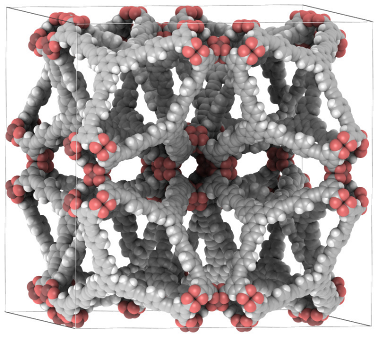
RASPA 2.0.45: Molecular Software Package for Adsorption and Diffusion in (Flexible) Nanoporous Materials



David Dubbeldam[1]
Van 't Hoff Institute of Molecular Sciences, University of Amsterdam,
Science Park 904, 1098XH, Amsterdam, The Netherlands

Sofia Calero[2]
Department of Applied Physics, Eindhoven University of Technology,
5600MB Eindhoven, The Netherlands

Thijs J.H. Vlugt[3]
Engineering Thermodynamics, Process & Energy Department,
Faculty of Mechanical, Maritime and Materials Engineering,
Delft University of Technology, Leeghwaterstraat 39, 2628CB Delft, The Netherlands

Donald E. Ellis[4]
Department of Physics and Astronomy, Northwestern University,
2145 Sheridan Road, Evanston IL 60208 USA

Randall Q. Snurr[5],
Chemical and Biological Engineering Department, Northwestern University,
2145 Sheridan Road, Evanston IL 60208, USA

May 20, 2021

[1]email: d.dubbeldam@uva.nl
[2]email: S.Calero@tue.nl
[3]email: T.J.H.Vlugt@tudelft.nl
[4]email: don-ellis@northwestern.edu
[5]email: snurr@northwestern.edu

Contents

Part I

RASPA

1 Introduction

1.1 Design philosophy

This software is a general purpose classical simulation package. It has been developed at Northwestern University (Evanston, USA; group of Prof. Randall Q. Snurr) during 2006-2009 in active collaboration with University Pablo de Olavide (Seville, Spain; group of Prof. Sofia Calero), and from 2010-2015 also at the University of Amsterdam (David Dubbeldam) and Technical University of Delft (group of Prof. T.J.H. Vlugt). It can be used for the simulation of molecules in gases, fluids, zeolites, aluminosilicates, metal-organic frameworks, and carbon nanotubes.

Programs can be written in various ways, but often it is true that the fastest codes are probably the hardest to read, while programs strictly based on readability lacks efficiency. RASPA is based on the following ideas:

- Correctness and accuracy
  For all the techniques and algorithms available in RASPA we have implemented the 'best' ones available in literature. For example, RASPA uses Configurational-Bias Monte-Carlo, it uses the Ewald summation for electrostatics, molecular dynamics is based on 'symplectic' integrators, all Monte-Carlo moves obey detailed balance etc.
- Functional design
  Looking at the source, you will notice that there are not a lot of files. The program is split up according to its function: 'grid.c' contains the code to make and use a grid of a framework, 'ewald.c' handles all the electrostatic,'mc_moves.c' contains all the moves to be used in Monte-Carlo,'potentials.c' contains all the VDW potentials etc.
- Input made easy
  The requirements for the input files is kept as minimal as possible. Only for more advanced options extra commands in the input file are needed. Also the format of the input is straightforward. Default settings are usually the best ones. Fugacity coefficients and excess adsorption are automatically computed.
- Integrated simulation environment
  The code is built up of many functions and routines which can be easily combined to do what you want. Molecular dynamics can be used in Monte Carlo and visa versa. Extension and modification of the code is relatively straightforward.

RASPA used three 'types' or 'groups' for the particles: 1) Framework atoms, 2) Adsorbates, and 3) Cations. The advantage is that all the energies are split and the interactions can be examined (also the energies are split in the Ewald Fourier part). Another example is when using thermostats in e.g. LTA5A where a different thermostat operates in the framework atoms, the adsorbates, and the cations. These all move at different length- and time scales. Note that it is not possible to exchange types during Identity-change moves (if defined they are ignored).

1.2 Units and conventions

- The standard units in RASPA from which all other units are derived are:

quantity	symbol	unit	value
length	l	Ångstrom	10^{-10} m
temperature	T	Kelvin	K
mass	m	atomic mass	$1.6605402 \times 10^{-27}$ kg
time	t	pico seconds	10^{-12} s
charge	q	atomic charge	$1.60217733 \times 10^{-19}$ C/particle

Some examples of derived units:

quantity	symbol	units	conversion value
energy	U	$J =$ mass $\times$ length^2/time^2	1.66054×10^{-23} (=10 J/mol)
pressure	p	Pa = mass/(length $\times$ time^2)	1.66054×10^{7}
diffusion constant	D	$D =$ length^2/time	1×10^{-8}
force	f	$f =$ length/time^2	1.66054×10^{-13}
…	…	…	…

A pressure input of 10 Pascal in the input file, is converted to 'internal units' by dividing by 1.66054×10^7. In the output any internal pressure is printed, multiplied by 1.66054×10^7. It is not necessary to convert units besides input and output, with a few exceptions. One of them is the Coulombic conversion factor

$$\frac{q_i q_j}{4\pi\epsilon_0} = \frac{\text{charge}^2}{4\pi \times \text{electric constant} \times \text{length} \times \text{energy}} = 138935.4834964017 \tag{1.1}$$

with the electric constant as $8.8541878176 \times 10^{-12}$ in units of C^2/(N.m^2). This factor is needed to convert the electrostatic energy to the internal units at every evaluation.

The Boltzmann's constant k_B is

$$k_B = \text{Boltzmann constant/energy} = 0.8314464919 \tag{1.2}$$

with the Boltzmann constant as $1.380650324 \times 10^{-23}$ in units of J/K, and $k_B = 0.8314464919$ in internal units.

- Numbering is based on the C-convention, i.e. starting from zero.

- Files in the current directory always have preference.
  Sometimes one would like to try various parameters for force field fitting for example. In order to avoid making a lot of directories for each force field it is more convenient to have the 'pseudo_atoms.def', 'force_field_mixing_rule.def' and 'force_field.def' files in the *current* directory.

1.3 Compiling and installing RASPA

1.3.1 Requirements

RASPA needs a C compiler, like 'gcc' or intel's 'icc' compilers, and optionally the libraries 'fftw', 'blas', and 'lapack'.

1.3.2 RASPA from 'git'

Working with 'git' and a remote repository means that you will have to distinguish between two locations of the code:

1. The repository (visible to everyone)
2. your local copy (only visible to you)

To check-out the code for the first time do:

```
git clone https://github.com/iraspa/RASPA2
```

After that, you can update the code by using

```
git pull
```

1.3.3 installing RASPA

The *RASPA_DIR* environment variable should be set to where you would like to install RASPA. A common way of defining it is using the bash-shell

export RASPA_DIR=${HOME}/RASPA/simulations/

or

setenv RASPA_DIR "${HOME}/RASPA/simulations/"

for 'csh' and 'tcsh' shells. It is possible to add this line to ".bashrc", "/etc/bashrc", "/etc/profile" etc, depending on the unix-version and shell version to automatically have the environment variable set at login.

Note that the source-code of RASPA is kept separate from the installation data. RASPA needs the environment variable to locate various files it needs, e.g. molecule definitions, framework definitions, force and field definitions. It looks for these files relative to the RASPA_DIR directory.

Before installing RASPA with

```
make install
```

from the top-directory, the code needs to be compiled.

1.3.4 compiling RASPA

RASPA uses the standard 'configure' utilities (autoconf, automake, libtool, and make). The steps to install from scratch, i.e. after a 'make distclean' or 'git clone' are

1. `rm -rf autom4te.cache`
2. `mkdir m4`
3. `aclocal`

4. `autoreconf -i`
5. `automake --add-missing`
6. `autoconf`
7. `./configure --prefix=${RASPA_DIR}`   or
   `./scripts/CompileScript/make-gcc-local`
8. `make`

where '${RASPA_DIR}' is the directory you would like to install RASPA, and the commands are executed in the top directory.

Usually (when recent automake and autoconf versions are installed), it is enough to do

1. `make clean`
2. `./configure --prefix=${RASPA_DIR}`
3. `make`

You can use the 'CFLAGS' environment variable to set compiler options and 'CC' to set the compiler. For example, for a gcc compiler one could use

```
export CFLAGS="-Wall -O3 -ffast-math"
export CC="gcc"
```

1.3.5 Running RASPA

Running RASPA is based on two files:

- A 'run' file to execute the program
  an example file is:

  ```
  #! /bin/sh -f
  export RASPA_DIR=${HOME}/Research/simulations/
  $RASPA_DIR/bin/simulate
  ```

  This type of file is know as a 'shell script'. RASPA needs the variable 'RASPA_DIR' to be set in order to look up the molecules, frameworks, etc. The scripts sets the variable and runs RASPA. RASPA can then be run from any directory you would like.

- An 'input'-file describing the type of simulation and the parameters
  In the same directory as the 'run'-file, there needs to be a file called 'simulation.input'. An example file is:

  ```
  SimulationType                    MonteCarlo
  NumberOfCycles                    100000
  NumberOfInitializationCycles      10000
  PrintEvery                        1000

  Box 0
  BoxLengths 30 30 30
  ExternalTemperature 300.0
  ```

```
    component 0 methane
                TranslationProbability           1.0
                CreateNumberOfMolecules          100
```

This tells RASPA to run a Monte-Carlo simulation of 100 methane molecules in a $30 \times 30 \times 30$ Å cubic box (with 90° angles) at 300 Kelvin. It will start with 10000 cycles to equilibrate the system and will use 100000 cycle to obtain thermodynamic properties of interest. Every 1000 cycles a status-report is printed to the output. The Monte-Carlo program will use only the 'translation move' where a particle is given a random translation and the move is accepted or rejected based on the energy difference.

In order to run it on a cluster using a queuing system one needs an additional file 'bsub.job' (arbitrary name)

- 'gridengine'

```
#!/bin/csh
# Serial sample script for Grid Engine
# Replace items enclosed by {}
#$ -S /bin/csh
#$ -N Test
#$ -V
#$ -cwd
echo $PBS_JOBID > jobid
setenv RASPA_DIR ${HOME}/RASPA/simulations/
$RASPA_DIR/bin/simulate
```

  The job can be submitted using 'qsub bsub.job'.

- 'torque'

```
#!/bin/bash
#PBS -N Test
#PBS -o pbs.out
#PBS -e pbs.err
#PBS -r n
#PBS -V
#PBS -mba
cd $PBS_O_WORKDIR
echo $PBS_JOBID > jobid
export RASPA_DIR=${HOME}/RASPA/simulations
${RASPA_DIR}/bin/simulate
```

  The job can be submitted using 'qsub bsub.job'.

- 'slurm'

```
#!/bin/bash
#SBATCH -N 1
#SBATCH --job-name=Test
#SBATCH --export=ALL
echo $SLURM_JOBID > jobid
```

```
valhost=$SLURM_JOB_NODELIST
echo $valhost > hostname
module load slurm
${RASPA_DIR}/bin/simulate
```

The job can be submitted using 'sbatch bsub.job'.

1.4 Output from RASPA

RASPA generates output from the simulation. Some data is just information on the status, while other data are written because you specifically asked the program to compute it for you. The output is written to be used with other programs like:

- gnuplot
- VTK
- iRASPA
- VMD

The main output is written to the directory 'Output/System_0/', 'Output/System_1/', ... for each of the simulated systems. Usually one simulates only a single system. However, the Gibbs ensemble requires 2 systems, one for vapor phase and one for the liquid phase, while n systems are used by the (hyper-) parallel-tempering technique(s).

1.5 Citing RASPA

If you are using RASPA and would like to cite it in your journal articles or book-chapters, then for RASPA:

D. Dubbeldam, S. Calero, D.E. Ellis, and R.Q. Snurr, RASPA: Molecular Simulation Software for Adsorption and Diffusion in Flexible Nanoporous Materials, *Mol. Simulat.*, http://dx.doi.org/10.1080/08927022.2015.1010082, 2015.

For the inner workings of Monte Carlo codes:

D. Dubbeldam, A. Torres-Knoop, and K.S. Walton, On the Inner Workings of Monte Carlo Codes, http://dx.doi.org/10.1080/08927022.2013.819102 *Mol. Simulat.*, 39(14-15), 1253-1292, 2013.

For the description of Molecular Dynamics and diffusion:

D. Dubbeldam and R.Q. Snurr, Recent Developments in the Molecular Modeling of Diffusion in Nanoporous Materials, http://dx.doi.org/10.1080/08927020601156418, *Mol. Simulat.*, 33(4-5), 305-325, 2007.

For the description of the implementation of force fields:

D. Dubbeldam, K.S. Walton, T.J.H. Vlugt, and S. Calero, Design, Parameterization, and Implementation of Atomic Force Fields for Adsorption in Nanoporous Materials, https://doi.org/10.1002/adts.201900135, *Adv. Theory Simulat.*, 2(11), 1900135, 2019.

2 Format of the Input Files

2.1 Introduction

In order to run a simulation you need several input-files:

- `‘simulation.input’`
  This file contains the information on the type of simulation, the amount of steps, the framework name, number of unit cells in each directions, the used molecules, the type of used Monte-Carlo moves etc.

- `‘structure-name.cif’`
  If a framework (e.g. a zeolite or MOF) is used, then the definition of the structure needs to be provided. CIF-files are supported and the default input. The name of the file should be equal to the one provided in `‘simulation.input’`, e.g. IRMOF-1.cif if `‘Frameworkname IRMOF-1’` is listed in `‘simulation.input’`.

- `‘pseudo_atoms.def’`
  The `‘pseudo_atoms.def’` file list all the information on used pseudo-atoms, e.g. charge, mass. Usually a pseudo-atom is an atom, but there are exceptions like united atoms (where CH3 is lumped into one unit) and off-atom sites in Tip5p water that represent oxygen lone pairs. Because in CIF-files for frameworks you can provide also information on atoms, there is no need to list framework atoms here if a CIF-file is used. On reading the CIF-file these defined atoms are added to the pseudo-atoms. If also provided in the `‘pseudo_atoms.def’` then the definition in the `‘pseudo_atoms.def’` file has priority.

- `‘force_field_mxing_rules.def’`,`‘force_field.def’`
  The force field defined on the pseudo-atoms in `‘pseudo_atoms.def’`. These files list the Van der Waals potential types, the parameters, whether to use tail-corrections, whether to shift to zero at the cutoff, and the type of mixing rule. Force fields in literature are usually published in two forms: 1) a list of potentials parameters per atom and a mixing rule, or 2) pairs of atoms and parameters. The first option corresponds to the file `‘force_field_mxing_rules.def’` and the latter option to the file `‘force_field.def’`. You can use both at the same time, where `‘force_field.def’` has precedence over `‘force_field_mxing_rules.def’`.

- `‘molecule-name.def’`
  The definition of the used molecules. The name of the file should be equal to the one provided in `‘simulation.input’`, e.g. propane.def if `‘MoleculeName propane’` is listed in `‘simulation.input’`.

- `zframework.def'`
  Used for a flexible framework to define all the bonds, bends, torsions, core-shells, etc.

The format of these files will be described in the remaining sections. Chapter 4 provides lots of examples to see everything in action. In addition to the input-files you will need either a `'run'` file that is executable, or a queuing-script to submit the job to the queue (see 1.3.5).

2.2 Simulation input

Leading spaces and comments at the end of each line are omitted. Empty lines are skipped, and case is not important except in file names (i.e. framework and molecule names).

Simulation types

- SimulationType MonteCarlo
  Starts the Monte Carlo part of RASPA. The particular ensemble is not specified but implicitly deduced from the specified Monte Carlo moves. Note that a MD-move can be used for hybrid MC/MD.
- SimulationType MolecularDynamics
  Starts the Molecular Dynamics part of RASPA. The ensemble is explicitly specified.
- SimulationType Spectra
  Starts the computation of the vibrational analysis. Possible options include infra red spectrum at zero Kelvin, powder diffraction, and mode analysis.
- SimulationType Minimization
  Starts the minimization routine. It produces configurations and crystal structures at zero Kelvin.
- SimulationType Visualization
  Output VTK-files for snapshots and crystal structures, including energy surface pictures.
- SimulationType BarrierCrossing
  Routine for the dynamical correction of dynamically corrected Transition State Theory.
- SimulationType Numerical
  Computes all the forces numerically from the energy and compares them to the analytical expressions. Also the strain-derivative tensor (related to the stress tensor), and the second derivative of the energy with respect to strain, as well as the Hessian matrix can be checked.
- SimulationType MakeGrid
  Creates pre-tabulated energy-grids for use in rigid frameworks.

Simulation duration

- NumberOfCycles [int]
  The number of cycles for the production run. For Monte Carlo a cycle consists of N steps, where N is the amount of molecules with a minimum of 20 steps. This means that on average during each cycle on each molecule a Monte Carlo move has been attempted (either successful or unsuccessful). For MD the number of cycles is simply the amount of integration steps.
- NumberOfInitializationCycles [int]
  The number of cycles used to initialize the system using Monte Carlo. This can be used for both Monte Carlo as well as Molecular Dynamics to quickly equilibrate the positions of the atoms in the system.

- NumberOfEquilibrationCycles [int]
  For Molecular Dynamics it is the number of MD steps to equilibrate the velocities in the systems. After this equilibration the production run is started. For Monte Carlo, in particular CFMC, the equilibration-phase is used to measure the biasing factors.

Restart and crash-recovery

- RestartFile [yes|no]
  Reads the positions, velocities, and force from the directory 'RestartInitial'. Any creation of molecules in the 'simulation.input' file will be in addition and after this first read from file. This is useful to load initial positions of cations for example, and after that create adsorbates. The restart file is written at 'PrintEvery' intervals.

- ContinueAfterCrash [yes|no]
  Write a binary file containing the complete status of the program. The file name is 'binary_restart.dat' and is located in the directory 'CrashRestart'. With this option to 'yes' the presence of this file will result in continuation from the point where the program was at the moment of outputting this file. The file can be quite big (several hundreds of megabytes) and will be outputted every 'WriteBinaryRestartFileEvery' cycles.

- WriteBinaryRestartFileEvery [int]
  The output frequency (i.e. every [int] cycles) of the crash-recovery file.

Printing options

- PrintEvery [int]
  Prints the loadings (when a framework is present) and energies every [int] cycles. For MD information like energy conservation and stress are printed.

- PrintPropertiesEvery [int]
  Output running averages of many properties (i.e. Henry coefficients and elastic constants).

- PrintForcefieldToOutput [yes|no]
  Prints the force field information to the output-file. Default: yes.

- PrintPseudoAtomsToOutput [yes|no]
  Prints the pseudo-atom information to the output-file. Default: yes.

- PrintMoleculeDefinitionToOutput [yes|no]
  Prints the molecule definition information to the output-file. Default: yes.

Force field definitions

- ChargeFromChargeEquilibration [yes|no]
  Compute the charges of the framework using the 'charge-equilibration'-method.

- SymmetrizeFrameworkCharges [yes|no]
  All charges of the framework are made equivalent for equivalent framework atoms. Using regular charge-equilibration the charges are different for symmtrically equivalent framework atoms, and this options restores the symmetry.

- ForceField [string]
  Reads in the force field [string], first the file 'pseudo_atoms.def' is read, then 'force_field_mixing_rules.def' and finally 'force_field.def'. The latter overwrites general settings for interactions based on mixing rules with specific ones for individual interactions.

Note that if any of these files are in the working directory then these will read and used instead of the ones in '${RASPA_DIR}/simulations/share/raspa/forcefield/[string]'.

- CutOffVDW [real]
The cutoff of the Van der Waals potentials. Interactions longer then this distance are omitted from the energy and force computations. The potential can either be shifted to zero at the cutoff, or interactions can just neglected after the cut off, or the remainder of the potential energy can be approximated using tail corrections. This is specified in the force field files and can be specified globally or for each interaction individually.

- CutOffVDWSwitch [real]
The distance at which VDW switching will start. The smoothing will make sure the value and derivatives are zero at the cutoff. The default: 0.9 times the CutOff.

- CutOffChargeCharge [real]
The cutoff of the charge-charge potential. The potential is truncated at the cutoff and only shifted when 'ChargeMethod CoulombShifted' or 'ChargeMethod CoulombSmoothed' is used. No tail-corrections are (or can be) applied. The only way to include the long-range part is to use 'ChargeMethod Ewald'. The parameter is also used in combination with the Ewald precision to compute the number of wave vectors and Ewald parameter α. For the Ewald summation using rather large unit cells, a charge-charge cutoff of about half the smallest box-length would be advisable in order to avoid the use of an excessive amount of wave-vectors in Fourier space. For non-Ewald methods the cutoff should be as large as possible (greater than about 30 Å).

- CutOffChargeChargeSwitch [real]
The distance at which charge-charge switching will start. The smoothing will make sure the value and derivatives are zero at the cutoff. The default: 0.65 times the CutOff.

- CutOffChargeBondDipole [real]
The cutoff of the charge-bonddipole potential.

- CutOffChargeBondDipoleSwitch [real]
The distance at which charge-bonddipole switching will start. The smoothing will make sure the value and derivatives are zero at the cutoff. The default: 0.70 times the CutOff.

- CutOffBondDipoleBondDipole [real]
The cutoff of the bonddipole-bonddipole potential.

- CutOffBondDipoleBondDipoleSwitch [real]
The distance at which bonddipole-bonddipole switching will start. The smoothing will make sure the value and derivatives are zero at the cutoff. The default: 0.75 times the CutOff.

- OmitAdsorbateAdsorbateVDWInteractions [yes|no]
Omits the Van der Waals interactions between adsorbates.

- OmitAdsorbateAdsorbateCoulombInteractions [yes|no]
Omits the Coulombic (i.e. Ewald) interactions between adsorbates.

- OmitInterMolecularInteractions [yes|no]
Omits the interactions between all molecules (only interactions with the framework). This also works with the Ewald summation on. The options implies the setting of both
  - OmitAdsorbateAdsorbateVDWInteractions [yes|no]
  - OmitAdsorbateAdsorbateCoulombInteractions [yes|no]

- InternalFrameworkLennardJonesInteractions [yes|no]
  Compute the Van der Waals interaction of the flexible framework. The Demontis flexible model for silicalite is defined with only bond, bend, and torsion for example. One can use this option and also use 'Charge None'.

- RemoveBondNeighboursFromLongRangeInteraction [yes|no]
  RemoveBendNeighboursFromLongRangeInteraction [yes|no]
  RemoveTorsionNeighboursFromLongRangeInteraction [yes|no]
  After construction of the connectivity table all interactions are removed from Van der Waals and charge interactions that are defined as 1-2 (i.e. bonds), 1-3 (i.e. bends, Urey-Bradley) and 1-4 (i.e. torsion, inversion-bend) respectively.

- Remove12NeighboursFromChargeChargeInteraction [yes|no]
  Remove13NeighboursFromChargeChargeInteraction [yes|no]
  Remove14NeighboursFromChargeChargeInteraction [yes|no]
  Remove all 1-2, 1-3, and/or 1-4 interactions within the framework from the long-range charge-charge interaction within the flexible framework respectively.

- Remove12NeighboursFromChargeBondDipoleInteraction [yes|no]
  Remove13NeighboursFromChargeBondDipoleInteraction [yes|no]
  Remove14NeighboursFromChargeBondDipoleInteraction [yes|no]
  Remove all 1-2, 1-3, and/or 1-4 interactions within the framework from the long-range charge-bond dipole interaction within the flexible framework respectively.

- Remove12NeighboursFromBondDipoleBondDipoleInteraction [yes|no]
  Remove13NeighboursFromBondDipoleBondDipoleInteraction [yes|no]
  Remove14NeighboursFromBondDipoleBondDipoleInteraction [yes|no]
  Remove all 1-2, 1-3, and/or 1-4 interactions within the framework from the long-range bond dipole-bond dipole interaction within the flexible framework respectively.

Thermostat and barostat parameters

- ExternalTemperature [list-of-reals]
  The external temperature in Kelvin for each system. Because the system is in contact with this imaginary reservoir the average temperature of the system can be controlled. Default: 298K.

- ExternalPressure [list-of-reals]
  The external pressure in Pascal for each system. Because the system is in contact with this imaginary reservoir the average pressure of the system can be controlled.

- ThermostatChainLength [int]
  The length of the chain to thermostat the system. Default: 5.

- BarostatChainLength [int]
  The length of the chain to thermostat the volume and/or cell parameters. Default 5.

- NumberOfYoshidaSuzukiSteps [int]
  The number of Yoshida/Suzuki multiple timesteps.

- TimeScaleParameterThermostat [real]
  The time scale on which the system thermostat evolves. Default: 0.15 ps.

- TimeScaleParameterBarostat [real]
  The time scale on which the thermostat for the volume and/or cell parameters evolve. Default: 0.15 ps.

Molecular dynamics parameters

- TimeStep [real]
  The time step in picoseconds for MD integration. Default value: 0.0005 ps (0.5 fs).

- Ensemble [list-of-NVE|NVT|NPT|NPH|NPTPR|NPHPR]
  Sets the ensemble as a list of NVE,NVT, NPT, NPH, NPTPR, or NPHPR for each system. If only a single ensemble is given, it is used for all systems. The given ensemble will be used for both initialization as well as the production run.

  - NVE
    The micro canonical ensemble, the number of particle N, the volume V, and the energy E are constant.
  - NVT
    The canonical ensemble, the number of particle N, the volume V, and the average temperature $\langle P\rangle$ are constant. Instantaneous values for the temperature are fluctuating.
  - NPT
    The isobaric-isothermal ensemble, the number of particle N, the average pressure $\langle P\rangle$, and the average temperature $\langle P\rangle$ are constant. Instantaneous values for the pressure and temperature are fluctuating.
  - NPH
    The isoenthalpic-isobaric ensemble, the number of particle N, the average pressure $\langle P\rangle$, and the enthalpy H are constant. Instantaneous values for the pressure and temperature are fluctuating.
  - NPTPR
    The isobaric-isothermal ensemble with a fully flexible cell (Parrinello-Rahman).

- InitEnsemble [list-of-NVE|NVT|NPT|NPH|NPTPR|NPHPR]
  Sets the ensemble as a list of NVE,NVT, NPH, NPTPR, or NPHPR for each system. If only a single ensemble is given, it is used for all systems. The given ensemble will be only used for the initialization run.

- RunEnsemble [list-of-NVE|NVT|NPT|NPH|NPTPR|NPHPR]
  Set the ensemble as a list of NVE,NVT, NPH, NPTPR, or NPHPR for each system. If only a single ensemble is given, it is used for all systems. The given ensemble will be only used for the production run.

- NPTPRCellType [list-of-Regular|Monoclinic|RegularUpperTriangle|MonoclinicUpperTriangle|Isotropic|Anisotropic]
  The type of constraints on the cell-matrix **h**. Default: RegularUpperTriangle.

  - Regular
    If the pressure tensor is asymmetric ($P_{\alpha\beta} \neq P_{\beta\alpha}$) at a given instant of time, then there will be a net torque acting on the cell that will cause it to rotate. Cell rotations can be eliminated by using the symmetrized tensor $P_{\alpha\beta} = (P_{\alpha\beta} + P_{\beta\alpha})/2$ in the equations of motion and setting the initial total angular momentum of the cell to zero. This approach is formally implemented by constraining the force on the cell $\mathbf{g} = \mathbf{g}^T$. All three angles α, β, γ are allowed to change, as well as the box lengths $\mathbf{a}, \mathbf{b}, \mathbf{c}$.
  - Monoclinic
    All three box lengths $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are allowed to vary, as well as cell angle β, but $\alpha = \gamma = 90°$.
  - RegularUpperTriangle
    Only the upper triangular part of the cell matrix is used to eliminate rotation of the box. All three angles α, β, γ are allowed to change, as well as the box lengths $\mathbf{a}, \mathbf{b}, \mathbf{c}$.

- MonoclinicUpperTriangle
  Only the upper triangular part of the cell matrix is used to eliminate rotation of the box. All three box lengths $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are allowed to vary, as well as cell angle β, but $\alpha = \gamma = 90°$.
- Isotropic
  All three box lengths $\mathbf{a} = \mathbf{b} = \mathbf{c}$ are allowed to vary isotropically, and the angles remain fixed $\alpha = \beta = \gamma = 90°$.
- Anisotropic
  All three box lengths $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are allowed to vary *independently*, but the angles remain fixed $\alpha = \beta = \gamma = 90°$.

Box parameters

- Box [int]
  [real] [real] [real]
  Set the system [int] to type 'Box' (other option is 'Framework' when a framework is present). The cell dimensions of rectangular box of system [int] in Angstroms. Default: 25 25 25 Å.

- BoxAngles [int]
  [real] [real] [real]
  Set the system [int] to type 'Box' (other option is 'Framework' when a framework is present). The cell angles of rectangular box of system [int] in Angstroms. Default: 90° 90° 90°.

- BoxMatrix [int]
  [real] [real] [real]
  [real] [real] [real]
  [real] [real] [real]

  Set the system [int] to type 'Box' (other option is 'Framework' when a framework is present). The 3×3 cell matrix of system [int], given as three vectors (as columns). This is the most general form and any box can be specified in this way. Units of the vectors are Angstrom.

Framework parameters

- Framework [int]
  Set the system [int] to type 'Framework' (other option is 'Box' when no framework is present). All other options listed in the section framework parameters refer to this system, so make sure this is before any other framework options.

- FrameworkName [string]
  Loads the framework with name [string]. Several frameworks can be read per system, which is useful for to study interpenetration of frameworks. Here the frameworks are allowed to move independently from each other.

- HeliumVoidFraction [real]
  The void fraction as measure by probing the structure with helium a room temperature. This quantity has to be obtained from a separate simulation and is essential to compute the *excess*-adsorption during the simulation.

- UnitCells [int] [int] [int]
  The number of unit cells in x,y, and z direction for the system. The full cell will contain the unit cells, and periodic boundary conditions will be applied on the box level (*not* on a unit cell level).

- ShiftUnitCells [real] [real] [real]
  Shift the fractional positions so that the center of a framework can be altered.

- FlexibleFramework [yes|no]
  Allow the current framework of the current system to be fully flexible. The name of the flexible model is provided using the 'FrameworkDefinitions [string]' input option.

- FrameworkDefinitions [string]
  The force field name [string] of the flexible framework. The file is read even when 'FlexibleFramework no' is specified (the reason is that framework bond-dipoles are defined using the 'framework.def' file).

- ModifyFrameworkAtomConnectedTo [atom-type-1] [atom-type-2] [atom-type-3] [atom-type-4]
  Modifies the atom-type-1 to atom-type-2, always if atom-type-3 and atom-type-4 are omitted, or only it is connected to atom-type-3 when atom-type-3 is specified, or only when it is connected to both atom-type-3 and atom-type-4 if both are specified.

- ModifyFrameworkDimer [atom-type-1] [atom-type-2] [atom-type-3] [atom-type-4]
  Modifies the connected atom-type-1 and atom-type-2 dimer to atom-type-3 and atom-type-4.

- ModifyFrameworkTriple [atom-type-1] [atom-type-2] [atom-type-3] [atom-type-4] [atom-type-5] [atom-type-6]
  Modifies the connected triple atom-type-1,atom-type-2,atom-type-3 to atom-type-4,atom-type-5,atom-type-6.

- RemoveAtomNumberCodeFromLabel [yes|no]
  Reading structure-files: the number is removed from the framework atom-types, e.g. `'O1'`, `'O2'`, `'O3'`, etc. are mapped to `'O'`.

- AddAtomNumberCodeToLabel [yes|no]
  Writing structure-files: the number is added to the framework atom-types, e.g. `'O'` are mapped to `'O1'`, `'O2'`, `'O3'`, etc.

- RestrictFrameworkAtomsToBox [yes|no]
  Restricts (places back) atoms to the unit cell dimensions, i.e. fractional positions between 0 and 1.

- ReadCIFAsCartesian [yes|no]
  Reads the position listed in the CIF-file as Cartesian. Only applicable to P1 systems (no symmetry).

System moves

- FrameworkChangeMoveProbability [real]
  The probability per cycle to randomly translate a framework atom. During this move the number of inner cycles is the amount of framework atoms, with a maximum of 500. This move is applicable to relatively rigid structures like zeolites. For other structure where movement is caused by collective behavior (for example, the rotation of a phenyl-ring in a metal-organic framework) the MC/MD move is more convenient. Such movement is hardly sampled at all by individual MC translation moves.

- VolumeChangeProbability [real]
  The probability per cycle to attempt a volume-change. Rigid molecules are scaled by center-of-mass, while flexible molecules and the framework is atomically scaled.

- VolumeChangeDirection [A|B|C|AB|AC|BC|ABC]
  Change the volume of the unit cell along a particular direction/directions in Monte Carlo. Default: ABC.

- BoxShapeChangeProbability [real]
  The probability per cycle to attempt a shape-change of the box. One of the 6 upper triangular elements of the box matrix is randomly chosen. Rigid molecules are scaled by center-of-mass, while flexible molecules and the framework is atomically scaled.

- GibbVolumeChangeProbability [real]
  The probability per cycle to attempt a Gibbs volume-change MC move during a Gibbs ensemble simulation. The total volume of the two boxes (usually one for the gas phase, one for the liquid phase) remains constant, but the individual volume of the boxes are changed. The volumes are changed by a random change in $\ln(V_I/V_{II})$.

- HybridNVEMoveProbability [real]
  The probability per cycle to attempt a hybrid Monte Carlo move using Molecular Dynamics in the NVE-ensemble. The whole system is integrated using Newton's equations of motion. The new configuration is then accepted or rejected using the standard MC rule. Note that the difference in energy ΔU is the integration error. The integration time step is set using 'TimeStep'.

- NumberOfHybridNVESteps [int]
  The number of integration steps for the hybrid MC/MD NVE move. Default: 5.

- ParallelTemperingProbability [real]
  A move where two neighboring systems are swapped that differ in their temperature.

- HyperParallelTemperingProbability [real]
  A move where two neighboring systems are swapped that differ in their temperature and chemical potentials.

- ParallelMolFractionProbability [real]
  A move where two neighboring systems (similar to parallel tempering) are swapped that differ in their mol-fraction of components A and B.

- ParallelMolFractionComponentA [int]
  The identifier of the first component.

- ParallelMolFractionComponentB [int]
  The identifier of the second component.

- ChiralInversionProbability [real]
  A move specifically designed for systems with chiral molecules to change all S-molecules into R-molecules and vica versa. Note that the spacegroup needs to be set. If you have a framework that is P1 but has higher symmetry then use '`CalculateSpaceGroup yes`' to determine the true space group of the framework. An error will be given if this move is impossible for your system (e.g. when the framework is chiral).

Component information

- Component [int] MoleculeName [string]
  Reads in the definition of component [int] using the file '*molecule-name-string*.def' from the directory '${RASPA_DIR}/share/raspa/molecules/*molecule-definitions-string*'.

- MoleculeDefinitions [string]
  The type of the molecule. For example, there could an OPLS version of the molecule, or a TraPPE version, etc. This *molecule-definitions-string* is actually the directory name under which the molecule file is found in '${RASPA_DIR}/share/raspa/molecules/'.

- StartingBead [int]
  The staring bead for the configurational bias Monte Carlo (CBMC). In CBMC the molecule is grown bead by bead biasing the growth towards energetically favorable configurations. Certain operations, like the rotation MC move and Widom particle insertion, use this bead as the center of rotation and position of the probe molecule, respectively.

- BlockPockets [yes|no]
  Block certain pockets in the simulation volume. The growth of a molecule is not allowed in a blocked pocket. A typical example is the sodalite cages in FAU and LTA-type zeolites, these are not accessible to molecules like methane and bigger.

- BlockPocketsFileName [string]
  The file name for the definitions of all the blocking spheres.

- MolFraction [real]
  The mol fraction of this component in the mixture. The values can be specified relative to other components, as the fractions are normalized afterwards. The partial pressures for each component are computed from the total pressure and the mol fraction per component.

- FugacityCoefficient [real]
  The fugacity coefficient for the current component. For values 0 (or by not specifying this line), the fugacity coefficients are automatically computed using the Peng-Robinson equation of state. Note the critical pressure, critical temperature, and acentric factor need to be specified in the molecule file.

- Intra14VDWScalingValue [real]
  The scaling factor for intra-molecular 1-4 van der Waals interactions. For example: OPLS uses a factor of $\frac{1}{2}$.

- Intra14ChargeChargeScalingValue [real]
  The scaling factor for intra-molecular 1-4 charge/charge interactions. For example: OPLS uses a factor of $\frac{1}{2}$.

- IdealGasRosenbluthWeight [real]
  The ideal Rosenbluth weight is the growth factor of the CBMC algorithm for a single chain in an empty box. The value only depends on temperature and therefore needs to be computed only once. For adsorption, specifying the value in advance is convenient because the applied pressure does not need to be corrected afterwards (the Rosenbluth weight corresponds to a shift in the chemical potential reference value, and the chemical potential is directly obtained from the fugacity). For equimolar mixtures this is essential.

- GibbsSwapProbability [real]
  The relative probability to attempt a Gibbs swap MC move for the current component. The 'GibbsSwapMove' transfers a randomly selected particle from one box to the other (50% probability to transfer a particle from box I to II, an 50% visa versa).

- TranslationProbability [real]
  The relative probability to attempt a translation move for the current component. A random displacement is chosen in the allowed directions (see 'TranslationDirection'). Note that the internal configuration of the molecule is unchanged by this move. The maximum displacement is scaled during the simulation to achieve an acceptance ratio of 50%.

- TranslationDirection [X|Y|Z|XY|XZ|YZ|XYZ|A|B|C|AB|AC|BC|ABC|
  ORTHOGONAL_TO_AB_DIR|ORTHOGONAL_TO_AC_DIR|ORTHOGONAL_TO_BC_DIR|
  ORTHOGONAL_TO_O_AB_DIR|ORTHOGONAL_TO_O_AC_DIR|ORTHOGONAL_TO_O_BC_DIR|
  ORTHOGONAL_TO_A_BC_DIR|ORTHOGONAL_TO_B_AC_DIR|ORTHOGONAL_TO_C_AB_DIR|
  ORTHOGONAL_TO_O_ABC_DIR]
  Specifies the allowed translation direction for the current component. Useful to sampling configuration with the starting bead restricted to a plane, i.e. see dcTST. Default: XYZ.

- RandomTranslationProbability [real]
  The relative probability to attempt a random translation move for the current component. The displacement is chosen such that any position in the box can reached. It is therefore similar as reinsertion, but 'reinsertion' changes the internal conformation of a molecule and uses biasing.

- RotationProbability [real]
  The relative probability to attempt a random rotation move for the current component. The rotation is around the starting bead. A random vector on a sphere is generated, and the rotation is random around this vector.
- CBMCProbability [real]
  The relative probability to attempt a partial reinsertion move for the current component. Part of the molecule is regrown, while part of the molecule can remain fixed. The list of partial reinsertion moves is specified in the 'molecule.def' file.
- ReinsertionProbability [real]
  The relative probability to attempt a full reinsertion move for the current component. Multiple first beads are chosen, and one of these is selected according to its Boltzmann weight. The remaining part of the molecule is grown using biasing. This move is very useful, and often necessary, to change the internal configuration of flexible molecules.
- SwapProbability [real]
  The relative probability to attempt a insertion or deletion move. Whether to insert or delete is decided randomly with a probability of 50% for each. The swap move imposes a chemical equilibrium between the system and an imaginary particle reservoir for the current component. The move starts with multiple first bead, and grows the remainder of the molecule using biasing.
- WidomProbability [real]
  The relative probability to attempt a Widom particle insertion move for the current component. The Widom particle insertion moves measure the chemical potential and can be directly related to Henry coefficients and heats of adsorption.
- SurfaceAreaProbability [real]
  The relative probability to attempt a surface-area move for the current component.
- ReinsertionInPlaceProbability [real]
  The relative probability to attempt a reinsertion-in-place move for the current component. The reinsertion position is the current position of the starting bead of the randomly selected molecule. Alternatively, one can use the partial reinsertion move leaving one bead fixed. The move is very useful to sample configuration on a plane for dcTST to change the internal configuration, e.g. bonds, bends, torsions, etc.
- IdentityChangeProbability [real]
  The relative probability to attempt an identity-change move for the current component. A molecule of type A is reinsertion, in the same place as the starting bead of A, as type B using the starting bead of component B. The $A - B$ list is defined using 'IdentityChangesList' defining B for each component A, i.e. the current component can be reinserted into any component defined in the 'IdentityChangesList' list, and from that list the component is chosen randomly.
- NumberOfIdentityChanges [int]
  The number of 'IdentityChangesList' elements for the current component.
- IdentityChangesList [list-of-int]
  The list of components that the current component can be changed into. The identity-change move will randomly choose the new component from this list.
- GibbsIdentityChangeProbability [real]
  The relative probability to attempt an identity change for the current component in the Gibbs ensemble. It is a very useful move to for mixture of n components. Out of the n components, two components $i \neq j$ are selected at random. At random, it is selected to switch the identity of component i in box I or in box II, and the identity of the component j in the other box. In each box, a particle is selected at random which matches the desired identity.

- NumberOfGibbsIdentityChanges [int]
  The number of 'GibbsIdentityChangesList' elements for the current component.

- GibbsIdentityChangesList [list-of-int]
  The list of components that the current component can be changed into. The Gibbs-identity-change move will randomly choose the new component from this list.

- ExtraFrameworkMolecule [yes|no]
  There are two major types of molecules, 'Adsorbates' and 'Cations'. The 'ExtraFrameworkMolecule' keyword sets whether the current component is a 'Cation' (yes) or a 'Adsorbate' (no). Energies in the output as splitted in Host-Host, Host-Adsorbate, Host-Cation, Adsorbate-Adsorbate, Cation-Cation, and Adsorbate-Cation. The distinction in two types of molecule is sometimes necessary. For example, consider a mixture of components, where polarization needs to be neglected between certain components (because they are parameterized without). The water model 'rpol' is defined including polarization, but CO2 using TraPPE is not. One can define water as 'Adsorbate', CO2 as 'Cation' and neglect polarization between cations.

- RestrictEnantionface [yes|no]
  Restricts all MC-moves to the enantioface defined by 'Enantioface'. Moves that result in an opposite enantioface are rejected.

- Enantioface [Re|Si]
  The enantioface of the component, either 'Re' or 'Si'.

- EnantiofaceAtoms [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int]
  The definition of the enantioface based on 5 atoms. The first 4 form a torsion, as well as the first 3 and the last atom. These two torsions form the definition of the enantioface.

- CreateNumberOfMolecules [int]
  The number of molecule to create for the current component. Note these molecules are *in addition* to anything read in by using a restart-file. Usually, when the restart-file is used the amount here should be put back to zero. A warning, putting this value unreasonably high results in an infinite loop. The routine accepts molecules that are grown causing no overlap (energy smaller than 'EnergyOverlapCriteria'). Also the initial starting configurations are far from optimal and substantial equilibration is needed to reduce the energy. However, the CBMC growth is able to reach very high densities.

Options to measure properties

- ComputeNumberOfMoleculesHistogram [yes|no]
  Sets whether or not to compute the histograms of the number of molecules for the current system. In open ensembles the number of molecules fluctuates.
  - WriteNumberOfMoleculesHistogramEvery [int]
    Output the histogram every [int] cycles.
  - NumberOfMoleculesRange [real]
    The range of the histograms.
  - NumberOfMoleculesHistogramSize [int]
    The number of elements of the histograms.

- ComputeDistanceHistograms [yes|no]
  Sets whether or not to compute the histograms of specified distance pairs for the current system. A directory 'DistanceHistograms' is created containing the histograms for each system.
  - WriteDistanceHistogramEvery [int]
    Output the distance histograms every [int] cycles.

  - MaxRangeDistanceHistogram [real]
    The range of the histograms.
  - NumberOfElementsDistanceHistogram [int]
    The number of elements of the histograms.
  - DistanceHistogramDefinition [F|A|C] [int] [int] [F|A|C] [int] [int]
    Define a distance histogram between two atoms.

- ComputeBendAngleHistograms [yes|no]
  Sets whether or not to compute the bend-angle histograms of specified trimers of atoms for the current system. A directory 'BendAngleHistograms' is created containing the histograms for each system.
  - WriteBendAngleHistogramEvery [int]
    Output the distance histograms every [int] cycles.
  - MaxRangeBendAngleHistogram [real]
  - NumberOfElementsBendAngleHistogram [int]
  - BendAngleHistogramDefinition [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int]

- ComputeDihedralAngleHistograms [yes|no]
  Sets whether or not to compute the dihedral-angle histograms of specified quads of atoms for the current system. A directory 'DihedralAngleHistograms' is created containing the histograms for each system.
  - WriteDihedralAngleHistogramEvery [int]
    Output the distance histograms every [int] cycles.
  - MaxRangeDihedralAngleHistogram [real]
  - NumberOfElementsDihedralAngleHistogram [int]
  - DihedralAngleHistogramDefinition [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int]

- ComputeAngleBetweenPlanesHistograms [yes|no]
  Sets whether or not to compute the histograms of angles between specified planes for the current system. A directory 'AngleBetweenPlanesHistograms' is created containing the histograms for each system.
  - WriteAngleBetweenPlanesHistogramEvery [int]
    Output the distance histograms every [int] cycles.
  - MaxRangeAngleBetweenPlanesHistogram [real]
  - NumberOfElementsAngleBetweenPlanesHistogram [int]
  - AngleBetweenPlanesHistogramDefinition [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int]

- ComputePSD [yes|no]
  Sets whether or not to compute the pore-size distribution (PSD) for the current system. A directory 'PoreSizeDistributionHistogram' is created containing the output 'HistogramPoreSizeDistribution.dat' per system.

  - WritePSDEvery [int]
    Output the PSD every [int] cycles.
  - PSDProbeDistance [Minimum|Sigma]
    Sets whether to use the minimum of the potential $\sigma^{1/6}$ as the probe distance or whether to use σ.
  - HistogramSizePoreSizeDistribution [int]
    default: 100.
  - MaxRangePoreSizeDistribution [real]
    default: 10.

- ComputeRDF [yes|no]
  Sets whether or not to compute the radial distribution function (RDF) for the current system. A directory 'RadialDistributionFunctions' is created containing the output per system. The RDF is computed for each atom type pair unless the option 'print' flag in 'pseudo_atoms.def' is 'no'.

  - WriteRDFEvery [int]
    Output the RDF every [int] cycles.

- ComputeMSD [yes|no]
  Sets whether or not to compute the mean-squared displacement (MSD) for the current system using a modified order-N algorithm. A directory 'MSDOrderN' is created containing the output per system. The output consists of files containing self-msd data per component, the total self-msd, the Onsager msd for each component pair, and the the total Onsager msd. The units in the files are Å^2 for the msd, and ps for time.

  - WriteMSDEvery [int]
    Output the MSD every [int] cycles.
  - SampleMSDEvery [int]
    Samples every [int] integration steps. Default: 1.
  - ComputeIndividualMSD [yes|no]
    Computes the msd, not only per component, but also per molecule.
  - NumberOfBlocksMSD [int]
    The number of blocks for the order-n correlation measurement. Each block represent a different time-scale of sampling. Default: 25.
  - NumberOfBlockElementsMSD [int]
    The number of elements in each block. For example, if the number is 10, then the first block samples: $1, 2, 3, \dots, 10$, the second block $10, 20, 30, \dots, 100$, the third block $100, 200, 300, \dots, 1000$, etc. Default: 25.

- ComputeVACF [yes|no]
  Sets whether or not to compute the velocity autocorrelation function (VACF) for the current system using a modified order-N algorithm. A directory 'VACFOrderN' is created containing the output per system. The output consists of files containing self-vacf data per component, the total self-vacf, the Onsager vacf for each component pair, and the the total Onsager vacf. The files start with the integration diffusivity-values, computed using a generalization of the Simpson's rule (in the sense that it is exact for cubic polynomials and is valid for an odd as well as even number of intervals). The units in the files are Å^2/ps for velocity, and ps for time.

  - WriteVACFEvery [int]
    Output the VACF every [int] cycles.
  - SampleVACFEvery [int]
    Samples every [int] integration steps. Default: 5.
  - ComputeIndividualVACF [yes|no]
    Computes the vacf, not only per component, but also per molecule.
  - NumberOfBlocksVACF [int]
    The number of blocks for the order-n correlation measurement. Each block represent a different time-scale of sampling. Default: 10.
  - NumberOfBlockElementsVACF [int]
    The number of elements in each block. For example, if the number is 10, then the first block samples: $1, 2, 3, \ldots, 10$, the second block $10, 20, 30, \ldots, 100$, the third block $100, 200, 300, \ldots, 1000$, etc. Default: 5000.

- ComputeRVACF [yes|no]
  Sets whether or not to compute the rotational velocity autocorrelation function (RVACF) for the current system using a modified order-N algorithm. A directory 'RVACFOrderN' is created containing the output per system. The output consists of files containing self-rvacf data per component, the total self-rvacf, the Onsager rvacf for each component pair, and the the total Onsager rvacf. The files start with the integration diffusivity-values, computed using a generalization of the Simpson's rule (in the sense that it is exact for cubic polynomials and is valid for an odd as well as even number of intervals). The units in the files are Å^2/ps for velocity, and ps for time.
  - WriteRVACFEvery [int]
    Output the RVACF every [int] cycles.
  - SampleRVACFEvery [int]
    Samples every [int] integration steps. Default: 5.
  - ComputeIndividualRVACF [yes|no]
    Computes the vacf, not only per component, but also per molecule.
  - NumberOfBlocksRVACF [int]
    The number of blocks for the order-n correlation measurement. Each block represent a different time-scale of sampling. Default: 10.
  - NumberOfBlockElementsRVACF [int]
    The number of elements in each block. For example, if the number is 10, then the first block samples: $1, 2, 3, \ldots, 10$, the second block $10, 20, 30, \ldots, 100$, the third block $100, 200, 300, \ldots, 1000$, etc. Default: 5000.

- ComputeMOACF [yes|no]
  Sets whether or not to compute the molecular orientation velocity autocorrelation function (MOACF) for the current system using a modified order-N algorithm. A directory 'MOACFOrderN' is created containing the output per system. The output consists of files containing self-moacf data per component and the total self-rvacf. The units in the files are rad^2/ps for velocity, and ps for time.
  - WriteMOACFEvery [int]
    Output the MOACF every [int] cycles.
  - SampleMOACFEvery [int]
    Samples every [int] integration steps. Default: 5.
  - ComputeIndividualMOACF [yes|no]
    Computes the moacf, not only per component, but also per molecule.

  - NumberOfBlocksMOACF [int]
    The number of blocks for the order-n correlation measurement. Each block represent a different time-scale of sampling. Default: 10.
  - NumberOfBlockElementsMOACF [int]
    The number of elements in each block. For example, if the number is 10, then the first block samples: $1, 2, 3, \dots, 10$, the second block $10, 20, 30, \dots, 100$, the third block $100, 200, 300, \dots, 1000$, etc. Default: 5000.

- ComputeMSDConventional [yes|no]
  Sets whether or not to compute the mean-squared displacement (MSD) for the current system using the conventional algorithm. A directory 'MSD' is created containing the output per system. The routine is available for legacy reasons, the same results can be obtained using the order-N method and 1 block of size 'BufferLengthMSD'. The units in the files are Å^2 for the msd, and ps for time.
  - WriteMSDConventionalEvery [int]
    Output the MSD every [int] cycles. Default: 5000.
  - SampleMSDConventionalEvery [int]
    Samples every [int] integration steps. Default: 1.
  - NumberOfBuffersMSDConventional [int]
    The number of (overlapping) buffers with a different offset in time. Default: 20.
  - BufferLengthMSDConventional [int]
    The length of the buffers. Default: 5000.

- ComputeVACFConventional [yes|no]
  Sets whether or not to compute the velocity autocorrelation function (VACF) for the current system using the conventional algorithm. A directory 'VACF' is created containing the output per system. The routine is available for legacy reasons, the same results can be obtained using the order-N method and 1 block of size 'BufferLengthVACF'. The units in the files are Å^2/ps for velocity, and ps for time.
  - WriteVACFConventionalEvery [int]
    Output the VACF every [int] cycles. Default: 5000.
  - SampleVACFConventionalEvery [int]
    Samples every [int] integration steps. Default: 1.
  - NumberOfBuffersVACFConventional [int]
    The number of (overlapping) buffers with a different offset in time. Default: 20.
  - BufferLengthVACFConventional [int]
    The length of the buffers. Default: 5000.

- ComputeDensityHistograms [yes|no]
  Sets whether or not to compute a density histogram for the current system. For example, during adsorption it keeps track of the amount of molecules.

- ComputeEnergyHistogram [yes|no]
  Sets whether or not to compute a histogram of the energy for the current system. For example, during adsorption it keeps track of the total energy, the VDW energy, the Coulombic energy, and the polarization energy. Output is written to the directory 'EnergyHistograms'.
  - WriteEnergyHistogramEvery [int]
    Sets to print the energy histogram of the system every [int] cycles.
  - EnergyHistogramSize [int]
    Sets the number of elements of the histogram. Default: 1000.

  - EnergyHistogramLowerLimit [real]
    Sets the lower limit of the histogram. Default: -10000.
  - EnergyHistogramUpperLimit [real]
    Sets the upper limit of the histogram. Default: 0.

- ComputeThermoDynamicFactor [yes|no]
  Sets whether or not to compute the thermodynamic factors of the energy for the current system. The output is written to the directory 'ThermoDynamicFactor'.
  - WriteThermoDynamicFactorEvery [int]
    Sets to print the thermodynamic factors every [int] cycles.
- ComputeEndToEndDistanceHistograms [yes|no]
  Sets whether or not to compute a histogram for end-to-end distances of molecules for the current system.
- ComputePrincipleMomentsOfInertia [yes|no]
  Sets whether or not to compute the average principle moments of inertia of molecules for the current system.
- ComputeSpectra [yes|no]
  Sets whether or not to compute the Infra-Red (IR) spectra of molecules for the current system.
  - WriteSpectraEvery [int]
    Sets to print the spectra of molecules every [int] cycles.
- ComputeMoleculeProperties [yes|no]
  Sets whether or not to compute properties of molecules like average bond-lengths, average bend-angles etc. for the current system.
- PrintMoleculePropertiesEvery [int]
  Sets to print the properties of molecules every [int] cycles.
- ComputeSurfaceArea [yes|no]
  Sets whether or not to compute the surface.
  - SurfaceAreaProbeAtom [string]
  - SurfaceAreaSamplingPointsPerSphere [int]
    Sets the number of points to sampling a sphere per iteration.
  - SurfaceAreaProbeDistance [Minimum|Sigma]
    Sets whether to use the minimum of the potential $\sigma^{1/6}$ as the probe distance or whether to use σ.
- DensityProfile [yes|no]
- DensityProfileGridPoints [int] [int] [int]
- ComputeElasticConstants [yes|no]
  Sets whether to compute elastic constants.
- ComputePowderDiffractionPattern [yes|no]
  Sets whether to compute the powder diffraction pattern for the framework.

- DiffractionType [Xray|Neutron|Electron]
  Sets the diffraction type as xray-scattering, neutron-scattering, or electron-scattering, respectively.
- DiffractionRadiationType [chromium|iron|copper|molybdenum|silver|synchrotron]
  Sets the type of the diffraction radiation as chromium, iron, copper, molybdenum, silver, or synchrotron, respectively.
- WaveLengthType [Single|Double]
  Set the type of the beam as single or as a doublet.
- PeakShape [Gaussian|Lorentzian|PseudoVoigt]
  Sets the shape of the peaks as Gaussian, Lorentzian, or Pseudo-Voigt, respectively.
- WaveLength [real]
  Sets the wavelength of the diffraction beam.
- TwoThetaMin [real]
  Sets the minimum value of 2θ.
- TwoThetaMax [real]
  Sets the maximum value of 2θ.
- TwoThetaStep [real]
  Sets the step size of 2θ.
- PeakWidthModifierU [real]
- PeakWidthModifierV [real]
- PeakWidthModifierW [real]

• ComputerNormalModes [yes|no]
  Sets whether to compute normal modes.

  - MinimumMode [int]
    Sets the minimum normal to compute.
  - MaximumMode [int]
    Sets the maximum normal to compute.
  - ModeResolution [int]

Energy/force grid options

• UseTabularGrid [yes|no]
  Use a pre-tabulated grid for the energy and forces. Default: no.

• SpacingVDWGrid [real]
  The grid spacing of the Van der Waals potentials. Default: 0.15 Angstrom.

• SpacingCoulombGrid [real]
  The grid spacing of the Coulomb potential. Default: 0.15 Angstrom.

• GridTypes [list-of-strings]
  A list of atom-types for each of the used grids.

Minimization/Saddle point search

- MinimizationMethod [Baker]
  The Baker minimization method uses the eigenvalues/vectors to find a true minimum where all eigenvalues are positive. Newton-Raphson uses the first and second derivatives, but not the eigenvalues/vectors. The saddle point search can best be started from a minimum energy configuration. The algorithm walks up hill along the softest eigen mode to find a first order saddle point.

- MinimizationVariables [Cartesian|Fractional]
  Whether the minimization is performed in Cartesian or fractional positions. For some crystal minimizations it might be more convenient to choose fractional positions. An example is when one wants to keep a particular fractional position fixed during the minimization.

- MaximumNumberOfMinimizationSteps [int]
  The maximum number of minimization steps after which the minimization is stopped. Default: 10000.

- RMSGradientTolerance [real]
  Stopping criteria: the maximum allowed RMS gradient. Default: 10^{-6}.

- MaxGradientTolerance [real]
  Stopping criteria: the maximum allowed gradient for each and every atom (and the strain elements for cell minimizations). Default: 10^{-6}.

- MaximumStepLength [real]
  The maximum length of a minimization step. The length is dependent on the problem at hand. A too low value converges slowly (i.e. the minimization takes more steps), while a too high value might not converge at all. Default value: 0.3.

- FrameworkFixedInitialization [free|fixed]
  Sets all framework atoms as 'free' or 'fixed'. This command must preceed individual overwrites and applies to the current system.

- AdsorbateFixedInitialization [free|fixed]
  Sets all adsorbate groups and atoms as 'free' or 'fixed'. This command must preceed individual overwrites and applies to the current system.

- CationFixedInitialization [free|fixed]
  Sets all cation groups and atoms as 'free' or 'fixed'. This command must preceed individual overwrites and applies to the current system.

- ActiveFrameworkAtom [int]
  Sets the atom of the current framework and system as 'active'.

- ActiveFrameworkAtoms [int] [list-of-ints]
  Sets the [int] atoms listed in [list-of-ints] of the current framework and system as 'active'.

- FixedFrameworkAtom [int]
  Sets the atom of the current framework and system as 'fixed'.

- FixedFrameworkAtoms [int] [list-of-ints]
  Sets the [int] atoms listed in [list-of-ints] of the current framework and system as 'fixed'.

- ActiveAdsorbateMolecule [int]
  Sets all atom and groups of the adsorbate molecule [int] as 'active'. Applies to the current system.

- FixedAdsorbateMolecule [int]
  Sets all atom and groups of the adsorbate molecule [int] as 'fixed'. Applies to the current system.

- ActiveAdsorbateAtom [int] [int]
  Sets an atom (second argument) of an adsorbate molecule (first argument) as ‘active’. Applies to the current system.
- FixedAdsorbateAtom [int] [int]
  Sets an atom (second argument) of an adsorbate molecule (first argument) as ‘fixed’. Applies to the current system.
- ActiveAdsorbateGroup [int] [int]
  Sets a group (second argument) of an adsorbate molecule (first argument) as ‘active’. Applies to the current system and both center of mass and the orientation are set as ‘active’.
- FixedAdsorbateGroup [int] [int]
  Sets a group (second argument) of an adsorbate molecule (first argument) as ‘fixed’. Applies to the current system and both center of mass and the orientation are set as‘fixed’.
- ActiveAdsorbateGroupCenterOfMass [int] [int]
  Sets a group (second argument) of an adsorbate molecule (first argument) as ‘active’. Applies to the current system and only the center of mass is set as ‘active’.
- FixedAdsorbateGroupCenterOfMass [int] [int]
  Sets a group (second argument) of an adsorbate molecule (first argument) as ‘fixed’. Applies to the current system and only the center of mass is set as ‘fixed’.
- ActiveAdsorbateGroupOrientation [int] [int]
  Sets a group (second argument) of an adsorbate molecule (first argument) as ‘active’. Applies to the current system and only the orientation is set as ‘active’.
- FixedAdsorbateGroupOrientation [int] [int]
  Sets a group (second argument) of an adsorbate molecule (first argument) as ‘fixed’. Applies to the current system and only the orientation is set as ‘fixed’.
- ActiveCationMolecule [int]
  Sets all atom and groups of the cation molecule [int] as ‘active’. Applies to the current system.
- FixedCationMolecule [int]
  Sets all atom and groups of the cation molecule [int] as ‘fixed’. Applies to the current system.
- ActiveCationAtom [int] [int]
  Sets an atom (second argument) of an cation molecule (first argument) as ‘active’. Applies to the current system.
- FixedCationAtom [int] [int]
  Sets an atom (second argument) of an cation molecule (first argument) as ‘fixed’. Applies to the current system.
- ActiveCationGroup [int] [int]
  Sets a group (second argument) of an cation molecule (first argument) as ‘active’. Applies to the current system and both center of mass and the orientation are set as ‘active’.
- FixedCationGroup [int] [int]
  Sets a group (second argument) of an cation molecule (first argument) as ‘fixed’. Applies to the current system and both center of mass and the orientation are set as ‘fixed’.
- ActiveCationGroupCenterOfMass [int] [int]
  Sets a group (second argument) of an cation molecule (first argument) as ‘active’. Applies to the current system and only the center of mass is set as ‘active’.

- FixedCationGroupCenterOfMass [int] [int]
  Sets a group (second argument) of an cation molecule (first argument) as ‘fixed’. Applies to the current system and only the center of mass is set as ‘fixed’.
- ActiveCationGroupOrientation [int] [int]
  Sets a group (second argument) of an cation molecule (first argument) as ‘active’. Applies to the current system and only the orientation is set as ‘active’.
- FixedCationGroupOrientation [int] [int]
  Sets a group (second argument) of an cation molecule (first argument) as ‘fixed’. Applies to the current system and only the orientation is set as ‘fixed’.
- FixAtomType [string]
  FixAtomTypes [int] [list-of-strings]
  The atom-types that are considered fixed during the minimization. All other atoms/groups will be optimized. If the atom-type is contained in a rigid unit, the entire unit will be frozen.
- DistanceConstraint [F|A|C] [int] [int] [F|A|C] [int] [int] [real]
  Defines a ‘hard’ distance constraint between two atoms and/or groups, and the distance.
- AngleConstraint [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int] [real]
  Defines a ‘hard’ angular constraint between three atoms and/or groups, and the constraint angle.
- DihedralConstraint [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int] [real]
  Defines a ‘hard’ dihedral constraint between four atoms and/or groups, and constraint dihedral.
- HarmonicDistanceConstraint [F|A|C] [int] [int] [F|A|C] [int] [int] [real] [real]
  Defines a ‘hard’ distance constraint between two atoms and/or groups, and the distance.
- HarmonicAngleConstraint [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int] [real] [real]
  Defines a ‘hard’ angular constraint between three atoms and/or groups, and the constraint angle.
- HarmonicDihedralConstraint [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int] [F|A|C] [int] [int] [real] [real]
  Defines a ‘hard’ dihedral constraint between four atoms and/or groups, and constraint dihedral.

Monte Carlo settings

- MinimumInnerCycles [int]
  The minimum number of inner cycles for each cycle. Default: 20.
- NumberOfTrialPositions [int]
  The number of trial positions during the growth of a molecule. Default: 10.
- NumberOfTrialPositionsForTheFirstBead [int]
  The number of trial positions for the first bead. Default: 10.
- NumberOfTrialPositionsTorsion [int]
  The number of trial positions for torsions over a single bond. Default: 100.
- NumberOfTrialMovesPerOpenBead [int]
  The number of trial moves per open bead during CBMC. Default: 200.
- TargetAccRatioSmallMCScheme [real]
- TargetAccRatioTranslation [real]

- EnergyOverlapCriteria [real]
  The energy criteria to consider an energy as 'overlap'. Default: 10^5 K.

- MinimumRosenbluthFactor [real]
  The minimum Rosenbluth weight, values lower are consider to be 'overlapping'. Default: 10^{-150}.

Biasing options

- BiasingDirection [A|B|C|AB_DIAGONAL|AC_DIAGONAL|BC_DIAGONAL|
  A_BC_DIAGONAL|B_AC_DIAGONAL|C_AB_DIAGONAL|
  O_ABC_DIAGONAL]

- BiasingMethod [UMBRELLA|RUIZMONTERO]

- BiasingProfile [string]
  The name of the file containing the biasing profile.

- RuizMonteroFactor [real]

- UmbrellaFactor [real]
  The biasing free energy is multiplied by the UmbrellaFactor. This is useful when the biasing free energy goes to infity in certain regions. if the exact free energy would be used to biased, then the histogram would be flat, even very close to atoms. To keep the repulsion one can lower the used free energy biasing by e.g. multiplying by 0.9.

- RestrictMovesToUnitCell [yes|no]
  Restrict the Monte-Carlo moves to the first unitcell for this component.

- BoxAxisABC_Min [real]
  When a particle is restricted in all Monte-Carlo moves (RestrictMovesToUnitCell or RestrictMovesToBox) then do not allow trial moves with a fractional position smaller than BoxAxisABC_Min.

- BoxAxisABC_Max [real]
  When a particle is restricted in all Monte-Carlo moves (RestrictMovesToUnitCell or RestrictMovesToBox) then do not allow trial moves with a fractional position greater than BoxAxisABC_Max.

