Scientific Computing & Modelling

DFTB Manual

ADF Program System
Release 2012

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

Copyright © 1993-2012: SCM / Vrije Universiteit, Theoretical Chemistry, Amsterdam, The Netherlands
All rights reserved

Table of Contents

Introduction

Our implementation of the DFTB method can perform single point calculations, geometry optimizations, transition state searches, frequency calculations, and molecular dynamics. Molecules as well as periodic systems can be handled ensuring a smooth link with our full DFT codes ADF and BAND. It can be used as a stand-alone command line program, or from the graphical interface.

The DFTB program is orders of magnitude faster than DFT, but requires parameter files to be installed for all pair-wise combinations of atoms in a molecule. Many elements can be handled with the Dresden parameter set included in the distribution, while many other parameter sets (from DFTB.org) can be enabled, free of charge for non-profit users. Alternatively, sets of parameters in the SKF format can be downloaded and used from third party sources.

Three models within the DFTB framework are available: standard DFTB, SCC-DFTB (DFTB with self-consistent-charge correction), and DFTB3 (SCC-DFTB with third-order correction). As they have been respectively parameterized, it is important to specify a proper parameter set when applying one of these models.

Release 2013

The 2013 release of our DFTB program features major updates:

Method improvements

- Car-Parrinello DFTB for second-order evaluation (CPBO) and molecular dynamics (CPMD).
- Conjugate Gradients geometry optimization

DFTB-GUI

Note that the graphical user interface DFTB-GUI enables all users to set up complicated calculations with a few mouse clicks, and provides graphical representations of calculated data fields, see the DFTB-GUI tutorials.

Input

The input for DFTB slightly differs from the one found in ADF. General nomenclature and structure are however similar to those found in ADF. We refer to the ADF User Guide for more information on this topic. Input files from the DFTB 2012 are compatible with the 2013 version. In the following sections, a list of the relevant keys and the contained sub-keys will be presented.

After the run, results of the computation are written to standard output. Binary information about the evaluation are also written to a keyed-file dftb.rkf.

Specification of the System

The input of the initial structure can be given with the key System. This key is generally mandatory, but for molecular dynamics restarts, it can be omitted. In that case, the molecular configuration of the last recorded iteration will be used.

```
System
    {Atoms
        Atom Coords
    End}
    {Charge NetQ}
    {Lattice
        Vectors
    End}

    {LatticeStrain
         eps1 value
         eps2 value
         eps3 value
         eps4 value
         eps5 value
         eps6 value
    }
    {FractionalCoords}
End
```

The System key accepts a set of sub-keys to specify various details of the chemical system under evaluation

`Atoms`

Specifies the geometry of the molecular system as a list of rows, one row per atom.

`Atom`

The name of an *atom type*. It must be the standard one- or two-characters symbol for the chemical element: H, He, Li, and so on.

`Coords`

This specifies the coordinates of the atom. The x, y, z values of the Cartesian coordinates are by default interpreted in Ångstrom.

`Charge`

The net charge of the molecule can be controlled with the optional sub-key CHARGE. If this sub-key is omitted the net total charge of the molecule is by default zero.

`NetQ`

The net total charge of the molecule.

`Lattice`

Information about the periodicity of the system is given through this sub-key. Its presence is optional, and it implies a periodic system. The subsequent computation will therefore evaluate the system accordingly. A list of up to three vectors (one per row) for the cell must be specified.

`Vector`

Three floating point values defining the periodicity vector along a given direction. One, two, or three vectors can be specified (each on a different row) to express linear, planar or bulk periodicity, respectively. If one vector is specified, periodicity must develop along the x axis. If two vectors are specified, periodicity must develop along the xy plane. The unit is the same of the Atoms section.

`LatticeStrain`

Allows the application of a strain tensor to the lattice. The values of eps1 to eps6 represent the unique elements of the strain tensor, as follows

eps1	eps6	eps5
eps6	eps2	eps4
eps5	eps4	eps3

`FractionalCoords`

This optional keyword modifies how the ATOMS coordinates are interpreted. When the keyword is present, coordinates will be interpreted as fractions of the periodicity lattice vectors, instead of absolute geometric positions in 3D space. Necessarily, the presence of this sub-key requires LATTICE to be specified.

Specification of the computational Task

The Task section is mandatory and allows to specify the computational task to perform. It accepts only one mandatory sub-key, runType.

```
Task
 runType type
End
```

`type`

The type of evaluation to perform.

