.75
 SFO PI:y
      8:    20.96   0.01   0.01   0.00
      9:     0.00   0.01   1.25  47.75
```

Bader Analysis (AIM)

Sample directory: adf/Bader/

Starting from the ADF2008.01 version in ADF one calculate Bader atomic charges using a grid based method. This is described in this example. Another possibility for Bader's analysis is to use the *adf2aim* utility such that a third party program Xaim can be used.

With the BADER input key the ADF program will calculate Bader charges (AIM charges) using a grid based method.

```
$ADFBIN/adf <<eor
Title Calculate  Bader analysis for water

Atoms
    O         0.000000    0.000000   -0.001658
    H        -0.769048    0.000000    0.595209
    H         0.769048    0.000000    0.595209
End

Basis
     Type TZP
     Core none
End

Bader
End Input

eor
```

Next a similar calculation for ferrocene is given, which is not repeated here.

Bond Orders

Sample directory: adf/BondOrder/

With the key `BONDORDER` a bond order analysis is performed based on SFOs. Note that symmetry used in the calculation should be NOSYM. Shown here is only the example for benzene, where the bond orders calculated are with respect to the atomic fragments.

```
$ADFBIN/adf <<eor
title benzene BP/SZ bondorders tol=0.05
define
```

```
 cc=1.38476576
 ccc=120.0
 dih=0.0
 hc=1.07212846
 hcc= 120.0
 dih2=180.0
end
atoms Z-matrix
 C  0 0 0
 C  1 0 0  cc
 C  2 1 0  cc ccc
 C  3 2 1  cc ccc dih
 C  4 3 2  cc ccc dih
 C  5 4 3  cc ccc dih
 H  2 1 3  hc hcc dih2
 H  3 2 4  hc hcc dih2
 H  4 3 5  hc hcc dih2
 H  5 4 3  hc hcc dih2
 H  6 5 4  hc hcc dih2
 H  1 2 3  hc hcc dih2
end
basis
 Type SZ
 Core None
end
symmetry NOSYM
xc
  gga becke perdew
end
bondorder tol=0.05 printall
noprint sfo
eor
```

NOCV: ethylene -- Ni-diimina & H^+ -- CO

Sample directories: adf/Diimina_NOCV/ and adf/Hplus_CO_etsnocv

Example for calculation of ETS-NOCV for spin-restricted fragments. ETS-NOCV: energy analysis using the Natural Orbitals for Chemical Valence. The ethylene molecule and a Ni-diimina form a complex together. This example will be discussed first. The other example is H^+ and CO form together HCO^+, this example is similar to the discussed example. All electron basis sets are required.

First the two fragments are calculated.

```
$ADFBIN/adf << eor
Title: et-----Ni-diimina: ethylene run
atoms cartesian
C   -0.430177075 -1.815433265  0.860288229
C   -0.363705637 -1.910722338 -0.515633302
H    0.533109934 -2.284970854 -1.016904201
H   -1.279922499 -1.884673940 -1.115144723
```

```
H   -1.389295819 -1.753589602  1.377541080
H    0.440296224 -2.041861443  1.484489314
end
basis
 Type DZP
 Core Small
end
symmetry NOSYM
xc
 gga scf becke perdew
end
endinput
eor
mv TAPE21 t21.etfrag

$ADFBIN/adf << eor
Title: et-----Ni-diimina: Ni-diimina run
atoms cartesian
Ni   0.022615419  0.037783871  0.025751533
N    0.386170317  1.871072585  0.306265538
C    1.612863056  2.248007643  0.148716016
C    2.540686607  1.163409862 -0.183603690
N    1.976290003  0.008161589 -0.301176178
H   -0.288333328  2.609667211  0.546869047
H    1.942601454  3.283060847  0.269249237
H    3.613259273  1.338293482 -0.302134814
H    2.621707427 -0.766258151 -0.517479818
H   -1.351756655  0.253389698  0.386197419
end
charge 1
basis
 Type DZP
 Core Small
end
symmetry NOSYM
xc
  gga scf becke perdew
end
endinput
eor
mv TAPE21 t21.Nifrag
```

Next these fragments are used in the calculation of the full complex. The keys ETSNOCV and 'PRINT etslowdin' are needed in this case to to analyze the bonding in the molecule with respect to the fragments. The symmetry must be NOSYM.

```
$ADFBIN/adf << eor
Title: et-----Ni-diimina run
atoms
Ni   0.022615419  0.037783871  0.025751533 f=k
N    0.386170317  1.871072585  0.306265538 f=k
C    1.612863056  2.248007643  0.148716016 f=k
C    2.540686607  1.163409862 -0.183603690 f=k
N    1.976290003  0.008161589 -0.301176178 f=k
```

```
H   -0.288333328  2.609667211  0.546869047 f=k
H    1.942601454  3.283060847  0.269249237 f=k
H    3.613259273  1.338293482 -0.302134814 f=k
H    2.621707427 -0.766258151 -0.517479818 f=k
H   -1.351756655  0.253389698  0.386197419 f=k
C   -0.430177075 -1.815433265  0.860288229 f=m
C   -0.363705637 -1.910722338 -0.515633302 f=m
H    0.533109934 -2.284970854 -1.016904201 f=m
H   -1.279922499 -1.884673940 -1.115144723 f=m
H   -1.389295819 -1.753589602  1.377541080 f=m
H    0.440296224 -2.041861443  1.484489314 f=m
end
charge 1
fragments
m t21.etfrag
k t21.Nifrag
end
symmetry NOSYM
xc
 gga scf becke perdew
end
ETSNOCV
print etslowdin
endinput
eor
```

Next one could do *densf* calculations, to view the natural orbitals in this method, see also the the documentation for the *densf* analysis program and the ADF-GUI. Input is the TAPE21 of the molecular calculation.