Transition State Theory settings

- FreeEnergyMappingType [A_MAPPING|B_MAPPING|C_MAPPING|ABC_MAPPING|
  MAP_AB_DIAGONAL|MAP_AC_DIAGONAL|MAP_BC_DIAGONAL|
  MAP_A_BC_DIAGONAL|MAP_B_AC_DIAGONAL|MAP_C_AB_DIAGONAL|
  MAP_O_ABC_DIAGONAL]
  Determines how the free energy profile is constructed from the contributions of points in the unit cell. The free energy is computed using Widom insertion by inserting probe molecules at many random position inside the unit cell. The 'FreeEnergyMappingType' maps a Cartesian position on a reaction coordinate 'q'. The mappings 'A_MAPPING', 'B_MAPPING', 'C_MAPPING' map the Cartesian position onto the 'a', 'b', 'c' lattice vectors. The diagonal mapping maps onto diagonal, either in 2D or in 3D. For example, 'MAP_A_BC_DIAGONAL' maps onto the line from 'A' to 'B+C' where 'A','B', and 'C' are the end points of the lattive vectors; and 'MAP_O_ABC_DIAGONAL' maps onto the line from the origin to the opposite point 'A+B+C' on the diagonal.

- PositionHistogramMappingType [A_MAPPING|B_MAPPING|C_MAPPING|ABC_MAPPING|
  MAP_AB_DIAGONAL|MAP_AC_DIAGONAL|MAP_BC_DIAGONAL|
  MAP_A_BC_DIAGONAL|MAP_B_AC_DIAGONAL|MAP_C_AB_DIAGONAL|
  MAP_O_ABC_DIAGONAL]
  Determines how the position histogram is constructed from the contributions of points in the unit cell. The free energy is computed from the histogram by using $F(q) = -\log[P(q)]$. The 'PositionHistogramMappingType' maps a Cartesian position on a reaction coordinate 'q'. The mappings 'A_MAPPING', 'B_MAPPING', 'C_MAPPING' map the Cartesian position onto the 'a', 'b', 'c' lattice vectors. The diagonal mapping maps onto diagonal, either in 2D or in 3D. For example, 'MAP_A_BC_DIAGONAL' maps onto the line from 'A' to 'B+C' where 'A','B', and 'C' are the end points of the lattive vectors; and 'MAP_O_ABC_DIAGONAL' maps onto the line from the origin to the opposite point 'A+B+C' on the diagonal.

- PutMoleculeOnBarrier [yes|no]
  Places the first molecule of component 0 at the position given by 'BarrierPosition'. This is used e.g. to start sampling configuration on top of a free energy barrier.

- BarrierPosition [real] [real] [real]
  The location of the free energy barrier in fractional units of the first unit cell.

- MaxBarrierDistance [real]
  The maximum distance in Ångstrom of the dcTST trajectory.

- MaxBarrierTime [real]
  The maximum time of the dcTST trajetory in picoseconds.

- NumberOfVelocities [int]
  The number of times the same initial position of the sampled dcTST starting configurations is used with different initial velocities.

- WritedcTSTSnapShotsToFile [yes|no]
  Whether to write out sampled configuration to a file. The file is stored in the directory 'dcTST_starting_configurations' and used as the tarting point to compute the transmission coefficient in dcTST.

- WritedcTSTSnapShotsEvery [int]
  The frequency in MC cycles of writing out the sampled configurations. Default: 1000.

2.3 Force field

2.3.1 Force fields

2.3.2 'pseudo_atoms.def'

The 'pseudo_atoms.def' files describes the (pseudo-)atoms to be used in the simulation. An example is is the definitions for the tip5p water model:

```
#number of pseudo atoms
3
#type  print as  scat  oxidation mass      charge  polarization B-factor radii connectivity anisotropic anisotropic-type  tinker-type
Ow     yes   O     O           0 15.9994   0.0     0.0          1.0      0.5   2            0.0         absolute          0
Hw     yes   H     H           0 1.0008    0.241   0.0          1.0      1.00  1            0.0         absolute          0
L      yes   L     -           0 0.0      -0.241   0.0          1.0      1.00  1            0.0         absolute          0
```

The first line is skipped, the second line is the number of (pseudo-)atoms, the third line is skipped again, and next all the (pseudo-)atoms are specified. The format and meaning is:

name	An unique string of character to be used to identify the atom. The same name has be used in other files to refer to this atom.
print	Whether or not this atom should be printed to movies. The dummy 'L' atoms of the tip5p water model are an example where you would like them to be skipped, only the 'O' and 'H' atoms should be printed.
as	The string to be printed to movies.
scat	The chemical symbol, e.g. O, O^-, O^{2-}. They are defined in 'scattering_factors.c' and are used only in powder diffraction and spectra.
oxidation	not used yet
mass	The mass of the atom in atomic units.
charge	The charge of the atom in atomic units.
polarization	not used yet
B-factor	The temperature factor of the atom, used only in powder diffraction.
radius	The radius of the atom to be used to decide what atoms are considered as 'neighbors'. The current rule is that two atoms i and j are considered 'bonded' if the distance between the atoms is smaller then 0.56+Radius$_i$+Radius$_j$.
connectivity	The connectivity of the atoms (not yet used).
anisotropic factor	The magntitude of the anisotropy.
anisotropic-type	The type of anisotropy, either 'relative' or 'absolute'. For example, a relative anisotropic factor of hydrogen of -0.077, used in the MM3 force field, means the site is pulled inward by 7.7% (and located at 92.3% of the C-H bondlength). An absolute anisotropic factor of e.g. 0.3 means the site is displaced outward by 0.3Å.
Tinker-type	The type of the atom in other codes, e.g. Tinker. This is used for output-files in formats used in other codes.

2.3.3 'force_field_mixing_rules.def'

```
# general rule for shifted vs truncated
shifted
# general rule tailcorrections
no
# number of defined interactions
9
# type interaction
Zn1            lennard-jones    0.42      2.7
O1             lennard-jones    700.0     2.98
O2             lennard-jones    70.5      3.11
C1             lennard-jones    48.5      3.76
C2             lennard-jones    47.86     3.47
C3             lennard-jones    47.86     3.47
H1             lennard-jones    7.65      2.85
O_co2          lennard-jones    80.507    3.033
C_co2          lennard-jones    28.129    2.757
# general mixing rule for Lennard-Jones
Jorgensen
```

The first line is skipped, the second line is the general cutoff rule for shifted vs truncated, the third line is skipped, the fourth line is the general tule for tail corrections, the fifth line is skipped, the sixth line is the number of defined self-interactions for the (pseudo-) atoms. The next line is skipped again followed by the defined potentials for the (pseudo-)atoms. The file is ended with a skipped line and the general rule for the mixing rule. Note all these interactions can be subsequently overwritten using the 'force_field.def' file for specific interactions.

For convenience, you can use pattern matching of the interactions. Any string s_1 ending with an underscore will match any string s_2 that starts with the substring s_1. Note that usually the '*' symbol is used, but this symbol has already a different meaning for CIF-files. Of course patterns can match more than one atom, e.g. `'C_'` matches `'C1'` (a carbon atom) but also `'Cl'` (chloride), and the rules are applied top to bottom. Therefore, list the generic ones first and the more specific after the generic patterns.

Example of a generic UFF/TraPPE force field for united atom alkanes in MOFs:

```
# general rule for shifted vs truncated
shifted
# general rule tailcorrections
no
# number of defined interactions
32
# type interaction, parameters.    IMPORTANT: define generic matches first
O_             lennard-jones    48.1581   3.03315
N_             lennard-jones    38.9492   3.26256
C_             lennard-jones    47.8562   3.47299
F_             lennard-jones    36.4834   3.0932
B_             lennard-jones    47.8058   3.58141
Cl_            lennard-jones   142.562    3.51932
Br_            lennard-jones   186.191    3.51905
H_             lennard-jones     7.64893  2.84642
Zn_            lennard-jones    62.3992   2.46155
Be_            lennard-jones    42.7736   2.44552
Cr_            lennard-jones     7.54829  2.69319
Fe_            lennard-jones     6.54185  2.5943
Mn_            lennard-jones     6.54185  2.63795
Cu_            lennard-jones     2.5161   3.11369
Co_            lennard-jones     7.04507  2.55866
Ga_            lennard-jones   208.836    3.90481
Ti_            lennard-jones     8.55473  2.8286
Sc_            lennard-jones     9.56117  2.93551
V_             lennard-jones     8.05151  2.80099
Ni_            lennard-jones     7.54829  2.52481
Zr_            lennard-jones    34.7221   2.78317
Mg_            lennard-jones    55.8574   2.69141
Ne_            lennard-jones    21.1352   2.88918
Ag_            lennard-jones    18.1159   2.80455
In_            lennard-jones   301.428    3.97608
Cd_            lennard-jones   114.734    2.53728
Sb_            lennard-jones   225.946    3.93777
Te_            lennard-jones   200.281    3.98232
He             lennard-jones    10.9      2.64
CH4_sp3        lennard-jones    158.5     3.72
CH3_sp3        lennard-jones    108.0     3.76
CH2_sp3        lennard-jones    56.0      3.96
# general mixing rule for Lennard-Jones
Lorentz-Berthelot
```

Here, `'CH4_sp3'`, `'CH3_sp3'`, and `'CH4_sp3'` are first matched by `'C_'` but later overwritten with the correct values. However, carbon atoms listed in the MOF CIF-file, like `'C1'`, `'C2'`,`'C3'`, etc. will be set to the `'C_'` value as intended.

general cutoff rule	‘shifted’ or ‘truncated’	‘shifted’ shifts the potentials to zero at the cutoff radius, ‘truncated’ leaves them unchanged.
general tail corrections rule	‘yes’ or ‘no’	‘yes’ applies the tail corrections to all interactions, ‘no’ omits the tail corrections for all interactions
general mixing-rule (only used for Lennard-Jones)	‘Jorgensen’ or ‘Lorentz-Berthelot’	‘Jorgensen’ $\{\epsilon_{ij}=\sqrt{\epsilon_i\epsilon_j},\sigma_{ij}=\sqrt{\sigma_i\sigma_j}\}$ ‘Lorentz-Berthelot’ $\{\epsilon_{ij}=\sqrt{\epsilon_i\epsilon_j},\sigma_{ij}=\frac{1}{2}(\sigma_i+\sigma_j)\}$
self interaction type	‘zero-potential’,‘12-6’,‘Lennard-Jones’, ‘Buckingham’, ‘MCY’, ‘generic’, ‘HIW’, ‘MIE’, ‘BHM’, or ‘Hydrogen’	the type of the potential determines the subsequent parameters, i.e. Lennard-Jones expects a strength parameter ϵ_{ii} and a size parameter σ_{ii}.

2.3.4 ‘force_field.def’

The ‘force_field_mixing_rules.def’ file given above can be used for the flexible model of the metal-organic framework IRMOF-1. It is defined using the Jorgensen mixing rule and uses shifted potentials cutoff at 12 Å. The EMP2-CO_2 model however uses the Lorentz-Berthelot for CO_2-CO_2 interactions and uses a truncated potential with tail corrections. Moreover, if we also want to use the DREIDING model for the CO_2-framework interactions the correction-file ‘force_field.def’ would look like:

```
# rules to overwrite
3
# pair      truncated/shifted tailcorrections
O_co2 O_co2 truncated         yes
O_co2 C_co2 truncated         yes
C_co2 C_co2 truncated         yes
# number of defined interactions
14
# type      type2       interaction
Zn1         C_co2       lennard-jones  27.34776042 3.420
Zn1         O_co2       lennard-jones  46.77926891 3.545
O1          C_co2       lennard-jones  36.07117963 2.915
O1          O_co2       lennard-jones  61.70097244 3.04
O2          C_co2       lennard-jones  36.07117963 2.915
O2          O_co2       lennard-jones  61.70097244 3.04
C1          C_co2       lennard-jones  35.94746166 3.135
C1          O_co2       lennard-jones  61.48934867 3.26
C2          C_co2       lennard-jones  35.94746166 3.135
C2          O_co2       lennard-jones  61.48934867 3.26
C3          C_co2       lennard-jones  35.94746166 3.135
C3          O_co2       lennard-jones  61.48934867 3.26
H1          C_co2       lennard-jones] 14.37184748 2.825
H1          O_co2       lennard-jones] 24.58353107 2.95
# mixing rules to overwrite
1
#
O_co2 C_co2 Lorentz-Berthelot
```

Tip: always double check the complete interaction set printed in the output !

2.4 Molecules

The format of the molecules is designed to allow for a combination of flexible and rigid subunits. A molecule is made up of ‘groups’, where a group is a collection of either rigid or flexible atoms.

2.4.1 Rigid molecule

An example of CO_2 as a rigid molecule.

```
# critical constants: Temperature [T], Pressure [Pa], and Acentric factor [-]
304.1282
7377300.0
0.22394
#Number Of Atoms
3
# Number of groups
1
# CO2-group
3
rigid
0 O_co2     0.0           0.0           1.16
1 C_co2     0.0           0.0           0.0
2 O_co2     0.0           0.0          -1.16
# Chiral centers Bond  BondDipoles Bend  UrayBradley InvBend  Torsion Imp. Torsion Bond/Bond Stretch/Bend Bend/Bend Stretch/Torsion Bend/Torsion IntraVDW IntraCoulomb
               0    2            0    0            0       0        0            0         0            0          0               0            0        0            0
# Bond stretch: atom n1-n2, type, parameters
0 1 RIGID_BOND
1 2 RIGID_BOND
# Number of config moves
0
```

The first three numbers are the critical constants: the critical temperature, the critical pressure, and the acentric factor. They are used to automatically compute the fugacity from the pressure using an equation of state, e.g. Peng Robinson. Then the number of atoms and the number of groups. The groups are listed one by one with first the number of atoms in the group, whether it is rigid or flexible, and the atoms as number, type, and for a rigid molecule the relative positions. After the groups follows the bond, bend, torsion etc. parameters. The file is ended with the config moves.

2.4.2 Flexible molecule

An example of a flexible molecule is the united-atom 2-methylbutane molecule. Note that for flexible units there is not need to list relative positions. Bond-potentials are listed as the two atoms on which the potential operates, the potential type and the corresponding parameters. At the end we have 2 config moves defined, one where atoms 0,1,2 are kept fixed and the rest is regrown, and another config move where only atoms 2,3 are kept fixed.

```
# critical constants: Temperature [T], Pressure [Pa], and Acentric factor [-]
460.35
3395700.0
0.2296
# Number Of Atoms
5
# Number Of Groups
1
# Alkane-group
5
flexible
0 CH3_sp3
1 CH_sp3
2 CH2_sp3
3 CH3_sp3
4 CH3_sp3
# Chiral centers Bond  BondDipoles Bend  UrayBradley InvBend  Torsion Imp. Torsion Bond/Bond Stretch/Bend Bend/Bend Stretch/Torsion Bend/Torsion IntraVDW IntraCoulomb
               0    4            0    4            0       0        2            0         0            0          0               0            0        0            0
# Bond stretch: atom n1-n2, type, parameters
0 1 HARMONIC_BOND 96500 1.54
1 2 HARMONIC_BOND 96500 1.54
1 4 HARMONIC_BOND 96500 1.54
2 3 HARMONIC_BOND 96500 1.54
# Bond bending: atom n1-n2-n3, type, parameters
0 1 2 HARMONIC_BEND 62500 112
0 1 4 HARMONIC_BEND 62500 112
4 1 2 HARMONIC_BEND 62500 112
1 2 3 HARMONIC_BEND 62500 114
# Torsion n1-n2-n3-n4 type
```

```
0 1 2 3 OPLS_DIHEDRAL -251.06  428.73  -111.85  441.27
4 1 2 3 OPLS_DIHEDRAL -251.06  428.73  -111.85  441.27
# Number of config moves
2
# nr_fixed followed by a list
3 0 1 2
2 2 3
```

2.4.3 Rigid/Flexible molecule

Flexible and rigid units can easily be combined, as shown for the 1,4-benzenedicarboxylate (BDC) molecule. Note that the relative positions in the rigid units are recomputed in the molecular reference frame.

```
# critical constants: Temperature [T], Pressure [Pa], and Acentric factor [-]
0.0
0.0
0.0
#Number Of Atoms
16
# Number of groups
3
# carboxyl-group
3
flexible
0  Mof_Ob
1  Mof_Ca
2  Mof_Ob
# phenyl-ring
10
rigid
3  Mof_Cb 6.458 6.458  11.526
4  Mof_Cc 7.308 7.308  12.221
5  Mof_Cc 5.608 5.608  12.221
6  Mof_H  7.876 7.876  11.759
7  Mof_H  5.04  5.04   11.759
8  Mof_Cc 7.308 7.308  13.611
9  Mof_Cc 5.608 5.608  13.611
10 Mof_H  7.876 7.876  14.073
11 Mof_H  5.04  5.04   14.073
12 Mof_Cb 6.458 6.458  14.306
# carboxyl-group
3
flexible
13 Mof_Ca
14 Mof_Ob
15 Mof_Ob
# Chiral centers Bond  BondDipoles Bend  UrayBradley InvBend  Torsion Imp. Torsion Bond/Bond Stretch/Bend Bend/Bend Stretch/Torsion Bend/Torsion IntraVDW IntraCoulomb
               0   16            0   10            0       0       16            0         0            0         0               0            0       80          80
# Bond stretch: atom n1-n2, type, parameters
0 1 HARMONIC_BOND  543840.64928424  1.27
1 2 HARMONIC_BOND  543840.64928424  1.27
1 3 HARMONIC_BOND  353750.919316375 1.44
3 4 RIGID_BOND
3 5 RIGID_BOND
4 6 RIGID_BOND
5 7 RIGID_BOND
4 8 RIGID_BOND
5 9 RIGID_BOND
8 10 RIGID_BOND
9 11 RIGID_BOND
8 12 RIGID_BOND
9 12 RIGID_BOND
12 13 HARMONIC_BOND  353750.919316375 1.44
13 14 HARMONIC_BOND  543840.64928424  1.27
13 15 HARMONIC_BOND  543840.64928424  1.27
...
...
```

2.4.4 Chiral molecules

44methylethyloctane, the left-handed form:

```
# critical constants: Temperature [T], Pressure [Pa], and Acentric factor [-]
535.6
2847232.5
0.325
# Number Of Atoms
11
# Number of groups
1
# octane-group
11
flexible
 0 CH3_sp3
 1 CH2_sp3
 2 CH2_sp3
 3 C_sp3
 4 CH2_sp3
 5 CH2_sp3
 6 CH2_sp3
 7 CH3_sp3
 8 CH3_sp3
 9 CH2_sp3
10 CH3_sp3
```

```
# Chiral centers Bond  BondDipoles Bend  UrayBradley InvBend  Torsion Imp. Torsion Bond/Bond Stretch/Bend Bend/Bend Stretch/Torsion Bend/Torsion IntraVDW IntraCoulomb
               1   10            0   12            0       0        6            0         0            0         0               0            0       21            0
# chiral center
2 3 4 8 L
# Bond stretch: atom n1-n2, type, parameters
0  1 HARMONIC_BOND 96500 1.54
1  2 HARMONIC_BOND 96500 1.54
2  3 HARMONIC_BOND 96500 1.54
3  4 HARMONIC_BOND 96500 1.54
4  5 HARMONIC_BOND 96500 1.54
5  6 HARMONIC_BOND 96500 1.54
6  7 HARMONIC_BOND 96500 1.54
3  8 HARMONIC_BOND 96500 1.54
3  9 HARMONIC_BOND 96500 1.54
9 10 HARMONIC_BOND 96500 1.54
# Bond bending: atom n1-n2-n3, type, parameters
0 1  2 HARMONIC_BEND 62500 114
1 2  3 HARMONIC_BEND 62500 114
2 3  4 HARMONIC_BEND 62500 109.47
2 3  8 HARMONIC_BEND 62500 109.47
2 3  9 HARMONIC_BEND 62500 109.47
8 3  4 HARMONIC_BEND 62500 109.47
9 3  4 HARMONIC_BEND 62500 109.47
3 4  5 HARMONIC_BEND 62500 114
9 3  8 HARMONIC_BEND 62500 109.47
3 9 10 HARMONIC_BEND 62500 114
4 5  6 HARMONIC_BEND 62500 114
5 6  7 HARMONIC_BEND 62500 114
# Torsion: atom n1-n2-n3-n4, type, parameters
 0 1 2  3 TRAPPE_DIHEDRAL      0.0   355.03  -68.19   791.32
 1 2 3  4 TRAPPE_DIHEDRAL      0.0     0.0      0.0   461.29
 1 2 3  8 TRAPPE_DIHEDRAL      0.0     0.0      0.0   461.29
 1 2 3  9 TRAPPE_DIHEDRAL      0.0     0.0      0.0   461.29
 2 3 4  5 TRAPPE_DIHEDRAL      0.0     0.0      0.0   461.29
 2 3 9 10 TRAPPE_DIHEDRAL      0.0     0.0      0.0   461.29
 8 3 4  5 TRAPPE_DIHEDRAL      0.0     0.0      0.0   461.29
10 9 3  4 TRAPPE_DIHEDRAL      0.0     0.0      0.0   461.29
 9 3 4  5 TRAPPE_DIHEDRAL      0.0     0.0      0.0   461.29
 3 4 5  6 TRAPPE_DIHEDRAL      0.0   355.03  -68.19   791.32
10 9 3  8 TRAPPE_DIHEDRAL      0.0     0.0      0.0   461.29
 4 5 6  7 TRAPPE_DIHEDRAL      0.0   355.03  -68.19   791.32
# Intra VDW: atom n1-n2
0  4
0  5
0  6
0  7
0  8
0  9
0 10
1  5
1  6
1  7
1 10
2  6
2  7
3  7
5 10
6  8
6  9
6 10
7  8
7  9
7 10
# Number of config moves
0
```

while the right-handed form has

```
# chiral center
2 3 4 8 R
```

2.5 Framework

2.5.1 Asymmetric unit cell

Frameworks are often presented in literature using as much symmetry as possible to reduced the amount of atoms needed to describe the structure. Usually only the fractional positions of the atoms in the *asymmetric unit cell* are given. Given a space group and the unit cell parameters (length and angles) all other positions in the full unit cell can be generated. For example, the isoreticular metal-organic framework IRMOF-1 is published as 7 fractional positions, space group 225, a cubic unit cell with cell lengths of 25.832 Å, and $\alpha = \beta = \gamma = 90°$. RASPA can read cif-files, and the structure can be put into a file (see 'IRMOF-1.cif' in 'structures/mofs/cif'):

```
data_IRMOF-1

_cell_length_a    25.832
```

```
_cell_length_b    25.832
_cell_length_c    25.832
_cell_angle_alpha 90
_cell_angle_beta  90
_cell_angle_gamma 90
_cell_volume      17237.5

_symmetry_cell_setting          cubic
_symmetry_space_group_name_Hall '-F 4 2 3'
_symmetry_space_group_name_H-M  'F m -3 m'
_symmetry_Int_Tables_number     225

loop_
_atom_site_label
_atom_site_type_symbol
_atom_site_fract_x
_atom_site_fract_y
_atom_site_fract_z
Zn1      Zn     0.2934     0.2066     0.2066
O1       O      0.25       0.25       0.25
O2       O      0.2819     0.2181     0.134
C1       C      0.25       0.25       0.1113
C2       C      0.25       0.25       0.0538
C3       C      0.2829     0.2171     0.0269
H1       H      0.3049     0.1951     0.0448
```

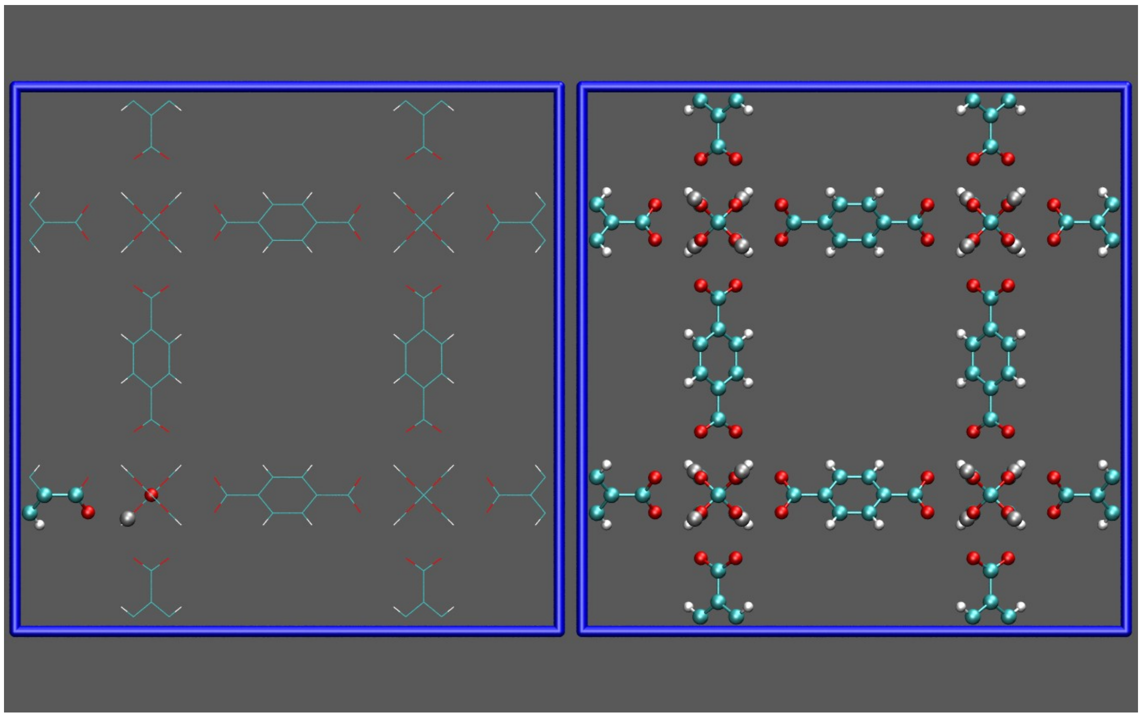
RASPA can then be run using:

```
SimulationType       MonteCarlo
NumberOfCycles       0
InitializationCycles 0

Forcefield           GenericMOFs

Framework     0
FrameworkName IRMOF-1
UnitCells     1 1 1
```

and in the directory 'Movies/System_0/i' several files appear:

- 'Framework_0_initial_1_1_1.cif'
  The framework in CIF-format at the start of the simulation.
- 'Framework_0_initial_1_1_1_P1.cif'
  The framework in CIF-format at the start of the simulation converted to P1 (no symmetry).
- 'Framework_0_initial.pdb'
  The framework in PDB-format at the start of the simulation converted to P1 (no symmetry).

The files named 'final' are the structures at the end of the simulation. There are several programs that can read and view CIF-files: e.g. Jmol (free), Mercury (free), Crystal Maker (commercial, free demo), Materials Studio (commercial), and Gaussview (commercial). The PDB-files can be viewed in the freely available VMD-program.



(a) *The seven asymmetric atoms of IRMOF-1.* **(b)** *The full unit cell of IRMOF-1 has 424 atoms.*

Figure 1: *Asymmetric unit cells: the left figure shows the seven crystallographicly different atoms in the IRMOF-1 structure in ball-and-stick format. The 'copies' (crystallographically identical atoms) are shown as lines. The right figure shows the full unit cell of IRMOF-1 in ball-and-stick.*

Tip: always double check the '`Framework_0_initial_1_1_1_P1.cif`', if you see something strange then check '`_symmetry_space_group_name_Hall`' and the fractional positions.

Space group 225 has 192 elements and the first 10 elements look like (see the file 'src/spacegroup.c' for the complete set):

$$\begin{array}{lll}
x' = x & y' = y & z' = z \\
x' = -x & y' = -y & z' = z \\
x' = -x & y' = y & z' = -z \\
x' = x & y' = -y & z' = -z \\
x' = z & y' = x & z' = y \\
x' = z & y' = -x & z' = -y \\
x' = -z & y' = -x & z' = y \\
x' = -z & y' = x & z' = -y \\
x' = y; & y' = z & z' = x \\
x' = -y & y' = z & z' = -x \\
x' = y & y' = -z & z' = -x \\
\ldots & &
\end{array}$$

The procedure to generate a unit cell is to loop over the elements of the spacegroup and the atoms in the asymmetric unit cell, and to apply simply all the rule. For each new x', y', z' position a check is needed whether the same position has already been added (doubles have to be removed). After this procedure the 7 positions have been expanded to 424 positions. The fractional positions are transformed in the final step to Cartesian positions.

2.5.2 Fractional occupancies in zeolites

The procedure from asymmetric to full unit cell is rather simple when the fractional occupancies are unity. However, quite often there is some disorder the type of atoms. For example, in zeolites like FAU the Si/Al ratio is specified, but it is unknown where the aluminum actually is. Zeolite X is faujasite with a high amount of aluminum. The FAU structure with a Si/Al ratio of unity is given by

```
data_NaX

_audit_creation_method RASPA-1.0
_audit_creation_date 2011-2-20
_audit_author_name 'David Dubbeldam'

_cell_length_a    25.099
_cell_length_b    25.099
_cell_length_c    25.099
_cell_angle_alpha 90
_cell_angle_beta  90
_cell_angle_gamma 90
_cell_volume      14273.9

_symmetry_cell_setting          cubic
_symmetry_space_group_name_Hall '-F 2uv 2vw 3'
_symmetry_space_group_name_H-M  'F d -3'
_symmetry_Int_Tables_number     203

loop_
_atom_site_label
_atom_site_type_symbol
_atom_site_fract_x
_atom_site_fract_y
_atom_site_fract_z
Si1      Si4+  -0.05381   0.12565   0.03508
Al1      Al3+  -0.05524   0.03639   0.12418
O1       O2-   -0.1099    0.0003    0.1056
O2       O2-   -0.0011   -0.0028    0.1416
O3       O2-   -0.0346    0.0758    0.0711
O4       O2-   -0.0693    0.0726    0.18
```

Now the aluminum and silicon are alternating and Löwestein rule is obeyed. For higher Si/Al ratios the 'Al' position is fractionally occupied and a certain percentage might actually be silicon. The procedure here is to first generate the full unit cell of FAU with 96 aluminum (the maximum amount) and judiciously replace aluminum by silicon in the full unit cell.

```
SimulationType               MC
NumberOfCycles               0
NumberOfInitializationCycles  0
```

```
PrintEvery                    10

Forcefield                    Local

Substitute 0 Al1 Si1
Substitute 5 Al1 Si1
Substitute 10 Al1 Si1
Substitute 15 Al1 Si1
Substitute 20 Al1 Si1
RandomlySubstitute 75 Al1 Si1

Framework 0
FrameworkName NaX
UnitCells 1 1 1
ExternalTemperature 300.0
```

It reads the CIF-file which is Nax with 96 aluminum. You can use two types of commands to replace an atom:

- Substitute
  For example. `Substitute 10 Al1 Si1` means replace the 10th `Al1` by `Si1`.

- RandomlySubstitute
  For example, `RandomlySubstitute 75 Al1 Si1` means randomly substitute 75 `Al1` by `Si1`.

When you do them both, first the fixed rules are substituted and next the random ones with the 'left-overs'. The first one 'Substitute' is useful to always have the same structure. You could make a random structure once, look in the output which Al was substituted and use the next time the 'Substitute' command. In this way, you always work with the spacegroup NaX structure (not in P1) which is afterwards change by specifying rules.

More problematic are when several atoms have fractional occupancies lower than unity. Consider IRMOF-8 shown in Fig. 2. The linker molecules are disordered over two possible positions. One of these needs to be selected per linker. First the unit cell is generated from the asymmetric unit cell and subsequently the unit cell needs to be edited. Program which can do just that are Materials Studio, Gaussview, etc. After the cell has been created and edited, the file needs to be placed in 'structures/mofs/cif'. Structures with disorder needs to be created at unit cell level (P1).

Even more difficult is MOF-**1**. Here the cif-file also contains several possibilities, but is not a priori known which ones to choose, i.e. what is the structure of the Dabco unit (1,4-diazabicyclo[2.2.2]octane) within the framework? One possibility is to choose a structure and use a quantum code and minimize the periodic unit cell. The result is shown in Fig. 3.

Note that all these procedures are necessary, but it is still an open question, especially for MOFs, whether you can keep the framework rigid or not. However, it is very hard to calibrate a flexible framework model and for this a substantial amount of reliable experimental data is required.

2.5.3 Format of the framework atoms

The atom-types in CIF-files are constructed from the name of the element and an identifier, e.g. '`C10`' carbon type 10. Usually these carbon atoms are different because they have either different charges or different Van der Waals parameters.

Sometimes a force field is defined to have interactions on an atom-type which depends on its neighbors. For example, the oxygen atom is different whether it is connected to a silicon or to an aluminum atom. Therefore the atom are labelled using

```
ModifyFrameworkAtomConnectedTo O1 Oa1 Al1
ModifyFrameworkAtomConnectedTo O2 Oa2 Al1
ModifyFrameworkAtomConnectedTo O3 Oa3 Al1
ModifyFrameworkAtomConnectedTo O4 Oa4 Al1
```

which modifies '`O1`' to z`Oa1`' when connected to '`Al1`', etc. In the CIF-file you can list the new framework atom with unknown position '`?`'.

```
loop_
_atom_site_label
_atom_site_type_symbol
_atom_site_fract_x
_atom_site_fract_y
_atom_site_fract_z
Si1      Si4+  -0.05381   0.12565   0.03508
Al1      Al3+  -0.05524   0.03639   0.12418
O1       O2-   -0.1099    0.0003    0.1056
O2       O2-   -0.0011   -0.0028    0.1416
O3       O2-   -0.0346    0.0758    0.0711
O4       O2-   -0.0693    0.0726    0.18
Oa1      O2-    ?          ?          ?
Oa2      O2-    ?          ?          ?
Oa3      O2-    ?          ?          ?
Oa4      O2-    ?          ?          ?
```

Alternatively, you can list the atom types '`Oa1`'–'`Oa4`' in your '`pseudo_atoms.def`' file.

Other times a force field is defined as

```
# rules to overwrite
0
# number of defined interactions
4
# type      type2       interaction
O           O           lennard-jones     29.4338257        3.062219744
O           Si          lennard-jones     49.05711264       3.483346249
Si          Si          lennard-jones     81.76308187       3.962387454
CH4_sp3     O           lennard-jones    115.00             3.47
# mixing rules to overwrite
0
```

Here, we have that all oxygens in the framework are of the same type, and all silicon is of the same type. In this case, we would like to map '`O1`', '`O2`', '`O3`', etc. to '`O`', and '`Si1`', '`Si2`', etc. to '`Si`'. You can acgieve this using

```
RemoveAtomNumberCodeFromLabel yes
```

Suppose you want to use MFI with only '`O`' and '`Si`'. MFI is defined using

```
Si1      Si4+   0.42238   0.0565   -0.33598
Si2      Si4+   0.30716   0.02772  -0.1893
\dots
O1       O2-    0.3726    0.0534   -0.2442
O2       O2-    0.3084    0.0587   -0.0789
\dots
```

The force field in '`force_field.def`'

```
# rules to overwrite
0
# number of defined interactions
1
# type      type2       interaction
CH4_sp3     O           lennard-jones    115.00          3.47
# mixing rules to overwrite
0
```

The ‘`pseudo_atom.def`’

```
#number of pseudo atoms
3
#type      print   as  scatt mass       charge  polarization B-factor radii   connectivity anisotropic anisotropic-type   tinker-type
O          yes      O     O  15.9994   -1.025   0.0          1.0      0.5     2            0           absolute           0
Si         yes     Si    Si  28.0855    2.05    0.0          1.0      1.18    4            0           absolute           0
CH4_sp3    yes      C     C  16.04246   0.0     0.0          1.0      1.00    0            0           absolute           0
```

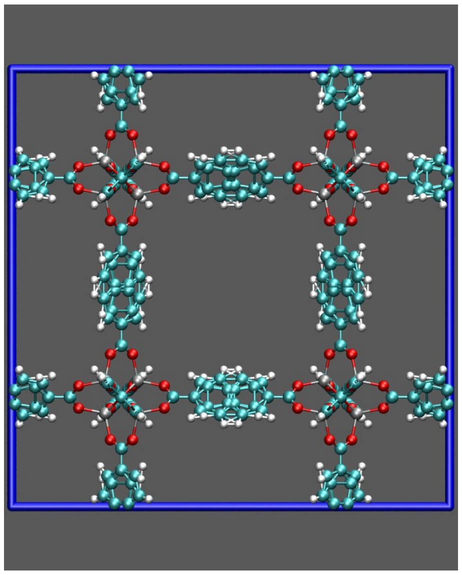
and the output file will show:

```
Pseudo atoms: 2
===========================================================================
Pseudo Atom[   0] Name Si       Oxydation:          Element: Si4+ pdb-name: Si   Scat. Types: 111  14 Mass=28.085498706 B-factor:0.000
                 Charge=2.050     Polarization=0.017    [A^3] (considered a charged atom and no polarization)  Interactions:  no
                 Anisotropic factor:    0.000 [-] (Absolute), Radius:    1.110 [A]
Pseudo Atom[   1] Name O        Oxydation:          Element: O2-  pdb-name: O    Scat. Types: 105   8 Mass=15.999404927 B-factor:0.000
                 Charge=-1.025    Polarization=3.880    [A^3] (considered a charged atom and no polarization)  Interactions:  no
                 Anisotropic factor:    0.000 [-] (Absolute), Radius:    0.660 [A]
```

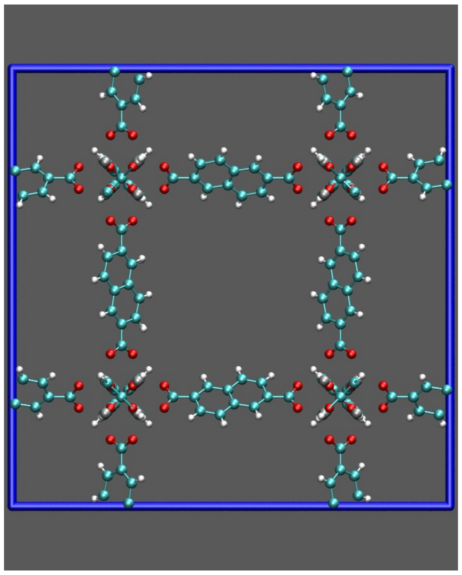
2.5.4 Typing the atoms of the framework

Atoms from a pdb- or cif-file are usually labeled e.g. ‘C’ for a carbon atom. In many force fields different carbon types have different charges. It is necessary to ‘type’ the structure and RASPA contains tools to do this. Let’s assume the original structure always contains elements like ‘H’, ‘C’, ‘N’, ‘O’, etc. and we want to type them ‘Mof_Ha’, ‘Mof_Hb’, etc. A force field type called ‘Typing’ preexists. It only defines the ‘pseudo_atoms.def’ file:

```
#number of pseudo atoms
43
#type      print   as   scat    mass       charge    polarization B-factor radii  connectivity
UNIT       no      H     H      1.0        1.0       0.0          1.0      1.0    0
He         yes     He    He     4.002602   0.0       0.0          1.0      1.0    0
Zn         yes     Zn1   Zn     65.37      0.0       0.0          1.0      1.448  0
Zn1        yes     Zn1   Zn     65.37      0.0       0.0          1.0      1.448  0
Cu         yes     Cu1   Cu     63.546     0.0       0.0          1.0      1.4    0
Cu1        yes     Cu1   Cu     63.546     0.0       0.0          1.0      1.4    0
O          yes     O     O      15.9994    0.0       0.0          1.0      0.68   2
O1         yes     O1    O      15.9994    0.0       0.0          1.0      0.68   2
O2         yes     O2    O      15.9994    0.0       0.0          1.0      0.68   2
O3         yes     O3    O      15.9994    0.0       0.0          1.0      0.68   2
O4         yes     O4    O      15.9994    0.0       0.0          1.0      0.68   2
C          yes     C     C      12.0107    0.0       0.0          1.0      0.720  0
C1         yes     C1    C      12.0107    0.0       0.0          1.0      0.720  0
C2         yes     C2    C      12.0107    0.0       0.0          1.0      0.720  0
C3         yes     C3    C      12.0107    0.0       0.0          1.0      0.720  0
C4         yes     C4    C      12.0107    0.0       0.0          1.0      0.720  0
C5         yes     C5    C      12.0107    0.0       0.0          1.0      0.720  0
C6         yes     C6    C      12.0107    0.0       0.0          1.0      0.720  0
C7         yes     C7    C      12.0107    0.0       0.0          1.0      0.720  0
C8         yes     C8    C      12.0107    0.0       0.0          1.0      0.720  0
C9         yes     C9    C      12.0107    0.0       0.0          1.0      0.720  0
```

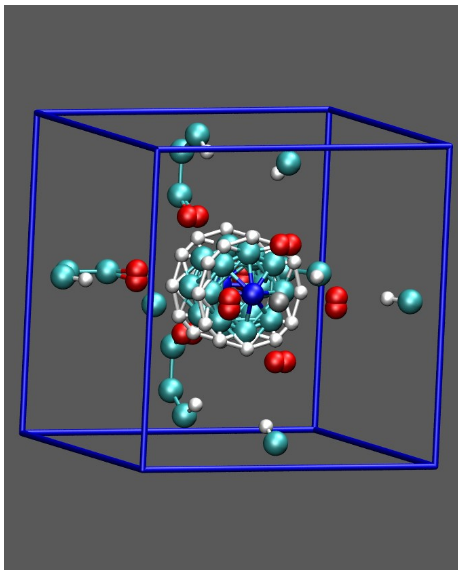


(a) *The IRMOF-8 structure as directly computed from the asymmetric positions and the space group.*

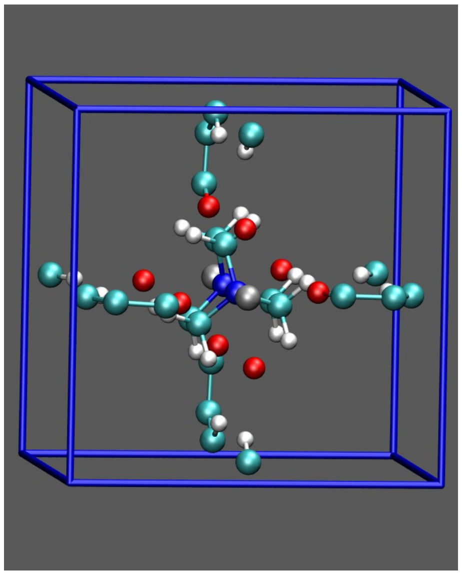


(b) *The IRMOF-8 after making a selection, shown is only one of the possibilities.*

Figure 2: *IRMOF-8 has linkers which are disordered, the linker atoms have a fractional occupancy of 0.5, The atoms however are not individually disorder and there are two disordered linker, one out of two possibilities needs to be selected per linker position.*



(a) *The MOF-***1** *structure from the cif-file.*



(b) *The MOF-***1** *structure edited and optimized with the quantum program dmol (plane wave code).*

Figure 3: *The MOF-1 structure is synthesized as [Zn_2(1,4-bdc)2(Dabco)]. The Dabco (1,4-diazabicyclo[2.2.2]octane) is very disordered with occupancies of 0.38 for the carbon and 0.5 for the hydrogen. The cif-file shown on the left shows all possibilities on top of each other. Here, just choosing one of the possibilities is difficult and it is not obvious which atoms to select. The brute force method is to select one possible choice and use a quantum plane wave for periodic structures and optimize the full unit cell. In this case it is feasible because of the low amount of atoms in the unit cell (only 54 atoms).*

```
C10      yes    C10   C    12.0107    0.0       0.0          1.0      0.720  0
C11      yes    C11   C    12.0107    0.0       0.0          1.0      0.720  0
C12      yes    C12   C    12.0107    0.0       0.0          1.0      0.720  0
C13      yes    C13   C    12.0107    0.0       0.0          1.0      0.720  0
C14      yes    C14   C    12.0107    0.0       0.0          1.0      0.720  0
C15      yes    C15   C    12.0107    0.0       0.0          1.0      0.720  0
C16      yes    C16   C    12.0107    0.0       0.0          1.0      0.720  0
N        yes    N     N    14.00674   0.0       0.0          1.0      0.68   0
N1       yes    N1    N    14.00674   0.0       0.0          1.0      0.68   0
N2       yes    N2    N    14.00674   0.0       0.0          1.0      0.68   0
N3       yes    N3    N    14.00674   0.0       0.0          1.0      0.68   0
N4       yes    N4    N    14.00674   0.0       0.0          1.0      0.68   0
H        yes    H     H    1.00794    0.0       0.0          1.0      0.320  0
H1       yes    H1    H    1.00794    0.0       0.0          1.0      0.320  0
H2       yes    H2    H    1.00794    0.0       0.0          1.0      0.320  0
H3       yes    H3    H    1.00794    0.0       0.0          1.0      0.320  0
H4       yes    H4    H    1.00794    0.0       0.0          1.0      0.320  0
H5       yes    H5    H    1.00794    0.0       0.0          1.0      0.320  0
H6       yes    H6    H    1.00794    0.0       0.0          1.0      0.320  0
H7       yes    H7    H    1.00794    0.0       0.0          1.0      0.320  0
H8       yes    H8    H    1.00794    0.0       0.0          1.0      0.320  0
H9       yes    H9    H    1.00794    0.0       0.0          1.0      0.320  0
```

As an example, let's type the structure 'NU-100' [**?**]. Figure 4 shows the NU-100 cluster with linkers and metal-corners. The pictures shows the different types of atoms and has been used to compute CHelpG charges. In the RASPA input-file you can use the typing command:

```
ModifyFrameworkAtomConnectedTo C Mof_Ca O
```

Look for a 'C' atom, check if it is connect to an 'O' atom and if so, type it 'Mof_Ca'. It is also possible to define two neighbors:

```
ModifyFrameworkAtomConnectedTo C Mof_Cc Mof_Cb Mof_Cb
```

Look for an 'C' atom, if it is connect to a 'Mof_Cb' and to another 'Mof_Cb' atom, then type is 'Mof_Cc'.

The input-file to type 'NU-100' is

```
SimulationType                  MC
NumberOfCycles                  0

Forcefield                      Local

Framework 0
FrameworkName NU-100SP
UnitCells 1 1 1
InputFileType cssr
ExternalTemperature 298.0

ModifyFrameworkAtomConnectedTo C C1 O
ModifyFrameworkAtomConnectedTo C C2 C1
ModifyFrameworkAtomConnectedTo C C3 C2 C2
ModifyFrameworkAtomConnectedTo C C4 C2
ModifyFrameworkAtomConnectedTo C C5 C4
ModifyFrameworkAtomConnectedTo C C6 C5
ModifyFrameworkAtomConnectedTo C C7 C6
ModifyFrameworkAtomConnectedTo C C8 C7
```

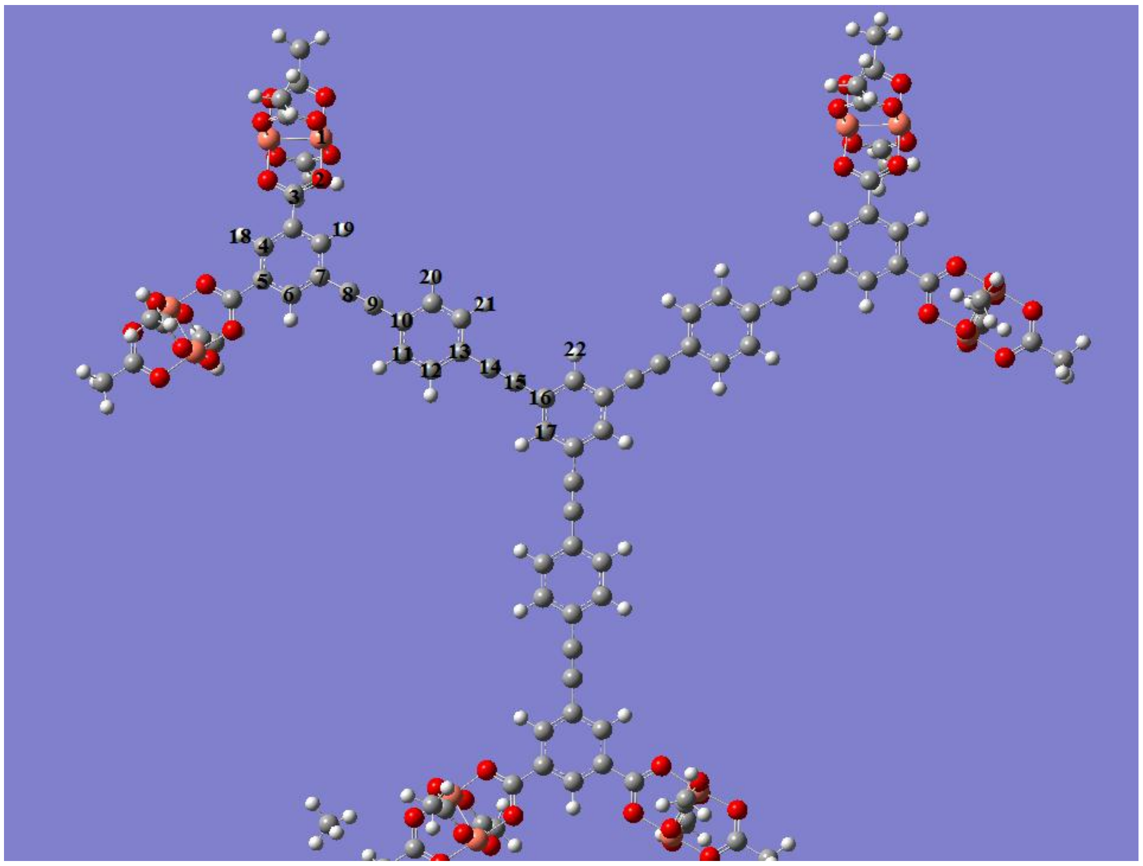


Figure 4: *Cluster used for deriving partial charges on atoms in NU-100SP [*? *].*

```
ModifyFrameworkAtomConnectedTo C C9 C8
ModifyFrameworkAtomConnectedTo C C10 C9
ModifyFrameworkAtomConnectedTo C C11 C10
ModifyFrameworkAtomConnectedTo C C12 C11
ModifyFrameworkAtomConnectedTo C C13 C12
ModifyFrameworkAtomConnectedTo C C14 C13
ModifyFrameworkAtomConnectedTo C C15 C14
ModifyFrameworkAtomConnectedTo H H1 C3
ModifyFrameworkAtomConnectedTo H H2 C4
ModifyFrameworkAtomConnectedTo H H3 C9
ModifyFrameworkAtomConnectedTo H H4 C10
ModifyFrameworkAtomConnectedTo H H5 C15
ModifyFrameworkAtomConnectedTo O O2 C1
ModifyFrameworkAtomConnectedTo Cu Cu O2
```

For MOFs, the easiest start-point to type is the carboxylate group. The carbon connected to the oxygen is typed 'Mof_Ca', the carbon connected to 'Mof_Cb' is typed 'Mof_Cc'. The third line is important: the carbon should only be typed 'Mof_Cc' when it is connected to an 'Mof_Cb' and another 'Mof_Cb'. This must be done like this, otherwise the atom which is above called 'Mof_Cd' would also be wrongly labeled 'Mof_Cc'. After running RASPA, the 'Movie'-directory contains the file 'Framework_intitial.cssr' which is the cssr-file with complete typing. This file can be copied to 'structures/mofs/cssr' and given an appropriate name. Each pseudatom type can now be assigned a different charge in the 'psuedo_atoms.def' file of the 'NU-100' forcefield.

Note that the lines containing the typing-rules are performed top to bottom and in later rules one can use the new names of the previous rules.

2.6 Using CIF-files

2.6.1 Definition of CIF-files

CIF files present crystallographic data in an human readable free format. Let's look at an example:

```
data_FAU_SI

_audit_creation_method RASPA-1.0
_audit_creation_date 2011-2-19
_audit_author_name 'David Dubbeldam'

_citation_author_name        'J.J. Hriljac, M.M. Eddy, A.K. Cheetham, J.A. Donohue,  and G.J. Ray'
_citation_title              'Powder Neutron Diffraction and Si-29 MAS NMR Studies of Siliceous Zeolite-Y'
_citation_journal_abbrev     'J. Solid State Chem.'
_citation_journal_volume     106
_citation_page_first         66
_citation_page_last          72
_citation_year               1993

_cell_length_a    24.2576
_cell_length_b    24.2576
_cell_length_c    24.2576
_cell_angle_alpha 90
_cell_angle_beta  90
_cell_angle_gamma 90
_cell_volume      14273.9

_symmetry_cell_setting          cubic
_symmetry_space_group_name_Hall '-F 4vw 2vw 3'
_symmetry_space_group_name_H-M  'F d -3 m'
_symmetry_Int_Tables_number     227

loop_
_symmetry_equiv_pos_as_xyz
 'x,y,z'
 '-x+3/4,-y+1/4,z+1/2'
  ...............
  ...............
  ...............
 'z,-y+3/4,-x+3/4'
 'z+1/2,y+1/2,x'

loop_
_atom_site_label
_atom_site_type_symbol
_atom_site_fract_x
_atom_site_fract_y
_atom_site_fract_z
_atom_site_charge
_atom_site_polarization
Si1      Si4+  -0.05392   0.1253    0.03589   2.05    0
O1       O2-    0        -0.10623   0.10623  -1.025   0
O2       O2-   -0.00323  -0.00323   0.14066  -1.025   0
O3       O2-    0.0757    0.0757   -0.03577  -1.025   0
O4       O2-    0.07063   0.07063   0.32115  -1.025   0
```

The 'data_' string signal the start of a data block. Each data block corresponds to a different structures, and typically only one structure is present (although it possible to combine more than one structure in a single file). The CIF instructions are divided into *data name categories*, such as '_atom_site_' to describe atomic site parameters, '_cell_' to describe the cell parameters, '_symmetry_' to specify space group symmetry, etc. CIF data names begin with an underscore. For some data name the data can be provided using a list of data items. Such data items are preceded by a 'loop_' string. The '_atom_site_' section is typical example. The order of the data items correspond to the order in which the actual data is provided.

A nice feature of CIFs is that one can easily extend the syntax to include nom-standard data item. For example. in the '_atom_site_' section, the data items '_atom_site_label', '_atom_site_type_symbol', '_atom_site_fract_x', '_atom_site_fract_y', '_atom_site_charge' belong to the official CIF specification, but '_atom_site_charge', '_atom_site_polarizatio '_atom_site_anisotropic_displacement', '_atom_site_anisotropic_type', and '_atom_site_print_to_pdb' have been added in RASPA CIFs. Used in this fashion, they provide a replacement for the 'pseudo_atoms.def' file. Note that if a 'pseudo_atoms.def' file is used, the value in that file will have preference over the CIF-file values (if they both define the same atom-type).

2.6.2 What charge definition is used? 'pseudo_atom.def' or from the CIF-file?

For adsorbates the charges are defined via the atom-type in the 'pseudo_atom.def' file. For the framework, there are several scenarios:

- define charges via the CIF-file
  If you want a possibly different charge for each atom, then use the option:

  ```
  UseChargesFromCIFFile yes
  ```

  and define the charge using the field '_atom_site_charge' in the CIF-file. Atom-types from the CIF-file that are not defined in the 'pseudo_atom.def' are automatically added, atoms that are already defined as a type in the 'pseudo_atom.def' get the charge from the CIF-file. In the output-file in the list of pseudo-atoms you will see e.g.

  ```
  Charge=0.111115012    (av)
  ```

  which signals that for this atom-type the averages charge is listed (because each atom potentially can have a different value in this case). This is a typical case for simulations based on CHelpG charges from quantum.

- define charges via the 'pseudo_atom.def' file
  If you want the same charge for all atoms of the atom-type, then you can list all of these in the 'pseudo_atom.def' file and use

  ```
  UseChargesFromCIFFile no
  ```

  which is the default. Any atoms with a type known in the CIF-file will get a charge given in the 'pseudo_atom.def' file; atoms of unknown type will be added to the pseudo-atoms but with a charge of zero. The latter is probably not what you want, so make sure you have listed all atom type in 'pseudo_atom.def' file.

- Define charges using 'Charge Equilibration'
  No matter what you define in the 'pseudo_atom.def' or CIF-file, the charges will be recompute using the charge-equilbration scheme of Wilmer and Snurr.

Tip: the charges that are actually used in the simulation are listed as the column '_atom_site_charge' in the file 'Movies/System_0/Framework_0_initial_P1.cif'. Also, check the ouput-file for the net-charge of the framework, and the smallest and largest charge it found, e.g.

```
Framework has net charge: 0.000000
largest charge : 0.931455
```

```
            smallest charge: -0.626799
```

2.6.3 How to choose atom-types?

The FAU structure above was defined with atom types: 'Si1', 'O1', 'O2', 'O3', and 'O4'. Using the option:

```
RemoveAtomNumberCodeFromLabel yes
```

these 5 types will be reduces to 2: 'Si' and 'O'. There are advantages and disadvantages to each of the options:

- Specific types
  Use if

  1. You want RDF between the adsorbate atoms and the specific framework atoms.
  2. If you have different VDW parameters for each specific framework atom (so 'O1', 'O2', 'O3', 'O4' would have different VDW parameters).

- Reduced types
  Use if you are not interested in the difference between 'O1', ..., 'O4', but only have a single VDW parameter set for that atom type 'O'. Note: you can still list different charges for each of these atoms in the CIF-file. This options avoid excessive number of pseudo-atoms, which can clutter the output, and avoids having lots of different RDFs (and manually having to averages these afterwards).