It can be:
- SinglePoint or SP
- GeometryOptimization or GO
- TransitionState or TS
- Frequencies or F
- MolecularDynamics or MD
- Phonons

Changing the default Units

The Units key is optional. It allows to specify different units for Length and Time, in place of the default ones.

```
Units
 {length    angstrom|bohr}
 {time      femtosec|au}
End
```

Setting DFTB Calculation details

```
DFTB
 ResourcesDir  relativepath
 {RadialExtrapolation  none|linear|improved|original|bezier}
 {SCC
    {iterations NIter}
    {thirdorder}
    {converge charge=QDiff}
 End}
 {CPBO
    {iterations NIter}
    {converge charge=QDiff energy=EDiff}
    {weight min=minWeight max=maxWeight initial=initialWeight}
 End}
 {CPMD
    {iterations NIter}
    {timestep tstep}
    {orbitalsMass orbMass}
    {converge charge=QDiff}
 End}
 {UseSymmetry  yes|no}
 {Repulsion
    forcePolynomial
 End}
 {Dispersion
    method UFF|D2|D3-BJ|D3-zero
 End}
 {Occupation hund|fermi {temperature=FermiTemp}}
End
```

This mandatory key allows to specify and control different aspects of the DFTB evaluation engine.

`ResourcesDir`

Allows to specify the path (relative to $ADFRESOURCES/DFTB) of the directory containing DFTB parameter files. Different parameters may be suitable for different DFTB evaluations. It is important to choose the appropriate parameter set for the type of calculation and molecular system under study. Alternatively, an absolute path (starting with a slash character) can be specified to have complete freedom over the location of the directory. Examples:

`ResourcesDir Dresden`

Uses the Resource directory $ADFRESOURCES/DFTB/Dresden

`ResourcesDir /home/myusername/myskfdir`

Uses the specified path /home/myusername/myskfdir as the resource directory

NOTE: each resource directory should contain a file called *metainfo.yaml*, which defines extra information (Hubbard derivatives, dispersion parameters, etc.) required by DFTB calculation. For details see metainfo.yaml.

`RadialExtrapolation`

Advanced control option. Overrides the extrapolation method for Slater-Koster grid values between the end of the tabulated grid and the cutoff distance (value for which atoms are considered too far to interact). Depending on the structure of your Slater-Koster tables, a different radial extrapolation method may be needed in order to guarantee correct behavior, in particular for large and periodic systems. Five different extrapolation strategies are available:

`none`

Performs no extrapolation, the value being forced to zero at distances greater than the grid last position.

`linear`

performs a linear interpolation between the last point of the grid and the value of zero, at cutoff distance.

`improved (default)`

Perform a 9th grade polynomial interpolation between 6 points of the grid and three zeros. This interpolation may prove unstable for particular Slater-Koster data

`original`

same as improved, but reproducing behavior of previous reference programs. Should not be used in general.

`bezier`

Uses a Bézier curve passing through the last grid point and the cutoff point, guaranteeing continuity and smoothness. This is the suggested method in case of unexpected behavior.

`SCC`

The SCC key is optional. By its presence the SCC-DFTB model is used, where the self-consistent-charge (SCC) iterative procedure is performed in DFTB by diagonalizing the charge-dependent Hamiltonian at each step. If this key is not present, DFTB will perform a non-SCC evaluation (standard DFTB). Subkeys are optional for setting up the SCC details and choosing the SCC level:

`thirdorder`

This option turns on the DFTB3 model, which applies a third-order correction in addition to SCC-DFTB.

`iterations NIter`

Allows to specify the maximum number of SCC iterations. Default is 100 iterations, which is a very high number. Most computations will converge in less iterations. Lack of convergence within this limit may be due to use of Hund Occupation. Fermi occupation may improve convergence. See the Occupation key below.

`converge charge=QDiff`

Specifies the tolerance for convergence on the variation of the atomic charges. The default is 1.0e-8.

```
CPBO
```

This key is mutually exclusive with the SCC and CPMD keys. Its presence enables the Car-Parrinello Born Oppenheimer method to perform optimization at the Self-Consistent Charge level (second order only). Details of the methodology can be found in the references section. The procedure is initiated via a single non-SCC evaluation. It is important to note that this methodology requires a Hund occupation scheme. Additional optional keys can be used to control the procedure

```
iterations NIter
```