```
$ADFBIN/densf << eor
GRID MEDIUM
NOCV
  THRESH 0.01
END
END INPUT
eor
mv TAPE41 nocv2.t41

$ADFBIN/densf << eor
GRID MEDIUM
NOCV
  ALL
END
END INPUT
eor
mv TAPE41 nocv3.t41
```

NOCV: CH_2 -- $Cr(CO)_5$

Sample directory: adf/NOCV_CrCO5-CH2/

Example for calculation of ETS-NOCV for spin-restricted fragments. ETS-NOCV: energy analysis using the Natural Orbitals for Chemical Valence. The CH_2 molecule and $Cr(CO)_5$ are the fragments, which form $Cr(CO)_5CH_2$ molecule.

First the two fragments are calculated.

```
$ADFBIN/adf -n1 << eor
Title CrCO5--[CH2] run from CrCO5 and CH2 closed shell fragments,FULL electron calc.!
atoms cartesian
    C        -0.429104    1.732058   -0.225052
    H         0.407023    2.440417   -0.352323
    H        -1.385325    2.281354   -0.254124
end
basis
 Type DZP
 Core None
end
symmetry NOSYM
xc
  gga becke perdew
end
endinput
eor
mv TAPE21 t21.CH2


$ADFBIN/adf -n1 << eor
Title [CrCO5] run
atoms cartesian
    Cr       -0.248053   -0.169062    0.005810
    C        -0.072963   -2.080685    0.229583
    O         0.030811   -3.223220    0.361925
    C        -0.182894    0.049840    1.909128
    O        -0.142780    0.212309    3.050403
    C        -0.299940   -0.409118   -1.894730
    O        -0.331795   -0.521589   -3.042336
    C        -2.138631   -0.242152    0.075713
    O        -3.295036   -0.249916    0.115045
    C         1.624487    0.092244   -0.083118
    O         2.763411    0.288575   -0.140976
end
basis
 Type DZP
 Core None
 Cr $ADFRESOURCES/TZP/Cr
end
symmetry NOSYM
xc
  gga becke perdew
end
endinput
eor
mv TAPE21 t21.Crfragment
```

Next these fragments are used in the calculation of the full complex. The keys ETSNOCV and 'PRINT etslowdin' are needed in this case to to analyze the bonding in the molecule with respect to the fragments. The symmetry must be NOSYM. Note the '-n1' flag for *adf*: this enforces that a single-node (non-parallel) run is performed.

```
$ADFBIN/adf -n1 << eor
Title:CrCO5--[CH2], etsnocv activated by etsnocv and print etslowdin
atoms
    C        -0.429104    1.732058   -0.225052 f=f1
    Cr       -0.248053   -0.169062    0.005810 f=f2
    C        -0.072963   -2.080685    0.229583 f=f2
    O         0.030811   -3.223220    0.361925 f=f2
    C        -0.182894    0.049840    1.909128 f=f2
    O        -0.142780    0.212309    3.050403 f=f2
    C        -0.299940   -0.409118   -1.894730 f=f2
    O        -0.331795   -0.521589   -3.042336 f=f2
    C        -2.138631   -0.242152    0.075713 f=f2
    O        -3.295036   -0.249916    0.115045 f=f2
    C         1.624487    0.092244   -0.083118 f=f2
    O         2.763411    0.288575   -0.140976 f=f2
    H         0.407023    2.440417   -0.352323 f=f1
    H        -1.385325    2.281354   -0.254124 f=f1
end
fragments
f1 t21.CH2
f2 t21.Crfragment
end
symmetry NOSYM
xc
  gga becke perdew
end
ETSNOCV RHOKMIN=1.e-3 EKMIN=1.5 ENOCV=0.05
print etslowdin
endinput
eor
```

NOCV: CH_3 -- CH_3

Sample directory: adf/CH3_CH3_etsnocv/

Example for calculation of ETS-NOCV for unrestricted fragments. ETS-NOCV: energy analysis using the Natural Orbitals for Chemical Valence. The ethane molecule is built from two methyl radicals

First the two methyl fragments are calculated. The fragments should be spin-restricted.

```
$ADFBIN/adf << eor
Title CH3-CH3 built from CH3 radicals,  FULL electron calc.!
atoms cartesian
    C         0.019664   -0.034069    0.009101
    H         0.039672   -0.069395    1.109620
    H         1.063205   -0.065727   -0.341092
    H        -0.474230   -0.953693   -0.341621
```

```
end
basis
H  $ADFRESOURCES/DZP/H
C  $ADFRESOURCES/DZP/C
end
symmetry NOSYM
SCF
  Iterations 2500
  Converge 1E-6
end
xc
  gga scf becke perdew
end
endinput
mv TAPE21 t21.frag1

$ADFBIN/adf << eor
Title CH3 radical
atoms cartesian
    C        -0.703210    1.217999   -0.497874
    H        -0.723753    1.252869   -1.598316
    H        -1.746567    1.250049   -0.147169
    H        -0.208833    2.137544   -0.147653
end
basis
H  $ADFRESOURCES/DZP/H
C  $ADFRESOURCES/DZP/C
end
symmetry NOSYM
SCF
  Iterations 2500
  Converge 1E-6
end
xc
  gga scf becke perdew
end
endinput
eor
mv TAPE21 t21.frag2
```

Next these fragments are used in the calculation of the molecule ethane, using the key FRAGOCCUPATIONS to use an unrestricted fragment occupation for the methyl radicals, such that they are prepared for bonding. In the one fragment the singly occupied orbital will be an α-orbital, and in the other fragment the singly occupied orbital will be a β-orbital, such that the calculated Pauli repulsion between the fragments will be small.

The keys ETSNOCV and 'PRINT etslowdin-unrestricted' are needed in this case to to analyze the bonding in a molecule with unpaired electrons in the fragments. The symmetry must be NOSYM.

