COTA 4.0: Molecular Software Package for Adsorption and Diffusion in Nanoporous Materials

D. Dubbeldam[1]
Chemical and Biological Engineering Department, Northwestern University,
2145 Sheridan Road, Evanston IL 60208 USA

August 19, 2007

[1]email: d-dubbeldam@northwestern.edu

Contents

Part I

Cota

Chapter 1

Introduction

1.1 Design philosophy

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

- Correctness and accuracy
  For all the techniques and algorithms available in COTA we have implemented the 'best' ones available in literature. For example, COTA 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 splitted up accoording 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.

- 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 vica versa. Extension and modification of the code is relatively straightforward.

1.2 Units and conventions

- The standard units in COTA 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

- 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 COTA

1.3.1 unpacking COTA

The 'tar' command can be used to extract the files from the archive

```
tar -zxvf cota-x.xx.tar.gz
```

To create a compressed archive use

```
tar -zcvf cota-x.xx.tar.gz cota-x.xx/
```

executing from the directory where the source code of COTA is in.

1.3.2 installing COTA

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

export MOLSIM_DIR=${HOME}/Research/simulations/

or

setenv MOLSIM_DIR "${HOME}/Research/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 COTA is kept separate from the installation data. COTA 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 MOLSIM_DIR directory.

Before installing COTA with

```
make install
```

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

1.3.3 compiling COTA

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

1. rm -rf autom4te.cache
2. aclocal
3. automake -i
4. autoconf

4.5. (optional to very slightly speed up code, possibly dangerous though): export CFLAGS="-Wall -O3"

5. ./configure - -prefix=${MOLSIM_DIR}

6. make

where '${MOLSIM_DIR}' is the directory you would like to install COTA, 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=${MOLSIM_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 -O4 -ffast-math"
export CC="gcc"
```

1.3.4 Running COTA

Running COTA is based on two files:

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

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

  This type of file is know as a 'shell script'. COTA needs the variable 'MOLSIM_DIR' to be set in order to look up the molecules, frameworks, etc. The scripts sets the variable and runs COTA. COTA 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
  InitializationCycles          10000
  PrintEvery                    1000

  Box 0
  30 30 30
  BoxAngles 0
  90 90 90

  ExternalTemperature system 0: 300.0

  component 0 methane
    TranslationProbability        1.0
    CreateNumberOfMolecules       100
  ```

This tells COTA 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.

1.4 Output from COTA

COTA 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
- 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 system, one for vapor phase and one for the liquid phase, while n systems are used by the (hyper-) parallel-tempering technique(s).

Chapter 2

Format of the Input Files

2.1 Simulation input

- Asymmetric [yes|no]
  Determines whether to read an asymmetric unit cell or the full unit cell. If an asymmetric unit cell is used the space group needs to specified using 'SpaceGroup [integer]'. Default: 'no'.

- BarostatChainLength [integer]

  The length of the baro stat chain when using the Nose-Hoover method to apply the desired pressure. Default: 5.

- BarrierNormal [real real real]

- BarrierPosition [real real real]

- BiasingMethod [Umbrella|RuizMontero]

- BiasingProfile [string]

- BlockPockets [yes|no]
  Whether to block certain pockets for the current component or not. The block-file is read from 'share/cota/structures/block', and specified in 'structures/mof/block' for MOFs, and 'structures/zeolite/block' for zeolites. Default: no.

- BlockPocketsFilename [string]

- CBMCProbability [real]

- Charge [None|Ewald]

- Component [integer] MoleculeName [string]

- ComputeDensityHistograms [yes|no]
- ComputeMoleculeProperties [yes|no]
- ComputeMSD [yes|no]
- ComputeRDF [yes|no]
- ComputeSurfaceArea [yes|no]
- CreateNumberOfMolecules [integer]
- CutOff [real]
- DeltaT [real]
- EnergyOverlapCriteria [real]
- Ensemble system [integer]: [NVE|NVT|NPT|NPTPR]
- EquilibrationSteps [integer]
- EwaldPrecision [real]
- ExternalSurfaceTension system [integer]: [real real real]
- ExternalTemperature system [integer]: [real]
- ExternalPressure system [integer]: [real]
- ExtraFrameworkMolecule [yes|no]
- FlexibleFramework [yes|no]
- Forcefield [string]
- Framework [string]
- FrameworkChangeMoveProbability [real]

- FrameworkDefinitions [string]
- GridTypes [list of strings]
- HeliumVoidFraction [real]
- Histogram [yes|no]
- HybridMCMDMoveProbability [real]
- IdealGasRosenbluthWeight [real]
- IdentityChangesList [list of integers]
- IdentityChangeProbability [real]
- InitializationSteps [integer]
- IntegrationScheme [verlet]
- InternalFrameworkLennardJonesInteractions [yes|no]
- InputFileType [cssr]
- MaxBarrierDistance [real]
- MaxBarrierTime [real]
- MinimumInnerCycles [integer]
- MinimumRosenbluthFactor [real]
- MoleculeDefinition [TraPPE]
- MolFraction [real]
- Movies [integer]
- NumberOfCycles [integer]
  Keword to set the amount of production cycles in any simulation.

- NumberOfGrids [integer]
- NumberOfIdentityChanges [integer]
- NumberOfSystems [integer]
- NumberOfTrialPositionsTorsion [integer]
- NumberOfVelocities [integer]
- NumberOfYoshidaSuzukiSteps [integer]
- Output [file|screen]
- PositionsFirstBead [integer]
- PrintEvery [integer]
- PrintPropertiesEvery [integer]
- PrintMoleculePropertiesEvery [integer]
- PrintMSDEvery [integer]
- PrintRDFEvery [integer]
- PutMoleculeOnBarrier [yes|no]
- QuenchCoreShellVelocities [yes|no]
- QuenchCoreShellVelocitiesEvery [integer]
- RandomTranslationProbability [real]
- RegrowProbability [real]
- ReinitializeVelocities [yes|no]
- RemoveBondNeighboursFromLongRangeInteraction [yes|no]

- RemoveBendNeighboursFromLongRangeInteraction [yes|no]
- RemoveTorsionNeighboursFromLongRangeInteraction [yes|no]
- RestartFile [yes|no]
- RestartStyle [Cota|Bigmac]
- RotationProbability [real]
- RuizMonteroFactor [real]
- SimulationType [MD|MC]
  Keyword to set the type of the simulation.
- SpaceGroup [integer]
- SpacingCoulombGrid [real]
- SpacingVDWGrid [real]
- StartingBead [integer]
- SurfaceAreaProbeAtom [string]
- SurfaceAreaSamplingPointsPerShere [integer]
- SwapProbability [real]
- TargetAccSmallMCScheme [real]
- TargetAccTranslation [real]
- ThermostatChainLength [integer]
- TimeScaleParameterBarostat [real]
- TimeScaleParameterThermostat [real]
- TranslationDirection [x|y|z|xy|xz|yz|xyz|a|b|c|ab|ac|bc|abc]

- TranslationNormal [real real real]
- TranslationProbability [real]
- TrialMovesPerOpenBead [integer]
- TrialPositions [integer]
- UnitCells [integer integer integer]
- UseTabularGrid [yes|no]
- VolumeChangeProbability [real]
- WidomProbability [real]
- WritedcTSTSnapShotsEvery [integer]
- WritedcTSTSnapShotsToFile [yes|no]
- WriteIRData [yes|no]
- WriteMoviesEvery [integer]

2.2 Force field

2.2.1 Force fields

2.2.2 'pseudo_atoms.def'

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

```
#number of pseudo atoms
5
#name      print   as   chem   mass       charge  B-factor  radius  connectivity
Ow         yes     O    O      15.9994    0.0     1.0       0.5     2
Hw1        yes     H    H      1.00794    0.241   1.0       1.00    1
Hw2        yes     H    H      1.00794    0.241   1.0       1.00    1
Lw1        no      L    -      0.0       -0.241   1.0       1.00    1
Lw2        no      L    -      0.0       -0.241   1.0       1.00    1
```

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 indentify 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.
chem	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.
mass	The mass of the atom in atomic units.
charge	The charge of the atom in atomic units.
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 $0.56+\text{Radius}_i+\text{Radius}_j$.
connectivity	The connectivity of the atoms (not yet used).

2.2.3 'force_field_mixing_rules.def'

```
# general rule for shifted vs truncated
shifted
# general rule tailcorrections
no
# number of defined interactions
9
# type interaction
Mof_Zn          lennard-jones    0.42      2.7
Mof_Oa          lennard-jones    700.0     2.98
Mof_Ob          lennard-jones    70.5      3.11
Mof_Ca          lennard-jones    48.5      3.76
Mof_Cb          lennard-jones    47.86     3.47
Mof_Cc          lennard-jones    47.86     3.47
Mof_H           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 tailcorrections, 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 the all these interactions can be subsequently overwritten using the 'force_field.def' file for specific interactions.

general cutoff rule	'shifted' or 'truncated'	'shifted' shifts the potentials to zero at the cutoff radius, 'truncated' leaves them unchanged.
general tailcorrections rule	'yes' or 'no'	'yes' applies the tailcorrections to all interactions, 'no' ommits the tailcorrections for all interactions
general mixing-rule (only used for Lennard-Jones)	'Jorgensen' or 'Lorentz-Berthelot'	'Jorgensen' $\left\{\epsilon_{ij} = \sqrt{\epsilon_i \epsilon_j}, \sigma_{ij} = \sqrt{\sigma_i \sigma_j}\right\}$ 'Lorentz-Berthelot' $\left\{\epsilon_{ij} = \sqrt{\epsilon_i \epsilon_j}, \sigma_{ij} = \frac{1}{2}(\sigma_i + \sigma_j)\right\}$
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}.

2.2.4 '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 tailcorrections. 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
Mof_Zn      C_co2       lennard-jones  27.34776042 3.420
Mof_Zn      O_co2       lennard-jones  46.77926891 3.545
Mof_Oa      C_co2       lennard-jones  36.07117963 2.915
Mof_Oa      O_co2       lennard-jones  61.70097244 3.04
Mof_Ob      C_co2       lennard-jones  36.07117963 2.915
Mof_Ob      O_co2       lennard-jones  61.70097244 3.04
Mof_Ca      C_co2       lennard-jones  35.94746166 3.135
Mof_Ca      O_co2       lennard-jones  61.48934867 3.26
Mof_Cb      C_co2       lennard-jones  35.94746166 3.135
Mof_Cb      O_co2       lennard-jones  61.48934867 3.26
Mof_Cc      C_co2       lennard-jones  35.94746166 3.135
Mof_Cc      O_co2       lennard-jones  61.48934867 3.26
Mof_H       C_co2       lennard-jones] 14.37184748 2.825
Mof_H       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 given in the output !

2.3 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.

2.3.1 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          1.163
1 C_co2     0.0          0.0          0.0
2 O_co2     0.0          0.0         -1.163
# Chiral centers Bond  Bend  UrayBradley InvBend  Torsion Imp. Torsion Bond/Bond Stretch/Bend Bend/Bend Bend/Torsion IntraVDW IntraCoulomb
              0    2     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 contants: the critical temperature, the critical pressure, and the acentric factor. They are used to 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 regid 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.

2.3.2 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  Bend  UrayBradley InvBend  Torsion Imp. Torsion Bond/Bond Stretch/Bend Bend/Bend Bend/Torsion IntraVDW IntraCoulomb
              0    4     4             0       0       2           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
```

2.3.3 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.

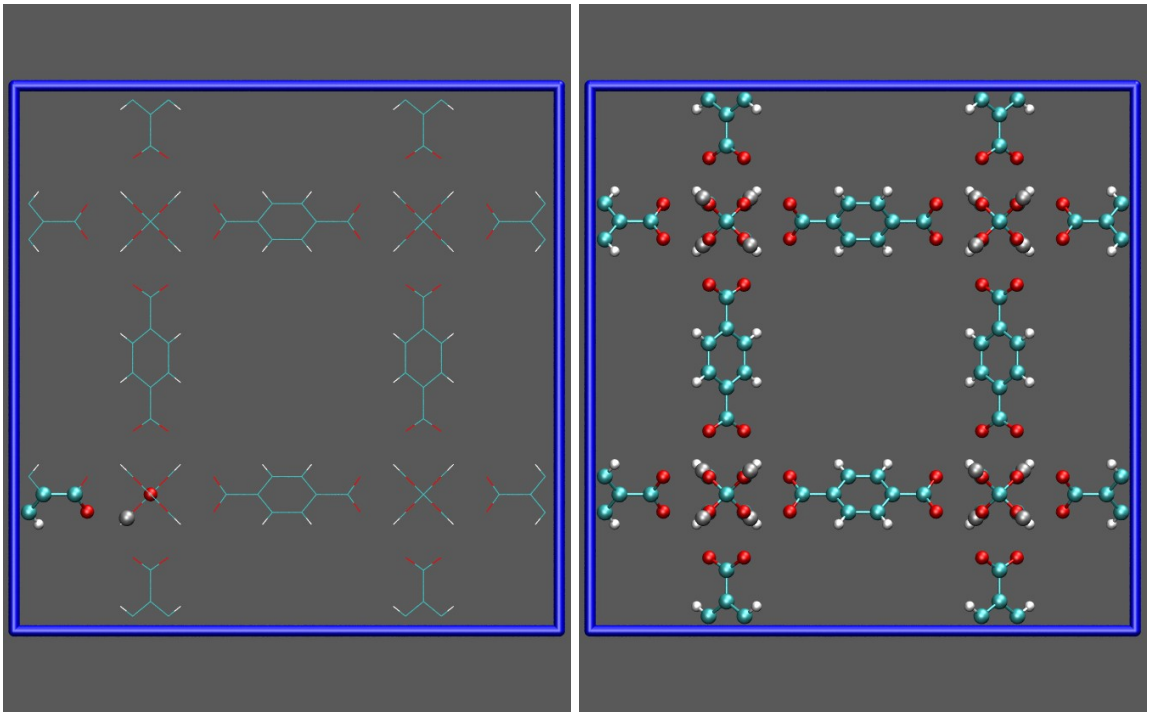
```
# critical constants: Temperature [T], Pressure [Pa], and Acentric factor [-]
0.0
0.0
0.0
#Number Of Atoms
16
# Number of groups
3
# carboxyl-group
3
flexible
0  Mof_Ob
1  Mof_Ca
2  Mof_Ob
# phenyl-ring
10
rigid
3  Mof_Cb 6.458 6.458  11.526
4  Mof_Cc 7.308 7.308  12.221
5  Mof_Cc 5.608 5.608  12.221
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  Bend  UrayBradley InvBend  Torsion Imp. Torsion Bond/Bond Stretch/Bend Bend/Bend Bend/Torsion IntraVDW IntraCoulomb
                0   16    10            0       0       16            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 Framework

2.4.1 Asymetric unit cell

Frameworks are often presented in literature using as much symmetry as possible to reduced the amount of atoms needed to decribe the structure. Usually only the fractional psotions 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 exmaple, 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 $\alpha = \beta = \gamma = 90°$. COTA can read cssr-files, and the structure can be put into a file (see 'IRMOF-1.cssr' in 'structures/mofs/asymmetric'):

```
                                25.832  25.832  25.832
                    90.0    90.0    90.0      SPGR =  1 P 1         OPT = 1
7   0 Created by Cota-4.0
  0 IRMOF-1         : IRMOF-1
1 Mof_Zn  0.2934    0.2066    0.2066     0   0   0   0   0   0   0   0   0.000
2 Mof_Oa  0.2500    0.2500    0.2500     0   0   0   0   0   0   0   0   0.000
3 Mof_Ob  0.2819    0.2181    0.1340     0   0   0   0   0   0   0   0   0.000
4 Mof_Ca  0.2500    0.2500    0.1113     0   0   0   0   0   0   0   0   0.000
```



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

Figure 2.1: Asymmetric unit cells: the left figure shows the zeven crystallographicly different atoms in the IRMOF-1 structure in ball-and-stick format. The 'copies' (crystallographicaly identical atoms) are shown as lines. The right figure shows the full unit cell of IRMOF-1 in ball-and-stick.

```
5 Mof_Cb  0.2500    0.2500    0.0538    0   0   0   0   0   0   0   0   0.000
6 Mof_Cc  0.2829    0.2171    0.0269    0   0   0   0   0   0   0   0   0.000
7 Mof_H   0.3049    0.1951    0.0448    0   0   0   0   0   0   0   0   0.000
```

COTA can then be run using:

```
SimulationType                  MonteCarlo
NumberOfCycles                  0
InitializationCycles            0

Forcefield                      IRMOF-1

NumberOfSystems 1
Framework IRMOF-1
UnitCells 1 1 1
InputFileType cssr
Asymmetric yes
SpaceGroup 225
```

and in the directory 'Movies/System[0]/' several files appear: 'AsymetricUnitCell.pdb', 'framework.pdb', and 'framework.cssr'. The pdb-files can be viewed in the VMD-program or Materials Studio programs for example.

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 asymetric 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 postions are transformed in the final step to Cartesian positions.

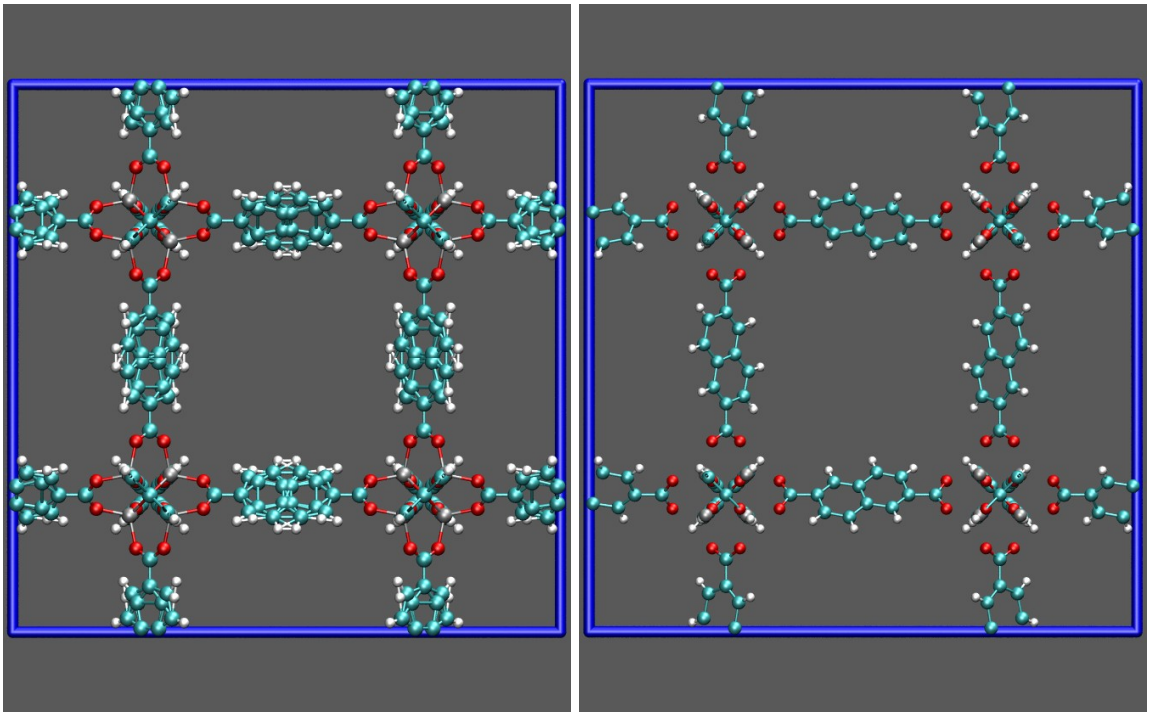
2.4.2 Fractional occupancies

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