Allows to specify the maximum number of Car-Parrinello iterations to converge the atomic charges and energy. Default is 500 iterations. A Car-Parrinello evaluation may require a high number of iterations than SCC, but the individual iteration will be faster.

```
converge charge=QDiff energy=EDiff
```

Specifies the tolerance for convergence on the variation of the atomic charges. The default is 1.0e-7 for both charges and energy, in their respective units.

```
weight min=minWeight max=maxWeight initial=initialWeight
```

Specifies the weight coefficient for the internal Verlet algorithm used to propagate the orbitals according to the Car-Parrinello method. Default values are min=0.01, max=0.5, initial=0.5. The weight determines how strongly the orbitals change after each iteration. A small value implies a small variation, while a larger one implies a large one. The algorithm adjusts the weight for optimal convergence, increasing the current value at each iteration if the process is too slow, or decreasing it if the minimum is closer to the current value and the next step would increment the energy. The current weight is always clamped between the min and max values. The initial value specifies the value for the first Car-Parrinello iteration.

```
CPMD
```

This key is mutually exclusive with the SCC and CPBO keys. Its presence enables the Car-Parrinello Molecular Dynamics method to propagate optimized orbitals during a molecular dynamics evaluation. As a consequence, CPMD cannot be used with the runType is anything else than MD. The procedure is initiated via two CPBO evaluations at successive geometries. It is important to note that this methodology requires a Hund occupation scheme. Additional optional keys can be used to control the procedure

```
iterations NIter
```

Specifies the maximum number of iterations allowed during the initial two Car Parrinello Born Oppenheimer steps to initialize the CPMD procedure. The defaults are the same as for CPBO.

```
converge charge=QDiff energy=EDiff
```

Specifies the tolerance for convergence of the initial two Car Parrinello Born Oppenheimer steps to initialize the CPMD procedure. The defaults are the same as for CPBO.

```
weight min=minWeight max=maxWeight initial=initialWeight
```

Specifies the weight coefficients for the initial two Car Parrinello Born Oppenheimer steps to initialize the CPMD procedure. The defaults are the same as for CPBO.

```
orbitalsMass mass
```

Specifies the fictitious orbitals mass for the Car Parrinello MD propagation method. The default is 25 × (*timestep*)^2. A small mass allows the orbitals to respond quickly to changes in geometry, but may confer excessive kinetic energy to the orbitals. A larger mass reduces this energy transfer but the orbitals have lower response to geometry change, requiring a smaller MD timestep.

`timestep tstep`

Specifies the timestep used for propagation of the orbitals. The default is the timestep as specified in the molecular dynamics (MD) section. It is generally required for the stability of the method for these two values to be the same.

`UseSymmetry yes|no`

Enables or disables the use of symmetry during the evaluation. The key is optional, and in its absence the default is yes. For MD calculations, NO symmetry will be used.

`Repulsion`

This key allows to specify some details about the repulsion contribution evaluation. It accepts only one sub-key "forcePolynomial", which forces the use of the polynomial representation (as defined in the header of the Slater-Koster parameter files), in place of the Spline description.

`Dispersion`

This key allows to specify options for the London dispersion correction. If it does not appear, as default, NO dispersion correction will be included.

`method UFF|D2|D3-BJ|D3-zero`

This subkey is used to specify a dispersion model.

- UFF: please see A. K. Rappé, C. J. Casewit, K. S. Colwell, W. A. Goddard III, W. M. Skiff, *J. Am. Chem. Soc.* **1992**, *114*, 10024-10035.
- D2: S. Grimme's D2 correction.
- D3-BJ or D3-zero: S. Grimme's D3 correction with "Becke-Johnson" or "zero" damping. See S. Grimme, J. Antony, S. Ehrlich, H. Krieg, *J. Chem. Phys.* **2010**, *132*, 154104.

  NOTE: at the moment neither D3-BJ nor D3-zero is yet available for periodicity. Additionally, D3-zero has been fitted for SCC-DFTB, but NOT DFTB or DFTB3.

  If no such "method" specified in the "Dispersion" block, default dispersion correction defined in the *metainfo.yaml* file (depends on "ResourcesDir", see metainfo.yaml) will be applied.

`d3parameters s8=S8Param a1=A1Param a2=A2Param`

User-customized D3 parameters can be specified with this subkey. Otherwise, the default parameters defined in *metainfo.yaml* (depends on "ResourcesDir", see above) are used.

