comatsci
computational **mat**erials **sci**ence utility package
Version 1.2.0

Jan M. Knaup, Bremen Center for Compuational Materials Science
Jan.Knaup@bccms.uni-bremen.de

January 18, 2012

Contents

1 Introduction **2**
1.1 Requirements . . . 2
1.2 Installation . . . 2
1.3 License . . . 2

2 Simulation Programs **2**
2.1 Pastafarian . . . 2
2.2 fitrep . . . 4

3 Utilities **5**
3.1 Single Geometry Utilities . . . 5
3.2 Path Utilities . . . 8
3.3 Generic Utilities . . . 10

4 Calculators **12**
4.1 DFTB$^+$/Noodle . . . 13
4.2 SIESTA . . . 14
4.3 Gaussian (03) . . . 15
4.4 Müller-Brown potential . . . 15
4.5 Pair-potential Calculator for E_{rep} fitting . . . 15

5 Embedding workflow **15**
5.1 Constructing an embedded cluster . . . 16
5.2 Fitting of embedding parameters . . . 17
5.3 Validating an embedded cluster . . . 18

6 Known geometry file formats **18**
6.1 .xyz (read/write) . . . 18
6.2 .gen (read/write) . . . 19
6.3 .fmg (read/write) . . . 19
6.4 .pdb (write only) . . . 22
6.5 .xyzq (write only) . . . 22
6.6 .tm (write only) . . . 22
6.7 .fdf (write only) . . . 22

A Non-Profit Open Software License ("Non-Profit OSL") 3.0 **22**

1 Introduction

1.1 Requirements

comatsci requires the following software/libraries to be present on a system:

- Python version $\geq$ 2.4 (not comatible with python 3)
  - POSIX compatible build environment and header files to compile python c-extensions
- numpy
- ElementTree (included in Python version $\geq$ 2.5)

Note that windows is not an officially supported platform for comatsci.

1.2 Installation

comatsci should be installed via the supplied setup script. Execute `python setup.py install` as root, to install comatsci system-wide. Any other installation tree can be chosen with the `--prefix` option to install, e.g. `--prefix=~` to install into `$HOME/bin` and `$HOME/lib`. The setup script also offers options to generate installers for different operating systems, depending on the platform and installed python version, please refer to the integrated documentation available by calling `python setup.py --help`.

Attention!

On some x86_64 platforms, libraries are separated into `lib` and `lib64` directories. In such a case, it is important to make sure, that pure python libraries are installed into the same `lib*` tree as platform dependent packages. Otherwise, extensions to comatsci-basic may not be found by the python interpreter. Cf. the `--install-platlib`, `--install-purelib` and `--install-lib` options to setup.py.

1.3 License

comatsci is provided without any warranty as open software under the terms of the Non-commercial Open Software License v3.0. Please refer to the attached LICENSE file or appendix A.

2 Simulation Programs

2.1 Pastafarian

The *Path Search Tool bAsed on Flexible Atomistic Reaction Image ANalysis* searches for a chemical transition path using the *Nudged Elastic Band* method with various extensions.

Command Line Options The general command line for pypath is as follows:
`pastafarian [options] <start geometry> [<intermediate geometries>] <end geometry>`
or
`pastafarian [options] <path file>`
where `<path file>` is a reaction path or NEB checkpoint in .fmg format.
The possible command line options are:

-h, –help show this help message and exit

-m N, –maxit=N Maximum number of NEB iterations for this run, default=100

-r RMS, –rmstol=RMS stop iterating if path RMS force in a.u. falls below RMS, default=1e-05

-f FMAX, –forcetol=FMAX stop iterating if path maximum of atomic forces in a.u. falls below FMAX, default=0.0001

-c CALC, –calculator=CALC use CALC to calculate energies and forces, default=dftb

-x LIST, –fixedatoms=LIST Keep atoms in comma delimeted LIST fixed, default=None

-s D_T, –stepwidth=D_T Use D_T as stepwidth for velocity Verlet relaxation, default=0.1

–charge=Q Set the system total charge to Q. Given in units of electrons, negative charge means electron excess, default=0

-v Increase verbosity level, default=1

-q, –quiet Limit output to fatal errors and critical warnings, no status output at all

–silence Limit output to fatal errors only. Not even warnings.

-d SCHEDULER, –scheduler=SCHEDULER Choose scheduler to manage execution of single calculations. Choices are 's': serial, 'p': MPI parallel. default=s

-t MAXTHREADS, –maxworkthreads=MAXTHREADS Set maximum number of concurrent parallel threads in thread scheduling mode. 0:= no limit. default=0 (currently unused)

–pathdebug Write individual .xyz and .fmg path files for each NEB step and write some energy and forces data to pathDebug.nrg for data debugging purposes

Default values for all command line options are overridden by the config file, `pastafarian --help` shows the default values actually used.

Experimental

MPI scheduling requires an installed pypar 1.9.2 or higher package and execution of the pastafarian script in an mpi environment, i.e. via mpirun or similar. *Caveat: currently the detection of MPI master or slave status is performed by evaluation of the LAMRANK or MPIRANK environment variables. This may break, depending on your installation!*

Config File The PaSTaFARIAn config file has the name **pypath.ini** and is in the form of a windows .ini type file. (Extensions in Windowx XP .reg files are not supported.) The config file contains one section labeled **[comatsci]**, which contains all configuration variables for the path search itself and possibly several sections with configuration variables for the different calculators. The calculator configuration sections are described int he Calculators section of this document.

All settings from the **[comatsci]** section of the config file are overridden by the respective command line options, if such are given.

The Configuration variables are:

`maxit=100` maximum number of NEB iterations

`rmstol=2e-4` rms force convergence criterion

`forcetol=2e-3` max force convergence criterion

`calculator=dftb` calculator to use, currently dftb, noodle (=dftb+), gaussian and siesta are supported

`stepwidth=0.25` velocity verlet stepwidth

`fixedatoms=[]` put a comma separated list of fixed atoms here

`charge=0` set the system total charge to Q. Given in units of electrons, negative charge means electron excess.

Additionally, the section **[NEB]** stores NEB algorithm-specific settings

`fmode=s` force calculation mode:

- **s** standard NEB forces
- **c** climbing image NEB forces

`relmode=v` relaxation mode:

- **v** projected velocity verlet
- **s** (constant displacement) steepest descent

`tangmode=s`

- **s** use standard NEB tangents
- **w** use weighted tangents after J. Chem. Phys. **113**, 9978 (2000). Can lead to NEB force discontinuities between iterations if used on paths far from the PES valley. *Attention:* Overshoot detection is disabled when using weighted tangents.
- **p** use spline tangents from a parametric cubic spline representation of the path. **Caution** highly experimental.