```
$ADFBIN/adf << eor
Title: final [CH3]-[CH3], etsnocv activated by etsnocv and etslowdin-unrestricted
atoms
    C         0.019664   -0.034069    0.009101 f=f1
    H         0.039672   -0.069395    1.109620 f=f1
```

```
    H         1.063205   -0.065727   -0.341092 f=f1
    H        -0.474230   -0.953693   -0.341621 f=f1
    C        -0.703210    1.217999   -0.497874 f=f2
    H        -0.723753    1.252869   -1.598316 f=f2
    H        -1.746567    1.250049   -0.147169 f=f2
    H        -0.208833    2.137544   -0.147653 f=f2
end
fragments
 f1 t21.frag1
 f2 t21.frag2
end
fragoccupations
 f1
  A 5 // 4
 subend
 f2
  A 4 // 5
 subend
end
symmetry NOSYM
SCF
  Iterations 800
  Converge 1E-6
end
xc
  gga scf becke perdew
end
ETSNOCV RHOKMIN=1.e-3 EKMIN=1.5 ENOCV=0.05
PRINT etslowdin-unrestricted
end input
eor
```

Next 2 *densf* calculations, to view the natural orbitals in this method, see also the the documentation for the *densf* analysis program and the ADF-GUI. Input is the TAPE21 of the molecular calculation.

```
$ADFBIN/densf << eor
GRID MEDIUM
NOCV
 Alpha
  1
  2
  59
  60
 Beta
  1
  2
  59
  60
 END
END INPUT
eor
mv TAPE41 nocv1.t41
$ADFBIN/densf << eor
GRID MEDIUM
```

```
NOCV
  THRESH 0.01
END
END INPUT
eor
mv TAPE41 nocv2.t41
eor
```

Post-ADF analysis utilities

NO_2: Contour Plots using *Densf* and *Cntrs*

Sample directory: adf/Cntrs_NO2/

This example illustrates using the utility programs *cntrs* and *densf*. See the Utilities document for details.

```
$ADFBIN/adf << eor
title NO2

atoms
N     0          0          0
O     1.009356   0          0.464189
O    -1.009356   0          0.464189
end

Basis
 Type DZ
 Core Small
End

unrestricted
charge 0 1

endinput
eor
```

After the normal ADF calculation on NO_2 has been completed, the utility program *densf* is executed to generate a TAPE41 file with user-specified items evaluated in a regular, user-specified grid.

The TAPE21 on which *densf* operates must be present as a local file with name TAPE21.

```
$ADFBIN/densf << eor
density  scf ortho frag
fitdensity  scf ortho frag
orbitals  scf
alpha
a1     1 2
a2     1
b1     2
beta
a2     1
```

```
b1     1 2
b2     1
end
coulpot  frag ortho scf
grid
  -7.5  -7.5 0.0
51  51
1.0  0.0  0.0  15.00
0.0  1.0  0.0  15.00
end
end input
eor
```

The charge density values in the grid are requested for all available types of density: exact and fitted, for the initial (sum-of-fragments), intermediate (orthogonalized fragments, see the ADF User's Guide) and final (SCF) situation.

Several SCF molecular orbitals are computed by specifying their indices in the energy-ordered list (a separate list for each symmetry subspecies).

The coulomb potentials (again: for sum-of-fragments, orthogonalized fragments, and SCF) are generated.

The grid is defined by an 'origin', the numbers of points in all independent grid directions and the direction vectors with the total grid size in each direction separately.

Since there are only two 'numbers-of-points' (51 each) a 2-dimensional grid is generated. 1D and 3D grids are also possible. See the Utilities document for a more detailed survey of the available options.

The result of the *densf* run is a file tape41 (binary, KF). This contains all computed data. tape41 can be used by *cntrs* to generate plot data.

```
$ADFBIN/cntrs << eor
scan
0.02  0.05  0.10  0.2  0.5  0.0  -.02  -.05  -.10  -.2  -.5
end
dash 0.2
file cont.d
SCF%Density_A
SCF%Density_B
endinput
eor
```

In this example eleven (11) scan values are defined to draw contours for, with a dash length of 0.2 bohr.

An ascii file *cont.d* will be opened by *cntrs* on which the specified items (SCF-densities for spin-up and spin-down) will be combined (by default: simply added) into one quantity.

For this quantity the contour lines that correspond to the specified scan values are stored. See the Utilities document for precise specifications and options.

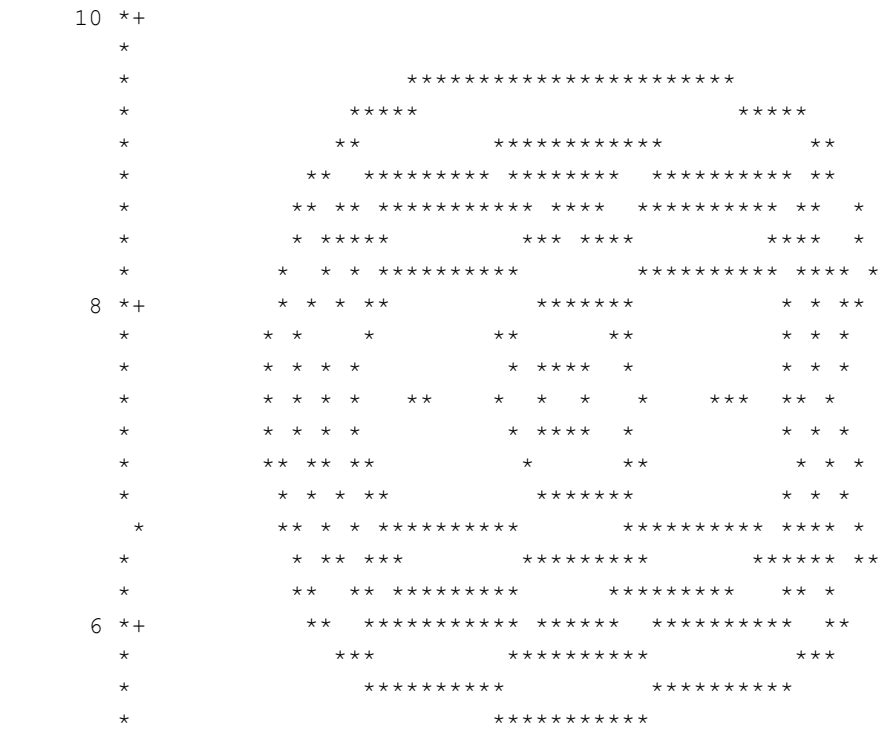
```
gnuplot << eor

set term dumb  100 80
set output "outplot"
plot "cont.d" using 1:2 with lines
```