`Occupation`

This optional key allows to specify the fill strategy to use for the orbitals. It can be either "hund", to fill the orbitals according to the Hund's rule, or "fermi", to perform electronic charge distribution over the orbitals. If "fermi" is specified, a further "temperature" option must be present, specifying the Fermi temperature in Kelvin (K). If this key is absent, the default is Fermi occupation with a temperature of 5 K. This option cannot be used with the Car Parrinello methods, requiring Hund occupation.

Geometry optimization

```
Geometry
 {Method      quasinewton|conjgrads}
 {CGType
fletcherreeves|polakribiere|hestenesstiefel|perry|birginmartinez|
                scaled|adaptedscaled1|adaptedscaled2|swartdyn1|swartdyn2}
 {SwartIter   Niter}
 {Optim       Cartesian|Delocal|Primitive|Internal}
 {Iterations  Niter}
 {Converge    {E=TolE} {Grad=TolG} {Rad=TolR}}
 {Step        {TrustRadius=MaxRadius}}
 {OptimizeLattice}
End
```

Geometry allows to specify information about the geometry optimization strategy. The keyword must be specified only for those Task runTypes requiring a geometry optimization (GeometryOptimization and TransitionState).

`Method`

`quasinewton|conjgrads`

Selects the optimization algorithm to use, between Quasi-Newton (Hessian-based) and Conjugate Gradients (gradient-based). The default is "quasinewton".

`CGType`

Defines the methodology to compute the β_n value to update the conjugate direction. This keyword is relevant only if the Method is conjgrads. By default, the Fletcher Reeves methodology is used.

`fletcherreeves`

Uses the Fletcher-Reeves formula $\beta_{n} = \frac{\Delta x_n^\top \Delta x_n} {\Delta x_{n-1}^\top \Delta x_{n-1}}$

`polakribiere`

Uses the Polak-Ribiere formula $\beta_{n} = \frac{\Delta x_n^\top (\Delta x_n-\Delta x_{n-1})} {\Delta x_{n-1}^\top \Delta x_{n-1}}$

`hestenesstiefel`

Uses the Hestenes-Stiefel formula $\beta_n = -\frac{\Delta x_n^\top (\Delta x_n-\Delta x_{n-1})} {s_{n-1}^\top (\Delta x_n-\Delta x_{n-1})}$

`perry`

Uses the Perry formula $\beta_n = -\frac{\Delta x_n^\top (\Delta x_n-\Delta x_{n-1})} {s_{n-1}^\top (\Delta x_n-\Delta x_{n-1})}$

`birginmartinez`

Similar to Perry, with a scaled $(\Delta x_n-\Delta x_{n-1})$ at the numerator by $\theta = \frac{s_{n-1}^\top{s_{n-1}}}{s_{n-1}^\top (\Delta x_n-\Delta x_{n-1})}

`scaled`

Uses $\beta_n^{Scaled} = - \frac{\Delta x_n^\top \Delta x_n - \Delta x_{n-1}^\top \Delta x_{n}}{s_{n-1}^\top \Delta x_{n-1}}

`adaptedscaled1`

Same as scaled, with $\beta = \max\left({1-\beta_n^{Scaled}, \beta_n^{Scaled}}\right)$

`adaptedscaled2`

Same as scaled, with $\beta = 1-\beta_n^{Scaled}$

`swartdyn1`

Uses adaptedscaled1 up to a given number of steps (specified as SwartIter), then uses Polak-Ribiere

`swartdyn2`

Uses adaptedscaled2 up to a given number of steps (specified as SwartIter), then uses Polak-Ribiere

`SwartIter`

Defines the number of steps required for the swartdyn CGTypes to switch methodology. Only relevant when the proper CGType is used. Default is 300

`Optim`

`cartesian|delocal|primitive|internal`

Optimization in delocalized coordinates (Delocal) can only be used in geometry optimizations or transition state searches with the quasi-newton method.

`Iterations`

`Niter`

The maximum number of geometry iterations allowed to locate the desired structure. The default is 50.
This is a fairly large number. If the geometry has not converged (at least to a reasonable extent) within that many iterations, there may be an underlying cause to consider, instead of simply increasing the allowed number of cycles.

`Converge`

Convergence is monitored for two items: the energy and the Cartesian gradients. Convergence criteria can be specified separately for each of these items:

`TolE`

The criterion for changes in the energy, in Hartrees. Default: 1e-5.

`TolG`

Applies to gradients, in Hartree/Ångstrom. Default: 1e-3.