Output files PaSTaFARIAn writes most of its output to stdout. Additionally, the following files are written:

energies.dat: This file contains the reaction path energy profile for each NEB cycle that was performed in a format suitable for gnuplot. Column one contains the image sequence number, col. 2 contains the total energy in H. NEB cycles are separated by blank lines.

path.xyz: This file contains the last reaction path geometries calculated as a multi-frame .xyz file. This is mainly intended to be able to conveniently visualize the path using standard molecular visualization software.

checkpoint: This directory contains the last reaction path calculated in the form of .gen files named path-X.gen, where X is in the range of $0 \ldots 1$, reflecting the image sequence. This can be used to restart an interupted calculation by calling `pypath.py [options] checkpoint/*`. Old files are not deleted, so be sure to check the contents of this directory before using wildcards!

checkpoint.fmg: The last reaction path geometries in a single, multi-geometry fmg file. Suitable for use as a restart file. This is intended to replace the **checkpoint** directory in the future.

checkpoint.xyz: The last reaction path in a single, multi-frame .xyz file. Intended for convenient visualization.

Additionally, calculators may create further files in additional directories, e.g. for data reuse purposes.

2.2 fitrep

Tool for evolutionaly fitting of repulsive potentials. Caveat: This process has many opportunities for error, consult the authors before attempting E_{rep} fitting.

Command line: `fitrep [options] [<repulsives>] [<targets>] [<ranges>]`
repulsives: optional filename of repulsives definition file, default = "repulsives" targets: optional filename of targets definition file, default = "targets"

Options:

-h, –help show this help message and exit

-o OPT, –optimizer=OPT Select which optimizer to use for fitting the repulsive potential. Default=genetic

-v Increase verbosity level,

-q, –quiet Limit output to fatal errors and critical warnings, no status output at all

–silence Limit output to fatal errors only. Not even warnings.

–optdebug Optimization debug: write various files recording the optimization history. *OUTPUT FILES ARE LARGE*

common options: options common to all optimizer types

-m N, –maxit=N Maximum number of iterations, default = 1000

-u M, –mutation=M Mutation factor, default = 0.1

-t M, –mutator=M Method used to mutate specimens, choices are local or global, default=global

gentic optimizer options: options only relevant to genetic optimization

-p POP, –population=POP Total number of E_rep sets to use, default = 100

-b B, –breeders=B Total number of E_rep sets to choose for the breeder group, default = 20

–keep-elders If selected, non-breeders fitter than offspring will survive into next generation, default = False

-c C, –combiner=C Method used to combine parents, either lr or rnd, default=lr

–initialmutation=M Mutation factor, default use same factor as for evolution

–initialmutator=M Method used to mutate specimens, either local or global or same as for evolution, default=same

3 Utilities

3.1 Single Geometry Utilities

3.1.1 geoconv

Geoconv.py serves to quickly convert geometries between different formats known to comatsci. It is can called as geoconv.py or via symbolic links named to[format], where [format] is the specified output format (cf. -f option). geoconv only works on single geometries but not on paths. Known formats are described in section 6.

Command lines are of the format

`<commandname> [options] <inputfilename> [output filename]`

The input file type is determined by the file name extension. If the output file name is ommitted, the program writes a file of the same name as the input file, with the extension replaced appropriately.

Possible command line options are:

-h, –help show this help message and exit

-f F, –format=F Write output geometry in format F. Default=gen. Choose from : ['fdf', 'xyz', 'xyzq', 'gen', 'tm', 'fmg', 'pdb']

-x XTND, –extend=XTND periodically extend the input geometry to include a:b:c original supercells in the a:b:c lattice directions, default = 1:1:1 (only the original supercell)

-p F, –population=F Read dftb Mulliken population from file F. No default.

-l L, –layer=L Write Only Atoms from Layer L into output geometry. default: write whole geometry

-e E, –element=E Write only atoms of element E into output geometry. default: write all elements

-s S, –scale=S scale the whole geometry by a factor of S.

-a F L, –append=F L Append geometry from file F to input geometry into layer named L. L *must* be specified, if L ist not present in primary input geometry, it will be created in the output geometry. The special layer name 'default' specifies the .fmg default layer 0.

-F, –foldback Fold periodic geometry back into unit cell.

–chargeconstraints Print charge constraints string for specified atoms

–tolayer=L Move atoms defined by –atomindices or –atomlist to layer L. L must be defined in the geometry, either –atomindices or –atomlist must be given. Cannot be combined with -L.

–translate=V Translate atoms defined by –atomindices or –atomlist by vector V. V must be in the form x:y:z. Either –atomindices or –atomlist must be given.

–atomindices=IDX Declare, which single atoms are to be modified, argument is a whitespace delimited list of integer atom index numbers. Counts from 0. Mutually exclusive with –atomlist.

–atomlist=LST Declare, which single atoms are to be modified, argument is a whitespace delimited list of integer atom serial numbers. Counts from 1. Mutually exlusive with –atomindices.

–addlayer=L Add layer with name L to geometry.

3.1.2 to[format]

Aliases of geoconv, allowing to quickly convert a readable input geometry to the [format] file format. Aliases are:

- togen
- tofmg
- toxyz
- tofdf
- toxyzq
- totm

All options for geoconv also apply to the to[format] aliases.

3.1.3 scale-linkdists

This helper script reads an input geometry and writes out a .gen file, after applying a distance scaling factor to the bonds of a block of atoms at the end of th einput geometry.

Command line: `scale-linkdists [input filename] [output filename] [# of non-link atoms] [distance factor]`
none of the positional arguments is optional. The number of non-link atoms specifies the length of the block of atoms at the beginning of the geometry which will not be moved.

3.1.4 coordination-check

THis helper utility reads a given input geometry, constructs the bond list and writes a pdb file in which the beta column contains the difference betweent the atomic number of covalently bonded neighbors and the canonical number of valences. Only works for elements H to Ca!.

Command line: `coordination_check [options] <input file> [<output file>]`
`if no output file name is specified, the output will be named <input basename>.pdb .`

Options:

-h, –help show this help message and exit

-l L, –layer=L Write only atoms from layer L into output geometry. default: write whole geometry

-e E, –element=E Write only atoms of lement E into output geometry. default: write all elements

-t T, –tolerance=T Detect bonds if distance is T times the canonical bond length

3.1.5 dosplot

Reads single particle eigenstates from QM program outputs and creates ascii data suitable for plotting.

Usage `dosplot <eigenstates> [<output file>]`

Options

-h, –help show help message and exit

-f PF, –peak-function=PF Calculate the DOS as a superposition of this type of PFs at the eigenvalues. Can be gauss or lorentz, default=lorentz

-w PF, –peak-width=PW Width parameter of the peak function (HWHM for lorentz, sigma for gauss). default=0.3

-s SW, –step-width=SW Sample the DOS-s at every SW eV. default=0.1

–min=E set lower bound of DOS range to E. default: lowest eigenvalue

–max=E Set upper bound of DOS range to E. default: highest eigenvalue

–correlate=F Correltate DOS to reference DOS from file F.

Output Prints DOS data to stdout our specified output file. File whitespace separated columns are:

energy [eV]	total DOS	occupied DOS	unoccupied DOS

3.2 Path Utilities

3.2.1 pathprepare

This utility is used to construct input reaction paths for *pastafarian* and *pesto* calculations.