```
eor

cat outplot
```

The public domain software *gnuplot* (not included in the adf package) is applied here to display the result from *cntrs*. The resulting picture on your screen (if you have *gnuplot* available) looks like



C_2H_2: Localization of Molecular Orbitals

*Sample direct*ory: adf/Cntrs.LocOrb_C2H2/

An *illustration* of the computation of localized molecular orbitals in C2H2.

The delocalized molecular orbitals as they result from the scf are localized in two different ways. In the first the three σ bonds are recombined only among themselves (no π bonds are mixed in), yielding two equivalent localized CH σ bonds and one localized σ bond. In a second step the localization of the remaining bond (the two π's) is performed, but this produces nothing new since no combination of the two π's is more localized then they are already by themselves.

```
$ADFBIN/adf << eor
title C2H2, localization Sigma and Pi separately

Atoms
C  0 0 .63
C 0 0 -.63
H 0 0 1.63
H 0 0 -1.63
END
```

```
Basis
 Type TZP
 Core Small
End

LocOrb
alfa  4 5
alfa  1 2 3
END

integration 4.0

end input
eor
```

In the first localization cycle the π-orbitals are left out: #4 and #5 in the list of all occupied valence MOs: first 3 MOs of the first irreducible representation (s), then the 2 from the second irrep (π). In the second localization step the first three (meanwhile localized) orbitals are kept aside.

With *densf* the local orbitals can be computed in a user-defined grid (for plotting purposes). *densf* requires a file with name TAPE21.

```
$ADFBIN/densf << eor

Grid
  0.  -5.  -5.
  100 100
  0. 0. 1. 10.
  0. 1. 0. 10.
End

Orbitals  Local
  1 2 3 4 5
End

END INPUT
eor
```

The program *cntrs* is applied to process the *densf* result file TAPE41.

```
$ADFBIN/cntrs << eor

SCAN
  0.01 0.02 0.04 0.08 0.16 0.32 0.64 1.28
  -0.01 -0.02 -0.04 -0.08 -0.16 -0.32 -0.64 -1.28
END

file ctr.a1
LocOrb%1 1.00

file ctr.a2
LocOrb%2 1.00
```

```
file ctr.a3
LocOrb%3 1.00

file ctr.a4
LocOrb%4 1.00

file ctr.a5
LocOrb%5 1.00

END INPUT
eor
```

Again, *gnuplot* may be used to display the result on your screen.

```
$gnuplot << eor
set term dumb  100 80
set output "outplot"
plot "ctr.a1" using 1:2 with lines
plot "ctr.a2" using 1:2 with lines
plot "ctr.a3" using 1:2 with lines
plot "ctr.a4" using 1:2 with lines
plot "ctr.a5" using 1:2 with lines
eor

cat outplot
```

This results in 5 pictures, the first one looking like:

The second illustration of the computation of localized molecular orbitals in C_2H_2 combines directly all MOs (σ and π). This yields 3 equivalent 'banana' bonds, mixtures of σ and π, and two equivalent pure σ bonds.

```
$ADFBIN/adf << eor
title C2H2   localization without frozen orbitals

Atoms
C 0 0 .63
C 0 0 -.63
H 0 0 1.63
H 0 0 -1.63
end

fragments
C  t21.C
H  t21.H
end

integration 4.0

locorb
end

end input
eor
```

Cu_4CO: Density of States

Sample directory: adf/DOS_Cu4CO/

This sample illustrates the DOS property program to compute density-of-states data, for energy-dependent analysis.

First, the Cu_4CO molecule is calculated (ADF), using single-atom fragments.

```
$ADFBIN/adf <<eor
title  Cu4CO (3,1) from atoms

units
length bohr
end

define
rCu=2.784
end

atoms
1. Cu     rCu      0.0              0.0
2. Cu    -rCu/2    rCu*sqrt(3)/2    0.0
3. Cu    -rCu/2   -rCu*sqrt(3)/2    0.0
4. Cu     0.0      0.0             -rCu*sqrt(2)
5. C      0.0      0.0              2.65
6. O      0.0      0.0              4.91
end

Basis
  Type TZP
  Core small
end

XC
  GGA  PostSCF    Becke Perdew
END

endinput
eor
```

The PostSCF feature in the specification of the XC functional is used: the 'Becke-Perdew' GGA corrections are not included self-consistently but applied to the energy evaluation after the self-consistent LDA solution has been obtained.

The utility program *dos* requires a file named TAPE21 in the current directory, unless overridden using a TAPE21 keyword (not used in this example).

```
$ADFBIN/dos << eor
file dostxt

energyrange npoint=36 e-start=-25 e-end=10
```

```
tdos

! Cu 3d partial DOS
gpdos
  a1   14:22 32:34
  a2    5:10
  e1:1 18:32 37:42
  e1:2 18:32 37:42
end

! The same but using BAS
gpdos
  BAS 17:34 57:74 97:114 137:154
end

! The same as above, but using much less complicated input
gpdos
  ATYPE Cu d
end