Appendix: space group information

triclinic									
id	Int. Nr.	long Hermann-Mauguin name	Hall name	cell choice	centered	#	Chiral	Centric	Enantio-morphic
1	1	P 1	P 1	cell choice 1	primitive	1	yes	no	no
2	2	P -1	-P 1	cell choice 1	primitive	2	yes	no	no

Table 2.1: *Triclinic spacegroup information.*

monoclinic									
id	Int. Nr.	long Hermann-Mauguin name	Hall name	cell choice	centered	#	Chiral	Centric	Enantio-morphic
3	3	P 1 2 1	P 2y	unique axis b	primitive	2	no	yes	no
4	3	P 1 1 2	P 2	unique axis c	primitive	2	no	yes	no
5	3	P 2 1 1	P 2x	unique axis a	primitive	2	no	yes	no
6	4	P 1 21 1	P 2yb	unique axis b	primitive	2	no	yes	no
7	4	P 1 1 21	P 2c	unique axis c	primitive	2	no	yes	no
8	4	P 21 1 1	P 2xa	unique axis a	primitive	2	no	yes	no
9	5	C 1 2 1	C 2y	b, cell choice 1	c	4	no	yes	no
10	5	A 1 2 1	A 2y	b, cell choice 2	a	4	no	yes	no
11	5	I 1 2 1	I 2y	b, cell choice 3	body	4	no	yes	no
12	5	A 1 1 2	A 2	c, cell choice 1	a	4	no	yes	no
13	5	B 1 1 2	B 2	c, cell choice 2	b	4	no	yes	no
14	5	I 1 1 2	I 2	c, cell choice 3	body	4	no	yes	no
15	5	B 2 1 1	B 2x	a, cell choice 1	b	4	no	yes	no
16	5	C 2 1 1	C 2x	a, cell choice 2	c	4	no	yes	no
17	5	I 2 1 1	I 2x	a, cell choice 3	body	4	no	yes	no
18	6	P 1 m 1	P -2y	unique axis b	primitive	2	no	no	no
19	6	P 1 1 m	P -2	unique axis c	primitive	2	no	no	no
20	6	P m 1 1	P -2x	unique axis a	primitive	2	no	no	no
21	7	P 1 c 1	P -2yc	b, cell choice 1	primitive	2	no	no	no
22	7	P 1 n 1	P -2yac	b, cell choice 2	primitive	2	no	no	no
23	7	P 1 a 1	P -2ya	b, cell choice 3	primitive	2	no	no	no
24	7	P 1 1 a	P -2a	c, cell choice 1	primitive	2	no	no	no
25	7	P 1 1 n	P -2ab	c, cell choice 2	primitive	2	no	no	no
26	7	P 1 1 b	P -2b	c, cell choice 3	primitive	2	no	no	no
27	7	P b 1 1	P -2xb	a, cell choice 1	primitive	2	no	no	no
28	7	P n 1 1	P -2xbc	a, cell choice 2	primitive	2	no	no	no
29	7	P c 1 1	P -2xc	a, cell choice 3	primitive	2	no	no	no
30	8	C 1 m 1	C -2y	b, cell choice 1	c	4	no	no	no
31	8	A 1 m 1	A -2y	b, cell choice 2	a	4	no	no	no
32	8	I 1 m 1	I -2y	b, cell choice 3	body	4	no	no	no
33	8	A 1 1 m	A -2	c, cell choice 1	a	4	no	no	no
34	8	B 1 1 m	B -2	c, cell choice 2	b	4	no	no	no
35	8	I 1 1 m	I -2	c, cell choice 3	body	4	no	no	no
36	8	B m 1 1	B -2x	a, cell choice 1	b	4	no	no	no
37	8	B m 1 1	C -2x	a, cell choice 2	c	4	no	no	no
38	8	I m 1 1	I -2x	a, cell choice 3	body	4	no	no	no
39	9	C 1 c 1	C -2yc	b, cell choice 1	c	4	no	no	no
40	9	A 1 n 1	A -2yab	b, cell choice 2	a	4	no	no	no
41	9	I 1 a 1	I -2ya	b, cell choice 3	body	4	no	no	no
42	9	A 1 a 1	A -2ya	-b, cell choice 1	a	4	no	no	no
43	9	C 1 n 1	C -2yac	-b, cell choice 2	c	4	no	no	no
44	9	I 1 c 1	I -2yc	-b, cell choice 3	body	4	no	no	no

45	9	A 1 1 a	A -2a	c, cell choice 1	a	4	no	no	no
46	9	B 1 1 n	B -2ab	c, cell choice 2	b	4	no	no	no
47	9	I 1 1 b	I -2b	c, cell choice 3	body	4	no	no	no
48	9	B 1 1 b	B -2b	-c, cell choice 1	b	4	no	no	no
49	9	A 1 1 n	A -2ab	-c, cell choice 2	a	4	no	no	no
50	9	I 1 1 a	I -2a	-c, cell choice 3	body	4	no	no	no
51	9	B b 1 1	B -2xb	a, cell choice 1	b	4	no	no	no
52	9	C n 1 1	C -2xac	a, cell choice 2	c	4	no	no	no
53	9	I c 1 1	I -2xc	a, cell choice 3	body	4	no	no	no
54	9	C c 1 1	C -2xc	-a, cell choice 1	c	4	no	no	no
55	9	B n 1 1	B -2xab	-a, cell choice 2	b	4	no	no	no
56	9	I b 1 1	I -2xb	-a, cell choice 3	body	4	no	no	no
57	10	P 1 2/m 1	-P 2y	unique axis b	primitive	4	yes	no	no
58	10	P 1 1 2/m	-P 2	unique axis c	primitive	4	yes	no	no
59	10	P 2/m 1 1	-P 2x	unique axis a	primitive	4	yes	no	no
60	11	P 1 21/m 1	-P 2yb	unique axis b	primitive	4	yes	no	no
61	11	P 1 1 21/m	-P 2c	unique axis c	primitive	4	yes	no	no
62	11	P 21/m 1 1	-P 2xa	unique axis a	primitive	4	yes	no	no
63	12	C 1 2/m 1	-C 2y	b, cell choice 1	c	8	yes	no	no
64	12	A 1 2/m 1	-A 2y	b, cell choice 2	a	8	yes	no	no
65	12	I 1 2/m 1	-I 2y	b, cell choice 3	body	8	yes	no	no
66	12	A 1 1 2/m	-A 2	c, cell choice 1	a	8	yes	no	no
67	12	B 1 1 2/m	-B 2	c, cell choice 2	b	8	yes	no	no
68	12	I 1 1 2/m	-I 2	c, cell choice 3	body	8	yes	no	no
69	12	B 2/m 1 1	-B 2x	a, cell choice 1	b	8	yes	no	no
70	12	C 2/m 1 1	-C 2x	a, cell choice 2	c	8	yes	no	no
71	12	I 2/m 1 1	-I 2x	a, cell choice 3	body	8	yes	no	no
72	13	P 1 2/c 1	-P 2yc	b, cell choice 1	primitive	4	yes	no	no
73	13	P 1 2/n 1	-P 2yac	b, cell choice 2	primitive	4	yes	no	no
74	13	P 1 2/a 1	-P 2ya	b, cell choice 3	primitive	4	yes	no	no
75	13	P 1 1 2/a	-P 2a	c, cell choice 1	primitive	4	yes	no	no
76	13	P 1 1 2/n	-P 2ab	c, cell choice 2	primitive	4	yes	no	no
77	13	P 1 1 2/b	-P 2b	c, cell choice 3	primitive	4	yes	no	no
78	13	P 2/b 1 1	-P 2xb	a, cell choice 1	primitive	4	yes	no	no
79	13	P 2/n 1 1	-P 2xbc	a, cell choice 2	primitive	4	yes	no	no
80	13	P 2/c 1 1	-P 2xc	a, cell choice 3	primitive	4	yes	no	no
81	14	P 1 21/c 1	-P 2ybc	b, cell choice 1	primitive	4	yes	no	no
82	14	P 1 21/n 1	-P 2yn	b, cell choice 2	primitive	4	yes	no	no
83	14	P 1 21/a 1	-P 2yab	b, cell choice 3	primitive	4	yes	no	no
84	14	P 1 1 21/a	-P 2ac	c, cell choice 1	primitive	4	yes	no	no
85	14	P 1 1 21/n	-P 2n	c, cell choice 2	primitive	4	yes	no	no
86	14	P 1 1 21/b	-P 2bc	c, cell choice 3	primitive	4	yes	no	no
87	14	P 21/b 1 1	-P 2xab	a, cell choice 1	primitive	4	yes	no	no
88	14	P 21/n 1 1	-P 2xn	a, cell choice 2	primitive	4	yes	no	no
89	14	P 21/c 1 1	-P 2xac	a, cell choice 3	primitive	4	yes	no	no
90	15	C 1 2/c 1	-C 2yc	b, cell choice 1	c	8	yes	no	no
91	15	A 1 2/n 1	-A 2yab	b, cell choice 2	a	8	yes	no	no
92	15	I 1 2/a 1	-I 2ya	b, cell choice 3	body	8	yes	no	no
93	15	A 1 2/a 1	-A 2ya	-b, cell choice 1	a	8	yes	no	no
94	15	C 1 2/n 1	-C 2yac	-b, cell choice 2	c	8	yes	no	no
95	15	I 1 2/c 1	-I 2yc	-b, cell choice 3	body	8	yes	no	no
96	15	A 1 1 2/a	-A 2a	c, cell choice 1	a	8	yes	no	no
97	15	B 1 1 2/n	-B 2ab	c, cell choice 2	b	8	yes	no	no
98	15	I 1 1 2/b	-I 2b	c, cell choice 3	body	8	yes	no	no
99	15	B 1 1 2/b	-B 2b	-c, cell choice 1	b	8	yes	no	no
100	15	A 1 1 2/n	-A 2ab	-c, cell choice 2	a	8	yes	no	no

101	15	I 1 1 2/a	-I 2a	-c, cell choice 3	body	8	yes	no	no
102	15	B 2/b 1 1	-B 2xb	a, cell choice 1	b	8	yes	no	no
103	15	C 2/n 1 1	-C 2xac	a, cell choice 2	c	8	yes	no	no
104	15	I 2/c 1 1	-I 2xc	a, cell choice 3	body	8	yes	no	no
105	15	C 2/c 1 1	-C 2xc	-a, cell choice 1	c	8	yes	no	no
106	15	B 2/n 1 1	-B 2xab	-a, cell choice 2	b	8	yes	no	no
107	15	I 2/b 1 1	-I 2xb	-a, cell choice 3	body	8	yes	no	no

Table 2.2: *Monoclinic spacegroup information.*

orthorhombic									
id	Int. Nr.	long Hermann-Mauguin name	Hall name	cell choice	centered	#	Chiral	Centric	Enantio-morphic
108	16	P 2 2 2	P 2 2	cell choice 1	primitive	4	yes	yes	no
109	17	P 2 2 21	P 2c 2	abc	primitive	4	yes	yes	no
110	17	P 21 2 2	P 2a 2a	cab	primitive	4	yes	yes	no
111	17	P 2 21 2	P 2 2b	bca	primitive	4	yes	yes	no
112	18	P 21 21 2	P 2 2ab	abc	primitive	4	yes	yes	no
113	18	P 2 21 21	P 2bc 2	cab	primitive	4	yes	yes	no
114	18	P 21 2 21	P 2ac 2ac	bca	primitive	4	yes	yes	no
115	19	P 21 21 21	P 2ac 2ab	cell choice 1	primitive	4	yes	yes	no
116	20	C 2 2 21	C 2c 2	abc	c	8	yes	yes	no
117	20	A 21 2 2	A 2a 2a	cab	a	8	yes	yes	no
118	20	B 2 21 2	B 2 2b	bca	b	8	yes	yes	no
119	21	C 2 2 2	C 2 2	abc	c	8	no	yes	no
120	21	A 2 2 2	A 2 2	cab	a	8	no	yes	no
121	21	B 2 2 2	B 2 2	bca	b	8	no	yes	no
122	22	F 2 2 2	F 2 2	cell choice 1	face	16	no	yes	no
123	23	I 2 2 2	I 2 2	cell choice 1	body	8	no	yes	no
124	24	I 21 21 21	I 2b 2c	cell choice 1	body	8	no	yes	no
125	25	P m m 2	P 2 -2	abc	primitive	4	no	no	no
126	25	P 2 m m	P -2 2	cab	primitive	4	no	no	no
127	25	P m 2 m	P -2 -2	bca	primitive	4	no	no	no
128	26	P m c 21	P 2c -2	abc	primitive	4	no	no	no
129	26	P c m 21	P 2c -2c	ba-c	primitive	4	no	no	no
130	26	P 21 m a	P -2a 2a	cab	primitive	4	no	no	no
131	26	P 21 a m	P -2 2a	-cba	primitive	4	no	no	no
132	26	P b 21 m	P -2 -2b	bca	primitive	4	no	no	no
133	26	P m 21 b	P -2b -2	a-cb	primitive	4	no	no	no
134	27	P c c 2	P 2 -2c	abc	primitive	4	no	no	no
135	27	P 2 a a	P -2a 2	cab	primitive	4	no	no	no
136	27	P b 2 b	P -2b -2b	bca	primitive	4	no	no	no
137	28	P m a 2	P 2 -2a	abc	primitive	4	no	no	no
138	28	P b m 2	P 2 -2b	ba-c	primitive	4	no	no	no
139	28	P 2 m b	P -2b 2	cab	primitive	4	no	no	no
140	28	P 2 c m	P -2c 2	-cba	primitive	4	no	no	no
141	28	P c 2 m	P -2c -2c	bca	primitive	4	no	no	no
142	28	P m 2 a	P -2a -2a	a-cb	primitive	4	no	no	no
143	29	P c a 21	P 2c -2ac	abc	primitive	4	no	no	no
144	29	P b c 21	P 2c -2b	ba-c	primitive	4	no	no	no
145	29	P 21 a b	P -2b 2a	cab	primitive	4	no	no	no
146	29	P 21 c a	P -2ac 2a	-cba	primitive	4	no	no	no
147	29	P c 21 b	P -2bc -2c	bca	primitive	4	no	no	no
148	29	P b 21 a	P -2a -2ab	a-cb	primitive	4	no	no	no
149	30	P n c 2	P 2 -2bc	abc	primitive	4	no	no	no
150	30	P c n 2	P 2 -2ac	ba-c	primitive	4	no	no	no

151	30	P 2 n a	P -2ac 2	cab	primitive	4	no	no	no
152	30	P 2 a n	P -2ab 2	-cba	primitive	4	no	no	no
153	30	P b 2 n	P -2ab -2ab	bca	primitive	4	no	no	no
154	30	P n 2 b	P -2bc -2bc	a-cb	primitive	4	no	no	no
155	31	P m n 21	P 2ac -2	abc	primitive	4	no	no	no
156	31	P n m 21	P 2bc -2bc	ba-c	primitive	4	no	no	no
157	31	P 21 m n	P -2ab 2ab	cab	primitive	4	no	no	no
158	31	P 21 n m	P -2 2ac	-cba	primitive	4	no	no	no
159	31	P n 21 m	P -2 -2bc	bca	primitive	4	no	no	no
160	31	P m 21 n	P -2ab -2	a-cb	primitive	4	no	no	no
161	32	P b a 2	P 2 -2ab	abc	primitive	4	no	no	no
162	32	P 2 c b	P -2bc 2	cab	primitive	4	no	no	no
163	32	P c 2 a	P -2ac -2ac	bca	primitive	4	no	no	no
164	33	P n a 21	P 2c -2n	abc	primitive	4	no	no	no
165	33	P b n 21	P 2c -2ab	ba-c	primitive	4	no	no	no
166	33	P 21 n b	P -2bc 2a	cab	primitive	4	no	no	no
167	33	P 21 c n	P -2n 2a	-cba	primitive	4	no	no	no
168	33	P c 21 n	P -2n -2ac	bca	primitive	4	no	no	no
169	33	P n 21 a	P -2ac -2n	a-cb	primitive	4	no	no	no
170	34	P n n 2	P 2 -2n	abc	primitive	4	no	no	no
171	34	P 2 n n	P -2n 2	cab	primitive	4	no	no	no
172	34	P n 2 n	P -2n -2n	bca	primitive	4	no	no	no
173	35	C m m 2	C 2 -2	abc	c	8	no	no	no
174	35	A 2 m m	A -2 2	cab	a	8	no	no	no
175	35	B m 2 m	B -2 -2	bca	b	8	no	no	no
176	36	C m c 21	C 2c -2	abc	c	8	no	no	no
177	36	C c m 21	C 2c -2c	ba-c	c	8	no	no	no
178	36	A 21 m a	A -2a 2a	cab	a	8	no	no	no
179	36	A 21 a m	A -2 2a	-cba	a	8	no	no	no
180	36	B b 21 m	B -2 -2b	bca	b	8	no	no	no
181	36	B m 21 b	B -2b -2	a-cb	b	8	no	no	no
182	37	C c c 2	C 2 -2c	abc	c	8	no	no	no
183	37	A 2 a a	A -2a 2	cab	a	8	no	no	no
184	37	B b 2 b	B -2b -2b	bca	b	8	no	no	no
185	38	A m m 2	A 2 -2	abc	a	8	no	no	no
186	38	B m m 2	B 2 -2	ba-c	b	8	no	no	no
187	38	B 2 m m	B -2 2	cab	b	8	no	no	no
188	38	C 2 m m	C -2 2	-cba	c	8	no	no	no
189	38	C m 2 m	C -2 -2	bca	c	8	no	no	no
190	38	A m 2 m	A -2 -2	a-cb	a	8	no	no	no
191	39	A b m 2	A 2 -2b	abc	a	8	no	no	no
192	39	B m a 2	B 2 -2a	ba-c	b	8	no	no	no
193	39	B 2 c m	B -2a 2	cab	b	8	no	no	no
194	39	C 2 m b	C -2a 2	-cba	c	8	no	no	no
195	39	C m 2 a	C -2a -2a	bca	c	8	no	no	no
196	39	A c 2 m	A -2b -2b	a-cb	a	8	no	no	no
197	40	A m a 2	A 2 -2a	abc	a	8	no	no	no
198	40	B b m 2	B 2 -2b	ba-c	b	8	no	no	no
199	40	B 2 m b	B -2b 2	cab	b	8	no	no	no
200	40	C 2 c m	C -2c 2	-cba	c	8	no	no	no
201	40	C c 2 m	C -2c -2c	bca	c	8	no	no	no
202	40	A m 2 a	A -2a -2a	a-cb	a	8	no	no	no
203	41	A b a 2	A 2 -2ab	abc	a	8	no	no	no
204	41	B b a 2	B 2 -2ab	ba-c	b	8	no	no	no
205	41	B 2 c b	B -2ab 2	cab	b	8	no	no	no
206	41	C 2 c b	C -2ac 2	-cba	c	8	no	no	no

207	41	C c 2 a	C -2ac -2ac	bca	c	8	no	no	no
208	41	A c 2 a	A -2ab -2ab	a-cb	a	8	no	no	no
209	42	F m m 2	F 2 -2	abc	face	16	no	no	no
210	42	F 2 m m	F -2 2	cab	face	16	no	no	no
211	42	F m 2 m	F -2 -2	bca	face	16	no	no	no
212	43	F d d 2	F 2 -2d	abc	face	16	no	no	no
213	43	F 2 d d	F -2d 2	cab	face	16	no	no	no
214	43	F d 2 d	F -2d -2d	bca	face	16	no	no	no
215	44	I m m 2	I 2 -2	abc	body	8	no	no	no
216	44	I 2 m m	I -2 2	cab	body	8	no	no	no
217	44	I m 2 m	I -2 -2	bca	body	8	no	no	no
218	45	I b a 2	I 2 -2c	abc	body	8	no	no	no
219	45	I 2 c b	I -2a 2	cab	body	8	no	no	no
220	45	I c 2 a	I -2b -2b	bca	body	8	no	no	no
221	46	I m a 2	I 2 -2a	abc	body	8	no	no	no
222	46	I b m 2	I 2 -2b	ba-c	body	8	no	no	no
223	46	I 2 m b	I -2b 2	cab	body	8	no	no	no
224	46	I 2 c m	I -2c 2	-cba	body	8	no	no	no
225	46	I c 2 m	I -2c -2c	bca	body	8	no	no	no
226	46	I m 2 a	I -2a -2a	a-cb	body	8	no	no	no
227	47	P 2/m 2/m 2/m	-P 2 2	cell choice 1	primitive	8	yes	no	no
228	48	P 2/n 2/n 2/n:1	P 2 2 -1n	cell choice 1	primitive	8	yes	no	no
229	48	P 2/n 2/n 2/n:2	-P 2ab 2bc	cell choice 2	primitive	8	yes	no	no
230	49	P 2/c 2/c 2/m	-P 2 2c	abc	primitive	8	yes	no	no
231	49	P 2/m 2/a 2/a	-P 2a 2	cab	primitive	8	yes	no	no
232	49	P 2/b 2/m 2/b	-P 2b 2b	bca	primitive	8	yes	no	no
233	50	P 2/b 2/a 2/n:1	P 2 2 -1ab	cell choice 1	primitive	8	yes	no	no
234	50	P 2/b 2/a 2/n:2	-P 2ab 2b	cell choice 2	primitive	8	yes	no	no
235	50	P 2/n 2/c 2/b:1	P 2 2 -1bc	cab	primitive	8	yes	no	no
236	50	P 2/n 2/c 2/b:2	-P 2b 2bc	cab, cell choice 2	primitive	8	yes	no	no
237	50	P 2/c 2/n 2/a:1	P 2 2 -1ac	bca	primitive	8	yes	no	no
238	50	P 2/c 2/n 2/a:2	-P 2a 2c	bca, cell choice 2	primitive	8	yes	no	no
239	51	P 21/m 2/m 2/a	-P 2a 2a	abc	primitive	8	yes	no	no
240	51	P 2/m 21/m 2/b	-P 2b 2	ba-c	primitive	8	yes	no	no
241	51	P 2/b 21/m 2/m	-P 2 2b	cab	primitive	8	yes	no	no
242	51	P 2/c 2/m 21/m	-P 2c 2c	-cba	primitive	8	yes	no	no
243	51	P 2/m 2/c 21/m	-P 2c 2	bca	primitive	8	yes	no	no
244	51	P 21/m 2/a 2/m	-P 2 2a	a-cb	primitive	8	yes	no	no
245	52	P 2/n 21/n 2/a	-P 2a 2bc	abc	primitive	8	yes	no	no
246	52	P 21/n 2/n 2/b	-P 2b 2n	ba-c	primitive	8	yes	no	no
247	52	P 2/b 2/n 21/n	-P 2n 2b	cab	primitive	8	yes	no	no
248	52	P 2/c 21/n 2/n	-P 2ab 2c	-cba	primitive	8	yes	no	no
249	52	P 21/n 2/c 2/n	-P 2ab 2n	bca	primitive	8	yes	no	no
250	52	P 2/n 2/a 21/n	-P 2n 2bc	a-cb	primitive	8	yes	no	no
251	53	P 2/m 2/n 21/a	-P 2ac 2	abc	primitive	8	yes	no	no
252	53	P 2/n 2/m 21/b	-P 2bc 2bc	ba-c	primitive	8	yes	no	no
253	53	P 21/b 2/m 2/n	-P 2ab 2ab	cab	primitive	8	yes	no	no
254	53	P 21/c 2/n 2/m	-P 2 2ac	-cba	primitive	8	yes	no	no
255	53	P 2/n 21/c 2/m	-P 2 2bc	bca	primitive	8	yes	no	no
256	53	P 2/m 21/a 2/n	-P 2ab 2	a-cb	primitive	8	yes	no	no
257	54	P 21/c 2/c 2/a	-P 2a 2ac	abc	primitive	8	yes	no	no
258	54	P 2/c 21/c 2/b	-P 2b 2c	ba-c	primitive	8	yes	no	no
259	54	P 2/b 21/a 2/a	-P 2a 2b	cab	primitive	8	yes	no	no
260	54	P 2/c 2/a 21/a	-P 2ac 2c	-cba	primitive	8	yes	no	no
261	54	P 2/b 2/c 21/b	-P 2bc 2b	bca	primitive	8	yes	no	no
262	54	P 21/b 2/a 2/b	-P 2b 2ab	a-cb	primitive	8	yes	no	no

263	55	P 21/b 21/a 2/m	-P 2 2ab	abc	primitive	8	yes	no	no
264	55	P 2/m 21/c 21/b	-P 2bc 2	cab	primitive	8	yes	no	no
265	55	P 21/c 2/m 21/a	-P 2ac 2ac	bca	primitive	8	yes	no	no
266	56	P 21/c 21/c 2/n	-P 2ab 2ac	abc	primitive	8	yes	no	no
267	56	P 2/n 21/a 21/a	-P 2ac 2bc	cab	primitive	8	yes	no	no
268	56	P 21/b 2/n 21/b	-P 2bc 2ab	bca	primitive	8	yes	no	no
269	57	P 2/b 21/c 21/m	-P 2c 2b	abc	primitive	8	yes	no	no
270	57	P 21/c 2/a 21/m	-P 2c 2ac	ba-c	primitive	8	yes	no	no
271	57	P 21/m 2/c 21/a	-P 2ac 2a	cab	primitive	8	yes	no	no
272	57	P 21/m 21/a 2/b	-P 2b 2a	-cba	primitive	8	yes	no	no
273	57	P 21/b 21/m 2/a	-P 2a 2ab	bca	primitive	8	yes	no	no
274	57	P 2/c 21/m 21/b	-P 2bc 2c	a-cb	primitive	8	yes	no	no
275	58	P 21/n 21n 2/m	-P 2 2n	abc	primitive	8	yes	no	no
276	58	P 2/m 21/n 21/n	-P 2n 2	cab	primitive	8	yes	no	no
277	58	P 21/n 2/m 21/n	-P 2n 2n	bca	primitive	8	yes	no	no
278	59	P 21/m 21/m 2/n:1	P 2 2ab -1ab	cell choice 1	primitive	8	yes	no	no
279	59	P 21/m 21/m 2/n:2	-P 2ab 2a	cell choice 2	primitive	8	yes	no	no
280	59	P 2/n 21/m 21/m:1	P 2bc 2 -1bc	cab	primitive	8	yes	no	no
281	59	P 2/n 21/m 21/m:2	-P 2c 2bc	cab, cell choice 2	primitive	8	yes	no	no
282	59	P 21/m 2/n 21/m:1	P 2ac 2ac -1ac	bca	primitive	8	yes	no	no
283	59	P 21/m 2/n 21/m:2	-P 2c 2a	bca, cell choice 2	primitive	8	yes	no	no
284	60	P 21/b 2/c 21/n	-P 2n 2ab	abc	primitive	8	yes	no	no
285	60	P 2/c 21/a 21/n	-P 2n 2c	ba-c	primitive	8	yes	no	no
286	60	P 21/n 21/a 2/b	-P 2a 2n	cab	primitive	8	yes	no	no
287	60	P 21/n 2/a 21/b	-P 2bc 2n	-cba	primitive	8	yes	no	no
288	60	P 2/b 21/n 21/a	-P 2ac 2b	bca	primitive	8	yes	no	no
289	60	P 21/c 21/n 2/b	-P 2b 2ac	a-cb	primitive	8	yes	no	no
290	61	P 21/b 21/c 21/a	-P 2ac 2ab	abc	primitive	8	yes	no	no
291	61	P 21/c 21/a 21/b	-P 2bc 2ac	ba-c	primitive	8	yes	no	no
292	62	P 21/n 21/m 21/a	-P 2ac 2n	abc	primitive	8	yes	no	no
293	62	P 21/m 21/n 21/b	-P 2bc 2a	ba-c	primitive	8	yes	no	no
294	62	P 21/b 21/n 21/m	-P 2c 2ab	cab	primitive	8	yes	no	no
295	62	P 21/c 21/m 21/n	-P 2n 2ac	-cba	primitive	8	yes	no	no
296	62	P 21/m 21/c 21/n	-P 2n 2a	bca	primitive	8	yes	no	no
297	62	P 21/n 21/a 21/m	-P 2c 2n	a-cb	primitive	8	yes	no	no
298	63	C 2/m 2/c 21/m	-C 2c 2	abc	c	16	yes	no	no
299	63	C 2/c 2/m 21/m	-C 2c 2c	ba-c	c	16	yes	no	no
300	63	A 21/m 2/m 2/a	-A 2a 2a	cab	a	16	yes	no	no
301	63	A 21/m 2/a 2/m	-A 2 2a	-cba	a	16	yes	no	no
302	63	B 2/b 21/m 2/m	-B 2 2b	bca	b	16	yes	no	no
303	63	B 2/m 21/m 2/b	-B 2b 2	a-cb	b	16	yes	no	no
304	64	C 2/m 2/c 21/a	-C 2ac 2	abc	c	16	yes	no	no
305	64	C 2/c 2/m 21/b	-C 2ac 2ac	ba-c	c	16	yes	no	no
306	64	A 21/b 2/m 2/a	-A 2ab 2ab	cab	a	16	yes	no	no
307	64	A 21/c 2/a 2/m	-A 2 2ab	-cba	a	16	yes	no	no
308	64	B 2/b 21/c 2/m	-B 2 2ab	bca	b	16	yes	no	no
309	64	B 2/m 21/a 2/b	-B 2ab 2	a-cb	b	16	yes	no	no
310	65	C 2/m 2/m 2/m	-C 2 2	abc	c	16	yes	no	no
311	65	A 2/m 2/m 2/m	-A 2 2	cab	a	16	yes	no	no
312	65	B 2/m 2/m 2/m	-B 2 2	bca	b	16	yes	no	no
313	66	C 2/c 2/c 2/m	-C 2 2c	abc	c	16	yes	no	no
314	66	A 2/m 2/a 2/a	-A 2a 2	cab	a	16	yes	no	no
315	66	B 2/b 2/m 2/b	-B 2b 2b	bca	b	16	yes	no	no
316	67	C 2/m 2/m 2/a	-C 2a 2	abc	c	16	yes	no	no
317	67	C 2/m 2/m 2/b	-C 2a 2a	ba-c	c	16	yes	no	no
318	67	A 2/b 2/m 2/m	-A 2b 2b	cab	a	16	yes	no	no

319	67	A 2/c 2/m 2/m	-A 2 2b	-cba	a	16	yes	no	no
320	67	B 2/m 2/c 2/m	-B 2 2a	bca	b	16	yes	no	no
321	67	B 2/m 2/a 2/m	-B 2a 2	a-cb	b	16	yes	no	no
322	68	C 2/c 2/c 2/a:1	C 2 2 -1ac	cell choice 1	c	16	yes	no	no
323	68	C 2/c 2/c 2/a:2	-C 2a 2ac	cell choice 2	c	16	yes	no	no
324	68	C 2/c 2/c 2/b:1	C 2 2 -1ac	ba-c	c	16	yes	no	no
325	68	C 2/c 2/c 2/b:2	-C 2a 2c	ba-c, cell choice 2	c	16	yes	no	no
326	68	A 2/b 2/a 2/a:1	A 2 2 -1ab	cab	a	16	yes	no	no
327	68	A 2/b 2/a 2/a:2	-A 2a 2b	cab, cell choice 2	a	16	yes	no	no
328	68	A 2/c 2/a 2/a:1	A 2 2 -1ab	-cba	a	16	yes	no	no
329	68	A 2/c 2/a 2/a:2	-A 2ab 2b	-cba, cell choice 2	a	16	yes	no	no
330	68	B 2/b 2/c 2/b:1	B 2 2 -1ab	bca	b	16	yes	no	no
331	68	B 2/b 2/c 2/b:2	-B 2ab 2b	bca, cell choice 2	b	16	yes	no	no
332	68	B 2/b 2/a 2/b:1	B 2 2 -1ab	a-cb	b	16	yes	no	no
333	68	B 2/b 2/a 2/b:2	-B 2b 2ab	a-cb, cell choice 2	b	16	yes	no	no
334	69	F 2/m 2/m 2/m	-F 2 2	cell choice 1	face	32	yes	no	no
335	70	F 2/d 2/d 2/d:1	F 2 2 -1d	cell choice 1	face	32	yes	no	no
336	70	F 2/d 2/d 2/d:2	-F 2uv 2vw	cell choice 2	face	32	yes	no	no
337	71	I 2/m 2/m 2/m	-I 2 2	cell choice 1	body	16	yes	no	no
338	72	I 2/b 2/a 2/m	-I 2 2c	abc	body	16	yes	no	no
339	72	I 2/m 2/c 2/b	-I 2a 2	cab	body	16	yes	no	no
340	72	I 2/c 2/m 2/a	-I 2b 2b	bca	body	16	yes	no	no
341	73	I 21/b 21/c 21/a	-I 2b 2c	abc	body	16	yes	no	no
342	73	I 21/c 21/a 21/b	-I 2a 2b	ba-c	body	16	yes	no	no
343	74	I 21/m 21/m 21/a	-I 2b 2	abc	body	16	yes	no	no
344	74	I 21/m 21/m 21/b	-I 2a 2a	ba-c	body	16	yes	no	no
345	74	I 21/b 21/m 21/m	-I 2c 2c	cab	body	16	yes	no	no
346	74	I 21/c 21/m 21/m	-I 2 2b	-cba	body	16	yes	no	no
347	74	I 21/m 21/c 21/m	-I 2 2a	bca	body	16	yes	no	no
348	74	I 21/m 21/a 21/m	-I 2c 2	a-cb	body	16	yes	no	no

Table 2.3: *Orthorhombic spacegroup information.*

tetragonal									
id	Int. Nr.	long Hermann-Mauguin name	Hall name	cell choice	centered	#	Chiral	Centric	Enantio-morphic
349	75	P 4	P 4	cell choice 1	primitive	4	no	yes	no
350	76	P 41	P 4w	cell choice 1	primitive	4	no	yes	yes
351	77	P 42	P 4c	cell choice 1	primitive	4	no	yes	no
352	78	P 43	P 4cw	cell choice 1	primitive	4	no	yes	yes
353	79	I 4	I 4	cell choice 1	body	8	no	yes	no
354	80	I 41	I 4bw	cell choice 1	body	8	no	yes	no
355	81	P -4	P -4	cell choice 1	primitive	4	no	no	no
356	82	I -4	I -4	cell choice 1	body	8	no	no	no
357	83	P 4/m	-P 4	cell choice 1	primitive	8	yes	no	no
358	84	P 42/m	-P 4c	cell choice 1	primitive	8	yes	no	no
359	85	P 4/n:1	P 4ab -1ab	cell choice 1	primitive	8	yes	no	no
360	85	P 4/n:2	-P 4a	cell choice 2	primitive	8	yes	no	no
361	86	P 42/n:1	P 4n -1n	cell choice 1	primitive	8	yes	no	no
362	86	P 42/n:2	-P 4bc	cell choice 2	primitive	8	yes	no	no
363	87	I 4/m	-I 4	cell choice 1	body	16	yes	no	no
364	88	I 41/a:1	I 4bw -1bw	cell choice 1	body	16	yes	no	no
365	88	I 41/a:2	-I 4ad	cell choice 2	body	16	yes	no	no
366	89	P 4 2 2	P 4 2	cell choice 1	primitive	8	no	yes	no
367	90	P 4 21 2	P 4ab 2ab	cell choice 1	primitive	8	no	yes	no
368	91	P 41 2 2	P 4w 2c	cell choice 1	primitive	8	no	yes	yes

369	92	P 41 21 2	P 4abw 2nw	cell choice 1	primitive	8	no	yes	yes
370	93	P 42 2 2	P 4c 2	cell choice 1	primitive	8	no	yes	no
371	94	P 42 21 2	P 4n 2n	cell choice 1	primitive	8	no	yes	no
372	95	P 43 2 2	P 4cw 2c	cell choice 1	primitive	8	no	yes	yes
373	96	P 43 21 2	P 4nw 2abw	cell choice 1	primitive	8	no	yes	yes
374	97	I 4 2 2	I 4 2	cell choice 1	body	16	no	yes	no
375	98	I 41 2 2	I 4bw 2bw	cell choice 1	body	16	no	yes	no
376	99	P 4 m m	P 4 -2	cell choice 1	primitive	8	no	no	no
377	100	P 4 b n	P 4 -2ab	cell choice 1	primitive	8	no	no	no
378	101	P 42 c m	P 4c -2c	cell choice 1	primitive	8	no	no	no
379	102	P 42 n m	P 4n -2n	cell choice 1	primitive	8	no	no	no
380	103	P 4 c c	P 4 -2c	cell choice 1	primitive	8	no	no	no
381	104	P 4 n c	P 4 -2n	cell choice 1	primitive	8	no	no	no
382	105	P 42 m c	P 4c -2	cell choice 1	primitive	8	no	no	no
383	106	P 42 b c	P 4c -2ab	cell choice 1	primitive	8	no	no	no
384	107	I 4 m m	I 4 -2	cell choice 1	body	16	no	no	no
385	108	I 4 c m	I 4 -2c	cell choice 1	body	16	no	no	no
386	109	I 41 m d	I 4bw -2	cell choice 1	body	16	no	no	no
387	110	I 41 c d	I 4bw -2c	cell choice 1	body	16	no	no	no
388	111	P -4 2 m	P -4 2	cell choice 1	primitive	8	no	no	no
389	112	P -4 2 c	P -4 2c	cell choice 1	primitive	8	no	no	no
390	113	P -4 21 m	P -4 2ab	cell choice 1	primitive	8	no	no	no
391	114	P -4 21 c	P -4 2n	cell choice 1	primitive	8	no	no	no
392	115	P -4 m 2	P -4 -2	cell choice 1	primitive	8	no	no	no
393	116	P -4 c 2	P -4 -2c	cell choice 1	primitive	8	no	no	no
394	117	P -4 b 2	P -4 -2ab	cell choice 1	primitive	8	no	no	no
395	118	P -4 n 2	P -4 -2n	cell choice 1	primitive	8	no	no	no
396	119	I -4 m 2	I -4 -2	cell choice 1	body	16	no	no	no
397	120	I -4 c 2	I -4 -2c	cell choice 1	body	16	no	no	no
398	121	I -4 2 m	I -4 2	cell choice 1	body	16	no	no	no
399	122	I -4 2 d	I -4 2bw	cell choice 1	body	16	no	no	no
400	123	P 4/m 2/m 2/m	-P 4 2	cell choice 1	primitive	16	yes	no	no
401	124	P 4/m 2/c 2/c	-P 4 2c	cell choice 1	primitive	16	yes	no	no
402	125	P 4/n 2/b 2/m:1	P 4 2 -1ab	cell choice 1	primitive	16	yes	no	no
403	125	P 4/n 2/b 2/m:2	-P 4a 2b	cell choice 2	primitive	16	yes	no	no
404	126	P 4/n 2/n 2/c:1	P 4 2 -1n	cell choice 1	primitive	16	yes	no	no
405	126	P 4/n 2/n 2/c:2	-P 4a 2bc	cell choice 2	primitive	16	yes	no	no
406	127	P 4/m 21/b 2/m	-P 4 2ab	cell choice 1	primitive	16	yes	no	no
407	128	P 4/m 21/n 2/c	-P 4 2n	cell choice 1	primitive	16	yes	no	no
408	129	P 4/n 21/m 2/m:1	P 4ab 2ab -1ab	cell choice 1	primitive	16	yes	no	no
409	129	P 4/n 21/m 2/m:2	-P 4a 2a	cell choice 2	primitive	16	yes	no	no
410	130	P 4/n 21/c 2/c:1	P 4ab 2n -1ab	cell choice 1	primitive	16	yes	no	no
411	130	P 4/n 21/c 2/c:2	-P 4a 2ac	cell choice 2	primitive	16	yes	no	no
412	131	P 42/m 2/m 2/c	-P 4c 2	cell choice 1	primitive	16	yes	no	no
413	132	P 42/m 2/c 2/m	-P 4c 2c	cell choice 1	primitive	16	yes	no	no
414	133	P 42/n 2/b 2/c:1	P 4n 2c -1n	cell choice 1	primitive	16	yes	no	no
415	133	P 42/n 2/b 2/c:2	-P 4ac 2b	cell choice 2	primitive	16	yes	no	no
416	134	P 42/n 2/n 2/m:1	P 4n 2 -1n	cell choice 1	primitive	16	yes	no	no
417	134	P 42/n 2/n 2/m:2	-P 4ac 2bc	cell choice 2	primitive	16	yes	no	no
418	135	P 42/m 21/b 2/c	-P 4c 2ab	cell choice 1	primitive	16	yes	no	no
419	136	P 42/m 21/n 2/m	-P 4n 2n	cell choice 1	primitive	16	yes	no	no
420	137	P 42/n 21/m 2/c:1	P 4n 2n -1n	cell choice 1	primitive	16	yes	no	no
421	137	P 42/n 21/m 2/c:2	-P 4ac 2a	cell choice 2	primitive	16	yes	no	no
422	138	P 42/n 21/c 2/m:1	P 4n 2ab -1n	cell choice 1	primitive	16	yes	no	no
423	138	P 42/n 21/c 2/m:2	-P 4ac 2ac	cell choice 2	primitive	16	yes	no	no
424	139	I 4/m 2/m 2/m	-I 4 2	cell choice 1	primitive	32	yes	no	no

425	140	I 4/m 2/c 2/m	-I 4 2c	cell choice 1	primitive	32	yes	no	no
426	141	I 41/a 2/m 2/d:1	I 4bw 2bw -1bw	cell choice 1	body	32	yes	no	no
427	141	I 41/a 2/m 2/d:2	-I 4bd 2	cell choice 2	body	32	yes	no	no
428	142	I 41/a 2/c 2/d:1	I 4bw 2aw -1bw	cell choice 1	body	32	yes	no	no
429	142	I 41/a 2/c 2/d:2	-I 4bd 2c	cell choice 2	body	32	yes	no	no

Table 2.4: *Tetragonal spacegroup information.*

trigonal									
id	Int. Nr.	long Hermann-Mauguin name	Hall name	cell choice	centered	#	Chiral	Centric	Enantio-morphic
430	143	P 3	P 3	cell choice 1	primitive	3	no	yes	no
431	144	P 31	P 31	cell choice 1	primitive	3	no	yes	yes
432	145	P 32	P 32	cell choice 1	primitive	3	no	yes	no
433	146	R 3:H	R 3	hexagonal	rhombohedral	9	no	yes	no
434	146	R 3:R	P 3*	Rhombohedral	primitive	3	no	yes	no
435	147	P -3	-P 3	cell choice 1	primitive	6	yes	no	no
436	148	R -3:H	-R 3	hexagonal	rhombohedral	18	yes	no	no
437	148	R -3:R	-P 3*	Rhombohedral	primitive	6	yes	no	no
438	149	P 3 1 2	P 3 2	cell choice 1	primitive	6	no	yes	no
439	150	P 3 2 1	P 3 2″	cell choice 1	primitive	6	no	yes	no
440	151	P 31 1 2	P 31 2 (0 0 4)	cell choice 1	primitive	6	no	yes	yes
441	152	P 31 2 1	P 31 2″	cell choice 1	primitive	6	no	yes	yes
442	153	P 32 1 2	P 32 2 (0 0 2)	cell choice 1	primitive	6	no	yes	yes
443	154	P 32 2 1	P 32 2″	cell choice 1	primitive	6	no	yes	yes
444	155	R 3 2:H	R 3 2″	hexagonal	rhombohedral	18	no	yes	no
445	155	R 3 2:R	P 3* 2	Rhombohedral	primitive	6	no	yes	no
446	156	P 3 m 1	P 3 -2″	cell choice 1	primitive	6	no	no	no
447	157	P 3 1 m	P 3 -2	cell choice 1	primitive	6	no	no	no
448	158	P 3 c 1	P 3 -2″c	cell choice 1	primitive	6	no	no	no
449	159	P 3 1 c	P 3 -2c	cell choice 1	primitive	6	no	no	no
450	160	R 3 m:H	R 3 -2″	hexagonal	rhombohedral	18	no	no	no
451	160	R 3 m:R	P 3* -2	Rhombohedral	primitive	6	no	no	no
452	161	R 3 c:H	R 3 -2″c	hexagonal	rhombohedral	18	no	no	no
453	161	R 3 c:R	P 3* -2n	Rhombohedral	primitive	6	no	no	no
454	162	P -3 1 2/m	-P 3 2	cell choice 1	primitive	12	yes	no	no
455	163	P -3 1 2/c	-P 3 2c	cell choice 1	primitive	12	yes	no	no
456	164	P -3 2/m 1	-P 3 2″	cell choice 1	primitive	12	yes	no	no
457	165	P -3 2/c 1	-P 3 2″c	cell choice 1	primitive	12	yes	no	no
458	166	R -3 2/m:H	-R 3 2″	hexagonal	rhombohedral	36	yes	no	no
459	166	R -3 2/m:R	-P 3* 2	Rhombohedral	primitive	12	yes	no	no
460	167	R -3 2/c:H	-R 3 2″c	hexagonal	rhombohedral	36	yes	no	no
461	167	R -3 2/c:R	-P 3* 2n	Rhombohedral	primitive	12	yes	no	no

Table 2.5: *Trigonal spacegroup information.*

hexagonal									
id	Int. Nr.	long Hermann-Mauguin name	Hall name	cell choice	centered	#	Chiral	Centric	Enantio-morphic
462	168	P 6	P 6	cell choice 1	primitive	6	no	yes	no
463	169	P 61	P 61	cell choice 1	primitive	6	no	yes	yes
464	170	P 65	P 65	cell choice 1	primitive	6	no	yes	yes
465	171	P 62	P 62	cell choice 1	primitive	6	no	yes	yes
466	172	P 64	P 64	cell choice 1	primitive	6	no	yes	yes
467	173	P 63	P 6c	cell choice 1	primitive	6	no	yes	no

468	174	P -6	P -6	cell choice 1	primitive	6	no	no	no
469	175	P 6/m	-P 6	cell choice 1	primitive	6	yes	no	no
470	176	P 63/m	-P 6c	cell choice 1	primitive	12	yes	no	no
471	177	P 6 2 2	P 6 2	cell choice 1	primitive	12	no	yes	no
472	178	P 61 2 2	P 61 2 (0 0 5)	cell choice 1	primitive	12	no	yes	yes
473	179	P 65 2 2	P 65 2 (0 0 1)	cell choice 1	primitive	12	no	yes	yes
474	180	P 62 2 2	P 62 2 (0 0 4)	cell choice 1	primitive	12	no	yes	yes
475	181	P 64 2 2	P 64 2 (0 0 2)	cell choice 1	primitive	12	no	yes	yes
476	182	P 63 2 2	P 6c 2c	cell choice 1	primitive	12	no	yes	no
477	183	P 6 m m	P 6 -2	cell choice 1	primitive	12	no	no	no
478	184	P 6 c c	P 6 -2c	cell choice 1	primitive	12	no	no	no
479	185	P 63 c m	P 6c -2	cell choice 1	primitive	12	no	no	no
480	186	P 63 m c	P 6c -2c	cell choice 1	primitive	12	no	no	no
481	187	P -6 m 2	P -6 2	cell choice 1	primitive	12	no	no	no
482	188	P -6 c 2	P -6c 2	cell choice 1	primitive	12	no	no	no
483	189	P -6 2 m	P -6 -2	cell choice 1	primitive	12	no	no	no
484	190	P -6 2 c	P -6c -2c	cell choice 1	primitive	12	no	no	no
485	191	P 6/m 2/m 2/m	-P 6 2	cell choice 1	primitive	24	yes	no	no
486	192	P 6/m 2/c 2/c	-P 6 2c	cell choice 1	primitive	24	yes	no	no
487	193	P 63/m 2/c 2/m	-P 6c 2	cell choice 1	primitive	24	yes	no	no
488	194	P 63/m 2/m 2/c	-P 6c 2c	cell choice 1	primitive	24	yes	no	no

Table 2.6: *Hexagonal spacegroup information.*

cubic									
id	Int. Nr.	long Hermann-Mauguin name	Hall name	cell choice	centered	#	Chiral	Centric	Enantio-morphic
489	195	P 2 3	P 2 2 3	cell choice 1	primitive	12	no	yes	no
490	196	F 2 3	F 2 2 3	cell choice 1	face	48	no	yes	no
491	197	I 2 3	I 2 2 3	cell choice 1	body	24	no	yes	no
492	198	P 21 3	P 2ac 2ab 3	cell choice 1	primitive	12	no	yes	no
493	199	I 21 3	I 2b 2c 3	cell choice 1	body	24	no	yes	no
494	200	P 2/m -3	-P 2 2 3	cell choice 1	primitive	24	yes	no	no
495	201	P 2/n -3:1	P 2 2 3 -1n	cell choice 1	primitive	24	yes	no	no
496	201	P 2/n -3:2	-P 2ab 2bc 3	cell choice 2	primitive	24	yes	no	no
497	202	F 2/m -3	-F 2 2 3	cell choice 1	face	96	yes	no	no
498	203	F 2/d -3:1	F 2 2 3 -1d	cell choice 1	face	96	yes	no	no
499	203	F 2/d -3:2	-F 2uv 2vw 3	cell choice 2	face	96	yes	no	no
500	204	I 2/m -3	-I 2 2 3	cell choice 1	body	48	yes	no	no
501	205	P 21/a -3	-P 2ac 2ab 3	cell choice 1	primitive	24	yes	no	no
502	206	I 21/a -3	-I 2b 2c 3	cell choice 1	body	48	yes	no	no
503	207	P 4 3 2	P 4 2 3	cell choice 1	primitive	24	no	yes	no
504	208	P 42 3 2	P 4n 2 3	cell choice 1	primitive	24	no	yes	no
505	209	F 4 3 2	F 4 2 3	cell choice 1	face	96	no	yes	no
506	210	F 41 3 2	F 4d 2 3	cell choice 1	face	96	no	yes	no
507	211	I 4 3 2	I 4 2 3	cell choice 1	body	48	no	yes	no
508	212	P 43 3 2	P 4acd 2ab 3	cell choice 1	primitive	24	no	yes	yes
509	213	P 41 3 2	P 4bd 2ab 3	cell choice 1	primitive	24	no	yes	yes
510	214	I 41 3 2	I 4bd 2c 3	cell choice 1	body	48	no	yes	no
511	215	P -4 3 m	P -4 2 3	cell choice 1	primitive	24	no	no	no
512	216	F -4 3 m	F -4 2 3	cell choice 1	face	96	no	no	no
513	217	I -4 3 m	I -4 2 3	cell choice 1	body	48	no	no	no
514	218	P -4 3 n	P -4n 2 3	cell choice 1	primitive	24	no	no	no
515	219	F -4 3 c	F -4a 2 3	cell choice 1	face	96	no	no	no
516	220	I -4 3 d	I -4bd 2c 3	cell choice 1	body	48	no	no	no
517	221	P 4/m -3 2/m	-P 4 2 3	cell choice 1	primitive	48	yes	no	no

518	222	P 4/n -3 2/n:1	P 4 2 3 -1n	cell choice 1	primitive	48	yes	no	no
519	222	P 4/n -3 2/n:2	-P 4a 2bc 3	cell choice 2	primitive	48	yes	no	no
520	223	P 42/m -3 2/n	-P 4n 2 3	cell choice 1	primitive	48	yes	no	no
521	224	P 42/n -3 2/m:1	P 4n 2 3 -1n	cell choice 1	primitive	48	yes	no	no
522	224	P 42/n -3 2/m:2	-P 4bc 2bc 3	cell choice 2	primitive	48	yes	no	no
523	225	F 4/m -3 2/m	-F 4 2 3	cell choice 1	face	192	yes	no	no
524	226	F 4/m -3 2/c	-F 4a 2 3	cell choice 1	face	192	yes	no	no
525	227	F 41/d -3 2/m:1	F 4d 2 3 -1d	cell choice 1	face	192	yes	no	no
526	227	F 41/d -3 2/m:2	-F 4vw 2vw 3	cell choice 2	face	192	yes	no	no
527	228	F 41/d -3 2/c	F 4d 2 3 -1ad	cell choice 1	face	192	yes	no	no
528	228	F 41/d -3 2/c	-F 4ud 2vw 3	cell choice 2	face	192	yes	no	no
529	229	I 4/m -3 2/m	-I 4 2 3	cell choice 1	body	96	yes	no	no
530	230	I 41/a -3 2/d	-I 4bd 2c 3	cell choice 1	body	96	yes	no	no

Table 2.7: *Cubic spacegroup information.*

3 Potentials

3.1 Functional forms of force fields

The molecular energy can be described as an Taylor expansion in bonds, bends, torsions, etc.

$$\begin{aligned}
U = &\sum_{\text{bonds}} U_r\left(r\right) + \sum_{\text{bends}} U_\theta\left(\theta\right) + \sum_{\text{torsions}} U_\phi\left(\phi\right) + \sum_{\text{out-of-plane bends}} U_\chi\left(\chi\right) + \sum_{\text{non-bonded}} U_{nb}\left(r\right) \\
&+ \sum_{\text{bond-bond}} U_{bb'}\left(r,r'\right) + \sum_{\text{bond-bend}} U_{b\theta'}\left(r,\theta\right) + \sum_{\text{bend-bend}} U_{\theta\theta'}\left(\theta,\theta'\right) \\
&+ \sum_{\text{bond-torsion}} U_{r\phi}\left(r,\phi,r'\right) + \sum_{\text{bend-torsion}} U_{\theta\phi}\left(\theta,\phi,\theta'\right) + \dots
\end{aligned} \tag{3.1}$$

This expansion is believed to capture all the chemical entities we can think of, such as atoms, bonds, angles, etc, and physical properties like equilibrium structures, vibrational spectra, etc. The cross terms are not ad-hoc functions, but arise naturally from this expansion. For example, bonds and bends interact, as the bend angle becomes smaller the bond lengths tend to increase. Their inclusion leads to two advantages: 1) they increase the accuracy of the force field (especially the vibrational frequencies), and 2) they increase the transferability of the diagonal terms $U_r\left(r\right), U_\theta\left(\theta\right), U_\phi\left(\phi\right), U_\chi\left(\chi\right)$. On top of the terms in Eq. 3.1 one can add ad hoc terms, such as hydrogen bonding, that are not adequately accounted for otherwise.

Eq. 3.1 is historically referred to as an *force field*. The name arose from the lowest order approximation using only springs with *force constants*. Force fields have matured and have become quite accurate and many parameters exists for a wide range of structure. These parameters are crucial and determine the quality of the force field. Unfortunately, deriving high quality parameters remains more than a art rather than a science. However, some progress has been made and in the end of the chapter some algorithms are described how to obtain them.

The terms in Eq. 3.1 consists of a functional form, force constants (a resistance against a change from the optimum value), and a reference value. The functional form is chosen such as to be an accurate description of the true potential energy (either known from experiment or from quantum mechanics), although one can simplify the functional form to decrease computational evaluation time of the energy at the cost of diminished accuracy. This tradeoff has almost vanished for intra-molecular potentials but is still an issue for the non-bonded terms. The reference value is *not* the equilibrium value (except by chance). For example, bond lengths are affected by all other terms in the force field and the more strained a molecule the farther the bond equilibrium length will deviate from its reference value. This means that one can not simply take the equilibrium values from known experiment.

3.2 Bonded potentials diagonal terms

3.2.1 Bond-stretching potentials

The bond stretching potential describes the change in energy as the bond stretches and contracts. The simplest functional form would be Hook's law:

$$U = \frac{1}{2}k\left(r - r_0\right)^2 \tag{3.2}$$

where k is the force constant and r_0 the reference value for the bond. This form is computationally very fast, but not very realistic. It is well known that it is easier to stretch a bond than it is to compress a bond. The 'Morse' potential is an-harmonic and provides a much better description of the energy

$$U = D\left(1 - e^{-\alpha(r-r_0)}\right)^2 \tag{3.3}$$

Expanding around the equilibrium value leads to

$$U = D\alpha^2\left(r - r_0\right)^2\left[1 - \alpha\left(r - r_0\right) + \frac{7}{12}\alpha^2\left(r - r_0\right)^2\ldots\right] \tag{3.4}$$

The first terms is the harmonic potential (with $k = 2D\alpha^2$) and for organic structures where distortions from equilibrium are small the difference between the potentials are small. However, for larger deviations the Morse potential provides a significantly better description. The Morse potential provides a restoring force which goes to zero at long distances. For minimizations starting far equilibrium could result in non-convergence. Some force fields solved this problem by using modification of Hook's law. MM2 added a cubic term making the bond an-harmonic. However, this leads to large negative energies for poor initial geometries with large distortions. MM3 added the quartic term to solve this. Note the 7/12 terms in the MM2/3 functional forms originate from the Taylor expansion of the Morse potential, and the cubic and quartic terms are chosen to mimic the Morse potentials for moderate distortions. Dinur and Hagler proposed a functional form based on inverse bond lengths which follows the true potential energy compared to QM over an even wider range

$$U = U_0 + C_2\left(\frac{1}{r} - \frac{1}{r_0}\right)^2 + C_3\left(\frac{1}{r} - \frac{1}{r_0}\right)^3 \tag{3.5}$$

The implemented bond-potentials:

- HARMONIC_BOND
$$U = \frac{1}{2}p_0\left(r - p_1\right)^2 \tag{3.6}$$
2 arguments: p_0/k_B in units of K/Å^2, p_1 in Å.
- CORE_SHELL_SPRING
$$U = \frac{1}{2}p_0 r^2 \tag{3.7}$$
1 argument: p_0/k_B in units of K/Å^2.
- MORSE_BOND
$$U = p_0\left[\left(1 - e^{-p_1(r-p_2)}\right)^2 - 1\right] \tag{3.8}$$
3 arguments: p_0/k_B in units of K, p_1 in Å^{-1}, and p_2 in Å.

- LJ_12_6_BOND
$$U = \frac{p_0}{r^{12}} - \frac{p_1}{r^6} \tag{3.9}$$
2 arguments: p_0/k_B in units of K Å^{12}, and p_1/k_B in units of K Å^6.

- LENNARD_JONES_BOND
$$U = 4p_0 \left[\left(\frac{p_1}{r}\right)^{12} - \left(\frac{p_1}{r}\right)^6\right] \tag{3.10}$$
2 arguments: p_0/k_B in units of K, p_1 in Å.

- BUCKINGHAM_BOND
$$U = p_0 e^{-p_1 r} - \frac{p_2}{r^6} \tag{3.11}$$
3 arguments: p_0/k_B in units of K, p_1 in units of Å^{-1}, and p_2/k_B in K Å^6.

- RESTRAINED_HARMONIC_BOND
$$U = \begin{cases} \frac{1}{2}p_0\left(r-p_1\right)^2 & |r-p_1| \le p_2 \\ \frac{1}{2}p_0 p_2^2 + p_0 p_2\left(|r-p_1| - p_2\right) & |r-p_1| > p_2 \end{cases} \tag{3.12}$$
3 arguments: p_0/k_B in units of K/Å^2, p_1 in Å, and p_2 in Å.

- QUARTIC_BOND
$$U = \frac{1}{2}p_0\left(r-p_1\right)^2 + \frac{1}{3}p_2\left(r-p_1\right)^3 + \frac{1}{4}p_3\left(r-p_1\right)^4 \tag{3.13}$$
4 arguments: p_0/k_B in units of K/Å^2, p_1 in Å, p_2/k_B in K/Å^3, and p_3/k_B in K/Å^4.

- CFF_QUARTIC_BOND
$$U = p_0\left(r-p_1\right)^2 + p_2\left(r-p_1\right)^3 + p_3\left(r-p_1\right)^4 \tag{3.14}$$
4 arguments: p_0/k_B in units of K/Å^2, p_1 in Å, p_2/k_B in K/Å^3, and p_3/k_B in K/Å^4.

- MM3_BOND
$$U = p_0\left(r-p_1\right)^2\left(1 - 2.55\left(r-p_1\right) + \frac{7}{12}2.55^2\left(r-p_1\right)^2\right) \tag{3.15}$$
2 arguments: p_0 in units of mdyne/Å molecule, p_1 in Å.

- RIGID_BOND
Use for connections between rigid units.

- FIXED_BOND
Use for bonds constraint using the 'SHAKE' and 'RATTLE'-algorithm. Applies to Monte-Carlo, Molecular Dynamics, and minimization.

- MEASURE_BOND
A histogram of the bond-distance can be computed.

3.2.2 Urey-Bradley potentials

The Urey-Bradley potential is sometimes used to account for the repulsion between two atoms bound to a common atom. In more modern force field they are replaced by bond/bend cross potentials. Urey-Bradley are essentially just bonds between 1-3 nearest neighbor atoms and the same range of potentials is offered as for 1-2 bonds in RASPA.

- HARMONIC_UREYBRADLEY
$$U = \frac{1}{2} p_0 \left(r - p_1\right)^2 \tag{3.16}$$
2 arguments: p_0/k_B in units of K/Å^2, p_1 in Å.

- MORSE_UREYBRADLEY
$$U = p_0 \left[\left(1 - e^{-p_1 (r - p_2)}\right)^2 - 1 \right] \tag{3.17}$$
3 arguments: p_0/k_B in units of K, p_1 in Å^{-1}, and p_2 in Å.

- LJ_12_6_UREYBRADLEY
$$U = \frac{p_0}{r^{12}} - \frac{p_1}{r^6} \tag{3.18}$$
2 arguments: p_0/k_B in units of K Å^{12}, and p_1/k_B in units of K Å^6.

- LENNARD_JONES_UREYBRADLEY
$$U = 4 p_0 \left[\left(\frac{p_1}{r}\right)^{12} - \left(\frac{p_1}{r}\right)^6 \right] \tag{3.19}$$
2 arguments: p_0/k_B in units of K, p_1 in Å.

- BUCKINGHAM_UREYBRADLEY
$$U = p_0 e^{-p_1 r} - \frac{p_2}{r^6} \tag{3.20}$$
3 arguments: p_0/k_B in units of K, p_1 in units of Å^{-1}, and p_2/k_B in K Å^6.

- RESTRAINED_HARMONIC_UREYBRADLEY
$$U = \begin{cases} \frac{1}{2} p_0 \left(r - p_1\right)^2 & |r - p_1| \leq p_2 \\ \frac{1}{2} p_0 p_2^2 + p_0 p_2 \left(|r - p_1| - p_2\right) & |r - p_1| > p_2 \end{cases} \tag{3.21}$$
3 arguments: p_0/k_B in units of K/Å^2, p_1 in Å, and p_2 in Å.

- QUARTIC_UREYBRADLEY
$$U = \frac{1}{2} p_0 \left(r - p_1\right)^2 + \frac{1}{3} p_2 \left(r - p_1\right)^3 + \frac{1}{4} p_3 \left(r - p_1\right)^4 \tag{3.22}$$
4 arguments: p_0/k_B in units of K/Å^2, p_1 in Å, p_2/k_B in K/Å^3, and p_3/k_B in K/Å^4.

- CFF_QUARTIC_UREYBRADLEY
$$U = p_0 \left(r - p_1\right)^2 + p_2 \left(r - p_1\right)^3 + p_3 \left(r - p_1\right)^4 \tag{3.23}$$
4 arguments: p_0/k_B in units of K/Å^2, p_1 in Å, p_2/k_B in K/Å^3, and p_3/k_B in K/Å^4.

- MM3_UREYBRADLEY
$$U = p_0 \left(r - p_1\right)^2 \left(1 - 2.55 \left(r - p_1\right) + \frac{7}{12} 2.55^2 \left(r - p_1\right)^2\right) \tag{3.24}$$
2 arguments: p_0 in units of mdyne/Å molecule, p_1 in Å.

- RIGID_UREYBRADLEY
Use for connections between rigid units.

- FIXED_UREYBRADLEY
Use for bonds constraint using the 'SHAKE' and 'RATTLE'-algorithm. Applies to Monte-Carlo, Molecular Dynamics, and minimization.

- MEASURE_UREYBRADLEY
A histogram of the Urey-Bradley distance can be computed.

3.2.3 Bending potential

The simplest approach for an angle potential is the harmonic potential

$$U = \frac{1}{2} k \left(\theta - \theta_0\right)^2 \tag{3.25}$$

Angles are much softer than bonds, especially in zeolites where a Si-O-Si angle ranges between 135 and 180 degrees. A problem with all polynomial representations of angles is that angles of 180 degrees results in singular point (unless the reference angle is 180 degrees). The case of 0 degree is not possible due to repulsion of the i and k atoms in the i-j-k bend. The singularity is due to the fact that the force expression of such a polynomial contains a factor $1/\sin(\theta)$. A common solution is to use a trigonometric function

$$U = \frac{1}{2} k \left[\cos(\theta) - \cos(\theta_0)\right]^2 \tag{3.26}$$

Note that close to the maximum these potentials have no restoring force, but for small distortions this is not a problem. The MM force fields use higher order terms. A six power term was needed to describe the highly bent bicyclo[1.1.1]pentane. Cubic terms and higher become desirable when the bending is more then 10-15 degrees. MM3 angle bending has been divided into in-plane and out-of-plane bending for planar trigonal centers.

- HARMONIC_BEND,CORE_SHELL_BEND

$$U = \frac{1}{2} p_0 \left(\theta_{ijk} - p_1\right)^2 \tag{3.27}$$

  2 arguments: p_0/k_B in units of K/rad^2 and p_1 in degrees.

- QUARTIC_BEND

$$U = \frac{1}{2} p_0 \left(\theta_{ijk} - p_1\right)^2 + \frac{1}{3} p_2 \left(\theta_{ijk} - p_1\right)^3 + \frac{1}{4} p_3 \left(\theta_{ijk} - p_1\right)^4 \tag{3.28}$$

  4 arguments: p_0/k_B in units of K/rad^2, p_1 in degrees, p_2/k_B in K/rad^3, and p_3/k_B in K/rad^4.

- CFF_QUARTIC_BEND

$$U = p_0 \left(\theta_{ijk} - p_1\right)^2 + p_2 \left(\theta_{ijk} - p_1\right)^3 + p_3 \left(\theta_{ijk} - p_1\right)^4 \tag{3.29}$$

  4 arguments: p_0/k_B in units of K/rad^2, p_1 in degrees, p_2/k_B in K/rad^3, and p_3/k_B in K/rad^4.

- HARMONIC_COSINE_BEND

$$U = \frac{1}{2} p_0 \left(\cos\theta_{ijk} - \cos p_1\right)^2 \tag{3.30}$$

  2 arguments: p_0/k_B in units of K and p_1 in degrees.

- COSINE_BEND

$$U = p_0 \left(1 + \cos\left(p_1 \theta_{ijk} - p_2\right)\right) \tag{3.31}$$

  3 arguments: p_0/k_B in units of K, p_1 dimensionless, and p_2 in degrees.

- MM3_BEND

$$U = \frac{1}{2} p_0 \left(\theta_{ijk} - p_1\right)^2 \Big(1 - 0.014 \left(\theta_{ijk} - p_1\right) + 5.6 \times 10^{-5} \left(\theta_{ijk} - p_1\right)^2 - 7 \times 10^{-7} \left(\theta_{ijk} - p_1\right)^3 + 2.2 \times 10^{-8} \left(\theta_{ijk} - p_1\right)^4\Big) \tag{3.32}$$

  2 arguments: p_0 in units of mdyne Å/rad^2, p_1 in degrees.

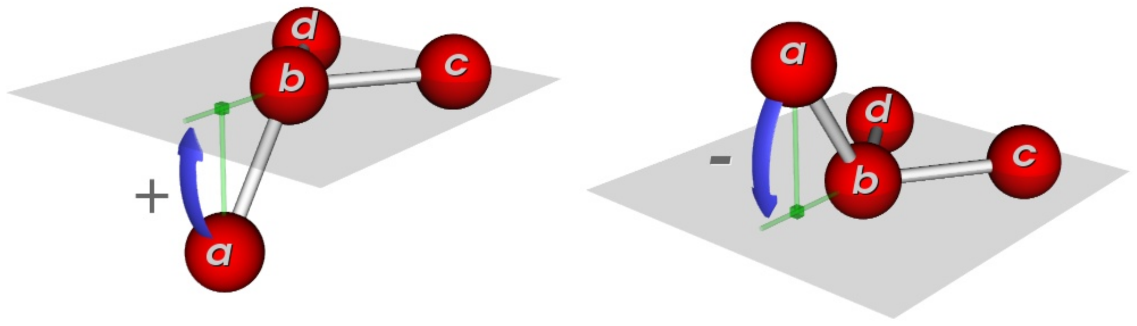


Figure 5: *The definition of the Wilson inversion-bend angle χ. On the left a positive Wilson angle, and on the right a negative Wilson angle.*

- MM3_IN_PLANE_BEND

$$U = \frac{1}{2} p_0 \left(\theta_{ijk} - p_1\right)^2 \Big(1 - 0.014 \left(\theta_{ijk} - p_1\right) + 5.6 \times 10^{-5} \left(\theta_{ijk} - p_1\right)^2 - 7 \times 10^{-7} \left(\theta_{ijk} - p_1\right)^3 + 2.2 \times 10^{-8} \left(\theta_{ijk} - p_1\right)^4\Big) \tag{3.33}$$

  2 arguments: p_0 in units of mdyne Å/rad^2, p_1 in degrees. The bend is 'in-plane' and only applicable to bends in a defined planar trigonal centers. The bend is dependend on the fourth atom of the trigonal center.

- FIXED_BEND
  Use for bend-angle constraint using the 'SHAKE' and 'RATTLE'-algorithm. Applies to Molecular Dynamics and minimization. Does not work (yet) in Monte-Carlo.

- MEASURE_BEND
  A histogram of the bend angle can be computed.

3.2.4 Wilson inversion-bend potential

Common planar molecule that contain a double bond or sp^2 hybridization form planar groups with trigonal centers. For example: the carbon and nitrogen centers in formamide, and the carbon centers in benzene. The mode of motion is different from bond stretching, bending, and internal rotation. The associated harmonic potential is

$$U = \frac{1}{2} k \left(\chi\right)^2 \tag{3.34}$$

with χ the out-of-plane angle. Two possible definitions are in use

1. the distance of the central atom from the plane defined by the other three atoms (pyramid height),
2. the average angle between any bond that extends from the central atom and the plane defined by the other two bonds.

Note that an alternative to the out-of-plane angle is the *improper torsion* using

$$U = \frac{1}{2} k \left(1 - \cos 2\chi\right) \tag{3.35}$$

The out-of-plane potential can also be used for non-planar structure, for example in united-atom for chiral centers to avoid inversion of the chiral center. Another example of its use is coordination complexes where

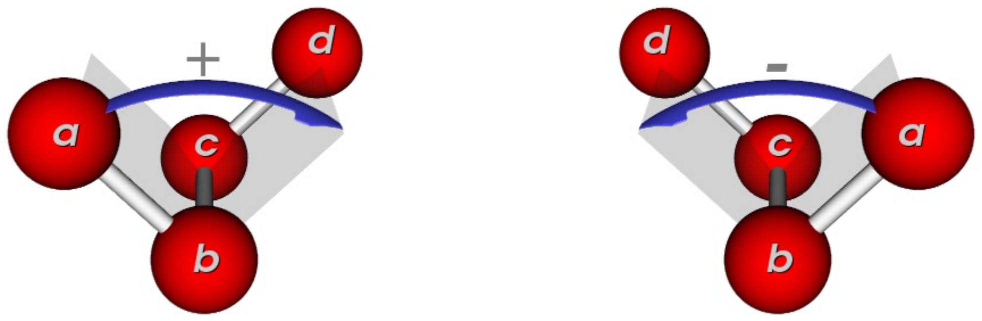


Figure 6: *The definition of the dihedral angle ϕ: the angle between the planes formed by atoms a-b-c and b-c-d. On the left a positive dihedral angle, and on the right a negative dihedral angle.*

now the plane of the ligands need no longer be defined exactly. In square planar complexes it is necessary to define an average plane through the ligands (usually the least-square plane). Note that the definition include one central atom which is listed as the second in $a-b-c-d$: a, c, and d are bonded to the central atom b. The inversion angle potential is the average of the three possible inversion angle terms.

- HARMONIC_INVERSION

$$U = \frac{1}{2}p_0\left(\chi_{ijk} - p_1\right)^2 \tag{3.36}$$

  2 arguments: p_0/k_B in units of K/rad^2 and p_1 in degrees.

- HARMONIC_COSINE_INVERSION

$$U = \frac{1}{2}p_0\left(\cos\left(\chi_{ijk}\right) - \cos\left(p_1\right)\right)^2 \tag{3.37}$$

  2 arguments: p_0/k_B in units of K and p_1 in degrees.

- PLANAR_INVERSION

$$U = p_0\left(1 - \cos\left(\chi\right)\right) \tag{3.38}$$

  1 argument: p_0/k_B in units of K.

- MM3_INVERSION

$$\begin{aligned} U = \frac{1}{2}p_0\left(\chi - p_1\right)^2 \Big(1 - 0.014\left(\chi - p_1\right) + 5.6\times10^{-5}\left(\chi - p_1\right)^2 - 7\times10^{-7}\left(\chi - p_1\right)^3 \\ + 2.2\times10^{-8}\left(\chi - p_1\right)^4\Big) \end{aligned} \tag{3.39}$$

  2 arguments: p_0 in units of mdyne Å/rad^2, p_1 in degrees.

- FIXED_INVERSION_BEND
  Use for inversion bend-angle constraint using the 'SHAKE' and 'RATTLE'-algorithm. Applies to Molecular Dynamics and minimization. Does not work (yet) in Monte-Carlo.

3.2.5 Torsion potential

Intramolecular rotations about bonds do not occur freely. A possible description with a physical interpretation is the three-term Fourier expansion

$$U = \frac{V_1}{2}\left[1+\cos\phi\right] + \frac{V_2}{2}\left[1-\cos 2\phi\right] + \frac{V_3}{2}\left[1+\cos 3\phi\right] \tag{3.40}$$

1. the 1 fold-term has been attributed to residual dipole-dipole interactions, to Van der Waal interactions, or to any other direct interaction between atoms not accounted for otherwise,
2. the 2-fold arises from conjugation or hyper conjugation, being geometrically related to p orbitals,
3. and the 3-fold term has a steric (or bonding/anti-bonding) origin.

The values for 4-fold or higher are small and it is not known whether these are essential to include. It may be that Van der Waals and dipole interactions already take care of these effects. Torsions are even softer than bond angles. All possible values can be found in structures. Therefore, the energy function must be valid over the entire range, the function must be periodic, and for reasons of symmetry have stationary points at 0 and 180 degrees. The periodicity is the number of minima for the potential, usually 3 for an sp^3-sp^3 bond and 2 for a conjugate bond.

The definition of a torsion includes two central and two terminal atoms. The term 'torsional' means an internal rigid rotation and 'dihedral' means a rotation of two vicinal bonds about a middle bond.

- HARMONIC_DIHEDRAL

$$U = \frac{1}{2}p_0\left(\phi_{ijkl}-p_1\right)^2 \tag{3.41}$$

2 arguments: p_0/k_B in units of K/rad^2, p_1 in degrees.

- HARMONIC_COSINE_DIHEDRAL

$$U = \frac{1}{2}p_0\left[\cos\left(\phi_{ijkl}\right)-\cos\left(p_1\right)\right]^2 \tag{3.42}$$

2 arguments: p_0/k_B in units of K, p_1 in degrees.