Command Line: pathprepare [options] <start> [<intermediates>] <end>
or : pathprepare [options] <pathfile>

Options:

-h, –help show this help message and exit

-t T, –filetype=T Filetype specification for input files.

-i N, –interpolate=N linearly interpolate N steps between each starting structure read from input files, default: no linear interpolation. Cannot be combined with -s, -r or -e

-s N, –spline-resample=N spline resamplepath to contain a total of N images, default: no spline resampling. Cannot be combined with -i, -e or -r

-e N, –equispline-resample=N spline resample path to contain a total of N images at equal arc length spacing, default: no equidistant spline resampling. Cannot be combined with -i, -s or -r.
Note that equidistant spline resampling can take considerable ammounts of time because of the necessary mapping between spline variable and atomic displacement vector length.

-c C, –calcinputs=C write inputs for energies and forces calculations using calculator C. Writes into subdirectories of ./calcinputs

–inconsistent Do not check subsequent geometries for consistency with preceeding one. Useful for creating targets of isodesmic reactions etc. Default: check for consistency.

-r N, –rener-resample=N resample path by Renner Subsplines to contain a total of N images at equal arc length spacing, default: no Renner subspline resampling. Cannot be combined with -i, -s or -e

Output files:

preppath.xyz The prepared reaction path in multi-frame .xyz format.

preppath.fmg The prepared reaction path in .fmg format. This is the preferred input for *pastafarian.*

preppath A directory containing the path as a series of sequentially numbered files in .gen files. May also contain neb.nrg and neb.frc files, containing the image energies and forces from previous NEB runs respectively.

3.2.2 pathprops

Utility to evaluate properties of a *pastafarian*, *pesto* or molecular dynamics simulation.

Command line: pathprops [options] <input path>

Options:

-h, –help show this help message and exit

-p N, –polyint=N Write a cubic polynomial interpolation of N energy values along the path to cubic.nrg file

-u N, –polypath=N Write a cubic polynomial interpolation of N path images to cubic.fmg and cubic.xyz files

-r, –rmsds Write Atomic displacement RMSDs along path to rmsds.dat and rmsds.pdb files

-c DIR, –deltacharges=DIR Write atomic charge deltas from DFTB charge files in directroy DIR to deltas.chr file –maxdeltacharges=DIR Write maximum atomic charge deltas from DFTB charge files in directroy DIR to maxdeltas.chr and maxdeltas.pdb files

-d CNT, –centerdists=CNT Write atomic distances in last image from atom CNT to file center.dst

–ereptarget=DFTBCPT Write an E_rep fitting target path checkpoint into 'target', based on the fit method path in <checkpoint directory> and the null-spline DFTB path in DFTBCPT. -e C,

–external-calcs=C read and parse outputs from external calculations using calculator C. Writes into subdirectories of ./externalcalcs

–dimer=N extract a dimer with a separation of 1.0 Bohr at image N and write to 'extracted_dimer.fmg'.

–inconsistent Do not check subsequent geometries for consistency with preceeding one. Useful for creating targets of isodesmic reactions etc. Default: check for consistency.

–lattice=LATTICE Set geometry Mode to 'S'(upercell) and apply the provided lattice vectors. Lattice vectors are provided as a whitespace separated list of 9 carthesian coordinates.

–bondcounts Write Element-Element bond counts per image to 'bondcounts.dat' file.

-v Increase verbosity level, default=*normal* verbosity. Higher levels are *talky*, *debug1* and *debug2*.

-q, –quiet Limit output to fatal errors and critical warnings, no status output at all

–silence Limit output to fatal errors only. Not even warnings.

–coordinations write coordination deviations from ideal into beta column pathcoordinations.pdb file and a coordinations analysis into pathcoordinations-[element].dat files

–anglehists=N write by central element bond angle histogram trajectories with N bins. If N is negative, -N bins from 0 to 180 degrees are used.

–anglestats write per central atom type bong angle statistics trajectories.

–lengthhists=N write per bond type bond length histogram trajectories with N bins. If N is negative, binning starts at 0, otherwise at minimum bond length -0.1 a.u.

–lengthstats write per bond type bond length statistics trajectories.

–trackvacancy=R locate vacancies in every single frame by comparison to reference geometry read from file R. Write vacancy postions for each frame to 'vacancies.xyz'.
Note that for vacancy tracking in .xyz format trajectories of MD runs, it is necessary to provide lattice vectors for the trajectory path via the –lattice option.

–vmd-charges write atomic charges to 'charges.vmd' suitable to be read into vmd user data fields

–vmd-bondcounts write per atom covalent bond counts to 'bondcounts.vmd' suitable to be read into vmd user data fields

–count-hops count atomic hopping events and print number of hops to standard out

Output files: see option descriptions.

3.3 Generic Utilities

3.3.1 multiaverage

Calculate averages of Data in ascii files of same column and line counts. If more than one datafile is specified, first calculate ensemble average over files. Other averages available via options.

Command line `multiaverage [options] datafile [[datafile], ...]`

Options:

-h, –help show help message and exit

-r N, –running-average=N Calculate running average of window size N over lines of input data. default: no running average.

-i N, –ignore columns=N Ignore N leading columns of input data. default: ignore no input columns.

3.3.2 splresample

Resample two column ascii data using cubic spline interpolation.

Command line `splresample.py <infile> <npoints> [<xmin>] [<xmax>]`

infile two column input data

npoints number of output data points

xmin start of output abscissa interval

xmax end of output abscissa interval

3.3.3 splderive

Calculate derivative of two column ascii data using cubic spline interpolation.

Command line `splderive.py <infile> <npoints> [<xmin>] [<xmax>]`

infile two column input data

npoints number of derivative output data points

xmin start of output abscissa interval

xmax end of output abscissa interval

3.3.4 chargeanalys-2D

Reads the outputs of a 2-D map of DFTB outputs and, calculated RMS and absolute charge deviations of the QMZ core atoms and outputs data in gnuplot-friendly 2D mapped data files. Command line:
`chargeanalys-2D [reference geometry]`
The reference geometry must be a .fdf file containing only the QM zone, with the QML atoms as one block at the end of the geometry and the QML atoms marked as subtybe "H_l". (This is the output of QM/MM embedding done via comatsci).

input files:

reference geometry chargeanalys-2D expects the reference geometry to contain atomic charges. Hence it must be in .fdf format, as it is the only supportef format for reading that supports this feature.

calculation results DFTB$^+$ calculation results are expected. Currently detailed.out files from DFTB versions 1.0 or 1.1 are supported. The files are expected to be named `detailed-[x]-[y].out`, as usual, gzip compressed files can be privided instead, which is highly recommended for the mapping calculation results. Conventionally, [x] should be the Gaussian blur width and [y] the link atom distance factor, but the meaning of the mapping parameters is of no practical interest to chargeanalys-2D.

output files:

dQ.dat Summed charge deviation of the QM zone atoms (QML excluded) mapped over the embedding parameters.

RMSQ.dat RMS charge deviation of the QM zone atoms (QML excluded) mapped over the embedding parameters.