```
                                    25.099  25.099  25.099
                    90.0    90.0    90.0      SPGR =  1 P 1          OPT = 1
6   0 Created by Cerius2
  0 LTA          : LTA
1 Si     -0.05381   0.12565   0.03508    0   0   0   0   0   0   0   0   0.000
2 Al     -0.05524   0.03639   0.12418    0   0   0   0   0   0   0   0   0.000
3 O1     -0.1099    0.0003    0.1056     0   0   0   0   0   0   0   0   0.000
4 O2     -0.0011   -0.0028    0.1416     0   0   0   0   0   0   0   0   0.000
5 O3     -0.0346    0.0758    0.0711     0   0   0   0   0   0   0   0   0.000
6 O4     -0.0693    0.0726    0.1800     0   0   0   0   0   0   0   0   0.000
```

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.

More problematic are when several atoms have fractional occupancies lower than unity. Consider IRMOF-8 shown in Fig. 2.2. 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 editten. Program which can do just that are Materials Studio, Gaussview, etc. After the cell has been created and eddited, the file neends to be converted to 'cssr'-format and placed in 'structures/mofs/cssr'. Structures with disorder needs to be created at unit cell level.



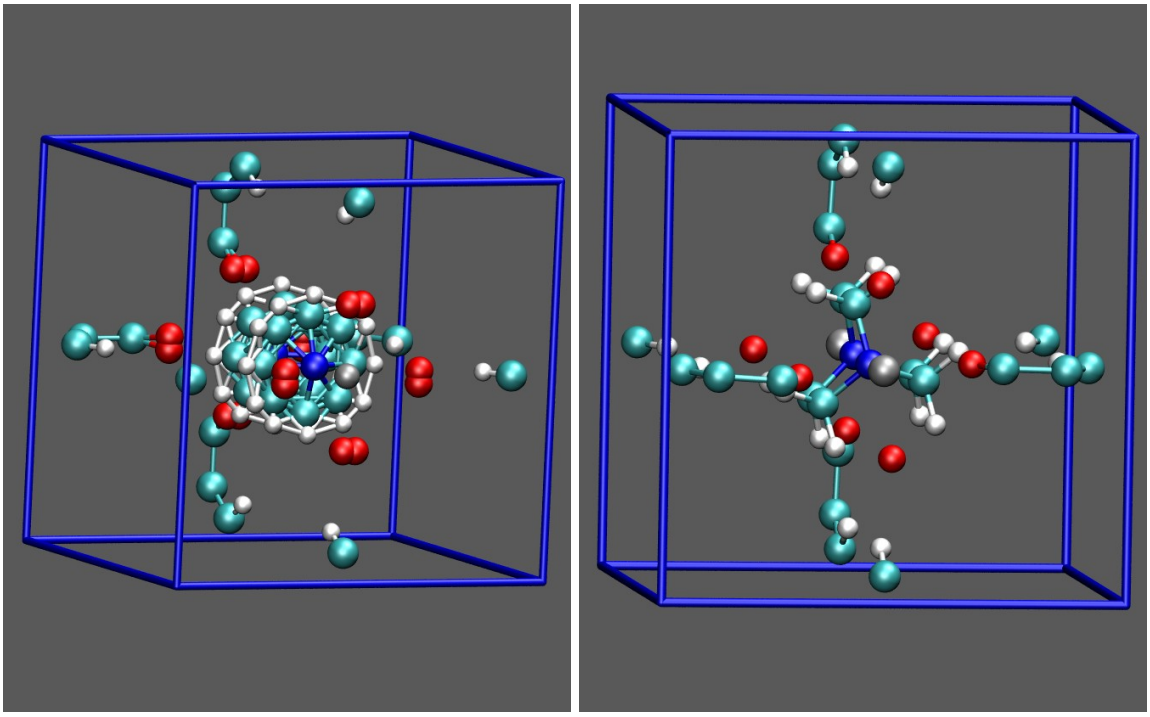
(a) The IRMOF-8 structure as directly computed from the asymmetric positions and the space group.

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

Figure 2.2: 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.

Even more difficult is MOF-**1**. Here the cif-file als contains several possibilites, 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. 2.3.

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.



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

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

Figure 2.3: 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 disorderd 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 eachother. 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).

Chapter 3

Examples

3.1 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.

> Hint: VMD is capable of showing pdb-files with several frames. This the way COTA 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:
>
> ```
> source pbctools.tcl
> source pbcbox.tcl
> pbcbox_update -width 2 -style tubes -origin {0.0 0.0 0.0}
> ```
>
> make sure the 'pbctools.tcl' and 'pbcbox.tcl' are in the current directory, they are located in the 'utils' directory of COTA. For NPT simulations the box is properly updated.

3.2 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 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 VMD: 'vmd AllComponents.pdb'.

```
SimulationType                  MC
NumberOfCycles                  1000
InitializationSteps             1000
PrintEvery                      100

Movies                          yes
WriteMoviesEvery                10

Forcefield                      TraPPE
CutOff                          12.8

Box 0
30 30 30
BoxAngles 0
90 90 90

ExternalTemperature system 0: 300.0

component 0 MoleculeName                    methane
            MoleculeDefinition              TraPPE
            TranslationProbability          1.0
            CreateNumberOfMolecules         100
```

Example 2: 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 regrow. 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 VMD: 'vmd AllComponents.pdb'.

```
SimulationType                MC
NumberOfCycles                1000
InitializationSteps           1000
PrintEvery                    100

Movies                        yes
WriteMoviesEvery              10

Forcefield                    TraPPE
CutOff                        12.8

Box 0
30 30 30
BoxAngles 0
90 90 90

ExternalTemperature system 0: 300.0

component 0 MoleculeName                  propane
            MoleculeDefinition            TraPPE
            TranslationProbability        1.0
            RotationProbability           1.0
            RegrowProbability             1.0
            CreateNumberOfMolecules       50

component 1 MoleculeName                  butane
            MoleculeDefinition            TraPPE
            TranslationProbability        1.0
            RotationProbability           1.0
            RegrowProbability             1.0
            CreateNumberOfMolecules       50
```

Example 3: Molecular dynamics of methane in a box measuring the mean-square displacement

A molecular dynamics run of 100 methane molecules in a $30 \times 30 \times 30$ Å box at 300 K. The simulations starts with 1000 InitializationSteps using Monte Carlo, the only MC move is translation. 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 'MSD/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'.

```
SimulationType                MD
NumberOfCycles                100000
InitializationSteps           1000
EquilibrationSteps            5000
PrintEvery                    1000

Ensemble system 0:            NVT

Forcefield                    TraPPE
CutOff                        12.8

ComputeMSD                    yes
PrintMSDEvery                 5000

Box 0
25 25 25
BoxAngles 0
90 90 90

ExternalTemperature system 0: 300.0

component 0 MoleculeName                  methane
            MoleculeDefinition            TraPPE
            TranslationProbability        1.0
            CreateNumberOfMolecules       300
```

Example 4: Adsorption of methane in MFI

```
SimulationType                MC
NumberOfCycles                50000
InitializationSteps           1000
PrintEvery                    1000

Forcefield                    ElenaSodiumCalcium
```

```
NumberOfSystems 1
Framework MFI
UnitCells 2 2 4
HeliumVoidFraction 0.29

ExternalTemperature system 0: 300.0

ExternalPressure system 0: 50000.0

component 0 MoleculeName                  methane
            MoleculeDefinition            TraPPE
            TranslationProbability        0.5
            RegrowProbability             0.5
            SwapProbability               1.0
            CreateNumberOfMolecules       0
```

Example 5: Henry coefficient of n-hexane in mono-clinic ERI

The monoclinic version of erionite (ERI) is names 'ERI_mono', the orthorhombic version is 'ERI'. The monoclinic version needs at least $3\times3\times3$ unit cells to be larger than twice the cutoff, while the orthorhombic needs $2\times2\times2$ (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. Note that by using multiple components several Henry coefficients can be computed simultaneously.

```
SimulationType                MC
NumberOfCycles                50000
PrintEvery                    1000

Forcefield                    ElenaSodiumCalcium

NumberOfSystems 1
Framework ERI_mono
UnitCells 3 3 3

ExternalTemperature system 0: 300.0

component 0 MoleculeName                  hexane
            MoleculeDefinition            TraPPE
            IdealRosenbluthValue          0.00312147
            WidomProbability              1.0
            CreateNumberOfMolecules       0
```

Example 6: Computing the radial distribution function of liquid water

The radial distribution function (RDF) is a good indication of the status of the fluid: solid, liquid or gas. COTA 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.

```
SimulationType                MD
NumberOfCycles                25000
InitializationSteps           100
EquilibrationSteps            5000
PrintEvery                    1000

Ensemble system 0:            NVT

Forcefield                    Tip5p-Ew

ComputeRDF                    yes
PrintRDFEvery                 1000

Box 0
24.83 24.83 24.83
BoxAngles 0
90 90 90

ExternalTemperature system 0: 298.0

component 0 MoleculeName                  water
            MoleculeDefinition            Tip5p
            TranslationProbability        0.5
            RotationProbability           0.5
            RegrowProbability             1.0
            CreateNumberOfMolecules       512
```

3.3 Non-basic examples

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

```
SimulationType                MC
```

```
NumberOfCycles                  50000
InitializationSteps             1000
PrintEvery                      100

Forcefield                      IRMOF-1

NumberOfSystems 1
Framework IRMOF-1
UnitCells 1 1 1
HeliumVoidFraction 0.81

ExternalTemperature system 0: 300.0

ExternalPressure system 0: 100000.0

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

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

Example 2: NPT Monte Carlo of methane

```
SimulationType                  MC
NumberOfCycles                  25000
InitializationSteps             5000
PrintEvery                      100

Forcefield                      TraPPE

Box 0
30 30 30
BoxAngles 0
90 90 90

ExternalTemperature system 0: 298.0

ExternalPressure system 0: 1000000.0

VolumeChangeProbability         0.1

component 0 MoleculeName                    methane
            MoleculeDefinition              TraPPE
            TranslationProbability          0.5
            RegrowProbability               0.5
            CreateNumberOfMolecules         256
```

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 recalibrated Tip5p-Ew model. Unfortunately, that model is defined using a cutoff always equal to half the box size, while COTA 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.

```
SimulationType                  MD
NumberOfCycles                  5000
InitializationSteps             0
EquilibrationSteps              5000
PrintEvery                      100

Ensemble system 0:              NPT

Forcefield                      Tip5p-Ew

Box 0
25 25 25
BoxAngles 0
```

```
90 90 90

ExternalTemperature system 0: 298.0

ExternalPressure system 0: 101325.0

component 0 MoleculeName                  water
            MoleculeDefinition            Tip5p
            CreateNumberOfMolecules       512
```

Example 4: Diffusion of methane in Ca/Na-LTA

The Linde Type A structure LTA-5A has 96 aluminum per unit cell. A common 5A sample has 32 Ca^{2+} and 32 Na^{+} charge balancing 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. A Monte Carlo initialization is used to distribute the ions properly over the cages, using the efficient regrow-move.

```
SimulationType                MD
NumberOfCycles                250000
InitializationSteps           5000
EquilibrationSteps            5000
PrintEvery                    1000

Ensemble system 0:            NVT

Forcefield                    ElenaSodiumCalcium

ComputeMSD                    yes
PrintMSDEvery                 5000

NumberOfSystems 1
Framework LTA5A
UnitCells 1 1 1

ExternalTemperature system 0: 600.0

component 0 MoleculeName                  calcium
            MoleculeDefinition            TraPPE
            TranslationProbability        1.0
            RegrowProbability             1.0
            ExtraFrameworkMolecule        yes
            CreateNumberOfMolecules       32

component 1 MoleculeName                  sodium
            MoleculeDefinition            TraPPE
            TranslationProbability        1.0
            RegrowProbability             1.0
            ExtraFrameworkMolecule        yes
            CreateNumberOfMolecules       32

component 2 MoleculeName                  methane
            MoleculeDefinition            TraPPE
            BlockPockets                  yes
            BlockPocketsFilename          LTA
            TranslationProbability        1.0
            RegrowProbability             1.0
            ExtraFrameworkMolecule        yes
            CreateNumberOfMolecules       64
```

Example 5: Gibbs ensemble simulation of CO_2

The Gibbs ensemble is way of computing coexistance 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 'GibbsSwapProbability' 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 COTA uses orientational biased insertions for small rigid molecules like CO_2. Fot this example about 10000 cycles are needed to equilibrate properly.

```
SimulationType                MC
NumberOfCycles                10000
InitializationSteps           10000
PrintEvery                    500

Forcefield                    TraPPE
CutOff                        12.8

Box 0
30 30 30
BoxAngles 0
```

```
90 90 90

Box 1
30 30 30
BoxAngles 1
90 90 90

ExternalTemperature system 0: 260.0
ExternalTemperature system 1: 260.0

GibbsVolumeChangeProbability     0.25

component 0 MoleculeName                    CO2
            MoleculeDefinitions             TraPPE
            TranslationProbability          0.5
            RotationProbability             0.5
            RegrowProbability               0.25
            GibbsSwapProbability            1.0
            CreateNumberOfMolecules         171 171
```

For this example at least 30000 cycles are needed to reach equilibrium (from scratch).

Example 6: Minimization of a flexible framework (fixed volume, NPT, NPT-PR)

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. 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 en positive eigenvalue means an decreas and increaes in energy, respectively. Most of the optimization time is spent on reaching a zero curvature structure, i.e. all positive eigenvalues. Note that minimization the structure in constant volume results in a finite (non-zero) stress.

```
SimulationType                  Minimization
NumberOfCycles                  1

Ensemble system 0:              NVT

Movies                          yes

Forcefield                      FlexibleIRMOF-1

NumberOfSystems 1
Framework IRMOF-1
UnitCells 1 1 1

Flexible yes
FrameworkDefinitions FlexibleIRMOF-1
```

The minimization needs 137 cycles to optimize IRMOF-1, the last steps are shown here. The convergence is very rapid (quadratic) near the minimium, and the fact that the minimum can be reached up to abitrary precision (the forces on all the atoms are 2.1×10^{-9} K/Å^2 or smaller). To compute spectra, frequencies and/or eigenmodes such a precison is needed.

```
Iteration: 0 Energy: -4285634.4052540446 RMS gradient: 198.577  Max gradient: 16375.5 Number of negative eigenvalues: 36 Number of zero eigenvalues: 0
      Box:      25.8320000000       0.0000000000       0.0000000000     Strain derivative: -432685.7300091403       0.0000000012       0.0000000012
                 0.0000000000      25.8320000000       0.0000000000                         0.0000000012 -432685.7300091118       0.0000000008
                 0.0000000000       0.0000000000      25.8320000000                         0.0000000013       0.0000000008 -432685.7300091651

Iteration: 1 Energy: -4326226.3748472529 RMS gradient: 120.363  Max gradient: 10035 Number of negative eigenvalues: 33 Number of zero eigenvalues: 0
      Box:      25.8320000000       0.0000000000       0.0000000000     Strain derivative: -467984.0448088938     303.9922161683     303.9922017320
                 0.0000000000      25.8320000000       0.0000000000                       303.9922161683 -467984.0448089051    -303.9922089491
                 0.0000000000       0.0000000000      25.8320000000                       303.9922017319    -303.9922089490 -467984.0448089215

.....................................................................

Iteration: 134 Energy: -4359378.7915293612 RMS gradient: 0.0159133  Max gradient: 4.07564 Number of negative eigenvalues: 0 Number of zero eigenvalues: 0
      Box:      25.8320000000       0.0000000000       0.0000000000     Strain derivative:  -97216.8253527574      20.6627104840     -47.3714185178
                 0.0000000000      25.8320000000       0.0000000000                        20.6627104840  -97244.1175874413      56.9694015542
                 0.0000000000       0.0000000000      25.8320000000                       -47.3714185178      56.9694015543  -97194.3047175675

Iteration: 135 Energy: -4359378.8284189086 RMS gradient: 0.00180126  Max gradient: 0.402827 Number of negative eigenvalues: 0 Number of zero eigenvalues: 0
      Box:      25.8320000000       0.0000000000       0.0000000000     Strain derivative:  -97393.9232945552       1.5675334613      -2.3936824593
                 0.0000000000      25.8320000000       0.0000000000                         1.5675334614  -97395.7314114243      -0.8216575973
                 0.0000000000       0.0000000000      25.8320000000                        -2.3936824593      -0.8216575973  -97383.4293638933

Iteration: 136 Energy: -4359378.8284616815 RMS gradient: 1.3358e-06  Max gradient: 0.000242751 Number of negative eigenvalues: 0 Number of zero eigenvalues: 0
      Box:      25.8320000000       0.0000000000       0.0000000000     Strain derivative:  -97413.4364964142       0.0009383593      -0.0019444096
                 0.0000000000      25.8320000000       0.0000000000                         0.0009383593  -97413.4392055222       0.0032563025
                 0.0000000000       0.0000000000      25.8320000000                        -0.0019444096       0.0032563024  -97413.4301569982

Iteration: 137 Energy: -4359378.8284616666 RMS gradient: 1.17958e-11  Max gradient: 2.10048e-09 Number of negative eigenvalues: 0 Number of zero eigenvalues: 0
      Box:      25.8320000000       0.0000000000       0.0000000000     Strain derivative:  -97413.4534586771       0.0000000045      -0.0000000034
```

```
                0.0000000000      25.8320000000       0.0000000000                          0.0000000044  -97413.4531071300       0.0000000036
                0.0000000000       0.0000000000      25.8320000000                         -0.0000000034       0.0000000036  -97413.4532545941

SUCCES: RMS Gradient tolerance 0.0001 reached (1.17958e-11)
       Max Gradient tolerance 0.0001 reached (2.10048e-09)
```

Note that minimization the structure in constant volume results in a finite (non-zero) stress. If one would like to also minimize the cell volume (isotropicly) use

```
Ensemble system 0:              NPT
```

or use for a change in cell-lengths and cell-angles

```
Ensemble system 0:              NPTPR
```