- THREE_COSINE_DIHEDRAL

$$U = \frac{1}{2}p_0\left[1+\cos\left(\phi_{ijkl}\right)\right] + \frac{1}{2}p_1\left[1-\cos\left(2\phi_{ijkl}\right)\right] + \frac{1}{2}p_2\left[1+\cos\left(3\phi_{ijkl}\right)\right] \tag{3.43}$$

3 arguments: $p_0/k_B, p_1/k_B, p_2/k_B$ in units of K

- MM3_DIHEDRAL

$$U = \frac{1}{2}p_0\left[1+\cos\left(\phi_{ijkl}\right)\right] + \frac{1}{2}p_1\left[1-\cos\left(2\phi_{ijkl}\right)\right] + \frac{1}{2}p_2\left[1+\cos\left(3\phi_{ijkl}\right)\right] \tag{3.44}$$

3 arguments: p_0, p_1, p_2 in units of kcal/mol.

- CFF_DIHEDRAL

$$U = p_0\left[1-\cos\left(\phi_{ijkl}\right)\right] + p_1\left[1-\cos\left(2\phi_{ijkl}\right)\right] + p_2\left[1-\cos\left(3\phi_{ijkl}\right)\right] \tag{3.45}$$

3 arguments: $p_0/k_B, p_1/k_B, p_2/k_B$ in units of K.

- CFF_DIHEDRAL2

$$U = p_0\left[1+\cos\left(\phi_{ijkl}\right)\right] + p_1\left[1+\cos\left(2\phi_{ijkl}\right)\right] + p_2\left[1+\cos\left(3\phi_{ijkl}\right)\right] \tag{3.46}$$

3 arguments: $p_0/k_B, p_1/k_B, p_2/k_B$ in units of K.

- SIX_COSINE_DIHEDRAL
  The Ryckaert-Bellemans potentials is often used for alkanes, the use implies exclusion of VDW-interactions between the first and last atoms of the dihedral, and $\phi' = \phi - \pi$ is defined according to the polymer convention $\phi'(trans) = 0$.

$$U = \sum_{n=0}^{5} p_n \cos^n\left(\phi'_{ijkl}\right) \tag{3.47}$$

$$= p_0 + p_1 \cos\left(\phi'_{ijkl}\right) + p_2 \cos^2\left(\phi'_{ijkl}\right) + p_3 \cos^3\left(\phi'_{ijkl}\right) p_4 \cos^4\left(\phi'_{ijkl}\right) + p_5 \cos^5\left(\phi'_{ijkl}\right) \tag{3.48}$$

  6 arguments: $p_0/k_B, \ldots, p_5/k_B$ in units of K. Rewritten in terms of ϕ the potential reads

$$U = p_0 - p_1 \cos\left(\phi_{ijkl}\right) + p_2 \cos^2\left(\phi_{ijkl}\right) - p_3 \cos^3\left(\phi_{ijkl}\right) + p_4 \cos^4\left(\phi_{ijkl}\right) - p_5 \cos^5\left(\phi_{ijkl}\right) \tag{3.49}$$

- TRAPPE_DIHEDRAL

$$U = p_0 + p_1\left[1 + \cos\left(\phi_{ijkl}\right)\right] + p_2\left[1 - \cos\left(2\phi_{ijkl}\right)\right] + p_3\left[1 + \cos\left(3\phi_{ijkl}\right)\right] \tag{3.50}$$

  4 arguments: $p_0/k_B, p_1/k_B, p_2/k_B, p_3/k_B$ in units of K.

- CVFF_DIHEDRAL

$$U = p_0\left[1 + \cos\left(p_1\phi_{ijkl} - p_2\right)\right] \tag{3.51}$$

  3 arguments: p_0/k_B in units of K, p_1 dimensionless, and p_2 in degrees.

- OPLS_DIHEDRAL

$$U = \frac{1}{2}p_0 + \frac{1}{2}p_1\left[1 + \cos\left(\phi_{ijkl}\right)\right] + \frac{1}{2}p_2\left[1 - \cos\left(2\phi_{ijkl}\right)\right] + \frac{1}{2}p_3\left[1 + \cos\left(3\phi_{ijkl}\right)\right] \tag{3.52}$$

  4 arguments: $p_0/k_B, p_1/k_B, p_2/k_B, p_3/k_B$ in units of K.

- FOURIER_SERIES_DIHEDRAL
  The general form of a Fourier expansion is:

$$U = \sum_{n=1}^{6}\left[a_n \cos\left(n\phi\right) + b_n \sin\left(n\phi\right)\right] \tag{3.53}$$

  This form uses equilibrium angles of 0 for $n = 1, 3, 5$ and 180 for $n = 2, 4, 6$

$$\begin{aligned} U =& \frac{1}{2}p_0\left[1 + \cos\phi\right] + \frac{1}{2}p_1\left[1 - \cos\left(2\phi\right)\right] + \frac{1}{2}p_2\left[1 + \cos\left(3\phi\right)\right] + \\ & \frac{1}{2}p_3\left[1 - \cos\left(4\phi\right)\right] + \frac{1}{2}p_4\left[1 + \cos\left(5\phi\right)\right] + \frac{1}{2}p_5\left[1 - \cos\left(6\phi\right)\right] \end{aligned} \tag{3.54}$$

  6 arguments: $p_0/k_B, p_1/k_B, p_2/k_B, p_3/k_B, p_4/k_B, p_5/k_B$ in units of K.

- FOURIER_SERIES_DIHEDRAL_2
  The general form of a Fourier expansion is:

$$U = \sum_{n=1}^{6}\left[a_n \cos\left(n\phi\right) + b_n \sin\left(n\phi\right)\right] \tag{3.55}$$

  This form uses equilibrium angles of 0 for $n = 1, 3, 4, 5, 6$ and 180 for $n = 2$

$$\begin{aligned} U =& \frac{1}{2}p_0\left[1 + \cos\phi\right] + \frac{1}{2}p_1\left[1 - \cos\left(2\phi\right)\right] + \frac{1}{2}p_2\left[1 + \cos\left(3\phi\right)\right] + \\ & \frac{1}{2}p_3\left[1 + \cos\left(4\phi\right)\right] + \frac{1}{2}p_4\left[1 + \cos\left(5\phi\right)\right] + \frac{1}{2}p_5\left[1 + \cos\left(6\phi\right)\right] \end{aligned} \tag{3.56}$$

  6 arguments: $p_0/k_B, p_1/k_B, p_2/k_B, p_3/k_B, p_4/k_B, p_5/k_B$ in units of K.

- FIXED_DIHEDRAL
  Use for dihedral-angle constraint using the 'SHAKE' and 'RATTLE'-algorithm. Applies to Molecular Dynamics and minimization. Does not work (yet) in Monte-Carlo.

The following identities are convenient when dealing with torsions:

$$\begin{aligned}
\cos 1x &= \cos x \\
\cos 2x &= -1 + 2\cos^2 x \\
\cos 3x &= -3\cos x + 4\cos^3 x \\
\cos 4x &= 1 - 8\cos^2 x + 8\cos^4 x \\
\cos 5x &= 5\cos x - 20\cos^3 x + 16\cos^5 x \\
\cos 6x &= -1 + 18\cos^2 x - 48\cos^4 x + 32\cos^6 x
\end{aligned} \tag{3.57}$$

$$\begin{aligned}
\sin 1x &= \sin x \\
\sin 2x &= (\sin x)(2\cos x) \\
\sin 3x &= (\sin x)(-1 + 4\cos^2 x) \\
\sin 4x &= (\sin x)(-4\cos x + 8\cos^3 x) \\
\sin 5x &= (\sin x)(1 - 12\cos^2 x + 16\cos^4 x) \\
\sin 6x &= (\sin x)(6\cos x - 32\cos^3 x + 32\cos^5 x)
\end{aligned} \tag{3.58}$$

3.2.6 Improper torsion potential

The improper torsion is an alternative for the out-of-plane angle, and a possible definition is

$$U = \frac{1}{2}k\left(1 - \cos 2\chi\right) \tag{3.59}$$

It is termed 'improper torsion' because it simply treats the four atoms in the plane as if they were bonded in the same way as in a true torsional angle. Note that the definition include one central atom which is listed as the second in $a - b - c - d$: a, c, and d are bonded to the central atom b. Improper torsions are often used to keep sp2 atoms planar and sp3 atoms in a tetrahedral geometry.

The CHARMM convention is to list the central atom first, while there are no rules how to order the other three atoms. Hence, six possibilities exist for the definition of an improper torsion. The AMBER convention is that the out-of-plane atom is listed in the third position and the order of the other atoms is determined alphabetically by atom type, and by the atom number (i.e. the order in the molecule) when atom types are identical.

- HARMONIC_IMPROPER_DIHEDRAL

$$U = \frac{1}{2}p_0\left(\phi_{ijkl} - p_1\right)^2 \tag{3.60}$$

  2 arguments: p_0/k_B in units of K/rad^2, p_1 in degrees.

- HARMONIC_COSINE_IMPROPER_DIHEDRAL

$$U = \frac{1}{2}p_0\left[\cos\left(\phi_{ijkl}\right) - \cos\left(p_1\right)\right]^2 \tag{3.61}$$

  2 arguments: p_0/k_B in units of K, p_1 in degrees.

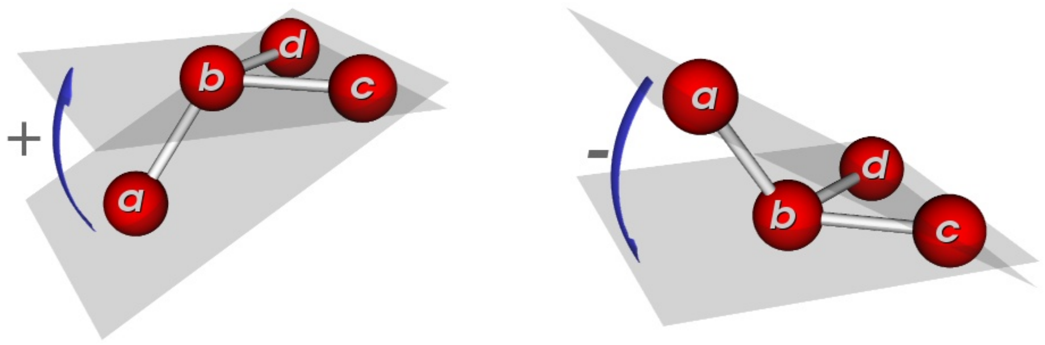


Figure 7: *The most common (CVFF, DLPOLY) definition of the improper dihedral angle ϕ: the angle between the planes formed by atoms 'a-c-d' and 'c-d-b'. On the left a positive improper dihedral angle, and on the right a negative improper dihedral angle. The atoms need to be listed in the order 'a-c-d-b'. Note that an exchange of atoms 'c' and 'd' leads to a change of sign, but* not *in magntitude.*

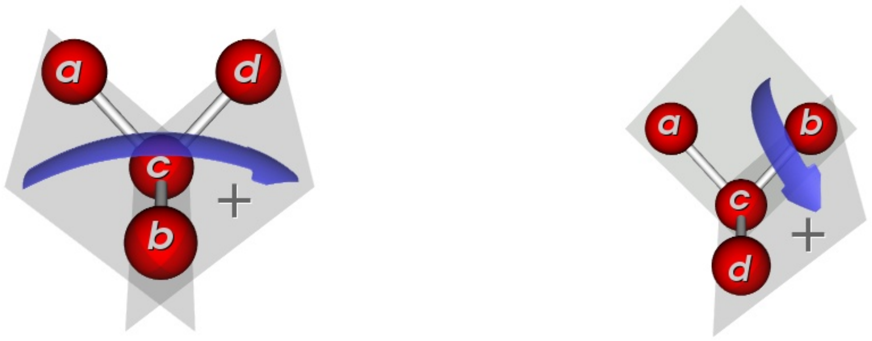


Figure 8: *A second definition of the improper dihedral angle (CHARMM, AMBER). The central atom is 'c', and the improper torsion is enter as 'a-b-c-d'. Howevere, an exchange of terminal atoms leads to a change in magntitude and the improper torsion needs to be symmetrized by adding two additional improper torsions 'b-d-c-a' and 'd-a-c-b' and rescaling the force constant by a factor of* 1/3.

- THREE_COSINE_IMPROPER_DIHEDRAL

$$U = \frac{1}{2}p_0\left[1+\cos\left(\phi_{ijkl}\right)\right] + \frac{1}{2}p_1\left[1-\cos\left(2\phi_{ijkl}\right)\right] + \frac{1}{2}p_2\left[1+\cos\left(3\phi_{ijkl}\right)\right] \tag{3.62}$$

  3 arguments: $p_0/k_B, p_1/k_B, p_2/k_B$ in units of K.

- MM3_IMPROPER_DIHEDRAL

$$U = \frac{1}{2}p_0\left[1+\cos\left(\phi_{ijkl}\right)\right] + \frac{1}{2}p_1\left[1-\cos\left(2\phi_{ijkl}\right)\right] + \frac{1}{2}p_2\left[1+\cos\left(3\phi_{ijkl}\right)\right] \tag{3.63}$$

  3 arguments: p_0, p_1, p_2 in units of kcal/mol.

- CFF_IMPROPER_DIHEDRAL

$$U = p_0\left[1-\cos\left(\phi_{ijkl}\right)\right] + p_1\left[1-\cos\left(2\phi_{ijkl}\right)\right] + p_2\left[1-\cos\left(3\phi_{ijkl}\right)\right] \tag{3.64}$$

  3 arguments: $p_0/k_B, p_1/k_B, p_2/k_B$ in units of K.

- CFF_IMPROPER_DIHEDRAL2

$$U = p_0\left[1+\cos\left(\phi_{ijkl}\right)\right] + p_1\left[1+\cos\left(2\phi_{ijkl}\right)\right] + p_2\left[1+\cos\left(3\phi_{ijkl}\right)\right] \tag{3.65}$$

  3 arguments: $p_0/k_B, p_1/k_B, p_2/k_B$ in units of K.

- SIX_COSINE_IMPROPER_DIHEDRAL
  The Ryckaert-Bellemans potentials is often used for alkanes, the use implies exclusion of VDW-interactions between the first and last atoms of the dihedral, and $\phi' = \phi - \pi$ is defined according to the polymer convention $\phi'(trans) = 0$.

$$U = \sum_{n=0}^{5} p_n \cos^n\left(\phi'_{ijkl}\right) \tag{3.66}$$

$$= p_0 + p_1 \cos\left(\phi'_{ijkl}\right) + p_2 \cos^2\left(\phi'_{ijkl}\right) + p_3 \cos^3\left(\phi'_{ijkl}\right) p_4 \cos^4\left(\phi'_{ijkl}\right) + p_5 \cos^5\left(\phi'_{ijkl}\right) \tag{3.67}$$

  6 arguments: $p_0/k_B, \ldots, p_5/k_B$ in units of K. Rewritten in terms of ϕ the potential reads

$$U = p_0 - p_1 \cos\left(\phi_{ijkl}\right) + p_2 \cos^2\left(\phi_{ijkl}\right) - p_3 \cos^3\left(\phi_{ijkl}\right) + p_4 \cos^4\left(\phi_{ijkl}\right) - p_5 \cos^5\left(\phi_{ijkl}\right) \tag{3.68}$$

- TRAPPE_IMPROPER_DIHEDRAL

$$U = p_0 + p_1\left[1 + \cos\left(\phi_{ijkl}\right)\right] + p_2\left[1 - \cos\left(2\phi_{ijkl}\right)\right] + p_3\left[1 + \cos\left(3\phi_{ijkl}\right)\right] \tag{3.69}$$

  4 arguments: $p_0/k_B, p_1/k_B, p_2/k_B, p_3/k_B$ in units of K.

- CVFF_IMPROPER_DIHEDRAL

$$U = p_0\left[1 + \cos\left(p_1\phi_{ijkl} - p_2\right)\right] \tag{3.70}$$

  3 arguments: p_0/k_B in units of K, p_1 dimensionless, and p_2 in degrees.

- OPLS_IMPROPER_DIHEDRAL

$$U = \frac{1}{2}p_0 + \frac{1}{2}p_1\left[1 + \cos\left(\phi_{ijkl}\right)\right] + \frac{1}{2}p_2\left[1 - \cos\left(2\phi_{ijkl}\right)\right] + \frac{1}{2}p_3\left[1 + \cos\left(3\phi_{ijkl}\right)\right] \tag{3.71}$$

  4 arguments: $p_0/k_B, p_1/k_B, p_2/k_B, p_3/k_B$ in units of K.

- FOURIER_SERIES_IMPROPER_DIHEDRAL
  The general form of a Fourier expansion is:

$$U = \sum_{n=1}^{6}\left[a_n \cos\left(n\phi\right) + b_n \sin\left(n\phi\right)\right] \tag{3.72}$$

  This form uses equilibrium angles of 0 for $n = 1, 3, 5$ and 180 for $n = 2, 4, 6$

$$\begin{aligned} U =& \frac{1}{2}p_0\left[1 + \cos\phi\right] + \frac{1}{2}p_1\left[1 - \cos\left(2\phi\right)\right] + \frac{1}{2}p_2\left[1 + \cos\left(3\phi\right)\right] + \\ & \frac{1}{2}p_3\left[1 - \cos\left(4\phi\right)\right] + \frac{1}{2}p_4\left[1 + \cos\left(5\phi\right)\right] + \frac{1}{2}p_5\left[1 - \cos\left(6\phi\right)\right] \end{aligned} \tag{3.73}$$

  6 arguments: $p_0/k_B, p_1/k_B, p_2/k_B, p_3/k_B, p_4/k_B, p_5/k_B$ in units of K.

- FOURIER_SERIES_IMPROPER_DIHEDRAL_2
  The general form of a Fourier expansion is:

$$U = \sum_{n=1}^{6}\left[a_n \cos\left(n\phi\right) + b_n \sin\left(n\phi\right)\right] \tag{3.74}$$

  This form uses equilibrium angles of 0 for $n = 1, 3, 4, 5, 6$ and 180 for $n = 2$

$$\begin{aligned} U =& \frac{1}{2}p_0\left[1 + \cos\phi\right] + \frac{1}{2}p_1\left[1 - \cos\left(2\phi\right)\right] + \frac{1}{2}p_2\left[1 + \cos\left(3\phi\right)\right] + \\ & \frac{1}{2}p_3\left[1 + \cos\left(4\phi\right)\right] + \frac{1}{2}p_4\left[1 + \cos\left(5\phi\right)\right] + \frac{1}{2}p_5\left[1 + \cos\left(6\phi\right)\right] \end{aligned} \tag{3.75}$$

  6 arguments: $p_0/k_B, p_1/k_B, p_2/k_B, p_3/k_B, p_4/k_B, p_5/k_B$ in units of K.

- FIXED_IMPROPER_DIHEDRAL
  Use for improper-dihedral-angle constraint using the 'SHAKE' and 'RATTLE'-algorithm. Applies to Molecular Dynamics and minimization. Does not work (yet) in Monte-Carlo.

3.3 Non-bonded potentials

3.3.1 Van der Waals potentials

The general expression for Van der Waals potentials when using a cutoff distance is

$$U_{ij}^{\text{VDW}} = \begin{cases} U_{ij}\left(r_{ij}\right) & \text{if } r_{ij} \leq r_c \\ 0 & \text{otherwise} \end{cases} \tag{3.76}$$

- NONE

$$U = 0 \tag{3.77}$$

  zero parameters.

- LENNARD_JONES
  LENNARD_JONES_SMOOTHED3
  LENNARD_JONES_SMOOTHED5

$$U = 4p_0\left[\left(\frac{p_1}{r}\right)^{12} - \left(\frac{p_1}{r}\right)^{6}\right] \tag{3.78}$$

  2 parameters: p_0/k_B in units of K, and p_1 in Å.

- FEYNMAN_HIBBS_LENNARD_JONES
  FEYNMAN_HIBBS_LENNARD_JONES_SMOOTHED3
  FEYNMAN_HIBBS_LENNARD_JONES_SMOOTHED5

$$U = 4p_0\left[\left(\frac{p_1}{r}\right)^{12} - \left(\frac{p_1}{r}\right)^{6}\right] + \frac{\hbar^2}{24p_2k_BT}4p_0\left[132\left(\frac{p_1}{r}\right)^{12} - 30\left(\frac{p_1}{r}\right)^{6}\right]\frac{1}{r^2} \tag{3.79}$$

  3 parameters: p_0/k_B in units of K, p_1 in Å, and p_2 is the reduced mass in unified atomic mass units.

- FEYNMAN_HIBBS2_LENNARD_JONES
  FEYNMAN_HIBBS_LENNARD_JONES2_SMOOTHED3
  FEYNMAN_HIBBS_LENNARD_JONES2_SMOOTHED5

$$U = 4p_0\left[\left(\frac{p_1}{r}\right)^{12} - \left(\frac{p_1}{r}\right)^{6}\right] + 4p_0\left[132\left(\frac{p_1}{r}\right)^{12} - 30\left(\frac{p_1}{r}\right)^{6}\right]\frac{p_2}{r^2} \tag{3.80}$$

  3 parameters: p_0/k_B in units of K, p_1 in Å, and p_2 in units of $\mathring{A}^2$.

- LENNARD_JONES_SHIFTED_FORCE

$$U = 4p_0\left\{\left[\left(\frac{p_1}{r}\right)^{12} - \left(\frac{p_1}{r}\right)^{6}\right] - \left[\left(\frac{p_1}{r_c}\right)^{12} - \left(\frac{p_1}{r_c}\right)^{6}\right] + \left[12\left(\frac{p_1}{r_c}\right)^{12} - 6\left(\frac{p_1}{r_c}\right)^{6}\right]\frac{(r-r_c)}{r_c}\right\} \tag{3.81}$$

  2 parameters: p_0/k_B in units of K, and p_1 in Å.

- LENNARD_JONES_SHIFTED_FORCE2

$$4p_0\left\{\left[\left(\frac{p_1}{r}\right)^{12} - \left(\frac{p_1}{r}\right)^{6}\right] + \left[6\left(\frac{p_1}{r_c}\right)^{12} - 3\left(\frac{p_1}{r_c}\right)^{6}\right]\frac{r^2}{r_c^2} + 7\left(\frac{p_1}{r_c}\right)^{12} + 4\left(\frac{p_1}{r_c}\right)^{6}\right\} \tag{3.82}$$

  2 parameters: p_0/k_B in units of K, and p_1 in Å.

- POTENTIAL_12_6
  POTENTIAL_12_6_SMOOTHED3
  POTENTIAL_12_6_SMOOTHED5

$$U = \frac{p_0}{r^{12}} - \frac{p_1}{r^6} \tag{3.83}$$

2 parameters: p_0/k_B in units of K Å^{12}, and p_1/k_B in units of K Å^6.

- POTENTIAL_12_6_2_0
  POTENTIAL_12_6_2_0_SMOOTHED3
  POTENTIAL_12_6_2_0_SMOOTHED5

$$U = \frac{p_0}{r^{12}} + \frac{p_1}{r^6} + \frac{p_2}{r^2} + p_3 \tag{3.84}$$

4 parameters: p_0/k_B in units of K Å^{12}, p_1/k_B in units of K Å^6, p_2/k_B in units of K Å^2, and p_3 in units of K.

- MORSE
  MORSE_SMOOTHED3
  MORSE_SMOOTHED5

$$U = p_0 \left[(1 - e^{-p_1 * (r - p_2)})^2 - 1 \right] \tag{3.85}$$

3 parameters: p_0/k_B in units of K, p_1 in units of $\AA^{-1}$ and p_2 in units of Å.

- MORSE2
  MORSE2_SMOOTHED3
  MORSE2_SMOOTHED5

$$U = p_0 \left[e^{p_1 * (1 - r/p_2)} - 2 e^{(p_1/2) * (1 - r/p_2)} \right] \tag{3.86}$$

3 parameters: p_0/k_B in units of K, p_1 in units of $\AA^{-1}$ and p_2 in units of Å.

- MORSE3
  MORSE3_SMOOTHED3
  MORSE3_SMOOTHED5

$$U = p_0 \left[\left(1 - e^{\left(\frac{-\ln 2}{2^{1/6} - 1} \right) \left(\frac{r}{p_2} - 2^{1/6} \right)} \right)^2 - 1 \right] \tag{3.87}$$

2 parameters: p_0/k_B in units of K p_2 in units of Å. This form of the Morse potential resembles the Lennard-Jones potential.

- CFF_9_6
  CFF_9_6_SMOOTHED3
  CFF_9_6_SMOOTHED5

$$U = \frac{p_0}{r^9} - \frac{p_1}{r^6} \tag{3.88}$$

2 parameters: p_0/k_B in units of K Å^9, and p_1/k_B in units of K Å^6.

- CFF_EPS_SIGMA
  CFF_EPS_SIGMA_SMOOTHED3
  CFF_EPS_SIGMA_SMOOTHED5

$$U_{ij} = p_0 \left[2 \left(\frac{p_1}{r} \right)^9 - 3 \left(\frac{p_1}{r} \right)^6 \right] \tag{3.89}$$

2 parameters: p_0/k_B in units of K, and p_1 in Å.

- BUCKINGHAM
  BUCKINGHAM_SMOOTHED3
  BUCKINGHAM_SMOOTHED5

$$U = p_0 e^{-p_1 r} - \frac{p_2}{r^6} \tag{3.90}$$

3 parameters: p_0/k_B in units of K, p_1 in units of Å^{-1}, and p_2 in K Å^6. Warning: in literature sometimes $\rho = \frac{1}{p_1}$ is given, ρ is usually around 0.3-0.4 Å, p_1 is usually around 2-4 Å^{-1}.

- BUCKINGHAM2
  BUCKINGHAM2_SMOOTHED3
  BUCKINGHAM2_SMOOTHED5

$$U = \begin{cases} 10^{10} & r < p_3 \\ p_0 e^{-p_1 r} - \frac{p_2}{r^6} & \text{otherwise} \end{cases} \tag{3.91}$$

4 parameters: p_0/k_B in units of K, p_1 in units of Å^{-1}, p_2 in K Å^6, and p_3 in [Å]. Warning: in literature sometimes $\rho = \frac{1}{p_1}$ is given, ρ is usually around 0.3-0.4 Å, p_1 is usually around 2-4 Å^{-1}.

- MM3_VDW
  MM3_VDW_SMOOTHED3
  MM3_VDW_SMOOTHED5

$$U_{ij} = \begin{cases} \sqrt{p_0^i p_0^j}\left[1.84 \times 10^5 e^{-\frac{12}{P}} - 2.25\, P^6\right] & \text{if } P \geq 3.02 \\ \sqrt{p_0^i p_0^j} 192.27 P^2 & \text{if } P < 3.02 \end{cases} \tag{3.92}$$

with $P = \frac{p_1^i + p_1^j}{r_{ij}}$ and where p_1^i and p_1^j are the VDW radii of atoms i and j, and r_{ij} the separation distance in Å between atoms i and j.
2 arguments: p_0 in units of kcal/mol, p_1 in units of Å.

- MATSUOKA_CLEMENTI_YOSHIMINE
  MATSUOKA_CLEMENTI_YOSHIMINE_SMOOTHED3
  MATSUOKA_CLEMENTI_YOSHIMINE_SMOOTHED5

$$U = p_0 e^{-p_1 r_{ij}} + p_2 e^{-p_3 r_{ij}} \tag{3.93}$$

4 arguments: p_0/k_B in units of K, p_1 in units of Å^{-1}, p_2/k_B in units of K, and p_3 in units of Å^{-1}.

- GENERIC
  GENERIC_SMOOTHED3
  GENERIC_SMOOTHED5

$$U = p_0 e^{-p_1 r} - \frac{p_2}{r^4} - \frac{p_3}{r^6} - \frac{p_4}{r^8} - \frac{p_5}{r^{10}} \tag{3.94}$$

6 arguments: p_0/k_B in units of K, p_1 in units of Å^{-1}, p_2/k_B in units of K Å^4, p_3/k_B in units of K Å^6, p_4/k_B in units of K Å^8, and p_5/k_B in units of K Å^{10}.

- PELLENQ_NICHOLSON
  PELLENQ_NICHOLSON_SMOOTHED3
  PELLENQ_NICHOLSON_SMOOTHED5

$$U = p_0 e^{-p_1 r} - f_6 \frac{p_2}{r^6} - f_8 \frac{p_3}{r^8} - f_{10} \frac{p_4}{r^{10}} \tag{3.95}$$

with

$$f_{2n} = 1 - \sum_{k=0}^{2n} \frac{(p_1 r_{ij})^k}{k!} e^{-p_1 r_{ij}} \tag{3.96}$$

5 arguments: p_0/k_B in units of K, p_1 in units of Å^{-1}, p_2/k_B in units of K Å^6, p_3/k_B in units of K Å^8, and p_4/k_B in units of K Å^{10}.

- HYDRATED_ION_WATER
  HYDRATED_ION_WATER_SMOOTHED3
  HYDRATED_ION_WATER_SMOOTHED5

$$U = p_0 e^{-p_1 r} - \frac{p_2}{r^4} - \frac{p_3}{r^6} - \frac{p_4}{r^{12}} \tag{3.97}$$

5 arguments: p_0/k_B in units of K, p_1 in units of Å^{-1}, p_2/k_B in units of K Å^4, p_3/k_B in units of K Å^6, and p_4/k_B in units of K Å^{12}.

- MIE
  MIE_SMOOTHED3
  MIE_SMOOTHED5
  The Mie-potential [**?**]

$$U = \left(\frac{p_0}{r^{p_1}} - \frac{p_2}{r^{p_3}}\right) \tag{3.98}$$

4 arguments: p_0/k_B in units of K Å^{p_1}, p_1 dimensionless, p_2/k_B in units of K Å^{p_3}, and p_3 dimensionless.

- BORN_HUGGINS_MEYER
  BORN_HUGGINS_MEYER_SMOOTHED3
  BORN_HUGGINS_MEYER_SMOOTHED5

$$U_{ij} = p_0 e^{p_1(p_2 - r_{ij})} - \frac{p_3}{r_{ij}^6} - \frac{p_4}{r_{ij}^8} \tag{3.99}$$

5 arguments: p_0/k_B in units of K, p_1 dimensionless, p_2 in units of Å, p_3/k_B in units of K Å^6, and p_4/k_B in units of K Å^8.

- HYDROGEN
  HYDROGEN_SMOOTHED3
  HYDROGEN_SMOOTHED5

$$U = \frac{p_0}{r^{12}} - \frac{p_1}{r^{10}} \tag{3.100}$$

2 arguments: p_0/k_B in units of K Å^{12}, and p_1/k_B in units of K Å^{10}.

3.3.2 Tail corrections

energy

$$U^{\text{Tail}} = \frac{2\pi}{V} \sum_a \sum_b N_a N_b \left[\int_{r_c}^{\infty} r^2 U(r)\, dr\right] \tag{3.101}$$

potential	$\int_{r_c}^{\infty} r^2 U(r)$
LENNARD_JONES	$\frac{4}{3} p_0 p_1^3 \left[\frac{1}{3}\left(\frac{p_1}{r}\right)^9 - \left(\frac{p_1}{r}\right)^3\right]$
LENNARD_JONES_SHIFTED_FORCE	-

pressure

$$P^{\text{Tail}} = -\sum_a \sum_b \frac{2\pi}{3V} N_a N_b \left[\int_{r_c}^{\infty} r^2\, r \frac{\partial U(r)}{\partial r}\, dr\right] \tag{3.102}$$

$$= \sum_a \sum_b \left(\frac{2\pi}{3V} r_c^3 N_a N_b U(r_c) + U^{\text{Tail}}\right) \tag{3.103}$$

chemical potential

$$\beta \mu^{\text{Tail}} = 2U^{\text{Tail}} \tag{3.104}$$

3.3.3 Electrostatics

Charge-charge interaction

- Ewald
  The potential energy for a system of charges in a periodic system can be written as

$$U = U^{\text{real}} + U^{\text{rec}} \tag{3.105}$$

  where

$$\begin{aligned} U^{\text{real}} &= \sum_{i<j} q_i q_j \frac{\text{erfc}\left(\alpha r_{ij}\right)}{r_{ij}} \\ U^{\text{rec}} &= \frac{2\pi}{V} \sum_{\mathbf{k}\neq 0} \frac{1}{k^2} e^{-\frac{k^2}{4\alpha^2}} \left(\left| \sum_{i=1}^{N} q_i \cos\left(\mathbf{k}\cdot\mathbf{r}_i\right) \right|^2 + \left| \sum_{i=1}^{N} q_i \sin\left(\mathbf{k}\cdot\mathbf{r}_i\right) \right|^2 \right) - \sum_i \frac{\alpha}{\sqrt{\pi}} q_i^2 \end{aligned} \tag{3.106}$$

  where q_i and q_j are the charges of particle i and j, respectively, $\mathbf{r}_i$ the position of atom i, V the volume of the cell, α a damping factor, k the wavelength, and 'erfc' the error function complement. The expression gives the *exact* solution for charges in a periodic system up to arbitrary precision. One part is computed in 'real' space, and the long-range part is more conveniently computed in Fourier space.

- CoulombTruncated

$$U = \begin{cases} \sum_{i<j} \frac{1}{4\pi\epsilon} \frac{q_i q_j}{r_{ij}} & \text{if } r_{ij} \leq r_c \\ 0 & \text{otherwise} \end{cases} \tag{3.107}$$

- CoulombShifted

$$U = \begin{cases} \sum_{i<j} \frac{q_i q_j}{4\pi\epsilon} \left(\frac{1}{r_{ij}} - \frac{1}{r_c}\right) & \text{if } r_{ij} \leq r_c \\ 0 & \text{otherwise} \end{cases} \tag{3.108}$$

- CoulombSmoothed
- Wolf

Charge-dipole interaction

- Ewald
- CoulombTruncated

$$U = \begin{cases} \sum_{i,j} \frac{1}{4\pi\epsilon} \frac{-q_i}{r_{ij}^2} \left(\mu_j \cdot \mathbf{r}_{ij}\right) & \text{if } r_{ij} \leq r_c \\ 0 & \text{otherwise} \end{cases} \tag{3.109}$$

Dipole-dipole interaction

- Ewald
- CoulombTruncated

$$U = \begin{cases} \sum_{i,j} \frac{1}{4\pi\epsilon} \frac{1}{r_{ij}^3} \left[\mu_i \cdot \mu_j - 3 \frac{\left(\mu_i \cdot \mathbf{r}_{ij}\right)\left(\mathbf{r}_{ij} \cdot \mu_j\right)}{r_{ij}^2} \right] & \text{if } r_{ij} \leq r_c \\ 0 & \text{otherwise} \end{cases} \tag{3.110}$$

3.4 Bonded potentials cross terms

3.4.1 Bond-bond potential

- CFF_BOND_BOND_CROSS,CVFF_BOND_BOND_CROSS

$$U = p_0 \left(r - p_1\right) \left(r' - p_2\right) \tag{3.111}$$

3 arguments: p_0/k_B in units of K/Å^2, p_0 and p_1 in Å.

3.4.2 Bond-bend potential

- CFF_BOND_BEND_CROSS,CVFF_BOND_BEND_CROSS

$$U = \left(\theta - p_0\right) \left[p_1 \left(r - p_2\right) + p_3 \left(r' - p_4\right)\right] \tag{3.112}$$

5 arguments: p_0 in degrees, p_1/k_B in units of K/Å/rad, p_2 in Å, p_3/k_B in units of K/Å/rad, p_4 in Å.

- MM3_BOND_BEND_CROSS

$$U = p_0 \left[\left(r - p_1\right) + \left(r' - p_2\right)\right] \left(\theta - p_3\right) \tag{3.113}$$

4 arguments: p_0 in mdyne/rad, p_1 and p_2 in Å, and p_3 in degrees.

- TRUNCATED_HARMONIC

$$U = \frac{1}{2} p_0 \left(\theta - p_1\right)^2 e^{-\frac{r_{ij}^8 + r_{ik}^8}{p_2^8}} \tag{3.114}$$

3 arguments: p_0/k_B in K/rad^2, p_1 in degrees, and p_2 in units of Å.

- SCREENED_HARMONIC

$$U = \frac{1}{2} p_0 \left(\theta - p_1\right)^2 e^{-\left(\frac{r_{ij}}{p_2} + \frac{r_{ik}}{p_3}\right)} \tag{3.115}$$

4 arguments: p_0 in K/rad^2, p_1 in degrees, p_2 and p_3 in units of Å.

- SCREENED_VESSAL

$$U = \frac{p_0}{8\left(\theta_{ijk} - \pi\right)^2} \left[\left(p_1 - \pi\right)^2 - \left(\theta_{ijk} - \pi\right)^2\right]^2 e^{-\left(\frac{r_{ij}}{p_2} + \frac{r_{ik}}{p_3}\right)} \tag{3.116}$$

4 arguments: p_0 in K/rad^2, p_1 in degrees, p_2 and p_3 in units of Å.

- TRUNCATED_VESSAL

$$U = p_0 \left[\theta_{ijk}^{p_2} \left(\theta_{ijk} - p_1\right)^2 \left(\theta_{ijk} + p_1 - 2\pi\right)^2 - \frac{p_2}{2} \pi^{p_2 - 1} \left(\theta_{ijk} - p_1\right)^2 \left(\pi - p_1\right)^3 e^{-\frac{r_{ij}^8 + r_{ik}^8}{p_3^8}}\right] \tag{3.117}$$

4 arguments: p_0 in K/rad$^{4+p_2}$, p_1 in degrees, p_2 dimensionless, and p_3 in Å.

3.4.3 Bend-bend potential

- CFF_BEND_BEND_CROSS,CVFF_BEND_BEND_CROSS

$$U = p_0 \left(\theta - p_1\right) \left(\theta' - p_2\right) \tag{3.118}$$

3 arguments: p_0 in units of K/rad^2, p_1 and p_2 in units of degrees.

- MM3_BEND_BEND_CROSS

$$U = -p_0 \left(\theta - p_1\right) \left(\theta' - p_2\right) \tag{3.119}$$

3 arguments: p_0 in units of mdyne/rad^2, p_1 and p_2 in units of degrees.

3.4.4 Bond-torsion potential

The bond-torsions potential correlates the torsion $i-j-k-l$ with the central bond $j-k$, or with the two terminating bonds.

- MM3_BOND_TORSION_CROSS
  The MM3 bond-torsion potential correlates the torsion $i-j-k-l$ with the central bond $j-k$
  $$U=\frac{1}{2}p_0\left(r-p_3\right)\left(1+\cos\phi\right)+\frac{1}{2}p_1\left(r-p_3\right)\left(1+\cos 2\phi\right)+\frac{1}{2}p_2\left(r-p_3\right)\left(1+\cos 3\phi\right) \tag{3.120}$$
  4 arguments: p_0,p_1,p_2 in units of kcal/mol, p_3 the reference length of the central bond in Å.

3.4.5 Bend-torsion potential

- CFF_BEND_TORSION_CROSS,CVFF_BEND_TORSION_CROSS
  $$U=p_0\left(\theta-p_1\right)\left(\theta'-p_2\right)\cos\phi \tag{3.121}$$
  3 arguments: p_0 in units of K/rad^3, p_1 and p_2 in units of degrees.
- SMOOTHED_DIHEDRAL
  $$U=p_0\left(1+\cos(p_1\phi_{ijkl}-p_2\right)S\left(\theta_{ijk}\right)S\left(\theta_{jkl}\right) \tag{3.122}$$
  3 arguments: p_0/k_B in units of K/rad^2, p_1 dimensionless, and p_2 in degrees.
- SMOOTHED_THREE_COSINE_DIHEDRAL
  $$U=\left\{\frac{1}{2}p_0\left[1+\cos\left(\phi_{ijkl}\right)\right]+\frac{1}{2}p_1\left[1-\cos\left(2\phi_{ijkl}\right)\right]+\frac{1}{2}p_2\left[1+\cos\left(3\phi_{ijkl}\right)\right]\right\}S\left(\theta_{ijk}\right)S\left(\theta_{jkl}\right) \tag{3.123}$$
  3 arguments: $p_0/k_B,p_1/k_B,p_2/k_B$ in units of K.
- SMOOTHED_CFF_DIHEDRAL
  $$U=\left\{p_0\left[1-\cos\left(\phi_{ijkl}\right)\right]+p_1\left[1-\cos\left(2\phi_{ijkl}\right)\right]+p_2\left[1-\cos\left(3\phi_{ijkl}\right)\right]\right\}S\left(\theta_{ijk}\right)S\left(\theta_{jkl}\right) \tag{3.124}$$
  3 arguments: $p_0/k_B,p_1/k_B,p_2/k_B$ in units of K.
- SMOOTHED_CFF_DIHEDRAL2
  $$U=\left\{p_0\left[1+\cos\left(\phi_{ijkl}\right)\right]+p_1\left[1+\cos\left(2\phi_{ijkl}\right)\right]+p_2\left[1+\cos\left(3\phi_{ijkl}\right)\right]\right\}S\left(\theta_{ijk}\right)S\left(\theta_{jkl}\right) \tag{3.125}$$
  3 arguments: $p_0/k_B,p_1/k_B,p_2/k_B$ in units of K/rad.
- NICHOLAS_DIHEDRAL
  $$U=\left\{\frac{1}{2}p_0\left[1+\cos\left(\phi_{ijkl}\right)\right]+\frac{1}{2}p_1\left[1-\cos\left(2\phi_{ijkl}\right)\right]+\frac{1}{2}p_2\left[1+\cos\left(3\phi_{ijkl}\right)\right]\right\}S\left(\theta_{ijk}\right) \tag{3.126}$$
  3 arguments: $p_0/k_B,p_1/k_B,p_2/k_B$ in units of K/rad.
- SMOOTHED_CFF_BEND_TORSION_CROSS
  $$U=S\left(\theta_1\right)\left[p_0*\left(Theta_1-p_1\right)*\left(\theta_2-p_2\right)\cos(\phi)\right]S\left(\theta_2\right) \tag{3.127}$$
  3 arguments: p_0/k_B in units K/rad^3, p_1 and p_2 in units of degrees.

The smoothing function $S\left(\theta\right)$ is defined as

$$S\left(\theta\right)=\begin{cases}1 & \theta<\theta_{\text{on}}\\ \left(\theta_{\text{off}}-\theta\right)^2\frac{\theta_{\text{off}}+2\theta-3\theta_{\text{on}}}{\left(\theta_{\text{off}}-\theta_{\text{on}}\right)^3} & \theta\geq\theta_{\text{on}}\end{cases} \tag{3.128}$$

with $\theta_{\text{on}}=170°$ and $\theta_{\text{off}}=180°$.

4 Examples

4.1 Introduction

Often the best way of learning a code is to look at various examples. Note these examples are just for that purpose and real simulation runs should be much longer, both in initialization time as well as production run time.

> Tip: VMD is capable of showing pdb-files with several frames. This the way RASPA produces movies. Standard VMD does not show the box itself but some extension scripts have been written. To show the unit cell in VMD you can input into the console:
>
> ```
> draw pbcbox -width 1.0 -style tubes -center unitcell
> ```
>
> make sure the 'pbctools.tcl' and 'pbcbox.tcl' are in the current directory, they are located in the 'utils' directory of RASPA. For NPT simulations the box is properly updated.

The output-files begin with some essential data about the program: the version number, whether a 64-bits or 32-bits executable is run, the used compiler, when the output-file was generated and on which node and system.

```
RASPA 2.0.45
Compiled as a 64-bits application
Compiler: gcc Apple LLVM 12.0.5 (clang-1205.0.22.9)
Compile Date = May 18 2021, Compile Time = 12:54:12

Tue May 18 12:57:51 2021
Simulation started on Tuesday, May 18.
The start time was 12:57 PM.

Cpu data:     x86_64
Cpu Model:    MacPro7,1
Host name:    MacPro.local
OS release:   20.4.0
OS type:      Darwin
OS version:   20E232
```

The files that RASPA uses for input are

- `simulation.input`
  The main required file for RASPA is the `simulation.input` file, which specifies the input setting and details of the simulation.

- `pseudo_atoms.def`, `force_field_mixing_rules.def`
  These files define the atom-types and the force field, respectively. A few example force field files are supplied with RASPA which can be specified by using

  ```
  Forcefield ExampleZeolitesForceField
  ```

  in the `simulation.input` file. RASPA then looks for these files in:

  ```
       ${RASPA_DIR}/share/raspa/forcefield/ExampleZeolitesForceField
  ```

  However, these files can also be placed in the same, local directory as the `run` file.

- Molecule files
  Molecules are specified in the `simulation.input` file, e.g.

  ```
  Component 0 MoleculeName             methane
              MoleculeDefinition       ExampleDefinitions
  ```

  which look for the file `methane.def` in the directory:

  ```
  ${RASPA_DIR}/share/raspa/molecules/ExampleDefinitions
  ```

  Again, these molecule files can be placed in the current directory which will then be read instead.

- Structure files
  A few example structure files are supplied with RASPA which can be specified by using

  ```
  FrameworkName ITQ-29
  ```

  in the `simulation.input` file. RASPA then looks for the file `ITQ-29.cif` in:

  ```
  ${RASPA_DIR}/share/raspa/share/raspa/structures/cif/
  ```

  Blocked pockets for this structure can be specified by

  ```
  Component 0 MoleculeName             methane
              BlockPockets             yes
              BlockPocketsFilename     ITQ-29
              ...
  ```

  in the `simulation.input` file. RASPA then looks for the file `ITQ-29.block` in:

  ```
  ${RASPA_DIR}/share/raspa/share/raspa/structures/block/
  ```

  Again, these molecule files can be placed in the current directory which will then be read instead.

4.2 Basic examples

Example 1: Monte Carlo of methane in a box

A Monte Carlo run of 100 methane molecules in a $30 \times 30 \times 30$ Å box. After 5000 cycles of initialization the production run is started. A movie is written and every 100th configuration is appended to the movie. The movie is stored in 'Movies/System_0', and can be viewed with iRASPA or VMD.

```
SimulationType                MonteCarlo
NumberOfCycles                10000
NumberOfInitializationCycles  5000
PrintEvery                    1000

Forcefield                    ExampleMoleculeForceField

Box 0
BoxLengths 30 30 30
ExternalTemperature 300.0
Movies yes
WriteMoviesEvery 100

Component 0 MoleculeName             methane
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   1.0
            CreateNumberOfMolecules  100
```

In RASPA, the cycle is define as max(20,N) steps, where N is the number of molecules in the system. In every cycle, each of the molecules has on average been used for a Monte Carlo move (accepted or rejected). There is a minimum of 20 steps to avoid that low-density systems or not sampled well. The definition of a cycle is less dependent on the system size. The number of Monte Carlo steps is roughly the number of cycles times the average number of molecules.

The output is written to the 'Output' directory (per system), and the temperature and pressure are appended to all output filenames. In the output file, the simulation writes an important check to the file

```
Energy-drift status
===========================================================================
Adsorbate/Adsorbate energy-drift:                                 1.05012e-10
    Adsorbate/Adsorbate VDW energy-drift:                           1.05012e-10
===========================================================================
Total energy-drift: 1.05012e-10
```

In Monte Carlo, only difference in energies are computed. These differences are continuously added to keep track of the current energies (from which average energies etc. are computed). Obviously, the current energy that is kept track off during the simulation should be equal to a full recalculation of the energies. The difference between the two signals an error. If the drift is higher than say $1e-3$ or $1e-4$ the results of the simulation are in error. This could be due to an error in one of the Monte Carlo moves or because the force field is "wrong" (a typical error is when one forgets to define required potentials).

The performance of Monte Carlo moves is monitored. Translation moves are usually scaled to achieve an acceptance rate of 50%. Here, the move reached its upper limit of 1 Å because of the low density of the system.

```
Performance of the translation move:
======================================
Component 0 [methane]
    total        332905.000000 333233.000000 333862.000000
    succesfull   283926.000000 284388.000000 284917.000000
    accepted   0.852874 0.853421 0.853398
    displacement 1.000000 1.000000 1.000000
```

Averages are computed along with an error bar. The error is computed by dividing the simulation in 5 blocks and calculating the standard deviation. The errors in RASPA are computed as the 95% confidence interval.

```
Total energy:
=============
    Block[ 0]       -18276.83475 [K]
    Block[ 1]       -18329.57756 [K]
    Block[ 2]       -18502.81990 [K]
    Block[ 3]       -18371.38298 [K]
    Block[ 4]       -19216.89509 [K]
    ------------------------------------------------------------------------------
    Average         -18539.50205 [K] +/-         481.43129 [K]
```

Example 2: Monte Carlo of CO2 in a box and N2 in another box (two independent simulations)

RASPA has a build-in structure of being able to simulate several systems at the same time. This has applications in Gibbs-ensembles and (hyper) parallel tempering for example. However, this capability can also be used for independent systems. The first box is $30 \times 30 \times 30$ Å with 90 ° angles, containing 50 N_2 and 25 CO_2 and molecules and moved around by translation, rotation and reinsertion. The second box is monoclinic and of size $25 \times 25 \times 25$ with $\beta = 120°, \alpha = \gamma = 90°$ containing 25 N_2 and 50 CO_2 molecules. The first system is at 300K, the second at 500K.

```
SimulationType                MonteCarlo
NumberOfCycles                10000
NumberOfInitializationCycles  1000
PrintEvery                    100

Forcefield                    ExampleMoleculeForceField

Box 0
BoxLengths 25 25 25
ExternalTemperature 300.0
Movies yes
WriteMoviesEvery 10

Box 1
BoxLengths 30 30 30
BoxAngles 90 120 90
ExternalTemperature 500.0
Movies yes
WriteMoviesEvery 10

Component 0 MoleculeName             N2
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   1.0
            RotationProbability      1.0
            ReinsertionProbability   1.0
            CreateNumberOfMolecules  50 25

Component 1 MoleculeName             CO2
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   1.0
            RotationProbability      1.0
            ReinsertionProbability   1.0
            CreateNumberOfMolecules  25 50
```

One thing to note is that system-dependent statements apply to the *current* box, following 'Box [int]'. The initialization of the systems with molecules is done using the 'CreateNumberOfMolecules' which applies similarly to the *current* component specified using 'component [int]'. The list of integers represent the initial amount of molecules for each system. Note that when the 'BoxAngles' line is omitted, $\alpha = \beta = \gamma = 90°$ is assumed as the default.

Note that we specify only relative probabilities of MC particle moves. They will be correctly rescaled as shown in the output-file:

```
Particle Moves:
    ProbabilityTranslationMove:                 33.333333
        TranslationDirection:     XYZ
    Percentage of rotation moves:                    33.333333
    Percentage of reinsertion moves:                 33.333333
```

At every MC-step, each move will be randomly selected with 1/3 probability.

Example 3: Monte Carlo of a binary mixture in a box

A Monte Carlo run of 50 propane and 50 butane molecules in a $30 \times 30 \times 30$ Å box. The MC moves are translation, rotation, and full reinsertion. After 1000 steps of initialization the production run is started. A movie is written and every 10th configuration is appended to the movie. The movie is stored in 'Movies/System_0', and can be viewed with iRASPA or VMD.

```
SimulationType                MonteCarlo
NumberOfCycles                10000
NumberOfInitializationCycles  2000
PrintEvery                    100

Forcefield                    ExampleMoleculeForceField

Box 0
BoxLengths 30 30 30
ExternalTemperature 300.0
Movies yes
WriteMoviesEvery 10
```

```
Component 0 MoleculeName            propane
            MoleculeDefinition      ExampleDefinitions
            TranslationProbability  1.0
            RotationProbability     1.0
            ReinsertionProbability  1.0
            CreateNumberOfMolecules 50

Component 1 MoleculeName            butane
            MoleculeDefinition      ExampleDefinitions
            TranslationProbability  1.0
            RotationProbability     1.0
            ReinsertionProbability  1.0
            CreateNumberOfMolecules 50
```

The propane and butane molecules are modeled as flexible united-atom beads. The intra-molecular force field contains bond, bend, and torsion terms

```
Average Adsorbate Bond stretch energy:
=====================================
    Block[ 0]       37377.65243 [K]
    Block[ 1]       37822.77336 [K]
    Block[ 2]       37216.91024 [K]
    Block[ 3]       37033.87935 [K]
    Block[ 4]       37658.50987 [K]
    ------------------------------------------------------------------------------
    Average         37421.94505 [K] +/-         398.05476 [K]

Average Adsorbate Bend angle energy:
====================================
    Block[ 0]       23136.71656 [K]
    Block[ 1]       22692.37638 [K]
    Block[ 2]       22046.60765 [K]
    Block[ 3]       22185.01877 [K]
    Block[ 4]       21419.84764 [K]
    ------------------------------------------------------------------------------
    Average         22296.11340 [K] +/-         810.78089 [K]

Average Adsorbate Torsion energy:
=================================
    Block[ 0]       13601.19894 [K]
    Block[ 1]       13749.89405 [K]
    Block[ 2]       13355.15893 [K]
    Block[ 3]       13339.11856 [K]
    Block[ 4]       13049.12955 [K]
    ------------------------------------------------------------------------------
    Average         13418.90000 [K] +/-         334.24478 [K]
```

The translation and rotation moves leave the internal structure invariant. The reinsertion-move regrows the molecule at a random position with a new internal structure.

```
Performance of the Reinsertion move:
====================================
Component [propane] total tried: 333613.000000 succesfull growth: 333407.000000 (99.938252 [%]) accepted: 85599.000000 (25.658173 [%])
Component [butane] total tried: 332088.000000 succesfull growth: 331383.000000 (99.787707 [%]) accepted: 46465.000000 (13.991773 [%])
```

The acceptance percentages are here high enough. But for dense systems, the insertion acceptance ratios become too small. In these cases, other moves (like partial-reinsertion or MC/MD hybrid moves) become essential to properly sample the internal structure of molecules.

Example 4: Monte Carlo of CO_2 and N_2 in two independent boxes

An example of a binary mixture of CO_2 and N_2 in two independent boxes. Box one contains 100 CO_2 molecules at 300 Kelvin, box two (monoclinic shape) contains 100 N_2 molecules at 500 Kelvin. The movies for box one are appended every 10 cycles, the movie for box two every 5 cycles. Three types of Monte Carlo moves are used: translation, rotation, and reinsertion.

```
SimulationType                MonteCarlo
NumberOfCycles                10000
NumberOfInitializationCycles  1000
PrintEvery                    100

Forcefield                    ExampleMoleculeForceField

Box 0
BoxLengths 25 25 25
ExternalTemperature 300.0
Movies yes
WriteMoviesEvery 10

Box 1
BoxLengths 30 30 30
BoxAngles 90 120 90
ExternalTemperature 500.0
Movies yes
WriteMoviesEvery 5

Component 0 MoleculeName            CO2
```

```
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   1.0
            RotationProbability      1.0
            ReinsertionProbability   1.0
            CreateNumberOfMolecules  100 0

Component 1 MoleculeName             N2
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   1.0
            RotationProbability      1.0
            ReinsertionProbability   1.0
            CreateNumberOfMolecules  0 100
```

Example 5: Molecular dynamics of methane in a box measuring the mean-square displacement

A molecular dynamics run of 100 methane molecules in a $25 \times 25 \times 25$ Å box at 300 K. The simulations starts with 1000 InitializationSteps using Monte Carlo, the only MC moves are translation and reinsertion. After 1000 steps of initialization the equilibration run is started. Here, the atoms are assigned a velocities, and during the equilibration run the distribution should attain the Maxwell-Boltzmann distribution. After the initialization and equilibration runs, the production is started. The mean-square displacement is measured and written to 'MSDOrderN/System_0' for both self-and collective diffusion (the slope of the mean square displacement is related to the diffusion coefficients). They can be plotted with 'gnuplot'. In contrast to Monte Carlo where the ensemble basically follows from the used MC moves, the ensemble for molecular dynamics needs to be explicitly specified using the 'Ensemble' keyword.

```
SimulationType                MolecularDynamics
NumberOfCycles                1000000
NumberOfInitializationCycles  1000
NumberOfEquilibrationCycles   10000
PrintEvery                    100000
PrintPropertiesEvery          100000

Ensemble                      NVT
TimeStep                      0.0005

Forcefield                    ExampleMoleculeForceField

Box 0
BoxLengths 25 25 25
ExternalTemperature 300.0
ComputeMSD yes
PrintMSDEvery 5000

Component 0 MoleculeName             methane
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   1.0
            ReinsertionProbability   1.0
            CreateNumberOfMolecules  100
```

In MD, it is important to have good energy-conservation. This is monitored

```
Conserved energy:       15808.0157258017 Energy drifts:  0.0000256196          0.0000100426
```

The first number is the conserved quantity, the second the current relative energy drift, and the last number is the average energy drift. The latter two numbers need to be small, usually smaller than say 10^{-3}. The NVT ensemble is achieved using a Nose-Hoover thermostat that maintains the system at the desired temperature of 300K.

```
Average temperature:
===================
    Block[ 0]         300.23475 [K]
    Block[ 1]         300.81631 [K]
    Block[ 2]         300.01936 [K]
    Block[ 3]         299.88634 [K]
    Block[ 4]         300.43421 [K]
    ------------------------------------------------------------------------------
    Average           300.27819 [K] +/-           0.45459 [K]
```

Example 6: Enthalpy of adsorption of methane in MFI at infinite dilution

The affinity of a molecule with the framework can be expressed as the binding energy, or more general, as the enthalpy of adsorption at infinite dilution ΔH [1]:

$$\Delta H = \langle U_{hg} \rangle - \langle U_h \rangle - \langle U_g \rangle - RT \tag{4.1}$$

where $\langle U_{hg}\rangle$, $\langle U_h\rangle$, and $\langle U_g\rangle$ are the average energy of the guest molecule inside the host-framework, the average energy of the host-framework, and the average energy of a single guest-molecule in the gas phase, respectively. The term RT is the enthalpy per particle of the ideal bulk phase. It accounts for the work to push the gas adsorbates into the fluid phase when it desorbs.
We can measure the guest-host energy of a single methane molecule in MFI with the following input

```
SimulationType                MonteCarlo
NumberOfCycles                25000
NumberOfInitializationCycles  5000
PrintEvery                    1000

Forcefield                    ExampleZeolitesForceField
RemoveAtomNumberCodeFromLabel yes

Framework 0
FrameworkName MFI_SI
UnitCells 2 2 2
HeliumVoidFraction 0.29
ExternalTemperature 300.0
ExternalPressure 0.0

Component 0 MoleculeName             methane
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   0.5
            ReinsertionProbability   0.5
            CreateNumberOfMolecules  1
```

The $\langle U_{hg}\rangle$ energy is the average total energy of the system for the simulation with a single adsorbate

```
Total energy:
=============
    Block[ 0]        -1983.98783 [K]
    Block[ 1]        -1983.81557 [K]
    Block[ 2]        -1988.29391 [K]
    Block[ 3]        -1987.85152 [K]
    Block[ 4]        -1991.39101 [K]
    ------------------------------------------------------------------------------
    Average          -1987.06797 [K] +/-           3.96837 [K]
```

Since the framework is rigid ($\langle U_h\rangle = 0$) and the molecule has no internal structure ($\langle U_g\rangle = 0$), the enthalpy of adsorption at infinite dilution is $\Delta H = (-1987.06797 - 300) * 8.314462618/1000 = 19.0$ kJ/mol.

Example 6: Adsorption isotherm of methane in MFI

Adsorption isotherms can be easily obtained by specifying a list of (increasing) pressures which will be subsequently run. If no `FugacityCoefficient` keyword is specified these pressure are converted to fugacity using the Peng-Robinson equation of state. Important: it is essential to specify the 'ideal gas Rosenbluth weight' for a component. This value needs to be computed separately and depends only on temperature (see auxiliary examples). This value is the reference state of the ideal gas. It is convenient to specify it in advance, otherwise the correct pressure needs to deduced afterwards and is different from the specified input. For mixtures this becomes cumbersome when the ideal gas Rosenbluth weight of the components is different. In this example, $2 \times 2 \times 2$ unit cells are required to meet the required that all of the perpendicular cell lengths are larger than twice the cutoff distance. The default cutoff of 12 Å means the perpendicular lengths should be larger than 24 Å.

For simulation with frameworks, the keyword 'Framework' is used, instead of 'Box' in the previous examples. The line `FrameworkName MFI_SI` will look for the file `MFI_SI.cif` in

```
${RASPA_DIR}/share/raspa/share/raspa/structures/cif/
```

The MFI-file contains atoms `Si1-Si12`, and `O1-O24`. Using `RemoveAtomNumberCodeFromLabel yes`, these will be relabeled as `Si` and `O`.

This example uses a generic zeolite force field based on TraPPE-zeo [2], while the force field for the methane adsorbate is taken from Martin et al. [3].

```
Forcefield ExampleZeolitesForceField
```

in the `simulation.input` file. RASPA then looks for these files in:

```
${RASPA_DIR}/share/raspa/forcefield/ExampleZeolitesForceField
```

The adsorption can be computed in grand-canonical ensemble. To swap particles in and out of the system at constant fugacity, use the swap-move

```
SwapProbability           1.0
```

The input for this adsorption example is

```
SimulationType               MonteCarlo
NumberOfCycles               25000
NumberOfInitializationCycles  2000
PrintEvery                   1000

Forcefield                   ExampleZeolitesForceField
RemoveAtomNumberCodeFromLabel yes

Framework 0
FrameworkName MFI_SI
UnitCells 2 2 2
HeliumVoidFraction 0.29
ExternalTemperature 300.0
ExternalPressure 1e4 1e5

ComputeNumberOfMoleculesHistogram yes
WriteNumberOfMoleculesHistogramEvery 5000
NumberOfMoleculesHistogramSize 1100
NumberOfMoleculesRange 80

ComputeEnergyHistogram yes
WriteEnergyHistogramEvery 5000
EnergyHistogramSize 400
EnergyHistogramLowerLimit -110000
EnergyHistogramUpperLimit -20000

Component 0 MoleculeName             methane
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   0.5
            ReinsertionProbability   0.5
            SwapProbability          1.0
            CreateNumberOfMolecules  0
```

The example does not specify an explicit `FugacityCoefficient` for the component, so the `ExternalPressure` is used as a pressure and converted to fugacity. For small pressures, pressure and fugacity are almost the same, as indicated by the computed fugacity coefficients being close to unity.

```
Partial pressure:    100000.00000000000000 [Pa]
                        750.00000000000000 [Torr]
                          1.00000000000000 [bar]
                          0.98692326671601 [atm]

Fugacity coefficient:       0.9978285867 [-]

Partial fugacity:     99782.85867089460953 [Pa]
                        748.37144003170954 [Torr]
                          0.99782858670895 [bar]
                          0.98478024841742 [atm]
```

One could do the simulation at a 100000 Pa fugacity by specifying `FugacityCoefficient 1.0` for the component. Also, if a better value for the fugacity coefficient is know (i.e. better than computed by the Peng-Robinson EOS), then it can be manually set in this way.
The energy histogram and the histogram of the number of molecules are computed during the run. The can be found in directories 'EnergyHistograms' and 'NumberOfMoleculesHistograms', respectively.
The output-file shows the performance of the various Monte Carlo moves. For adsorption, a good check is that the acceptance ratio of the 'swap addition' and the 'swap deletion' should be close.

```
Performance of the swap addition move:
======================================
Component [methane] total tried: 125026.000000 succesfull growth: 112028.000000 (89.603762 [%]) accepted: 41478.000000 (33.175499 [%])

Performance of the swap deletion move:
======================================
Component [methane] total tried: 125202.000000 succesfull growth: 117708.000000 (94.014473 [%]) accepted: 41476.000000 (33.127266 [%])

Performance of the Reinsertion move:
====================================
Component [methane] total tried: 117282.000000 succesfull growth: 105024.000000 (89.548268 [%]) accepted: 26993.000000 (23.015467 [%])
```

The output-file shows information on the structure:

```
Number of framework atoms: 2304
Number of framework atoms in the unit cell: 288
Framework Mass: 46144.748974602669 [g/mol]
Framework Density:  1796.342406023652 [kg/m^3]    0.5566867411506 [cm^3/g]
Helium void fraction:    0.29000000
Available pore volume: 12370.29828530 [A^3]    0.16143915 [cm^3/g]
Conversion factor from molecule/unit cell -> kmol/m^3: 0.0389284, kmol/m^3 accesible pore volume: 0.134236
```

We have input the helium void fraction (for MFI, about 0.29) in advance. This value has to be computed separately first (see Auxiliary examples). This has several advantages. First, the correct available pore volume can be computed: 0.16143915 cm^3/g. Secondly, it allows for automatic computation of excess adsorption. At high pressures and temperatures the excess adsorption can be substantially lower than absolute adsorption.
Adsorption results are displayed in various units for both absolute and excess adsorption.

```
Component 0 [methane]
-------------------------------------------------------------
    Block[ 0] 2.77660           [-]
    Block[ 1] 2.77920           [-]
    Block[ 2] 2.70020           [-]
    Block[ 3] 2.88600           [-]
    Block[ 4] 2.78400           [-]
    ------------------------------------------------------------------------------
    Average loading absolute                              2.7852000000 +/-       0.0821082425 [-]
    Average loading absolute [molecules/unit cell]        0.3481500000 +/-       0.0102635303 [-]
    Average loading absolute [mol/kg framework]                  0.0603578969 +/-       0.0017793626 [-]
    Average loading absolute [milligram/gram framework]          0.9682891463 +/-       0.0285453540 [-]
    Average loading absolute [cm^3 (STP)/gr framework]           1.3528604368 +/-       0.0398825911 [-]
    Average loading absolute [cm^3 (STP)/cm^3 framework]         2.4302005720 +/-       0.0716427897 [-]

    Block[ 0] 2.74673           [-]
    Block[ 1] 2.74933           [-]
    Block[ 2] 2.67033           [-]
    Block[ 3] 2.85613           [-]
    Block[ 4] 2.75413           [-]
    ------------------------------------------------------------------------------
    Average loading excess                              2.7553276026 +/-       0.0821082425 [-]
    Average loading excess [molecules/unit cell]        0.3444159503 +/-       0.0102635303 [-]
    Average loading excess [mol/kg framework]                  0.0597105340 +/-       0.0017793626 [-]
    Average loading excess [milligram/gram framework]          0.9579038533 +/-       0.0285453540 [-]
    Average loading excess [cm^3 (STP)/gr framework]           1.3383504609 +/-       0.0398825911 [-]
    Average loading excess [cm^3 (STP)/cm^3 framework]         2.4041356871 +/-       0.0716427897 [-]
```

Note that the difference between absolute and excess adsorption is very small at low pressures.
The enthalpy of adsorption can be computed using [4, 5, 6]:

$$\Delta H = \left(\frac{\partial U}{\partial \langle N \rangle}\right)_{V,T} - \langle U_g \rangle - RT \tag{4.2}$$

$$= \frac{\langle U \times N \rangle_\mu - \langle U \rangle_\mu \langle N \rangle_\mu}{\langle N^2 \rangle_\mu - \langle N \rangle_\mu^2} - \langle U_g \rangle - RT \tag{4.3}$$

where N is the number of adsorbates in the system. Both equations can be used in grand-canonical MC [5].

```
Total enthalpy of adsorption
----------------------------
Block[ 0] -2279.12725       [K]
Block[ 1] -2276.89281       [K]
Block[ 2] -2285.95920       [K]
Block[ 3] -2282.91728       [K]
Block[ 4] -2281.21265       [K]
------------------------------------------------------------------------------
Average          -2281.22184 +/-           4.320449 [K]
                   -18.96714 +/-           0.035922 [KJ/MOL]
Note: Ug should be subtracted from this value
Note: The heat of adsorption Q=-H
```

Since $\langle U_g \rangle = 0$, the enthalpy of adsorption computed from the fluctuation formula is -18.96±0.04 and matches the value from the previous example for the limiting case of infinite dilution.