3.3.5 dosanalys-2D

Reads the outputs of a 2-D map of DFTB outputs and writes overall band shift, maximum cross-correlation and, if requested, the state sum in a band-shifted energy range to gnuplot friedly 2D scalar map files. Command line:
`dosanalys-2D {options} [reference spectrum]`

options:

-h, –help show this help message and exit

-f PF, –peak-function=PF Calculate the DOS as a superposition of this type of PFs at the eigenvalues. default=lorentz

-w PF, –peak-width=PF Width parameter of the peak function (HWHM for lorentz, sigma for gauss). default=0.3

-s SW, –step-width=SW Sample the DOS-s at every SW eV. default=0.1

–min=ESet lower bound of DOS range to E. default: lowest eigenvalue

–max=E Set upper bound of DOS range to E. default: highest eigenvalue

–shifted-range-states=EL,EH calculate shifted DOS integral difference, between EL and EH, shifted by cross-correlation maximum position

input files:

reference spectrum The reference spectrum must be a band.out file from DFTB$^+$.

calculation results DFTB$^+$ calculation results in band.out format are expected. The files are expected to be named `band-[x]-[y].out`, as usual, gzip compressed files can be privided instead, which is highly recommended for the mapping calculation results. Conventionally, [x] should be the Gaussian blur width and [y] the link atom distance factor, but the meaning of the mapping parameters is of no practical interest to dosanalys-2D.

output files:

bandshifts.dat total shift of the band structured determined from the peak in cross-correlation between calculation result and reference DOS.

maxcorrelation.dat maximum cross-correlation value for each mapping point.

shiftedrangestated.dat state sum in the shifted energy range. Cf. **–shifted-range-states** option, only written if this option is supplied.

3.3.6 dosanalys-3D

Reads the outputs of a 2-D map of DFTB outputs and writes all calculated DOS values into an ASCII legacy format vtk file, suitable for vtk and various data visualizers based on that. Command line:
`dosanalys-3D {options}`

options:

-h, –help show this help message and exit

-f PF, –peak-function=PF Calculate the DOS as a superposition of this type of PFs at the eigenvalues. default=lorentz

-w PF, –peak-width=PF Width parameter of the peak function (HWHM for lorentz, sigma for gauss). default=0.3

-s SW, –step-width=SW Sample the DOS-s at every SW eV. default=0.1

–min=E Set lower bound of DOS range to E. default: lowest eigenvalue

–max=E Set upper bound of DOS range to E. default: highest eigenvalue

output files:

DOSvalues-3D.vtk Calulated Density Of States in .vtk format. Data is organized as a 3D rectilinear grid of scalars.

4 Calculators

comatsci-barrier is designed to support any atomic simulation package as a calculator, as long as the package provides total energies and atomic forces in Cartesian coordinates. Adding another calculator is done by subclassing an abstract calculator class, implementing a few methods which write geometry and control input files, run the calculation binary and read the results. Currently, the following calculators are implemented and reasonably stable:

4.1 DFTB$^+$/Noodle

The noodle calculator uses DFTB$^+$ to calculate energies and forces. It has been tested with Version 0.2p0 and uses the new hsd input parser. Most of the dftb parameters are set via a parameter file included into the main hsd input via an <<+ or <<! directive. Parameters, which are to be set via the **[NOODLE]** section of the pypath.ini file are:

binary=/usr/local/bin/noodle full path to the noodle binary. Must be accessible on all nodes in parallel calculations

workdir=TEMP Name prefix for the directories in which to run the dftb+ calculations. The special Value "TEMP" uses mktmpdir to create a temporary directory in the path specified by the $TMPDIR environment variable.

skdir=/home/knaup/SlKo/ifam/alsicnoh-mavo full path to the SK files. The SK file naming convention follows the dftb calculator. This may change with future versions of noodle!

chrdir=charges name of the directory in which noodle restart files are stored

rchr=t read charges from last NEB iteration pypath checks whether the charge file is present, so it's safe to say t here

paraminclude=params.ndl A .hsd or .xml noodle input file which is included at the beginning of the input.hsd. System Charge, SK Files, LMAX and some options are overridden by automatically generated entries, so setting them here is useless.

oldSKnames=true use old Slater-Koster file naming convention, rather than the new one. *Defaults to true for Version 1.0, this may change in the future!*

All the usual caveats for dftb calculations apply here.

The directory specified in chargesdir must exist, even if charges are not reused! When in doubt remove the charges directory, damaged or incomplete charges files will lead to wrong results and may or may not cause error messages.

Slater-Koster file names

In the old convention, the dftb calculator expects to find SK files named by concatenating the lowercase element symbols for each interaction in the SK directory. E.g. for s C,O,H system, the file names would be `cc`,`co`, `ch`, `oc`, `oo`, `oh`, `hc`, `ho`, `hh`. Both files for must always be present, even for fully symmetric interactions. In such cases, use symlinks.

The new naming convention for SK files is to join the capitalized Element names by a dash `-` and add the extension `.skf`. The files from the example above would thus be called `C-C.skf`, `C-O.skf` `C-H.skf...`, a file for the Si – Na interaction would be called `Si-Na.skf`.

Point charges

The Noodle calculator supports the point charges external electric field option of noodle. There are two possible ways to employ it:

1. The point charges specification can be added to the noodle parameter include file. This requires no further intervention or even awareness of the calculator. The approach can be cumbersome for the user, as it requires the manual generation of the point charges to the parameter file. It also counteracts the possibility to use a single parameter file for a whole series of calculations to ensure consistency.

2. The internal Geometry representation supports grouping atoms into layers and the selection of subgeometries based on atomic layer assignment. Layers are identified by a unique integer layer index. Currently the Layer object only contains a Name for the layer. Layer names need not be unique. If the geometry object passed to the noodle calculator contains a layer named "PCHR", the calculator automatically adds an external field/pointcharges specification, adding the atoms from the "PCHR" layer as point charges. *If several "PCHR" layers exist, only on of them is used, the choice is not strictly determined!*

DEPRECATED

In this case, only the atoms from the default layer "0" are added to the noodle geometry input. This means that all atoms not contained in the default layer or the "PCHR" layer will be ignored.

Currently, geometry layers and atomic charge specification are only supported via the .fmg file format, cf. comatsci documentation. The geostats.py tool of comatsci provides a more user friendly interface to edit these properties than a plain text editor does.

This way to specify external charges is deprecated, because in most situations tens of thousands of point-charges are necessary, which have to be kept in memory for each image along the path. Additionally, the computational effort to separate the layers in geometry of such size is considerable and may lead to significant overhead.

Both methods can be mixed, as the automatic point charges specification is added to any possible point charges spec from the parameter file.

Be aware that this is by no means an automatic QM/MM scheme. The user is responsible for providing link atoms and specifying the necessary geometry constraints.