`TolR`

The maximum Cartesian step allowed for a converged geometry, in Ångstrom. Default: 0.001 Ångstrom.

```
Step
```

Controls that changes in geometry from one cycle to another are not too large:

```
MaxRadius
```

By default, the trust radius is set to 0.2. Using the key, the user can override this, setting a constant value. A conservative value is 0.2. A large system (e.g., 100 atoms) typically needs a larger trust radius (e.g., 0.8).

```
OptimizeLattice
```

Enables optimization of the Lattice parameters, in addition to the molecular geometry. This can only be applied to periodic systems.

Restart

```
Restart
      {RestartFile}
      {RestartMulliken}
      {RestartOrbitals}
End
```

Molecular Dynamics

```
MD
 Steps      NSteps
 TimeStep   TStep
 {Restart  file=path}
 {Checkpoint frequency=ChkFreq}
 {Trajectory samplingFreq=SFreq}
 {Preserve   [TotalMomentum AngularMomentum CenterOfMass All None]}
 {InitialVelocities zero|inline|random {temperature=InitTemp}}
 {InlineVelocities
   velocityVector
 End}
 {Thermostat type=ThermoType {thermostat options}}
End
```

The DFTB program supports molecular dynamics (with Velocity Verlet) with and without thermostats. This key, used with Task runType is set to MD, allows to specify the information needed by the molecular dynamics evaluation. This implementation of MD supports periodic systems.

```
Steps NSteps
```

Specifies the number of steps to be taken in the MD simulations. It accepts a simple integer number NSteps.

```
TimeStep TStep
```

Specifies the time for each step. By default, the unit is femtoseconds. Through the Units key, it can be changed to atomic units of time.

```
Restart file=path
```

Triggers a restart procedure, recovering the latest known information from the specified file (either a final .rkf file, or a checkpoint .chk file). When this keyword is present, System, Velocity, previous average values and energy transfers will be recovered from the file, ignoring any redundant specification made in the input file. This is the only situation where the System keyword can be omitted.

```
Checkpoint frequency=ChkFreq
```

Sets the frequency (in steps) for checkpoint the current status to a file. This allows to restart from an intermediate configuration in case of a crash of the program or the system. The keyword is optional; if not specified, by default is equal to the number of steps divided by 4. Only the most recent checkpoint is preserved. In case of crash, the checkpoint may be found in the execution temporary directory, instead of the working path. Checkpoint files can be inspected with the GUI for the latest configuration.

```
Trajectory samplingFreq=SFreq
```

Sets the frequency for printing to stdout and storing the molecular configuration on the .rkf file. This keyword is optional, and the default is the number of steps divided by 1000 (minimum one).

```
Preserve [TotalMomentum AngularMomentum CenterOfMass All None]
```

Constrains the molecular dynamics simulation to preserve different whole-system parameters. Note that this option has poor meaning for periodic systems. The keys can be given as a sequence out of the allowed list, with words separated by spaces

`TotalMomentum`

removes the overall velocity of the system from the atomic velocities.

`AngularMomentum`

removes the overall angular velocity of the system from the atomic velocities.

`CenterOfMass`

keeps the molecular system centered on the current center of mass.

`All`

Specifying "All" is equivalent of specifying all of the above keywords

`None`

None of the above options will be enabled. This is the default setup if the Preserve keyword is not specified.

```
InitialVelocities zero|inline|random {temperature=InitTemp}
```

Specifies the initial velocities to assign to the atoms. Three methods to assign velocities are available

`zero`

All atom's velocities are set to zero

`inline`

Atom's velocities are set to the values specified in the key InlineVelocities (see below)

`random temperature=InitTemp`

Atom's velocities are set to random values according to the specified temperature InitTemp, in kelvin. The temperature keyword is mandatory for this choice.

`InlineVelocities`

This optional key is read when InitialVelocities inline option is used. It allows to specify the velocities for each atom. Each row must contain three floating point values (corresponding to the x,y,z component of the velocity vector) and a number of rows equal to the number of atoms must be present, given in the same order as the System Atoms specification.

Available Thermostats

The key Thermostat allows to specify the use of a thermostat during the simulation. Depending on the selected thermostat type, different additional options may be needed to characterize the specific thermostat behavior. At the moment, the following choices for the type parameter are available

`None`

No thermostat applied. This is the default if no Thermostat key is present.

`Scale`

Applies a scaling of the velocities in agreement to the specified temperature. The following options are required for this thermostat

`frequency=NSteps`

This parameter is optional. If specified, the thermostat will be applied every NSteps, using that step's ensemble temperature and the specified thermostat temperature to compute the scaling factor. If not specified, the thermostat will be applied at every step, using the mean temperature of the ensemble and the specified thermostat temperature to compute the scaling factor.