3.4 Advanced examples

Example 1: NPT molecular dynamics of flexible IRMOF-1

A NPT-ensemble simulation of a flxible 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, althought slow, is very much faster than Monte Carlo. The example show the code for flexible IRMOF-1 at 298K and 20 bar.

```
SimulationType                  MD
NumberOfCycles                  200000
EquilibrationSteps              100000
PrintEvery                      1000

Ensemble system 0:              NPT

Forcefield                      FlexibleIRMOF-1

NumberOfSystems 1
Framework IRMOF-1
UnitCells 1 1 1

Flexible yes
FrameworkDefinitions FlexibleIRMOF-1

ExternalTemperature system 0: 298.0
ExternalPressure    system 0: 2000000.0
```

Example 2: CO_2 adsorption in flexible IRMOF-1

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 conditions. The directory 'Restart' produced in the previous example should be copied to 'RestartInitial' and the option 'RestartFile' should be set to 'yes'.

```
SimulationType                  MC
NumberOfCycles                  100000
InitializationSteps             10000
PrintEvery                      1000

Forcefield                      FlexibleIRMOF-1

NumberOfSystems 1
Framework IRMOF-1
UnitCells 1 1 1
HeliumVoidFraction 0.82

Flexible yes
FrameworkDefinitions FlexibleIRMOF-1

ExternalTemperature system 0: 298.0
ExternalPressure    system 0: 2000000.0

VolumeChangeProbability         1.0
HybridMCMDMoveProbability       1.0

component 0 MoleculeName                    CO2
            MoleculeDefinition              TraPPE
            TranslationProbability          0.5
            RotationProbability             0.5
            RegrowProbability               0.5
            SwapProbability                 1.0
            CreateNumberOfMolecules         0
```

Example 3: argon adsorption in flexible MFI

Example 4: Benzene diffusion in flexible IRMOF-10

Example 5: Computing the infra-red spectrum using MD

Example 6: Converting pressure to fugacity

Example 7: Molecular properties (bond, bend, torsion angles)

Example 8: Gibbs multiple components (NPT-ensemble)

For systems of two or more components the NPT Gibbs can be used. Here, volume moves are performed on both boxes, similar to the usual NPT-ensemble. The other moves are similar to the NVT-version of Gibbs. An example for a CO_2/N_2 mixture at 253.15 K and 50 bar.

```
SimulationType                  MC
NumberOfCycles                  100000
InitializationSteps             50000
PrintEvery                      1000

Forcefield                      TraPPE
CutOff                          12.8

Box 0
30 30 30
BoxAngles 0
90 90 90

Box 1
30 30 30
BoxAngles 1
90 90 90

ExternalTemperature system 0: 253.15
ExternalTemperature system 1: 253.15

ExternalPressure system 0: 5000000.0
ExternalPressure system 1: 5000000.0

VolumeChangeProbability     0.25

component 0 MoleculeName                    CO2
            MoleculeDefinitions             TraPPE
            TranslationProbability          0.5
            RotationProbability             0.5
            RegrowProbability               0.25
            GibbsSwapProbability            1.0
            GibbsIdentityChangeProbability  1.0
               NumberOfIdentityChanges       2
               IdentityChangesList           0 1
            CreateNumberOfMolecules         75 75

component 1 MoleculeName                    N2
            MoleculeDefinitions             TraPPE
            TranslationProbability          0.5
            RotationProbability             0.5
            RegrowProbability               0.25
            GibbsSwapProbability            1.0
            GibbsIdentityChangeProbability  1.0
               NumberOfIdentityChanges       2
               IdentityChangesList           0 1
            CreateNumberOfMolecules         75 75
```

Example 9: Water vapor adsorption using fugacities

Example 10: Parallel tempering

Example 11: Hyper parallel tempering

Example 12: Elastic constants (0 Kelvin)

Example 13: Elastic constants (finite temperature)

Example 14: Normal mode analysis (frequencies and movies of the modes)

The input for the computation of the normal modes and frequencies is a fully minimized configuration (see example 6 of the nonbasic examples). if not, COTA first optimies the structure, and then procedes with the vibrational analysis.

```
SimulationType                Spectra
NumberOfCycles                1
RestartFile                   yes

Ensemble system 0:            NVT

Forcefield                    FlexibleIRMOF-1

ComputeNormalModes            yes
MinimumMode                   3
MaximumMode                   25
ModeResolution                500

NumberOfSystems 1
Framework IRMOF-1
UnitCells 1 1 1

Flexible yes
FrameworkDefinitions FlexibleIRMOF-1

ExternalTempersture system 0:   50.0
```

The normal-mode movies are written to 'VibrationalData/Modes/'. The maximum amplitude is computed from the temperature (remember that the virbrational analysis is only correct in the limit of zero temperature). The modes correpsonds to the eigenvalues which are order increasingly from lowest to highest frequencies. The first modes have zero eigenvalues and correspond to translation and rotation of the whole system. The directory 'VibrationalData/Spectrum/' contains 3 files: 'Frequencies.data','NormalModes.data', and 'Spectrum.data'. The first lists the frequencies in cm^{-1}, the second file contains the eigenmodes, and the latter can be used to plot the spectrum. The first column is the frequency, column 2 and 3 the non-charge weighted intensities and spectrum, columns 4 and 5, 6 and 7, and 8 and 9, contains the charge-weighted intensities in spectrum in the x, y, z-direction, respectively, the and the last two columns (10 and 11) are the isotropic charge-weighted intensities and spectrum.

Example 15: NPT Parinello-Rahman molecular dynamics (flexible box shape)

Example 16: Finding transition states (Baker's algorithm)

Example 17: Transitions state free energy barriers (dcTST)

Example 18: Sampling snapshots on the free energy barrier (dcTST)

Example 19: Computing dynamical recrossings (dcTST)

Example 20: 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 characterisation 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 IAZ 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. The diffraction pattern takes a few seonds of computation, and the result is written to 'PowderDiffraction/System[0]/'. It contains two files: 'PeakInformation.dat' and 'Spectrum.dat'.

```
SimulationType                MC
NumberOfCycles                0

NumberOfSystems 1
Framework TON
UnitCells 1 1 1

Forcefield                    ElenaSodiumCalcium

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'.

3.5 Auxiliary examples

Example 1: Computing the ideal gas Rosenbluth weight of a molecule

To compare simualtion 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 simulataneously, since they are computed from Widom insertions where the molecule is never actually inserted in the system.

```
SimulationType                  MC
NumberOfCycles                  25000
PrintEvery                      1000
PrintPropertiesEvery            1000

Forcefield                      ElenaSodiumCalcium

Box 0
30 30 30
BoxAngles 0
90 90 90

ExternalTemperature system 0: 298.0

component 0 MoleculeName                    pentane
            MoleculeDefinition              TraPPE
            WidomProbability                1.0
            CreateNumberOfMolecules         0

component 1 MoleculeName                    hexane
            MoleculeDefinition              TraPPE
            WidomProbability                1.0
            CreateNumberOfMolecules         0

component 2 MoleculeName                    heptane
            MoleculeDefinition              TraPPE
            WidomProbability                1.0
            CreateNumberOfMolecules         0

component 3 MoleculeName                    octane
            MoleculeDefinition              TraPPE
            WidomProbability                1.0
            CreateNumberOfMolecules         0
```

The output contains

```
Henry coefficients
        Component 0:               inf [mol/kg/Pa] (Rosenbluth factor new:      0.0197365763 [-])
        Component 1:               inf [mol/kg/Pa] (Rosenbluth factor new:      0.0029490119 [-])
        Component 2:               inf [mol/kg/Pa] (Rosenbluth factor new:      0.0004454807 [-])
        Component 3:               inf [mol/kg/Pa] (Rosenbluth factor new:      0.0000676949 [-])
Energy <U_gh>_1-<U_h>_0 from Widom
        Component 0:    1573.0718943505 [K]  (     13.0792510806 [kJ/mol])
        Component 1:    2106.2865896810 [K]  (     17.5126459593 [kJ/mol])
        Component 2:    2638.9423414500 [K]  (     21.9413935212 [kJ/mol])
        Component 3:    3164.8618412790 [K]  (     26.3141327528 [kJ/mol])
```

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 helium void-fraction of a structure (pore volume)

The void fraction is the empty space of a structure devided 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 roomtemperature. In practice it is easily computed from Widom particle insertion as the void fraction corresponds to the new Rosenbluth weight. The Rosenbluth weight, and therefore the helium void fraction of IRMOF-1 is approximately 0.8255. The pore volume is the void fraction times the unit cell volume.

```
SimulationType                MC
NumberOfCycles                500000
PrintEvery                    1000
PrintPropertiesEvery          1000

Forcefield                    IRMOF-1
CutOff                        12.8

NumberOfSystems 1
Framework IRMOF-1
UnitCells 1 1 1

ExternalTemperature system 0: 298.0

component 0 MoleculeName                  helium
            MoleculeDefinition            TraPPE
            WidomProbability              1.0
            CreateNumberOfMolecules       0
```

Example 3: 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 less than 5 minutes.

```
SimulationType                MC
NumberOfCycles                0

NumberOfSystems 1
Framework IRMOF-1
UnitCells 1 1 1

Forcefield                    IRMOF-1

ComputeSurfaceArea yes
SurfaceAreaProbeAtom H_com
SurfaceAreaSamplingPointsPerShere 100000
SurfaceAreaProbeDistance Minimum
```

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 4: Writing and using '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 gridpoints 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 COTA can handle a 'mixture' of grids and fully computed interactions.

```
SimulationType                MakeGrid

Forcefield                    FlexibleIRMOF-1

NumberOfSystems 1
Framework IRMOF-1
UnitCells 1 1 1

NumberOfGrids 2
GridTypes CH3_sp3 CH2_sp3
SpacingVDWGrid 0.1
SpacingCoulombGrid 0.1
```

The grids are stored in '/share/cota/grids/FlexibleIRMOF-1/IRMOF-1/0.100000' and the names are 'IRMOF-1_CH3_sp3_shifted.grid','IRMOF-1_CH2_sp3_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                  MC
NumberOfCycles                  50000
InitializationSteps             5000
PrintEvery                      1000

Forcefield                      FlexibleIRMOF-1

NumberOfSystems 1
Framework IRMOF-1
UnitCells 1 1 1

NumberOfGrids 2
GridTypes CH3_sp3 CH2_sp3
SpacingVDWGrid 0.1
SpacingCoulombGrid 0.1
UseTabularGrid yes

ExternalTemperature system 0: 298.0

ExternalPressure system 0: 50000.0

component 0 MoleculeName                    propane
            MoleculeDefinition              TraPPE
            TranslationProbability          0.5
            RegrowProbability               0.5
            SwapProbability                 1.0
            CreateNumberOfMolecules         0
```

In the beginning of the output file, in the framework section, the used grids are tested one by one. Make sure the relative error is smaller than 0.001. If not, either the wrong grid is used (difference in force field, cutoff etc.) or the structure requires a higher interpolation density.

```
PseudoAtom 10 [CH3_sp3]-Framework
================================================================================
       Boltzmann average energy VDW (table)               :   271.823233238181
       Boltzmann average energy VDW (full)                :   271.834362847671
       Boltzmann relative error VDW                       :     0.000156749008

PseudoAtom 11 [CH2_sp3]-Framework
================================================================================
       Boltzmann average energy VDW (table)               :   239.342931294268
       Boltzmann average energy VDW (full)                :   239.346105081400
       Boltzmann relative error VDW                       :     0.000136042234
```

Example 5: Writing and using binary restart "crash-recovery" files

Most supercomputer clusters have a limited run time. To be able to run longer than the queue limit or to recover after a computer crash one can use the crash-rocovery file. The file is binary, often around 300 MB in size, because one can not permit any change in the variables. A continuation using a restart-file results in *exactly* the same results. If "ContinueAfterCrash" is "yes", COTA checks for the file 'binary_restart.dat" in the directory "CrashRestart". If the file is present, then the restart is done otherwise the program just starts from scratch. During the production run every "WriteBinaryRestartFileEvery" steps the crash-file is written. This construction is useful, because some clusters automatically restart the jobs after a crash and COTA will automatically recover and continue where it left off. To use the crash-recovery feature, simply add

```
ContinueAfterCrash              yes
WriteBinaryRestartFileEvery     100000
```

Here, example 4 of the basic examples using the restart crash-recovery option. It writes the binary crash-file for the first time at cycle 10000, let it run a bit longer, stop the run, restart and check the difference in the output to check they are numerically the same after restarting.

```
SimulationType                  MD
NumberOfCycles                  25000
InitializationSteps             1000
EquilibrationSteps              5000
PrintEvery                      5000

ContinueAfterCrash              yes
WriteBinaryRestartFileEvery     10000

Ensemble system 0:              NVT

Forcefield                      ElenaSodiumCalcium
DeltaT                          0.001

ComputeMSD                      yes
PrintMSDEvery                   5000

NumberOfSystems 1
Framework LTA5A
```

```
UnitCells 1 1 1

ExternalTemperature system 0: 600.0

component 0 MoleculeName                    calcium
            MoleculeDefinition              TraPPE
            TranslationProbability          1.0
            RegrowProbability               1.0
            ExtraFrameworkMolecule          yes
            CreateNumberOfMolecules         32

component 1 MoleculeName                    sodium
            MoleculeDefinition              TraPPE
            TranslationProbability          1.0
            RegrowProbability               1.0
            ExtraFrameworkMolecule          yes
            CreateNumberOfMolecules         32

component 2 MoleculeName                    methane
            MoleculeDefinition              TraPPE
            BlockPockets                    yes
            BlockPocketsFilename            LTA
            TranslationProbability          1.0
            RegrowProbability               1.0
            ExtraFrameworkMolecule          no
            CreateNumberOfMolecules         64
```

Chapter 4

Data

4.1 framework properties: densities, helium void fractions and pore volumes

4.2 Ideal Rosenbluth values of alkanes

All values obtained from at least 500000 cycles using Widom insertions, 10 trial orientations.

4.2.1 Temperature 298 Kelvin

TraPPE (12.8 Å)			ElenaSodiumCalcium shifted potential (12 Å)		
molecule	Rosenbluth weight	energy [K]	molecule	Rosenbluth weight	energy [K]
methane	1.0	0	methane	1.0	1.0
ethane	1.0	149.329	ethane	1.0	149.331
propane	1.0	446.954	propane	1.0	446.987
butane	0.12954	1057.192	butane	0.129532	1057.673
pentane	0.0196737	1576.210	pentane	0.020119	1569.210
hexane	0.00293686	2108.807	hexane	0.00241585	2243.334
heptane	0.000442649	2639.330	heptane	0.000322586	2804.436
octane	6.71698e-05	3167.810	octane	4.72078e-05	3335.100
nonane	1.02506e-05	3697.629	nonane	7.21451e-06	3855.329
decane	1.56932e-06	4226.299	decane	1.12857e-06	4373.473
C11	2.40968e-07	4754.903	C11	1.78498e-07	4890.705
C12	3.71117e-08	5284.992	C12	2.84351e-08	5407.342
C13	5.72718e-09	5813.704	C13	4.55297e-09	5923.473
C14	8.85824e-10	6343.092	C14	7.30879e-10	6440.597
C15	1.37409e-10	6870.979	C15	1.17807e-10	6957.675
C16	2.13795e-11	7398.479	C16	1.90483e-11	7472.711
C17	3.33322e-12	7924.122	C17	3.09174e-12	7987.682
C18	5.21653e-13	8449.345	C18	5.03687e-13	8502.413
C19	8.19033e-14	8972.172	C19	8.24733e-14	9011.750
C20	1.29409e-14	9492.002	C20	1.35671e-14	9520.047
C21			C21		
C22			C22		
C23			C23		
C24			C24		
C25			C25		
C26			C26		
C27			C27		
C28			C28		
C29			C29		
C30			C30		
C31			C31		
C32			C32		
C33			C33		
C34			C34		
C35			C35		
C36			C36		
C37			C37		
C38			C38		
C39			C39		
C40			C40		

TraPPE (12.8 Å)			ElenaSodiumCalcium shifted potential (12 Å)		
molecule	Ros. weight	energy [K]	molecule	Ros. weight	energy [K]
ethane	1.0	149.384	ethane		
ethene	1.0	149.435	ethene		
propane	1.0	447.014	propane		
propene	1.0	447.223	propene		
butane	0.129545	1057.339	butane		
1-butene	0.367297	962.811	1-butene		
isobutane	1.0	905.402	isobutane		
pentane	0.0196726	1575.956	pentane		
2-methylbutane	0.180002	1416.199	2-methylbutane		
hexane	0.00293715	2108.440	hexane		
2-methylpentane	0.0295241	1813.108	2-methylpentane		
3-methylpentane	0.0329547	1849.841	3-methylpentane		
22-dimethylbutane	0.184165	1802.634	22-dimethylbutane		
23-dimethylbutane	0.0562399	2036.400	23-dimethylbutane		
heptane	0.000442687	2638.905	heptane		
2-methylhexane	0.00441973	2356.015	2-methylhexane		
3-methylhexane	0.0052706	2255.123	3-methylhexane		
22-dimethylpentane	0.0344851	2046.169	22-dimethylpentane		
23-dimethylpentane	0.00953315	2398.229	23-dimethylpentane		
24-dimethylpentane	0.0248979	2060.813	24-dimethylpentane		
33-dimethylpentane	0.0302527	2197.787	33-dimethylpentane		
223-trimethylbutane	0.027949	2317.772	223-trimethylbutane		
octane	6.71857e-05	3168.044	octane		
2-methylheptane	0.000676252	2880.950	2-methylheptane		
3-methylheptane	0.000801343	2796.112	3-methylheptane		
4-methylheptane	0.000852958	2660.211	4-methylheptane		
22-dimethylhexane	0.00491649	2615.482	22-dimethylhexane		
25-dimethylhexane	0.00671134	2618.868	25-dimethylhexane		
24-dimethylhexane	0.00427094	2500.988	24-dimethylhexane		
224-trimethylpentane	0.00067726	3757.658	224-trimethylpentane		

Chapter 5

Modifying the Source Code

Part II

Theory

Chapter 6

Molecular Simulations

Chapter 7

Potentials

7.1 Intra-molecular potentials

7.1.1 Bond-stretching potentials

- HARMONIC_BOND

$$U = \frac{1}{2} p_0 \left(r - p_1\right)^2 \tag{7.1}$$

p_0/k_B in units of Kelvin $\cdot$ Ångstrom^{-2}, p_1 in Ångstrom.

- MORSE_BOND

$$U = p_0 \left[\left(1 - e^{-p_1 (r - p_2)} \right)^2 - 1 \right] \tag{7.2}$$

p_0/k_B in units of Kelvin, p_1 dimensionless, and p_2 in Ångstrom.

- LJ_12_6_BOND

$$U = \frac{p_0}{r^{12}} - \frac{p_1}{r^6} \tag{7.3}$$

$p_0/k_B, p_1/k_B$ in units of Kelvin.