4.2 SIESTA

The siesta calculator uses SIESTA to do the energy and forces calculations. It has only been tested with v1.3 and v1.3-f1p, but older versions might work as well. Most of the SIESTA parameters are set in an .fdf file, which must be supplied by the user and is includes into the siesta input using an "%include" directive. The Parameters to be set in the **[SIESTA]** section of the pypath.ini file are:

binary=/usr/local/bin/siesta Full path to the SIESTA binary.

ppdir=/home/user/siesta-pseudo Directory containing the pseudo potential files to use for the SIESTA calculations.

workdir=TEMP Name prefix for the directories in which to run the SIESTA calculations. The special Value "TEMP" uses mktmpdir to create a temporary directroy in the path specified by the $TMPDIR environment variable.

dmdir=DMS Directory in which to store the .DM files from the calculation for the different images. *Warning: If the basis set has changed, the old .DM files cannot be reused. SIESTA will inevitably fail.*

rdms=t Reuse the .DM file from the previous calculation on the same image. This should usually bet set to "t" as it speeds up the calculations enormously. It is safe to to say "t" here, if no old .DM files are present.

paraminclude=params.fdf An .fdf file which should contain all calculation parameters not automatically set by pypath. Pypath automatically sets the following parameters, which *must not* be set in this file:

- Atomic Coordinates
- Cell Vectors
- Chemical Species

- NumCGSteps
- any MD parameters
- systemlabel
- %include directives

Besides the usual caveats for SIESTA calculations, the following tip might be useful: SIESTA has difficulties finding a proper density matrix if the atomic configuration is not close to an equilibrium state. This might lead to convergence failure in the vicinity an interpolated intermediate state. In Such cases it is useful, to copy a .DM file from a neighboring image to the image with convergence problems. A converged density map for a similar isoelectronic configuration is usually a much better starting point than SIESTA's initial guess.

4.3 Gaussian (03)

The Gaussian calculator is only tested for Gaussian03. Please consider your Gaussian Manual for extensive documentation on Gaussian use and features.

Parameters for the Gaussian calculator are entered in the [Gaussian] section of pypath.ini. Parameters are:

binary=g03 path to the gaussian binary

workdir=TEMP Name prefix for the directories in which to run the SIESTA calculations. The special Value "TEMP" uses mktmpdir to create a temporary directroy in the path specified by the $TMPDIR environment variable.

chkdir=g03chk directroy in which gaussian checkpoint files will be stored

rchk=false read checkpoint file (**warning:** this is untested and may go horribly wrong)

hamiltonian=BPL/6-31G*/AUTO Hamiltonian part of the route section

routeopts= further options to incorporate into the route section. Do not specify SP, OPT or FREQ statements here!

spinmul=1 spin multiplicity

4.4 Müller-Brown potential

Called as `muellerbrown`. The two-dimensional test potential for reaction path analysis algorithms suggested by **?**]. The potential is dependent on the absolute atomic coordinate in Bohr. Intended for testing and demonstration purposes. Be aware that the potential operates on shorter distances than interatomic potentials, i.e. the gradients are very steep and smaller step sizes are necessary.

Takes no options and has therefore no config file section.

4.5 Pair-potential Calculator for E_{rep} fitting

4.5.1 xyspline

4.5.2 repulsive

5 Embedding workflow

Note that the workflow described in this setion relies on an existing installation of the comatsci GUI geostats. Comatsci-gui is distributed separately to allow smooth installation of comatsci on headless cluster nodes which very often do not provide the necessary gui libraries.

The QM/MM embedding scheme described here only regards the embedding of the QM zone and the termination of the dangling bonds resulting from the partitioning of the extended system into QM and MM zones. Mechanical embedding, coupling on the MM side and simulations based on the QM/MM scheme are not the subject of comatsci embedding or this manual.

5.1 Constructing an embedded cluster

The QM/MM embedding relies on the availability of QM reference data, which has to be calculated in advance.

1. Perform a DFTB reference calculation of the target system. Typically this would be a periodic supercell model. In the reference calculation, the atomic coordinates should be optimized.

2. Obtain a geometry file of the target system in .fdf format, containing atomic charges.
   E.g. by converting the output of a DFTB$^+$ calculation using:
   `tofdf geo_end.gen -p detailed.out QM-reference.fdf`

3. (only if BCTC embedding is intended) Obtain bond charge transfer coefficients for the reference system:
   `geostats QM-reference.fdf`
   select: statistics->save BCT coefficients

4. Periodically extend the reference system to the intended size of the MM Zone:
   `tofdf QM-reference.fdf -x a:b:c fullsystem.fdf`
   a,b,c denote the numbers of reference unitcells that the final supercell should contain. Alternatively, the "edit->peridoc expand" dialog of geostats can be used.

5. Add a QM zone layer named "QMZ" to the .fdf file, to which all atoms that should form the QM zone can be moved.
   `tofdf --addlayer=''QMZ'' fullsystem.fdf`
   Alternatively, add a layer "PCHR" for all atoms that should act as external charges. The layer names "QMZ" and "PCHR" are crucial, naming is case-sensitive.
   Note the layer index of the newly created layer for later use!

6. Obtain a list of atom numbers which will be moved to the QMZ or PCHR layer.
   in VMD console:
   `qmzone=atomselect <molecule number> ''<atom selection string>''`
   `qmzone get serial`

7. Move QMZ or PCHR layers into the respective layer.
   choose one of:
   `tofdf --tolayer <layer index> --atomlist=''<list of atom serial numbers counting from 1>'' fullsystem.fdf partitioned.fdf`
   `tofdf --tolayer <layer index> --atomindices=''<list of atom serial numbers counting from 0>'' fullsystem.fdf partitioned.fdf`
   as appropriate for your atom list.

8. Load the partitioned geometry into geostats and select the desired embedding method from the edit menu.

9. In the embedding dialog, enter the embedding parameters. If you are constructiong a cluster for parameter fitting/validation, set the link atom distance factor to 1.0. Klick "embed" and copy/paste the embedding results displayed in the dialog for your documentation.

10. Save the embedded geometry to a new .fdf file, using the file->save as menu.

11. Save the QM zone atoms to a .gen file as DFTB input
```
togen -l <QMZ layer index> embedded.fdf input.gen
```
If your extended system is periodic, *manually!* Change the geometry mode of input.gen to "c" for cluster and delete the last 4 lines in the file, which describe the periodic boundary conditions.

12. Obtain a list of the serial numbers of atoms in the QMZ input file (input.gen) to keep fixed during the calculations, this should at least include the QMH and QML atoms. Cf. step 6.

13. Save the external charge distribution as an .xyzq file for DFTB input.
```
toxyzq -l <PCHR layer index> embedded.fdf pchr.xyzq
```

14. Add the appropriate external electic field specification to the dftb_in.hsd file for your later calculations. Set the Gaussian blur width as determined during fitting.

The external charges files are independent of the link atom distance factor, however, they depend upon the embedding method and any additional parameters an individual method may use. Make sure, not to re-use inappropriate .xyzq files for newly-embedded clusters!

5.2 Fitting of embedding parameters

To determine the proper Gaussian blur widths of the external charge distributin and the appropriate link-atom distance factor, the self-consitent atomic partial charges and the density of states of the embedded cluster must be mapped with respect to the two embedding parameters and compared to the reference system. Refer to Ref. for an in-depth scientific discussion of the fitting criteria.