`temperature=Temp`

Specifies the temperature of the thermostat, in kelvin. This parameter is mandatory.

`Berendsen`

Applies the Berendsen thermostat. The following options are required for this thermostat

`tau`

Specifies the initial tau parameter for the Berendsen thermostat, in femtoseconds (can be changed via Units key).

`apply=local|global`

Defines the scope of application of the scaling correction, either per-atom-velocity (option local) or on the molecular system as a whole (option global)

`temperature=Temp`

Specifies the temperature of the thermostat, in kelvin. This parameter is mandatory.

Additional Periodicity Data

```
Periodic
    {KSpace NK}
    {BZStruct {enabled=yes|no} {automatic=yes|no} {interpol=intVal}}
    {Phonon reorderatoms=yes|no interpol=N rcelx=N stepsize=N}
    {BZPath
        KMesh NMesh
        {Path
        End}
    End}
    {SuperCell
    End}
    {LatticeStepSize}
    {stressTensor}
    {Screening}
End
```

`KSpace`

This parameter controls the number of k-points used in the calculation. For very small unit cells (one atom wide) a value of 5 is advised. For medium sized unit cells 3 is adequate. For very large ones (10 atoms wide) kspace=1 suffices. (Default is 5)

`BZStruct`

This controls the path taken through the Brillouin zone (for plotting purposes). It has no effect on the calculated energy. Specifying *intVal* the band structure is interpolated along the path (so extra k-points are generated). You can also specify a path by hand, setting automatic to *no*. The points in the path should be entered in the BZPath key.

`enabled`

By default this feature is enabled.

`automatic`

Whether of not to use the automatic path generation.

`interpol`

Level of interpolation to use along the path

`Phonon`

This enables a phonon run. One should start from a completely optimized system. Next one should choose a super cell. The phonon spectrum converges with super cell size. How big it should be depends on the system.

`reorderatoms`

Technical option: put atoms of the same type after each other.

`stepsize`

Step size to be taken to obtain the force constants (second derivative) from the analytical gradients.

`BZPath`

Allows the user to specify manually a path through the BZ. The points are in terms of reciprocal lattice vectors.

`KMesh`

The amount of points on each line segment

`Path`

Each path consists of a number of points, which are assumed to be connected.

`SuperCell`

Used for the phonon run. The super lattice is expressed in the lattice vectors. Most people will find a diagonal matrix easiest to understand.

Examples

The $ADFHOME/examples/dftb directory contains many different example files, covering various DFTB options.
The run script below is for the geometry optimization of aspirin at the SCC-DFTB level:

```
$ADFBIN/dftb << eor

Task
  RunType GO
End

System
    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

DFTB
    ResourcesDir Dresden
    SCC
        iterations 200
        converge charge=1.0e-8
    End
End

Geometry
    iterations 100
End

eor
```

Parameter files

The set of DFTB parameter files available in the ADF package were designed by J. Frenzel, A.F. Oliveira, N. Jardillier, T. Heine, and G. Seifert, mainly at the Technische Universität in Dresden, Germany, see also some additional information about the generation of these parameter files. These parameter files are kept in the directory $ADFHOME/atomicdata/DFTB/Dresden.

You can also use a different set of parameter files. Note that different sets of parameter files are often not compatible. Note also that often parameter files were designed for a specific purpose, which may be different than your application, and therefore may give not the desired accuracy.

The DFTB implementation shipped by SCM provides the most up-to-date parameter sets available on the DFTB.org website. Additional licensing requirements may however be needed to access the content of the files. Please contact our licensing department to evaluate the available options.

The following sets are currently shipped

- mio-0-1 and mio-1-1 (H, C, N, O, S, P): for organic molecules
- pbc-0-3 (Si, F, O, N, C, H, Fe): for solid and surfaces
- matsci-0-3 (Al, Si, Cu, Na, Ti, Ba): for various compounds in material science

In addition, the following extension sets are provided to the mio set (either version 0-1 or 1-1):