- LENNARD_JONES_BOND

$$U = 4p_0 \left[\left(\frac{p_1}{r} \right)^{12} - \left(\frac{p_1}{r} \right)^6 \right] \tag{7.4}$$

p_0/k_B in units of Kelvin, p_1 in Ångstrom, and p_2/k_B in units of Kelvin $\cdot$ Ångstrom^6

- BUCKINGHAM_BOND

$$U = p_0 e^{-p_1 r} - \frac{p_2}{r^6} \tag{7.5}$$

p_0/k_B in units of Kelvin, p_1 in units of Ångstrom^{-1}, and p_2 in units of Kelvin.

- RESTRAINED_HARMONIC_BOND

$$U = \begin{cases} \frac{1}{2} p_0 \left(r - p_1\right)^2 & |r - p_1| \leq p_2 \\ \frac{1}{2} p_0 p_2^2 + p_0 p_2 \left(|r - p_1| - p_2\right) & |r - p_1| > p_2 \end{cases} \tag{7.6}$$

p_0/k_B in units of Kelvin, p_2 in Ångstrom, and p_3 in Ångstrom.

- QUARTIC_BOND

$$U = \frac{1}{2} p_0 \left(r - p_1\right)^2 + \frac{1}{3} p_2 \left(r - p_1\right)^3 + \frac{1}{4} p_3 \left(r - p_1\right)^4 \tag{7.7}$$

p_0/k_B in units of Kelvin, p_1 in Ångstrom, p_2/k_B in units of Kelvin, and p_3/k_B in units of Kelvin.

- RIGID_BOND
- FIXED_BOND
- MEASURE_BOND

7.1.2 Urey-Bradley potentials

- HARMONIC_UREYBRADLEY

$$U = \frac{1}{2} p_0 \left(r - p_1\right)^2 \tag{7.8}$$

p_0/k_B in units of Kelvin, p_1 in Ångstrom.

- MORSE_UREYBRADLEY

$$U = p_0 \left[\left(1 - e^{-p_1 (r - p_2)} \right)^2 - 1 \right] \tag{7.9}$$

p_0/k_B in units of Kelvin, p_1 dimensionless, and p_2 in Ångstrom.

- LJ_12_6_UREYBRADLEY

$$U = \frac{p_0}{r^{12}} - \frac{p_1}{r^6} \tag{7.10}$$

$p_0/k_B, p_1/k_B$ in units of Kelvin.

- RESTRAINED_HARMONIC_UREYBRADLEY

$$U = \begin{cases} \frac{1}{2} p_0 \left(r - p_1\right)^2 & |r - p_1| \leq p_2 \\ \frac{1}{2} p_0 p_2^2 + p_0 p_2 \left(|r - p_1| - p_2\right) & |r - p_1| > p_2 \end{cases} \tag{7.11}$$

p_0/k_B in units of Kelvin, p_2 in Ångstrom, and p_3 in Ångstrom.

- QUARTIC_UREYBRADLEY

$$U = \frac{1}{2} p_0 \left(r - p_1\right)^2 + \frac{1}{3} p_2 \left(r - p_1\right)^3 + \frac{1}{4} p_3 \left(r - p_1\right)^4 \tag{7.12}$$

p_0/k_B in units of Kelvin, p_1 in Ångstrom, p_2/k_B in units of Kelvin, and p_3/k_B in units of Kelvin.

- RIGID_UREYBRADLEY
- FIXED_UREYBRADLEY
- MEASURE_UREYBRADLEY

7.1.3 Bending potential

- HARMONIC_BEND

$$U = \frac{1}{2} p_0 \left(\theta_{ijk} - p_1\right)^2 \tag{7.13}$$

p_0/k_B in units of Kelvin and p_1 in degrees.

- QUARTIC_BEND

$$U = \frac{1}{2} p_0 \left(\theta_{ijk} - p_1\right)^2 + \frac{1}{3} p_2 \left(\theta_{ijk} - p_1\right)^3 + \frac{1}{4} p_3 \left(\theta_{ijk} - p_1\right)^4 \tag{7.14}$$

p_0/k_B in units of Kelvin, p_1 in degrees, p_2/k_B in units of Kelvin, and p_3/k_B in units of Kelvin.

- HARMONIC_COSINE_BEND

$$U = \frac{1}{2} p_0 \left(\cos\theta_{ijk} - \cos p_1\right)^2 \tag{7.15}$$

p_0/k_B in units of Kelvin and p_1 in degrees.

- COSINE_BEND

$$U = p_0 \left(1 + \cos\left(p_1 \theta_{ijk} - p_2\right)\right) \tag{7.16}$$

p_0/k_B in units of Kelvin, p_1 dimensionless, and p_2 in degrees.

7.1.4 Torsion potential

- HARMONIC_COSINE_DIHEDRAL

$$U = \frac{1}{2} p_0 \left[\cos\left(\phi_{ijkl}\right) - \cos\left(p_2\right)\right]^2 \tag{7.17}$$

p_0/k_B in units of Kelvin, p_2 in degrees.

- HARMONIC_DIHEDRAL

$$U = \frac{1}{2} p_0 \left(\phi_{ijkl} - p_2\right)^2 \tag{7.18}$$

p_0/k_B in units of Kelvin, p_2 in degrees.

- THREE_COSINE_DIHEDRAL

$$U = \frac{1}{2} p_0 \left[1 + \cos\left(\phi_{ijkl}\right)\right] + \frac{1}{2} p_1 \left[1 - \cos\left(2\phi_{ijkl}\right)\right] + \frac{1}{2} p_2 \left[1 + \cos\left(3\phi_{ijkl}\right)\right] \tag{7.19}$$

$p_0/k_B, p_1/k_B, p_2/k_B$ in units of Kelvin.

- 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 accoording to the polymer convention $\phi'(trans) = 0$.

$$U = \sum_{n=0}^{5} p_n \cos^n\left(\phi'_{ijkl}\right) \tag{7.20}$$

$$= p_0 + p_1 \cos\left(\phi'_{ijkl}\right) + p_2 \cos^2\left(\phi'_{ijkl}\right) + p_3 \cos^3\left(\phi'_{ijkl}\right) p_4 \cos^4\left(\phi'_{ijkl}\right) + p_5 \cos^5\left(\phi'_{ijkl}\right) \tag{7.21}$$

$p_0/k_B, \ldots, p_5/k_B$ in units of Kelvin. Rewritten in terms of ϕ the potential reads

$$U = p_0 - p_1 \cos\left(\phi_{ijkl}\right) + p_2 \cos^2\left(\phi_{ijkl}\right) - p_3 \cos^3\left(\phi_{ijkl}\right) + p_4 \cos^4\left(\phi_{ijkl}\right) - p_5 \cos^5\left(\phi_{ijkl}\right) \tag{7.22}$$

- TRAPPE_DIHEDRAL

$$U = p_0 + p_1 \left[1 + \cos\left(\phi_{ijkl}\right)\right] + p_2 \left[1 - \cos\left(2\phi_{ijkl}\right)\right] + p_3 \left[1 + \cos\left(3\phi_{ijkl}\right)\right] \tag{7.23}$$

$p_0/k_B, p_1/k_B, p_2/k_B, p_3/k_B$ in units of Kelvin.

- CVFF_DIHEDRAL

$$U = p_0 \left[1 + \cos\left(p_1 \phi_{ijkl} - p_2\right)\right] \tag{7.24}$$

p_0/k_B in units of Kelvin, p_1 dimensionless, and p_2 in degrees.

- OPLS_DIHEDRAL

$$U = \frac{1}{2}p_0 + \frac{1}{2}p_1\left[1 + \cos\left(\phi_{ijkl}\right)\right] + \frac{1}{2}p_2\left[1 - \cos\left(2\phi_{ijkl}\right)\right] + \frac{1}{2}p_3\left[1 + \cos\left(3\phi_{ijkl}\right)\right] \tag{7.25}$$

$p_0/k_B, p_1/k_B, p_2/k_B, p_3/k_B$ in units of Kelvin.

7.1.5 Improper torsion potential

- HARMONIC_COSINE_IMPROPER_DIHEDRAL

$$U = \frac{1}{2}p_0\left[\cos\left(\phi_{ijkl}\right) - \cos\left(p_2\right)\right]^2 \tag{7.26}$$

p_0/k_B in units of Kelvin, p_2 in degrees.

- HARMONIC_IMPROPER_DIHEDRAL

$$U = \frac{1}{2}p_0\left(\phi_{ijkl} - p_2\right)^2 \tag{7.27}$$

p_0/k_B in units of Kelvin, p_2 in degrees.

- THREE_COSINE_IMPROPER_DIHEDRAL

$$U = \frac{1}{2}p_0\left[1 + \cos\left(\phi_{ijkl}\right)\right] + \frac{1}{2}p_1\left[1 - \cos\left(2\phi_{ijkl}\right)\right] + \frac{1}{2}p_2\left[1 + \cos\left(3\phi_{ijkl}\right)\right] \tag{7.28}$$

$p_0/k_B, p_1/k_B, p_2/k_B$ in units of Kelvin.

- 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 accoording to the polymer convention $\phi'(trans) = 0$.

$$U = \sum_{n=0}^{5} p_n \cos^n\left(\phi'_{ijkl}\right) \tag{7.29}$$

$$= p_0 + p_1\cos\left(\phi'_{ijkl}\right) + p_2\cos^2\left(\phi'_{ijkl}\right) + p_3\cos^3\left(\phi'_{ijkl}\right) p_4\cos^4\left(\phi'_{ijkl}\right) + p_5\cos^5\left(\phi'_{ijkl}\right) \tag{7.30}$$

$p_0/k_B, \ldots, p_5/k_B$ in units of Kelvin. Rewritten in terms of ϕ the potential reads

$$U = p_0 - p_1\cos\left(\phi_{ijkl}\right) + p_2\cos^2\left(\phi_{ijkl}\right) - p_3\cos^3\left(\phi_{ijkl}\right) + p_4\cos^4\left(\phi_{ijkl}\right) - p_5\cos^5\left(\phi_{ijkl}\right) \tag{7.31}$$

- TRAPPE_IMPROPER_DIHEDRAL

$$U = p_0 + p_1\left[1 + \cos\left(\phi_{ijkl}\right)\right] + p_2\left[1 - \cos\left(2\phi_{ijkl}\right)\right] + p_3\left[1 + \cos\left(3\phi_{ijkl}\right)\right] \tag{7.32}$$

$p_0/k_B, p_1/k_B, p_2/k_B, p_3/k_B$ in units of Kelvin.

- CVFF_IMPROPER_DIHEDRAL

$$U = p_0\left[1 + \cos\left(p_1\phi_{ijkl} - p_2\right)\right] \tag{7.33}$$

p_0/k_B in units of Kelvin, p_1 dimensionless, and p_2 in degrees.

- OPLS_IMPROPER_DIHEDRAL

$$U = \frac{1}{2}p_0 + \frac{1}{2}p_1\left[1 + \cos\left(\phi_{ijkl}\right)\right] + \frac{1}{2}p_2\left[1 - \cos\left(2\phi_{ijkl}\right)\right] + \frac{1}{2}p_3\left[1 + \cos\left(3\phi_{ijkl}\right)\right] \tag{7.34}$$

$p_0/k_B, p_1/k_B, p_2/k_B, p_3/k_B$ in units of Kelvin.

Chapter 8

Monte Carlo

8.1 The Metropolis algorithm

The Markov Chain Monte Carlo method (MCMC) is an important tool to estimate the average properties of systems with a very large number of accessible states. Often the quantity of interest are not the configurational part of the partition function itself, but averages of the type

$$\langle A \rangle = \frac{\int e^{-\beta U\left(\mathbf{r}^N\right)} A\left(\mathbf{r}^N\right)\, d\mathbf{r}^N}{\int e^{-\beta U\left(\mathbf{r}^N\right)}\, d\mathbf{r}^N} \tag{8.1}$$

where $\beta = 1/(k_B T)$, with k_B the Boltzmann constant, and $U\left(\mathbf{r}^N\right)$ is the total energy of the system with N particles at positions $\mathbf{r}^N$. The configurational part of the partition function is denoted by Z

$$Z \equiv \int e^{-\beta U\left(\mathbf{r}^N\right)}\, d\mathbf{r}^N \tag{8.2}$$

the ration $e^{-\beta U}/Z$ is the probability density of finding the system in a configuration around $\mathbf{r}^N$. The Monte Carlo scheme makes use of the fact that only the relative probability of visiting points in configuration space is needed, *not* the absolute probability. To visit points with the correct frequency, the MCMC algorithm generates random trial moves from the current ("old") state (o) to a new state (n). To show that an arbitrary initial distribution eventually relaxes to the equilibrium distribution, it is often convenient to apply the condition of detailed balance (as is used in the original Metropolis scheme). If $P_B(o)$ and $P_B(n)$ denote the probability of finding the system in in state (o) and (n), respectively, and $\alpha(o \to n)$ and $\alpha(n \to o)$ denote the conditional probability to perform a trial move from $o \to n$ and $o \to n$, respectively, then the probability $P_{\text{acc}}(o \to n)$ to accept the trial move from $o \to n$ is related to $P_{\text{acc}}(n \to o)$ by the following

$$P_B(o)\alpha(o \to n)P_{\text{acc}}(o \to n) = P_B(n)\alpha(n \to o)P_{\text{acc}}(n \to o) \tag{8.3}$$

Metropolis et al. assumed that

$$\alpha(o \to n) = \alpha(n \to o) \tag{8.4}$$

and fixed the acceptance probability using

$$P_{\text{acc}}(o \to n) = \min\left(1, \frac{P_B(n)}{P_B(o)}\right) \tag{8.5}$$

8.2 Configurational-bias Monte-Carlo (CBMC)

Conventional Monte Carlo is time-consuming for long chain molecules. The fraction of successful insertions into the sieve is too low. To increase the number of successfully inserted molecules we apply the CBMC

technique. In the CBMC scheme it is convenient to split the total potential energy U of a trial site into two parts.

$$U = U^{\text{int}} + U^{\text{ext}}. \tag{8.6}$$

The first part is the internal, bonded potential U^{int} which is used for the generation of trial orientations. The second part of the potential, the external potential U^{ext}, is used to bias the selection of a site from the set of trial sites. This bias is exactly removed by adjusting the acceptance rules. In the CBMC technique a molecule is grown segment-by-segment. For each segment we generate a set of k trial orientations according to the internal energy U^{int} and compute the external energy $U_i^{\text{ext}}(j)$ of each trial position j of segment i. In this work the number of trial positions k for both NVT and μVT is set to 10. We select one of these trial positions with a probability

$$P_i(j) = \frac{e^{-\beta U_i^{\text{ext}}(j)}}{\sum_{l=1}^{k} e^{-\beta U_i^{\text{ext}}(l)}} = \frac{e^{-\beta U_i^{\text{ext}}(j)}}{w(i)}. \tag{8.7}$$

The selected trial orientation is added to the chain and the procedure is repeated until the entire molecule has been grown. For this newly grown molecule we compute the so-called Rosenbluth factor

$$W^{\text{new}} = \prod_i w(i). \tag{8.8}$$

To compute the old Rosenbluth factor W^{old} of an already existing chain, $k-1$ trial orientations are generated for each segment. These orientations, together with the already existing bond, form the set of k trial orientations. In a dynamic scheme, a Markov chain of states is generated. The average of a property is the average of over the elements of the Markov chain. For an infinite Markov chain the expression is exact. Every new configuration is accepted or rejected using an acceptance/rejection rule.

We have defined μ^{ex} as the difference in chemical potential of the interacting alkane and an alkane in the ideal gas state. The Rosenbluth weight $\langle W^{\text{IG}} \rangle$ of the reference state of the ideal gas is needed when comparing with real experimental data. When CBMC is used, it is straightforward to show that $e^{-\beta \Delta U}$ has to be replaced by $\frac{W(\text{new chain})}{W(\text{IG})}$ for inserting a particle and by $\frac{W(\text{IG})}{W(\text{old chain})}$ for the deletion of a particle. There are two ways to obey detailed balance:

- Every time a transfer attempt to and from the reservoir, the Rosenbluth factor W^{IG} is computed. Because the particle reservoir is an ideal gas, there are only intramolecular interactions present.
- Detailed balance is also obeyed when W^{IG} is replaced by $\langle W^{\text{IG}} \rangle$, i.e. the *average* Rosenbluth weight of a chain in the reservoir. This implies that $\langle W^{\text{IG}} \rangle$ has to be computed only *once* for a given molecule and temperature.

8.3 Monte Carlo moves

8.3.1 Single particle moves

- Translation move
  A chain is selected at random and given a random displacement. The maximum displacement is taken such that 50% of the moves is accepted. The acceptance rule is

  $$\text{acc}\,(\text{old} \rightarrow \text{new}) = \min\left(1, e^{-\beta\left(U^{\text{new}} - U^{\text{old}}\right)}\right). \tag{8.9}$$

  Note that the energy of the new configuration U^{new} and the energy of the old configuration U^{old} only differ in the external energy.
- Random translation move
  A chain is selected at random and given a new random position in the simulation cell. The acceptance rule is given by Eq. 8.9. Note that the energy of the new configuration U^{new} and the energy of the old configuration U^{old} only differ in the external energy.

- Rotation move
  A chain is selected at random and given a new random orientation. The center of the rotation is the center of mass. The acceptance rule is given by Eq. 8.9. Again, the energy of the new configuration U^{new} and the energy of the old configuration U^{old} only differ in the external energy.

- Full regrow move
  A chain is selected at random and is completely regrown at a random position. This move is essential for N, V, T to change the internal configuration of a molecule, and during this move data for the average Rosenbluth weight can be collected. The acceptance rule for full regrow is given by

$$\text{acc}\,(\text{old} \rightarrow \text{new}) = \min\left(1, \frac{W^{\text{new}}}{W^{\text{old}}}\right). \tag{8.10}$$

- Partial regrow move
  A chain is selected at random and part of the molecule is regrown. It is decided at random which part of the chain is regrown and with which segment the regrown is started. The acceptance rule for partial regrow is given by Eq. 8.10.

- Swap insertion/deletion move

  - Insertion move
    A chain is grown at a random position. The acceptance rule for insertion of the particle is given by

$$\text{acc}(N \rightarrow N+1) = \min\left(1, \frac{W^{\text{new}}\beta V}{N+1}\frac{f}{\langle W^{\text{IG}}\rangle}\right). \tag{8.11}$$

  - Deletion move
    A chain is chosen at a random position and the old Rosenbluth factor is computed. The acceptance rule for deletion of the particle is given by

$$\text{acc}(N \rightarrow N-1) = \min\left(1, \frac{N}{W^{\text{old}}\beta V}\frac{\langle W^{\text{IG}}\rangle}{f}\right). \tag{8.12}$$

- Identity change move (mixtures)
  The identity-change trial move is called semi-grand ensemble, but it can also be seen as a special case of the Gibbs ensemble. One of the components is selected at random and an attempt is made to change its identity. The acceptance rule is given by

$$\text{acc}\,(\text{A} \rightarrow \text{B}) = \min\left(1, \frac{W^{\text{new}} f_B \langle W_A^{\text{IG}}\rangle N_A}{W^{\text{old}} f_A \langle W_B^{\text{IG}}\rangle (N_B+1)}\right), \tag{8.13}$$

  where f_A and f_B are the fugacities of components A and B, and N_A and N_B are the number of particles.