! Overlap PDOS between Cu 3d and CO 2p
opdos
  ATYPE Cu 3d
SUBEND
  ATOM 5 2p
  ATOM 6 2p
end

end input
eor
```

Here, the total density of states, as well as various *partial* densities of states, are computed. You may feed the results found in the dostxt file into a plotting program such as *gnuplot*. The result is not displayed here. See the *Utilities* document for more detailed info about the *dos* program.

Third party analysis software

adf2aim: convert an ADF TAPE21 to WFN format (for Bader analysis)

Sample directory: adf/AIM_HF/

Starting from the ADF2008.01 version in ADF one calculate Bader atomic charges using a grid based method. Another possibility for Bader's analysis, an example is described here, is to use the *adf2aim* utility such that a third party program Xaim can be used.

ADF utility adf2aim (original name rdt21) developed by Xavi López, Engelber Sans and Carles Bo (see http://www.quimica.urv.es/ADF_UTIL):
rdt21: convert an ADF TAPE21 to WFN format (for Bader analysis)
This program rdt21 is now called adf2aim and is part of the ADF package, starting from ADF2004.01.

The WFN file is an input file for the third party program Xaim (see http://www.quimica.urv.es/XAIM for details), which is a graphical user interface to programs that can perform the Bader analysis.

Usage of adf2aim:

```
$ADFBIN/adf <<eor
TITLE HF

ATOMS
 1. H  .0000  .0000  .0000
 2. F  .0000  .0000  0.917
End

Basis
End

End input
eor

$ADFBIN/adf2aim TAPE21
echo 'Contents of rdt21.res:'
cat rdt21.res
echo 'Contents of WFN:'
cat WFN
```

NBO analysis: adfnbo, gennbo

Sample directory: adf/H2O_ADFNBO/

Dr. Autschbach, SCM, and Prof. Weinhold have collaborated to prepare a simple in put file generator, called adfnbo, for the GENNBO program of Prof. Weinholds Natural Bond Orbital (NBO) 5.0 package: http://www.chem.wisc.edu/~nbo5
The GENNBO executable is included in the ADF distribution and can be enabled via the license file for all those who buy an NBO manual from either the NBO authors or from SCM (info@scm.com).

Usage:

```
$ADFBIN/adf <<eor
Title simple NBO example for water

Atoms    Z-Matrix
O   0 0 0
H   1 0 0   0.9
H   1 2 0   0.9  100
End

Basis
CORE NONE
TYPE DZ
End

FULLFOCK
```

```
AOMAT2FILE
SAVE TAPE15
SYMMETRY NOSYM

End Input
eor

$ADFBIN/adfnbo <<eor
write
fock
end input
eor

$ADFBIN/gennbo < FILE47
```

A File named FILE47 is generated by adfnbo which is an input file for the general NBO program gennbo. ADF needs to write some data to file, which is done by including these keywords in the adf input file:

```
FULLFOCK
AOMAT2FILE
SAVE TAPE15
SYMMETRY NOSYM
```

GENNBO

This section contains a brief summary of the capabilities of GENNBO, made available by Prof. Weinhold.

GENNBO implements most capabilities of the full NBO 5.0 program suite as described on the NBO website: http://www.chem.wisc.edu/~nbo5
These include determination of natural atomic orbitals (NAOs), bond orbitals (NBOs), and localized MOs (NLMOs), as well as the associated NPA (atomic charges and orbital populations) and NRT (resonance structures, weightings, bond orders) valence descriptors, for a wide variety of uncorrelated and correlated (variational, perturbative, or density functional) theoretical levels. GENNBO-supported options include all keywords except those explicitly requiring interactive communication with the host electronic structure system (viz., $DEL deletions, NEDA, NCS, NJC). The GENNBO program typically sits conveniently on the PC desktop, ready to analyze (or re-analyze at will, with altered options) the final results of a complex ADF calculation performed on a remote cluster.

GENNBO "communicates" with the original ADF calculation through an archive file (JOB.47 file, preserving all necessary details of the final density) that is initially generated by ADF and subsequently becomes the input file for GENNBO. The .47 file contains a standard $NBO ... $END keylist that can be edited with a standard word processor or text editor to include chosen NBO keyword options, just as though they might have appeared in the original input stream of an interactive ADFNBO run. The stand-alone GENNBO program therefore allows many alternative NBO analysis options to be explored at leisure, without costly re-calculation of the wavefunction.

Accuracy

BSSE, SCF convergence, Frequencies

$Cr(CO)_5$+CO: Basis Set Superposition Error

Sample directory: adf/BSSE_CrCO6/

A study of the Basis Set Superposition Error (BSSE) in the formation of $Cr(CO)_6$. from CO and $Cr(CO)_5$.

The basis set superposition error (BSSE) can be calculated with the help of the option to create *Alternative Chemical Elements*. or *Ghost* atoms.

An alternative chemical element is an element with a special feature, not corresponding to one of the predefined chemical elements. It may have, for instance, a different effective nuclear charge or a 'special' atomic mass.

For the BSSE calculation special chemical elements must be created to describe the 'ghost' atoms, which have zero nuclear charge and mass. They do have basis functions (and fit functions), however, and they are used to calculate the lowering of the energy of the system to which the ghost atoms are added, due to the enlargement of the basis by the ghost basis. The ghost atom has the same basis and fit set as the normal element but no nuclear charge and no frozen core (there must be no core description in the Create data file for the ghost atom!).

The following calculations are carried out:

- 1. CO from C and O. This yields the bond energy of CO with respect to the (restricted) basic atoms.
- 2. CO from the fragments CO (as calculated in 1) and the ghost atom Cr and 5 Carbon and 5 Oxygen ghost atoms. The ghost atomic fragments provide basis and fit functions but do not contribute charge or potential to the molecule.
  The bond energy of this calculation is the energy lowering of CO due to the additional basis functions.
  This is the BSSE for CO.
- 3. $Cr(CO)_5$ from Cr and 5 CO's.
  This yields the ('normal') bond energy with respect to the given fragments.
- 4. $Cr(CO)_5$ from $Cr(CO)_5$ as fragment (as calculated in 3) but with the CO basis functions added on the position of the 6th CO ('ghost' CO).
  The bond energy is the BSSE for $Cr(CO)_5$.
- 5. $Cr(CO)_6$ with $Cr(CO)_5$ and CO as fragments.
  The bond energy is the one without BSSE. This bond energy can now be corrected by the sum of superposition contributions of calculations 2 and 4.

This series of calculations is carried out with basis set DZ.

Next, the two BSSE runs (#2 and #4 in the list above) are repeated, but now with the core orthogonalization functions omitted from the ghost bases. One may argue about whether these functions should be included in the ghost basis sets, but since they are very contracted around the ghost nuclei they are not expected to contribute significantly anyway and may then just as well be omitted. This is explicitly verified in the current example by demonstrating that the BSSE is not significantly affected by omitting these functions.

Finally, the whole thing might be redone with basis set TZP, to show that the BSSE decreases with larger basis.

The calculations for the type DZ basis are contained in the sample script (with input- and output files). Those for type TZP bases can be set up easily and may be done as an exercise.

For the first series of calculations, with basis type DZ, the input files are discussed below and the relevant parts are echoed from the output files where the energy decomposition and the total bond energy are printed.

For the other series, using type TZP basis sets, only a summary of the results is given.

Computational details

The calculations in this example all use:

- 1. Frozen core level for the Chromium atom: 2p (for Carbon and Oxygen: 1s);
- 2. Numerical integration precision 4.0 (in Create runs 10.0, the default);
- 3. Default settings for model parameters such as density functional (key XC) and for the remaining computational settings

Basis DZ, including Core Functions

Creation of ghost atoms

Ghost atoms must be created like normal chemical elements. The adf database does not provide the ghost database files. They are easily constructed from the normal database file of the pertaining chemical element: only the frozen core references have to be adapted such that the ghost atom will not have a frozen core. This affects the sections 'core' and 'description' in the database file (see the User's Guide).

For the creation of the Carbon ghost atom with basis DZ the database file is:

```
Carbon (II, ghost)

BASIS
1S   5.40

2S   1.24
2S   1.98
2P   0.96
2P   2.20
END

CORE    0  0  0  0
END

DESCRIPTION
END

FIT
 1S  10.80
 2S  11.59
 2S   7.59
```

```
 2S   4.97
 3S   4.79
 3S   3.35
 3S   2.34
 3S   1.64
 2P   8.34
 2P   5.14
 3P   4.67
 3P   3.10
 3P   2.06
 3D   5.88
 3D   3.84
 3D   2.51
 3D   1.64
 4F   5.40
 4F   3.55
 5G   4.50
END

FITCOEFFICIENTS
/
END
```

Observe that there are four integers zero after the keyword core, indicating that there are no s-, p-, d-, or f-type frozen core shells. Specification of any frozen core shells would imply the insertion of (core) electrons around the ghost atoms in the calculation.

Consequently, the data block directly below core is empty: no Slater-type functions are required to describe any frozen core orbitals.

Finally, the description data block is empty: no expansion coefficients that would describe the frozen core orbitals in terms of the Slater-type expansion functions.

All other data (apart from the title, which is just a label) in the Create data file are unchanged. The ghost file has the same Basis set, the same Fit set as for a normal atom. The values of the fit coefficients are irrelevant and could as well be put zero altogether: in the scf part of the create run on the ghost atom the fit coefficients will be set to zero after the first cycle since there is no charge density to be fitted.

Then the corresponding Create run is carried out.

```
$ADFBIN/adf -n1 << eor
Create  Gh.C  q=0  m=0   file=in.ghost
end input
eor

mv TAPE21 t21.C_ghost
```

The options 'q=' and 'm=' specify the nuclear charge and atomic mass respectively. Both are zero for a ghost atom: it is not a physical object, only the center for a set of functions.

In the same fashion the Oxygen and Chromium ghost atoms are created. The inputs for these are not shown here.

For the BSSE calculations we first do the 'normal' calculations of CO and $Cr(CO)_5$, yielding the fragment files t21.CO and t21.CrCO5. The input files for these calculations are not shown here.

BSSE for CO

For the CO BSSE calculation the standard CO must have been computed first. In the BSSE run a $Cr(CO)_5$ ghost fragment basis set is then added to the 'normal' CO input. The energy change (the printed 'bond energy' in the output) is the BSSE.

The input file for the CO-BSSE run is:

```
title  BSSE for CO due to Cr(CO)5 ghost
noprint sfo,frag,functions

atoms
 Gh.Cr    0       0       0
 Gh.C    -1.86    0       0
 Gh.C     1.86    0       0
 Gh.C     0       1.86    0
 Gh.C     0      -1.86    0
 Gh.C     0       0      -1.86
 Gh.O     3.03    0       0
 Gh.O    -3.03    0       0
 Gh.O     0       3.03    0
 Gh.O     0      -3.03    0
 Gh.O     0       0      -3.03
    C     0       0       1.86       f=CO
    O     0       0       3.03       f=CO
end

fragments
 Gh.Cr  t21.Cr_ghost
 Gh.C   t21.C_ghost
 Gh.O   t21.O_ghost
 CO    t21.CO
end

symmetry  C(4V)
integration 4

endinput
```

In the output we find in the Bond Energy section:

	hartree	eV	kcal/mol	kJ/mol
Pauli Repulsion				
Kinetic (Delta T^0):	0.000000000000025	0.0000	0.00	0.00
Delta V^Pauli Coulomb:	-0.000000000000021	0.0000	0.00	0.00
Delta V^Pauli LDA-XC:	-0.000000000000007	0.0000	0.00	0.00
Total Pauli Repulsion:	-0.000000000000004	0.0000	0.00	0.00
(Total Pauli Repulsion = Delta E^Pauli in BB paper)				
Steric Interaction				
Pauli Repulsion (Delta E^Pauli):	-0.000000000000004	0.0000	0.00	0.00
Electrostatic Interaction:	0.000000000000057	0.0000	0.00	0.00
(Electrostatic Interaction = Delta V_elstat in the BB paper)				
Total Steric Interaction:	0.000000000000054	0.0000	0.00	0.00
(Total Steric Interaction =				

```
  Delta E^0 in the BB paper)