- hyb-0-1 (Ag, Ga, As, Si) + mio-0-1: for organic and inorganic hybrid systems
- chalc-0-1 (As, S) + mio-0-1: for Chalcogenide glasses
- miomod-hh-0-1 + mio-1-1: contains a modified parameter set for H2
- miomod-nh-0-1 + mio-1-1: contains a modified parameter set fo N-H to improve N-H binding energies
- tiorg-0-1 (Ti-(C, H, N, O, S, Ti)) + mio-0-1 : for Ti bulk, TiO2 bulk, TiO2 surfaces, and TiO2 with organic molecules
- trans3d-0-1 (Sc, Ti, Fe, Co, Ni)) + mio-0-1: Transition metal elements for biological systems
- znorg-0-1 (Zn-(C, H, N, O, S, Zn)) + mio-0-1: for Zn bulk, ZnO bulk, ZnO surfaces, and ZnO with organic molecules

We recommend to visit the DFTB.org web site for more detailed information about each set. Please note that our implementation of DFTB does not support parameter sets files containing f-functions, such as the "rare" set.

Installing additional DFTB.org parameter files

To install new parameter sets released by the DFTB.org website in the future, we recommend the following procedure

1. Unpack the tar.gz file containing the parameters (for example, newset-0-1.tar.gz) with the command `tar -C $ADFRESOURCES/DFTB/DFTB.org -xzvf newset-0-1.tar.gz`.
2. Make sure the files have the name in the format X-Y.skf, with X and Y element symbols (for example, C-C.skf, C-H.skf, Al-H.skf). If this is not the case, rename the files to follow this naming.
3. Take note of the new directory name created in $ADFRESOURCES/DFTB/DFTB.org while unpacking (for example, newset-0-1)

The new parameter set can be now specified with the key ResourcesDir

```
ResourcesDir DFTB.org/newset-0-1
```

Third Order parameter files

The parameter files for third-order evaluation are available under a separate license agreement. Contact our licensing department for more information. Third-order parametrization uses the values classified as "DFTB3 fit" in the reference paper (Gaus, Cui, and Elstner). The special parameters for DFTB3 (Hubbard derivatives and Zeta) are defined in file *metainfo.yaml* (see below).

metainfo.yaml

There should be a file named *metainfo.yaml* in each resources directory (see ResourcesDir), for example *DFTB.org/3ob-1-1/metainfo.yaml*. The file is in accordance with the YAML syntax convention. It contains necessary information required for DFTB calculation. An example with explanation comments (beginning with #) is shown here:

```
# version info
metainfo_version: 1

# SCC models the current resources directory supports ("thirdorder" is aliased to "dftb3"):
supports: [ thirdorder ]

# Hubbard derivatives for DFTB3, see DOI: 10.1021/ct300849w
# note: the list must be { } rather than [ ]
Hubbard_derivatives:
    zeta: 4.000
    parameters: { H: -0.1857, C: -0.1492, N: -0.1535, O: -0.1575 }

# dispersion section
dispersion:
# default method to use, if none specified in the input script
    default: UFF
    UFF:
    D2:
# D3 parameters fitted by S. Grimme, see DOI: 10.1021/ct300849w
    D3-BJ:
        parameters: { s8: 3.2090, a1: 0.7461, a2: 4.1906 }
        factor    :   1.000
    D3-zero:
        parameters: { s8: 0.0000, rs6: 1.1372 }
        factor    :   1.000

# references
reference: |
    M. Gaus, A. Goez, M. Elstner
    "Parametrization and Benchmark of DFTB3 for Organic Molecules"
    J. Chem. Theory Comput. 2013, 9 (1), pp 338-354.

# references in short format
short_reference: |
    [O-N-C-H] J. Chem. Theory Comput. 2013, 9, 338-354.

# web link
url: http://www.dftb.org/parameters/download/3ob/3ob_1_1/
```

```
# Slater-Koster files: "txtn" indicates encrypted, "txt" indicates unencripted
format: txtn
```

References

DFTB: general description

M. Elstner, D. Porezag, G. Jungnickel, J. Elsner, M. Haugk, T. Frauenheim, S. Suhai, and G. Seifert, *Self-consistent charge density functional tight-binding method for simulation of complex material properties*, Physical Review B **58**, 7260 (1998)

T. Frauenheim, G. Seifert, M. Elstner, Z. Hajnal, G. Jungnickel, D. Porezag, S. Suhai, and R. Scholz, *A self-consistent charge density-functional based tight-binding method for predictive materials simulations in physics, chemistry and biology*, Physica Status Solidi (b) **217**, 41 (2000)

M. Elstner, T. Frauenheim, E. Kaxiras, G. Seifert, and S. Suhai, *A self-consistent charge density-functional based tight-binding scheme for large biomolecules*, Physica Status Solidi (b) **217**, 357 (2000)