The workflow for the parameter mapping is

1. Construct a QM surface cluster as described in section 5.1.

2. Save the QM zone atoms to an .fdf file as reference for the fitting
```
tofdf -l <QMZ layer index> embedded.fdf QMZ-reference.fdf
```

3. Save the quasi single particle eigenvalues from the reference calculation

4. Do test calculations of the QM zone mapped over Gaussian blur widths and link atom distance factors. The necessary mapping ranges may vary between materials. In aluminna, GBW ranges from 0.1...5 a.u. and LADF ranged between $0.5 \ldots 1.2$ have proved reasonable.
In the mapping calculations, set the maximum number of SCC iterations to a high value (e.g. 1000) to avoid convergence problems. Keep all atomic positions fixed at the coordinates from the reference calculation.

5. Map the atomic charge and DOS reproduction parameters
```
chargeanalys-2D QMZ-reference.fdf
dosanalys-2D reference-band.out --min=<minimum energy> --max=<maximum
energy> --shifted-range-states=<reference VBM>,<reference CBM>
```
The energy ranges should be chosen to encompass the valence and conduction bands of the reference calculations with a few eV headroom on both ends to allow for band structure shifts. It may be useful to change the peak function width and sampling step-widths for the DOS analysis from their default values, which are rather coarse.
The DOS analyis is slow, expect runtimes of ~20 minutes to 1 hour for typical mappings.

6. Plot the mapped embedding results and chose a combination of Gaussian blur width and link atom distance factor based on the results. It has proved reasonable to rank the targets in the following manner:

(a) RMS charge deviation

(b) summed charge deviation

(c) band shift

(d) state sum in shifted energy range

7. Perform further validation of the embedded cluster at the chosen parameters (cf. section 5.3).

Mapping script example

(Note that mapping ranges are shortened for readability.)

```
#!/bin/bash
binary=/usr/locla/bin/dftb+_1.1
# i runs over Gaussian blur widths
for i in 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.1 1.2 1.3 1.4
do
# j runs over link atom distance factors
     for j in 0.600 0.625 0.650 0.675 0.700 0.725 0.750 0.775 0.800
     do
          /usr/local/bin/scale_linkdists qmz.gen qmz-moved.gen 100 $j
          echo -n .
          echo "GaussianBlurWidth="$i > gbw.in
          echo -n .
          $binary > gbw-$i-$j.out
          mv detailed.out detailed-$i-$j.out
          mv band.out band-$i-$j.out
          if [ -f results.tag ];
          then
          mv results.tag results-$i-$j.tag
          fi
          echo -n .
          gzip -9 -f *-$i-$j.out *-$i-$j.tag
          echo -n :
     done;
done;
```

5.3 Validating an embedded cluster

Validation is based on qualitative and quantitative checks, how far the embedded cluster deviates from the reference geometry, when atomic positions of the inner atoms (i.e. all but the QML and QMH atoms) are optimized. No bond-braking or formation should be tolerated, the RMS displacement must be as small as possible and within acceptable limits for the intended application.

Further checks of the differences between reaction energies etc. of small reaction between fully QM reference caluculations and the embedded system are advised.

6 Known geometry file formats

6.1 .xyz (read/write)

The widely used xmol format for carthesian corrdinates of atoms in molecules. The first line starts with an integer giving the number of atoms in the file, the second line is a comment line and

ignored. Following lines list the element symbol and x,y,z coordinates in Angstrom of one atom each. If additional fields are encountered, the 5th column is interpreteed as the atomic valence population (number of electrons in the valence shell, not the atomic charge), further columns are ignored. THis follows the convention of DFTB$^+$. In most cases, additional geometries after the first one are ignored. only the pathprepare and pathprops utilities read multi-frame .xyz files as reaction paths.

example: CO molecule

```
2

C 0.0 0.0 0.0
O 1.2 0.0 0.0
```

6.2 .gen (read/write)

Generic molecular and crystalline geometry file format, originally from DFTB implementations. This file format is flexible in its support of cluster and supercell geometries. Please refer to the DFTB$^+$ documentation for a detailed description of the geometry file format.

example: CO molecule

```
# number_of_atoms mode flag
2 C
# chemical symbols of atom types
O C
# number_of_atom type_of_atom cartesian_coordinates_in_angstroms
1 1 0.0 0.0 0.0
2 2 1.2 0.0 0.0
```

4 more lines follow for periodic structures. These contain the coordinates of the origin and three vectors characterizing the supercell.

- The atom type number refers to the ordering of the chemical symbols in line 2. These together determine the element of each atom, other than in dftb.
- The atom numbers (first column of the coordinate lines) are for convenience only but required.
- The flag (first line, second column) is either “C” for clusters/molecules or “S” for supercells.

6.3 .fmg (read/write)

The Flexible Molecular Geometry file format is an xml format intended to store all geometry information that can be processed by comatsci. As an .xml format it is user-editable in principle, however, an xml-aware editor is strongly recommended. Currently fmg provides the following features:

- Supercell and cluster geometries
- Specification of multiple geometries, e.g. for storing reaction paths
- Specification of geometry layers to partition the geometry
- Specification of geometry dimers for use with the dimer method
- Specification of atomic properties:
  - charge

  - layer
  - subtype (e.g. for different force-field atom types within one element)

- Storage of additional trajectory information outside the geometry declaration:
  - Energy
  - Velocities
  - Forces
  - General trajectory information:
    - Number of geometry iterations so far

C.f. appendix B for the xml document type definition of the flexible molecular geometry format.