Orbital Interactions
  A1:                              -0.001838637105191      -0.0500         -1.15          -4.83
  A2:                               0.000000000000000       0.0000          0.00           0.00
  B1:                               0.000000000000000       0.0000          0.00           0.00
  B2:                               0.000000000000000       0.0000          0.00           0.00
  E1:                              -0.002025936206895      -0.0551         -1.27          -5.32
                                  --------------------    --------        -------        --------
  Total Orbital Interactions:      -0.003864573312086      -0.1052         -2.43         -10.15

Alternative Decomposition Orb.Int.
  Kinetic:                         -0.056036607909737      -1.5248        -35.16        -147.12
  Coulomb:                          0.048666200804427       1.3243         30.54         127.77
  XC:                               0.003505833793224       0.0954          2.20           9.20
                                  --------------------    --------        -------        --------
  Total Orbital Interactions:      -0.003864573312086      -0.1052         -2.43         -10.15

  Residu (E=Steric+OrbInt+Res):    -0.000000000000002       0.0000          0.00           0.00

Total Bonding Energy:              -0.003864573312034      -0.1052         -2.43         -10.15


Summary of Bonding Energy (energy terms are taken from the energy decomposition above)
================================================================================================

  Electrostatic Energy:             0.000000000000057       0.0000          0.00           0.00
  Kinetic Energy:                  -0.056036607909712      -1.5248        -35.16        -147.12
  Coulomb (Steric+OrbInt) Energy:   0.048666200804404       1.3243         30.54         127.77
  XC Energy:                        0.003505833793217       0.0954          2.20           9.20
                                  --------------------    --------        -------        --------
  Total Bonding Energy:            -0.003864573312034      -0.1052         -2.43         -10.15
```

The BSSE for CO is computed as 2.42 kcal/mole

BSSE for $Cr(CO)_5$

In similar fashion the BSSE is computed for $Cr(CO)_5$. In the BSSE run a ghost CO is added to the normal $Cr(CO)_5$ input:

```
title BSSE for Cr(CO)5 due to CO ghost
noprint sfo,frag,functions

atoms
 Cr    0       0       0        f=CrCO5
 C     1.86    0       0        f=CrCO5
 C    -1.86    0       0        f=CrCO5
 C     0       1.86    0        f=CrCO5
 C     0      -1.86    0        f=CrCO5
 C     0       0      -1.86     f=CrCO5
 O     3.03    0       0        f=CrCO5
 O    -3.03    0       0        f=CrCO5
 O     0       3.03    0        f=CrCO5
 O     0      -3.03    0        f=CrCO5
 O     0       0      -3.03     f=CrCO5
 Gh.C   0       0       1.86
 Gh.O   0       0       3.03
end

fragments
 CrCO5  t21.CrCO5
 Gh.C    t21.C_ghost
 Gh.O    t21.O_ghost
end
```

```
symmetry C(4v)
integration 4

endinput
```

The Bond Energy result yields 1.93 kcal/mole for the BSSE.

Bond Energy calculation with BSSE correction

The bonding of CO to $Cr(CO)_5$ is computed in the normal way (not included in the sample): from fragments CO and $Cr(CO)_5$. The obtained value for the bond energy is then simply corrected for the two BSSE terms, 4.35 kcal/mole together.

Relevance of Core Functions

The whole procedure explained above is repeated with now the Core Functions (the functions in the valence basis set that serve only for core-orthogonalization, for instance the 1s 5.40 in the Carbon basis set) removed from the Create data files used for the creation of the ghost atoms.

This yields as BSSE values for CO and $Cr(CO)_5$ respectively 2.33 and 1.88 kcal/mole (compare 2.42 and 1.93 kcal/mole for the case with Core Functions included). The net total effect of including/removing the Core Functions is therefore (2.42-2.33)+(1.93-1.88)=0.14 kcal/mole. This is an order of magnitude smaller than the BSSE effect itself.

In the last calculation a PRINT instruction is inserted in the input file to let the program output the symmetry group representations, character table and multiplication table. This information is printed after the lists of basis and fit sets.

BSSE and the size of the basis set

BSSE effects should diminish with larger bases and disappear in the limit of a perfect basis. This can be studied by comparing the BSSE for basis DZ, see above, with the BSSE for basis TZP. The procedure is completely similar to the one above and yields:

For the BSSE terms, using basis sets with Core Functions included: 0.7 kcal/mole for CO (compare: 2.4 kcal/mole for basis DZ), and 0.6 kcal/mole for $Cr(CO)_5$ (1.9 for basis DZ)

Without Core Functions the numbers are similar.

The total BSSE drops from 4.3 kcal/mole in basis DZ to 1.3 in basis TZP (if Core Functions are included in the Create runs for the ghosts), and changes very slightly when the Core Functions are omitted.

Create runs for the ghosts), and changes very slightly when the Core Functions are omitted.

A systematic study with adf of the BSSE in metal-carbonyl complexes can be found in
Rosa, A., et al., Basis Set Effects in Density Functional Calculations on the Metal-Ligand and Metal-Metal Bonds of Cr(CO)5-CO and (CO)5. Journal of Physical Chemistry, 1996, 100: p. 5690-5696

Ti_2O_4: troubleshooting SCF convergence

Sample directory: adf/SCF_Ti2O4/

One can run into SCF convergence problems when calculating certain types of systems. Some of the notorious examples are transition metal oxides and lantanide compounds. Below, several approaches to solving the SCF convergence problem are demonstrated.

NewDIIS keyword

The first approach is to try a new DIIS algorithm, which will probably become default in a future version. The new algorithm is switched on by using the keyword *NewDIIS* anywhere in the input file:

```
$ADFBIN/adf << eor
Title Ti2O4 SCF aid test (NewDIIS)
Atoms
  Ti  1.730   0.000   0.000
  Ti -1.730   0.000   0.000
  O   0.000   1.224   0.000
  O   0.000  -1.224   0.000
  O   3.850   0.000   0.000
  O  -3.850   0.000   0.000
End
XC
 GGA Becke Perdew
End
Basis
 Type DZ
 Core Small
End