Example 7: Adsorption isotherm of CO2 in Cu-BTC

The Cu-BTC structure file provided with RASPA as an example defines the atoms as:

```
loop_
_atom_site_label
_atom_site_type_symbol
_atom_site_fract_x
_atom_site_fract_y
_atom_site_fract_z
_atom_site_charge
Cu1  Cu     0.2853     0.2853     0           1.248
O1   O      0.3166     0.2431     0.9478     -0.624
C1   C      0.2968     0.2032     0.9313      0.494
C2   C      0.322      0.178      0.887       0.130
C3   C      0.3655     0.1994     0.8655     -0.156
H1   H      0.3802     0.228      0.8802      0.156
```

Since we have three carbon-atoms with different charges, it is more convenient to specify the charges in the CIF-file using the tag `_atom_site_charge`. In the `simulation.input` file we then have to specify `UseChargesFromCIFFile yes`. In the output, we can check that the framework is charge neutral, and that the largest and smallest charge correspond to the expected values.

```
Framework has net charge: 0.000000
         largest charge : 1.248000
         smallest charge: -0.624000
```

If the charges are computed using QM, e.g. REPEAT, then put these computed charges in the CIF-file and check they add up to net charge zero.

The force field for this example is a very generic force field based on DREIDING [7] and UFF [8], while the force field for the CO_2 adsorbate is taken from Garcia-Sanchez et al. [9].

```
SimulationType                MonteCarlo
NumberOfCycles                10000
NumberOfInitializationCycles  5000
PrintEvery                    1000
RestartFile                   no

Forcefield                    ExampleMOFsForceField
UseChargesFromCIFFile         yes

Framework 0
FrameworkName Cu-BTC
UnitCells 1 1 1
HeliumVoidFraction 0.29
ExternalTemperature 323.0
ExternalPressure 100000.0

Component 0 MoleculeName             CO2
            MoleculeDefinition       ExampleDefinitions
            FugacityCoefficient      1.0
            TranslationProbability   0.5
            RotationProbability      0.5
            ReinsertionProbability   0.5
            SwapProbability          1.0
            CreateNumberOfMolecules  0
```

The `FugacityCoefficient 1.0` sets the fugacity coefficient to unity and we compute adsorption at 1 bar fugacity instead of pressure.

```
Partial pressure:    100000.00000000000000 [Pa]
                        750.00000000000000 [Torr]
                          1.00000000000000 [bar]
                          0.98692326671601 [atm]

Fugacity coefficient:       1.0000000000 [-]

Partial fugacity:    100000.00000000000000 [Pa]
                        750.00000000000000 [Torr]
                          1.00000000000000 [bar]
                          0.98692326671601 [atm]


Component 0 [CO2]
-------------------------------------------------------------
        Block[ 0] 113.35630         [-]
        Block[ 1] 113.86440         [-]
        Block[ 2] 113.56280         [-]
        Block[ 3] 113.59460         [-]
        Block[ 4] 114.04460         [-]
        ------------------------------------------------------------------------------
        Average loading absolute                           113.6845400000 +/-       0.3357951266 [-]
        Average loading absolute [molecules/unit cell]     113.6845400000 +/-       0.3357951266 [-]
        Average loading absolute [mol/kg framework]               11.7467584695 +/-       0.0346969275 [-]
        Average loading absolute [milligram/gram framework]      516.8432765492 +/-       1.5266231756 [-]
        Average loading absolute [cm^3 (STP)/gr framework]       263.2915594489 +/-       0.7776960925 [-]
        Average loading absolute [cm^3 (STP)/cm^3 framework]     231.4587585851 +/-       0.6836701203 [-]

        Block[ 0] 112.11359         [-]
        Block[ 1] 112.62169         [-]
        Block[ 2] 112.32009         [-]
        Block[ 3] 112.35189         [-]
        Block[ 4] 112.80189         [-]
        ------------------------------------------------------------------------------
        Average loading excess                           112.4418268337 +/-       0.3357951266 [-]
        Average loading excess [molecules/unit cell]     112.4418268337 +/-       0.3357951266 [-]
        Average loading excess [mol/kg framework]                 11.6183518154 +/-       0.0346969275 [-]
        Average loading excess [milligram/gram framework]        511.1935378542 +/-       1.5266231756 [-]
        Average loading excess [cm^3 (STP)/gr framework]         260.4134558167 +/-       0.7776960925 [-]
        Average loading excess [cm^3 (STP)/cm^3 framework]       228.9286269881 +/-       0.6836701203 [-]


Performance of the translation move:
======================================
Component 0 [CO2]
        total        378713.000000 378139.000000 379463.000000
        succesfull   189217.000000 199384.000000 196127.000000
        accepted   0.499632 0.527277 0.516854
        displacement 0.847781 0.850624 0.860931

Performance of the rotation move:
=================================
Component 0 [CO2]
```

```
        total        379274.000000 379013.000000 379022.000000
        succesfull   184255.000000 204253.000000 189456.000000
        accepted   0.485810 0.538908 0.499855
        angle-change 42.136262 42.791984 42.579446

Performance of the swap addition move:
======================================
Component [CO2] total tried: 1136797.000000 succesfull growth: 1089498.000000 (95.839275 [%]) accepted: 247115.000000 (21.737830 [%])

Performance of the swap deletion move:
======================================
Component [CO2] total tried: 1135121.000000 succesfull growth: 1135121.000000 (100.000000 [%]) accepted: 247121.000000 (21.770454 [%])

Performance of the Reinsertion move:
====================================
Component [CO2] total tried: 1138685.000000 succesfull growth: 1093296.000000 (96.013911 [%]) accepted: 96918.000000 (8.511397 [%])
```

Example 8: Henry coefficient of n-hexane in mono-clinic ERI

The monoclinic version of erionite (ERI) is named 'ERI_mono', the orthorhombic version is 'ERI'. The monoclinic version needs at least $3 \times 3 \times 3$ unit cells to be larger than twice the cutoff, while the orthorhombic needs $2 \times 2 \times 2$ (the unit cell shapes and size are different). To compute the Henry coefficient of hexane in erionite two simulations need to be performed. First the ideal Rosenbluth gas value needs to be computed at the desired temperature (see Auxiliary examples). This value needs to be filled in first. Next the simulation is started and the Henry coefficient is listed in the output.

```
SimulationType                MonteCarlo
NumberOfCycles                20000
NumberOfInitializationCycles  0
PrintEvery                    1000
PrintPropertiesEvery          1000

Forcefield                    ExampleZeolitesForceField

Framework 0
FrameworkName ERI_SI
RemoveAtomNumberCodeFromLabel yes
UnitCells 3 3 3
ExternalTemperature 573.0

Component 0 MoleculeName             hexane
            MoleculeDefinition       ExampleDefinitions
            IdealRosenbluthValue     0.0164786
            WidomProbability         1.0
            CreateNumberOfMolecules  0
```

The average Widom Rosenbluth weight and Henry coefficient are printed:

```
Average Widom Rosenbluth factor:
================================
    Block[ 0] 1.44801 [-]
    Block[ 1] 1.44939 [-]
    Block[ 2] 1.43931 [-]
    Block[ 3] 1.47096 [-]
    Block[ 4] 1.48635 [-]
    ------------------------------------------------------------------------------
    [hexane] Average Widom Rosenbluth-weight:   1.45881 +/- 0.023974 [-]

Average Henry coefficient:
==========================
    Block[ 0] 1.94214e-07 [mol/kg/Pa]
    Block[ 1] 1.94399e-07 [mol/kg/Pa]
    Block[ 2] 1.93047e-07 [mol/kg/Pa]
    Block[ 3] 1.97292e-07 [mol/kg/Pa]
    Block[ 4] 1.99356e-07 [mol/kg/Pa]
    ------------------------------------------------------------------------------
    [hexane] Average Henry coefficient:  1.95661e-07 +/- 3.21546e-09 [mol/kg/Pa]
```

Example 9: Henry coefficient of n-pentane to n-nonane in MFI

By using multiple components several Henry coefficients can be computed simultaneously. The Widom insertion probe move never actually inserts the molecules, it just compute the energy at randomly chosen insertion positions. Note that the ideal gas Rosenbluth weights decrease with chain length.

```
SimulationType                MonteCarlo
NumberOfCycles                10000
NumberOfInitializationCycles  0
PrintEvery                    1000
PrintPropertiesEvery          1000

Forcefield                    ExampleZeolitesForceField

Framework 0
FrameworkName MFI_SI
RemoveAtomNumberCodeFromLabel yes
UnitCells 2 2 2
ExternalTemperature 573.0
```

```
Component 0 MoleculeName             pentane
            MoleculeDefinition       ExampleDefinitions
            IdealGasRosenbluthWeight 0.0639633
            WidomProbability         1.0
            CreateNumberOfMolecules  0

Component 1 MoleculeName             hexane
            MoleculeDefinition       ExampleDefinitions
            IdealGasRosenbluthWeight 0.0164786
            WidomProbability         1.0
            CreateNumberOfMolecules  0

Component 2 MoleculeName             heptane
            MoleculeDefinition       ExampleDefinitions
            IdealGasRosenbluthWeight 0.00425633
            WidomProbability         1.0
            CreateNumberOfMolecules  0

Component 3 MoleculeName             octane
            MoleculeDefinition       ExampleDefinitions
            IdealGasRosenbluthWeight 0.00110671
            WidomProbability         1.0
            CreateNumberOfMolecules  0

Component 4 MoleculeName             nonane
            MoleculeDefinition       ExampleDefinitions
            IdealGasRosenbluthWeight 0.000289443
            WidomProbability         1.0
            CreateNumberOfMolecules  0
```

The resulting Henry coefficients are:

```
Average Henry coefficient:
==========================
    [pentane] Average Henry coefficient:  3.46392e-06 +/- 2.08318e-08 [mol/kg/Pa]
    [hexane] Average Henry coefficient:  7.80806e-06 +/- 2.66917e-07 [mol/kg/Pa]
    [heptane] Average Henry coefficient:  1.70348e-05 +/- 5.97899e-07 [mol/kg/Pa]
    [octane] Average Henry coefficient:  3.76354e-05 +/- 1.09689e-06 [mol/kg/Pa]
    [nonane] Average Henry coefficient:  8.61517e-05 +/- 9.87367e-06 [mol/kg/Pa]
```

Example 10: Computing the radial distribution function of water using MD

The radial distribution function (RDF) is a good indication of the status of the fluid: solid, liquid or gas. Water is expensive to compute. Here we start from a 'restart'-file obtained from a previous run. The simulations will always write a file `Restart` at the end of the simulation. Rename this file to `RestartInitial` and specify

```
RestartFile                  yes

Component 0 MoleculeName            Tip5p
            CreateNumberOfMolecules 0
```

The `RestartFile yes` will read the `RestartInitial`-files. Set `CreateNumberOfMolecules` back to zero. You can also a non-zero number here, to create *additional* molecules, if e.g. you would like to increase the density. The input looks like

```
SimulationType               MolecularDynamics
NumberOfCycles               25000
NumberOfInitializationCycles 1000
NumberOfEquilibrationCycles  5000
PrintEvery                   1000
RestartFile                  yes

Ensemble                     NVT

Forcefield                   Local

Box 0
BoxLengths 24.83 24.83 24.83
ComputeRDF yes
WriteRDFEvery 1000
ExternalTemperature 298.0

Component 0 MoleculeName            Tip5p
            MoleculeDefinition      Local
            TranslationProbability  0.5
            RotationProbability     0.5
            ReinsertionProbability  1.0
            CreateNumberOfMolecules 0
```

In the basic examples before, the pre-stored forcefield and molecule files were used. In this example, they are stored locally. The `Forcefield` and `MoleculeDefinition` keywords can be set to anything (here `Local`), the local files will always be read first and used if they are found. The Tip5p model uses sites for the oxygen and hydrogen, and additional dummy sites to place charges. RASPA computes the RDF for all (pseudo-)atoms pairs, unless you specified `no` to the `PrintToPDB`-field of the `pseudo_atoms` file. For example, the `L`-atoms of water should not be printed to movie-files, and there would be little point generating the RDF for interactions with these 'dummy' interaction sites.

```
#number of pseudo atoms
3
#type      print   as    chem  oxidation   mass        charge   polarization B-factor radii  connectivity anisotropic anisotropic-type   tinker-type
Ow         yes     O     O     0           15.9996     0.0      0.0          1.0      0.5    2            0           absolute           0
Hw         yes     H     H     0           1.0008      0.241    0.0          1.0      1.00   1            0           absolute           0
Lw         no      L     -     0           0.0        -0.241    0.0          1.0      1.00   1            0           absolute           0
```

The `pseudo_atoms` file also defines the charge used for the computation of the electrostatic interactions. With these definitions in place, the molecule can be defined. Many small molecules are simulated as small rigid units with no internal degrees of freedom. For these types molecules, the relative positions of the atoms need to be specified.

```
# critical constants: Temperature [T], Pressure [Pa], and Acentric factor [-]
304.1282
7377300.0
0.22394
# total number Of atoms
5
# Number of groups
1
# water-group
rigid
# number of atoms
5
# atomic positions
0 Ow      0.0                     0.0                     0.0
1 Hw     -0.75695032726366118157  0.0                    -0.58588227661829499395
2 Hw      0.75695032726366118157  0.0                    -0.58588227661829499395
3 Lw      0.0                    -0.57154330164408200866  0.40415127656087122858
4 Lw      0.0                     0.57154330164408200866  0.40415127656087122858
# Chiral centers Bond  BondDipoles Bend  UrayBradley InvBend  Torsion Imp. Torsion Bond/Bond Stretch/Bend Bend/Bend Stretch/Torsion Bend/Torsion IntraVDW IntraCoulomb
               0    4           0    0             0       0        0            0         0            0         0               0            0        0            0
# Bond stretch: atom n1-n2, type, parameters
0 1 RIGID_BOND
0 2 RIGID_BOND
0 3 RIGID_BOND
0 4 RIGID_BOND
# Number of config moves
0
```

Note that the molecule file starts with the critical temperature and pressure, and the acentric factor. The Peng-Robinson equations of state potentially uses this information to convert pressure to fugacity in open ensembles, and also to automatically compute *excess* adsorption.

The long-range Van der Waals interactions are defined in the `force_field_mixing_rules.def` file.

```
# general rule for shifted vs truncated
shifted
# general rule tailcorrections
no
# number of defined interactions
3
# type interaction
Ow      lennard-jones   89.633     3.097
Lw      none
Hw      none
# general mixing rule for Lennard-Jones
Jorgensen
```

Types that have no Van der Waals interaction are listed with `none`. Note that, if the sigma of the oxygen would be chosen to small, the `Lw` sites, and also `Hw` sites might overlap. This can lead to numerically problems. RASPA warns therefore for missing defined interactions, and specifying them as `none` here avoids the warning. The most common (and convenient) way of creating a force field, is to list all defined self-interactions here, and then use a mixture rule to compute all cross-interactions.

Example 11: Computing the radial distribution function of water using MC

The radial distribution function is a static property, and can therefore also be computed using MC. The input is:

```
SimulationType               MonteCarlo
NumberOfCycles               25000
NumberOfInitializationCycles 10000
PrintEvery                   1000
RestartFile                  no

Forcefield                   Local

Box 0
BoxLengths 24.83 24.83 24.83
ComputeRDF yes
WriteRDFEvery 1000
ExternalTemperature 298.0

Component 0 MoleculeName            Tip5p
            MoleculeDefinition      Local
            TranslationProbability   0.5
            RotationProbability      0.5
            ReinsertionProbability   1.0
            CreateNumberOfMolecules  512
```

This allows you to compare the differences between MD and MC. MD is very efficient for equilibrating homogeneous systems. In contrast to MC, it can handle collective motions, which sometimes can cause a difference between MD and MC results. For this simple system, we obtain identical RDFs, as shown in Figure 9.

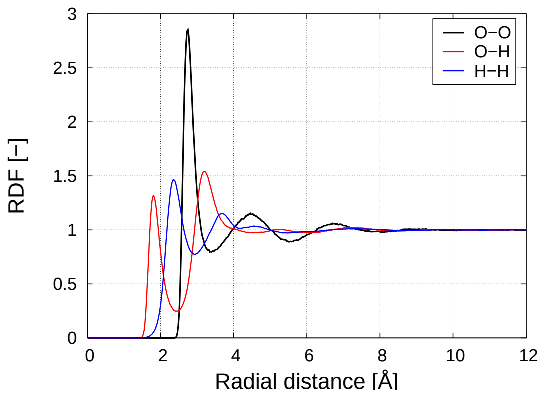


Figure 9: *The radial distribution function of water at 298K.*

Example 12: Measuring bond/bend/dihedral angle distributions MC

We revisit the computation measuring the bond/bend/dihedral angle distributions, but now use MC. The input is given as

```
SimulationType                   MonteCarlo
NumberOfCycles                   5000000
NumberOfInitializationCycles     10000
PrintEvery                       50000
RestartFile                      no

Forcefield                       ExampleMoleculeForceField

Box 0
BoxLengths 25 25 25
ExternalTemperature 298.0
ExternalPressure 0.0
ComputeMoleculeProperties yes

component 0 MoleculeName                     2-methylbutane
            FugacityCoefficient              1.0
            MoleculeDefinition               ExampleDefinitions
            TranslationProbability           1.0
            RotationProbability              1.0
            ReinsertionProbability           1.0
            PartialReinsertionProbability    1.0
            CreateNumberOfMolecules          32
```

The `pseudo_atoms.def` is defined as

```
#number of pseudo atoms
5
#type      print   as    chem  oxidation   mass        charge    polarization B-factor radii  connectivity anisotropic anisotropic-type   tinker-type
CH4        yes     C     C     0           16.04246    0.0       0.0          1.0      1.00   0            0           relative           0
CH3        yes     C     C     0           15.03452    0.0       0.0          1.0      1.00   0            0           relative           0
CH2        yes     C     C     0           14.02658    0.0       0.0          1.0      1.00   0            0           relative           0
CH         yes     C     C     0           13.01864    0.0       0.0          1.0      1.00   0            0           relative           0
C          yes     C     C     0           12.0        0.0       0.0          1.0      1.00   0            0           relative           0
```

To properly sample the internal structure we use the `Reinsertion`-move. However, the acceptance of this move is often low, especially at high densities and/or low temperatures. A move that helps is the `PartialReinsertion` move. This move keeps certain atoms of the molecule fixes, and regenerates the others. The `PartialReinsertion` move is defined by adding 'config'-moves to the molecule definition.

```
# critical constants: Temperature [T], Pressure [Pa], and Acentric factor [-]
460.35
3395700.0
0.2296
```

```
# Number Of Atoms
5
# Number Of Groups
1
# Alkane-group
flexible
# number of atoms
5
# atomic positions
0 CH3
1 CH
2 CH2
3 CH3
4 CH3
# Chiral centers Bond  BondDipoles Bend  UrayBradley InvBend  Torsion Imp. Torsion Bond/Bond Stretch/Bend Bend/Bend Stretch/Torsion Bend/Torsion IntraVDW IntraCoulomb
               0    4           0    4             0       0        2            0         0            0         0               0            0        0            0
# Bond stretch: atom n1-n2, type, parameters
0 1 HARMONIC_BOND 96500 1.54
1 2 HARMONIC_BOND 96500 1.54
1 4 HARMONIC_BOND 96500 1.54
2 3 HARMONIC_BOND 96500 1.54
# Bond bending: atom n1-n2-n3, type, parameters
0 1 2 HARMONIC_BEND 62500 112
0 1 4 HARMONIC_BEND 62500 112
4 1 2 HARMONIC_BEND 62500 112
1 2 3 HARMONIC_BEND 62500 114
# Torsion n1-n2-n3-n4 type
0 1 2 3 TRAPPE_DIHEDRAL   -251.06  428.73  -111.85  441.27
4 1 2 3 TRAPPE_DIHEDRAL   -251.06  428.73  -111.85  441.27
# Number of config moves
4
# nr fixed, list
2 0 1
2 3 2
3 3 2 1
4 0 1 2 4
```

Here, 4 config moves are defined. The first number is the number of fixed atoms, and then a list of the atom identifiers that are kept fixed (the other atoms are regrown). Note that in CBMC, all branches must be grown simultaneously. That means, you *cannot* keep more than one branch fixes. The long-range Van der Waals interactions are defined in the `force_field_mixing_rules.def` file.

```
# general rule for shifted vs truncated
shifted
# general rule tailcorrections
no
# number of defined interactions
5
# type interaction, parameters.
CH4           lennard-jones    158.5     3.72         // M. G. Martin et al., J. Chem. Phys. 2001, 114, 7174-7181.
CH3           lennard-jones    108.0     3.76         // D. Dubbeldam et al., J. Phys. Chem. B, 108(33), 12301-12313
CH2           lennard-jones    56.0      3.96         // idem
CH            lennard-jones    17.0      4.67         // idem
C             lennard-jones     0.8      6.38         // idem
# general mixing rule for Lennard-Jones
Lorentz-Berthelot
```

The most common (and convenient) way of creating a force field, is to list all defined self-interactions here, and then use a mixture rule to compute all cross-interactions. The more atom-types you have, the more convenient this becomes.

The acceptance ratios of the partial-reinsertion-move is much high then the reinsertion-move. This is due to the fact that space already exists at the position of the molecule to regrow parts of its internal structure.

```
Performance of the Reinsertion move:
===================================
Component [2-methylbutane] total tried: 8001488.000000 succesfull growth: 7990960.000000 (99.868424 [%]) accepted: 1114281.000000 (13.925922 [%])

Performance of the partial reinsertion move:
===========================================
Component [2-methylbutane] total tried: 7995526.000000 succesfull growth: 7994430.000000 (99.986292 [%]) accepted: 4356062.000000 (54.481244 [%])
```

Example 13: Measuring bond/bend/dihedral angle distributions MD

We can also compute the angle distributions using MD with the following input:

```
SimulationType                   MolecularDynamics
NumberOfCycles                   5000000
NumberOfInitializationCycles     5000
NumberOfEquilibrationCycles      10000
PrintEvery                       10000

Ensemble NVT

Forcefield                       ExampleMoleculeForceField

Box 0
BoxLengths 25 25 25
ExternalTemperature 298.0
ExternalPressure 0.0
```

```
ComputeMoleculeProperties yes

component 0 MoleculeName                    2-methylbutane
            MoleculeDefinition              ExampleDefinitions
            TranslationProbability          1.0
            RotationProbability             1.0
            ReinsertionProbability          1.0
            CreateNumberOfMolecules         32
```

4.3 Non-basic examples

Example 1: Adsorption of a binary CO_2/CH_4 (1:3) mixture in IRMOF-1

Appreciable adsorption in MOF materials occurs at higher pressure than zeolites, usually in the range up to 10 bar. At these high pressures absolute and excess adsorption are different, and excess adsorption eventually even goes down. This is due to the fact that excess adsorption is relative to what would have been in the free pore volume at these conditions. So one can compress the outside fluid but eventually the pores are filled up. At that maximum absolute loading the excess adsorption will go down.

```
SimulationType                MonteCarlo
NumberOfCycles                50000
NumberOfInitializationCycles  5000
PrintEvery                    1000

Forcefield                    Dubbeldam2007FlexibleIRMOF-1

Framework 0
FrameworkName IRMOF-1
UnitCells 1 1 1
HeliumVoidFraction 0.81
ExternalTemperature 300.0
ExternalPressure  10e5

Component 0 MoleculeName               CO2
            MoleculeDefinition         ExampleDefinitions
            MolFraction                0.25
            TranslationProbability     0.5
            RegrowProbability          0.5
            IdentityChangeProbability  1.0
              NumberOfIdentityChanges  2
              IdentityChangesList      0 1
            SwapProbability            1.0
            CreateNumberOfMolecules    0

Component 1 MoleculeName               methane
            MoleculeDefinition         ExampleDefinitions
            MolFraction                0.75
            TranslationProbability     0.5
            RegrowProbability          0.5
            IdentityChangeProbability  1.0
              NumberOfIdentityChanges  2
              IdentityChangesList      0 1
            SwapProbability            1.0
            CreateNumberOfMolecules    0
```

To compute the excess adsorption the void fraction of a structure needs to be specified using 'HeliumVoidFraction [real]'. RASPA automatically uses an equation of state (default: Peng-Robinson) to compute the fugacities from the pressure and mol-fraction as is done here for a mixture of CO_2 and CH_4. It also computes the amount of excess molecules from this equation of state.

```
Component 0 [CO2] (Adsorbate molecule)

    MoleculeDefinitions: ExampleDefinitions
    Component contains (at least some) atoms which are charged
    Component contains no atoms with point dipoles (polarization)
    Component has a net charge of 0.000000

    Ideal chain Rosenbluth weight: 1
    Ideal chain total energy: 0.000000

    Critical temparure [K]: 304.128200
    Critical pressure [Pa]: 7377300.000000
    Acentric factor [-]: 0.223940

    RXMC partition factor ln(q/V) [ln(A^(-3))]:       0.0000000000

    Fluid is a vapour

    MolFraction:           0.2500000000 [-]
    Compressibility:       0.9714389725 [-]

    Density of the bulk fluid phase:      18.1580726483 [kg/m^3]

    Binary mixture EOS parameters:  (0): 0.000000 (1): 0.000000

    Amount of excess molecules:       0.8675190741 [-]

    Conversion factor molecules/unit cell -> mol/kg:       0.1623747175 [-]
    Conversion factor molecules/unit cell -> mg/g:       7.1442927209 [-]
    Conversion factor molecules/unit cell -> cm^3 STP/gr:       3.6394629804 [-]
    Conversion factor molecules/unit cell -> cm^3 STP/cm^3:       2.1592046669 [-]
    Conversion factor mol/kg -> cm^3 STP/gr:      22.4139757476 [-]
    Conversion factor mol/kg -> cm^3 STP/cm^3:      13.2976654244 [-]

    Partial pressure:    250000.00000000000000 [Pa]
                           1874.99999999999977 [Torr]
                              2.50000000000000 [bar]
                              2.46730816679003 [atm]

    Fugacity coefficient:       0.9503504709 [-]

    Partial fugacity:    237587.61773457151139 [Pa]
                           1781.90713300928633 [Torr]
                              2.37587617734572 [bar]
                              2.34480747825879 [atm]

Component 1 [methane] (Adsorbate molecule)

    MoleculeDefinitions: ExampleDefinitions
    Component contains no atoms with charge
    Component contains no atoms with point dipoles (polarization)
    Component has a net charge of 0.000000

    Ideal chain Rosenbluth weight: 1
    Ideal chain total energy: 0.000000

    Critical temparure [K]: 190.564000
    Critical pressure [Pa]: 4599200.000000
    Acentric factor [-]: 0.011420

    RXMC partition factor ln(q/V) [ln(A^(-3))]:       0.0000000000

    Fluid is a vapour

    MolFraction:           0.7500000000 [-]
    Compressibility:       0.9714389725 [-]

    Density of the bulk fluid phase:       6.6206386115 [kg/m^3]

    Binary mixture EOS parameters:  (0): 0.000000 (1): 0.000000

    Amount of excess molecules:       2.6025572224 [-]

    Conversion factor molecules/unit cell -> mol/kg:       0.1623747175 [-]
    Conversion factor molecules/unit cell -> mg/g:       2.6048899107 [-]
    Conversion factor molecules/unit cell -> cm^3 STP/gr:       3.6394629804 [-]
    Conversion factor molecules/unit cell -> cm^3 STP/cm^3:       2.1592046669 [-]
    Conversion factor mol/kg -> cm^3 STP/gr:      22.4139757476 [-]
    Conversion factor mol/kg -> cm^3 STP/cm^3:      13.2976654244 [-]

    Partial pressure:    750000.00000000011642 [Pa]
                           5625.00000000000091 [Torr]
                              7.50000000000000 [bar]
                              7.40192450037010 [atm]

    Fugacity coefficient:       0.9790119494 [-]

    Partial fugacity:    734258.96201743301935 [Pa]
                           5506.94221513074717 [Torr]
                              7.34258962017433 [bar]
                              7.24657253409754 [atm]
```

At each 'PrintEvery' steps the loadings are shown in a variety of units for both excess and absolute adsorption:

```
Loadings per component:
----------------------------------------------------------------------------------------------------------------------------------------------------
Component 0 (CO2), current number of integer/fractional/reaction molecules: 16/0/0 (avg.  14.98769), density:  67.81662 (avg.  63.52592) [kg/m^3]
    absolute adsorption:  16.00000 (avg.  14.98769) [mol/uc],   2.5979954802 (avg.   2.4336226003) [mol/kg], 114.3086835349 (avg. 107.0764740672) [mg/g]
                          58.2314076859 (avg.  54.5471579425) [cm^3 STP/g],   34.5472746700 (avg.  32.3614991084) [cm^3 STP/cm^3]
    excess adsorption:    15.1324809259 (avg.  14.1201750545) [mol/uc],   2.4571323156 (avg.   2.2927594357) [mol/kg], 108.1108733282 (avg. 100.8786638605) [mg/g]
                          55.0741041308 (avg.  51.3898543873) [cm^3 STP/g],   32.6741234365 (avg.  30.4883478749) [cm^3 STP/cm^3]
Component 1 (methane), current number of integer/fractional/reaction molecules: 19/0/0 (avg.  19.62858), density:  29.36296 (avg.  30.33438) [kg/m^3]
    absolute adsorption:  19.00000 (avg.  19.62858) [mol/uc],   3.0851196328 (avg.   3.1871849717) [mol/kg],  49.4929083037 (avg.  51.1302874213) [mg/g]
                          69.1497966270 (avg.  71.4374866590) [cm^3 STP/g],   41.0248886706 (avg.  42.3821193995) [cm^3 STP/cm^3]
    excess adsorption:    16.3974427776 (avg.  17.0260217862) [mol/uc],   2.6625301390 (avg.   2.7645954779) [mol/kg],  42.7135332529 (avg.  44.3509123705) [mg/g]
                          59.6778859617 (avg.  61.9655759937) [cm^3 STP/g],   35.4054349701 (avg.  36.7626656990) [cm^3 STP/cm^3]
----------------------------------------------------------------------------------------------------------------------------------------------------
```

and at the end error bars are computed for all properties:

```
Component 0 [CO2]
-------------------------------------------------------------
    Block[ 0] 14.98680          [-]
    Block[ 1] 14.91280          [-]
    Block[ 2] 15.16230          [-]
    Block[ 3] 14.95220          [-]
    Block[ 4] 14.89510          [-]
    ------------------------------------------------------------------------------
    Average loading absolute                           14.9818400000 +/-       0.1327835001 [-]
    Average loading absolute [molecules/unit cell]     14.9818400000 +/-       0.1327835001 [-]
    Average loading absolute [mol/kg framework]                2.4326720378 +/-       0.0215606833 [-]
    Average loading absolute [milligram/gram framework]      107.0346504582 +/-       0.9486441930 [-]
    Average loading absolute [cm^3 (STP)/gr framework]        54.5258520578 +/-       0.4832606329 [-]
    Average loading absolute [cm^3 (STP)/cm^3 framework]      32.3488588464 +/-       0.2867067530 [-]

    Block[ 0] 14.11928          [-]
    Block[ 1] 14.04528          [-]
    Block[ 2] 14.29478          [-]
    Block[ 3] 14.08468          [-]
    Block[ 4] 14.02758          [-]
    ---------------------------------------------------------------------------
    Average loading excess                           14.1143209259 +/-       0.1327835001 [-]
    Average loading excess [molecules/unit cell]     14.1143209259 +/-       0.1327835001 [-]
    Average loading excess [mol/kg framework]                2.2918088732 +/-       0.0215606833 [-]
    Average loading excess [milligram/gram framework]      100.8368402515 +/-       0.9486441930 [-]
    Average loading excess [cm^3 (STP)/gr framework]        51.3685485027 +/-       0.4832606329 [-]
    Average loading excess [cm^3 (STP)/cm^3 framework]      30.4757076129 +/-       0.2867067530 [-]

Component 1 [methane]
-------------------------------------------------------------
    Block[ 0] 19.54180          [-]
    Block[ 1] 19.78000          [-]
    Block[ 2] 19.77410          [-]
    Block[ 3] 19.46900          [-]
    Block[ 4] 19.55980          [-]
    ---------------------------------------------------------------------------
    Average loading absolute                           19.6249400000 +/-       0.1774959204 [-]
    Average loading absolute [molecules/unit cell]     19.6249400000 +/-       0.1774959204 [-]
    Average loading absolute [mol/kg framework]                3.1865940887 +/-       0.0288208499 [-]
    Average loading absolute [milligram/gram framework]       51.1208082045 +/-       0.4623573324 [-]
    Average loading absolute [cm^3 (STP)/gr framework]        71.4242426220 +/-       0.6459898316 [-]
    Average loading absolute [cm^3 (STP)/cm^3 framework]      42.3742620351 +/-       0.3832500198 [-]

    Block[ 0] 16.93924          [-]
    Block[ 1] 17.17744          [-]
    Block[ 2] 17.17154          [-]
    Block[ 3] 16.86644          [-]
    Block[ 4] 16.95724          [-]
    ---------------------------------------------------------------------------
    Average loading excess                           17.0223827776 +/-       0.1774959204 [-]
    Average loading excess [molecules/unit cell]     17.0223827776 +/-       0.1774959204 [-]
    Average loading excess [mol/kg framework]                2.7640045949 +/-       0.0288208499 [-]
    Average loading excess [milligram/gram framework]       44.3414331537 +/-       0.4623573324 [-]
    Average loading excess [cm^3 (STP)/gr framework]        61.9523319566 +/-       0.6459898316 [-]
    Average loading excess [cm^3 (STP)/cm^3 framework]      36.7548083346 +/-       0.3832500198 [-]
```

Also noteworthy is the use of the identity-change move for mixtures. A molecule of a certain type can be changed at the same position into a molecule of another type. It is specified per component as a list of other components that are allowed for this move. The identity-change move is highly recommended at high loadings.

```
Performance of the identity change move:
======================================
Component [CO2]->[CO2] total tried: 289262.000000 succesfull growth: 289249.000000 (99.995506 [%]) accepted: 165408.000000 (57.182762 [%])
Component [CO2]->[methane] total tried: 288518.000000 succesfull growth: 288518.000000 (100.000000 [%]) accepted: 140423.000000 (48.670447 [%])
Component [methane]->[CO2] total tried: 288616.000000 succesfull growth: 288616.000000 (100.000000 [%]) accepted: 140676.000000 (48.741581 [%])
Component [methane]->[methane] total tried: 288680.000000 succesfull growth: 288680.000000 (100.000000 [%]) accepted: 274211.000000 (94.987876 [%])
```

Example 2: NPT Monte Carlo of propane

The density of propane at 250K and 10 bar is about 559.53 kg/m^3 (NIST database). In this example the density is computed using Monte Carlo in the NPT-ensemble. Given the pressure P, the temperature T, and the amount of molecules N, the density is computed.

```
SimulationType                MonteCarlo
NumberOfCycles                50000
NumberOfInitializationCycles  10000
PrintEvery                    1000
RestartFile                   no

Forcefield                    ExampleMoleculeForceField

Box 0
BoxLengths 30 30 30
ExternalTemperature 250.0
ExternalPressure 1e6
ComputeMolecularPressure yes

VolumeChangeProbability       0.05

Component 0 MoleculeName             propane
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   0.5
            RotationProbability      0.5
            ReinsertionProbability   0.5
            CreateNumberOfMolecules  256
```

The simulation for propane gives 568.77 ± 1.99 kg/m^3. The measured pressure for this short simulation is 10.08 ± 1.98 bar.

Example 3: NPT molecular dynamics of water

A molecular dynamics simulation of water in the NPT-ensemble (constant amount of particles N, constant average pressure P, and constant average temperature T). Many water models are defined, but most are defined with simple Coulombic potentials using cutoffs of 9Å. None are optimized with the Ewald-summation except for the re-calibrated Tip5p-Ew model. Unfortunately, that model is defined using a cutoff always equal to half the box size, while RASPA uses a fixed cutoff (default: 12 Angstrom). A fixed cutoff is more realistic, but requires the shortest perpendicular width to be twice the cutoff, thus here larger than 24 Å. All this results in having to simulate more than 512 water molecules. The tip5p models use 5 fixed charges placed in the water geometry, so for each step 2560 charge sites needs to be computed with Ewald. Conclusion: liquid water is computationally expensive to compute when done properly.

In MD-NPT the average pressure $\langle P \rangle$ and average temperature $\langle T \rangle$ are imposed. The instantaneous values, especially for the pressure, are different. RASPA uses the Nose-Hoover chain method, and NPT-MD methods of Martyna and Tuckermann.

Several options are introduced here: "TimeStep [real]" to set the time step. For rigid molecules the time step can be a bit larger because the high frequency movement is removed (the O-H is around 3000 cm^{-1}). The cutoff can be set with 'CutOff [real]'. The method to compute charge interactions is set with 'ChargeMethod [Ewald|None]', although Ewald is the default. The precision can specified using 'Ewald-Precision [real]' from which the Ewald parameters κ and the amount of wave vectors is inferred. The initial positions of the water are read from file ('RestartFile yes'), the file is located in directory 'RestartInitial/System[int]'.

The experimental density of water at 300K and 1 bar is about 996.56 kg/m^3 (NIST database).

```
SimulationType                MolecularDynamics
NumberOfCycles                100000
NumberOfInitializationCycles  0
NumberOfEquilibrationCycles   10000
PrintEvery                    5000
RestartFile                   yes

Ensemble                      NPT
TimeStep                      0.001

ChargeMethod                  Ewald
CutOff                        10.0
Forcefield                    Local
EwaldPrecision                1e-6

Box 0
BoxLengths 24.83 24.83 24.83
ExternalTemperature 300.0
ExternalPressure 1.0e5
ComputeMSD yes
PrintMSDEvery 5000

Component 0 MoleculeName             Tip5p
            MoleculeDefinition       Local
            TranslationProbability   1.0
            RotationProbability      1.0
            ReinsertionProbability   1.0
            CreateNumberOfMolecules  0
```

The output shows some details of intermediate status during the run: the time run, the current box and average box, etc. The total linear momentum is conserved and zero (the center of mass movement of the system is removed at initialization). For this relatively short run, the average pressure of 1.19 ± 0.62 bar is already quite close to the applied 1 bar, and the density is 993.4 ± 2.3 kg/m^3. Also, the temperature of the water, and of the simulation cell (it is a degree of freedom and has therefore an associated temperature) can also been seen to converge to the applied value of 300K. Energy conservation is adequate with a 0.001 ps time step.

Example 4: Adsorption of CO_2 in Na-LTA

The Linde Type A structure LTA-4A has 96 aluminum per unit cell. A common 4A sample has 96 charge balancing sodium ions. The ions are small enough to access the sodalite cages, but the bigger methane molecules are exclusively in the big α-cages and not in the sodalite cages. They need to be artificially blocked. Because the adsorption is dependent on the positions of the ions it is important to start from the crystallographic positions and use *only* translation for the ions. Reinsertion moves may transport the ions to positions in the windows and this is especially important for diffusion (the next example).

```
SimulationType                  MonteCarlo
NumberOfCycles                  25000
NumberOfInitializationCycles    10000
```

```
PrintEvery                      1000

Forcefield                      Local

Framework 0
FrameworkName LTA4A
RemoveAtomNumberCodeFromLabel yes
ModifyOxgensConnectedToAluminium yes
UnitCells 1 1 1
ExternalTemperature 298.0
ExternalPressure 10000.0

Component 0 MoleculeName                  sodium
            MoleculeDefinition            Local
            TranslationProbability        1.0
            RandomTranslationProbability  1.0
            ExtraFrameworkMolecule        yes
            CreateNumberOfMolecules       96

Component 1 MoleculeName                  CO2
            MoleculeDefinition            Local
            BlockPockets                  yes
            BlockPocketsFilename          LTA
            TranslationProbability        1.0
            ReinsertionProbability        1.0
            SwapProbability               1.0
            ExtraFrameworkMolecule        no
            CreateNumberOfMolecules       0
```

The force field is taken from Garcia-Sanchez et al.[9].

```
#number of pseudo atoms
7
#type      print   as    chem  oxidation   mass        charge    polarization B-factor radii  connectivity anisotropic anisotropic-type   tinker-type
O          yes     O     O     0           15.9994    -0.39299  0.0          1.0      0.5    2            0           absolute           0
Oa         yes     Oa     O    0           15.9994    -0.41384  0.0          1.0      0.5    2            0           absolute           0
Si         yes     Si    Si    0           28.0855     0.78598  0.0          1.0      1.18   4            0           absolute           0
Al         yes     Al    Al    0           26.981539   0.48598  0.0          1.0      1.18   4            0           absolute           0
C_co2      yes     C     C     0           12.0        0.6512   0.0          1.0      0.720  0            0           relative           0
O_co2      yes     O     O     0           15.9994    -0.3256   0.0          1.0      0.68   0            0           relative           0
Na         yes     Na    Na    0           22.98977    0.3834   0.0          1.0      1.00   0            0           absolute           0
```

The force field uses a different charge for the framework oxygen that is connected to an aluminum atom. The line

```
ModifyOxgensConnectedToAluminium yes
```

modifies the framework oxygen type 'O' to 'Oa' when the oxygen is found to be connected to an aluminum atom. For the LTA4A and LTA5A frameworks, *every* oxygen atom is of type 'Oa'.
The file `force_field_mixing_rules.def` is used to define the CO_2 model based on mixing rules

```
# general rule for shifted vs truncated
shifted
# general rule tailcorrections
no
# number of defined interactions
4
# type interaction, parameters.
O_co2          lennard-jones    85.671    3.017        // A. Garcia-Sanchez et al., J. Phys. Chem. C 2009, 113, 8814-8820.
C_co2          lennard-jones    29.933    2.745        // idem
Al             none
Si             none
# general mixing rule for Lennard-Jones
Lorentz-Berthelot
```

The interactions of the CO_2 with the zeolite and cations are directly described as pairs. For that, we can use the `force_field.def`-file.

```
# rules to overwrite
0
# number of defined interactions
13
# type      type2       interaction
Na          O           lennard-jones     23.0          3.4        // S. Calero et al., J. Phys. Chem. B 2003, 107(44), 12088-12096.
Na          Oa          lennard-jones     23.0          3.4        // S. Calero et al., J. Phys. Chem. B 2003, 107(44), 12088-12096.
Na          C_co2       lennard-jones    362.292        3.320      // A. Garcia-Sanchez et al., J. Phys. Chem. C 2009, 113(20), 8814-8820.
Na          O_co2       lennard-jones    200.831        2.758      // idem
O           C_co2       lennard-jones     37.595        3.511      // idem
O           O_co2       lennard-jones     78.980        3.237      // idem
Oa          C_co2       lennard-jones     37.595        3.511      // idem
Oa          O_co2       lennard-jones     78.980        3.237      // idem
Na          Si          none
Na          Al          none
Na          Na          none
O            O          none
Oa          Oa          none
# mixing rules to overwrite
0
```

The file overwrites any pairs already determined by the mixing rules. Both types of force fields (via mixing rules, and via pairs of interactions) are commonly found in the literature.
Note the line

```
ExtraFrameworkMolecule        yes
```

for the sodium component. This will allow for the computation of the (average) energies between the framework, adsorbates, and cations.

```
Current total potential energy:         -13661140.1319969334 [K]  (avg.   -13660492.8183784448)
    Current Host-Host energy:                   0.0000000000 [K]  (avg.           0.0000000000)
    Current Host-Adsorbate energy:          -38467.0736525305 [K]  (avg.      -39177.6800752787)
    Current Host-Cation energy:          -25997044.5562428236 [K]  (avg.   -25999347.1558655351)
    Current Adsorbate-Adsorbate energy:      -12290.0845394197 [K]  (avg.      -11985.7555443291)
    Current Cation-Cation energy:         12511975.9583617374 [K]  (avg.    12512947.5563156120)
    Current Adsorbate-Cation energy:        -125314.3759244105 [K]  (avg.     -122929.7832096771)
```

Also note the line

```
RandomTranslationProbability   1.0
```

Because of the large energies involved when using cations, the CBMC biasing algorithms might experience numerical under/overflow. Since the sodium-ion is a small molecule, the reinsertion (which is biased) can be replaced with a random translation. For a single atom molecules, both do the same thing: a reinsertion randomly in the simulation box.
Lastly, LTA contains pockets called β-cages that are not accessible from the main large cavities called α-cages. In MC, the reinsertion and swap-move pick random location inside the simulation cell volume, and therefore we need to artificially block the inaccessible pockets. For that, we can a blocking file, here names `LTA.block`.

```
BlockPockets yes
BlockPocketsFilename LTA
```

This file basically contains 32 pockets with a fractional center point and a radius in Angstrom.

```
32
0.0       0.0       0.0       4.0
0.5       0.0       0.0       4.0
0.0       0.5       0.0       4.0
0.5       0.5       0.0       4.0
...
```

Any MC move that picks a random location inside this volume will be automatically rejected.

Example 5: Diffusion of CO_2 in Na-LTA

An example of molecular dynamics of an adsorbate (CO_2) diffusing through the pores of LTA 4A loaded with ions. The mean-square displacement is computed during the run.

```
SimulationType                  MolecularDynamics
NumberOfCycles                  250000
NumberOfInitializationCycles    5000
NumberOfEquilibrationCycles     10000
PrintEvery                      5000
RestartFile                     no

Ensemble                        NVT

Forcefield                      Local
ModifyOxgensConnectedToAluminium yes
TimeStep                        0.0005

Framework 0
FrameworkName LTA4A
RemoveAtomNumberCodeFromLabel yes
UnitCells 1 1 1
ExternalTemperature 600.0
ComputeMSD yes
PrintMSDEvery 5000

component 0 MoleculeName             sodium
            MoleculeDefinition       Local
            TranslationProbability   1.0
            ReinsertionProbability   1.0
            ExtraFrameworkMolecule   yes
            CreateNumberOfMolecules  96

component 1 MoleculeName             CO2
            MoleculeDefinition       Local
            BlockPockets             yes
            BlockPocketsFilename     LTA
            TranslationProbability   1.0
            RotationProbability      1.0
            ReinsertionProbability   1.0
            ExtraFrameworkMolecule   no
            CreateNumberOfMolecules  64
```

The examples shows that diffusion, even of small molecules, can be *very* slow in nanoporous materials. In 120 ps, the MSD is below 50 Å^2, meaning the adsorbates have not moved further than 7.1 Å. In LTA4A, the cations tend to sit in the windows separating the cages. Therefore, diffusion in these types of systems might not be accessible to MD, and techniques like Transitions State Theory (TST) have to be used.

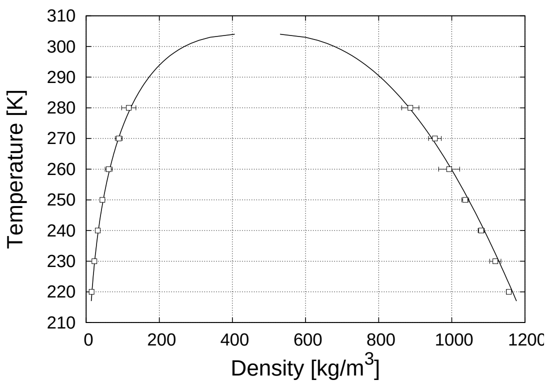


Figure 10: *Gibbs ensemble simulation of CO_2 at 250K. Two simulation boxes are used: one for the gas-branch and one for the liquid branch. The simulation can only be conducted below a certain temperature because otherwise the boxes can swap between gas and liquid. At 250K, the boxes are initialized with an equal amount of molecules, but soon split into gas and liquid. The average densities are straightforward to measure. As shown, the model for CO_2 does a good job when compare to experimental data (NIST database).*

Example 6: Diffusion of benzene in rigid IRMOF-1

Benzene (and aromatic molecules in general) are usually kept rigid. RASPA uses quaternions for the description of the orientation of the molecules. The integration schemes of Martyna and Tuckermann are symplectic and conserve energy very well. Even though the molecule is described as a center of mass and a orientation, the forces are still computed atomically. In this example the diffusivity the mean-square displacement of benzene at 298K in IRMOF-1 is computed. The forcefield is specifically optimized for iso-reticular metal-organic frameworks [10]. The force field for benzene is taken from Rai and Siepmann [11].

```
SimulationType                MolecularDynamics
NumberOfCycles                100000
NumberOfEquilibrationCycles   10000
NumberOfInitializationCycles  1000
PrintEvery                    5000
RestartFile                   no

Ensemble                      NVT

ChargeMethod                  Ewald
CutOff                        12.0
TimeStep                      0.0005
Forcefield                    Local
EwaldPrecision                1e-6

Framework 0
FrameworkName IRMOF-1
UnitCells 1 1 1
ExternalTemperature 298.0

Component 0 MoleculeName              benzene
            MoleculeDefinition        Local
            IdealGasRosenbluthWeight  1.0
            TranslationProbability    1.0
            RotationProbability       1.0
            ReinsertionProbability    1.0
            CreateNumberOfMolecules   16
```

```
Conserved energy:      -36557.3872141186 Energy drifts:  0.0000001015           0.0000059765
Temperature:             307.434 (avg.  298.597), Translational (avg.  300.133), Rotational (avg.  297.060)
Temperature Adsorbates:  307.434 (avg.  298.597), Translational (avg.  300.133), Rotational (avg.  297.060)
```

Example 7: Gibbs ensemble simulation of CO_2

The Gibbs ensemble is way of computing coexistence without interfaces. It is one the most used methods to study vapor-liquid and liquid-liquid equilibria, it is not suitable for very dense systems. The conditions for coexistence of two or more phases I, II, . . . is that the pressure and temperature of all the phases must be equal, as well as the chemical potential of all the species. The Gibbs ensemble example for the single component CO_2 is listed below. two boxes will be used, one will correspond to the liquid phase, the other one to the gas phase. The 'GibbsVolumeChange' move changes the individual volume leaving the total volume in tact, the 'GibbsSwap' move swaps particles from one box to the other. One of the practical problems is to make sure both boxes remain larger than twice the cutoff length. If not, the program will exit with an error message, and the simulation should be restarted with a bigger volume. Note that RASPA uses orientational biased insertions for small rigid molecules like CO_2. For this example about 10000-20000 cycles are needed to equilibrate properly.

```
SimulationType                MonteCarlo
NumberOfCycles                25000
NumberOfInitializationCycles  10000
PrintEvery                    1000
RestartFile                   no

Forcefield                    ExampleMoleculeForceField

Box 0
BoxLengths 30 30 30
BoxAngles 90 90 90
ExternalTemperature 240.0

Box 1
BoxLengths 30 30 30
BoxAngles 90 90 90
ExternalTemperature 240.0

GibbsVolumeChangeProbability 0.1

Component 0 MoleculeName             CO2
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   1.0
            RotationProbability      1.0
            ReinsertionProbability   1.0
            GibbsSwapProbability     1.0
            CreateNumberOfMolecules  150 150
```

The computed densities are

```
Average density component 0 [CO2]
-------------------------------------------------------------
    Block[ 0]           33.53583 [kg/m^3]
    Block[ 1]           33.36214 [kg/m^3]
    Block[ 2]           34.55914 [kg/m^3]
    Block[ 3]           34.05442 [kg/m^3]
    Block[ 4]           34.14516 [kg/m^3]
    ------------------------------------------------------------------------------
    Average             33.93134 [kg/m^3] +/-           0.60035 [kg/m^3]

Average density component 0 [CO2]
-------------------------------------------------------------
    Block[ 0]         1104.22167 [kg/m^3]
    Block[ 1]         1103.62933 [kg/m^3]
    Block[ 2]         1101.86174 [kg/m^3]
    Block[ 3]         1107.16121 [kg/m^3]
    Block[ 4]         1106.36198 [kg/m^3]
    ------------------------------------------------------------------------------
    Average           1104.64719 [kg/m^3] +/-           2.65080 [kg/m^3]
```

The experimental densities at 240K are 33.295 and 1088.9 kg/m^3 respectively.
In the high density phase, the acceptance probabilities of the reinsertion-move and Gibbs swap-move are low. The Gibbs volume change is adjusted to achieve roughly 50% acceptance.

```
Performance of the Reinsertion move:
====================================
Component [CO2] total tried: 1730548.000000 succesfull growth: 1421729.000000 (82.154843 [%]) accepted: 18572.000000 (1.073186 [%])

Performance of the Gibbs volume change move:
============================================
total tried: 346089.000000 accepted: 206146.000000 (59.564447 [%])
    maximum volume change 0.049783

Performance of the Gibbs swap move:
============================================
Component [CO2] transfered to system [0] total tried: 1729616.000000 accepted: 83910.000000 (4.851366 [%])
```

Example 8: Minimization of octane

Energy minimization is often used to understand the behavior of molecules. This example shows the input for the minimization of octane using a united-atom model[12].

```
SimulationType                Minimization
NumberOfCycles                10
NumberOfInitializationCycles  100
RestartFile                   no
PrintEvery                    100

MaximumNumberOfMinimizationSteps 1000
RMSGradientTolerance 1e-6
MaxGradientTolerance 1e-6
```

```
RemoveTranslationFromHessian yes
RemoveRotationFromHessian yes

Ensemble        NVT

Forcefield      ExampleMoleculeForceField
ChargeMethod    Coulomb
CutOffCoulomb   15.0

Box 0
BoxLengths 30 30 30
UnitCells 1 1 1
ExternalTemperature 298.0
Movies yes
WriteMoviesEvery 1

Component 0 MoleculeName             octane
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   0.5
            RotationProbability      0.5
            ReinsertionProbability   0.5
            CreateNumberOfMolecules  1
```

Note that the minimization algorithm guarantees that all positive eigenvalues are found. The first 6 eigenvalues are zero, corresponding to translation and rotation of the isolated octane molecule. A requirement is that all the forces are zero.

```
Forces
Adsorbate[0] Atom: 0  3.90833e-10 1.43088e-10 6.56558e-10
Adsorbate[0] Atom: 1  -9.72811e-10 -2.7887e-10 -7.84603e-10
Adsorbate[0] Atom: 2  5.27156e-10 6.9297e-11 3.52394e-11
Adsorbate[0] Atom: 3  -1.91779e-10 -1.80862e-10 -6.57963e-11
Adsorbate[0] Atom: 4  6.83851e-10 1.25738e-09 5.36808e-10
Adsorbate[0] Atom: 5  -1.2857e-09 -3.34806e-09 -1.38364e-09
Adsorbate[0] Atom: 6  1.11448e-09 3.29423e-09 1.90479e-09
Adsorbate[0] Atom: 7  -2.66031e-10 -9.56199e-10 -8.99352e-10
```

The minimization needs to be repeated many times from different initial conditions. Here, there 10 minimization attempts with 100 cycles of MC initialization at 298K.

```
Final energy after minimization:     479.7192896697 in 7 steps
Final energy after minimization:    -207.1377558989 in 6 steps
Final energy after minimization:     516.7656903515 in 7 steps
Final energy after minimization:     491.0018422214 in 7 steps
Final energy after minimization:     471.6063292385 in 8 steps
Final energy after minimization:    -207.1377558989 in 6 steps
Final energy after minimization:     523.2720371093 in 7 steps
Final energy after minimization:     134.0679023161 in 6 steps
Final energy after minimization:    -207.1377558989 in 6 steps
Final energy after minimization:     119.7394458177 in 6 steps
```

The results are usually "groups" of energies corresponding to a specific configuration. The lowest in energy is the stretched-out, linear configuration.

Example 9: Adsorption of hexane-isomers in MFI

Adsorption selectivities are computed by simulating a multiple-component mixture where the molecules compete for adsorption sites. In this example, the adsorption of hexane-isomers in MFI at 433 K and 2000 Pa is computed. Before running this, the ideal Rosenbluth values for the molecules at the desired temperature need to be computed (see Auxiliary examples). The identity-change move is used to try to change the identity to any type of molecule (including itself). Also partial-reinsertion moves are defined for all the adsorbates. We also define `FugacityCoefficient 1.0` for all components, which means we do a simulation at a *fugacity* (not pressure) of 2000 Pa.

```
SimulationType               MonteCarlo
NumberOfCycles               200000
NumberOfInitializationCycles 200000
PrintEvery                   5000
RestartFile                  no

ChargeMethod                 None
Forcefield                   ExampleZeolitesForceField
CutOffVDW                    12.0
RemoveAtomNumberCodeFromLabel yes

Framework            0
FrameworkName        MFI_SI
UseChargesFromCIFFile no
UnitCells            2 2 2
HeliumVoidFraction   0.29
ExternalTemperature  433.0
ExternalPressure     2000.0

Component 0 MoleculeName                  hexane
            MoleculeDefinition            Local
            IdealGasRosenbluthWeight      0.00811779
```

```
                 FugacityCoefficient            1.0
                 TranslationProbability         1.0
                 RotationProbability            1.0
                 ReinsertionProbability         1.0
                 PartialReinsertionProbability  1.0
                 IdentityChangeProbability      1.0
                   NumberOfIdentityChanges      4
                   IdentityChangesList          0 1 2 3
                 SwapProbability                1.0
                 CreateNumberOfMolecules        0

     Component 1 MoleculeName                   2-methylpentane
                 MoleculeDefinition             Local
                 IdealGasRosenbluthWeight       0.0470006
                 FugacityCoefficient            1.0
                 TranslationProbability         1.0
                 RotationProbability            1.0
                 ReinsertionProbability         1.0
                 PartialReinsertionProbability  1.0
                 IdentityChangeProbability      1.0
                   NumberOfIdentityChanges      4
                   IdentityChangesList          0 1 2 3
                 SwapProbability                1.0
                 CreateNumberOfMolecules        0

     Component 2 MoleculeName                   3-methylpentane
                 MoleculeDefinition             Local
                 IdealGasRosenbluthWeight       0.0536003
                 FugacityCoefficient            1.0
                 TranslationProbability         1.0
                 RotationProbability            1.0
                 ReinsertionProbability         1.0
                 PartialReinsertionProbability  1.0
                 IdentityChangeProbability      1.0
                   NumberOfIdentityChanges      4
                   IdentityChangesList          0 1 2 3
                 SwapProbability                1.0
                 CreateNumberOfMolecules        0

     Component 3 MoleculeName                   22-dimethylbutane
                 MoleculeDefinition             Local
                 IdealGasRosenbluthWeight       0.226526
                 FugacityCoefficient            1.0
                 TranslationProbability         1.0
                 RotationProbability            1.0
                 ReinsertionProbability         1.0
                 PartialReinsertionProbability  1.0
                 IdentityChangeProbability      1.0
                   NumberOfIdentityChanges      4
                   IdentityChangesList          0 1 2 3
                 SwapProbability                1.0
                 CreateNumberOfMolecules        0

Component 0 [hexane]
-------------------------------------------------------------
    Average loading absolute [molecules/unit cell]        1.2595631250 +/-       0.0066198404 [-]

Component 1 [2-methylpentane]
    Average loading absolute [molecules/unit cell]        0.6311325000 +/-       0.0086401923 [-]

Component 2 [3-methylpentane]
-------------------------------------------------------------
    Average loading absolute [molecules/unit cell]        0.2982025000 +/-       0.0080754705 [-]

Component 3 [22-dimethylbutane]
-------------------------------------------------------------
    Average loading absolute [molecules/unit cell]        0.1239156250 +/-       0.0013021057 [-]

     Performance of the swap addition move:
     ======================================
     Component [hexane] total tried: 343107.000000 succesfull growth: 225219.000000 (65.641039 [%]) accepted: 20776.000000 (6.055254 [%])
     Component [2-methylpentane] total tried: 341490.000000 succesfull growth: 212073.000000 (62.102258 [%]) accepted: 10655.000000 (3.120150 [%])
     Component [3-methylpentane] total tried: 342604.000000 succesfull growth: 205370.000000 (59.943842 [%]) accepted: 10411.000000 (3.038785 [%])
     Component [22-dimethylbutane] total tried: 341932.000000 succesfull growth: 193866.000000 (56.697238 [%]) accepted: 9767.000000 (2.856416 [%])

     Performance of the swap deletion move:
     ======================================
     Component [hexane] total tried: 342845.000000 succesfull growth: 342845.000000 (100.000000 [%]) accepted: 20563.000000 (5.997754 [%])
     Component [2-methylpentane] total tried: 342464.000000 succesfull growth: 341147.000000 (99.615434 [%]) accepted: 10612.000000 (3.098720 [%])
     Component [3-methylpentane] total tried: 342281.000000 succesfull growth: 313715.000000 (91.654226 [%]) accepted: 10675.000000 (3.118783 [%])
     Component [22-dimethylbutane] total tried: 342573.000000 succesfull growth: 218648.000000 (63.825228 [%]) accepted: 9761.000000 (2.849320 [%])

     Performance of the Reinsertion move:
     ====================================
     Component [hexane] total tried: 682800.000000 succesfull growth: 459936.000000 (67.360281 [%]) accepted: 9769.000000 (1.430726 [%])
     Component [2-methylpentane] total tried: 679289.000000 succesfull growth: 434051.000000 (63.897840 [%]) accepted: 4340.000000 (0.638903 [%])
     Component [3-methylpentane] total tried: 627118.000000 succesfull growth: 385061.000000 (61.401682 [%]) accepted: 4115.000000 (0.656176 [%])
     Component [22-dimethylbutane] total tried: 436366.000000 succesfull growth: 252934.000000 (57.963728 [%]) accepted: 3080.000000 (0.705830 [%])


     Performance of the partial reinsertion move:
     ============================================
     Component [hexane] total tried: 683886.000000 succesfull growth: 661852.000000 (96.778118 [%]) accepted: 158568.000000 (23.186321 [%])
     Component [2-methylpentane] total tried: 681914.000000 succesfull growth: 679831.000000 (99.694536 [%]) accepted: 256922.000000 (37.676599 [%])
     Component [3-methylpentane] total tried: 627634.000000 succesfull growth: 625746.000000 (99.699188 [%]) accepted: 176488.000000 (28.119573 [%])
     Component [22-dimethylbutane] total tried: 437324.000000 succesfull growth: 435874.000000 (99.668438 [%]) accepted: 210354.000000 (48.100264 [%])

     Performance of the identity change move:
     ======================================
     Component [hexane]->[hexane] total tried: 170814.000000 succesfull growth: 153200.000000 (89.688199 [%]) accepted: 11192.000000 (6.552156 [%])
     Component [hexane]->[2-methylpentane] total tried: 171350.000000 succesfull growth: 143781.000000 (83.910709 [%]) accepted: 5675.000000 (3.311935 [%])
     Component [hexane]->[3-methylpentane] total tried: 170612.000000 succesfull growth: 137177.000000 (80.402902 [%]) accepted: 5576.000000 (3.268234 [%])
     Component [hexane]->[22-dimethylbutane] total tried: 171284.000000 succesfull growth: 136622.000000 (79.763434 [%]) accepted: 4771.000000 (2.785432 [%])
     Component [2-methylpentane]->[hexane] total tried: 170176.000000 succesfull growth: 162611.000000 (95.554602 [%]) accepted: 5714.000000 (3.357700 [%])
     Component [2-methylpentane]->[2-methylpentane] total tried: 169789.000000 succesfull growth: 163167.000000 (96.099865 [%]) accepted: 9153.000000 (5.390809 [%])
     Component [2-methylpentane]->[3-methylpentane] total tried: 170132.000000 succesfull growth: 156426.000000 (91.943902 [%]) accepted: 4905.000000 (2.883056 [%])
     Component [2-methylpentane]->[22-dimethylbutane] total tried: 170500.000000 succesfull growth: 164951.000000 (96.745455 [%]) accepted: 8378.000000 (4.913783 [%])
     Component [3-methylpentane]->[hexane] total tried: 156737.000000 succesfull growth: 143803.000000 (91.747960 [%]) accepted: 5459.000000 (3.482904 [%])
     Component [3-methylpentane]->[2-methylpentane] total tried: 156602.000000 succesfull growth: 140396.000000 (89.651473 [%]) accepted: 4772.000000 (3.047215 [%])
     Component [3-methylpentane]->[3-methylpentane] total tried: 156796.000000 succesfull growth: 135085.000000 (86.153346 [%]) accepted: 8233.000000 (5.250772 [%])
     Component [3-methylpentane]->[22-dimethylbutane] total tried: 156055.000000 succesfull growth: 134198.000000 (85.994041 [%]) accepted: 3891.000000 (2.493352 [%])
     Component [22-dimethylbutane]->[hexane] total tried: 109564.000000 succesfull growth: 104319.000000 (95.212844 [%]) accepted: 4638.000000 (4.233142 [%])
     Component [22-dimethylbutane]->[2-methylpentane] total tried: 108833.000000 succesfull growth: 104601.000000 (96.111474 [%]) accepted: 8506.000000 (7.815644 [%])
     Component [22-dimethylbutane]->[3-methylpentane] total tried: 109359.000000 succesfull growth: 100691.000000 (92.073812 [%]) accepted: 3902.000000 (3.568065 [%])
     Component [22-dimethylbutane]->[22-dimethylbutane] total tried: 109054.000000 succesfull growth: 106124.000000 (97.313258 [%]) accepted: 7485.000000 (6.863572 [%])
```

4.4 Advanced examples

Example 1: Adsorption of CO_2 in using the Gibbs-ensemble

```
SimulationType                MonteCarlo
NumberOfCycles                25000
NumberOfInitializationCycles  5000
PrintEvery                    1000

Forcefield                    ExampleZeolitesForceField
RemoveAtomNumberCodeFromLabel yes

Framework 0
FrameworkName MFI_SI
UnitCells 2 2 2
HeliumVoidFraction 0.29
ExternalTemperature 300.0
ExternalPressure 1e4

Box 1
BoxLengths 110 110 110
CutOffCoulomb 50
ExternalTemperature 300.0
ExternalPressure 1e4

VolumeChangeProbability 0.2

Component 0 MoleculeName             methane
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   0.5
            ReinsertionProbability   0.5
            GibbsSwapProbability     1.0
            CreateNumberOfMolecules  0 100
```

The absolute loading is 2.77± 0.05 vs 2.76± 0.02 in example 6 of the basic-examples.