6.3.1 example: water

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
   <!DOCTYPE fmg>
   <fmg>
   <geometry>
         <mode>C</mode>
         <lattice orgx=''0.0'' orgy=''0.0'' orgz=''0.0'' lunit=''ang''>
               <latvec_a>1.0 0.0 0.0</latvec_a>
               <latvec_b>0.0 1.0 0.0</latvec_b>
               <latvec_c>0.0 0.0 1.0</latvec_c>
         </lattice>
         <layer>
               <lname>default</lname>
               <li>0</li>
         </layer>
         <atom lunit="ang">
               <x>0.98</x> <y>0.0</y> <z>0.0</z>
               <el>1</el>
               <st>H</st>
               <chr>0.0</chr>
               <li>0</li>
         </atom>
         <atom lunit="ang">
               <x>-0.49</x> <y>0.85</y> <z>0.0</z>
               <el>1</el>
               <st>H</st>
               <chr>0.0</chr>
         </atom>
         <atom lunit="ang">
               <x>0.0</x> <y>0.0</y> <z>0.0</z>
               <el>8</el>
               <st>O</st>
               <chr>0.0</chr>
         </atom>
   </geometry>
   </fmg>
```

The following rules apply:

- the `<fmg>` document tag is mandatory

- multiple <geometry> tags are allowed. The geoconv and geostats tools ignore all but the first geometry.
- all attributes are optional
- possible value for lunit are:
  - ang: Angstrom, default value
  - au: Bohr radii
- <mode> is optional, possible values are:
  - C — cluster, default value
  - S — supercell
- <lattice> is optional
  The org[xyz] attributes specify the supercell origin (as in the .gen format) and are mostly ignored
- <lattice> must contain exactly one of each of the following elements:
  - <latvec_a>: first lattice vector, 3 coordinates
  - <latvec_b>: second lattice vector, 3 coordinates
  - <latvec_c>: third lattice vector, 3 coordinates
- <layer> is optional, multiple layers may be specified. <layer> must contain exactly one <lname> and <li> element
- <atom> must contain exactly one of each of the following elements:
  - <x>: x coordinate
  - <y>: y coordinate
  - <z>: z coordinate
  - <el>: element number
- <atom> may contain one of each of the following elements:
  - : layer index, default 0
  - <st>: subtype, default [element symbol]
  - <chr>: charge, default 0.0. Unit is electrons, negative values signify electron excess
  - <lpop>: l-shell populations, default is empty
- <trjstep> is outside the geometry definition. It stores data applying to one single <geometry>. To store data for a geometry trajectory, an equal number of trjstep elements in the same order should be present. It may contain the following subelements:
  - <nrg>: Image total Energy
  - <velocities>: Image Velocities data block
  - <forces>: Image forces data block
- <trjinfo> is outside the geometry definition. It stores general information applying to the whole trajectory, rather than stepwise data, as in <trjstep>. Only one <trjinfo> element should be present in one <fmg> and only the first element encountered is evaluated. subelements:

  - `<stepcount>`: Number of geometry iterations performed so far.

- `<dimer>` must contain exactly one `<geometry>` element, specifying the dimer midpoint. It must also contain exactly one `<DeltaR>` element, specifying the dimer translation vector $\vec{R_1} - \vec{R_0}$. Additionally the following subelements are allowed:

  - `<NoGradInRot />` Boolean specifier not to use gradient calculation in rotation step. *deprecated!*
  - `<curvature>` The curvature in dimer direction
  - `<E0 eunit=''>` Calculated energy at dimer midpoint, possible units are eV and H.
  - `<E1 eunit=''>` Calculated energy at dimer endpoint 1, units as in E0.
  - `<E2 eunit=''>` Extrapolated energy at dimer endpoint 2, units as in E0.
  - `<f0>` Calculated force at dimer midpoint in a.u..
  - `<f1>` Calculated force at dimer endpoint 1 in a.u..
  - `<f2>` Extrapolated force at dimer endpoint 2 in a.u..
  - `<fN>` Projected rotational force on dimer in a.u..

6.4 .pdb (write only)

Protein database format. Refer to PDB documentation for details. .fmg layers are written as segments.

6.5 .xyzq (write only)

Carthesian coordinates and atomic charges in a.u.. Suitable as external electric field input in DFTB$^+$.

example: water

```
1.85193164 0.00000000 0.00000000 0.2938
-0.54141424 1.51278662 0.00000000 0.2938
0.05430593 -0.19438394 0.00000000 -0.5877
```

6.6 .tm (write only)

Turbomole `coord:` block. Refer to Turbomole documentation for details.

6.7 .fdf (write only)

Geometry specification in SIESTA fdf input format. Refer to SIESTA documentation for details.

A Non-Profit Open Software License ("Non-Profit OSL") 3.0

Licensed under the Non-Profit Open Software License version 3.0

1) Grant of Copyright License. Licensor grants You a worldwide, royalty-free, non-exclusive, sublicensable license, for the duration of the copyright, to do the following:

a) to reproduce the Original Work in copies, either alone or as part of a collective work;

b) to translate, adapt, alter, transform, modify, or arrange the Original Work, thereby creating derivative works ("Derivative Works") based upon the Original Work;

c) to distribute or communicate copies of the Original Work and Derivative Works to the public, with the proviso that copies of Original Work or Derivative Works that You distribute or communicate shall be licensed under this Non-Profit Open Software License or as provided in section 17(d);

d) to perform the Original Work publicly; and

e) to display the Original Work publicly.

2) Grant of Patent License. Licensor grants You a worldwide, royalty-free, non-exclusive, sublicensable license, under patent claims owned or controlled by the Licensor that are embodied in the Original Work as furnished by the Licensor, for the duration of the patents, to make, use, sell, offer for sale, have made, and import the Original Work and Derivative Works.

3) Grant of Source Code License. The term "Source Code" means the preferred form of the Original Work for making modifications to it and all available documentation describing how to modify the Original Work. Licensor agrees to provide a machine-readable copy of the Source Code of the Original Work along with each copy of the Original Work that Licensor distributes. Licensor reserves the right to satisfy this obligation by placing a machine-readable copy of the Source Code in an information repository reasonably calculated to permit inexpensive and convenient access by You for as long as Licensor continues to distribute the Original Work.

4) Exclusions From License Grant. Neither the names of Licensor, nor the names of any contributors to the Original Work, nor any of their trademarks or service marks, may be used to endorse or promote products derived from this Original Work without express prior permission of the Licensor. Except as expressly stated herein, nothing in this License grants any license to Licensor's trademarks, copyrights, patents, trade secrets or any other intellectual property. No patent license is granted to make, use, sell, offer for sale, have made, or import embodiments of any patent claims other than the licensed claims defined in Section 2. No license is granted to the trademarks of Licensor even if such marks are included in the Original Work. Nothing in this License shall be interpreted to prohibit Licensor from licensing under terms different from this License any Original Work that Licensor otherwise would have a right to license.

5) External Deployment. The term "External Deployment" means the use, distribution, or communication of the Original Work or Derivative Works in any way such that the Original Work or Derivative Works may be used by anyone other than You, whether those works are distributed or communicated to those persons or made available as an application intended for use over a network. As an express condition for the grants of license hereunder, You must treat any External Deployment by You of the Original Work or a Derivative Work as a distribution under section 1(c).

6) Attribution Rights. You must retain, in the Source Code of any Derivative Works that You create, all copyright, patent, or trademark notices from the Source Code of the Original Work, as well as any notices of licensing and any descriptive text identified therein as an "Attribution Notice." You must cause the Source Code for any Derivative Works that You create to carry a prominent Attribution Notice reasonably calculated to inform recipients that You have modified the Original Work.

7) Warranty of Provenance and Disclaimer of Warranty. The Original Work is provided under this License on an "AS IS" BASIS and WITHOUT WARRANTY, either express or implied, including, without limitation, the warranties of non-infringement, merchantability or fitness for a particular purpose. THE ENTIRE RISK AS TO THE QUALITY OF THE ORIGINAL WORK IS WITH YOU. This DISCLAIMER OF WARRANTY constitutes an essential part of this License. No license to the Original Work is granted by this License except under this disclaimer.

8) Limitation of Liability. Under no circumstances and under no legal theory, whether in tort (including negligence), contract, or otherwise, shall the Licensor be liable to anyone for any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or the use of the Original Work including, without limitation, damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses. This limitation of liability shall not apply to the extent applicable law prohibits such limitation.

9) Acceptance and Termination. If, at any time, You expressly assented to this License, that assent indicates your clear and irrevocable acceptance of this License and all of its terms and

conditions. If You distribute or communicate copies of the Original Work or a Derivative Work, You must make a reasonable effort under the circumstances to obtain the express assent of recipients to the terms of this License. This License conditions your rights to undertake the activities listed in Section 1, including your right to create Derivative Works based upon the Original Work, and doing so without honoring these terms and conditions is prohibited by copyright law and international treaty. Nothing in this License is intended to affect copyright exceptions and limitations (including "fair use" or "fair dealing"). This License shall terminate immediately and You may no longer exercise any of the rights granted to You by this License upon your failure to honor the conditions in Section 1(c).

10) Termination for Patent Action. This License shall terminate automatically and You may no longer exercise any of the rights granted to You by this License as of the date You commence an action, including a cross-claim or counterclaim, against Licensor or any licensee alleging that the Original Work infringes a patent. This termination provision shall not apply for an action alleging patent infringement by combinations of the Original Work with other software or hardware.

11) Jurisdiction, Venue and Governing Law. Any action or suit relating to this License may be brought only in the courts of a jurisdiction wherein the Licensor resides or in which Licensor conducts its primary business, and under the laws of that jurisdiction excluding its conflict-of-law provisions. The application of the United Nations Convention on Contracts for the International Sale of Goods is expressly excluded. Any use of the Original Work outside the scope of this License or after its termination shall be subject to the requirements and penalties of copyright or patent law in the appropriate jurisdiction. This section shall survive the termination of this License.

12) Attorneys' Fees. In any action to enforce the terms of this License or seeking damages relating thereto, the prevailing party shall be entitled to recover its costs and expenses, including, without limitation, reasonable attorneys' fees and costs incurred in connection with such action, including any appeal of such action. This section shall survive the termination of this License.

13) Miscellaneous. If any provision of this License is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable.

14) Definition of "You" in This License. "You" throughout this License, whether in upper or lower case, means an individual or a legal entity exercising rights under, and complying with all of the terms of, this License. For legal entities, "You" includes any entity that controls, is controlled by, or is under common control with you. For purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

15) Right to Use. You may use the Original Work in all ways not otherwise restricted or conditioned by this License or by law, and Licensor promises not to interfere with or be responsible for such uses by You.

16) Modification of This License. This License is Copyright © 2005 Lawrence Rosen. Permission is granted to copy, distribute, or communicate this License without modification. Nothing in this License permits You to modify this License as applied to the Original Work or to Derivative Works. However, You may modify the text of this License and copy, distribute or communicate your modified version (the "Modified License") and apply it to other original works of authorship subject to the following conditions: (i) You may not indicate in any way that your Modified License is the "Open Software License" or "OSL" and you may not use those names in the name of your Modified License; (ii) You must replace the notice specified in the first paragraph above with the notice "Licensed under <insert your license name here>" or with a notice of your own that is not confusingly similar to the notice in this License; and (iii) You may not claim that your original works are open source software unless your Modified License has been approved by Open Source Initiative (OSI) and You comply with its license review and certification process.

17) Non-Profit Amendment. The name of this amended version of the Open Software License ("OSL 3.0") is "Non-Profit Open Software License 3.0". The original OSL 3.0 license has been amended as follows:

(a) Licensor represents and declares that it is a not-for-profit organization that derives no

revenue whatsoever from the distribution of the Original Work or Derivative Works thereof, or from support or services relating thereto.

(b) The first sentence of Section 7 ["Warranty of Provenance"] of OSL 3.0 has been stricken. For Original Works licensed under this Non-Profit OSL 3.0, LICENSOR OFFERS NO WARRANTIES WHATSOEVER.

(c) In the first sentence of Section 8 ["Limitation of Liability"] of this Non-Profit OSL 3.0, the list of damages for which LIABILITY IS LIMITED now includes "direct" damages.

(d) The proviso in Section 1(c) of this License now refers to this "Non-Profit Open Software License" rather than the "Open Software License". You may distribute or communicate the Original Work or Derivative Works thereof under this Non-Profit OSL 3.0 license only if You make the representation and declaration in paragraph (a) of this Section 17. Otherwise, You shall distribute or communicate the Original Work or Derivative Works thereof only under the OSL 3.0 license and You shall publish clear licensing notices so stating. Also by way of clarification, this License does not authorize You to distribute or communicate works under this Non-Profit OSL 3.0 if You received them under the original OSL 3.0 license.

(e) Original Works licensed under this license shall reference "Non-Profit OSL 3.0" in licensing notices to distinguish them from works licensed under the original OSL 3.0 license.

B flexible molecular geometry DTD