SCF
  Iterations 300
End

NewDIIS

End input
eor
```

Multi-step smearing

Second approach is an extension to the so-called "electron smearing" method. In this method, the electrons are distributed among orbitals around Fermi-level using a pseudo-thermal distribution function. Although the result with fractional occupation number has no physical sense, the method can be used to achieve integer occupation numbers by reducing the smearing parameter step-wise. In the example above, replace the NewDIIS keyword with the following line of text:

```
Occupations Smear=0.2,0.1,0.07,0.05,0.03,0.02,0.01,0.007,0.005,0.001
```

A few notes:

- You can specify up to ten comma-delimited values after Smear= (no spaces are allowed). ADF will start from the first value and try to converge SCF using it. If it succeeds, the next value will be picked and so on.
- Because the whole process may require many cycles to converge it is important to set the number of SCF cycles to a large value to avoid a premature termination.

Steepest descent method

The third example demonstrates the use of the *Occupations Steep=* option (see the User's Guide for details). There are two differences from the previous example shown below:

```
SCF
 Iterations 300
 Mixing 0.05
 DIIS N=0
End

Occupations Steep=0.5,0.3
```

One difference is, obviously, in the *Occupations* keyword. The other difference is more subtle. For stable convergence, it is often essential to switch off DIIS and set the mixing parameter to a low value. Of course, it will make convergence quite (sometimes very) slow. Ultimately you should get either an aufbau configuration or a configuration with exactly degenerate HOMO. In this example, the result is an aufbau solution.

Both methods should, in principle, give the same result, which is the case in this example.

Energy-DIIS

The fourth example uses the so called Energy-DIIS method. Please note that similar to ARH and unlike the standard SCF procedure in ADF this method requires energy evaluation at each SCF cycle, which makes it significantly slower compared to energy-free SCF.

```
SCF
 Iterations 300
 Mixing 0.05
 EDIIS
End
```

Augmented Roothaan-Hall

The fifth example uses the Augmented Roothaan-Hall (ARH) method. The basic idea of this method is that the density matrix is optimized directly to minimize the total energy. Important: the ARH method can be used with SYMMETRY NOSYM only.

```
Symmetry NSOYM
SCF
 Iterations 300
 Mixing 0.05
 EDIIS
End
```

NH_3: rescan frequencies

Sample directory: adf/Freq_NH3_Scan/

Sometimes spurious imaginary frequencies are calculated where one would expect a very low (nearly zero) frequency. Most frequently this happens when there is a barrier-free rotation of, for example, methyl groups. The SCANFREQ keyword allows one to rescan calculated frequencies in order to find out if they wre calculated accurately.

In this example analytical frequencies are calculated. Next recalculation of certain NH3 frequencies are performed by scanning along normal modes from a restart file. In this calculation the frequencies are calculated numerically with finite displacements using symmetry.

```
$ADFBIN/adf <<eor
title NH3 analytic frequencies
atoms
   N               0.0000    0.0000    0.0000
   H               0.4729    0.8190    0.3821
   H              -0.9457    0.0000    0.3821
   H               0.4729   -0.8190    0.3821
end
Basis
 Type TZP
 Core Small
End
AnalyticalFreq
End
integration  5.0
end input
eor

mv TAPE21 NH3_anl.t21

$ADFBIN/adf <<eor
title Re-calculate NH3 frequencies by scanning along normal modes from a restart file
atoms
   N               0.0000    0.0000    0.0000
   H               0.4729    0.8190    0.3821
   H              -0.9457    0.0000    0.3821
   H               0.4729   -0.8190    0.3821
end
Fragments
  N t21.N
  H t21.H
End
ScanFreq 0 4000
Restart NH3_anl.t21
integration  5.0
end input
eor
```

Scripting

Prepare an ADF job and generate a report

Bakerset: GO optimization for multiple xyz files

Sample directory: adf/BakersetSP/

In this example you will find how to use *adfprep* to run a particular job (a single point calculation in this case) for all molecules in the Baker set. The molecules are simply xyz files and contain no ADF specific information. *adfreport* is used to collect the resulting bonding energies.

```
rm -f runset
for f in $ADFHOME/examples/adf/BakersetSP/Bakerset/*.xyz
do
  "$ADFBIN/adfprep" -t SP -i 2.5 -b DZ -c Large -m "$f" -j `basename $f .xyz`>> runset
done

chmod +x runset
./runset

echo Results
ls -t -1 *.t21 | while read f
do
    "$ADFBIN/adfreport" "$f" BondingEnergy
done
echo Ready
```

Methane: basis set and integration accuracy convergence test

Sample directory: adf/ConvergenceTestCH4/

In this example you will find how to use *adfprep* to test convergence of the bonding energy with respect to basis set and integration accuracy. *adfreport* is used to collect the resulting bonding energies.

```
rm -f runset
for b in SZ DZ DZP TZP TZ2P QZ4P
do
    "$ADFBIN/adfprep" -t "$ADFHOME/examples/adf/ConvergenceTestCH4/methane.adf" \
                      -b $b -j methane.$b >> runset
done

chmod +x runset
./runset

echo Results
echo Basis set convergence of Bonding Energy, SZ DZ DZP TZP TZ2P QZ4P
for b in SZ DZ DZP TZP TZ2P QZ4P
```

```
do
    "$ADFBIN/adfreport" "methane.$b.t21" BondingEnergy
done

rm -f runset
for i in 2 3 4 5
do
    "$ADFBIN/adfprep" -t "$ADFHOME/examples/adf/ConvergenceTestCH4/methane.adf" \
                      -b DZP -i $i -j methane.$i >> runset
done

chmod +x runset
./runset

echo Integration convergence of Bonding Energy, 2 3 4 5
for i in 2 3 4 5
do
    "$ADFBIN/adfreport" "methane.$i.t21" BondingEnergy
done

echo Ready
```

List of examples