Example 2: Benzene diffusion in flexible IRMOF-10

Molecules with a phenyl-ring are usually quite rigid. In Monte Carlo rigid units are not a problem, because the MC moves can be developed in such a way that the constraints remain satisfied, i.e. translation and rotation of the whole rigid unit. In molecular dynamics, there are two general approaches. The first is to integrate the molecules atomically and afterwards satisfy the constraints iteratively using for example the shake algorithm. For bigger molecules complications arise, convergence becomes more difficult, and for a planar molecule like benzene additional sites above the molecule are needed. Therefore, the second approach has become more popular. Using quaternions (or Euler angles) one can describe the configurations of the molecule as a center-of-mass position and an orientation. The translation and rotation are integrated and when the forces are needed the atoms positions are computed from the com position and the orientation. The forces are then summed to the center of mass and the torque is computed. Miller et al. have developed an integration algorithm for rigid units (using quaternions) that is symplectic.

All these techniques are combined in the example of diffusion of benzene in IRMOF-10. The integration is performed in the NVT ensemble using the Nose-Hoover thermostats. Three separate Nose-Hoover chains are operating on (i) the translation, (ii) the rotation of the molecules, and (iii) on the framework.

```
SimulationType                MolecularDynamics
NumberOfCycles                1000000
NumberOfEquilibrationCycles   10000
NumberOfInitializationCycles  100
PrintEvery                    5000
RestartFile                   no

Ensemble                      NVT

ChargeMethod                  Ewald
CutOff                        12.0
TimeStep                      0.0005
```

```
Forcefield                    Local
EwaldPrecision                1e-6

Framework 0
FrameworkName IRMOF-10
UnitCells 1 1 1
ExternalTemperature 298.0
Movies no
WriteMoviesEvery 1000

FrameworkDefinitions Local
FlexibleFramework yes

Component 0 MoleculeName              benzene
            MoleculeDefinition        ExampleDefinitions
            IdealGasRosenbluthWeight  1.0
            TranslationProbability    1.0
            RotationProbability       1.0
            ReinsertionProbability    1.0
            CreateNumberOfMolecules   16


#number of pseudo atoms
12
#type      print   as    chem  oxidation   mass        charge   polarization B-factor radii  connectivity anisotropic anisotropic-type   tinker-type
Zn1        yes     Zn    Zn    0           65.37       1.275    0.0          1.0      1.448  0            0           relative           0
O1         yes     O     O     0           15.9994    -1.5      0.0          1.0      0.68   2            0           relative           0
O2         yes     O     O     0           15.9994    -0.6      0.0          1.0      0.68   2            0           relative           0
C1         yes     C     C     0           12.0107     0.475    0.0          1.0      0.720  0            0           relative           0
C2         yes     C     C     0           12.0107     0.125    0.0          1.0      0.720  0            0           relative           0
C3         yes     C     C     0           12.0107    -0.15     0.0          1.0      0.720  0            0           relative           0
C4         yes     C     C     0           12.0107    -0.15     0.0          1.0      0.720  0            0           relative           0
C5         yes     C     C     0           12.0107     0.0      0.0          1.0      0.720  0            0           relative           0
H1         yes     H     H     0           1.00794     0.15     0.0          1.0      0.320  0            0           relative           0
H2         yes     H     H     0           1.00794     0.15     0.0          1.0      0.320  0            0           relative           0
C_benz     yes     C     C     0           12.0       -0.095    0.0          1.0      0.70   0            0           relative           0
H_benz     yes     H     H     0           1.00794     0.095    0.0          1.0      0.320  0            0           relative           0

# general rule for shifted vs truncated
shifted
# general rule tailcorrections
no
# number of defined interactions
12
# type interaction
Zn1            lennard-jones    0.42      2.7        // D. Dubbeldam, K.S. Walton, D.E. Ellis, R.Q. Snurr, Angew. Chem. Int. Ed. 2007, 46, 4496-4499.
O1             lennard-jones    700.0     2.98       // idem
O2             lennard-jones    70.5      3.11       // idem
C1             lennard-jones    47.0      3.74       // idem
C2             lennard-jones    47.86     3.47       // idem
C3             lennard-jones    47.86     3.47       // idem
C4             lennard-jones    47.86     3.47       // idem
C5             lennard-jones    47.86     3.47       // idem
H1             lennard-jones    7.65      2.85       // idem
H2             lennard-jones    7.65      2.85       // idem
C_benz         lennard-jones    30.70     3.60       // N. Rai and J.I. Siepmann, J. Phys. Chem. B 2007, 111, 10790-10799.
H_benz         lennard-jones    25.45     2.36       // idem
# general mixing rule for Lennard-Jones
Lorentz-Berthelot


#CoreShells bond  BondDipoles UreyBradley bend  inv  tors improper-torsion bond/bond bond/bend bend/bend stretch/torsion bend/torsion
          0    8           0           0   12    0    14                5         0         0         0               0            0
#bond stretch atom n1-n2, equilibrium distance, bondforce-constant
O2 C1  HARMONIC_BOND  543840.64928424  1.25
C1 C2  HARMONIC_BOND  353750.919316375 1.42
C2 C3  HARMONIC_BOND  483413.91047488  1.36
C3 C4  HARMONIC_BOND  483413.91047488  1.36
C4 C5  HARMONIC_BOND  483413.91047488  1.36
C3 H1  HARMONIC_BOND  366001.13136396  0.95
C4 H2  HARMONIC_BOND  366001.13136396  0.95
C5 C5  HARMONIC_BOND  483413.91047488  1.36
#bond bending atom n1-n2-n3, equilibrium angle, bondforce-constant
O2 C1 O2 HARMONIC_BEND 135960.162321060 130.0
O2 C1 C2 HARMONIC_BEND 54882.4848123699 115.0
C1 C2 C3 HARMONIC_BEND 34926.5543205787 120.0
C2 C3 C4 HARMONIC_BEND 90640.10821404 120.0
C3 C4 C5 HARMONIC_BEND 90640.10821404 120.0
C3 C2 C3 HARMONIC_BEND 90640.10821404 120.0
C4 C5 C4 HARMONIC_BEND 90640.10821404 120.0
C4 C5 C5 HARMONIC_BEND 90640.10821404 120.0
C4 C3 H1 HARMONIC_BEND 37263.15559911 120.0
C2 C3 H1 HARMONIC_BEND 37263.15559911 120.0
C3 C4 H2 HARMONIC_BEND 37263.15559911 120.0
C5 C4 H2 HARMONIC_BEND 37263.15559911 120.0
#torsion atom n1-n2-n3-n4,
O2 C1 C2 C3  TRAPPE_DIHEDRAL 0.0 0.0 1258.890391861 0.0
C1 C2 C3 C4  TRAPPE_DIHEDRAL 0.0 0.0 1510.668470234 0.0
C1 C2 C3 H1  TRAPPE_DIHEDRAL 0.0 0.0 1510.668470234 0.0
C2 C3 C4 C5  TRAPPE_DIHEDRAL 0.0 0.0 1510.668470234 0.0
C2 C3 C4 H2  TRAPPE_DIHEDRAL 0.0 0.0 1510.668470234 0.0
C3 C4 C5 C4  TRAPPE_DIHEDRAL 0.0 0.0 1510.668470234 0.0
C3 C4 C5 C5  TRAPPE_DIHEDRAL 0.0 0.0 1510.668470234 0.0
C3 C2 C3 C4  TRAPPE_DIHEDRAL 0.0 0.0 1510.668470234 0.0
C3 C2 C3 H1  TRAPPE_DIHEDRAL 0.0 0.0 1510.668470234 0.0
C4 C5 C4 H2  TRAPPE_DIHEDRAL 0.0 0.0 1510.668470234 0.0
C4 C5 C5 C4  TRAPPE_DIHEDRAL 0.0 0.0 1510.668470234 0.0
C5 C4 C3 H1  TRAPPE_DIHEDRAL 0.0 0.0 1510.668470234 0.0
C5 C5 C4 H2  TRAPPE_DIHEDRAL 0.0 0.0 1510.668470234 0.0
H2 C4 C3 H1  TRAPPE_DIHEDRAL 0.0 0.0 1510.668470234 0.0
# improper torsion atom n1-n2-n3-n4,
C1 C2 C3  C3  TRAPPE_IMPROPER_DIHEDRAL 0.0 0.0 5035.561567446 0.0
C2 C1 O2  O2  TRAPPE_IMPROPER_DIHEDRAL 0.0 0.0 5035.561567446 0.0
C4 C5 C4  C5  TRAPPE_IMPROPER_DIHEDRAL 0.0 0.0 186.3157779955 0.0
```

```
C3 C4 C5  H2  TRAPPE_IMPROPER_DIHEDRAL 0.0 0.0 186.3157779955 0.0
C2 C3 C4  H1  TRAPPE_IMPROPER_DIHEDRAL 0.0 0.0 186.3157779955 0.0


Conserved energy:    -3219446.1895550787 Energy drifts:  0.0000622481           0.0000214757
Temperature:               290.106 (avg.  298.454), Translational (avg.  298.503), Rotational (avg.  296.360)
Temperature Framework:     289.372 (avg.  298.035)
Temperature Adsorbates:    296.272 (avg.  297.806), Translational (avg.  299.251), Rotational (avg.  296.360)
Cell temperature:    0.000 (avg.    0.000)
Current total kinetic energy:           251459.2049205256 [K]
Current total Nose-Hoover energy:       125636.8930809353 [K]
Current total potential energy:         -3672988.8242843263 [K]  (avg.    -3659267.1764687141)
    Current Host-Host energy:              -4130487.5188890938 [K]  (avg.    -4121521.1949522868)
    Current Host-Adsorbate energy:           -47517.6323708648 [K]  (avg.      -48952.8846306008)
    Current Host-Cation energy:                  0.0000000000 [K]  (avg.           0.0000000000)
    Current Adsorbate-Adsorbate energy:       -3346.3844134994 [K]  (avg.       -2945.2800260990)
    Current Cation-Cation energy:                0.0000000000 [K]  (avg.           0.0000000000)
    Current Adsorbate-Cation energy:             0.0000000000 [K]  (avg.           0.0000000000)
        Current Host-Bond energy:                  238316.7857221146 [K]  (avg.      245308.7523960742)
        Current Host-Bend energy:                  138107.4824254393 [K]  (avg.      135190.5831477965)
        Current Host-Torsion energy:               122909.7354270181 [K]  (avg.      124102.0477715168)
        Current Host-Improper torsion energy:        9028.7078145591 [K]  (avg.        9550.7998248921)
```

Example 3: NPT molecular dynamics of flexible IRMOF-1

An NPT-ensemble simulation of a flexible framework IRMOF-1. This type of simulation can be used to compute the average unit cell size at the desired temperature and pressure (and properties like the 'volumetric expansion coefficient' etc). The equilibration, although slow, is very much faster than Monte Carlo. The example show the code for flexible IRMOF-1 at 298K and 1 atm.

```
SimulationType              MolecularDynamics
NumberOfCycles              500000
NumberOfEquilibrationCycles 5000
PrintEvery                  5000
RestartFile                 no

Ensemble                    NPT

Forcefield                  Dubbeldam2007FlexibleIRMOF-1
CutOff                      12.0

Framework 0
FrameworkName IRMOF-1
UnitCells 1 1 1
ExternalTemperature 298.0
ExternalPressure 101325.0

FlexibleFramework yes
FrameworkDefinitions Dubbeldam2007FlexibleIRMOF-1


Average Volume:
=================
    Block[ 0]       17177.52409 [A^3]
    Block[ 1]       17138.06375 [A^3]
    Block[ 2]       17150.92159 [A^3]
    Block[ 3]       17123.83896 [A^3]
    Block[ 4]       17161.56950 [A^3]
    ------------------------------------------------------------------------------
    Average         17150.38358 [A^3] +/-          25.73171 [A^3]

Average Box-lengths:
====================
    Block[ 0]          25.80193 [A^3]
    Block[ 1]          25.78216 [A^3]
    Block[ 2]          25.78860 [A^3]
    Block[ 3]          25.77502 [A^3]
    Block[ 4]          25.79394 [A^3]
    ------------------------------------------------------------------------------
    Average Box.ax           25.78833 [A^3] +/-           0.01290 [A^3]
```

Example 4: Adsorption of CO_2 in fully-flexible IRMOF-1 (μVT-ensemble)

Flexibility in MOFs is more important than in zeolites. A very efficient move to change the whole framework (and actually also the adsorbates) is have a short NVE MD-run and accept or reject the new configuration. This hybrid MD/MC move can be switched on using 'HybridMCMDMoveProbability [real]', where [real] is the fraction of the move at each cycle.

```
SimulationType               MonteCarlo
NumberOfCycles               50000
NumberOfInitializationCycles 10000
PrintEvery                   5000

ChargeMethod                 Ewald
CutOff                       12.0
Forcefield                   Dubbeldam2007FlexibleIRMOF-1
EwaldPrecision               1e-6
TimeStep                     0.0005
```

```
Framework 0
FrameworkName IRMOF-1
UnitCells 1 1 1
HeliumVoidFraction 0.801937
ExternalTemperature 298.0
ExternalPressure 1e5

FrameworkDefinitions Dubbeldam2007FlexibleIRMOF-1
FlexibleFramework yes

HybridNVEMoveProbability 1.0
  NumberOfHybridNVESteps 5

Component 0 MoleculeName             CO2
            MoleculeDefinition       ExampleDefinitions
            IdealGasRosenbluthWeight  1.0
            TranslationProbability    1.0
            RotationProbability       1.0
            ReinsertionProbability    1.0
            SwapProbability           1.0
            CreateNumberOfMolecules   0
```

```
Component 0 [CO2]
-------------------------------------------------------------
    Block[ 0] 6.06650           [-]
    Block[ 1] 6.02670           [-]
    Block[ 2] 5.93730           [-]
    Block[ 3] 6.02350           [-]
    Block[ 4] 6.06040           [-]
    ------------------------------------------------------------------------------
    Average loading absolute                              6.0228800000 +/-       0.0640569684 [-]
    Average loading absolute [molecules/unit cell]        6.0228800000 +/-       0.0640569684 [-]
    Average loading absolute [mol/kg framework]                  0.9779634386 +/-       0.0104012322 [-]
    Average loading absolute [milligram/gram framework]         43.0292177430 +/-       0.4576417333 [-]
    Average loading absolute [cm^3 (STP)/gr framework]          21.9200487952 +/-       0.2331329652 [-]
    Average loading absolute [cm^3 (STP)/cm^3 framework]        13.0046306040 +/-       0.1383121052 [-]
```

Example 5: CO_2 adsorption in flexible IRMOF-1 (osmotic ensemble).

Adsorption simulations using a flexible framework are very computationally demanding, the current example will probably run about a week. The equilibration is very important and it is best to start with a restart-file obtained from the previous example at the same temperature. The directory 'Restart' produced in the previous example should be copied to 'RestartInitial' and the option 'RestartFile' should be set to 'yes'.

```
SimulationType                MonteCarlo
NumberOfCycles                50000
NumberOfInitializationCycles  10000
PrintEvery                    5000
RestartFile                   no

ChargeMethod                  Ewald
CutOff                        12.0
Forcefield                    Dubbeldam2007FlexibleIRMOF-1
EwaldPrecision                1e-6
TimeStep                      0.0005

Framework 0
FrameworkName IRMOF-1
UnitCells 1 1 1
HeliumVoidFraction 0.801937
ExternalTemperature 298.0
ExternalPressure 1e5

FrameworkDefinitions Dubbeldam2007FlexibleIRMOF-1
FlexibleFramework yes

HybridNVEMoveProbability 1.0
  NumberOfHybridNVESteps 5
VolumeChangeProbability  1.0

Component 0 MoleculeName             CO2
            MoleculeDefinition       ExampleDefinitions
            IdealGasRosenbluthWeight  1.0
            TranslationProbability    1.0
            RotationProbability       1.0
            ReinsertionProbability    1.0
            SwapProbability           1.0
            CreateNumberOfMolecules   0
```

```
Component 0 [CO2]
-------------------------------------------------------------
    Block[ 0] 5.97360           [-]
    Block[ 1] 6.00530           [-]
    Block[ 2] 6.07740           [-]
    Block[ 3] 6.08170           [-]
    Block[ 4] 5.99290           [-]
    ------------------------------------------------------------------------------
    Average loading absolute                              6.0261800000 +/-       0.0621171422 [-]
    Average loading absolute [molecules/unit cell]        6.0261800000 +/-       0.0621171422 [-]
    Average loading absolute [mol/kg framework]                  0.9784992752 +/-       0.0100862534 [-]
    Average loading absolute [milligram/gram framework]         43.0527939090 +/-       0.4437830465 [-]
    Average loading absolute [cm^3 (STP)/gr framework]          21.9320590230 +/-       0.2260730393 [-]
    Average loading absolute [cm^3 (STP)/cm^3 framework]        13.0117559794 +/-       0.1341236232 [-]
```

Example 6: Minimization of a flexible framework (fixed volume)

Physically, energy minimization corresponds to an instantaneous freezing of the system; a static structure in which no atom feels a net force corresponds to a temperature of 0 K. In the early 1980's, energy minimization was about all one could afford to do and was dubbed 'molecular mechanics.' Here, a difficult optimization problem: a flexible framework, IRMOF-1, in a periodic unit cell, with many low energy modes. The energy landscape of a framework is very complex. A true minimum is characterized by all positive eigenvalues of the Hessian matrix (the matrix of second derivatives with respect to position). A zero eigenvalue means that moving in the direction of the associated eigenvector does not result in a change in energy. Likewise, a negative and positive eigenvalue means an decrease and increase in energy, respectively. Most of the optimization time is spent on reaching a zero curvature structure, i.e. all positive eigenvalues.

```
SimulationType  Minimization
NumberOfCycles  1
RestartFile     no
PrintEvery      1

MaximumNumberOfMinimizationSteps 1000
RMSGradientTolerance 1e-6
MaxGradientTolerance 1e-6

Ensemble        NVT

Forcefield                                 Dubbeldam2007FlexibleIRMOF-1
ChargeMethod                               Ewald
EwaldPrecision                             1e-10
InternalFrameworkLennardJonesInteractions  yes

Framework 0
FrameworkName IRMOF-1
UnitCells 1 1 1
ExternalTemperature 298.0
Movies yes
WriteMoviesEvery 1

FlexibleFramework     yes
FrameworkDefinitions  Dubbeldam2007FlexibleIRMOF-1
```

The minimization needs 145 cycles to optimize IRMOF-1, the last steps are shown here. The convergence is very rapid (quadratic) near the minimum, and the minimum energy can be reached up to arbitrary precision (the forces on all the atoms are 1×10^{-8} K/Å^2 or smaller). To compute spectra, frequencies and/or eigenmodes a high precision is needed.

```
Starting configuration:
      Box:      25.8320000000       0.0000000000       0.0000000000     Strain derivative:  129316.6369463501       0.0000000016       0.0000000015
                 0.0000000000      25.8320000000       0.0000000000                              0.0000000015  129316.6369463478       0.0000000016
                 0.0000000000       0.0000000000      25.8320000000                              0.0000000016       0.0000000015  129316.6369463608


Beginning Baker minimization:
-----------------------------
Computing generalized Hessian matrix
Projecting constraints from generalized Hessian matrix
Computing eigenvalues and vectors
Shifting parameter: -144554.0726719970 Lowest eigenvalue:   -1933.2863574224
Iteration: 0 Energy: -4210329.1678245096 Volume:   17237.4927303680 RMS gradient: 254.796  Max gradient: 16621.8 Number of negative eigenvalues: 30 Number of zero eigenvalues: 3
      Box:      25.8320000000       0.0000000000       0.0000000000     Strain derivative:  129217.9969168001       0.0000000015       0.0000000020
                 0.0000000000      25.8320000000       0.0000000000                              0.0000000014  129217.9969168052       0.0000000014
                 0.0000000000       0.0000000000      25.8320000000                              0.0000000020       0.0000000014  129217.9969168293
      Lengths:      25.8320000000      25.8320000000      25.8320000000, Angles:      90.0000000000      90.0000000000      90.0000000000

Computing generalized Hessian matrix
Projecting constraints from generalized Hessian matrix
Computing eigenvalues and vectors
Shifting parameter:  -54112.1867514350 Lowest eigenvalue:    -942.0192504639
Iteration: 1 Energy: -4291340.7600458413 Volume:   17237.4927303680 RMS gradient: 132.812  Max gradient: 9119.09 Number of negative eigenvalues: 30 Number of zero eigenvalues: 3
      Box:      25.8320000000       0.0000000000       0.0000000000     Strain derivative: -131849.2753307962       0.0000000004       0.0000000006
                 0.0000000000      25.8320000000       0.0000000000                              0.0000000004 -131849.2753308120       0.0000000007
                 0.0000000000       0.0000000000      25.8320000000                              0.0000000007       0.0000000007 -131849.2753307976
      Lengths:      25.8320000000      25.8320000000      25.8320000000, Angles:      90.0000000000      90.0000000000      90.0000000000

Computing generalized Hessian matrix
Projecting constraints from generalized Hessian matrix
Computing eigenvalues and vectors
Shifting parameter:   -6990.7022818167 Lowest eigenvalue:    -218.8977429655
Iteration: 2 Energy: -4325484.5920118261 Volume:   17237.4927303680 RMS gradient: 38.7493  Max gradient: 2721.17 Number of negative eigenvalues: 36 Number of zero eigenvalues: 3
      Box:      25.8320000000       0.0000000000       0.0000000000     Strain derivative: -282267.1212960631       0.0000000008       0.0000000006
                 0.0000000000      25.8320000000       0.0000000000                              0.0000000007 -282267.1212960599       0.0000000008
                 0.0000000000       0.0000000000      25.8320000000                              0.0000000006       0.0000000008 -282267.1212960533
      Lengths:      25.8320000000      25.8320000000      25.8320000000, Angles:      90.0000000000      90.0000000000      90.0000000000


..........................................................................
Computing generalized Hessian matrix
Projecting constraints from generalized Hessian matrix
Computing eigenvalues and vectors
Shifting parameter:      -0.0000000000 Lowest eigenvalue:       3.4802598329
Iteration: 145 Energy: -4331230.0933064902 Volume:   17237.4927303680 RMS gradient: 4.26892e-09  Max gradient: 8.50378e-07 Number of negative eigenvalues: 0 Number of zero eigenvalues: 3
      Box:      25.8320000000       0.0000000000       0.0000000000     Strain derivative: -114964.7683996162      -0.0000117235       0.0000000038
                 0.0000000000      25.8320000000       0.0000000000                             -0.0000117235 -114964.7682995356      -0.0000000023
```

```
                0.0000000000       0.0000000000       25.8320000000                              0.0000000038      -0.0000000023 -114964.7682508142
        Lengths:      25.8320000000       25.8320000000       25.8320000000, Angles:      90.0000000000       90.0000000000       90.0000000000

    SUCCES: RMS Gradient tolerance 1e-06 reached (4.26892e-09)
            Max Gradient tolerance 1e-06 reached (8.50378e-07)
```

The shifting values are always lower than the lowest eigenvalues, both are negative and approach zero. At iteration 2, the lowest eigenvalues is closer to zero, but still the amount of negative eigenvalues is 6 higher. Also increases in energy can occur. However, eventually the system is driven to all positive eigenvalues (a true energy minimum without saddle points) and the lowest energy. Note that minimization the structure in constant volume results in a finite (non-zero) stress. Minimization taking volume and shape changes into account are usually easier, because the system is less constrained. If one would like to also minimize the cell volume (isotropicly) use

```
    Ensemble                    NPT
```

or use for a change in cell-lengths and cell-angles

```
    Ensemble                    NPTPR
```

Example 7: Minimization of a flexible framework and elastic constants

```
    SimulationType  Minimization
    NumberOfCycles  1
    RestartFile     no
    PrintEvery      1

    MaximumNumberOfMinimizationSteps 1000
    RMSGradientTolerance 1e-6
    MaxGradientTolerance 1e-6

    Ensemble NPTPR
    NPTPRCellType RegularUpperTriangle

    ComputeElasticConstants yes

    Forcefield                             Dubbeldam2007FlexibleIRMOF-1
    ChargeMethod                           Ewald
    EwaldPrecision                         1e-10
    InternalFrameworkLennardJonesInteractions  yes

    Framework 0
    FrameworkName IRMOF-1
    UnitCells 1 1 1
    ExternalTemperature 298.0
    Movies yes
    WriteMoviesEvery 1

    FlexibleFramework     yes
    FrameworkDefinitions  Dubbeldam2007FlexibleIRMOF-1

    Final energy after minimization: -4332572.8369031781 in 8 steps
    Final pressure after minimization:       0.0000393183 [Pa]
    Final stress after minimization: {{0.000046,0.000002,-0.000001},{-,0.000027,0.000002},{-,-,0.000046}} [Pa]


    Volume [A^3]:   17506.0601028158
          Box:      25.9654670251       0.0000000000      -0.0000000000     Strain derivative:      -0.0000000481      -0.0000000018       0.0000000008
                     0.0000000000      25.9654670251       0.0000000000                             -0.0000000018      -0.0000000283      -0.0000000025
                     0.0000000000       0.0000000000      25.9654670251                              0.0000000009      -0.0000000025      -0.0000000480
          Final Lengths:      25.9654670251       25.9654670251       25.9654670251, Angles:      90.0000000000       90.0000000000       90.0000000000

    Elastic constant (Voigt notation) [GPa]
    ---------------------------------------
        29.30276     11.91226     11.91226     -0.00000      0.00000     -0.00000
        11.91226     29.30276     11.91226     -0.00000      0.00000     -0.00000
        11.91226     11.91226     29.30276      0.00000     -0.00000     -0.00000
         0.00000      0.00000      0.00000      0.98480     -0.00000      0.00000
         0.00000      0.00000      0.00000      0.00000      0.98480     -0.00000
         0.00000      0.00000     -0.00000     -0.00000      0.00000      0.98480
```

Example 8: Reaction ensemble

As an example, the industrially important propene metathesis is described by three equilibrium reactions

- 2 $C_3H_6 \leftrightarrow C_2H_4$ + trans-C_4H_8
- 2 $C_3H_6 \leftrightarrow C_2H_4$ + cis-C_4H_8
- cis-$C_4H_8 \leftrightarrow$ trans-C_4H_8

Only two reactions are independent and need to be included. In addition to the MC moves associated with simulating a chosen ensemble, also "reaction" moves are performed:

1. randomly choose a reaction,
2. randomly choose whether to do a forward or backward reaction (this determines the "reactant" and "product" molecule types),
3. randomly select the reactant molecules and remove them from the system,
4. insert the product molecules at random positions,
5. accept or reject the reaction step with the appropriate acceptance probability.

```
SimulationType                MC
NumberOfCycles                100000
NumberOfInitializationCycles  0
NumberOfEquilibrationCycles   25000
RestartFile                   no
PrintEvery                    1000

ChargeMethod                  none
Forcefield                    local
CutOff                        12.0
EwaldPrecision                1e-6

Box 0
BoxLengths 150 150 150
ExternalTemperature 450.0
ExternalPressure 101300.0
CutOff  14.0
ComputeNumberOfMoleculesHistogram yes
WriteNumberOfMoleculesHistogramEvery 5000

Reaction 2 0 0 0 0 1 0 1
Reaction 0 0 0 1 0 0 1 0

ProbabilityCFCRXMCLambdaChangeMove 1.0
VolumeChangeProbability            0.1

Component 0 MoleculeName             propene
            MoleculeDefinition       ExampleDefinitions
            LnPartitionFunction      87.1384
            TranslationProbability   35.0
            RotationProbability      53.9
            ReinsertionProbability   10.0
            ExtraFrameworkMolecule   no
            CreateNumberOfMolecules  400

Component 1 MoleculeName             ethene
            MoleculeDefinition       ExampleDefinitions
            LnPartitionFunction      82.0298
            TranslationProbability   35.0
            RotationProbability      53.9
            ReinsertionProbability   10.0
            ExtraFrameworkMolecule   no
            CreateNumberOfMolecules  0

Component 2 MoleculeName             cis-2-butene
            MoleculeDefinition       ExampleDefinitions
            LnPartitionFunction      89.0386
            TranslationProbability   35.0
            RotationProbability      53.9
            ReinsertionProbability   10.0
            ExtraFrameworkMolecule   no
            CreateNumberOfMolecules  0

Component 3 MoleculeName             trans-2-butene
            MoleculeDefinition       ExampleDefinitions
            LnPartitionFunction      89.4937
            TranslationProbability   35.0
            RotationProbability      53.9
            ReinsertionProbability   10.0
            ExtraFrameworkMolecule   no
            CreateNumberOfMolecules  0
```

Reactions are given as a list of stoichometries for the reactants and then the products, so there should two times the number-of-components integer numbers.

In the output you will see for each PrintEvery the number of integer, fractional, and reaction molecules. For each reaction the biasing factors are listed.

```
Reactions:
------------------------------------------------------------------------------------------------------------------------------------------------------
Reaction 0, current Lambda:       0.2408690700, maximum Lambda-change:       1.0000000000
        Fractional molecules:  15 (ethene) 93 (trans-2-butene) <--> 31 (propene) 134 (propene)
        Biasing Factors: 0.000000 0.076250 0.005000 0.018750 -0.028750 0.000625 0.024375 0.023125 0.052500 0.066250
                         -0.001250 0.003750 -0.069375 -0.011875 -0.015625 0.043750 -0.023750 0.026250 0.009375 -0.016875
                         -0.043750
Reaction 1, current Lambda:       0.1549750044, maximum Lambda-change:       1.0000000000
        Fractional molecules:  319 (cis-2-butene) <--> 225 (trans-2-butene)
        Biasing Factors: 0.000000 0.046875 -0.013125 -0.008125 0.037500 0.009375 0.025625 0.069375 0.003750 -0.014375
                         0.005000 0.045625 0.016250 0.045625 -0.023750 -0.049375 -0.009375 0.028125 0.002500 -0.003750
                         0.016875

Amount of molecules per component:
```

```
-----------------------------------------------------------------------------------------------------------------------------------------
Component 0 (propene), current number of integer/fractional/reaction molecules: 246/0/2 (average 244.20869/  0.00000), density:   0.60310 (average   0.68690) kg/m^3]
Component 1 (ethene), current number of integer/fractional/reaction molecules: 77/0/1 (average  77.89566/  0.00000), density:   0.12585 (average   0.14607) kg/m^3]
Component 2 (cis-2-butene), current number of integer/fractional/reaction molecules: 31/0/1 (average  30.44928/  0.00000), density:   0.10133 (average   0.11419) kg/m^3]
Component 3 (trans-2-butene), current number of integer/fractional/reaction molecules: 46/0/2 (average  47.44638/  0.00000), density:   0.15037 (average   0.17794) kg/m^3]
-----------------------------------------------------------------------------------------------------------------------------------------
```

At the end of the output, after the run has finished, the statistics of the RXMC are printed:

```
Performance of the Reaction MC lambda move:
===========================================
Reaction [0] total tried: 221218.000000 constant-lambda accepted: 212793.000000 (96.191540 [%])
               total tried: 91700.000000 forward-reaction accepted: 82910.000000 (90.414395 [%])
               total tried: 93319.000000 backward-reaction accepted: 82829.000000 (88.758988 [%])

Reaction [1] total tried: 220399.000000 constant-lambda accepted: 212964.000000 (96.626573 [%])
               total tried: 92618.000000 forward-reaction accepted: 83044.000000 (89.662916 [%])
               total tried: 92537.000000 backward-reaction accepted: 83009.000000 (89.703578 [%])
```

Example 9: CO2 in MFI (Continuous Fractional Component Monte Carlo)

A mixture simulation of CO2 and N2 in DMOF. The charges of DMOF are listed in the CIF-File using the '_atom_site_charge' keyword, and RASPA makes use of these using the keyword 'UseChargesFromCIFFile yes'. The CFCMC method is switched on by using 'CFSwapLambdaProbability' (instead of 'SwapProbability') to swap molecules in and out of the system at a fixed fugacity. The biasing factors are measured using Wang-Landau sampling during 'NumberOfEquilibrationCycles 50000'.

```
SimulationType                MonteCarlo
NumberOfCycles                200000
NumberOfInitializationCycles  50000
NumberOfEquilibrationCycles   50000
PrintEvery                    5000
RestartFile                   no

ChargeMethod                  Ewald
Forcefield                    ExampleMoleculeForceField
CutOffVDW                     12.0
RemoveAtomNumberCodeFromLabel yes

Framework            0
FrameworkName        MFI_SI
UseChargesFromCIFFile no
UnitCells            2 2 2
HeliumVoidFraction   0.29
ExternalTemperature  353
ExternalPressure     10e6

Component 0 MoleculeName                CO2
            MoleculeDefinition          ExampleDefinitions
            IdealGasRosenbluthWeight    1.0
            FugacityCoefficient         1.0
            TranslationProbability      1.0
            RotationProbability         1.0
            ReinsertionProbability      1.0
            PartialReinsertionProbability 0.0
            IdentityChangeProbability   0.0
            SwapProbability             0.0
            CFSwapLambdaProbability     1.0
            CreateNumberOfMolecules     0
```

The performance of the CFCMC is written at the end of the output file (after the run has finished). The biasing factors have lead to relatively flat distribution of Lambda. The efficiency of insertion is much higher (sometimes dramatically higher) than using conventional MC or even CBMC.

```
Component 0 [CO2]
-------------------------------------------------------------
    Block[ 0] 115.58143          [-]
    Block[ 1] 114.49658          [-]
    Block[ 2] 114.85775          [-]
    Block[ 3] 115.32147          [-]
    Block[ 4] 113.71846          [-]
    ------------------------------------------------------------------------------
    Average loading absolute                           114.7951373274 +/-       0.9096558820 [-]
    Average loading absolute [molecules/unit cell]      14.3493921659 +/-       0.1137069852 [-]
    Average loading absolute [mol/kg framework]                2.4877183185 +/-       0.0197130963 [-]
    Average loading absolute [milligram/gram framework]        109.4566207527 +/-       0.8673525831 [-]
    Average loading absolute [cm^3 (STP)/gr framework]          55.7596580580 +/-       0.4418488632 [-]
    Average loading absolute [cm^3 (STP)/cm^3 framework]       100.1634383150 +/-       0.7937118500 [-]

    Block[ 0] 77.45147           [-]
    Block[ 1] 76.36662           [-]
    Block[ 2] 76.72779           [-]
    Block[ 3] 77.19151           [-]
    Block[ 4] 75.58850           [-]
    ------------------------------------------------------------------------------
    Average loading excess                           76.6651768099 +/-       0.9096558820 [-]
    Average loading excess [molecules/unit cell]       9.5831471012 +/-       0.1137069852 [-]
    Average loading excess [mol/kg framework]                  1.6614063033 +/-       0.0197130963 [-]
    Average loading excess [milligram/gram framework]         73.0998836570 +/-       0.8673525831 [-]
    Average loading excess [cm^3 (STP)/gr framework]          37.2387205887 +/-       0.4418488632 [-]
    Average loading excess [cm^3 (STP)/cm^3 framework]        66.8934929396 +/-       0.7937118500 [-]
```

```
Performance of the CFCMC swap lambda move:
=========================================
Component [CO2] total tried: 3846494.000000 constant-lambda accepted: 1923421.000000 (50.004524 [%])
               total tried: 975099.000000 insert-lambda accepted: 215827.000000 (22.133855 [%])
               total tried: 966875.000000 remove-lambda accepted: 215832.000000 (22.322637 [%])

    Lambda probabilities:
    ---------------------
    Lambda [ 0.000000 - 0.047619 ]:       0.0462590756, Boltzmann:       0.0911221461  (biasing factor:       0.7344726563 [k_BT])
    Lambda [ 0.047619 - 0.095238 ]:       0.0480085685, Boltzmann:       0.0766660842  (biasing factor:       0.9443359375 [k_BT])
    Lambda [ 0.095238 - 0.142857 ]:       0.0490226657, Boltzmann:       0.0515327816  (biasing factor:       1.3624804687 [k_BT])
    Lambda [ 0.142857 - 0.190476 ]:       0.0472608045, Boltzmann:       0.0292034997  (biasing factor:       1.8938085937 [k_BT])
    Lambda [ 0.190476 - 0.238095 ]:       0.0484703415, Boltzmann:       0.0170042890  (biasing factor:       2.4599023438 [k_BT])
    Lambda [ 0.238095 - 0.285714 ]:       0.0476399033, Boltzmann:       0.0120588340  (biasing factor:       2.7862890625 [k_BT])
    Lambda [ 0.285714 - 0.333333 ]:       0.0467011560, Boltzmann:       0.0101492084  (biasing factor:       2.9387890625 [k_BT])
    Lambda [ 0.333333 - 0.380952 ]:       0.0473834510, Boltzmann:       0.0098690128  (biasing factor:       2.9812890625 [k_BT])
    Lambda [ 0.380952 - 0.428571 ]:       0.0476798757, Boltzmann:       0.0102887016  (biasing factor:       2.9458789063 [k_BT])
    Lambda [ 0.428571 - 0.476190 ]:       0.0471400926, Boltzmann:       0.0113810076  (biasing factor:       2.8335937500 [k_BT])
    Lambda [ 0.476190 - 0.523810 ]:       0.0465315928, Boltzmann:       0.0130231304  (biasing factor:       2.6858203125 [k_BT])
    Lambda [ 0.523810 - 0.571429 ]:       0.0466784923, Boltzmann:       0.0154473527  (biasing factor:       2.5182617188 [k_BT])
    Lambda [ 0.571429 - 0.619048 ]:       0.0470303325, Boltzmann:       0.0188036023  (biasing factor:       2.3291601563 [k_BT])
    Lambda [ 0.619048 - 0.666667 ]:       0.0475317325, Boltzmann:       0.0231419138  (biasing factor:       2.1321679688 [k_BT])
    Lambda [ 0.666667 - 0.714286 ]:       0.0477450339, Boltzmann:       0.0294814031  (biasing factor:       1.8945312500 [k_BT])
    Lambda [ 0.714286 - 0.761905 ]:       0.0480476427, Boltzmann:       0.0380011621  (biasing factor:       1.6469921875 [k_BT])
    Lambda [ 0.761905 - 0.809524 ]:       0.0482210408, Boltzmann:       0.0502161310  (biasing factor:       1.3718750000 [k_BT])
    Lambda [ 0.809524 - 0.857143 ]:       0.0474246672, Boltzmann:       0.0676212061  (biasing factor:       1.0576367187 [k_BT])
    Lambda [ 0.857143 - 0.904762 ]:       0.0483568503, Boltzmann:       0.0935233602  (biasing factor:       0.7528125000 [k_BT])
    Lambda [ 0.904762 - 0.952381 ]:       0.0484768021, Boltzmann:       0.1327829810  (biasing factor:       0.4047851562 [k_BT])
    Lambda [ 0.952381 - 1.000000 ]:       0.0483898785, Boltzmann:       0.1986821921  (biasing factor:       0.0000000000 [k_BT])

Extrapolated excess chemical potential, linear:    -329.3309928079 [K], quadratic:    -344.2360838203
Extrapolated chemical potential, linear:   -2415.7173606304 [K], quadratic:   -2430.6224516428
Ideal gas value:   -2086.3863678225 [K]
```

The computed loadings are averages of integer molecules.

Example 10: CO2/N2-mixture in DMOF (Continuous Fractional Component Monte Carlo)

```
SimulationType               MonteCarlo
NumberOfCycles               250000
NumberOfEquilibrationCycles  50000
PrintEvery                   5000
RestartFile                  no

ChargeMethod                 Ewald
Forcefield                   local
CutOffVDW                    10.0
RemoveAtomNumberCodeFromLabel no

Framework            0
FrameworkName        DMOF
UseChargesFromCIFFile yes
UnitCells            1 1 1
HeliumVoidFraction   0.614
ExternalTemperature  300
ExternalPressure     1e5

Component 0 MoleculeName             CO2
            MoleculeDefinition        ExampleDefinitions
            IdealGasRosenbluthWeight  1.0
            FugacityCoefficient       1.0
            TranslationProbability    1.0
            RotationProbability       1.0
            ReinsertionProbability    1.0
            IdentityChangeProbability 1.0
              NumberOfIdentityChanges 2
              IdentityChangesList     0 1
            SwapProbability           0.0
            CFSwapLambdaProbability   1.0
            CreateNumberOfMolecules   0

Component 1 MoleculeName             N2
            MoleculeDefinition        ExampleDefinitions
            IdealGasRosenbluthWeight  1.0
            FugacityCoefficient       1.0
            TranslationProbability    1.0
            RotationProbability       1.0
            ReinsertionProbability    1.0
            IdentityChangeProbability 1.0
              NumberOfIdentityChanges 2
              IdentityChangesList     0 1
            SwapProbability           0.0
            CFSwapLambdaProbability   1.0
            CreateNumberOfMolecules   0


Component 0 [CO2]
-------------------------------------------------------------
        Block[ 0] 20.13453          [-]
        Block[ 1] 20.29165          [-]
        Block[ 2] 21.02350          [-]
        Block[ 3] 20.14621          [-]
        Block[ 4] 20.08433          [-]
        ------------------------------------------------------------------------------
        Average loading absolute                          20.3360422358 +/-       0.4866205718 [-]
        Average loading absolute [molecules/unit cell]    20.3360422358 +/-       0.4866205718 [-]
        Average loading absolute [mol/kg framework]               2.2253672141 +/-       0.0532507483 [-]
        Average loading absolute [milligram/gram framework]         97.9134869813 +/-       2.3429690238 [-]
        Average loading absolute [cm^3 (STP)/gr framework]          49.8793267671 +/-       1.1935609807 [-]
        Average loading absolute [cm^3 (STP)/cm^3 framework]        40.5330299870 +/-       0.9699137129 [-]

Component 1 [N2]
-------------------------------------------------------------
        Block[ 0] 1.43833           [-]
        Block[ 1] 1.40067           [-]
        Block[ 2] 1.49400           [-]
        Block[ 3] 1.42106           [-]
        Block[ 4] 1.41534           [-]
        ------------------------------------------------------------------------------
        Average loading absolute                              1.4338783636 +/-       0.0449541709 [-]
        Average loading absolute [molecules/unit cell]        1.4338783636 +/-       0.0449541709 [-]
        Average loading absolute [mol/kg framework]                  0.1569088942 +/-       0.0049193219 [-]
        Average loading absolute [milligram/gram framework]          4.3955641692 +/-       0.1378073259 [-]
        Average loading absolute [cm^3 (STP)/gr framework]           3.5169521490 +/-       0.1102615620 [-]
        Average loading absolute [cm^3 (STP)/cm^3 framework]         2.8579521047 +/-       0.0896009527 [-]


Performance of the CFCMC swap lambda move:
==========================================
Component [CO2] total tried: 616942.000000 constant-lambda accepted: 386152.000000 (62.591297 [%])
               total tried: 298756.000000 insert-lambda accepted: 74855.000000 (25.055564 [%])
               total tried: 296719.000000 remove-lambda accepted: 75209.000000 (25.346877 [%])

Component [N2] total tried: 614706.000000 constant-lambda accepted: 423919.000000 (68.962886 [%])
               total tried: 299692.000000 insert-lambda accepted: 93908.000000 (31.334837 [%])
               total tried: 297257.000000 remove-lambda accepted: 93539.000000 (31.467383 [%])
```

Example 11: CO2 Gibbs (Continuous Fractional Component Monte Carlo)

```
SimulationType                MonteCarlo
NumberOfCycles                50000
NumberOfInitializationCycles  10000
NumberOfEquilibrationCycles   20000
PrintEvery                    5000

ChargeMethod                  Ewald
Forcefield                    ExampleMoleculeForceField

Box 0
BoxLengths 26.0 26.0 26.0
BoxAngles 90.0 90.0 90.0
ExternalTemperature 260.0

Box 1
BoxLengths 60.0 60.0 60.0
BoxAngles 90.0 90.0 90.0
ExternalTemperature 260.0

GibbsVolumeChangeProbability 0.1

Component 0 MoleculeName              CO2
            MoleculeDefinition        ExampleDefinitions
            IdealGasRosenbluthWeight  1.0
            TranslationProbability    1.0
            RotationProbability       1.0
            ReinsertionProbability    1.0
            CFGibbsProbability        1.0
            CreateNumberOfMolecules   400 100


Average density component 0 [CO2]
-------------------------------------------------------------
    Block[ 0]           69.54095 [kg/m^3]
    Block[ 1]           62.05945 [kg/m^3]
    Block[ 2]           60.45748 [kg/m^3]
    Block[ 3]           67.07674 [kg/m^3]
    Block[ 4]           61.66529 [kg/m^3]
    ------------------------------------------------------------------------------
    Average             64.15998 [kg/m^3] +/-           4.88002 [kg/m^3]

Average density component 0 [CO2]
-------------------------------------------------------------
    Block[ 0]         1015.09218 [kg/m^3]
    Block[ 1]         1013.31658 [kg/m^3]
    Block[ 2]         1000.18506 [kg/m^3]
    Block[ 3]         1012.73437 [kg/m^3]
    Block[ 4]         1009.78704 [kg/m^3]
    ------------------------------------------------------------------------------
    Average           1010.22305 [kg/m^3] +/-           7.35866 [kg/m^3]


Performance of the CFCMC Gibbs lambda move:
===========================================
Component [CO2] total tried: 5458693.000000 constant-lambda accepted: 2807196.000000 (51.426156 [%])
               total tried: 1103450.000000 insert-lambda accepted: 254903.000000 (23.100548 [%])
               total tried: 1093604.000000 remove-lambda accepted: 254943.000000 (23.312186 [%])
```

Example 12: MD MuVT

A.O. Yazaydin added the computational Grand Canonical Molecular Dynamics (GCMD) methodology to RASPA[13, 14]. This approach incorporates GCMC and MD procedures, allowing for the determination of adsorption and structural and dynamical details using the same simulation run.

```
SimulationType              MolecularDynamics
NumberOfCycles              10000
NumberOfEquilibrationCycles 10000
PrintEvery                  1000
RestartFile                 no

Ensemble                    MuVT

Forcefield                  Dubbeldam2007FlexibleIRMOF-1
CutOff                      12.0

Movies yes
WriteMoviesEvery 10000

Framework 0
FrameworkName IRMOF-1
HeliumVoidFraction 0.801937
UnitCells 1 1 1
ExternalTemperature 298.0
ExternalPressure 3000000.0

FlexibleFramework yes
FrameworkDefinitions Dubbeldam2007FlexibleIRMOF-1

Component 0 MoleculeName             CO2
            MoleculeDefinition       ExampleDefinitions
            IdealGasRosenbluthWeight 1.0
            SwapProbability          1.0
            CreateNumberOfMolecules  1
```

Example 13: MD MuPT

```
SimulationType              MolecularDynamics
NumberOfCycles              10000
NumberOfEquilibrationCycles 10000
PrintEvery                  1000

Ensemble                    MuPT

Forcefield                  Dubbeldam2007FlexibleIRMOF-1
CutOff                      12.0

Movies yes
WriteMoviesEvery 10000

Framework 0
FrameworkName IRMOF-1
HeliumVoidFraction 0.801937
UnitCells 1 1 1
ExternalTemperature 298.0
ExternalPressure 3000000.0

FlexibleFramework yes
FrameworkDefinitions Dubbeldam2007FlexibleIRMOF-1

Component 0 MoleculeName             CO2
            MoleculeDefinition       ExampleDefinitions
            IdealGasRosenbluthWeight 1.0
            SwapProbability          1.0
            CreateNumberOfMolecules  1
```

Example 14: MD MuPT-PR

```
SimulationType              MolecularDynamics
NumberOfCycles              10000
NumberOfEquilibrationCycles 10000
PrintEvery                  1000

Ensemble                    MuPTPR
NPTPRCellType               RegularUpperTriangle

Forcefield                  Dubbeldam2007FlexibleIRMOF-1
CutOff                      12.0

Framework 0
FrameworkName IRMOF-1
HeliumVoidFraction 0.801937
UnitCells 1 1 1
ExternalTemperature 298.0
ExternalPressure 1500000.0

FlexibleFramework yes
FrameworkDefinitions Dubbeldam2007FlexibleIRMOF-1

Component 0 MoleculeName             CO2
            MoleculeDefinition       ExampleDefinitions
            SwapProbability          1.0
            CreateNumberOfMolecules  100
```

Example 15: Minimization Core-Shell model of CHA and elastic constants

```
SimulationType                  Minimization
NumberOfCycles                  1
```

```
NumberOfEquilibrationCycles      0
NumberOfInitializationCycles     0
PrintEvery                       10
RestartFile                      no

Ensemble                         NPTPR
NPTPRCellType                    RegularUpperTriangle

ComputeElasticConstants yes

RemoveBondNeighboursFromLongRangeInteraction yes
RemoveBendNeighboursFromLongRangeInteraction no
RemoveTorsionNeighboursFromLongRangeInteraction no

ChargeMethod                              Ewald
InternalFrameworkLennardJonesInteractions  yes
CutOff                                    12.0
TimeStep                                  0.0005
Forcefield                                CoreShellSchroderSauer
EwaldPrecision                            1e-10

Framework 0
FrameworkName CHA_SI
RemoveAtomNumberCodeFromLabel yes
UnitCells 1 1 1
ReplicaUnitCells 3 3 3
ExternalTemperature 77.0
ExternalPressure    0.0
Movies yes
WriteMoviesEvery 1

FlexibleFramework                yes
FrameworkDefinitions             CoreShellSchroderSauer
```

```
Volume [A^3]:     805.3836251300
      Box:       9.3347733108      -0.7362944074      -0.7362944074     Strain derivative:      -0.0000000137      -0.0000000231      -0.0000000200
                 0.0000000000       9.3056898353      -0.7968534272                             -0.0000000231      -0.0000000552      -0.0000000355
                 0.0000000000       0.0000000000       9.2715094740                             -0.0000000198      -0.0000000356      -0.0000000529
      Final Lengths:       9.3347733108       9.3347733108       9.3347733108, Angles:      94.5239908381      94.5239908381      94.5239908381

Elastic constant (Voigt notation) [GPa]
------------------------------
   126.31539     73.49814     72.71438     -4.55953     -5.31953     -4.88195
    73.49814    124.46650     72.06681     -5.95845     -4.39911     -6.80166
    72.71438     72.06681    122.05241     -8.08571     -7.47121     -4.09221
    -4.55953     -5.95845     -8.08571     13.31224     -1.73067     -1.82589
    -5.31953     -4.39911     -7.47121     -1.73067     13.58611     -1.77468
    -4.88195     -6.80166     -4.09221     -1.82589     -1.77468     13.89117
```

Example 16: Minimization Nicholas model of CHA and elastic constants

```
SimulationType                   Minimization
MinimizationMethod               Baker
NumberOfCycles                   1
RestartFile                      no
PrintEvery                       1

RemoveBondNeighboursFromLongRangeInteraction yes
RemoveBendNeighboursFromLongRangeInteraction yes
RemoveTorsionNeighboursFromLongRangeInteraction no

Ensemble                         NPTPR
NPTPRCellType                    RegularUpperTriangle

ComputeElasticConstants yes

Forcefield                       Nicholas
ChargeMethod                     Ewald
EwaldPrecision 1e-10

Framework 0
FrameworkName CHA_SI
RestrictFrameworkAtomsToBox yes
RemoveAtomNumberCodeFromLabel yes
ReplicaUnitCells 3 3 3
ExternalTemperature 300.0
Movies yes
WriteMoviesEvery   1

FlexibleFramework                yes
FrameworkDefinitions             Nicholas
```

```
Volume [A^3]:     665.6486445072
      Box:       8.7698325037      -0.7934349952      -0.7934349952     Strain derivative:      -0.0000001414      -0.0000001830      -0.0000000514
                 0.0000000000       8.7338664434      -0.8687825891                             -0.0000001830      -0.0000001055      -0.0000000794
                 0.0000000000       0.0000000000       8.6905488817                             -0.0000000514      -0.0000000794      -0.0000001456
      Final Lengths:       8.7698325037       8.7698325037       8.7698325037, Angles:      95.1908317301      95.1908317301      95.1908317301

Elastic constant (Voigt notation) [GPa]
------------------------------
   101.05928     33.28918     33.04129     -1.23984     -1.70149     -1.53987
    33.28918    100.05217     32.65362     -2.54735     -1.75724     -4.00308
    33.04129     32.65362     98.44307     -5.50069     -5.02362     -2.13371
    -1.23984     -2.54735     -5.50069     18.67329     -4.20936     -4.05074
    -1.70149     -1.75724     -5.02362     -4.20936     19.43810     -3.75113
    -1.53987     -4.00308     -2.13371     -4.05074     -3.75113     20.18809
```

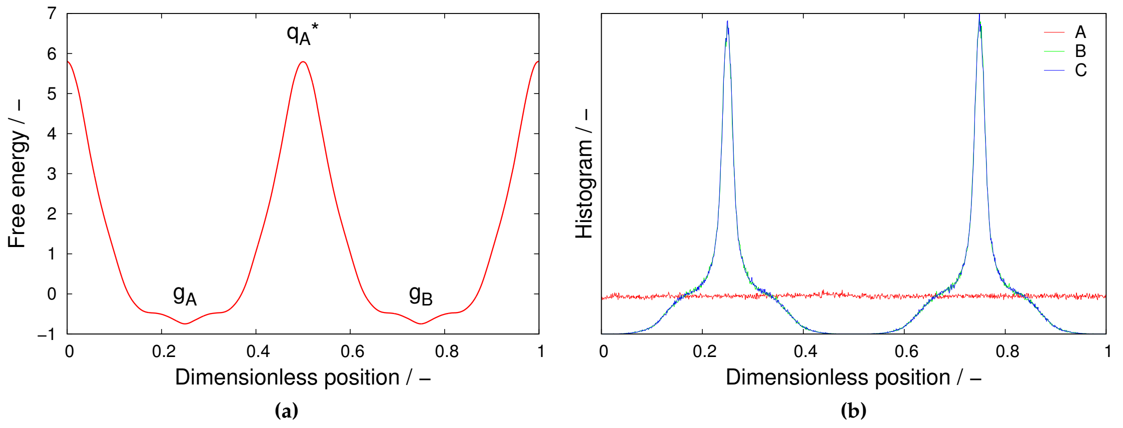


Figure 11: *Umbrella sampling: (a) free energy profile from Widom insertion (the inverse will be used as a biasing potential), (b) the histograms of the position of the tagged particle in the direction A (biasing direction), B, and C.*

Example 17: Umbrella sampling

In Umbrella sampling we can tag one particle and add a biasing potential to it. Figure 11(a) shows the used biasing potential, which is directly obtained from Widom insertion of methane in LTA. Any profile will do as long as it close enough. As can be seen in Figure 11(b), if we do a MC simulation with only this particle (second component zero particles), then the resulting histogram will be flat in the direction that we bias. The other directions are unbiased. Using this profile, we can easily obtain the free energy at higher loadings. Here is the input for methane in LTA at 4 methane/cage. The output will have a directory 'Histograms' containing the histograms for each component in the A, B, C directions. Also, it automatically computes the true free energy (from the biasing-spline plus the histograms) in file starting with 'FreeEnergy'.

```
SimulationType                   MonteCarlo
NumberOfCycles                   10000000000
NumberOfInitializationCycles     1000
PrintEvery                       5000

Forcefield                       ExampleZeolitesForceField

Framework                        0
FrameworkName                    LTA_SI
ShiftUnitCells                   0.0 0.0 0.0
UnitCells                        1 1 1
ExternalTemperature              600.0

ComputePositionHistogram         yes
WritePositionHistogramEvery      10000

component 0 MoleculeName                     methane
            MoleculeDefinition               ExampleDefinitions
            BiasingProfile                   Profile.dat
            BiasingMethod                    Umbrella
            BiasingDirection                 A
            BlockPockets                     yes
            BlockPocketsFileName             LTA_SI
            TranslationProbability           1.0
            RotationProbability              1.0
            ReinsertionProbability           1.0
            CreateNumberOfMolecules          1

component 1 MoleculeName                     methane
            MoleculeDefinition               ExampleDefinitions
            BlockPockets                     yes
            BlockPocketsFileName             LTA_SI
            TranslationProbability           1.0
            RotationProbability              1.0
            ReinsertionProbability           1.0
            CreateNumberOfMolecules          31
```

The biasing spline (here called 'Profile.dat') has a header describing the spline:

```
# 1801
# 0.5
# 0.25 0.75 12.2775
# 0.0  0.5
0 5.87638 0.0919053
0.000555556 5.79087 0.0924271
...
```

The lines have the following meaning:

- the number of data points in the file,
- the dimensionless position of the barrier q_A^*
- the dimensionless position of the minimum of the free energy g_A and g_B, and the distance d between g_A and g_B in Angstrom
- the left and right boundary of g_A

followed by the actual data points:

- dimensionless position
- dimensionless free energy βF or F in unit of $k_B T$
- error in the free energy

Example 18: dcTST diffusivities

The first step for dcTST is to compute the free energy profile as a function of a one-dimensional reaction coordinate. In general this mapping is complex, but for certain zeolites the mapping is trivial. As an example, we use the LTA-type zeolite with a cubic unit cell of 24.555 Å. For this structure, the mapping can be done in x, y, or z and all three give identical results. We can define a reaction coordinate from $x = 0$ to $x = 1$ with several key values:

- x=0: the window on the left.
- x=0.25 the center of the left cage A.
- x=0.5: the window in the middle separating the left cage A from the right cage B.
- x=0.75 the center of the right cage A.
- x=1: the window on the right.

Computing the free energy profile

The first step is to compute the free energy profile. A convenient way at low loading is to use Widom insertion.

```
SimulationType                  MonteCarlo
NumberOfCycles                  1000000000000000
NumberOfInitializationCycles    1000
PrintEvery                      10000

Forcefield                      ExampleZeolitesForceField

Framework                       0
FrameworkName                   LTA_SI
RemoveAtomNumberCodeFromLabel   yes
ShiftUnitCells                  0.0 0.0 0.0
UnitCells                       1 1 1
ExternalTemperature             600.0

WriteFreeEnergyProfileEvery     5000

component 0 MoleculeName                    methane
            MoleculeDefinition              ExampleDefinitions
            ComputeFreeEnergyProfile        yes
            BlockPockets                    yes
            BlockPocketsFileName            LTA_SI
            TranslationProbability          1.0
            RotationProbability             1.0
            ReinsertionProbability          1.0
            CreateNumberOfMolecules         0
```

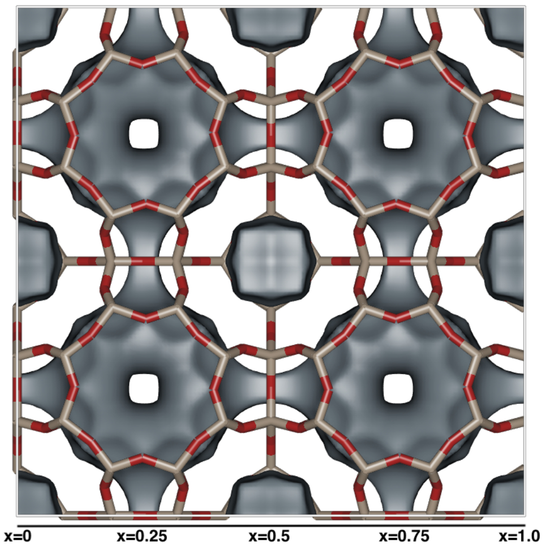


Figure 12: *The reaction coordinate mapping for the LTA-type structure.*

The result converge slowly to a nice profile. The scatter is the highest at places where the free energy is high, and the scatter in the data is low at places of low free energy. We can now define the diffusion in terms of figure 13: we compute the diffusion of a molecule from g_A in cage A to g_B in cage B across barrier q^*_A. To input this information we make a "biasing profile"-file (`Profile.dat`) with this data at the top

```
# 1801
# 0.5
# 0.25 0.75 12.2775
# 0.0  0.5
......
```

First line is the number of data points (here 1801), then the position of the barrier q^*_A (here 0.5), then g_A, g_B, and the distance in Angstrom between them (here 0.25, 0.75, and 12.2775, respectively). and lastly the range of cage A (here from 0.0 to 0.5).

Computing TST-estimates

We can now use this file

```
  BiasingProfile                    Profile.dat
  BiasingMethod                     Umbrella
  BiasingDirection                  A
```

to for example perform Umbrella sampling. In addition, it will create a spline-file `BiasingSpline_methane_0.dat`, that contains a lot of information and a fitting spline.

```
# Dividing surfaces:     0.500000000000 [-]
# Free energy minima:     0.250000000000 [-]     0.750000000000 [-]  lattice distance:    12.277500000000 [A]
# Left and right boundary:     0.000000000000 [-]     0.500000000000 [-]
# F(QstarA):     5.794749230708
# Exp(-Beta QStarA):   0.00304349354572
# Integral Exp(-Beta q) over region left boundary (0) to q* (0.5):  1.10187629401e-09
# Mass reaction bead:    16.042460000000 [au]
# |v|=Sqrt(k_B T/(2.0*PI*Mass)):   222.467902978164 [m/s]
```

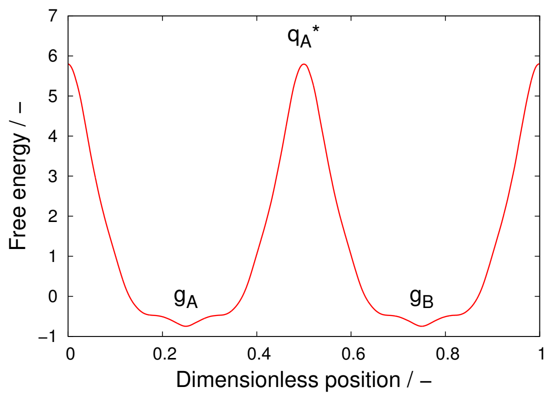


Figure 13: *The free energy obtained with Widom insertion.*

```
# P(q*) dq:      2762100.93845 [1/m]
# k^TST= |v| P(q*) dq, i.e. the TST hopping rate:       614478803.59 [1/s]
# D^TST:  9.26246952572e-10 [m^2/s]

# RM Int1, Integral Exp(Beta q) over region gA to gB:    18.077489357418
# RM Int2, Integral Exp(-Beta q) over full region:     0.448738054981
# RM Int1*Int2:     8.112057413187
# RM 1/(Int1*Int2):     0.123273289261

# <n_A>:    24.555000000000
# 1/<n_A>:     0.040724903278
```

It uses the computed smoothed spline that fits the data to calculate the integrals and dcTST information. The k^{TST} is 614478803.59 events per second, and $D^{\text{TST}} = 9.26246952572e - 10$ m^2/s. The spline (column 1 and 2) is convenient as it is continuous and smooth.

Computing free energies at finite loading with brute-force MD

Of course, one can try to compute the free energy using brute-force MD, for example at an average of 8 molecules per cage.

```
SimulationType                   MD
NumberOfCycles                   100000000
NumberOfInitializationCycles     5000
NumberOfEquilibrationCycles      10000
PrintEvery                       10000

Forcefield                       ExampleZeolitesForceField

Framework                        0
FrameworkName                    LTA_SI
RemoveAtomNumberCodeFromLabel    yes
ShiftUnitCells                   0.0 0.0 0.0
UnitCells                        1 1 1
ExternalTemperature              600.0

ComputePositionHistogram         yes
WritePositionHistogramEvery      10000

component 0 MoleculeName                     methane
            MoleculeDefinition               ExampleDefinitions
            BlockPockets                     yes
            BlockPocketsFileName             LTA_SI
            TranslationProbability           1.0
            RotationProbability              1.0
            ReinsertionProbability           1.0
            CreateNumberOfMolecules          64
```

However, this only works for low free energy barriers.