```
<!ELEMENT mode (#PCDATA)>
<!ELEMENT lattice (latvec_a, latvec_b, latvec_c)>
<!ATTLIST lattice
orgx CDATA "0.0"
orgy CDATA "0.0"
orgz CDATA "0.0"
lunit (ang|au) "ang"
>
<!ELEMENT latvec_a (#PCDATA)>
<!ELEMENT latvec_b (#PCDATA)>
<!ELEMENT latvec_c (#PCDATA)>
<!ELEMENT layer (li, lname)>
<!ELEMENT li (#PCDATA)>
<!ELEMENT lname (#PCDATA)>
<!ELEMENT atom (x, y, z, el, st?, chr?, li?, lpop?)>
<!ATTLIST atom
lunit (ang|au) "ang"
>
<!ELEMENT x (#PCDATA)>
<!ELEMENT y (#PCDATA)>
<!ELEMENT z (#PCDATA)>
<!ELEMENT el (#PCDATA)>
<!ELEMENT st (#PCDATA)>
<!ELEMENT chr (#PCDATA)>
<!ELEMENT nrg (#PCDATA)>
<!ELEMENT lpop (#PCDATA)>
<!ATTLIST nrg
eunit (eV|au) "au"
>
<!ELEMENT velocities (#PCDATA)>
<!ELEMENT forces (#PCDATA)>
<!ELEMENT geometry (mode?, lattice?, layer?, atom+)>
```

```
<!ELEMENT trjstep (nrg?,velocities?,forces?)>
<!ELEMENT fmg (geometry+,trjstep*,trjinfo?)>
<!ELEMENT stepcount (#PCDATA)>
<!ELEMENT trjinfo (stepcount?)>
<!ELEMENT DeltaR (#PCDATA)>
<!ATTLIST DeltaR
lunit (ang|au) "ang"
>
<!ELEMENT NoGradInRot EMPTY>
<!ELEMENT curvature (#PCDATA)>
<!ELEMENT E0 (#PCDATA)>
<!ATTLIST E0
eunit (eV|au|H) "au"
>
<!ELEMENT E1 (#PCDATA)>
<!ATTLIST E1
eunit (eV|au|H) "au"
>
<!ELEMENT E2 (#PCDATA)>
<!ATTLIST E2
eunit (eV|au|H) "au"
>
<!ELEMENT f0 (#PCDATA)>
<!ELEMENT f1 (#PCDATA)>
<!ELEMENT f2 (#PCDATA)>
<!ELEMENT fN (#PCDATA)>
<!ELEMENT Dimer (Geometry,DeltaR,NoGradInRot?,(E0,E1)?,E2?,(f0,f1)?,f2?,fN?,curvature?)>
```