C. Koehler, G. Seifert, U. Gerstmann, M. Elstner, H. Overhof, and T. Frauenheim, *Approximate density-functional calculations of spin densities in large molecular systems and complex solids*, Physical Chemistry Chemical Physics **3**, 5109 (2001)

T. Frauenheim, G. Seifert, M. Elstner, T. Niehaus, C. Kohler, M. Armkreutz, M. Sternberg, Z. Hajnal, A. di Carlo, and S. Suhai, *Atomistic Simulations of complex materials: ground and excited state properties*, Journal of Physics: Condensed Matter **14**, 3015 (2002)

M. Gaus, Q. Cui, and M. Elstner, *DFTB3: Extension of the Self-Consistent-Charge Density-Functional Tight-Binding Method (SCC-DFTB)*, Journal of Chemical Theory and Computation **7**, 931 (2011)

DFTB: parameter sets

Dresden

The DFTB parameter files in $ADFHOME/atomicdata/DFTB/Dresden are distributed with the ADF package. For more detailed information, see also the README file in the directory $ADFHOME/atomicdata/DFTB/Dresden.

General reference for the construction of all integral tables in $ADFHOME/atomicdata/DFTB/Dresden:
J. Frenzel, A. F. Oliveira, N. Jardillier, T. Heine, and G. Seifert, *Semi-relativistic, self-consistent charge Slater-Koster tables for density-functional based tight-binding (DFTB) for materials science simulations*, TU-Dresden 2004-2009.

For construction and application of integral tables for Al-O-H:
J. Frenzel, A. F. Oliveira, H. A. Duarte, T. Heine, and G. Seifert, *Structural and electronic properties of bulk gibbsite and gibbsite, surfaces*, Zeitschrift für Anorganische und Allgemeine Chemie **631**, 1267 (2005)

For construction and application of integral tables for Al-Si-O-H:
L. Guimarães, A. N. Enyashin, J. Frenzel, T. Heine, H. A. Duarte, and G. Seifert, *Imogolite Nanotubes: Stability, electronic and mechanical properties*, Nano **1**, 362 (2007)

For construction and application of integral tables for Al-O-P-C-H:
R. Luschtinetz, A. F. Oliveira, J. Frenzel, J. Joswig, G. Seifert, and H. A. Duarte, *Adsorption of phosphonic and ethylphosphonic acid on aluminum oxide surfaces*, Surface Science **602**, 1347 (2008)

For construction and application of integral tables for Ti-O-P-C-H:
R. Luschtinetz, J. Frenzel, T. Milek, and G. Seifert, *Adsorption of phosphonic acid at the TiO2 anatase (101) and rutile (110) surface*, Journal of Physical Chemistry C **113**, 5730 (2009)

DFTB.org

For detailed information please visit the official DFTB webpage: www.dftb.org. Detailed references of each specific parameter set are available in the corresponding *metainfo.yaml* file (See Section 'ResourcesDir' and 'metainfo.yaml' in this manual).

The newest parameter set (H, C, N, and O) for DFTB3 (DFTB with third-order correction to SCC) *DFTB.org/3ob-1-1*:
M. Gaus, A. Goez, and M. Elstner *Parametrization and Benchmark of DFTB3 for Organic Molecules*, Journal of Chemical Theory and Computation **9**, 338 (2013)

Dispersion

UFF

A. K. Rappé, C. J. Casewit, K. S. Colwell, W. A. Goddard III, W. M. Skiff, *UFF, a full periodic table force field for molecular mechanics and molecular dynamics simulations*, Journal of the American Chemical Society **114**, 10024 (1992)

D2

S. Grimme, *Accurate description of van der Waals complexes by density functional theory including empirical corrections*, The Journal of Computational Chemistry **25**, 1463 (2004)

S. Grimme, *Semiempirical GGA-type density functional constructed with a long-range dispersion correction*, The Journal of Computational Chemistry **27**, 1787 (2006)

D3

S. Grimme, J. Antony, S. Ehrlich, and H. Krieg, A consistent and accurate ab initio parametrization of density functional dispersion correction (DFT-D) for the 94 elements H-Pu, The Journal of Chemical Physics **132**, 154104 (2010)

S. Grimme, S. Ehrlich, and L. Goerigk, Effect of the Damping Function in Dispersion Corrected Density Functional Theory, Journal of Computational Chemistry **32**, 1457 (2011)

Keywords