- Widom particle insertion move

- Gibbs particle transfer move

- Framework change move
  We randomly select a framework atom and give it a random displacement [**?**]. The maximum displacement is taken such that 50% of the moves are accepted. The acceptance rule is

$$\text{acc}\,(\text{old} \rightarrow \text{new}) = \min\left(1, e^{-\beta\left(U^{\text{new}} - U^{\text{old}}\right)}\right). \tag{8.14}$$

Note that this scheme works for zeolites but is insufficient for MOFs because concerted movement, such as the rotation of the phenyl-group in a linker molecule is not accounted for using individual atom moves.

8.3.2 System moves

- Volume change move
  If we perform a random walk in $\ln V$, the probability of finding volume V is given by

$$\mathcal{N}\left(\ln\left(V\right);\mathbf{s}^N\right) \propto e^{-\beta(PV+U)+(N+1)\ln V} \tag{8.15}$$

  For a random walk in $\ln V$, instead of in V, the domain of this walk coincides with all possible values of V [**?**]. Furthermore, the average step size turns out to be less sensitive to the density. The acceptance criterion is based on the difference of enthalpies:

$$\Delta H = \Delta U + P\left(V' - V\right) - (N+1)\frac{1}{\beta}\ln\frac{V'}{V} \tag{8.16}$$

$$\mathrm{acc}\left(o \to n\right) = \min\left(1, e^{-\beta\Delta H}\right) \tag{8.17}$$

  To change the volume we scale all the framework atoms "atomically', all flexible adsorbates "atomically" and all rigid adsorbates "molecularly". Molecular scaling changes the center of mass of the molecule but leaves the internal configuration untouched.

- Box shape change volume move

- Gibbs particle transfer move

- Gibbs volume change move

- Hybrid MC/MD move
  We perform a small molecular dynamics run of M time steps to obtain a new configuration [**?**]. The acceptance rule is

$$\mathrm{acc}\left(\mathrm{old} \to \mathrm{new}\right) = \min\left(1, e^{-\beta\left(U^{\mathrm{new}} - U^{\mathrm{old}}\right)}\right). \tag{8.18}$$

  Note that $\Delta U = U^{\mathrm{new}} - U^{\mathrm{old}}$ is the integration error. In relative units this error is quite small, but not in absolute units, i.e. with $M = 5$ and $\Delta t = 0.5$ fs for a CO_2 isotherm results in an acceptance of approximately 80%.

Chapter 9

Molecular Dynamics

9.1 Ensembles

9.1.1 NVE

The Hamiltonian for an N-particle system subjected inly to interparticle interactions is

$$H(\mathbf{p},\mathbf{r}) \equiv H(\mathbf{p}_1,\ldots,\mathbf{p}_N,\mathbf{r}_1,\ldots,\mathbf{r}_N) = \sum_{i=1}^{N}\frac{\mathbf{p}_i^2}{2m_i} + U(\mathbf{r}_1,\ldots,\mathbf{r}_N) \tag{9.1}$$

where $\mathbf{p}_1,\ldots,\mathbf{p}_N$ are the momenta of the particles defined by $\mathbf{p}_i = m_i\mathbf{v}_i$, and $U(\mathbf{r}_1,\ldots,\mathbf{r}_N)$ is the interaction potential. The forces are given by

$$\mathbf{f}_i = -\frac{\partial U}{\partial \mathbf{r}_i} \tag{9.2}$$

Newton's equation of motion

$$m_i\ddot{r}_i = \mathbf{f}_i \tag{9.3}$$

can be derived from Eq. 9.2 according to Hamilton's equations

$$\dot{r}_i = \frac{\partial H}{\partial \mathbf{p}_i} = \frac{\mathbf{p}_i}{m_i} \tag{9.4}$$

$$\dot{\mathbf{p}}_i = -\frac{\partial H}{\partial \mathbf{r}_i} = -\frac{\partial U}{\partial \mathbf{r}_i} = \mathbf{f}_i(\mathbf{r}_i,\ldots,\mathbf{r}_N) \tag{9.5}$$

These equations can be cast in the general form

$$\dot{x} = \imath L x \tag{9.6}$$

where x is the phase space vector and $\imath L$ is the Liouville operator given by

$$\imath L = \{\ldots,H\} \equiv \sum_{i=1}^{N}\left[\frac{\partial H}{\partial \mathbf{p}_i}\frac{\partial}{\partial \mathbf{r}_i} - \frac{\partial H}{\partial \mathbf{r}_i}\frac{\partial}{\partial \mathbf{r}_i}\right] = \sum_{i=1}^{N}\left[\frac{\mathbf{p}_i}{m_i}\frac{\partial}{\partial \mathbf{r}_i} + \mathbf{f}_i\frac{\partial}{\partial \mathbf{p}_i}\right] \tag{9.7}$$

Eq. 9.6 has the formal solution

$$x(t) = e^{\imath L t}x(0) \tag{9.8}$$

Eq. 9.8 is the starting point for the derivation of numerical integration procedures. The unitary operator, $e^{\imath L t}$, is the *classical propagator*. Its action on $x(0)$ cannot be determined analytically for any but a few simple cases. However, the formal solution to Hamilton's equations can be used to generate practical numerical integrators through introduction of an approximation to the classical propagator into Eq. 9.8. Suppose,

for example, that the Liouville operator $\imath L$ can be written as the sum of two parts, $\imath L = \imath L_1 + \imath L_2$, such that the action of the classical propagator on $x\,(0)$ for each part can be evaluated analytically. The classical propagator can be rewritten using the Trotter theorem, which states:

$$e^{\imath L t} = e^{(\imath L_1 + \imath L_2)t} = \lim_{P\to\infty} \left[e^{\left(\frac{\imath L_2 t}{2P}\right)} e^{\left(\frac{\imath L_1 t}{P}\right)} e^{\left(\frac{\imath L_2 t}{2P}\right)} \right]^P \tag{9.9}$$

Defining $\Delta t = \frac{t}{P}$ for finite P, the approximation

$$e^{\imath L \Delta t} \approx e^{\frac{\imath L_1 \Delta t}{2}} e^{\imath L_2 \Delta t} e^{\frac{\imath L_1 \Delta t}{2}} + \mathcal{O}\left(\Delta t^3\right) \tag{9.10}$$

$$e^{\imath L P \Delta t} \approx \prod_{k=1}^{P} e^{\frac{\imath L_1 \Delta t}{2}} e^{\imath L_2 \Delta t} e^{\frac{\imath L_1 \Delta t}{2}} + \mathcal{O}\left(t \Delta t^2\right) \tag{9.11}$$

can be made, which yields a numerical integration procedure that is accurate to second order in the time step at long times. Consider the choice

$$\imath L_1 = \sum_{i=1}^{N} \frac{\mathbf{f}_i}{m_i} \nabla_{\mathbf{v}_i} = \sum_{i=1}^{N} \mathbf{f}_i \cdot \frac{\partial}{\partial \mathbf{p}_i} \tag{9.12}$$

$$\imath L_2 = \sum_{i=1}^{N} \mathbf{v}_i \nabla_{\mathbf{r}_i} = \sum_{i=1}^{N} \frac{\mathbf{p}_i}{m_i} \cdot \frac{\partial}{\partial \mathbf{r}_i} \tag{9.13}$$

We can use an explicit property of any operator of the form $e^{a\frac{\partial}{\partial q(x)}}$

$$e^{a\left(\frac{\partial}{\partial x}\right)} f\left((x\right) = f\left(x + a\right) \tag{9.14}$$

and use it in

$$e^{\imath L_1 \Delta t/2}\left(x\right) e^{\imath L_2 \Delta t}\left(x\right) e^{\imath L_1 \Delta t/2}\left(x\right) = e^{\frac{1}{2}\Delta t \mathbf{f}_i \cdot \frac{\partial}{\partial \mathbf{p}_i}}\left(x\right) e^{\Delta t \frac{\mathbf{p}_i}{m_i} \cdot \frac{\partial}{\partial \mathbf{r}_i}}\left(x\right) e^{\frac{1}{2}\Delta t \mathbf{f}_i \cdot \frac{\partial}{\partial \mathbf{p}_i}}\left(x\right) \tag{9.15}$$

$$e^{\frac{1}{2}\Delta t \mathbf{f}_i \cdot \frac{\partial}{\partial \mathbf{p}_i}}\left(\mathbf{p}_i\right) = \mathbf{p}_i + \frac{1}{2}\Delta t \mathbf{f}_i \tag{9.16}$$

$$e^{\Delta t \frac{\mathbf{p}_i}{m_i} \cdot \frac{\partial}{\partial \mathbf{r}_i}}\left(\mathbf{r}_i\right) = \mathbf{r}_i + \Delta t \frac{\mathbf{p}_i}{m_i} \tag{9.17}$$

The phase space vector x consists of the momenta $\mathbf{p}_i$ and the positions $\mathbf{r}_i$. Since the forces $\mathbf{f}_i$ are taken to depend only on the positions, the operator $e^{(\imath L_2 \Delta t/2)}$ becomes a translation operator on the momenta: $\mathbf{p}_i \to \mathbf{p}_i + (\Delta t/2)\,\mathbf{f}_i\,(\mathbf{r})$. Similarly, $e^{(\imath L_1 \Delta t/2)}$ is a translation operator on the positions: $\mathbf{r}_i \to \mathbf{r}_i + \Delta t\,(\mathbf{p}_i/m_i)$. Combining these two facts allows the action of the operator in Eq. 9.11 on the full set of positions and momenta to be evaluated analytically, yielding the approximate evolution,

for $i \leftarrow 1$ **to** N **do**
    $\mathbf{v}_i \leftarrow \mathbf{v}_i + \frac{\Delta t}{2m_i}\mathbf{f}_i$;
end
for $i \leftarrow 1$ **to** N **do**
    $\mathbf{r}_i \leftarrow \mathbf{r}_i + \Delta t\ \mathbf{v}_i$;
end
Compute new forces on all the particles;
for $i \leftarrow 1$ **to** N **do**
    $\mathbf{v}_i \leftarrow \mathbf{v}_i + \frac{\Delta t}{2m_i}\mathbf{f}_i$;
end

Algorithm 1: NVE integration algorithm.

9.1.2 NVT

$$\begin{aligned}
\dot{\mathbf{r}}_i &= \frac{\mathbf{p}_i}{m_i} \\
\dot{\mathbf{p}}_i &= \mathbf{f}_i - \frac{p_{\eta_1}}{Q_1}\mathbf{p}_i \\
\dot{\eta}_k &= \frac{p_{\eta_k}}{Q_k} \qquad k = 1, \ldots, M \\
\dot{p}_{\eta_k} &= G_k - \frac{p_{\eta_{k+1}}}{Q_{k+1}} p_{\eta_k} \\
\dot{p}_{\eta_M} &= G_M
\end{aligned} \tag{9.18}$$

where the heat-bath "forces" are given by

$$G_1 = \sum_{i=1}^{N} \frac{\mathbf{p}_i^2}{m_i} - N_f k_B T \tag{9.19}$$

$$G_k = \frac{p_{\eta_{k-1}}^2}{Q_{k-1}} - k_B T \tag{9.20}$$

Equations 9.18 have the conserved energy

$$H' = H(\mathbf{p}, \mathbf{r}) + \sum_{k=1}^{M} \frac{p_{\eta_k}^2}{2Q_k} + N_f k_B T \eta_1 + k_B T \sum_{k=2}^{M} \eta_k \tag{9.21}$$

and a compressibility

$$\kappa(x) = -N_f \dot{\eta}_1 - \sum_{k=2}^{M} \dot{\eta}_k \tag{9.22}$$

The Liouville operator for the equations of motion is

$$\imath L = \imath L_1 + \imath L_2 + \imath L_T \tag{9.23}$$

where

$$\imath L_T = \sum_{k=1}^{M} \left[\frac{p_{\eta_k}}{Q_k} \frac{\partial}{\partial \eta_k} + G_k \frac{\partial}{\partial p_{\eta_k}} \right] - \sum_{i=1}^{N} \frac{p_{\eta_1}}{Q_1} \mathbf{p}_i \cdot \frac{\partial}{\partial \mathbf{p}_i} - \sum_{k=1}^{M-1} \frac{p_{\eta_{k+1}}}{Q_{k+1}} p_{\eta_k} \frac{\partial}{\partial p_{\eta_k}} \tag{9.24}$$

Using the generalization of the Trotter formula, the evolution operator can be written as

$$e^{\imath L \Delta t} = e^{\imath L_T \frac{\Delta t}{2}} e^{\imath L_1 \frac{\Delta t}{2}} e^{\imath L_2 \Delta t} e^{\imath L_1 \frac{\Delta t}{2}} e^{\imath L_T \frac{\Delta t}{2}} + \mathcal{O}\left(\Delta t^3\right) \tag{9.25}$$

The following factorization obeys Eq. **??**

$$
\begin{aligned}
&\exp\left[\imath L_T \frac{w_j \Delta t}{2n_c}\right] = \\
&\quad \exp\left[\frac{w_j \Delta t}{4n_c} G_M \frac{\partial}{\partial p_{\eta_M}}\right] \\
&\quad \prod_{k=M-1}^{1} \left\{ \exp\left[-\frac{w_j \Delta t}{8n_c} \frac{p_{\eta_{k+1}}}{Q_{k+1}} p_{\eta_k} \frac{\partial}{\partial p_{\eta_k}}\right] \exp\left[\frac{w_j \Delta t}{4n_c} G_k \frac{\partial}{\partial p_{\eta_k}}\right] \exp\left[-\frac{w_j \Delta t}{8n_c} \frac{p_{\eta_{k+1}}}{Q_{k+1}} p_{\eta_k} \frac{\partial}{\partial p_{\eta_k}}\right] \right\} \\
&\quad \prod_{i=1}^{N} \exp\left[-\frac{w_j \Delta t}{2n_c} \frac{p_{\eta_1}}{Q_1} \mathbf{p}_i \cdot \frac{\partial}{\partial \mathbf{p}_i}\right] \\
&\quad \prod_{k=1}^{M} \exp\left[-\frac{w_j \Delta t}{2n_c} \frac{p_{\eta_k}}{Q_k} \frac{\partial}{\partial \eta_k}\right] \\
&\quad \prod_{k=1}^{M-1} \left\{ \exp\left[-\frac{w_j \Delta t}{8n_c} \frac{p_{\eta_{k+1}}}{Q_{k+1}} p_{\eta_k} \frac{\partial}{\partial p_{\eta_k}}\right] \exp\left[\frac{w_j \Delta t}{4n_c} G_k \frac{\partial}{\partial p_{\eta_k}}\right] \exp\left[-\frac{w_j \Delta t}{8n_c} \frac{p_{\eta_{k+1}}}{Q_{k+1}} p_{\eta_k} \frac{\partial}{\partial p_{\eta_k}}\right] \right\} \\
&\quad \exp\left[\frac{w_j \Delta t}{4n_c} G_M \frac{\partial}{\partial p_{\eta_M}}\right]
\end{aligned}
\tag{9.26}
$$

9.1.3 NPT

$$\dot{\mathbf{r}}_i = \frac{\mathbf{p}_i}{m_i} + \frac{p_\epsilon}{W} \mathbf{r}_i \tag{9.27}$$

$$\dot{\mathbf{p}}_i = \mathbf{f}_i - \left(1 + \frac{d}{N_f}\right) \frac{p_\epsilon}{W} \mathbf{p}_i - \frac{p_{\eta_1}}{Q_1} \mathbf{p}_i \tag{9.28}$$

$$\dot{V} = \frac{dV}{W} p_\epsilon \tag{9.29}$$

$$\dot{p}_\epsilon = dV\,(P_{\text{int}} - P) + \frac{d}{N_f} \sum_{i=1}^{N} \frac{\mathbf{p}_i^2}{m_i} - \frac{p_{\xi_1}}{Q'_1} p_\epsilon \tag{9.30}$$

$$\begin{cases} \dot{\eta}_k &= \frac{p_{\eta_k}}{Q_k} \qquad k = 1, \ldots, M \\ \dot{p}_{\eta_k} &= G_k - \frac{p_{\eta_{k+1}}}{Q_{k+1}} p_{\eta_k} \\ \dot{p}_{\eta_M} &= G_M \end{cases} \tag{9.31}$$

$$\begin{cases} \dot{\xi}_k &= \frac{p_{\xi_k}}{Q'_k} \qquad k = 1, \ldots, M \\ \dot{p}_{\xi_k} &= G'_k - \frac{p_{\xi_{k+1}}}{Q'_{k+1}} p_{\xi_k} \\ \dot{p}_{\xi_M} &= G'_M \end{cases} \tag{9.32}$$

Two Nosé-Hoover chains are coupled to the system, one to the particles and the other to the barostat. This is important because the barostat tends to evolve on a much slower time scale than the particles. The heat-bath "forces" are given by

$$\begin{cases} G_1 &= \sum_{i=1}^{N} \frac{\mathbf{p}_i^2}{m_i} - N_f k_B T \\ G_k &= \frac{p_{\eta_{k-1}}^2}{Q_{k-1}} - k_B T \end{cases} \tag{9.33}$$

$$\begin{cases} G'_1 &= \frac{p_\epsilon^2}{W} - k_B T \\ G'_k &= \frac{p_{\eta_{k-1}}^2}{Q'_{k-1}} - k_B T \end{cases} \tag{9.34}$$

The equations have the conserved energy

$$H' = H\left(\mathbf{p},\mathbf{r}\right) + \frac{p_\epsilon^2}{2W} + PV + \sum_{k=1}^{M}\left(\frac{p_{\eta_k}^2}{2Q_k} + \frac{p_{\xi_k}^2}{2Q_k'}\right) + \left(N_f k_B T \eta_1 + k_B T \sum_{k=2}^{M} \eta_k\right) + k_B T \sum_{k=1}^{M} \xi_k \tag{9.35}$$

and a phase-space metric factor

$$\sqrt{g\left(x\right)} = \exp\left(N_f \eta_1 + \sum_{k=2}^{M} + \sum_{k=1}^{M} \eta_k\right) \tag{9.36}$$