Computing free energies at finite loading using Umbrella sampling

With Umbrella sampling we can bias the movement of a single tagged molecule at the proper chosen loading. We therefore need two components: component one is a single biased molecule, and component two are the other (unbiased) particles. We can compute the histogram of the positions and for component one, recomputed the actual free energy taking the biasing into account.

```
SimulationType                 MonteCarlo
NumberOfCycles                 1000000000000000
NumberOfInitializationCycles   1000
PrintEvery                     1000

Forcefield                     ExampleZeolitesForceField

Framework                      0
FrameworkName                  LTA_SI
ShiftUnitCells                 0.0 0.0 0.0
UnitCells                      1 1 1
ExternalTemperature            600.0

ComputePositionHistogram       yes
WritePositionHistogramEvery    10000

component 0 MoleculeName                   methane
            MoleculeDefinition             ExampleDefinitions
            BiasingProfile                 Profile.dat
            BiasingMethod                  Umbrella
            BiasingDirection               A
            BlockPockets                   yes
            BlockPocketsFileName           LTA_SI
            TranslationProbability         1.0
            RotationProbability            1.0
            ReinsertionProbability         1.0
            CreateNumberOfMolecules        1

component 1 MoleculeName                   methane
            MoleculeDefinition             ExampleDefinitions
            BlockPockets                   yes
            BlockPocketsFileName           LTA_SI
            TranslationProbability         1.0
            RotationProbability            1.0
            ReinsertionProbability         1.0
            CreateNumberOfMolecules        63
```

Computing the dynamical correction

The TST estimates are . . . estimates. In reality, not all particles at the dividing surface actually cross the boundary. We have to explicitly compute this property using many short MD trajectories. Step one is to compute initial state for these trajectories:

```
SimulationType                 MonteCarlo
NumberOfCycles                 100000000
NumberOfInitializationCycles   1000
PrintEvery                     100

Forcefield                     ExampleZeolitesForceField

Framework                      0
FrameworkName                  LTA_SI
RemoveAtomNumberCodeFromLabel  yes
ShiftUnitCells                 0.0 0.0 0.0
UnitCells                      1 1 1
ExternalTemperature            600.0

WritedcTSTSnapShotsToFile      yes
PutMoleculeOnBarrier           yes
BarrierPosition                0.5 0.25 0.25
WritedcTSTSnapShotsEvery       100


component 0 MoleculeName                   methane
            MoleculeDefinition             ExampleDefinitions
            TranslationProbability         1.0
              TranslationDirection           bc
            RotationProbability            1.0
            RegrowInPlaceProbability       1.0
            CreateNumberOfMolecules        0

component 1 MoleculeName                   methane
            MoleculeDefinition             ExampleDefinitions
            ComputeFreeEnergyProfile       yes
            BlockPockets                   yes
            BlockPocketsFileName           LTA_SI
            TranslationProbability         1.0
            RotationProbability            1.0
            ReinsertionProbability         1.0
            CreateNumberOfMolecules        63
```

To sample configurations, we need to write out ”snapshots”, but with some sampling in between to diminish the correlation between the snapshots. We also need to place the particle at the barrier and define the barrier position.

```
  WritedcTSTSnapShotsToFile        yes
  PutMoleculeOnBarrier             yes
  BarrierPosition                  0.5 0.25 0.25
  WritedcTSTSnapShotsEvery         100
```

Also note, that the particle on the barrier is restricted to only move one the barrier plane.

```
  TranslationProbability           1.0
  TranslationDirection             bc
```

We can do uses the sampled snapshots to run many barrier-recrossing MD trajectories.

```
SimulationType                   BarrierRecrossing

Forcefield                       ExampleZeolitesForceField

Framework                        0
FrameworkName                    LTA_SI
RemoveAtomNumberCodeFromLabel    yes
ShiftUnitCells                   0.0 0.0 0.0
UnitCells                        1 1 1
ExternalTemperature              600.0

PutMoleculeOnBarrier             yes
FreeEnergyMappingType            A
BarrierPosition                  0.5 0.25 0.25
MaxBarrierDistance               4.0
MaxBarrierTime                   10.0
NumberOfVelocities               1

component 0 MoleculeName                     methane
            MoleculeDefinition               ExampleDefinitions
            CreateNumberOfMolecules          0

component 1 MoleculeName                     methane
            MoleculeDefinition               ExampleDefinitions
            CreateNumberOfMolecules          63
```

4.5 Auxiliary examples

Example 1: Computing the ideal gas Rosenbluth weights of linear alkanes C_5 - C_9

To compare simulation values to experiments a reference state should be chosen. A convenient reference state is the ideal gas. The reference Rosenbluth value can be computed from a simulation of a single chain at the desired temperature. Note that for Rosenbluth weights several chains can be computed simultaneously, since they are computed from Widom insertions where the molecule is never actually inserted in the system.

```
SimulationType        MonteCarlo
NumberOfCycles        25000
PrintEvery            1000
PrintPropertiesEvery  1000

Forcefield            ExampleZeolitesForceField

Box 0
BoxLengths 30 30 30
ExternalTemperature 573.0

Component 0 MoleculeName              pentane
            MoleculeDefinition        ExampleDefinitions
            WidomProbability          1.0
            CreateNumberOfMolecules   0

Component 1 MoleculeName              hexane
            MoleculeDefinition        ExampleDefinitions
```

```
           WidomProbability          1.0
           CreateNumberOfMolecules   0

Component 2 MoleculeName             heptane
           MoleculeDefinition        ExampleDefinitions
           WidomProbability          1.0
           CreateNumberOfMolecules   0

Component 3 MoleculeName             octane
           MoleculeDefinition        ExampleDefinitions
           WidomProbability          1.0
           CreateNumberOfMolecules   0

Component 4 MoleculeName             nonane
           MoleculeDefinition        ExampleDefinitions
           WidomProbability          1.0
           CreateNumberOfMolecules   0
```

The output contains

```
Average Widom Rosenbluth factor:
================================
        [C5] Average Widom:   0.0668555 +/- 0.000131 [-]
        [C6] Average Widom:   0.0175062 +/- 0.000067 [-]
        [C7] Average Widom:   0.00462547 +/- 0.000010 [-]
        [C8] Average Widom:   0.00122842 +/- 0.000005 [-]
        [C9] Average Widom:   0.000328228 +/- 0.000001 [-]
```

which is printed every 'PrintPropertiesEvery' cycles. The 'Rosenbluth factor new' are the values of interest. The average and error estimated from block averages is printed at the end of the simulation.

Example 2: Computing the ideal gas Rosenbluth weights of hexane isomers

```
SimulationType        MonteCarlo
NumberOfCycles        25000
PrintEvery            1000
PrintPropertiesEvery  1000

Forcefield            ExampleZeolitesForceField

Box 0
BoxLengths 30 30 30
ExternalTemperature 433.0

Component 0 MoleculeName             hexane
            MoleculeDefinition       ExampleDefinitions
            WidomProbability         1.0
            CreateNumberOfMolecules  0

Component 1 MoleculeName             2-methylpentane
            MoleculeDefinition       ExampleDefinitions
            WidomProbability         1.0
            CreateNumberOfMolecules  0

Component 2 MoleculeName             3-methylpentane
            MoleculeDefinition       ExampleDefinitions
            WidomProbability         1.0
            CreateNumberOfMolecules  0

Component 3 MoleculeName             22-dimethylbutane
            MoleculeDefinition       ExampleDefinitions
            WidomProbability         1.0
            CreateNumberOfMolecules  0
```

Example 3: Computing the helium void-fraction of a structure (pore volume)

The void fraction is the empty space of a structure divided by the total volume. In experiment it is measured using helium, because helium does (almost) not adsorb. It would be consistent to also measure this fraction using helium at room temperature. In practice it is easily computed from Widom particle insertion as the void fraction corresponds to the new Rosenbluth weight.

```
SimulationType        MonteCarlo
NumberOfCycles        500000
PrintEvery            10000
PrintPropertiesEvery  10000

Forcefield            ExampleMOFsForceField

Framework 0
FrameworkName IRMOF-1
UnitCells 1 1 1
ExternalTemperature 298.0

Component 0 MoleculeName            helium
            MoleculeDefinition      ExampleDefinitions
            WidomProbability        1.0
            CreateNumberOfMolecules 0
```

The Rosenbluth weight, and therefore the helium void fraction of IRMOF-1 is approximately 0.80. The pore volume is the void fraction times the unit cell volume. Note that the values dependent slightly on the cutoff, and shifted vs. truncated potentials.

```
Average Widom Rosenbluth factor:
================================
        Block[ 0] 0.803749 [-]
        Block[ 1] 0.803741 [-]
        Block[ 2] 0.803497 [-]
        Block[ 3] 0.803818 [-]
        Block[ 4] 0.803536 [-]
        ------------------------------------------------------------------------------
        [helium] Average Widom:   0.803668 +/- 0.000255 [-]
```

Example 4: Computing the surface area of IRMOF-1

The geometric surface area can easily be computed by 'rolling an atom over the surface' and measure the surface. In practice, for each framework atom points are generate on a sphere around the framework atom, and the amount of overlap with other framework atoms is determined. The fraction of overlap is multiplied times the area of the sphere. The summation over all framework atoms gives the geometric surface area. This example shows how to compute the surface area of IRMOF-1. 'SurfaceAreaSamplingPointsPerShere' is the amount of points generated on sphere at a distance dependent on the mixing rule, the probe-atom and the current framework atom type. The more points the higher the accuracy. The simulation usually takes between 5 and 30 minutes.

In this example the structure is probed with hydrogen using the second bead ('H_com' with $\sigma = 2.958$ Å). The option 'SurfaceAreaProbeDistance Sigma' sets the overlap criteria to σ instead of the default $\sigma^{1/6}$.

```
SimulationType        MonteCarlo
NumberOfCycles        10000
PrintEvery            100
PrintPropertiesEvery  100

Forcefield Dubbeldam2007FlexibleIRMOF-1
CutOff 12.8

Framework 0
FrameworkName IRMOF-1
UnitCells 1 1 1
SurfaceAreaProbeDistance Sigma

Component 0 MoleculeName             argon
            MoleculeDefinition       ExampleDefinitions
            SurfaceAreaProbability   1.0
            CreateNumberOfMolecules  0
```

The area depends on the probe atom and on whether the well-depth at $2^{1/6}\sigma$ ($\approx 1.12246\sigma$) is used ('SurfaceAreaProbeDistance Minimum')

```
Surface area: 2082.509853 [m^2/cm^3]
Surface area: 3510.189484 [m^2/g]
```

or σ is used as the distance criteria ('SurfaceAreaProbeDistance Sigma'):

```
Surface area: 2266.243128 [m^2/cm^3]
Surface area: 3819.882429 [m^2/g]
```

Example 5: Powder diffraction pattern

Powder diffraction is a scientific technique using X-Ray or neutron diffraction on powder or microcrystalline samples for structural characterization of materials. The most widespread use of powder diffraction is in the identification and characterization of crystalline solids, each of which produces a distinctive diffraction pattern. Both the positions (corresponding to lattice spacings) and the relative intensity of the lines are indicative of a particular phase and material, providing a "fingerprint" for comparison. The database of IZA for zeolite has the option to generate the powder diffraction pattern:

```
http://izasc.ethz.ch/fmi/xsl/IZA-SC/xrd.xsl
```

Here, an example of the powder diffraction pattern for the TON-type zeolite. Only one unit cell is sufficient for the computation (interactions are not needed in the computation, just the position and types of the atoms and the shape and size of the unit cell). The diffraction pattern usually takes a few seconds of computation, and the result is written to 'PowderDiffraction/System[0]/'. It contains two files: 'PeakInformation.dat' and 'Spectrum.dat'.

```
SimulationType  MonteCarlo
NumberOfCycles  0

Forcefield      ExampleZeolitesForceField

Framework 0
FrameworkName TON
UnitCells 1 1 1

ComputePowderDiffractionPattern yes
DiffractionType Xray
DiffractionRadiationType Copper
WaveLengthType single
TwoThetaMin 1
TwoThetaMax 50
TwoThetaStep 0.02
PeakShape PseudoVoigt
PeakWidthModifierU 0.005
```

The first elements of the file 'PeakInformation.dat' look like:

```
# 2-theta   d          h  k  l  Mult Lp          Scat. Factor    Intensity
  8.15213   0.09220 [ 1, 1, 0]    4 392.85927   19014.2044440544   100.000000
 10.15550   0.11481 [ 0,-2, 0]    2 252.33302   12381.2641234081    20.911920
 12.77464   0.14431 [ 2, 0, 0]    2 158.63285   19714.5657150741    20.933178
 16.34589   0.18441 [ 2, 2, 0]    4  96.01738    6730.9888808237     8.651941
 16.55216   0.18672 [-1,-3, 0]    4  93.58434     739.8429009358     0.926889
 19.42690   0.21886 [-1,-1,-1]    4  67.33674    3040.0925085839     2.740461
 ......................................
```

So, the elements are the angle 2θ, the d-spacing, the Miller indices h,k, and l, the multiplicity, the Lorentz-Polarization factor, the scattering factor (including anomalous scattering), and the relative intensity (where the largest intensity is set to 100). The second file 'Spectrum.dat' can be plotted using gnuplot, the first column is 2θ, the second column the intensity. The shape of the peaks can be influenced with 'PeakShape', and the peak width modifiers 'PeakWidthModifierU', 'PeakWidthModifierV', and 'PeakWidthModifierW'.

Example 6: Making 'grids'

For rigid frameworks one can precompute the energy-grid, because the potential energy field induces by the framework does not evolve in time. For each of the pseudo atoms one can generate a 3D grid where the spacing can be defined. In the example the grid points are 0.1 Å spaced apart (a=b=c=25.832Å, $258 \times 258 \times 258 = 17173512$ points). A shorter distance results in more points, more accuracy, but also a bigger grid (more memory is needed). Note that RASPA can handle a 'mixture' of grids and fully computed interactions. The table stores U, $\frac{\partial U}{\partial x}$, $\frac{\partial U}{\partial y}$, $\frac{\partial U}{\partial z}$, $\frac{\partial^2 U}{\partial x \partial y}$, $\frac{\partial^2 U}{\partial x \partial z}$, $\frac{\partial^2 U}{\partial y \partial z}$, and $\frac{\partial^3 U}{\partial x \partial y \partial z}$ at each grid point. The interpolation can handle non-orthorhombic cells and can also be used for molecular dynamics (i.e. the force interpolation is consistent with the energy interpolation).

```
SimulationType  MakeGrid

Forcefield      FlexibleIRMOF-1

Framework 0
FrameworkName IRMOF-1
UnitCells 1 1 1

NumberOfGrids 2
GridTypes C_co2 O_co2
SpacingVDWGrid 0.1
SpacingCoulombGrid 0.1
```

Example 7: Using 'grids'

The grids are stored in '/share/raspa/grids/FlexibleIRMOF-1/IRMOF-1/0.100000' and the names are 'IRMOF-1_C_co2_shifted.grid', 'IRMOF-1_O_co2_shifted.grid', and 'IRMOF-1_Electrostatics_Ewald.grid'. The last grid is the real part of the Ewald summation, i.e. erfc(r)/r using a probe charge of +1. They can be used like:

```
SimulationType                MonteCarlo
NumberOfCycles                5000
NumberOfInitializationCycles  5000
PrintEvery                    100

Forcefield                    FlexibleIRMOF-1
ChargeMethod                  Ewald
EwaldPrecision                1e-6

Framework 0
FrameworkName IRMOF-1
UnitCells 1 1 1
ExternalTemperature 298.0
```

```
ExternalPressure 5000000.0

NumberOfGrids 2
GridTypes C_co2 O_co2
SpacingVDWGrid 0.1
SpacingCoulombGrid 0.1
UseTabularGrid yes

Component 0 MoleculeName                    CO2
            MoleculeDefinition              ExampleDefinitions
            TranslationProbability          1.0
            RotationProbability             1.0
            ReinsertionProbability          1.0
            SwapProbability                 1.0
            CreateNumberOfMolecules         0
```

In the output file, in the framework section, the used grids are tested one by one. Make sure the relative error is smaller than about 0.001 for the energies. If not, either the wrong grid is used (the current settings for the force field, cutoff etc. are different from what the grid has been made with) or the structure requires a higher interpolation density.

```
PseudoAtom 9 Framework-[C_co2]
===========================================================================================
    Boltzmann average energy VDW (table)                :  -166.674268647739
    Boltzmann average energy VDW (full)                 :  -166.672515492945
    Boltzmann relative error VDW                        :     0.000050342536
    Boltzmann average energy Coulomb (table)            :  -132.261348754564
    Boltzmann average energy Coulomb (full)             :  -132.258792721618
    Boltzmann relative error Coulomb                    :     0.000048335817
===========================================================================================
    Boltzmann average Force[x] VDW (table)              :     2.814676235131
    Boltzmann average Force[x] VDW (full)               :     2.813903005890
    Boltzmann relative error VDW                        :     0.000679313669
    Boltzmann average Force[x] Coulomb (table)          :    -6.914879772888
    Boltzmann average Force[x] Coulomb (full)           :    -6.906268341227
    Boltzmann relative error Coulomb                    :     0.001165040569
===========================================================================================
    Boltzmann average Force[y] VDW (table)              :     5.650590778180
    Boltzmann average Force[y] VDW (full)               :     5.646815005685
    Boltzmann relative error VDW                        :     0.000625049225
    Boltzmann average Force[y] Coulomb (table)          :    -6.131228972874
    Boltzmann average Force[y] Coulomb (full)           :    -6.143469499297
    Boltzmann relative error Coulomb                    :     0.001198288617
===========================================================================================
    Boltzmann average Force[z] VDW (table)              :    -7.613158899110
    Boltzmann average Force[z] VDW (full)               :    -7.613093463878
    Boltzmann relative error VDW                        :     0.000638624417
    Boltzmann average Force[z] Coulomb (table)          :    -5.277718273766
    Boltzmann average Force[z] Coulomb (full)           :    -5.272042035095
    Boltzmann relative error Coulomb                    :     0.001211372128


PseudoAtom 10 Framework-[O_co2]
===========================================================================================
    Boltzmann average energy VDW (table)                :  -385.683245095266
    Boltzmann average energy VDW (full)                 :  -385.679393087921
    Boltzmann relative error VDW                        :     0.000023203661
    Boltzmann average energy Coulomb (table)            :    92.387158042874
    Boltzmann average energy Coulomb (full)             :    92.385385971036
    Boltzmann relative error Coulomb                    :     0.000049328766
===========================================================================================
    Boltzmann average Force[x] VDW (table)              :   -12.358055145720
    Boltzmann average Force[x] VDW (full)               :   -12.364288510033
    Boltzmann relative error VDW                        :     0.000522114664
    Boltzmann average Force[x] Coulomb (table)          :    -1.639803867255
    Boltzmann average Force[x] Coulomb (full)           :    -1.640207574883
    Boltzmann relative error Coulomb                    :     0.001302056759
===========================================================================================
    Boltzmann average Force[y] VDW (table)              :    -3.248927867445
    Boltzmann average Force[y] VDW (full)               :    -3.245131932968
    Boltzmann relative error VDW                        :     0.000521447609
    Boltzmann average Force[y] Coulomb (table)          :    -3.993441639796
    Boltzmann average Force[y] Coulomb (full)           :    -3.990850472425
    Boltzmann relative error Coulomb                    :     0.001252842918
===========================================================================================
    Boltzmann average Force[z] VDW (table)              :     7.195354496737
    Boltzmann average Force[z] VDW (full)               :     7.193452593912
    Boltzmann relative error VDW                        :     0.000556132560
    Boltzmann average Force[z] Coulomb (table)          :     1.773911985304
    Boltzmann average Force[z] Coulomb (full)           :     1.774122551801
    Boltzmann relative error Coulomb                    :     0.001236592291
```

Example 8: Charge-equilibrium IRMOF-1

Example 9: Pore-Size Distribution

Example 10: Typing framework atoms

```
SimulationType                  MonteCarlo
NumberOfCycles                  0

Forcefield                      Local
```

```
Framework 0
FrameworkName NU-100SP
UnitCells 1 1 1
ExternalTemperature 298.0

ModifyFrameworkAtomConnectedTo C C1 O
ModifyFrameworkAtomConnectedTo C C2 C1
ModifyFrameworkAtomConnectedTo C C3 C2 C2
ModifyFrameworkAtomConnectedTo C C4 C2
ModifyFrameworkAtomConnectedTo C C5 C4
ModifyFrameworkAtomConnectedTo C C6 C5
ModifyFrameworkAtomConnectedTo C C7 C6
ModifyFrameworkAtomConnectedTo C C8 C7
ModifyFrameworkAtomConnectedTo C C9 C8
ModifyFrameworkAtomConnectedTo C C10 C9
ModifyFrameworkAtomConnectedTo C C11 C10
ModifyFrameworkAtomConnectedTo C C12 C11
ModifyFrameworkAtomConnectedTo C C13 C12
ModifyFrameworkAtomConnectedTo C C14 C13
ModifyFrameworkAtomConnectedTo C C15 C14
ModifyFrameworkAtomConnectedTo H H1 C3
ModifyFrameworkAtomConnectedTo H H2 C4
ModifyFrameworkAtomConnectedTo H H3 C9
ModifyFrameworkAtomConnectedTo H H4 C10
ModifyFrameworkAtomConnectedTo H H5 C15
ModifyFrameworkAtomConnectedTo O O2 C1
ModifyFrameworkAtomConnectedTo Cu Cu O2
```

Example 11: Random Aluminum Distribution

The Nax structure is defined in the CIF-file as

```
loop_
_atom_site_label
_atom_site_type_symbol
_atom_site_fract_x
_atom_site_fract_y
_atom_site_fract_z
_atom_site_charge
Si1      Si4+  -0.05381   0.12565   0.03508   2.05
Al1      Al3+  -0.05524   0.03639   0.12418   1.75
O1       O2-   -0.1099    0.0003    0.1056   -1.025
O2       O2-   -0.0011   -0.0028    0.1416   -1.025
O3       O2-   -0.0346    0.0758    0.0711   -1.025
O4       O2-   -0.0693    0.0726    0.18     -1.025
Oa1      O2-    ?         ?         ?        -1.2
Oa2      O2-    ?         ?         ?        -1.2
Oa3      O2-    ?         ?         ?        -1.2
Oa4      O2-    ?         ?         ?        -1.2
```

The atom-types found in the cif-file, and not already defined in the `Pseudo_atoms.def`-file, will be automatically added. The `Oa1`, `Oa2`, `Oa3`, and `Oa4` atoms are added but have no defined positions. The initial structure has the maximum amount of aluminum (given the Lowenstein rule)

```
Pseudo Atoms    8 [     Si1]:   96 atoms
Pseudo Atoms    9 [     Al1]:   96 atoms
Pseudo Atoms   10 [      O1]:   96 atoms
Pseudo Atoms   11 [      O2]:   96 atoms
Pseudo Atoms   12 [      O3]:   96 atoms
Pseudo Atoms   13 [      O4]:   96 atoms
```

To create a NaY version with 58 aluminum we can take 38 aluminum randomly and change them to silicon. We also modify the `O1-O4` to types `Oa1-Oa4` when connected to an aluminum.

```
SimulationType               MC
NumberOfCycles               0
NumberOfInitializationCycles 0
PrintEvery                   10

Forcefield                   Local

RandomlySubstitute 38 Al1 Si1

ModifyFrameworkAtomConnectedTo O1 Oa1 Al1
ModifyFrameworkAtomConnectedTo O2 Oa2 Al1
ModifyFrameworkAtomConnectedTo O3 Oa3 Al1
ModifyFrameworkAtomConnectedTo O4 Oa4 Al1

Framework 0
FrameworkName NaX
UnitCells 1 1 1
ExternalTemperature 300.0
```

The new structure can be found in the `Movie` directory and has the desired content

```
Pseudo Atoms    8 [     Si1]:  134 atoms
Pseudo Atoms    9 [     Al1]:   58 atoms
Pseudo Atoms   10 [      O1]:   38 atoms
Pseudo Atoms   11 [      O2]:   38 atoms
Pseudo Atoms   12 [      O3]:   38 atoms
Pseudo Atoms   13 [      O4]:   38 atoms
Pseudo Atoms   14 [     Oa1]:   58 atoms
Pseudo Atoms   15 [     Oa2]:   58 atoms
Pseudo Atoms   16 [     Oa3]:   58 atoms
Pseudo Atoms   17 [     Oa4]:   58 atoms
```

4.6 Where to go from here?

Constructing your own input for your systems

The first thing to do is to scan the scientific literature whether people have already developed models for your system. The models for small adsorbates by the Calero-group using shifted LJ potentials (cutoff 12 Å) for molecules calibrated on experimental vapor-liquid equilibrium data. These potentials can be used by both MC and MD, which is a great advantage.

molecule type	all-atom	type	ϵ/k_B [K]	σ [Å]	q [e]	cutoff	shifted
N_2, bond-distance 1.1 Å[15]	yes	N (N_2)	38.298	3.306	-0.405	12	yes
(0.55 Å from N atom)	-	Dummy			0.810	12	yes
O_2, bond-distance 1.2 Å[15]	yes	O (O_2)	53.023	3.045	-0.112	12	yes
(0.6 Å from O atom)	-	Dummy			0.224	12	yes
Ar [15]	yes	Ar	124.070	3.380	0.0	12	yes
CCl_4 [15]	no	CCl_4	519.730	5.140	0.0	12	yes
CO, bond-distance 1.128 Å[16]	yes	C	16.141	3.636	-0.2424	12	yes
	yes	O	98.014	2.979	-0.2744	12	yes
(0.6443 Å from C atom)	-	Dummy			0.5168	12	yes
CO_2, bond-distance 1.149 Å [9]	yes	O	85.671	3.017	-0.3256	12	yes
	yes	C	29.933	2.745	-0.6512	12	yes
SO_2, bond-distance 1.431 Å [17]	yes	S	189.353	3.41	0.402	12	yes
(bond angle of 119°)	yes	O	58.725	3.198	-0.201	12	yes
SF_6, bond-distance 1.565 Å [18]	yes	F	73.130	2.843	-	12	yes
	yes	S	-	-	-	12	yes
H2S, bond-distance 1.34 Å [19]	yes	S	275	3.7	-0.32	12	yes
(bond angle of 92°)	yes	H	-	-	0.16	12	yes
Alkenes [20]	no		-	-	-	12	yes
propylene [21]	no	CH_3	93.0	3.685	0.87	12	yes
	no	CH	51.0	4.0	0.87	12	yes
(Dummy-CH_2 bond length 0.704 Å)	-	Dummy			-1.74	12	yes
	-	CH_3	108.0	3.76	-	12	yes

Table 4.1: *Selection of models by the Calero-group using shifted LJ potentials (cutoff 12 Å) for molecules calibrated on experimental vapor-liquid equilibrium data. These potentials can be used by both MC and MD. Cross terms are computed using Lorentz-Berthelot mixing rules.*

A computational very efficient model is the Transferable potentials for Phase Equilibria TraPPE force field by Martin and Siepmann[22, 23]. The force field describes linear, mono-branched and di-branched alkanes[22, 23], benzene, pyridine, pyrimidine, pyrazine, pyridazine, thiophene, furan, pyrrole, thiazole, oxazole, isoxazole, imidazole, and pyrazole[11], primary, secondary, and tertiary amines, nitroalkanes and nitrobenzene, nitriles, amides, pyridine, and pyrimidine[24], ethers, glycols, ketones, and aldehydes[25], thiols, sulfides, disulfides, and thiophene [26], as well as some smaller molecule like CO_2 and N_2 [27] and ethane and ethylene[28]. Despite the fact that the model lumps CH_3, CH_2, and CH into single interaction centers, it very accurately reproduces the experimental phase diagram and critical points. This united atom approach allows for much longer simulation times and larger systems because each of the CH_x-groups is charge-neutral and charge-charge interaction can be omitted. Some TraPPE models for small molecules are listed in Table 4.2.

molecule type	all-atom	type	ϵ/k_B [K]	σ [Å]	q [e]	cutoff	shifted
Alkanes [22]	no	CH_4	148	3.73	-	14	no
	no	CH_3	98	3.75	-	14	no
	no	CH_2	46	3.95	-	14	no
Branched alkanes[23]	no	CH	10	4.68	-	14	no
	no	C	0.5	6.4	-	14	no
CO_2, bond-distance 1.16 Å [27]	yes	O	79.0	3.05	-0.35	10	no
	yes	C	27.0	2.80	-0.70	10	no
N_2, bond-distance 1.1 Å [27]	yes	N	36.0	3.31	-0.482	10	no
	-	Dummy	0.0	0.0	+0.964	10	no
CH_4 [3]	no	CH_4	158.5	3.72	0.0	12	yes

Table 4.2: *Selection of TraPPE models for molecules calibrated on experimental vapor-liquid equilibrium data. Cross terms are computed using Lorentz-Berthelot mixing rules.*

A good resource to check, for TraPPE parameters, is

http://trappe.oit.umn.edu

The advantage of the models of the Calero-group and the TraPPE parameters is that they (by design) reproduce the Vapor-Liquid Equilibrium (VLE) curves. The advantage of adsorbate models that reproduce phase equilibrium data is that the saturation value of the adsorption isotherm is well-reproduced *by construction*. This is important, because it allows an examination of the state of the pores of the framework, i.e. is there pore-blocking? are there remaining solvent or template molecules in the structure? The TraPPE model and many others use the approximation of fixed point charges. For small molecules, the charges follow from the dipole or quadrupole moment.

The models for the adsorbates need to be combined with the model for the framework. Even when keeping the framework rigid, this still includes charges and Van der Waals parameters. Many charge sets have been published for frameworks like MOFs[29, 30, 31], COFs[32], ZIFs[33, 34], and siliceous zeolites[35]. The different methods available to calculate atomic partial charges in MOFs have been reviewed by Hamad et al.[36]. A good force field for zeolite modeling is the TraPPE-zeo model[2]. In this model, the Lennard-Jones interaction sites and partial charges are placed at both the oxygen and the silicon atoms of the zeolite lattice. This allows for a better balance of dispersive and first-order electrostatic interactions than is achievable with the Lennard-Jones potential used only for the oxygen atoms. Early MOF work initially also adapted the Kiselev approach where the framework has been kept rigid. However, the force field was replaced by a more generic solution such as DREIDING[7] or UFF[8] for example, to tackle the larger chemical diversity of MOFs. These approaches were very successful without the need for re-parameterization[37, 38, 39, 40]. Over the years several challenges were found. The first major force field challenge is to take flexibility of the framework into account. For more information on flexible framework force fields and generic force fields, see Refs.[41, 42].

Work-flow:

1. Scan scientific literature for parameters and model for the adsorbate molecule.

2. If none can be found, you either have to be based them on more generic force fields, or develop your own models and fine-tune the parameters to reproduce the VLE.

3. Obtain the charge for the framework from models from literature, or from QM algorithm like REPEAT, or from charge-equilibration methods.

4. Obtain the Van der Waals parameters for the framework from literature, or else from generic force fields.

Finally, validate your models by comparing to experiments.

Number of cycles and run-times

Although classical simulations are faster then quantum simulation, they still require significant amount of computation times. That is because properties are computed at thermodynamics conditions (finite temperature) and sampling is often difficult. For small systems, like methane or argon in MFI, 10,000 cycles might be sufficient. No charge-interactions are needed, which is usually the most expensive force field term. Therefore, CO_2 and N_2 are already much more expensive. Flexible molecules require also the sampling of the internal structure. For complex mixtures, the number of cycles needed is usually in the millions. Note that RASPA is developed for relatively small adsorbates in nanoporous materials using open ensembles. For MD, a better option is to use LAMMPS[43].

Note that the most important important step, is to get an equilibrated system. In equilibrium, measured averaged properties do not change anymore as a function of simulation time, but beware that there can always be an unknown order parameter that is not equilibrated yet. The equipartition theorem states that the available energy will be shared evenly amongst the accessible modes of motion. Translation, rotational, and vibrational degrees of freedom will (on average) possess an energy $(1/2)k_BT$, where $k_B = 1.38064852 \times 10^{-23}$ J/K is the Boltzmann constant and T is the temperature. Fast modes like bond-stretching are quickly equilibrated, but equilibration will take longer for slower modes (e.g. torsions and inter-molecular interactions). It is highly system dependent how long equilibration takes, i.e. there are *no* general rules on how many equilibration or production cycles are required. Once an equilibrated system is used as a restart, you can run many Monte Carlo jobs and simply average the results to get better statistics.

Writing and using binary restart ”crash-recovery” files

Usually, and unfortunately sometimes often, computers crash, are rebooted to upgrade software or the ”walltime”-limit on the cluster has been reached etc. One can force RASPA to write a ”binary-restart-file” from which the program can exactly recover and continued where it left off. The results are identical because the data has been written in binary format and even the random number generator picks up where it left off. One has to add two lines to the ‘simulation.input’ file:

```
ContinueAfterCrash           no
WriteBinaryRestartFileEvery  1000
```

The second line tells the program to write the file every 1000 cycles. Initially, the ‘ContinueAfterCrash’ is ‘no’. For example, the adsorption of methane in MFI (Basic example 6) should be changed to

```
SimulationType               MonteCarlo
NumberOfCycles               10000
NumberOfInitializationCycles 1000
PrintEvery                   100

ContinueAfterCrash           no
WriteBinaryRestartFileEvery  1000

Forcefield                   ExampleZeolitesForceField

Framework 0
FrameworkName MFI
UnitCells 2 2 2
HeliumVoidFraction 0.29
ExternalTemperature 300.0
ExternalPressure 10000.0 20000.0 30000.0 40000.0

Component 0 MoleculeName             methane
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   0.5
            ReinsertionProbability   0.5
            SwapProbability          1.0
            CreateNumberOfMolecules  0
```

It will write a file ‘binary_restart.dat’ in the directory ‘CrashRestart’. The size of the file is usually small (a few MB). To restart the code, simply change ‘ContinueAfterCrash no’ to ‘ContinueAfterCrash yes’

```
ContinueAfterCrash           yes
WriteBinaryRestartFileEvery  1000
```

Bibliography

[1] Wood, G. B.; Panagiotopoulos, A. Z.; Rowlinson, J. S. *Mol. Phys.* **1988**, *63*, 49-63.

[2] Bai, P.; Tsapatsis, M.; Siepmann, J. I. *J. Phys. Chem. C* **2013**, *117*, 24375-24387.

[3] M. G. Martin, T. M. Nenoff A. P. Thompson *J. Chem. Phys.* **2001**, *114*, 7174-7181.

[4] Karavias, F.; Myers, A.L. *Langmuir* **1991**, *7*, 3118-3126.

[5] Snurr, R. Q.; Bell, A. T.; Theodorou, D. N. *Journal of Physical Chemistry* **1993**, *97*, 13742-13752.

[6] Vlugt, T. J. H.; Garcia-Perez, E.; Dubbeldam, D.; Ban, S.; Calero, S. *J. Chem. Theory. Comput.* **2008**, *4*, 1107-1118.

[7] Mayo, S.L.; Olafson, B.D.; Goddard, W.A. *J. Phys. Chem.* **1990**, *94*, 8897-8909.

[8] Rappé, A.K.; Casewit, C.J.; Colwell, K.S.; Goddard, W.A.; Skiff, W.M. *J. Am. Chem. Soc.* **1992**, *114*, 10024-10035.

[9] Garcia-Sanchez, A.; Ania, C. O.; Parra, J. B.; Dubbeldam, D.; Vlugt, T. J. H.; Krishna, R.; Calero, S. *J. Phys. Chem. C* **2009**, *113*, 8814-8820.

[10] Dubbeldam, D.; Walton, K.S.; Ellis, D.E.; Snurr, R.Q. *Angew. Chem. Int. Ed.* **2007**, *46*, 4496-4499.

[11] Rai, N.; Siepmann, J.I. *J. Phys. Chem. B* **2007**, *111*, 10790-10799.

[12] Dubbeldam, D.; Calero, S.; Vlugt, T.J.H.; Krishna, R.; Maesen, T.L.M.; Smit, B. *J. Phys. Chem. B* **2004**, *108*, 12301-12313.

[13] Loganathan, N.; Yazaydin, A. O.; Bowers, G. M.; Kalinichev, A. G.; Kirkpatrick, R. J. *J. Phys. Chem. C* **2017**, *121*, 24527-24540.

[14] Loganathan, N.; Bowers, G. M.; Yazaydin, A. O.; Schaef, H. T.; Loring, J. S.; Kalinichev, A. G.; Kirkpatrick, R. J. *J. Phys. Chem. C* **2018**, *122*, 4391-4402.

[15] Martin-Calvo, A.; Garcia-Perez, E.; Garcia-Sanchez, A.; Bueno-Perez, R.; Hamad, S.; Calero, S. *Phys. Chem. Chem. Phys.* **2011**, *13*, 11165-11174.

[16] Martin-Calvo, A.; Lahoz-Martin, F. D.; Calero, S. *J. Phys. Chem. C* **2012**, *116*, 6655.

[17] Matito-Martos, I.; Martin-Calvo, A.; Gutierrez-Sevillano, J. J.; Haranczyk, M.; Doblare, M.; Parra, J. B.; Ania, C. O.; Calero, S. *Phys. Chem. Chem. Phys.* **2014**, *16*, 19884.

[18] Matito-Martos, I.; Alvarez-Ossorio, J.; Gutierrez-Sevillano, J. J.; Doblare, M.; Martin-Calvo, A.; Calero, S. *Phys. Chem. Chem. Phys.* **2015**, *17*, 18121.

[19] Gutierrez-Sevillano, J. J.; Martin-Calvo, A.; Dubbeldam, D.; Calero, S.; Hamad, S. *Rsc Advances* **2013**, *3*, 14737.

[20] Liu, B.; Smit, B.; Rey, F.; Valencia, S.; Calero, S. *J. Phys. Chem. C* **2008**, *112*, 2492.

[21] Gutierrez-Sevillano, J. J.; Dubbeldam, D.; Rey, F.; Valencia, S.; Palomino, M.; Martin-Calvo, A.; Calero, S. *J. Phys. Chem. C* **2010**, *114*, 14907.

[22] Martin, M. G.; Siepmann, J. I. *J. Phys. Chem. B* **1998**, *102*, 2569.

[23] Martin, M. G.; Siepmann, J. I. *J. Phys. Chem. B.* **1999**, *103*, 4508.

[24] Wick, C. D.; Stubbs, J. M.; Rai, N.; Siepmann, J. I. *J. Phys. Chem. B* **2005**, *109*, 18974.

[25] Stubbs, J. M.; Potoff, J. J.; Siepmann, J. I. *J. Phys. Chem. B* **2004**, *108*, 17596.

[26] Lubna, N.; Kamath, G.; Potoff, J. J.; Rai, N.; Siepmann, J. I. *J. Phys. Chem. B* **2005**, *109*, 24100.

[27] Potoff, J. J.; Siepmann, J. I. *AIChE J.* **2001**, *47*, 1676.

[28] Shah, M. S.; Siepmann, J. I.; Tsapatsis, M. *Aiche J.* **2017**, *63*, 5098.

[29] Xu, Q.; Zhong, C. L. *J. Phys. Chem. C* **2010**, *114*, 5035.

[30] Parkes, M. V.; Staiger, C. L.; Perry IV, J. J.; Allendorf, M. D.; Greathouse, J. A. *Phys. Chem. Chem. Phys.* **2013**, *15*, 9093.

[31] Nazarian, D.; Camp, J. S.; Sholl, D. S. *Chem. Mater.* **2016**, *28*, 785.

[32] Zheng, C.; Zhong, C. *J. Phys. Chem. C* **2010**, *114*, 9945.

[33] Rana, M. K.; Pazzona, F. G.; Suffritti, G. B.; Demontis, P.; Masia, M. *J. Chem. Theory Comput.* **2011**, *7*, 1575.

[34] Gutierrez-Sevillano, J. J.; Calero, S.; Ania, C. O.; Parra, J. B.; Kapteijn, F.; Gascon, J.; Hamad, S. *J. Phys. Chem. C* **2013**, *117*, 466.

[35] Wolffis, J. J.; Vanpoucke, D. E. P.; Sharma, A.; Lawler, K. V.; Forster, P. M. *Micropor. Mesopor. Mat.* **2019**, *277*, 184.

[36] Hamad, S.; Balestra, S. R. G.; Bueno-Perez, R.; Calero, S.; Ruiz-Salvador, A. R. *J. Solid State Chem.* **2015**, *223*, 144.

[37] Duren, T.; Sarkisov, L.; Yaghi, O. M.; Snurr, R. Q. *Langmuir* **2004**, *20*, 2683.

[38] Sarkisov, L.; Duren, T.; Snurr, R. Q. *Mol. Phys.* **2004**, *102*, 211.

[39] Düren, T.; Snurr, R. Q. *J. Phys. Chem. B* **2004**, *108*, 15703.

[40] Sarkisov, L.; Düren, T.; Snurr, R. Q. *Mol. Phys.* **2004**, *102*, 211.

[41] Heinen, J.; Dubbeldam, D. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2018**, *8*, 1363.

[42] Dubbeldam, D.; Walton, K.S.; Vlugt, T.J.H.; Calero, S. *Adv. Theory Simulat.* **2019**, *2*, 1900135.

[43] Plimpton, S. *J. Comput. Phys.* **1995**, *117*, 1-19.

5 The source code

5.1 Introduction

5.2 Data types

There are several new types, the two most important ones are

- REAL
  REAL is a floating point number. It is defined in 'src/constants.h' as

  ```
  #define REAL double
  ```

  but if one needs higher precision one could use

  ```
  #define REAL long double
  ```

  and using the 'qd' library it is even possible to use arbitrary precision.

- VECTOR
  An structure with three elements 'x', 'y', and 'z'.

  ```
  typedef struct point
  {
    REAL x;
    REAL y;
    REAL z;
  } POINT,VECTOR;
  ```

- REAL_MATRIX3x3
  A 3×3 matrix, used as transformations on vectors (like 'strain') and for the three cell-vectors making up the cell matrix. It is defined in 'src/matrix.h'.

```
typedef struct real_matrix3x3
{
  REAL ax;
  REAL ay;
  REAL az;

  REAL bx;
  REAL by;
  REAL bz;

  REAL cx;
  REAL cy;
  REAL cz;
} REAL_MATRIX3x3;
```

5.3 Datastructures

Box properties and periodic boundaries

For each system, a cell box and other properties are defined in 'src/simulation.h'

```
REAL_MATRIX3x3 *Box;                  // the cell matrix
REAL_MATRIX3x3 *InverseBox;           // the inverse of the cell matrix
REAL_MATRIX3x3 *ReplicaBox;           // the cell matrix of the replica system
REAL_MATRIX3x3 *InverseReplicaBox;    // the inverse of the the cell matrix of the replica system
INT_VECTOR3 *NumberOfReplicaCells;    // the integere number of replicas in each direction a,b,c
int *TotalNumberOfReplicaCells;       // the total number of replica cells
VECTOR *ReplicaShift;                 // the shift in a,b,c for each replica cell
int *UseReplicas;                     // whether or not to use replicas
REAL_MATRIX3x3 *BoxProperties;        // properties of the cell matrix (i.e. perpendicular lengths)
REAL_MATRIX3x3 *InverseBoxProperties; // properties of the inverse cell matrix
REAL *Volume;                         // the volume
REAL *AlphaAngle;                     // the alpha-angle of the cell
REAL *BetaAngle;                      // the beta-angle of the cell
REAL *GammaAngle;                     // the gamma-angle of the cell
int *BoundaryCondition;               // the boundary condition (i.e. 'RECTANGULAR' or 'TRICLINIC')
```

These are dynamically allocated arrays and have the same length as the amount of systems present. For example, in a Gibbs simulation two systems are needed, one for the gas-phase and one for the liquid phase. 'Volume[0]' would give the volume of the first cell, and 'Volume[1]' would give the volume of the second cell.

Periodic boundaries are applied after each distance computation calling the function 'ApplyBoundaryCondition' (defined in 'src/potentials.h') It operates on a 'VECTOR' and give the corrected vector back. The system is specified with the global variable 'CurrentSystem'.

```
VECTOR ApplyBoundaryCondition(VECTOR dr)
{
  VECTOR s,t;

  switch(BoundaryCondition[CurrentSystem])
  {
    case FINITE:
      break;
    case RECTANGULAR:
    case CUBIC:
      dr.x-=Box[CurrentSystem].ax*(REAL)NINT(dr.x*InverseBox[CurrentSystem].ax);
      dr.y-=Box[CurrentSystem].by*(REAL)NINT(dr.y*InverseBox[CurrentSystem].by);
      dr.z-=Box[CurrentSystem].cz*(REAL)NINT(dr.z*InverseBox[CurrentSystem].cz);
      break;
    case TRICLINIC:
      // convert from xyz to abc
      s.x=InverseBox[CurrentSystem].ax*dr.x+InverseBox[CurrentSystem].bx*dr.y+InverseBox[CurrentSystem].cx*dr.z;
      s.y=InverseBox[CurrentSystem].ay*dr.x+InverseBox[CurrentSystem].by*dr.y+InverseBox[CurrentSystem].cy*dr.z;
      s.z=InverseBox[CurrentSystem].az*dr.x+InverseBox[CurrentSystem].bz*dr.y+InverseBox[CurrentSystem].cz*dr.z;

      // apply boundary condition
      t.x=s.x-(REAL)NINT(s.x);
      t.y=s.y-(REAL)NINT(s.y);
      t.z=s.z-(REAL)NINT(s.z);

      // convert from abc to xyz
      dr.x=Box[CurrentSystem].ax*t.x+Box[CurrentSystem].bx*t.y+Box[CurrentSystem].cx*t.z;
      dr.y=Box[CurrentSystem].ay*t.x+Box[CurrentSystem].by*t.y+Box[CurrentSystem].cy*t.z;
      dr.z=Box[CurrentSystem].az*t.x+Box[CurrentSystem].bz*t.y+Box[CurrentSystem].cz*t.z;
      break;
    default:
      fprintf(stderr,"Error: Unkown boundary condition....\n");
      exit(0);
      break;
  }
  return dr;
}
```

The function ‘NINT’ is faster version of ‘rint’ (or ‘floor’).

```
#define NINT(x) ((int)((x)>=0.0?((x)+0.5):((x)-0.5)) )
```

A common occurrence of the boundary conditions application is for two positions of atoms ‘posA’ and ‘posB’ (of type ‘VECTOR’)

```
dr.x=posA.x-posB.x;
dr.y=posA.y-posB.y;
dr.z=posA.z-posB.z;
dr=ApplyBoundaryCondition(dr);
rr=SQR(dr.x)+SQR(dr.y)+SQR(dr.z);
r=sqrt(rr);
```

There are functions you can use to transform from Cartesian to fractional coordinates (defined in ‘src/potentials.h’)

```
VECTOR ConvertFromXYZtoABC(VECTOR t)
{
  VECTOR s;

  s.x=InverseBox[CurrentSystem].ax*t.x+InverseBox[CurrentSystem].bx*t.y+InverseBox[CurrentSystem].cx*t.z;
  s.y=InverseBox[CurrentSystem].ay*t.x+InverseBox[CurrentSystem].by*t.y+InverseBox[CurrentSystem].cy*t.z;
  s.z=InverseBox[CurrentSystem].az*t.x+InverseBox[CurrentSystem].bz*t.y+InverseBox[CurrentSystem].cz*t.z;
  return s;
}
```

and from fractional coordinates to Cartesian

```
VECTOR ConvertFromABCtoXYZ(VECTOR t)
{
  VECTOR dr;

  dr.x=Box[CurrentSystem].ax*t.x+Box[CurrentSystem].bx*t.y+Box[CurrentSystem].cx*t.z;
  dr.y=Box[CurrentSystem].ay*t.x+Box[CurrentSystem].by*t.y+Box[CurrentSystem].cy*t.z;
  dr.z=Box[CurrentSystem].az*t.x+Box[CurrentSystem].bz*t.y+Box[CurrentSystem].cz*t.z;
  return dr;
}
```

(Pseudo-)atoms

The data structure ‘PSEUDO_ATOM’ contains information on atoms, either real atoms or united atoms where several atoms are lumped together (for example: CH3).

```
// Pseudoatoms
typedef struct PseudoAtom
{
  char Name[256];               // the Name of the pseudo-atom (‘CH3’,‘H’,‘O’ etc).
  char PrintToPDBName[256];     // the string to print to a pdb-file as name
  int  PrintToPDB;              // whether to write this atom to the pdf-file or not
  char ChemicalElement[256];    // the chemical element (‘O’, ‘H’, etc)
  int ScatteringType;           // the scattering type (powder diffraction)
  int AnomalousScatteringType;  // the anmalous scattering type (powder diffraction)
  REAL TemperatureFactor;       // the temperature factor (powder diffraction)
  REAL Mass;                    // the mass of the pseudo-atom
  REAL Charge;                  // the charge of the pseudo-atom
  REAL Polarization;            // the polarization of the atom
  int HasCharges;               // whether or not the atom has atoms with charges
  int IsPolarizable;            // whether or not the atom has a induced point dipole
  int Interaction;              // whether or not the atom has interactions
  REAL Radius;                  // the radius (used for calculating Bonds in the zeolite)
  int Connectivity;             // the connectivity (used for calculating Bonds/Bends/Torsion in the framework)
} PSEUDO_ATOM;
```

A typical use is, once the type is known, to retrieve the charge for a pseudo-atoms:

```
REAL q;
q=PseudoAtom[type].Charge;
```

Use the following to find out to what pseudoatom a string corresponds to

```
int type;
type=ReturnPseudoAtomNumber("CH4");
```

However, usually the type is a property of each of the atoms of a molecule.

```
int type;
type=Framework[1].Atoms[0][10].Type;
```

and ‘type’ can then be used to get the mass, charge, polarization, etc. Here, the type is retrieve for atom number 11 (c is starting from 0, unlike Fortran) of the first framework of the second system.

Framework

Atoms make up a framework, several frameworks can make up 1 system. The definition of a framework atom ‘FRAMEWORK_ATOM’ is

```
typedef struct framework_atom
{
  int Type;                       // the pseudo-atom type of the atom
  int AssymetricType;             // the ‘asymmetric’ type

  // MC/MD properties
  POINT Position;                 // the position of the atom
  POINT ReferencePosition;        // the ‘reference’ position of the atom

  // MD properties
  VECTOR Velocity;                // the velocity of the atom
  VECTOR ReferenceVelocity;       // the ‘reference’ velocity of the atom
  VECTOR Force;                   // the force acting on the atom

  VECTOR ElectricField;           // the electricfield vector
  VECTOR ReferenceElectricField;  // the ‘reference’ electricfield vector
  VECTOR InducedElectricField;    // the induced electric field
  VECTOR InducedDipole;           // the induced dipole moment on this atom
  int HessianIndex;               // the index in the Hessian matrix for this atom
} FRAMEWORK_ATOM;
```

It contains the properties you'd expect, like type, position, velocity, and force. For polarization, also electric field, induced electric field, and induced dipole are needed. For many applications, one needs to backup the positions and/or velocities. The field 'ReferencePosition' and 'ReferenceVelocity' are useful for that. Also they can be used for some algorithms which need the 'old' values to. An example is the numerical computation of stress. First all positions are copied to the 'ReferencePosition', then the positions 'Position' are generated from the strain at infinite small strain difference and the finite difference scheme is applied. A framework-structure 'FRAMEWORK_COMPONENT' is defined per system

```
FRAMEWORK_COMPONENT *Framework;
```

with

```
typedef struct FrameworkComponent
{
  char (*Name)[256];                        // the name of the frameworks

  int TotalNumberOfAtoms;                   // the total number of atoms of the frameworks
  int TotalNumberOfUnitCellAtoms;           // the total number of atoms of the unit cell
  REAL FrameworkDensity;                    // the total density of the frameworks
  REAL FrameworkMass;                       // the total mass of the frameworks

  int NumberOfFrameworks;                   // the number of frameworks
  REAL *FrameworkDensityPerComponent;       // the density per framework
  REAL *FrameworkMassPerComponent;          // the mass per framework

  int *NumberOfAtoms;                       // the number of atoms per framework
  int *NumberOfUnitCellAtoms;               // the number of unit cell atoms per framework
  FRAMEWORK_ATOM **Atoms;                   // list of framework-atoms per framework
  ..................
  ..................
} FRAMEWORK_COMPONENT;
```

The structure had the element 'Atoms' which is a list of framework-atoms per framework. So, to get the type of the 11 atom of the first framework of the second system, use

```
int type;
type=Framework[1].Atoms[0][10].Type;
```

Finally, a small example where we print out the positions of all the framework atoms for all frameworks and systems

```
int i,j,f1;
for(i=0;i<NumberOfSystem;i++)
{
  for(f1=0;f1<Framework[i].NumberOfSystems;f1++)
  {
    for(j=0;j<Framework[i].NumberOfAtoms[f1];j++)
      printf("system: %d framework: %d atom: %d -> position: %g %g %g\n",
      i,f1,j,
      Framework[i].Atoms[f1][j].Position.x,
      Framework[i].Atoms[f1][j].Position.y,
      Framework[i].Atoms[f1][j].Position.z);
  }
}
```

Components

Everything that is independent of a molecule's positions but still a property of molecules is stored in the structure 'COMPONENT'. Here you find the number of atoms for this type of molecule per system, the mass for the component etc. Also computed values for densities of the bulk fluid, compressibility, and the amount of excess molecules are stored. These are computed from the mol fraction, pressure, and critical pressure/temperature and acentric factor. After these properties there are data on the potentials defined for the component: bond, Urey-Bradley, bends, torsions, cross-terms, intra Van der Waals etc. For Monte Carlo the structure contains the probability of all the moves.

```
typedef struct Component
{
  char Name[256];                 // the name of the component ("methane","C12","propane" etc).
  int NumberOfAtoms;              // the number of atoms in the component
  int StartingBead;               // the bead of the molecule used for starting the growing process in CBMC
  REAL Mass;                      // the mass of the component
  int *NumberOfMolecules;         // the number of molecules of the component for each system
  int *Type;                      // the pseudo-atom Type of each atom
  int *Connectivity;              // the connectivity of each atom
  int HasCharges;                 // whether the molecule contains charges or not
  int IsPolarizable;              // whether the molecule has point dipoles or not
  int ExtraFrameworkMolecule;     // TRUE: Cation, FALSE: Adsorbate
  int Swapable;                   // whether or not the number of molecules is fluctuating (i.e. GCMC)
  int Widom;                      // whether this component is used for Widom insertions

  REAL *IdealGasRosenbluthWeight; // the Rosenbluth weight of an ideal-chain per system
  REAL *IdealGasTotalEnergy;      // the total energy of an ideal-chain per system

  REAL *PartialPressure;          // the partial pressure of the component per system
  REAL *FugacityCoefficient;      // the fugacity coefficient of the component per system
  REAL *BulkFluidDensity;         // the bulkfluid-density of the component per system
  REAL *Compressibility;          // the compresibility of the fluid-fase per system
  REAL *MolFraction;              // the mol-fraction of the component per system
  REAL *AmountOfExcessMolecules;  // the amount of excess molecules per syste,

  REAL CriticalTemperature;       // the critical temperature of the component
  REAL CriticalPressure;          // the critical pressure of the component
  REAL AcentricFactor;            // the acentric factor of the component

  int NumberOfGroups;             // the number of groups
  GROUP_DEFINITION *Groups;       // the definition of the groups
  int *group;                     // to which group an atom belongs
  VECTOR *Positions;              // the positions in the body-fixed frame
     ..................
     ..................
  int NumberOfBonds;                                  // the number of bonds of the component
  PAIR *Bonds;                                        // the list of bond-pairs
  int *BondType;                                      // the type of the bond for each bond-pair
  REAL (*BondArguments)[MAX_BOND_POTENTIAL_ARGUMENTS]; // the arguments needed for this bond-pair
     ..................
     ..................
  REAL ProbabilityTranslationMove;  // the probability of the translation MC-move for the component
  REAL ProbabilityRotationMove;     // the probability of the rotation MC-move for the component
  REAL ProbabilityCBMCMove;         // the probability of the partial-regrow MC-move for the component
  REAL ProbabilityReinsertionMove;  // the probability of the reinsertion MC-move for the component
     ..................
     ..................
} COMPONENT;
```

A component consists of 'groups', which is a collection of atoms that are either treated as rigid or as flexible. The component has elements that lists how many of these groups there are, the definition of the group, and the positions of all the atoms in the body-fixed frame. The definition of the group is the structure 'GROUP_DEFINITION'. Important elements are whether or not the group is rigid, the number of atoms in the group, and the list of atom number present in the groups.

```
typedef struct group_definitions
{
  int Rigid;                        // whether or not the group is rigid
  int Type;                         // the type, NONLINEAR_MOLECULE, LINEAR_MOLECULE, or POINT_PARTICLE

  REAL Mass;                        // the mass of the group

  int NumberOfGroupAtoms;           // the numer of atoms in the group
  int *Atoms;                       // the atoms in the group

  REAL_MATRIX3x3 InertiaTensor;     // the inertia tensor
  VECTOR InertiaVector;             // the inertia vector
  VECTOR InverseInertiaVector;      // the inverse of inertia vector

  REAL_MATRIX3x3 RotationalMatrix;  // the rotational matrix
  TRIPLE orientation;               // three atoms A,B,C to compute quaternions
  REAL rot_min;

  int RotationalDegreesOfFreedom;   // the rotational degrees of freedom
} GROUP_DEFINITION;
```

The inertia tensor, vector and rotational matrix etc. are the same for a certain type of molecule. Together with the actually atom positions, the orientations can be computed for all the rigid units (i.e. the quaternions are computed).

Adsorbate and cations

The definition of an adsorbate atom 'ADSORBATE_ATOM' is very similar to a framework atom

```
typedef struct adsorbate_atom
{
  int Type;                       // the pseudo-atom type of the atom

  // MC/MD properties
  POINT Position;                 // the position of the atom
  POINT ReferencePosition;        // the 'reference' position of the atom

  // MD properties
  VECTOR Velocity;                // the velocity of the atom
  VECTOR ReferenceVelocity;       // the 'reference' velocity of the atom
  VECTOR Force;                   // the force acting on the atom

  VECTOR ElectricField;           // the electricfield vector
  VECTOR ReferenceElectricField;  // the 'reference' electricfield vector
  VECTOR InducedElectricField;    // the induced electric field
  VECTOR InducedDipole;           // the induced dipole moment on this atom
  int HessianIndex;               // the index in the Hessian matrix for this atom
} ADSORBATE_ATOM;
```

The definition for cations is identical except it is called 'CATION_ATOM'. The definition of an adsorbate molecule is

```
typedef struct adsorbate
{
  int Type;               // the component type of the molecule
  int NumberOfAtoms;      // the number of atoms in the molecule
  GROUP *Groups;          // data of the rigid groups
  ADSORBATE_ATOM *Atoms;  // list of atoms
} ADSORBATE_MOLECULE;
```

The definition of a cation is called 'CATION_MOLECULE'. Note that a molecule can consists of atoms, but also can contain rigid units. The atoms are accessible through the 'Atoms' field, and rigid units are accessible through the 'Groups' field. A 'GROUP' consists of

```
typedef struct group
{
  REAL Mass;                             // mass of the rigid unit
```

```
    QUATERNION Quaternion;                // orientation of the unit
    QUATERNION QuaternionMomentum;        // quaternion momentum
    QUATERNION QuaternionForce;           // quaternion force
    VECTOR Torque;                        // torque vector
    VECTOR CenterOfMassPosition;          // the center of mass position
    VECTOR CenterOfMassReferencePosition; // the reference position for the center of mass
    VECTOR CenterOfMassVelocity;          // the center of mass velocity
    VECTOR CenterOfMassForce;             // the center of mass force
    VECTOR AngularVelocity;               // the angular velocity of the rigid unit
  } GROUP;
```

which contains elements like position and orientation, and fields for the integration of rigid units, i.e. QuaternionMomentum etc.
Molecules are stored as a list of molecules for each system

```
ADSORBATE_MOLECULE **Adsorbates;
```

To get the type of the 5th atom of the 11th adsorbate of the first system, use

```
int type;
type=Adsorbates[0][10].Atoms[4].Type;
```

As an example, here a function to measure the velocity drift of all the adsorbates in the current system

```
VECTOR MeasureVelocityDrift(void)
{
  int i,k,l,Type,A,f;
  REAL Mass,TotalMass;
  VECTOR com;

  TotalMass=0.0;
  com.x=com.y=com.z=0.0;
  for(i=0;i<NumberOfAdsorbateMolecules[CurrentSystem];i++)
  {
    Type=Adsorbates[CurrentSystem][i].Type;
    for(l=0;l<Components[Type].NumberOfGroups;l++)
    {
      if(Components[Type].Groups[l].Rigid)
      {
        Mass=Components[Type].Groups[l].Mass;
        TotalMass+=Mass;
        com.x+=Mass*Adsorbates[CurrentSystem][i].Groups[l].CenterOfMassVelocity.x;
        com.y+=Mass*Adsorbates[CurrentSystem][i].Groups[l].CenterOfMassVelocity.y;
        com.z+=Mass*Adsorbates[CurrentSystem][i].Groups[l].CenterOfMassVelocity.z;
      }
      else
      {
        for(k=0;k<Components[Type].Groups[l].NumberOfGroupAtoms;k++)
        {
          A=Components[Type].Groups[l].Atoms[k];
          Mass=PseudoAtoms[Adsorbates[CurrentSystem][i].Atoms[A].Type].Mass;
          TotalMass+=Mass;
          com.x+=Mass*Adsorbates[CurrentSystem][i].Atoms[A].Velocity.x;
          com.y+=Mass*Adsorbates[CurrentSystem][i].Atoms[A].Velocity.y;
          com.z+=Mass*Adsorbates[CurrentSystem][i].Atoms[A].Velocity.z;
        }
      }
    }
  }
  com.x/=TotalMass;
  com.y/=TotalMass;
  com.z/=TotalMass;
  return com;
}
```

It loops over all the adsorbate molecules, and asks for the type. The component-type is important to get the number of groups for the current molecule. Then, there is a inner loop over all of the groups of the current molecule. If the group is rigid, then the center of mass velocity is used, otherwise it is flexible and it loops over all the atoms of the flexible group. In general, if something is the same for a type of molecule then it is a property of the component. If it is different for each molecule, it is a property of a molecule.

5.4 Modifying

5.4.1 Monte Carlo

Selecting MC moves

The file 'src/monte_carlo.c' is the main Monte Carlo simulation routine. The bulk of the code deals with how to select a particular Monte carlo move. Some requirements and conveniences:

- The moves should be chosen in random order
- System move should be chosen much less frequent than particle moves. The particles need to be able to adapt to the new system.
- For n systems, the amount of steps should be n times larger.
- For n times as many molecules, the amount of steps should be n times larger.
- For multi-component systems one needs more steps.
- For systems at low loadings, the sampling lengths should be increase a bit (i.e. set a minimum amount of inner steps).
- The relative probabilities of particle moves should be taken into account.

A code which achieves all the above is listed here (there are many other ways of doing this). For each MC 'cycle'

```
for(i=0;i<NumberOfSystems;i++)
{
  // choose system at random
  CurrentSystem=(int)(RandomNumber()*(REAL)NumberOfSystems);

  NumberOfSystemMoves=9;
  NumberOfMolecules=NumberOfAdsorbateMolecules[CurrentSystem]+NumberOfCationMolecules[CurrentSystem];
  NumberOfParticleMoves=MAX(MinimumInnerCycles,NumberOfMolecules);
  NumberOfSteps=(NumberOfSystemMoves+NumberOfParticleMoves)*NumberOfComponents;

  // loop over the MC 'steps' per MC 'cycle'
  for(j=0;j<NumberOfSteps;j++)
  {
    // choose any of the MC moves randomly
    ran_int=(int)(RandomNumber()*NumberOfSteps);
    switch(ran_int)
    {
      case 0: if(RandomNumber()<ProbabilityParallelTemperingMove) ParallelTemperingMove(); break;
      case 1: if(RandomNumber()<ProbabilityHybridNVEMove) HybridNVEMove(); break;
      case 2: if(RandomNumber()<ProbabilityHybridNPHMove) HybridNPHMove(); break;
      case 3: if(RandomNumber()<ProbabilityHybridNPHPRMove) HybridNPHPRMove(); break;
      case 4: if(RandomNumber()<ProbabilityVolumeChangeMove) VolumeMove(); break;
      case 5: if(RandomNumber()<ProbabilityBoxShapeChangeMove) BoxShapeChangeMove(); break;
      case 6: if(RandomNumber()<ProbabilityGibbsVolumeChangeMove) GibbsVolumeMove(); break;
      case 7: if(RandomNumber()<ProbabilityFrameworkChangeMove) FrameworkChangeMove(); break;
      case 8: if(RandomNumber()<ProbabilityFrameworkShiftMove) FrameworkShiftMove(); break;
      default:
        // choose component at random
        CurrentComponent=(int)(RandomNumber()*(REAL)NumberOfComponents);

        // choose the Monte Carlo move at random
        ran=RandomNumber();
        if(ran<Components[CurrentComponent].ProbabilityTranslationMove) TranslationMove();
        else if(ran<Components[CurrentComponent].ProbabilityRandomTranslationMove) RandomTranslationMove();
        else if(ran<Components[CurrentComponent].ProbabilityRotationMove) RotationMove();
        else if(ran<Components[CurrentComponent].ProbabilityCBMCMove) CBMCMove();
        else if(ran<Components[CurrentComponent].ProbabilityReinsertionMove) ReinsertionMove();
        else if(ran<Components[CurrentComponent].ProbabilityReinsertionInPlaceMove) ReinsertionInPlaceMove();
        else if(ran<Components[CurrentComponent].ProbabilityReinsertionInPlaneMove) ReinsertionInPlaneMove();
        else if(ran<Components[CurrentComponent].ProbabilityIdentityChangeMove) IdentityChangeMove();
        else if(ran<Components[CurrentComponent].ProbabilitySwapMove)
        {
          if(RandomNumber()<0.5) SwapAddMove();
          else SwapRemoveMove();
        }
        else if(ran<Components[CurrentComponent].ProbabilityWidomMove) WidomMove();
        else if(ran<Components[CurrentComponent].ProbabilitySurfaceAreaMove) SurfaceAreaMove();
        else if(ran<Components[CurrentComponent].ProbabilityGibbsSwapChangeMove) GibbsParticleTransferMove();
        else if(ran<Components[CurrentComponent].ProbabilityGibbsIdentityChangeMove) GibbsIdentityChangeMove();
        break;
    }
  }
}
```

First is a loop over the amount of systems, and a random system is chosen. Suppose we have 200 single component molecules in this system, then each of the system move is chosen with 1/209 probability (case 0-8), and there is a 200/209 chance to select a particle move (case 9-209). The probability of the particle moves are scaled in such a way that the proper relative occurrence is obeyed (as specified in the input). Note that the swap-move has 50% to be swap insertion and 50% to be swap remove. This is necessary to obey detailed balance. For multi-components more moves are performed.

Sampling properties during Monte Carlo

The Monte Carlo routine has two parts:

- The initialization part. Here, no properties are computed and MC moves are performed just to reach equilibrium.
- The production run, where properties are computed.