Introducing the variables

$$\epsilon = \frac{1}{d}\ln\left(\frac{V}{V_0}\right) \tag{9.37}$$

$$\alpha = 1 + \frac{d}{N_f} \tag{9.38}$$

and writing the total Liouville operator as

$$\imath L = \imath L_1 + \imath L_2 + \imath L_{\epsilon,1} + \imath L_{\epsilon,2} + \imath L_{\text{T-barostat}} + \imath L_{\text{T-particles}} \tag{9.39}$$

where

$$\imath L_1 = \sum_{i=1}^{N}\left[\frac{\mathbf{p}_i}{m_i} + \frac{p_\epsilon}{W}\mathbf{r}_i\right]\cdot\frac{\partial}{\partial \mathbf{r}_i} \tag{9.40}$$

$$\imath L_2 = \sum_{i=1}^{N}\left[\mathbf{f}_i - \alpha\frac{p_\epsilon}{W}\mathbf{p}_i\right]\cdot\frac{\partial}{\partial \mathbf{p}_i} \tag{9.41}$$

$$\imath L_{\epsilon,1} = \frac{p_\epsilon}{W}\frac{\partial}{\partial \epsilon} \tag{9.42}$$

$$\imath L_{\epsilon,2} = G_\epsilon \frac{\partial}{\partial p_\epsilon} \tag{9.43}$$

$$\imath L_{\text{T-particles}} = \sum_{k=1}^{M}\left[\frac{p_{\eta_k}}{Q_k}\frac{\partial}{\partial \eta_k} + G_k\frac{\partial}{\partial p_{\eta_k}}\right] - \sum_{i=1}^{N}\frac{p_{\eta_1}}{Q_1}\mathbf{p}_i\cdot\frac{\partial}{\partial \mathbf{p}_i} - \sum_{k=1}^{M-1}\frac{p_{\eta_{k+1}}}{Q_{k+1}}p_{\eta_k}\frac{\partial}{\partial p_{\eta_k}} \tag{9.44}$$

$$\imath L_{\text{T-baro}} = \sum_{k=1}^{M}\left[\frac{p_{\xi_k}}{Q_k'}\frac{\partial}{\partial \xi_k} + G_k'\frac{\partial}{\partial p_{\xi_k}}\right] - \sum_{i=1}^{N}\frac{p_{\xi_1}}{Q_1'}\mathbf{p}_i\cdot\frac{\partial}{\partial \mathbf{p}_i} - \sum_{k=1}^{M-1}\frac{p_{\xi_{k+1}}}{Q_{k+1}'}p_{\xi_k}\frac{\partial}{\partial p_{\xi_k}} \tag{9.45}$$

and where

$$G_\epsilon = \alpha\sum_{i=1}^{N}\frac{\mathbf{p}_i^2}{m_i} + \sum_{i=1}^{N}\mathbf{r}_i\cdot\mathbf{f}_i - dV\frac{\partial U}{\partial V} - PV \tag{9.46}$$

The propagator is factorized as

$$\exp\left[\imath L \Delta t\right] =$$

$$\exp\left[\imath L_{\text{T-barostat}} \frac{\Delta t}{2}\right] \tag{9.47}$$

$$\exp\left[\imath L_{\text{T-particles}} \frac{\Delta t}{2}\right] \tag{9.48}$$

$$\exp\left[\imath L_{\epsilon,2} \frac{\Delta t}{2}\right] \tag{9.49}$$

$$\exp\left[\imath L_{2} \frac{\Delta t}{2}\right] \tag{9.50}$$

$$\exp\left[\imath L_{\epsilon,1} \Delta t\right] \tag{9.51}$$

$$\exp\left[\imath L_{1} \Delta t\right] \tag{9.52}$$

$$\exp\left[\imath L_{2} \frac{\Delta t}{2}\right] \tag{9.53}$$

$$\exp\left[\imath L_{\epsilon,2} \frac{\Delta t}{2}\right] \tag{9.54}$$

$$\exp\left[\imath L_{\text{T-particles}} \frac{\Delta t}{2}\right] \tag{9.55}$$

$$\exp\left[\imath L_{\text{T-barostat}} \frac{\Delta t}{2}\right] \tag{9.56}$$

9.1.4 μPT

Chapter 10

Pressure, Stress and Strain

10.1 Bend angle potentials

$$\frac{\partial U}{\partial \epsilon_{\alpha\beta}} = \sum_{ijk} \frac{\partial U_{ijk}}{\partial \cos\theta_{ijk}} \frac{\partial \cos\theta_{ijk}}{\partial \epsilon_{\alpha\beta}} \tag{10.1}$$

$$\frac{\partial U}{\partial \epsilon_{\alpha\beta}\partial \epsilon_{\mu\nu}} = \sum_{ijk} \left[\frac{\partial^2 U_{ijk}}{\partial \cos\theta_{ijk}^2} \frac{\partial \cos\theta_{ijk}}{\partial \epsilon_{\alpha\beta}} \frac{\partial \cos\theta_{ijk}}{\partial \epsilon_{\mu\nu}} + \frac{\partial U_{ijk}}{\partial \cos\theta_{ijk}} \frac{\partial^2 \cos\theta_{ijk}}{\partial \epsilon_{\alpha\beta}\partial \epsilon_{\mu\nu}} \right] \tag{10.2}$$

$$\frac{1}{\cos\theta_{ijk}} \frac{\partial^2 \cos\theta_{ijk}}{\partial \epsilon_{\alpha\beta}\partial \epsilon_{\mu\nu}} = \frac{\partial^2 \ln\cos\theta_{ijk}}{\partial \epsilon_{\alpha\beta}\partial \epsilon_{\mu\nu}} + \frac{\partial \ln\cos\theta_{ijk}}{\partial \epsilon_{\alpha\beta}} \frac{\partial \ln\cos\theta_{ijk}}{\partial \epsilon_{\mu\nu}} \tag{10.3}$$

where

$$\frac{\partial \cos\theta_{ijk}}{\partial \epsilon_{\alpha\beta}} = \cos\theta_{ijk} \frac{\partial \ln\cos\theta_{ijk}}{\partial \epsilon_{\alpha\beta}} \tag{10.4}$$

$$\frac{\partial \ln\cos\theta_{ijk}}{\partial \epsilon_{\alpha\beta}} = \frac{\tilde{u}_\alpha \tilde{v}_\beta + \tilde{v}_\alpha \tilde{u}_\beta}{\bar{u}\cdot\bar{v}} - \left(\frac{\tilde{u}_\alpha \tilde{u}_\beta}{u^2} + \frac{\tilde{v}_\alpha \tilde{v}_\beta}{v^2} \right) \tag{10.5}$$

$$\frac{\partial^2 \ln\cos\theta_{ijk}}{\partial \epsilon_{\alpha\beta}\partial \epsilon_{\mu\nu}} = -\frac{\left(\tilde{v}_\alpha \tilde{u}_\beta + \tilde{u}_\alpha \tilde{v}_\beta\right)\left(\tilde{v}_\mu \tilde{u}_\nu + \tilde{u}_\mu \tilde{v}_\nu\right)}{\left(\bar{u}\cdot\bar{v}\right)^2} + 2\left(\frac{\tilde{u}_\alpha \tilde{u}_\beta \tilde{u}_\mu \tilde{u}_\nu}{u^4} + \frac{\tilde{v}_\alpha \tilde{v}_\beta \tilde{v}_\mu \tilde{v}_\nu}{v^4} \right) \tag{10.6}$$

10.2 Torsion potentials

$$\frac{\partial U}{\partial \epsilon_{\alpha\beta}} = \sum_{ijkl} \frac{\partial U_{ijkl}}{\partial \cos\omega_{ijkl}} \frac{\partial \cos\omega_{ijkl}}{\partial \epsilon_{\alpha\beta}} \tag{10.7}$$

$$\frac{\partial U}{\partial \epsilon_{\alpha\beta}\partial \epsilon_{\mu\nu}} = \sum_{ijk} \left[\frac{\partial^2 U_{ijkl}}{\partial \cos\omega_{ijkl}^2} \frac{\partial \cos\omega_{ijkl}}{\partial \epsilon_{\alpha\beta}} \frac{\partial \cos\omega_{ijkl}}{\partial \epsilon_{\mu\nu}} + \frac{\partial U_{ijkl}}{\partial \cos\omega_{ijkl}} \frac{\partial^2 \cos\omega_{ijkl}}{\partial \epsilon_{\alpha\beta}\partial \epsilon_{\mu\nu}} \right] \tag{10.8}$$

with

$$\frac{\partial \ln\cos^2\omega_{ijkl}}{\partial \epsilon_{\alpha\beta}} = \frac{1}{\left(\bar{m}\cdot\bar{n}\right)^2} \frac{\partial \left(\bar{m}\cdot\bar{n}\right)^2}{\partial \epsilon_{\alpha\beta}} - \frac{1}{m^2}\frac{\partial m^2}{\partial \epsilon_{\alpha\beta}} - \frac{1}{n^2}\frac{\partial n^2}{\partial \epsilon_{\alpha\beta}} \tag{10.9}$$

We can use

$$\frac{\partial^2 \ln\left[f^2(x,y)\right]}{\partial x \partial y} = \frac{2}{f(x,y)}\frac{\partial^2 f(x,y)}{\partial x \partial y} - \frac{2}{f^2(x,y)}\frac{\partial f(x,y)}{\partial x}\frac{\partial f(x,y)}{\partial y} \tag{10.10}$$

$$\frac{\partial^2 f(x,y)}{\partial x \partial y} = \frac{f(x,y)}{2}\frac{\partial^2 \ln\left[f^2(x,y)\right]}{\partial x \partial y} + \frac{1}{f(x,y)}\frac{\partial f(x,y)}{\partial x}\frac{\partial f(x,y)}{\partial y} \tag{10.11}$$

So,

$$\frac{\partial^2 \cos\omega_{ijkl}}{\partial \epsilon_{\alpha\beta}\partial \epsilon_{\mu\nu}} = \frac{\cos\omega_{ijkl}}{2}\frac{\partial^2 \ln\cos^2\omega_{ijkl}}{\partial \epsilon_{\alpha\beta}\partial \epsilon_{\mu\nu}} + \frac{1}{\cos\omega_{ijkl}}\frac{\partial \cos\omega_{ijkl}}{\partial \epsilon_{\alpha\beta}}\frac{\partial \cos\omega_{ijkl}}{\partial \epsilon_{\mu\nu}} \tag{10.12}$$

and

$$\begin{aligned}\frac{\partial \left(\bar{m}\cdot\bar{n}\right)}{\partial \epsilon_{\alpha\beta}} =& \left(\bar{v}\cdot\bar{u}\right)\left(\tilde{u}_\alpha\tilde{w}_\beta + \tilde{w}_\alpha\tilde{u}_\alpha\right) + \left(\bar{u}\cdot\bar{w}\right)\left(\tilde{v}_\alpha\tilde{u}_\beta + \tilde{u}_\alpha\tilde{v}_\alpha\right) \\ & - \left(\bar{v}\cdot\bar{w}\right)\left(\tilde{u}_\alpha\tilde{u}_\beta + \tilde{u}_\alpha\tilde{u}_\alpha\right) - \left(\bar{u}\cdot\bar{u}\right)\left(\tilde{v}_\alpha\tilde{w}_\beta + \tilde{w}_\alpha\tilde{v}_\alpha\right)\end{aligned} \tag{10.13}$$

$$\frac{\partial \left(\bar{m}\cdot\bar{n}\right)^2}{\partial \epsilon_{\alpha\beta}} = 2\left(\bar{m}\cdot\bar{n}\right)\frac{\partial \left(\bar{m}\cdot\bar{n}\right)}{\partial \epsilon_{\alpha\beta}} \tag{10.14}$$

$$\frac{\partial^2 \left(\bar{m}\cdot\bar{n}\right)^2}{\partial \epsilon_{\alpha\beta}\partial \epsilon_{\mu\nu}} = 2\frac{\partial \left(\bar{m}\cdot\bar{n}\right)}{\partial \epsilon_{\alpha\beta}}\frac{\partial \left(\bar{m}\cdot\bar{n}\right)}{\partial \epsilon_{\mu\nu}} + 2\left(\bar{m}\cdot\bar{n}\right)\frac{\partial^2 \left(\bar{m}\cdot\bar{n}\right)}{\partial \epsilon_{\alpha\beta}\partial \epsilon_{\mu\nu}} \tag{10.15}$$

10.3 Stress

$$\sigma^B_{\alpha\beta} = \frac{1}{V}\sum_{i<j} f^\alpha_{ij} r^\beta_{ij} \tag{10.16}$$

$$\Theta_{\alpha\beta} = \delta_{\alpha\beta} - 2\frac{k_\alpha k_\beta}{\lambda^2} \tag{10.17}$$

$$\frac{1}{\lambda^2} = \frac{1}{4\alpha^2} + \frac{1}{k^2} \tag{10.18}$$

10.4 Strain

$$\sigma^B_{\alpha\beta} = -\frac{1}{V}\frac{\partial U}{\partial \epsilon_{\alpha\beta}} \tag{10.19}$$

$$C_{\alpha\beta\mu\nu} = \frac{1}{V}\frac{\partial^2 U}{\partial\epsilon_{\alpha\beta}\partial\epsilon_{\mu\nu}}\bigg|_{F=0} = -\frac{\partial\sigma^B_{\alpha\beta}}{\partial\epsilon_{\mu\nu}}\bigg|_{F=0} \tag{10.20}$$

10.4.1 Derivatives

General potential

$$\frac{\partial^2 U}{\partial\epsilon_{\alpha\beta}\partial\epsilon_{\mu\nu}} = \sum_{i<j}\sum_{k<l}\left(\frac{\partial^2 U}{\partial r_{ij}\partial r_{kl}} - \frac{1}{r_{ij}}\frac{\partial U}{\partial r_{ij}}\delta_{ik}\delta_{jl}\right)\frac{r^\alpha_{ij} r^\beta_{ij} r^\mu_{kl} r^\nu_{kl}}{r_{ij} r_{kl}} \tag{10.21}$$

$$\frac{\partial^2 U}{\partial\epsilon_{\alpha\beta}\partial\epsilon_{\mu\nu}} = \sum_{i,j} r_{i\beta} r_{j\nu}\mathcal{H}_{i\alpha,j\mu} + \delta_{\beta\nu}\delta_{ij} r_{i\mu}\frac{\partial U}{\partial r_{i\alpha}} + 2\delta_{\beta\nu}\sigma_{\alpha\mu} \tag{10.22}$$

$$\frac{\partial^2 U}{\partial\epsilon_{\alpha\beta}\partial\epsilon_{\mu\nu}} = \sum_{i,j} r_{i\beta} r_{j\nu}\mathcal{H}_{i\alpha,j\mu} + \delta_{\beta\nu}\sigma_{\alpha\mu} \tag{10.23}$$

Pair potentials

$$\frac{\partial U}{\partial\epsilon_{\alpha\beta}} = \sum_{i<j}\frac{1}{r_{ij}}\frac{\partial U}{\partial r_{ij}} r^\alpha_{ij} r^\beta_{ij} \tag{10.24}$$

$$\frac{\partial^2 U}{\partial\epsilon_{\alpha\beta}\partial\epsilon_{\mu\nu}} = \sum_{i<j}\frac{1}{r^2_{ij}}\left(\frac{\partial^2 U}{\partial r^2_{ij}} - \frac{1}{r_{ij}}\frac{\partial U}{\partial r_{ij}}\right) r^\alpha_{ij} r^\beta_{ij} r^\mu_{ij} r^\nu_{ij} \tag{10.25}$$

Ewald summation

$$\frac{\partial^2 U^{\text{real}}}{\partial\epsilon_{\alpha\beta}\partial\epsilon_{\mu\nu}} = \sum_{i<j} q_i q_j B_2 r_\alpha r_\beta r_\mu r_\nu \tag{10.26}$$

$$\frac{\partial^2 U^{\text{Fourier}}}{\partial\epsilon_{\alpha\beta}\partial\epsilon_{\mu\nu}} = \sum_{k\neq 0} A(k)\, SS^*\Omega_{\alpha\beta\mu\nu} \tag{10.27}$$

with

$$\Omega_{\alpha\beta\mu\nu} = \Theta_{\alpha\beta}\Theta_{\mu\nu} + (\delta_{\alpha\beta}\delta_{\mu\nu} + \delta_{\alpha\nu}\delta_{\beta\mu}) + 4\frac{1}{k^2}k_\alpha k_\beta k_\mu k_\nu - 2\frac{1}{\lambda^2}\left(k_\alpha\delta_{\beta\mu k_\nu + k_\beta\delta_{\alpha\mu}k_\nu} + k_\beta\delta_{\alpha\nu}k_\mu + k_\alpha\delta_{\alpha\nu}k_\mu\right) \tag{10.28}$$

Relation to the Hessian for pair potentials

$$\mathcal{H}_{i\alpha,j\beta} = \frac{\partial U}{\partial r_{i\alpha}\partial r_{j\beta}} \tag{10.29}$$

$$= \frac{1}{r_{ij}^2}\left(\frac{\partial^2 U\left(r_{ij}\right)}{\partial^2 r_{ij}} - \frac{1}{r_{ij}}\frac{\partial U\left(r_{ij}\right)}{\partial r_{ij}}\right) r_{ij}^{\alpha} r_{ij}^{\beta} + \delta_{\alpha\beta}\frac{1}{r_{ij}}\frac{\partial U\left(r_{ij}\right)}{\partial r_{ij}} \tag{10.30}$$

Chapter 11

Flexible Frameworks

11.1 Introduction

11.2 Monte Carlo moves

To obtain an isotherm using a flexible framework we need to perform Monte Carlo moves that change the internal conformation of the framework.

- Framework atom translation
  We randomly select a framework atom and give it a random displacement [1]. The maximum displacement is taken such that 50% of the moves are accepted. The acceptance rule is

$$\text{acc}\,(\text{old} \rightarrow \text{new}) = \min\left(1, e^{-\beta\left(U^{\text{new}} - U^{\text{old}}\right)}\right). \tag{11.1}$$

  Note that this scheme works for zeolites but is insufficient for MOFs because concerted movement, such as the rotation of the phenyl-group in a linker molecule, is not accounted for using individual atom moves.

  COTA has the option

```
ProbabilityFrameworkChangeMove   1.0
```

  where the parameter is a fraction of performing the move per cycle. The move tries to displace N randomly chosen framework atoms (where N is the amount of framework atoms). This implies the code runs slower for bigger frameworks, and a lower fraction speeds it up (but at the expense of proper sampling of the framework).

- Hybrid move
  We perform a small NVE molecular dynamics run of M time steps to obtain a new configuration [2]. The starting velocities are chosen from a Maxwell-Boltzmann distribution at the desired temperature. The acceptance rule is

$$\text{acc}\,(\text{old} \rightarrow \text{new}) = \min\left(1, e^{-\beta\left(U^{\text{new}} - U^{\text{old}}\right)}\right). \tag{11.2}$$

  Note that $\Delta U = U^{\text{new}} - U^{\text{old}}$ is the integration error. In relative units this error is quite small, but not in absolute units, i.e. with $M = 5$ and $\Delta t = 0.5$ fs for the CO_2 isotherm this results in an acceptance of approximately 80%.

  COTA has the option

```
ProbabilityHybridMCMDMove   1.0
NumberOfHybridMCMDSteps     5
```

Tip: to avoid a lengthy and costly initialization period, one can perform a long MD simulation at the desired temperature once, and use this configuration as the the start for subsequent adsorption runs. Equilibration of flexible framework is difficult because energy is only slowly redistributed properly amongst the various potentials like stretch, bend, torsions, etc. ("equipartition" is hard to reach).

11.3 Demontis et al. model [3, 4]

1. Bond stretching T-O, with T={Si,Al}

$$U_{\text{bond}} = \frac{1}{2} k_r \left(r - r_{\text{eq}}\right)^2 \tag{11.3}$$

with $k_r = 500.0$ kcal/mol/Å^2 and $r_{\text{eq}} = 1.605$ Å for Si-O, and with $k_r = 500.0$ kcal/mol/Å^2 and $r_{\text{eq}} = 1.76$ Å for Al-O.

2. Urey-Bradley O-T-O, with T={Si,Al}

$$U_{\text{bond}} = \frac{1}{2} k_r \left(r_{\text{Si-Si}} - r_{\text{eq}}\right)^2 \tag{11.4}$$

with $k_r = 103.0$ kcal/mol/Å^2 and $r_{\text{eq}} = 2.61786$ for O-(Si)-O, $k_r = 103.0$ kcal/mol/Å^2 and $r_{\text{eq}} = 2.87070$ for O-(Al)-O.

The Demontis model is restricted to the simulation of adsorbates without the use of charge. This includes alkanes as simulated in the united-atom approach. The advantage of the model is not its accuracy, but its effeciency.

COTA implements two versions of the Demontis model, "Demontis" and "DemontisModified", and can be use like

```
Charge None
Flexible yes
FrameworkDefinitions DemontisModified
```

The modified version takes the reference value r_{eq} from the crystal structure as proposed by Vlugt and schenk [1]. The original version does not converge to the crystal structure in the limit of low temperatures.

11.4 Nicholas et al. model [5]

1. Bond stretching O-Si

$$U_{\text{bond}} = \frac{1}{2} k_r \left(r - r_{\text{eq}}\right)^2 \tag{11.5}$$

with $k_r = 597.32$ kcal/mol/Å^2 and $r_{\text{eq}} = 1.61$ Å.

2. Bond bending O-Si-O

$$U_{\text{bend}} = \frac{1}{2} k_\theta \left(\theta - \theta_{\text{eq}}\right)^2 \tag{11.6}$$

with $k_\theta = 138.12$ kcal/mol/rad^2 and $\theta_{\text{eq}} = 109.5°$.

3. Bend bending Si-O-Si

$$U_{\text{bend}} = \frac{1}{2}k_{\theta 1}\left(\theta - \theta_{\text{eq}}\right)^2 - \frac{1}{2}k_{\theta 2}\left(\theta - \theta_{\text{eq}}\right)^3 + \frac{1}{2}k_{\theta 3}\left(\theta - \theta_{\text{eq}}\right)^4 \tag{11.7}$$

with $k_{\theta 1} = 10.85$ kcal/mol/rad^2, $k_{\theta 2} = 22.72$ kcal/mol/rad^3, $k_{\theta 3} = 13.26$ kcal/mol/rad^4, and $\theta_{\text{eq}} = 149.5°$.

4. Urey-Bradley Si-O-Si

$$U_{\text{bond}} = \frac{1}{2}k_r\left(r_{\text{Si-Si}} - r_{\text{eq}}\right)^2 \tag{11.8}$$

with $k_r = 54.6$ kcal/mol/Å^2 and $r_{\text{eq}} = 3.1261$.

5. Dihedral Si-O-Si-O

$$U_{\text{torsion}} = \frac{1}{2}k_\phi\left(1.0 + \cos\left(3\phi\right)\right) * S\left(\theta_{\text{Si-O-Si}}\right) \tag{11.9}$$

where $k_\phi = -0.7$ kcal/mol, and the screening term S is defined as:

$$\begin{aligned} &S\left(\theta\right) = \frac{\left(\theta_{\text{off}} - \theta\right)^2\left(\theta_{\text{off}} + 2\theta - 3\theta_{\text{on}}\right)}{\left(\theta_{\text{off}} - \theta_{\text{on}}\right)^3} \qquad 170° \leq \theta \leq 180° \qquad \theta_{\text{on}} = 170° \qquad \theta_{\text{off}} = 180° \\ &S\left(\theta\right) = 1 \qquad \theta < 170° \end{aligned} \tag{11.10}$$

6. Lennard-Jones

$$U_{\text{VDW}} = \frac{A}{r^{12}} - \frac{A}{r^6} \tag{11.11}$$

with $A = 9730572.0$ kcal Å^{12}/mol and $B = 2514.1821$ kcal Å^6/mol for Si, and $A = 158994.0156$ kcal Å^{12}/mol and $B = 192.8247$ kcal Å^6/mol for O, and $A_{\text{Si-O}} = \sqrt{A_{\text{Si}} * A_{\text{O}}}$ and $B_{\text{Si-O}} = \sqrt{B_{\text{Si}} * B_{\text{O}}}$ for Si-O. Rewritten in the Lennard-Jones potential the parameters read: $\epsilon = 0.162403$ kcal/mol ($\epsilon/k_B = 81.779$ K) and $\sigma = 3.96239$ Å for Si, $\epsilon = 0.0584635$ kcal/mol ($\epsilon/k_B = 29.43965$ K) and $\sigma = 3.06222$ Å for O, and $\epsilon = 0.0974406$ kcal/mol ($\epsilon/k_B = 49.0668$ K) and $\sigma = 3.48335$ Å for Si-O.

7. Electrostatics

$$U_{\text{elec.}} = \frac{q_i q_j}{\epsilon r} \tag{11.12}$$

with $\epsilon = 1$, $q = 1.1$ a.u. for Si and $q = -0.55$ a.u. for O.

The Nicholas model is suitable for all type of adsorbates, including ions and water (althought the model is only defined for siliceous zeolites). It reproduce the infra-red spectra well. Note that the model can be used with different charge-sets as they seem to have a neglible effect on the infra-red spectra. The model *excludes* 1-2 and 1-3 neighbors from both the Van der Waals and the charge interaction (the longer ranged interactions).

COTA implements two versions of the Nicholas model, "Nicholas" and "NicholasModified", and can be use like

```
Charge Ewald
Flexible yes
FrameworkDefinitions NicholasModified
RemoveBondNeighboursFromLongRangeInteraction     yes
RemoveBendNeighboursFromLongRangeInteraction     yes
RemoveTorsionNeighboursFromLongRangeInteraction  no
```

The modified version takes the reference value $r_{\rm eq}$ for the bond-stretching and Urey-Bradley term from the crystal structure. The reference angle for bending are untouched. The original version does not converge to the crystal structure in the limit of low temperatures.

11.5 IRMOF-1 model (Greathouse and Allendorf)

11.6 IRMOFs model (Dubbeldam et al.)

11.7 Core-shell models

11.8 Normal mode models

A useful physical test model is to expand the energy into a harmonic approximation and sample the framework flexibility using normal mode coordinates. These normal modes need to be computed only once at the minimum energy configuration, and the subsequent energy evaluations are very efficient.

$$U\left(\mathbf{r}\right) \approx U\left(\mathbf{r}_0\right) + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\Delta\mathbf{r}_i\left(\frac{\partial^2 U}{\partial\mathbf{r}_i\partial\mathbf{r}_j}\right)\Delta\mathbf{r}_j + \ldots \tag{11.13}$$

$$= U\left(\mathbf{r}_0\right) + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\Delta\mathbf{s}_i\left[\frac{1}{\sqrt{m_i}}\left(\frac{\partial^2 U}{\partial\mathbf{r}_i\partial\mathbf{r}_j}\right)\frac{1}{\sqrt{m_j}}\right]\Delta\mathbf{s}_j + \ldots \tag{11.14}$$

$$\tag{11.15}$$

The mass-weighted displacement $\Delta\mathbf{s} = \left(\sqrt{m_1}\Delta\mathbf{r}_1, \sqrt{m_2}\Delta\mathbf{r}_2, \ldots, \sqrt{m_1}\Delta\mathbf{r}_n\right)$. The quantity in $[\ldots]$ is the mass-weighted Hessian matrix of second derivatives of the energy with respect to position. It is the central object in normal-mode analysis. The key-part is to diagonalize this matrix to obtain the eigenvalues and eigenvectors of the matrix. The eigenvalues λ_i have units of force constants per mass and yield the square of the vibrational frequencies:

$$k_i = \left(2\pi c\nu_i\right)^2 \qquad \nu_i = \frac{\sqrt{k_i}}{2\pi c} \tag{11.16}$$

where c is the speed of light and ν_i is the vibrational frequency of the ith normal mode (typically given in cm^{-1}). The eigenvectors ($\mathbf{E}_i$) form an orthonormal basis, i.e. $\mathbf{E}_i \cdot \mathbf{E}_j = \delta_{ij}$. The energy per mode is given by

$$U\left(\mathbf{Q}_i\right) = \frac{1}{2}\lambda_i\sum_{j}^{3N}\left[\left(\mathbf{E}_i\right)_j\Delta\mathbf{s}_j\right]^2 \tag{11.17}$$

The total harmonic potential energy is given by

$$U\left(\mathbf{Q}\right) = U\left(\mathbf{r}_0\right) + \sum_{i=1}^{3N}U\left(\mathbf{Q}_i\right) \tag{11.18}$$

Both the Cartesian framework positions $\mathbf{r}_0$ and the mass weighted displacement $\Delta\mathbf{s}$ is stored. A new trial displacement can be chosen accoording to

$$\Delta\mathbf{s}^{\rm new} = \Delta\mathbf{s}^{\rm old} + \gamma\mathbf{E}_i \tag{11.19}$$

where $\mathbf{E}_i$ is a randomly chosen eigenvector and γ a scalar amplitude parameter, $\gamma \in [-\gamma_{\rm max}, \gamma_{\rm max}]$. The energy is then

$$U\left(\mathbf{Q}_i\right) = \frac{1}{2}\lambda_i\sum_{j}^{3N}\left[\left(\mathbf{E}_i\right)_j\left(\Delta\mathbf{s}^{\rm old} + \gamma\mathbf{E}_i\right)_j\right]^2 \tag{11.20}$$

Bibliography

[1] T. J. H. Vlugt and M. Schenk. Influence of framework flexibility on the adsorption properties of hydrocarbons in the zeolite silicalite. *J. Phys. Chem. B.*, 106:12757–12763, 2002.

[2] S. Chempath, L. A. Clark, and R. Q. Snurr. Two general methods for grand canonical ensemble simulation of molecules with internal flexibility. *J. Chem. Phys.*, 118(16):7635–7643, 2003.

[3] P. Demontis, G. B. Suffritti, S. Quartieri, E. S. Fois, and A. Gamba. Molecular dynamics studies on zeolites 2. a simple-model for silicates applied to anhydrous natrolite. *Zeolites*, 7(6):522–527, 1987.

[4] P. Demontis, G. B. Suffritti, S. Quartieri, E. S. Fois, and A. Gamba. Molecular dynamics on zeolites 3. dehydrated zeolite-a. *J. Phys. Chem.*, 92(4):867871, 1988.

[5] J. B. Nicholas, A. J. Hopfinger, F. R. Trouw, and L. E. Iton. Molecular modeling of zeolite structure .2. structure and dynamics of silica sodalite and silicate force-field. *J. Am. Chem. Soc.*, 113(13):4792–4800, 1991.

Part III

Applications

Chapter 12

Adsorption

12.1 Introduction

Wikipidia defines:

> Adsorption is a process that occurs when a gas or liquid or solute (called adsorbate) accumulates on the surface of a solid or more rarely a liquid (adsorbent), forming a molecular or atomic film (adsorbate). It is different from absorption, where a substance diffuses into a liquid or solid to form a "solution". The term sorption encompasses both processes, while desorption is the reverse process.

12.2 Equations of state (pressure → fugacity → chemical potential)

12.3 Absolute vs. excess adsorption

The conversion from absolute to excess adsorption is

$$n_{\text{excess}} = n_{\text{absolute}} - \frac{PV_{\text{pore}}}{zRT} \tag{12.1}$$

where P is the pressure of the bluk fluid phase, R the gas constant, T the temperature, z the compressibility in the bluk fluid phase, and V_{pore} the pore volume. For consistency with experiment, the pore volume has be determined once using a simulation of a single helium molecule at the reference conditions. Probing the framework with a nonadsorbing helium molecule with Widom particle insertion will give the *helium void fraction* ξ

$$\xi = \int e^{-\beta U}\, d\mathbf{r} \tag{12.2}$$

The pore volume is simply

$$V_{\text{pore}} = \xi V \tag{12.3}$$

Usually a reference temperature of $25°C$ (298 K) is chosen in experiment for the experimental determination of the helium void volume. Note that adsorption is always relative to this reference state even though the real pore volume changes as a function of temperature.

> COTA computes the compressibility z automatically using the Peng-Robinson equation of state. The helium void fraction can be specified in the input file as
>
> ```
> HeliumVoidFraction 0.818
> ```

and the reference crystal volume as

```
ExcessVolume 17237.5
```

If no reference crystal volume is specified the volume of the framework unit cell is used. The excess volume needs to explicitly specified for NPT simulations for flexible framwork using a restart-file, because then the starting volume is *not* the volume to be used to compute the reference pore volume.

12.4 Gas or liquid phase adsorption

12.5 Computing low-coverage properties

12.5.1 Henry coeffficients

12.5.2 Heats of adsorption

12.5.3 Entropies

12.6 Computing finite loading properties

12.6.1 Heats of adsorption

12.6.2 Isotherms

Chapter 13

Diffusion

13.1 Theoretical frameworks for describing diffusion in nanoporous materials

Many different diffusion coefficients can be defined for guest molecules in nanoporous materials, but it is useful to put them into two general classes: transport diffusivities and self-diffusivities. Transport diffusivity describes the transport of mass and the decay of density fluctuations in the system, while self-diffusion describes the diffusive motion of a single particle [1, 2, 3]. The transport diffusivities are measured under *nonequilibrium* conditions in which finite concentration gradients exists. They are determined by macroscopic measurement methods like gravimetric, volumetric, chromatographic, or frequency-response techniques. In other experiments, the self-diffusivity is measured under *equilibrium* conditions by microscopic techniques, such as quasi elastic neutron scattering (QENS) and pulsed field gradient (PFG) NMR. Using the QENS method it is possible to measure self- and transport diffusivities simultaneously [4]. Most of the early MD simulations in nanoporous materials concentrated on computing self-diffusivities [5]. With growing computer power, the emphasis has gradually shifted toward computing transport diffusivities, which are more relevant for technological applications.

From a phenomenological point of view, there are three different approaches to setting up the flux – driving force relationship for transport diffusion in nanoporous materials under nonequilibrium conditions. [6, 7]. The formulas presented here are taken from recent publications by Krishna and van Baten [8, 9].

1. Fick formulation
   In the Fick approach the fluxes $\mathbf{N}$ are taken to be linearly dependent on the gradients of the loadings ∇c_i of all species with the "constants" of proportionality being the Fick diffusivity matrix D^T,

$$(\mathbf{N}) = -\rho \left[D^T\right] (\nabla \boldsymbol{c}) \tag{13.1}$$

   where N_i is the molecular flux of species i in a mixture consisting of n components in units of molecules $\mathrm{m}^{-2}\ \mathrm{s}^{-1}$, ρ is the framework density in number of unit cells per m^3, c_i is the molecule loading of species i in molecules per unit cell, and D^T is the matrix of Fick diffusivities in $\mathrm{m}^2\ \mathrm{s}^{-1}$. The matrix is non-diagonal, and the cross-coefficients represent the coupling between species diffusion. The advantage of using D^T is that the Fick relations can be incorporated in the equipment design equations. However, the D^T show complex dependences on loading and mixture composition.

2. Onsager formulation
   In the Onsager approach the fluxes are postulated as linear functions of the chemical potential gradients $(\nabla \mu_i)$, with the proportionality constants being the Onsager coefficients L_{ij}.

$$(\mathbf{N}) = -\left[L\right] (\nabla \boldsymbol{\mu}) \tag{13.2}$$

The Onsager matrix $[L]$ is the square matrix of Onsager elements where $L_{ij}k_BT$ are in units of molecules m^{-2} s^{-1}. Here, k_B is the Boltzmann constant and T is the temperature. The Onsager reciprocal relations demand that the matrix $[L]$ is symmetric. It is sometimes convenient to define a modified Onsager matrix $[\Delta]$ [10]

$$(\mathbf{N}) = -\frac{\rho}{k_BT}[\mathbf{c}][\Delta](\nabla\boldsymbol{\mu}) \tag{13.3}$$

where $[\mathbf{c}]$ denotes the diagonal matrix having c_i as elements. The chemical potential gradients in Eq. 13.2 are the true driving force behind diffusive mass transport but may be expressed in terms of the occupancy gradients by introducing a matrix of thermodynamic factors Γ

$$\frac{c_i}{k_BT}\nabla\mu_i \equiv \sum_j \Gamma_{ij}\nabla c_j \tag{13.4}$$

$$\Gamma_{ij} = \frac{c_i}{c_j}\frac{\partial \ln f_i}{\partial \ln c_j} \tag{13.5}$$

where f_i denotes the fugacity of component i in the bulk fluid phase. The thermodynamic factor can be obtained from the sorption isotherms after differentiation.

3. Maxwell-Stefan formulation
The Maxwell-Stefan formulation balances diffusive and drag forces, while Fick and Onsager postulate phenomenological flux expressions. Using the Maxwell-Stefan theory, the following expression can be derived for diffusion of species i in a nanoporous material:

$$-\rho\frac{\theta_i}{k_BT}\nabla\mu_i = \sum_{\substack{j=1\\ j\neq i}}^{n}\frac{c_jN_i - c_iN_j}{c_{i,\mathrm{sat}}c_{j,\mathrm{sat}}D_{ij}^{\mathrm{MS}}} + \frac{N_i}{c_{i,\mathrm{sat}}D_i^{\mathrm{MS}}} \tag{13.6}$$

where $\theta_i \equiv c_i/c_{i,\mathrm{sat}}$ is the dimensionless fractional occupancy of component i, $\nabla\mu_i$ is the force acting on species i tending to move it within the framework, D_i^{MS} is the Maxwell-Stefan diffusivity describing the interaction between component i and the confinement, and D_{ij}^{MS} are the binary exchange Maxwell-Stefan diffusivities describing the correlation effects between components i and j within the framework structure. The correlations are loading and topology dependent. Lower values imply a stronger correlation effect, and for $D_{ij}^{\mathrm{MS}} \to \infty$ correlation effects vanish. For a single component system, D_i^{MS} is sometimes called the "corrected" diffusivity D^C [1]. The Maxwell-Stefan diffusivities can be recast into the Fickian formulation by

$$(N) = -\rho[B]^{-1}[\Gamma](\nabla\mathbf{c}) \tag{13.7}$$

where the elements of the matrix $[B]$ are

$$B_{ii} = \frac{1}{D_i^{\mathrm{MS}}} + \sum_{\substack{j=1\\ j\neq i}}^{n}\frac{\theta_j}{D_{ij}^{\mathrm{MS}}} \tag{13.8}$$

$$B_{ij} = -\frac{c_{i,\mathrm{sat}}}{c_{j,\mathrm{sat}}}\frac{\theta_i}{D_{ij}^{\mathrm{MS}}} \quad (i \neq j) \tag{13.9}$$

and we note that

$$[\Delta] = [B]^{-1} \tag{13.10}$$

Using the Maxwell-Stefan formulation, a simple expression can be obtained for the self diffusion D_i^S of species i in a mixture [11, 12]

$$D_i^S = \frac{1}{\frac{1}{D_i^{\mathrm{MS}}} + \sum_{j=1}^{n}\frac{\theta_j}{D_{ij}^{\mathrm{MS}}}} \tag{13.11}$$

This equation can also be viewed as a definition of D_{ii}^{MS}.