The basic outline of the production run is

```
// initialize sampling-routines at the start of the production run
SampleInfraRedSpectra(INITIALIZE);
SampleMeanSquareDisplacementOrderN(INITIALIZE);
SampleOnsagerMeanSquareDisplacementOrderN(INITIALIZE);
SampleRadialDistributionFunction(INITIALIZE);
SampleFrameworkSpacingHistogram(INITIALIZE);
SamplePositionHistogram(INITIALIZE);
SampleNumberOfMoleculesHistogram(INITIALIZE);
SampleEnergyHistogram(INITIALIZE);
SampleDensityProfile3DVTKGrid(INITIALIZE);
SampleEndToEndDistanceHistogram(INITIALIZE);
SampleMoleculePropertyHistogram(INITIALIZE);
SamplePDBMovies(INITIALIZE);
SampleDcTSTConfigurationFiles(INITIALIZE);
SampleFreeEnergyProfile(INITIALIZE);
SampleCationAndAdsorptionSites(INITIALIZE);

for(CurrentCycle=0;CurrentCycle<NumberOfCycles;CurrentCycle++)
```

```
{
  // sample energy average and system/particle properties
  for(CurrentSystem=0;CurrentSystem<NumberOfSystems;CurrentSystem++)
  {
    UpdateEnergyAveragesCurrentSystem();

    SampleRadialDistributionFunction(SAMPLE);
    SampleFrameworkSpacingHistogram(SAMPLE);
    SamplePositionHistogram(SAMPLE);
    SampleNumberOfMoleculesHistogram(SAMPLE);
    SampleEnergyHistogram(SAMPLE);
    SampleDensityProfile3DVTKGrid(SAMPLE);
    SampleEndToEndDistanceHistogram(SAMPLE);
    SampleMoleculePropertyHistogram(SAMPLE);
    SampleFreeEnergyProfile(SAMPLE);
    SampleCationAndAdsorptionSites(SAMPLE);
  }

  // SELECTION OF MC-MOVES (SEE CODE OF THE PREVIOUS SECTION)

  for(CurrentSystem=0;CurrentSystem<NumberOfSystems;CurrentSystem++)
  {
    SampleRadialDistributionFunction(PRINT);
    SampleFrameworkSpacingHistogram(PRINT);
    SamplePositionHistogram(PRINT);
    SampleNumberOfMoleculesHistogram(PRINT);
    SampleEnergyHistogram(PRINT);
    SampleDensityProfile3DVTKGrid(PRINT);
    SampleEndToEndDistanceHistogram(PRINT);
    SampleMoleculePropertyHistogram(PRINT);
    SamplePDBMovies(PRINT);
    SampleDcTSTConfigurationFiles(PRINT);
    SampleFreeEnergyProfile(PRINT);
    SampleCationAndAdsorptionSites(PRINT);
  }
}

// finalize output
SampleRadialDistributionFunction(FINALIZE);
SampleFrameworkSpacingHistogram(FINALIZE);
SamplePositionHistogram(FINALIZE);
SampleNumberOfMoleculesHistogram(FINALIZE);
SampleEnergyHistogram(FINALIZE);
SampleDensityProfile3DVTKGrid(FINALIZE);
SampleEndToEndDistanceHistogram(FINALIZE);
SampleMoleculePropertyHistogram(FINALIZE);
SamplePDBMovies(FINALIZE);
SampleDcTSTConfigurationFiles(FINALIZE);
SampleFreeEnergyProfile(FINALIZE);
SampleCationAndAdsorptionSites(FINALIZE);
```

Each of the sampling routine (in 'src/sample.c') has 5 scaling options:

- ALLOCATE to allocate memory needed for the sampling.
- INITIALIZE to initialized the routine if needed.
- SAMPLE to sample the properties.
- PRINT to periodically write the output to file.
- FINALIZE to free the requested memory and clean up.

Adding your own sampling routines requires an additional routine in 'src/sample.c', the definition in 'src/sample.h' and addition to calls to 'src/monte_carlo.c'.

5.4.2 Molecular Dynamics

A molecular dynamics simulation is performed in several steps:

- The proper amount of molecules are created and they are inserted as as no overlaps occurred with the framework or other particles.
- Initialization: during the initialization period an NVT Monte-Carlo (MC) simulation is performed to rapidly achieve an equilibrium molecular arrangement.
- After the initialization period, velocities are assigned, drawn from the Maxwell-Boltzmann distribution at the desired average temperature to all the atoms. The total momentum of the system can be set to zero.
- Equilibration: Next, the system is further equilibrated by performing an NVT MD simulation using a specified ensemble.
- Production run: the simulation is performed in the requested ensemble and properties are measured.

The amount of cycles for each of these steps can be specified. For example, when starting from a restart-file there is no need for the Monte Carlo initialization, and if also the velocities are used from the restart-file then also the MD equilibration could be skipped. Moreover, the equilibration can be done in a different ensemble as the production run. This is most useful for NVE simulations, where the equilibration could be done using NVT. The final temperature of the NVE production run is then quite close the desired temperature (in NVE the temperature is not imposed).

The initialization part is not shown here, as it is very similar to regular Monte Carlo. The basic outline for the equilibration and production run are listed below. The most important lines are the 'Integration();' ones, which evolve the system a single time step. This routine is implemented in 'src/integration.c' and makes use of 'src/thermo_baro_stats.c' for temperature and pressure control.

```
// initialize
InitializesEnergiesAllSystems();
InitializeSmallMCStatisticsAllSystems();
InitializeMCMovesStatisticsAllSystems();

// compute initial energy
InitializeNoseHooverAllSystems();
InitializeForcesAllSystems();

// set the current ensemble to the initialization ensemble
for(i=0;i<NumberOfSystems;i++)
  Ensemble[i]=InitEnsemble[i];

InitializesEnergyAveragesAllSystems();

for(CurrentSystem=0;CurrentSystem<NumberOfSystems;CurrentSystem++)
{
  ReferenceEnergy[CurrentSystem]=ConservedEnergy[CurrentSystem];
  Drift[CurrentSystem]=0.0;
}

// Molecular-Dynamics initializing period to achieve a rapid equilibration of the velocities
for(CurrentCycle=0;CurrentCycle<NumberOfEquilibrationCycles;CurrentCycle++)
{
  for(CurrentSystem=0;CurrentSystem<NumberOfSystems;CurrentSystem++)
  {
    // regularly output system status and restart files
    if(CurrentCycle%PrintEvery==0)
    {
      PrintIntervalStatusEquilibration(CurrentCycle,NumberOfEquilibrationCycles,OutputFilePtr[CurrentSystem]);
      PrintRestartFile();
    }

    // evolve the system a full time-step
    Integration();

    // update the current energy-drift
    Drift[CurrentSystem]+=fabs((ConservedEnergy[CurrentSystem]-ReferenceEnergy[CurrentSystem])/
          ReferenceEnergy[CurrentSystem]);
  }
}


// initialize sampling-routines at the start of the production run
for(CurrentSystem=0;CurrentSystem<NumberOfSystems;CurrentSystem++)
{
  Ensemble[CurrentSystem]=RunEnsemble[CurrentSystem];

  ReferenceEnergy[CurrentSystem]=ConservedEnergy[CurrentSystem];
  Drift[CurrentSystem]=0.0;
}
SampleInfraRedSpectra(INITIALIZE);
SampleEndToEndDistanceHistogram(INITIALIZE);
SampleMeanSquareDisplacementOrderN(INITIALIZE);
SampleOnsagerMeanSquareDisplacementOrderN(INITIALIZE);
SampleEnergyHistogram(INITIALIZE);
SamplePositionHistogram(INITIALIZE);
SampleRadialDistributionFunction(INITIALIZE);
SamplePositionHistogram(INITIALIZE);
SampleMoleculePropertyHistogram(INITIALIZE);
SamplePDBMovies(INITIALIZE);
SampleCationAndAdsorptionSites(INITIALIZE);

// Molecular-Dynamics production run
// loop over the amount of production cycles (MD integration steps)
for(CurrentCycle=0;CurrentCycle<NumberOfCycles;CurrentCycle++)
{
  // loop over all the systems and handle one by one
  for(CurrentSystem=0;CurrentSystem<NumberOfSystems;CurrentSystem++)
  {
    SampleInfraRedSpectra(SAMPLE);
    SampleEndToEndDistanceHistogram(SAMPLE);
    SampleMeanSquareDisplacementOrderN(SAMPLE);
    SampleOnsagerMeanSquareDisplacementOrderN(SAMPLE);
    SampleEnergyHistogram(SAMPLE);
    SamplePositionHistogram(SAMPLE);
    SampleRadialDistributionFunction(SAMPLE);
    SamplePositionHistogram(SAMPLE);
    SampleMoleculePropertyHistogram(SAMPLE);
    SampleCationAndAdsorptionSites(SAMPLE);

    // update all the average energies
    UpdateEnergyAveragesCurrentSystem();

    if(CurrentCycle%PrintPropertiesEvery==0)
      PrintPropertyStatus(CurrentCycle,NumberOfCycles,OutputFilePtr[CurrentSystem]);

    if(CurrentCycle%PrintEvery==0)
    {
      PrintIntervalStatus(CurrentCycle,NumberOfCycles,OutputFilePtr[CurrentSystem]);
      PrintRestartFile();
    }

    // regulary output radial distribution function
    SampleInfraRedSpectra(PRINT);
    SampleEndToEndDistanceHistogram(PRINT);
```

```
        SampleMeanSquareDisplacementOrderN(PRINT);
        SampleOnsagerMeanSquareDisplacementOrderN(PRINT);
        SampleEnergyHistogram(PRINT);
        SamplePositionHistogram(PRINT);
        SampleRadialDistributionFunction(PRINT);
        SamplePositionHistogram(PRINT);
        SampleMoleculePropertyHistogram(PRINT);
        SamplePDBMovies(PRINT);
        SampleCationAndAdsorptionSites(PRINT);

        // evolve the current system a full time step
        Integration();

        // update the current energy-drift
        Drift[CurrentSystem]+=fabs((ConservedEnergy[CurrentSystem]-ReferenceEnergy[CurrentSystem])/
              ReferenceEnergy[CurrentSystem]);
      }
    }

    // finalize and clean up
    for(CurrentSystem=0;CurrentSystem<NumberOfSystems;CurrentSystem++)
    {
      SampleInfraRedSpectra(FINALIZE);
      SampleEndToEndDistanceHistogram(FINALIZE);
      SampleMeanSquareDisplacementOrderN(FINALIZE);
      SampleOnsagerMeanSquareDisplacementOrderN(FINALIZE);
      SampleEnergyHistogram(FINALIZE);
      SamplePositionHistogram(FINALIZE);
      SampleRadialDistributionFunction(FINALIZE);
      SamplePositionHistogram(FINALIZE);
      SampleMoleculePropertyHistogram(FINALIZE);
      SamplePDBMovies(FINALIZE);
      SampleCationAndAdsorptionSites(FINALIZE);
    }
```

Adding your own sampling routines requires an additional routine in 'src/sample.c', the definition in 'src/sample.h' and addition to calls to 'src/molecular_dynamics.c'.

5.5 Debugging

5.5.1 Linux

There are several debuggers like 'gdb', and memory check utilities available, i.e. valgrind.

5.5.2 Mac OSX

Debugging memory error under Max OsX is easy. One can replace the standard library to allocate memory by different ones that check memory allocation and use. It can catch a lot of array out-of-bound error, even for dynamically allocated memory. See

```
man libgmalloc
```

An example, export 'RASPA_DIR' to the installation directory, start the debugger, load the debugging libraries and start running the code.

```
export RASPA_DIR=${HOME}/RASPA/simulations/
gdb ~/RASPA/simulations/bin/simulate

GNU gdb 6.3.50-20050815 (Apple version gdb-768) (Tue Oct  2 04:07:49 UTC 2007)
```

```
Copyright 2004 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are
welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB.  Type "show warranty" for details.
This GDB was configured as "i386-apple-darwin"...Reading symbols for shared libraries ... done

(gdb) set env DYLD_INSERT_LIBRARIES /usr/lib/libgmalloc.dylib
(gdb) r
```

6 Troubleshooting

The numerical value computed from finite differences is not equal to the analytical expression Using 'SimulationType Numerical' the analytical expression for the force, stress etc. are compared to numerical values from finite differences. If just one (or at the most a few values) are different, then this might be an artifact arising from a finite cutoff in the Van der Waals potential. This can be checked by changing the value of the cutoff by about 10^{-3} Å. This is a very small change, but larger than the displacements used in the finite differences. The problem is that for a finite difference scheme like:

$$f'(x) = \frac{f(x-\Delta) - 8f\left(x - \frac{1}{2}\Delta\right) + 8f\left(x + \frac{1}{2}\Delta\right) - f(x+\Delta)}{6\Delta} \tag{6.1}$$

it is possible that one of the displacements Δ places the particle outside of the cutoff, while the original position was inside (or visa versa). For a force-shifted Van der Waals potential there is no problem, but for shifted potentials, or potentials with a simple truncation, the divergence becomes a problem.

Excess loading is negative This usually happens when computing an isotherm and the next pressure is above the vapor pressure. The boundary from gas to liquid adsorption has been crossed and the amount of excess molecules increases by orders of magnitude. There is a reason why experimental gas-phase isotherms are of finite range, they usually stop at the vapor pressure. Also, if the pressure is *very* high the fluid outside the crystal is compressed more and more while the loading inside the crystal remains the same (at maximum loading). Hence, excess adsorption evantually becomes negative.

Large drift in Monte Carlo energies This should *not* happen and signals an error in (one of) the Monte Carlo routines. During the Monte Carlo simulations, the running-energies are stored. These are starting energy, and all the added energy *differences*. At the final stage, the energy is recomputed again, and these should match within an error of about 10^{-5} or lower. If you have added your own MC move, check whether you have properly added the energy differences to the running energies.

Energy is not conserved in molecular dynamics Usually, this happens because the time step is too large. Also, at initialization, the system can be far from equilibrated and a smaller time step is needed.

RASPA ”hangs” at initialization Put 'CreateNumberOfMolecules 0' and check if that solves the problem. If so, then you have tried to add too many molecules in the system (i.e. more than actually fit in the system). For systems without a framework, one can also increase the size of the box.

Segmentation fault A memory access that is not allowed has occured. This could happen when the input is incorrect. For example, if it is listed that there are 4 bonds, but you put in 5 lines, then all bends and what follows next will be read in wrong. This is the most common cause of segmentation faults.

Mean-square displacement is not linear There are several known causes:

- Your system is one-dimensional and particles are unable to pass each other. This is known as 'single-file-diffusion' and the mean square displacement is propertial to the square root of time,
- You did not simulate long enough. In some systems it can take up to several nanoseoconds before the msd becomes linear in time,
- You forgot to specify interactions between the molecules and they are not interacting.

Minimization does not converge A likely cause is that you minimize a system that would like to change angles, but you do not allow it to. In such a system, there is a non-vanishing stress. Try to minimize using NPT-PR with cell type 'Regular' or 'RegularUppertriangle'. Another reason could be that the electrostatics are not computed accurate enough. Increase the precision to 1e-10, using 'EwaldPrecision 1e-10'.

Output is not written to file Check with 'df -k' whether the disk is full.

Molecule can not be grown Check if the connectivity, i.e. the bonds, are correct.

Framework flies apart Check bonds for the framework and whether electrostatics and intra framework Van der Waals interactions are computed.

Energy during molecular dynamics with a flexible framework is not well conserved In zeolites, a common problem is that the the angle of a Si-O-Si bend can become 180 degrees. This leads to a undefined torsion angle. If this occures, try to use a smoothing function that slowly switches of the energy and force contributions for these 3 atoms as the angle approaches 180 degrees. See Bend/Torsion cross potentials.

Amount of detected bonds/bends/torsions etc. for a flexible framework is wrong For the detection of intra-framework potentials a connectivity tabls is made, where two are considered bonded when their length is smaller than $0.56 + r_i + r_j$, where r_i and r_j are the covalent radii for the two atoms. The radii are specified in 'pseudo_atoms.def'. The most likely cause is a wrong value for the radius. Note that even when starting from a restart-file, the connectivity table is based on the crystal structure.

Oxgens connected to aluminum type 'O' are not automatically converted to 'Oa' Use the option

```
ModifyOxgensConnectedToAluminium yes
```

Strange behaviour when using cations The problem could be related to CBMC of net-charged molecules. The Rosenbluth weights can become very large or very small, because the energy difference when displacing an ion is large. This can lead to numerical problems for ratio's of combination of small/large, for example in the reinsertion move. To see whether this is the cause use 'RandomTranslationProbability' and set 'ReinsertionProbability' to zero.

Minimization does not converge Minimization code and algorithms are complex. Due to the harmonic approximation the jumps through the energy landscape can not be too large. A possible remedy therefore is to decrease the maximum step-length (default 0.3) using

```
    MaximumStepLength 0.1
```

Another issue is the rotational degrees of freedom of the system. For periodic systems the system is invariant with respect to translation but not with respect to rotation, i.e. the energy changes for rotation of the whole system. In contrast, a molecule in a finite system is invariant with respect to both translation and rotation. However, for periodic systems at low loading with molecules without charges, groups/clusters of molecules can occur that do not have interactions with their images. In effect this has reduced the periodic system to a non-periodic one. If this occurs one can remove the system rotation explicitly with

```
    RemoveRotationFromHessian yes
```

Note that this option should *not* be used on a truly periodic system. In addition to these algorithm settings, it is possible that the definition of the molecule and/or framework contains errors.

Parallel tempering does not work for systems with cations The problem is physical, it is just that all the energy distributions are more 'spiked' and overlap between the energies of the systems is rare. The only solution is to use more systems (and smaller temperature differences) to increase the acceptance rates of swapped between neighboring system. The same problem happens when one increases the system size.

Results do not match data from literature Common reasons include difference in simulation length, system size, cutoff value, tail-corrections vs shifted potentials, and handling of electrostatics. For adsorption, is the crystal structure that you used the same? Another very often made error, is comparing against different units, or an error in the conversion of units.

Error during compiling The 'gcc' and 'icc' compilers are tested. These compilers have C extensions that other C compilers could potentially lack.

Error during linking Make sure that the 'blas' and 'lapack' libraries are installed on your system.

The results are different on different machines We have come across one compiler-error: gcc 4.3.0 using optimizations "-O3" and "-O4" generated wrong code. Gcc 4.3.2 has resolved this bug.

The program crashes with a 'segmentation fault' Usually caused by wrong input-files, for example supplying 3 arguments to a torsion when 4 are expected. This causes all input to have strange values. To identify the cause make sure the job will use the same random number sequence are written in the output.

```
RandomSeed [int]
```

This will generate exactly the same sequence of events. Make sure the program is allowed to 'dump a core' (See the unix 'ulimit' command). Also, the executable needs to be compiled with the '-g' option which includes debugger information into the exectubale. Now restart the program and when it crashes, it will write a 'core-dump-file'. Start the debugger using a command similar to

```
gdb ~/RASPA/simulate/bin/simulate core
```

and type 'where' to obtain the line where the code crashed. It is also possible to just run the code inside the debugger.

Part II

Utilities

7 Visualization

7.1 Making pictures using VTK

VTK is a nice visualization toolkit tailored for scientific purposes. It builds on top of OpenGL and is available for most platforms. One of the most useful features is the ability to define scientific data in e.g. grid form (”structured points”) and to manipulate that data. Each grid point can contain various data forms: scalars like temperature and pressure, but also vector data like velocity or fields.

There are several ways to visualize frameworks in RASPA using VTK:

- Ball and stick
  RASPA will output vtk-files for all the molecules in the system as well as the framework itself.

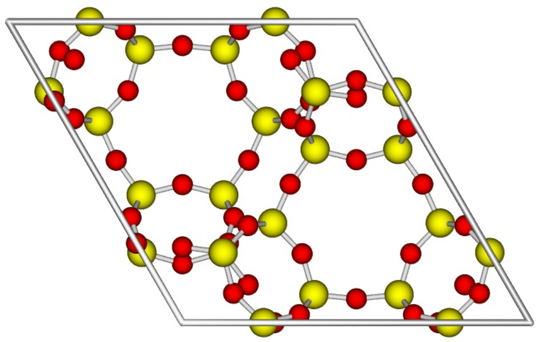
- Volume rendered surface area
  RASPA will output a ”structured point” grid of the adsorption energy. The structure is probed using Widom insertion at random positions and the result is averaged. The lowest and highest values are recorded and then scaled between 0 and 2^{16}.

- Volume rendered density plots of adsorbates
  RASPA computes a 3D histogram of the positions of adsobates per component and for the total fluid. This type of plots are very useful to find out where and how the molecules adsorb.

7.2 Ball and stick

At the start of any run, RASPA outputs the current state in VTK files, located in ‘VTK/System[int]’. The files are ‘FrameworkAtoms.vtk’, ‘AdsorbateAtoms.vtk’, ‘CationAtoms.vtk’, and ‘Frame.vtk’. In ‘Example/Visualization/BallStickRASPA’ the example for erionite is shown.

```
SimulationType                MC
NumberOfCycles                0

Forcefield                    ExampleZeolitesForceField

Framework 0
```


(a)

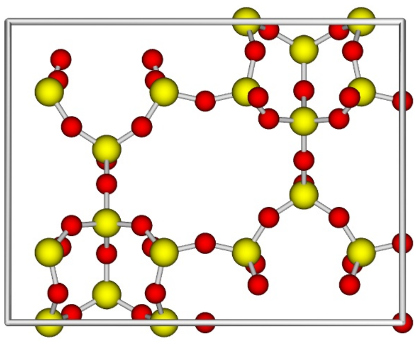

(b)

Figure 14: *Ball and stick picture of erionite (ERI): (a) front view, (b) side view. The erionite structure is monoclinic:* $a = b = 13.27$ *Å and* $c = 15.05$ *Å,* $\alpha = \beta = 90^\circ$ *and* $\gamma = 120^\circ$.

```
    FrameworkName ERI_mono
    UnitCells 1 1 1
```

After copying the vtk-files to 'Examples/Visualization/ERI/VTK' one can run the VTK code. The VTK program will produce a picture 'Picture.jpg' and looks like Figure 14. The file 'Frame.vtk' looks like:

```
    # vtk DataFile Version 1.0
    Frame
    ASCII

    DATASET POLYDATA
    POINTS 8 float
    50.000000 -0.000000 -0.000000
    150.000000 0.000000 0.000000
    100.000000 86.602540 0.000000
    0.000000 86.602540 0.000000
    50.000000 0.000000 113.413715
    150.000000 0.000000 113.413715
    100.000000 86.602540 113.413715
    0.000000 86.602540 113.413715
    LINES 6 36
    5 0 1 2 3 0
    5 4 5 6 7 4
    5 0 1 5 4 0
    5 2 3 7 6 2
    5 0 4 7 3 0
    5 1 2 6 5 1
```

It contains 8 points: the corners of the frame, and 6 closed poly-lines that form the ribbons. Using the 'vtk-TubeFilter' we can use these lines to turn them into bigger tubes and color the tubes white. The coordinate system is chosen as $150 \times 150 \times 150$ to be compatible with structure grids (for the density and surface). The information about the framework is listed in 'FrameworkAtoms.vtk':

```
    # vtk DataFile Version 1.0
```

```
Cube
ASCII

DATASET POLYDATA
POINTS 108 float
38.346000 20.221693 11.847197
38.125000 36.762778 28.353429
35.205000 30.250267 18.259608
....
....
LINES 232 696
2 0 2
2 0 3
2 0 13
....
....
POINT_DATA 108
SCALARS my_scalars float
LOOKUP_TABLE default
2.1
2.1
1.52
....
....
VECTORS vectors float
0.125 0 0
0.125 0 0
0.03125 0 0
....
....
```

The first points are the 108 framework atoms, next the lines section describes the bonding between them. The last two sections denote the size and color of the atoms (Note that the VECTORS section is a trick to allow the VTK 'glyphs', here spheres, to be scaled by the scalar data, but colored by the magnitude of the VECTOR data. Hopefully this will be easier in future versions of VTK).

The VTK program is also interactive, one can zoom in and out (scroll button) and rotate (click on the canvas, closer to the center rotates less then further away). In computer graphics, a sphere is not a sphere, but a collection of polygons. More polygons means a smoother surface but less responsive in the interactive mode. For final pictures, one should use many polygons and anti-aliasing, which really improve the quality of the picture.

The VTK files are written in 'src/movies.c' in the routine 'void WriteVTK(int system)'. The top of this file also defines the colors. This same color definition is also used in the VTK 'main.c'.

7.3 Framework surface

The ball and stick pictures are useful, but still do not provide information about pore shape and connectivity. A more suitable approach is to visualize the energy landscape for a certain probe atom. For energy landscape pictures, we divide the unit cell into e.g. $150 \times 150 \times 150$ voxels (volume-elements). At millions of random positions in the unit cell the free energy of a test-particle (usually a helium or methane unit atom) is calculated and assigned to the appropriate voxel. To visualize this energy landscape the three-dimensional dataset is volume rendered, removing the parts that generate overlap (the structure itself) by making it completely transparent. Low energy values are rendered with medium transparency, allowing the inside of the pores/cages to be viewed as voids. Higher energy values are rendered less and less transparent until the energy approaches a cutoff energy and is regarded as part of the zeolite wall. Also color is assigned according to the energy value (green for the outside view of a cage).

To speed up computation of surface and density pictures it is advisable to use energy-grids. For the upcoming example we need grids for CO_2-atoms and helium:

```
SimulationType                  MakeGrid

Forcefield                      ExampleZeolitesForceField

Framework 0
FrameworkName ERI_mono
UnitCells 3 3 2
ExternalTemperature 300.0

NumberOfGrids 3
GridTypes O_co2 C_co2 He
SpacingVDWGrid 0.1
SpacingCoulombGrid 0.1
```

We need $3 \times 3 \times 2$ unit cells to obey the minimum-image convention.

Next we are going to generate the VTK 'FrameworkatomsSurface.vtk' that contains data on the energy-grid for a chosen probe atom. Here, we use helium. An example input to generate the surface-grid is listed here ('Example/Visualization/ERI/SurfaceRASPA')

```
SimulationType                  Visualization
NumberOfCycles                  10000000000
PrintEvery                      100000

Forcefield                      ExampleZeolitesForceField
ChargeMethod                    None

Framework 0
FrameworkName ERI_mono
UnitCells 3 3 2
ExternalTemperature 300.0

NumberOfGrids 1
GridTypes He
SpacingVDWGrid 0.1
SpacingCoulombGrid 0.1
UseTabularGrid yes

component 0 MoleculeName                    helium
            MoleculeDefinition              ExampleDefinitions
            IdealGasRosenbluthWeight        1.0
            BlockPockets                    no
            BlockPocketsFileName            ERI_mono
            CreateNumberOfMolecules         0
```

Even though the grid is generated for a single unit, in general one still needs $3 \times 3 \times 2$ unit cells because the charge interaction is dependent on the amount of chosen unit cells. The grid file 'FrameworkSurface.vtk' is located in 'VTK/System[int]'.

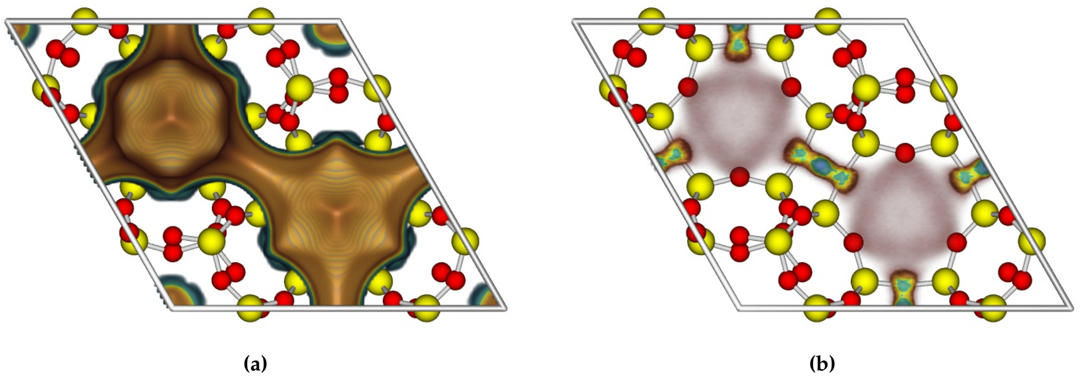

(a) (b)

Figure 15: *Picture of ERI: (a) surface picture, (b) density picture (1 CO_2 at 300K).*

To visualize the pore-shape, copy the 'FrameworkSurface.vtk' to 'Examples/Visualization/ERI/VTK'. Do not copy the other VTK files because they are generated for a $3 \times 3 \times 2$ grid, and we need the framework-atoms and frame- VTK files for a $1 \times 1 \times 1$ structure. Running the VTK-program now shows the surface inside the structure, as shown in Figure 15(a). If we compare Figure 15(a) to Figure 14 we see that we have visualized the pore structure itself. Also note, that small "pockets" have shown up that are not a part of the main pore system. These pockets should be blocked (See next section).

The ERI-case by default offers a nice view inside the cage. This is not always the case. Using

```
ShiftUnitCells 0.25 0 0
```

one can change the LTA case to a outside-cage-view to an inside-cage-view. (TODO, check whether grids takes this into account.)

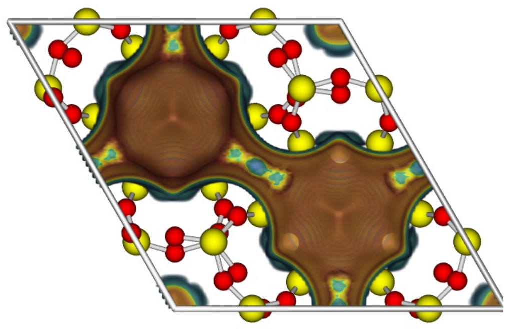
The 'FrameworkSurface.vtk' is a structured points VTK-file. It is rectangular grid of, in this case, $150 \times 150 \times 150$ points (a total of 3375000 points). All these values are listed sequentially, but one can convert between 1D and 3D by using

$$\text{index} = x + y * \text{SIZEY} + z * \text{SIZEX} * \text{SIZEY} \tag{7.1}$$

Note that the proper aspect ratios can be used. The VTK file looks like

```
# vtk DataFile Version 1.0
Free energy zeolite: ERI_mono (300.000000 K)
ASCII
DATASET STRUCTURED_POINTS
DIMENSIONS 150 150 150
ASPECT_RATIO 1.000000 0.577350 0.756091
ORIGIN 0.0 0.0 0.0

POINT_DATA 3375000
SCALARS scalars unsigned_short
LOOKUP_TABLE default
0
0
....
....
```

The stored values are 'unsigned short', so between 0 and 65536 (2^{16}). The value are always clipped to this region using the minimum and maximum values of the simulation data.


(a)

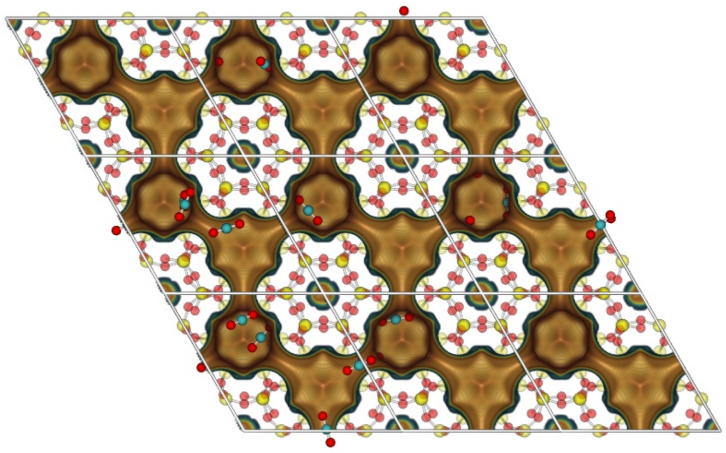

(b)

Figure 16

7.4 Density plots

During a Monte Carlo simulation a 3-dimensional histogram of the positions of all atoms of the molecules is collected (per component). The unit cell is divided into 150x150x150 ”voxels”. During the simulation the molecules move around in the box, and every cycle data is collected for the histogram. First a position is mapped back from the full simulation box (3x3x2 unit cells) to the main unit cell, and for every atom the voxel corresponding to the mapped position is incremented. At certain intervals the histogram is written to file so that it can be visualized using VTK. The data is always normalized using the highest occurring voxel value. However, the overall brightness is still influenced by the loading of the specific adsorbate in the mixture.

In VTK the data is ”volume rendered”, more dense regions are less transparent, less dense regions are more transparent. In addition the color changes, less dense regions are grey, more dense are orange, then yellow, and the highest is rendered light blue. The original framework is placed in the picture as a ball-and-stick model, and every position can be related to the framework. We can therefore e.g. decipher a molecular picture of why selectivity occurs.

An example input for RASPA is

```
SimulationType                  MC
NumberOfCycles                  100000000
NumberOfInitializationCycles    100
PrintEvery                      100
PrintPropertiesEvery            10000

Forcefield                      ExampleZeolitesForceField

Framework 0
FrameworkName ERI_mono
UnitCells 3 3 2
ExternalTemperature 300.0
ComputeDensityProfile3DVTKGrid   yes
WriteDensityProfile3DVTKGridEvery 10000
DensityProfile3DVTKGridPoints 150 150 150
```

```
NumberOfGrids 2
GridTypes C_co2 O_co2
SpacingVDWGrid 0.1
SpacingCoulombGrid 0.1
UseTabularGrid yes

component 0 MoleculeName                     CO2
            MoleculeDefinition               ExampleDefinitions
            IdealGasRosenbluthWeight         1.0
            TranslationProbability           1.0
            RegrowProbability                1.0
            SwapProbability                  0.0
            CreateNumberOfMolecules          1
```

Copy the 'VTK/System[int]/DensityProfile_methane.vtk' to 'Examples/Visualization/ERI/VTK' as 'Density.vtk', rename the surface VTK-file, and run the vtk-code. It will now produce a picture like Figure 15(b). If you did not rename the file (or rename it again to 'FrameworkatomsSurface.vtk'), a picture with the frameworks atoms, the pore surface and the density of CO_2 is produced. It is now easy to show that CO_2 preferentially adsorbs in the 8-ring windows separating the erionite cages (in contrast to an alkane which prefers the cages).

Of course, one is not restricted to a unit cell and it is possible to make pictures of bigger volumes. The first way is to use $3 \times 3 \times 2$ unit cells, and use the file 'Movies/System[int]/Framework_initial.cssr'. Copy this file as 'structure_name_3x3x2.cssr' and from then on use $1 \times 1 \times 1$ using this new enlarged unit cell. The second method is to use $3 \times 3 \times 2$ unit cell but copy the surface and density in the x, y, z directions in the picture. You have to edit the 'main.c' file of the VTK directory and recompile. The relevant settings are:

```
// the resolution of spheres and tubes, the higher the more smooth
// use 10, but 50 for the final picture
const int Resolution=10;

// anti-aliasing, use 1, but 16 for final picture
const int AA=1;

// control the transparancy of framework, adsorbates, and cations
const double FrameworkOpacity=1.0;
const double AdsorbateOpacity=1.0;
const double CationOpacity=1.0;

// zoom in or out by increasing/decreasing the zoom-factor
const double ZoomFactor=2.0;

// scale the size of the atoms and bonds
const double ScaleFactor=1.0;

// control the view-point of the oject (input in degrees)
const double Azimuth=0.0;
const double Elevation=0.0;
const double Roll=0.0;

// the size of the image in pixels
const int ImageSizeX=800;
const int ImageSizeY=500;
```

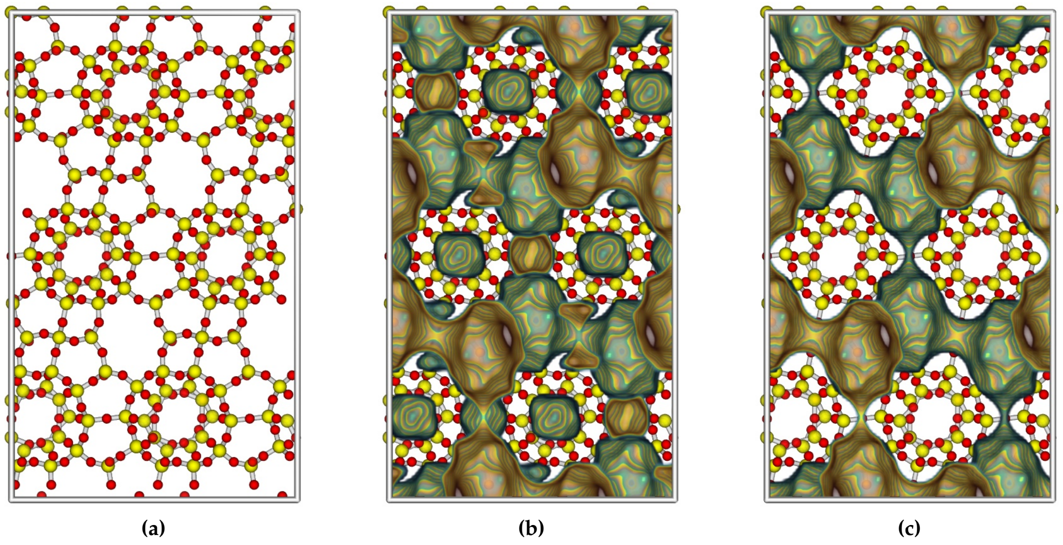


Figure 17: *Blocking pockets in DDR. The DDR structure is converted to a orthorhombic unit cell of $a = 24.006$, $b = 13.86$, and $c = 40.892$ Å. In (a) we show the ball-and-stick structure, in (b) the structure and the pore surface probed with helium, and (c) the structure with proper blocking of the small disconnected pockets.*

```
// the number of duplicates in x,y,z (same as the number of unit cells)
const int NrDuplicatesX=3;
const int NrDuplicatesY=3;
const int NrDuplicatesZ=2;

// the lengths of the edge-vectors
const double A=13.27;
const double B=13.27;
const double C=15.05;

// the angles of the unit cell
const double AlphaAngle=90*M_PI/180.0;
const double BetaAngle=90*M_PI/180.0;
const double GammaAngle=120*M_PI/180.0;
```

This can then be used to make a 'snapshot' of molecules. For the $3 \times 3 \times 2$ structure we need the file 'VTK/System[int]/AdsorbateAtoms.vtk' from a simulation. This file is generated at the start of a simulation. After a sufficiently long run to equilibrate the molecules, one could copy the 'Restart' to 'RestartInitial', put the amount of created molecules at zero and restart from the restart-files using zero cycles to generate the new 'AdsorbateAtoms.vtk' file. The picture of a snapshot of 64 CO_2 in the $3 \times 3 \times 2$ ERI-structure is shown in Figure 16(b). Note the many CO_2 molecules that occupy the barrier. Conclusions are hard to draw based on snapshots. The 'density'-plots give average information and therefore the same for each unit cell (because each unit cell is the same [using a rigid structure]). The density plots are based on atoms, and one can clearly see the orientation of CO_2 on the barrier. The 3 'blobs' corresponds to the oxygen, carbon, and oxygen of CO_2.

7.5 Determining blocking pockets

Some structures have inaccessible parts, i.e. areas that are not reachable from the main pore system. Examples are the sodalite cages in FAU- and LTA-type zeolites. The surface pictures allow us to visualize these pockets, locate the position, and construct a 'blocking-file'.

The unit cell of the DDR structure has edge lengths of a=24.006 Angstrom, b=13.86 Angstrom, 40.892 Angstrom with cell angles of 90 degrees. The atomic structure is shown in Figure 17(a). It is difficult to envision the details of the pore structure from this picture. One can obtain more insight from energy-landscapes. In Figure 17(b) we show the same structure with the energy landscape a helium atom would feel. In practice, the simulation cell is divided into 150x150x150 bins and during a Monte-Carlo simulation one keeps track of the average energy a molecule feels inside that bin. Here we volume-rendered the resulting energy grid making very high energies transparent, i.e. the part that overlaps with the framework, as well as very favorable energies, i.e. the positions inside the cage. The resulting surface layer can be viewed as the "wall" of the pores. Alternatively, one can make a isocontour (a surface representing a constant, high value of the energy). In Figure 17(b) the main pore structure is apparent, but also some disconnect pockets show up. It is very important to artificially block these pockets for Monte-Carlo simulations. Also, in Molecular Dynamic simulations, initial positions should be chosen in the main channel system. The blocking procedure can be a simple distance-check from the center of the small pockets and a rejection of all Monte-Carlo trial moves that would place a molecule inside a certain radius. This radius should not be chosen to small or too big, because otherwise one would block not enough, or block parts of the main channel system. In Figure 17(c) we show the structure with the appropriate blocking centers and radii; all small pockets have disappeared but the main channel system is unchanged.

The blocking procedure is dependent on the type of probe atom. Helium is a good procedure to find small pockets and therefore to obtain the proper unit cell pore volume. This accessible pore volume is in simulation usually obtained via a helium-probe procedure. Helium can also be used to find pockets that could be occupied by other small molecule like CO2, N2, H2, methane, etc. The adsorption results can be dramatically different with or without blocking. Whether the selectivity of mixtures changes to higher or lower depends on the match of the molecule with the small pockets. The small pockets are very favorable for the small molecules because they tend to have a very surface high curvature, i.e.. a very favorable interaction energy).

7.6 Making movies

7.6.1 Using VMD

7.6.2 Combining pictures into a movie

Using "ffmpeg", from png-files to a mov-file with h264-encoding

```
ffmpeg -i %03d.png -s:v 1280x720  -acodec aac -ac 2 -strict experimental -ab 160k
-vcodec libx264 -preset slow -profile:v baseline -level 30 -maxrate 10000000
-bufsize 10000000 -b 1200k -f mp4 -threads 0  -crf 23 -pix_fmt yuv420p -r 30 Movie.mov
```

or using "mencoder" with settings

```
export opt="vbitrate=1280000:mbd=2:keyint=132:vqblur=1.0:cmp=2:subcmp=2:dia=2:mv0:last_pred=3"
mencoder -ovc lavc -lavcopts vcodec=msmpeg4v2:vpass=1:$opt -mf type=jpg:fps=25 -nosound -o /dev/null mf://\*.jpg
mencoder -ovc lavc -lavcopts vcodec=msmpeg4v2:vpass=2:$opt -mf type=jpg:fps=25 -nosound -o output.avi mf://\*.jpg
```

Part III

Tutorial

8 Tutorial

8.1 Adsorption isotherm of N_2 in a metal-organic framework (MOF), Henry coefficients, enthalpy of adsorption

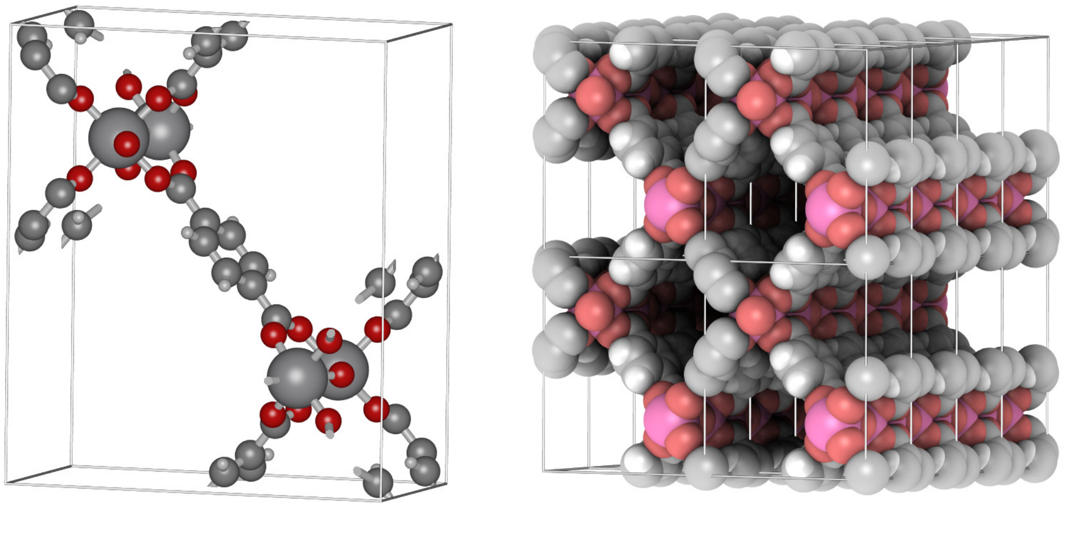

(a) (b)

Figure 18: *(left) The MIL-47 unit cell,* $6.8179 \times 16.1430 \times 13.9390$ *Å,* $\alpha = \beta = \gamma = 90°$, *(right) the* $4 \times 2 \times 2$ *supercell.*

We are going to look at adsorption properties of methane in MIL-47 [1]. The MIL-47 structure is shown in Fig. 18. The first step is to select the size of the system. We are going to choose a VDW cutoff of 12 Å. This implies that all *perpendicular* lengths of the unit cell should be larger than twice the cutoff, i.e. 24 Å. So, as a *minimum*, we should use at least a $4 \times 2 \times 2$ unit cells. This requirement follows from the ”minimum-image convention” where the interactions are computed with only the closest periodic image.

type	V^{0+}	V^{2+}	V^{4+}	DFT
V	2.67677	1.45833	1.83592	1.68
O1	-0.986909	-0.527963	-0.661157	-0.6
O2	-0.712381	-0.439958	-0.516643	-0.52
O3	-0.693279	-0.427135	-0.501819	-0.52
C1	0.00680379	-0.0146643	-0.0110838	-0.15
H1	0.0434488	0.0574796	0.055858	0.12
C2	0.0116383	-0.0118276	-0.00782077	-0.15
H2	0.0444475	0.0586772	0.0570134	0.12
C3	-0.150672	-0.0720208	-0.083311	0.0
C4	0.605064	0.384265	0.420426	0.56

Table 8.1: *Obtaining charges for MIL-47: charge equilibration vs. DFT-derived charges.*

The positions of the atoms are usually known from experiments, or alternatively can be obtained from QM optimizations. Using the atomic information of the framework only, we can compute the frameworks-mass as 14787.6 g mol^{-1}, and we can already compute a few interesting properties. The first is a measurement of how void the structure is.

Exercise 1: go to the sub-directory '`1-Helium-void-fraction`'. Compute the helium void-fraction (look for '`Rosenbluth factor new: 0.608 [-]`' in the output file).

About 61% of the structure is empty!

Exercise 2: go to the sub-directory '`1-Helium-void-fraction`'. Add a line '`HeliumVoidFraction 0.61`' below '`Framework 0`', use zero cycles, and check the structural properties (i.e. accesible pore volume and loading conversion factors) of the system.'

We see that the available pore volume is 0.60977519 cm^3 g^{-1}. This density is important to know, because, using the liquid density, it gives a first approximation of the "maximum" number of molecules in the pore.

Two other very useful properties are the accessible surface and pore-size distribution.

Exercise 3: go to the sub-directory '`3-Surface-area`'. Run and compute the surface area.

The nitrogen surface area of MIL-47 is about 1650 m^2 g^{-1}. This is much larger than most zeolites, but smaller than most large pore MOFs which can go up to an incredible 5000-7000 m^2 g^{-1}.

Exercise 4: go to the sub-directory '`4-Pore-size-distribution`'. Run and compute the pore-size-distribution. Plot the output-file in '`PoreSizeDistributionHistogram`' in gnuplot using column 1 vs. 3 (`plot 'PoreSizeDistributionHistogram' us 1:3 with lines`) to see what the typical pore sizes are.

In general, the individual framework atoms are charged, but the overall framework should be charge neutral (or compensated by cations when the framework itself has a net-charge). For the charges there is ambiguity, since charge is not an ab-initio observable. That means that different methods to obtain charges give different answers. For adsorption however, we are not really interested in the charges themselves but rather of their influence on the electrostatic potential energy field *inside* the cavities. The CHELPG-type methods aim to do just that: they optimize the classical point charges on the framework work atoms to match the PES (potential energy surface) computed with ab-initio methods. For crystals, the REPEAT method is a very nice variant taking the periodicity into account [2]. However, such computations can take several hours (or even days). A fast alternative is "charge-equilibration" [3, 4, 5].

Exercise 5: go to the sub-directory '5-ChargeEquilibration'. Compute the charges using charge-equilibration for various oxidation states of vanadium (edit the 'pseudo_atoms.def'-file). The output-charges are written to 'Movies/System_0/Framework_0_final_1_1_1.cif'.

In Table 8.1 we summarize the results: charge-equilibration can give a good approximation in a matter of seconds (or minutes) provided the charge-expansion is performed around the appropriate oxidation state [5].

Next we are going to choose N_2 as the adsorbate molecule. Since this is a small molecule, it is probably fine to use the small $4 \times 2 \times 2$ supercell. For much larger molecules finite-size effects occur and a larger system should be used. For example, a long chain-molecule could even interacts with *itself* if the system was small, which obviously leads to erroneous results. In order to compute an adsorption isotherm we need the framework positions and charges, a force field for the adsorbate and interactions with the framework. Here we will use a generic force field based on DREIDING and UFF [6, 7].

An adsorption isotherm describes the adsorption at a fixed (chosen) temperature as a function of pressure. The first question is to examine the appropriate pressure range for adsorption.

Exercise 6: go to the sub-directory '6-Adsorption'. Use a few thousand cycles and determine at what pressure adsorption starts to occur (pressures units are Pascal). Do this for 600K (and if time permits 600K).

Exercise 7: go to the sub-directory '6-Adsorption'. Use 10000 cycles initialization, a few ten-thousand cycles for production and compute 5 to 10 points from the starting pressure to 1000 bar. Put the data in a file and plot the loading vs. pressure in normal scale and in log-scale.

```
SimulationType                MonteCarlo
NumberOfCycles                10000
NumberOfInitializationCycles  10000
PrintEvery                    100
RestartFile                   no

Forcefield                    ExampleMOFsForceField
UseChargesFromCIFFile         yes

Framework 0
FrameworkName MIL-47
UnitCells 4 2 2
HeliumVoidFraction 0.61
ExternalTemperature 300.0
ExternalPressure 100000.0

Component 0 MoleculeName             N2
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   0.5
            RotationProbability      0.5
            ReinsertionProbability   0.5
            SwapProbability          1.0
            CreateNumberOfMolecules  0
```

By examing the isotherm, the slope of the curve can be related to the Henry's coefficient. This property can also be conveniently computed by Widom's insertion using a single probe adsorbate and is directly related to the excess chemical potential and the free energy [8]. The Henry coefficient can be computed by

$$K_H = \frac{1}{RT\rho_f}\frac{\langle W\rangle}{\langle W^{IG}\rangle} \tag{8.1}$$

where ρ_f is the density of the framework, and $\langle W\rangle$ is the Rosenbluth weight. This weight is in general the Rosenbluth weight when configurational bias is used, and reduces to the Boltzmann factor $\langle\exp\left(-\beta\Delta U\right)\rangle$ without biasing. The ideal Rosenbluth weight $\left\langle W^{IG}\right\rangle$ is the value for a single molecule in the ideal gas phase and serves as the reference state.

Exercise 8: go to the sub-directory '`7-Henry coefficient`'. Compute the Henry coefficient and compare it to the value you obtain from the isotherm at low loading.

Similarly, the limit of the enthalpy of adsorption can be computed from the limit of using a single adsorbate in the NVT-ensemble The affinity of a molecule with the framework can be expressed as the binding energy, or more general, as the enthalpy of adsorption at infinite dilution ΔH:

$$\Delta H = \Delta U - RT = \langle U_{hg}\rangle - \langle U_h\rangle - \langle U_g\rangle - RT \tag{8.2}$$

where ΔU is the internal energy, and $\langle U_{hg}\rangle$, $\langle U_h\rangle$, and $\langle U_g\rangle$ are the average energy of the guest molecule inside the host-framework, the average energy of the host-framework, and the average energy of the guest-molecule, respectively. In simulations a common approximation is to assume the framework is rigid, and in this case the enthalpy of adsorption at infinite dilution can be understood to be the difference in internal energy of a single molecule outside and inside the confinement of the host framework. In the limit of zero temperature, the enthalpy of adsorption becomes the binding energy. Note: for rigid molecules $\langle U_g\rangle = 0$.

Exercise 9: go to the sub-directory '`8-Heat of adsorption`'. Compute the limit of the enthalphy of adsorption at zero loading. Compare this value to the values from the fluctuation formula computed during the isotherm.

Infinite dilution enthalpy of adsorption ΔH is related to the Henry's coefficient K_H as

$$\Delta H = -\frac{\partial \ln K_H}{\partial\beta} \tag{8.3}$$

where $\beta = 1/\left(k_BT\right)$ is the inverse temperature, and k_B the Boltzmann's constant. The Henry's coefficient is the slope of the isotherm at zero pressure/loading.

Exercise 10 (optional): check relation Eq. 8.3 with Henry's coefficient simulations as a function of temperature.

8.2 NPT density of super-critical CO_2, RDF, diffusion

The density of CO2 at 400 and 100 bar is about 161.53 kg m^{-3}, at 500 bar 745.45 kg m^{-3} and at 1000 bar the density is 932.81 kg m^{-3} (NIST chemical database). In this example the density is computed using Monte Carlo in the NPT-ensemble. Given the pressure P, the temperature T, and the amount of molecules N, the density is computed.

```
SimulationType                 MonteCarlo
NumberOfCycles                 50000
NumberOfInitializationCycles   10000
PrintEvery                     1000
```

```
RestartFile                  no

Forcefield                   ExampleZeolitesForceField

Box 0
BoxLengths 30 30 30
ExternalTemperature 400.0
ExternalPressure 10e5
ComputeMolecularPressure yes

VolumeChangeProbability      0.05

Component 0 MoleculeName             CO2
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   0.5
            RotationProbability      0.5
            RegrowProbability        0.5
            CreateNumberOfMolecules  256
```

Exercise 1: go to the sub-directory 'FluidCO2/MC_NPT'. Verify the three densities listed from NIST experimental data.

Next we are going to compute two important fluid properties that give inside in the structure of the fluid: the radial distribution function (RDF) and the self-diffusion.

Exercise 2: go to the sub-directory 'FluidCO2/MC_NPT'. Set the volume to the average of the previous step and switch off the volume move, e.g. remove 'VolumeChangeProbability 0.05'. Also replace 'ComputeMolecularPressure yes' by
ComputeRDF yes
WriteRDFEvery 1000
Plot and analyze the output rdf's that can be found in the directory 'RadialDistributionFunctions'. Analyze what the peaks mean.

Finally, we are going to compute a dynamic property. Therefore, we change 'SimulationType MonteCarlo' to 'SimulationType MolecularDynamics' and we are going to use Molecular Dynamics.

```
SimulationType               MolecularDynamics
NumberOfCycles               1000000
NumberOfInitializationCycles 1000
NumberOfEquilibrationCycles  10000
PrintEvery                   10000
PrintPropertiesEvery         10000

Ensemble                     NVT
TimeStep                     0.0005

Forcefield                   ExampleZeolitesForceField

Box 0
BoxLengths 25 25 25
ExternalTemperature 400.0
```

```
ComputeMSD yes
PrintMSDEvery 5000

Component 0 MoleculeName             CO2
            MoleculeDefinition       ExampleDefinitions
            TranslationProbability   1.0
            ReinsertionProbability   1.0
            CreateNumberOfMolecules  100
```

Exercise 3: Compute the diffusion via the mean-square displacement. Using gnuplot, plot the file 'MSDOrderN/System_0/msd_self_methane_0.dat'. Use the slope to extract the diffusion coefficient.

8.3 Reaction-ensemble of ammonia

We are going to study the ammonia synthesis reaction

$$N_2 + 3H_2 \leftrightarrow 2NH_3 \tag{8.4}$$

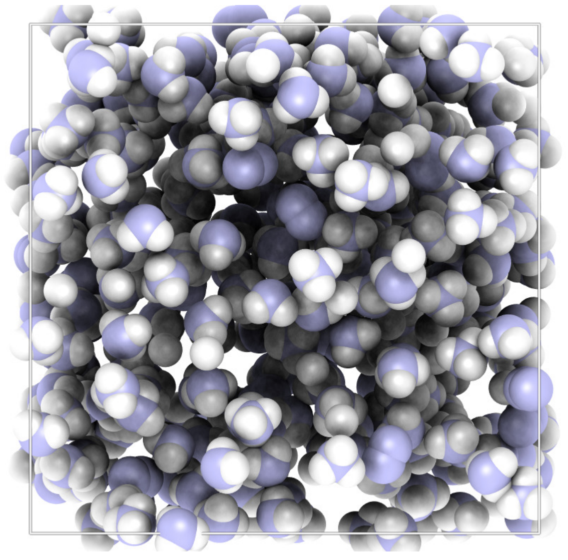
Ammonia ranks second among synthesis chemicals in amount produced, and there has been a great deal of experimental and theoretical research into the ammonia synthesis reaction over the past 100 years. Thus there is an abundance of experimental reference data on this reaction, allowing an accurate check of simulation models [9].

One of the most commonly used approaches in molecular simulation is to simulate reaction equilibria in the reaction ensemble (RxMC). In this approach, the chemical reaction is carried out by a Monte Carlo (MC) trial move. Beside thermalization (translation, rotation, etc), trial moves are carried out in which reactants are removed and reaction products are inserted in the system, in such a way that an equilibrium distribution of reactants and reaction products is obtained. The mechanism and the transition state of the reaction are not considered as this approach is purely thermodynamic. As a result, the efficiency of this simulation technique is not affected by the height of the activation energy barrier of the reaction as reaction kinetics are not considered. The RxMC method requires the ideal gas partition functions of all reactant and reaction product molecules, a list of all possible chemical reactions in the system, and an appropriate force field accurately describing interactions between molecules.

Figure 19 shows a snapshot of the N_2-$3H_3$-$2NH_3$ system. To efficiently perform the reaction we use the reaction-ensemble using continuous fractional component MC. The reaction is performed along a λ-parameter from 0 to 1, where 0 denotes the full N_2-$3H_3$ reactant state for the fractional components and 1 the full product state $2NH_3$. Using fractional molecules for each component the reaction can be performed gradually. In addition to the usual thermalization moves we have a λ-move that attempts to change λ with three possible outcomes:

1. λ remains between 0 and 1.
2. λ goes beyond 1.
   We have formed real $2NH_3$ molecules and choose new fractional molecule (randomly) with a value $\lambda - 1$.
3. λ goes below 0.
   We have formed real N_2-$3H_3$ molecules and choose new fractional molecule (randomly) with a value $\lambda + 1$.

The λ-moves are switched on by the 'ProbabilityCFCRXMCLambdaChangeMove' input-parameter. We also perform volume moves to impose the pressure using 'VolumeChangeProbability' option. The example input below defines the box, the 3 components, and the reaction using


(a)

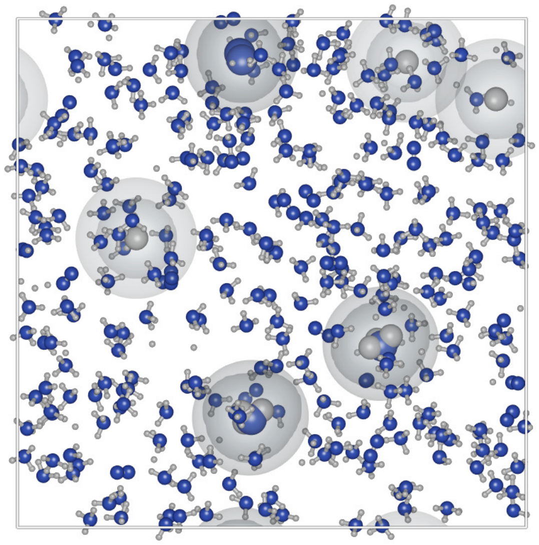

(b)

Figure 19: *(left) the N_2-$3H_3$-$2NH_3$ system, (right) the fractional molecules involved in the reaction.*

```
 Reaction 1 3 0 0 0 2
```

which list the stoichiometry of the reactants and the product. So, 1 of component 0 and 3 of component 1 forms 2 molecules of component 2.

```
SimulationType                 MC
NumberOfCycles                 15000
NumberOfInitializationCycles   10000
NumberOfEquilibrationCycles    20000
RestartFile                    no
PrintEvery                     500

ChargeMethod                   Ewald
Forcefield                     Local
CutOffVDW                      9.0
CutOffCoulomb                  9.0
EwaldPrecision                 1e-5


Box 0
BoxLengths  38  38   38
ExternalTemperature 573.0
ExternalPressure 3e7


Reaction 1 3 0 0 0 2

ProbabilityCFCRXMCLambdaChangeMove 1.0
VolumeChangeProbability            0.1
```

```
Component 0 MoleculeName             N2
            MoleculeDefinition       Local
            LnPartitionFunction      208.188
            TranslationProbability   40.0
            RotationProbability      53.9
            ReinsertionProbability   5.0
            ExtraFrameworkMolecule   no
            CreateNumberOfMolecules  13

Component 1 MoleculeName             H2
            MoleculeDefinition       Local
            LnPartitionFunction      93.9084
            TranslationProbability   40.0
            RotationProbability      53.9
            ReinsertionProbability   5.0
            ExtraFrameworkMolecule   no
            CreateNumberOfMolecules  39

Component 2 MoleculeName             NH3
            MoleculeDefinition       Local
            LnPartitionFunction      253.69
            TranslationProbability   40.0
            RotationProbability      53.9
            ReinsertionProbability   5.0
            ExtraFrameworkMolecule   no
            CreateNumberOfMolecules  134
```

P [bar]	573K	673K	773K	873K
100	0.53	0.25	0.10	0.05
200	0.67	0.39	0.18	0.09
300	0.75	0.48	0.25	0.13
400	0.80	0.55	0.32	0.16
500	0.84	0.61	0.37	0.20
600	0.87	0.66	0.42	0.24
700	0.89	0.70	0.47	0.27
800	0.91	0.74	0.51	0.31
900	0.93	0.77	0.55	0.34
1000	0.94	0.80	0.58	0.37

(a) *Experiments*

P [bar]	573K	673K	773K	873K
100	0.56	0.27	0.12	0.05
200	0.69	0.41	0.20	0.09
300	0.78	0.49	0.26	0.14
400	0.82	0.57	0.32	0.17
500	0.86	0.62	0.37	0.20
600	0.88	0.66	0.42	0.24
700	0.90	0.69	0.45	0.27
800	0.91	0.73	0.50	0.30
900	0.93	0.77	0.53	0.33
1000	0.94	0.79	0.56	0.35

(b) *Simulations*

Figure 20: *Mol-fractions of the NH_3 in the ammonio bulk phase reaction of N_2 and H_2 computed from simulation compared to experiments over a wide range of temperatures and pressures.*

T [K]	N_2	H_2	NH_3	Eq. constant K_p
573	2.60E+90	6.08E+40	1.50E+110	0.006327104
673	6.89E+77	1.28E+35	5.42E+94	0.000244159
773	3.44E+68	8.28E+30	2.12E+83	2.06653E-05
873	2.42E+61	5.08E+27	3.65E+74	2.97405E-06

Table 8.2: *Input partition function in units of Å^3 and the equilibrium constant K_p. The partition functions are computed based on the vibrational and rotational constants reported in the book by McQuarrie [10].*

Exercise 1: go to the sub-directory '`Tutorial/ReactionEnsembleAmmonia`'. Using the input-parameters of Table 8.2 reproduce the simulation results.

Bibliography

[1] Barthelet, K.; Marrot, J.; Riou, D.; Ferey, G. *Angew. Chem. Int. Ed.* **2002**, *41*, 281.

[2] Campana, C.; Mussard, B.; Woo, T.K. *J. Chem. Theory Comput.* **2009**, *5*, 2866-2878.

[3] Rappe, A.K.; Goddard, W.A. *J. Phys. Chem.* **1991**, *95*, 3358-3363.

[4] Wilmer, C.E.; Snurr, R.Q. *Chem. Eng. J.* **2011**, *171*, 775-781.

[5] Wilmer, C.E.; Kim, K.C.; Snurr, R.Q. *J. Phys. Chem. Lett.* **2012**, *3*, 2506-2511.

[6] Mayo, S.L.; Olafson, B.D.; Goddard, W.A. *J. Phys. Chem.* **1990**, *94*, 8897-8909.

[7] Rappé, A.K.; Casewit, C.J.; Colwell, K.S.; Goddard, W.A.; Skiff, W.M.J. *J. Am. Chem. Soc.* **1992**, *114*, 10024-10035.

[8] Frenkel, D.; Smit, B., *Understanding Molecular Simulation*, 2nd ed. Academic Press; London, UK, 2002.

[9] Turner, C.H.; Johnson, J.K.; Gubbins, K.E. *J. Chem. Phys.* **2001**, *114*, 1851-1859.

[10] McQuarrie, D.A., *Statistical Mechanics* University Science Books; Mill Valley, California, 2000.

Appendix

Random numbers

32-bits version

```
A C-program for MT19937, with initialization improved 2002/1/26.
Coded by Takuji Nishimura and Makoto Matsumoto.

Before using, initialize the state by using init_genrand(seed)
```

64-bits version

```
A C-program for MT19937-64 (2004/9/29 version).
Coded by Takuji Nishimura and Makoto Matsumoto.
```

This is a 64-bit version of Mersenne Twister pseudorandom number
generator.

Before using, initialize the state by using init_genrand64(seed)

References:
T. Nishimura, ‘‘Tables of 64-bit Mersenne Twisters’’
  ACM Transactions on Modeling and
  Computer Simulation 10. (2000) 348--357.
M. Matsumoto and T. Nishimura,
  ‘‘Mersenne Twister: a 623-dimensionally equidistributed
    uniform pseudorandom number generator’’
  ACM Transactions on Modeling and
  Computer Simulation 8. (Jan. 1998) 3--30.

Acknowledgements

We would like to thank the following people for their help and input to improve the program and for the very helpful discussions about the algorithms: Sayee Prasaad Balaji, Youn-Sang Bae, Xiaoying Bao, Rocío Bueno Pérez, Tim Becker, Nicholas C. Burtch, Tom Caremans, Ana Martín Calvo, Yamil Colon, Juan Manuel Castillo Sanchez, Allison Dickey, Tina Düren, Titus van Erp, Denise Ford, Houston Frost, Rachel Getman, Pritha Ghosh, Seyed H. Jamali, Jurn Heinen, Remco Hens, Elena García Pérez, Gloria Oxford, Ali Poursaeidesfahani, Sudeep Punnathanam, Reza Rahbari, Almudena Garcia Sanchez, Juan Jose Gutierrez Sevillano, John J. Low, Patrick Merkling, Mahinder Ramdin, Patrick Ryan, Lev Sarkisov, Ben Sikora, Ariana Torres Knoop, Krista S. Walton, Chris Wilmer, Ozgur Yazaydin, and Decai Yu.