The three formulations are strictly equivalent, and all three viewpoints are needed for different purposes. The Fickian formulation is convenient for engineering calculations because the flux equations are written in terms of concentration. The Onsager approach has a strong basis in statistical mechanics, and the Onsager coefficients can be calculated from MD simulations, as described below. One of the chief advantages of the Maxwell-Stefan formulation is that it provides a useful framework for models to predict mixture diffusion based on information from the pure components, especially from D_{ii}^{MS}.

The self-, corrected, and transport diffusivities are equal only in the limit of zero loading. In general one finds that [3]

$$D^T = D^C = D^S \qquad \text{for } c = 0 \tag{13.12}$$

$$D^T \neq D^C > D^S \qquad \text{for } c > 0 \tag{13.13}$$

The self-diffusivities are more strongly influenced by correlation effects (kinetic and vacancy correlations) than the transport and Maxwell-Stefan diffusivities. In chemical engineering applications it is sometimes assumed that the corrected diffusion coefficient of particles under confinement is independent of loading [4, 13, 1]. The assumption that the corrected diffusivity is loading-independent is sometimes used to compare different data sets and relate diffusivities measured by microscopic and macroscopic techniques. For a more detailed discussion, see Ref. [14].

13.2 Calculating diffusion coefficients from molecular dynamics simulations

13.2.1 Self-diffusion

In molecular dynamics (MD) simulations [15, 16, 17], successive configurations of the system are generated by integrating Newton's laws of motion, which then yields a trajectory that describes the positions, velocities and accelerations of the particles as they vary with time. The self-diffusivity describes the motion of an individual particle. In an equilibrium molecular dynamics (EMD) simulation the self-diffusion coefficient D_α^S of component α is computed by taking the slope of the mean-squared displacement (MSD) at long times

$$D_\alpha^S = \frac{1}{2dN_\alpha} \lim_{t\to\infty} \frac{d}{d\,t} \left\langle \sum_{i=1}^{N_\alpha} \left(r_i^\alpha(t) - r_i^\alpha(0)\right)^2 \right\rangle \tag{13.14}$$

where N_α is the number of molecules of component α, d is the spatial dimension of the system, t is the time, and r_i^α is the center-of-mass of molecule i of component α. Equivalently, D_α^S is given by the time integral of the velocity autocorrelation function

$$D_\alpha^S = \frac{1}{dN_\alpha} \int_0^\infty \left\langle \sum_{i=1}^{N_\alpha} v_i^\alpha(t)\, v_i^\alpha(0) \right\rangle dt \tag{13.15}$$

where v_i^α is the center-of-mass-velocity of molecule i of component α. Eq. 13.14 is known as the Einstein equation and Eq. 13.15 is often referred to as the Green-Kubo relation. A separation of time scales occurs for interacting particles. At very short time scales the MSD has a quadratic dependence on time (a slope of two on a log-log plot). This is known as the ballistic regime, where particles on average do not yet collide. In nanoporous materials, an intermediate regime starts when particles are colliding, but only with a subset of the other particles. This is due to the confinement. Only when particles are able to escape the local environment and explore the full periodic lattice is the diffusional regime reached. In the diffusive regime, the mean-squared displacement bends over to attain a different slope and becomes linear with time (a slope of unity on a log-log plot). It is the long-time diffusion coefficient that is of general interest here.

The Einstein and Green-Kubo equations given above can be applied to each x, y, z-direction individually (when the dimension of the system is taken in each case as $d = 1$), applied to the two dimensional case $d = 2$,

or applied to the three dimensional system $d = 3$. In this case the directionally averaged diffusion coefficient is given by

$$D = \frac{D_x + D_y + D_z}{3} \tag{13.16}$$

13.2.2 Single-component transport diffusion

For a single adsorbed component, the transport diffusion coefficient D^T is given by

$$D^T = \frac{\Gamma}{2dN} \lim_{t\to\infty} \frac{d}{d\,t} \left\langle \left(\sum_{i=1}^{N} \left(r_i\left(t\right) - r_i\left(0\right) \right) \right)^2 \right\rangle \tag{13.17}$$

or

$$D^T = \frac{\Gamma}{dN} \int_0^\infty \left\langle \left(\sum_{i=1}^{N} v_i(t) \right) \left(\sum_{i=1}^{N} v_i(0) \right) \right\rangle dt \tag{13.18}$$

The thermodynamic factor Γ is

$$\Gamma = \left(\frac{\partial \ln f}{\partial \ln c} \right)_T \tag{13.19}$$

and can be obtained from the adsorption isotherm. Isotherms can be obtained experimentally or predicted from GCMC simulations. Alternatively Γ can be computed during a GCMC simulation as [18]

$$\Gamma = \frac{\langle N \rangle}{\langle N^2 \rangle - \langle N \rangle^2} \tag{13.20}$$

The omission of the thermodynamic factor in Eqs. 13.17 and 13.18, leads to the corrected diffusivity D^C

$$D^C = \frac{1}{2dN} \lim_{t\to\infty} \frac{d}{d\,t} \left\langle \left(\sum_{i=1}^{N} \left(r_i\left(t\right) - r_i\left(0\right) \right) \right)^2 \right\rangle \tag{13.21}$$

and

$$D^C = \frac{1}{dN} \int_0^\infty \left\langle \left(\sum_{i=1}^{N} v_i(t) \right) \left(\sum_{i=1}^{N} v_i(0) \right) \right\rangle dt \tag{13.22}$$

13.2.3 Multi-component transport diffusion (mixtures)

For mixtures, the Onsager Δ elements for components α and β can be computed from equilibrium MD simulations using the Einstein form

$$\begin{aligned} \Delta_{\alpha\beta} = \frac{1}{2dN_\alpha} \lim_{t\to\infty} \frac{1}{t} \Bigg\langle & \left(\sum_{i=1}^{N_\alpha} \left(\mathbf{r}_i^\alpha(t) - \mathbf{r}_i^\alpha(0) \right) \right) \\ & \left(\sum_{i=1}^{N_\beta} \left(\mathbf{r}_i^\beta(t) - \mathbf{r}_i^\beta(0) \right) \right) \Bigg\rangle \end{aligned} \tag{13.23}$$

or

$$\Delta_{\alpha\beta} = \frac{1}{dN_\alpha} \int_0^\infty \left\langle \left(\sum_{i=1}^{N_\alpha} v_i^\alpha(t) \right) \left(\sum_{i=1}^{N_\beta} v_i^\beta(0) \right) \right\rangle dt \tag{13.24}$$

$$= \frac{1}{dN_\alpha} \int_0^\infty \left\langle \sum_{i=1}^{N_\alpha} \sum_{j=1}^{N_\beta} v_i^\alpha(t) v_j^\beta(0) \right\rangle dt \tag{13.25}$$

Using Eq. 13.10, the elements of $[B] = [\Delta]^{-1}$ can be obtained by matrix inversion. The Maxwell-Stefan diffusivities D_i^{MS} and D_{ij}^{MS} for an n-component system are then given by [9]

$$D_i^{\mathrm{MS}} = \frac{1}{B_{ii} - \sum_{\substack{j=1 \\ j \neq i}}^{n} \frac{\theta_j}{D_{ij}^{\mathrm{MS}}}} \tag{13.26}$$

$$D_{ij}^{\mathrm{MS}} = -\frac{c_{i,\mathrm{sat}}}{c_{j,\mathrm{sat}}} \frac{\theta_i}{B_{ij}} \tag{13.27}$$

Equations relating $L_{\alpha\beta}$ (and thus $\Delta_{\alpha\beta}$) to the Fickian diffusion coefficients can also be derived [2].

13.2.4 Frame of reference (removing drift)

Center-of-mass motion for single components can be due to an external field, e.g. the confinement, or due to a lack of linear momentum conservation, e.g. individual components in a mixture. Some thermo- and barostat methods are known to violate energy conservation for single components and also momentum conservation in mixtures, causing center-of-mass drift. It is standard practice to remove the center-of-mass motion from the system at the beginning of an EMD simulation. However, the system may still show center-of-mass drift, and it is desirable to compute the self-diffusivity relative to the center-of-mass motion. To achieve this, self-diffusivities can be computed by

$$D_\alpha^S = \frac{1}{2dN_\alpha} \lim_{t\to\infty} \frac{d}{d\,t} \left\langle \sum_{i=1}^{N_\alpha} \left(\Delta r_i^\alpha\left(t\right) - \Delta r_i^\alpha\left(0\right)\right)^2 \right\rangle \tag{13.28}$$

and

$$D_\alpha^S = \frac{1}{dN_\alpha} \int_0^\infty \left\langle \sum_{i=1}^{N_\alpha} \Delta v_i^\alpha\left(t\right) \Delta v_i^\alpha\left(0\right) \right\rangle dt \tag{13.29}$$

where

$$\Delta r_i^\alpha(t) = r_i^\alpha\left(t\right) - r^{\mathrm{com}}\left(t\right) \tag{13.30}$$

$$\Delta v_i^\alpha(t) = v_i^\alpha\left(t\right) - v^{\mathrm{com}}\left(t\right) \tag{13.31}$$

Here, r^{com} and v^{com} are the displacement and velocity of the center-of-mass of the system, respectively, including the displacement and velocity of the framework itself if the framework atoms are allowed to move. In this case, the drift of the framework should also be subtracted for the corrected and transport diffusivities, Eqs. 13.21, 13.22, 13.23, 13.24, and 13.25. Note that for diffusion in disconnected channel systems that are unable to exchange momentum, the equations apply individually, i.e. the center-of-mass drift is to be computed per channel system.

13.3 Theoretical modeling

13.3.1 The Langmuir gas

13.3.2 The Reed-Ehrlich model

Bibliography

[1] J. Kärger and D. M. Ruthven. *Diffusion in Zeolites and Other Microporous Solids*. John Wiley & Sons, Inc., New York, 1992.

[2] D. N. Theodorou, R. Q. Snurr, and A. T. Bell. *Comprehensive Supramolecular Chemistry*, volume 7. Pergamon Oxford, Oxford, 1996.

[3] T. Ala-Nissila, R. Ferrando, and S. C. Ying. Collective and single particle diffusion on surfaces. *Advances in Physics*, 51(3):949–1078, 2002.

[4] H. Jobic, J. Karger, and M. Bee. Simultaneous measurement of self- and transport diffusivities in zeolites. *Phys. Rev. Lett.*, 82(21):4260–4263, 1999.

[5] P. Demontis and G. B. Suffritti. Structure and dynamics of zeolites investigated by molecular dynamics. *Chem. Rev.*, 97(8):2845–2878, 1997.

[6] J. A. Wesselingh and R. Krishna. *Mass Transfer in Multicomponent Mixtures*. Delft University Press, Delft, 2000.

[7] R. Krishna and R. Baur. Modelling issues in zeolite based separation processes. *Sep. Purif. Techn.*, 33(3):213–254, 2003.

[8] R. Krishna and J. M. van Baten. Describing binary mixture diffusion in carbon nanotubes with the maxwell-stefan equations. an investigation using molecular dynamics simulations. *Ind. Eng. Chem. Res.*, 45(6):2084–2093, 2006.

[9] R. Krishna and J. M. van Baten. Diffusion of alkane mixtures in zeolites: Validating the maxwell-stefan formulation using md simulations. *J. Phys. Chem. B.*, 109(13):6386–6396, 2005.

[10] J. M. van Baten and R. Krishna. Entropy effects in adsorption and diffusion of alkane isomers in mordenite: An investigation using cbmc and md simulations. *Microporous Mesoporous Mater.*, 84(1-3):179–191, 2005.

[11] D. Paschek and R. Krishna. Inter-relation between self- and jump-diffusivities in zeolites. *Chem. Phys. Lett.*, 333(3-4):278–284, 2001.

[12] R. Krishna and D. Paschek. Self-diffusivies in multicomponent mixtures in zeolites. *Phys. Chem. Chem. Phys.*, 4:1891–1898, 2002.

[13] S. M. Auerbach. Theory and simulation of jump dynamics, diffusion and phase equilibrium in nanopores. *International reviews in physical chemistry*, 19(2):155–198, 2000.

[14] A. I. Skoulidas and D. S. Sholl. Direct tests of the darken approximation for molecular diffusion in zeolites using equilibrium molecular dynamics. *J. Phys. Chem. B.*, 105(16):3151–3154, 2001.

[15] M. P. Allen and D. J. Tildesley. *Computer simulation of liquids*. Clarendon Press, Oxford, 1987.

[16] D. C. Rapaport. *The art of molecular dynamics simulation 2nd edition*. Cambridge University Press, Cambridge, 2004.

[17] D. Frenkel and B. Smit. *Understanding molecular simulation 2nd edition*. Academic Press, London, UK, 2002.

[18] D. A. Reed and G. Ehrlich. Surface diffusivity and the time correlation of concentration fluctuations. *Surf. Sci.*, 105(2-3):603–628, 1981.

[19] K. Kremer and G. S. Grest. Dynamics of entangled linear polymer melts - a molecular-dynamics simulation. *J. Chem. Phys.*, 92(8):5057–5086, 1990.

[20] S. Jakobtorweihen, C. P. Lowe, F. J. Keil, and B. Smit. A novel algorithm to model the influence of host lattice flexibility in molecular dynamics simulations: Loading dependence of self-diffusion in carbon nanotubes. *J. Chem. Phys.*, 124:154706, 2006.

[21] S. Jakobtorweihen, M. G. Verbeek, C. P. Lowe, F. J. Keil, and B. Smit. Understanding the loading dependence of self-diffusion in carbon nanotubes. *Phys. Rev. Lett.*, 95(4):Art. No. 044501, 2005.

[22] A. I. Skoulidas, D. M. Ackerman, J. K. Johnson, and D. S. Sholl. Rapid transport of gases in carbon nanotubes. *Phys. Rev. Lett.*, 89:Art. No. 185901, 2002.

[23] H. B. Chen and D. S. Sholl. Rapid diffusion of ch_4/h_2 mixtures in single-walk carbon nanotubes. *J. Am. Chem. Soc.*, 126(25):7778–7779, 2004.

[24] H. B. Chen, J. K. Johnson, and D. S. Sholl. Transport diffusion of gases is rapid in flexible carbon nanotubes. *J. Phys. Chem. B.*, 110(5):1971–1975, 2006.

[25] G. Arora and N. J. Wagner. Adsorption and diffusion of molecular nitrogen in single wall carbon nanotubes. *Langmuir*, 20(15):6268–6277, 2004.

Chapter 14

Minimization and Saddle-Points

14.1 Introduction

There are a variety of techniques for energy minimizations, both local and global techniques. Whether one prefers a local or global technique depends on one needs. Usually, one would like to use a local technique. For instance, otherwise a minimization of a siliceous zeolite would always end up as an α-quartz. Thus, the most common use is to find the "global" lowest energy for only a *subset* of potential energy minima's. There are roughly the following sets of algorithms available for local minimization:

1. Methods which only use the energy,
   Example: Simplex method.

2. Methods which use the energy and first derivatives,
   Examples: steepest descent, conjugent gradient, Snyman's method.

3. Methods which use the energy, first derivatives, and approximate second derivatives.
   Examples: Quasi-Newton methods.

4. Methods which use the energy, first derivatives, and exact second derivatives.
   Examples: Newton-Rapson method.

5. Methods which use the energy, first derivatives, second derivatives, and the eigenvalues of eigenvectors of the Hessian matrix
   Example: Baker's method (mode following method).

The potential energy surface of a (periodic) system can be Taylor-expanded. An example coordinate system could consist of $3N$ fractional positions and the nine lattice vector component variations, $\delta x = \{\delta \mathbf{s}, \delta \mathbf{h}\}$ leading to

$$U\left(\mathbf{x}+\delta\mathbf{x}\right) = U\left(\mathbf{x}_0\right) + \left(\frac{\partial U}{\partial x}\right)^T \delta\mathbf{x} + \frac{1}{2}\delta\mathbf{x}^T \frac{\partial U}{\partial x_\alpha \partial x_\beta}\delta\mathbf{x} + \ldots \tag{14.1}$$

The first derivative is minus the force

$$\frac{\partial U}{\partial x} \equiv \mathbf{h} = -\mathbf{f} \tag{14.2}$$

The elements of the column vector $\mathbf{f}$ are either the forces on the particles $\mathbf{f}_i^\alpha = -dU/ds_i^\alpha$, or minus the lattice vector derivatives of the total energy $\mathbf{f}_\alpha^\beta = -dU/dh_{\alpha\beta}$. The latter are related to the stress tensor elements

$$\sigma_{\alpha\beta} = -\frac{1}{V}\frac{dU}{d\epsilon_{\alpha\beta}} \tag{14.3}$$

using the relationship

$$\frac{dE}{dh_{\alpha\beta}} = -V \sum_{\gamma} \sigma_{\alpha\gamma} \left(h^T\right)^{-1}_{\gamma\beta} \tag{14.4}$$

The generalized Hessian matrix is of size $(3N+9) \times (3N+9)$ and build up from several blocks:

- the second derivative of the total energy with respect to particle position $\frac{\partial U}{\partial r_i^\alpha \partial r_j^\beta} \equiv \mathcal{H}$
  The Hessian matrix $\mathcal{H}$ is symmetric and of size $3N \times 3N$.
- the second derivatives of the total energy with respect to the lattice vectors $\frac{\partial U}{\partial h_{\alpha\beta} \partial h_{\mu\nu}}$
  The matrix is symmetric and of size 9×9.
- the crossterms $\frac{\partial U}{\partial r_i^\alpha \partial h_{\mu\nu}}$ and $\frac{\partial U}{\partial h_{\alpha\beta} \partial r_i^\mu}$
  The two matrices are the transpose of eachother and of size $3N \times 9$ and $9 \times 3N$ respectively.

However, there are several pitfalls when using this set of variables. First of all, using the lattice vectors as variables leads to very slow convergence. Secondly, only 6 elements of the 3×3 cell matrix are independent. Thirdly, one would like to avoid overall rotation of the system. COTA therefore uses the 6 independent strain elements $(\epsilon_{xx}, \epsilon_{xy}, \epsilon_{xz}, \epsilon_{yy}, \epsilon_{yz}, \epsilon_{zz})$ operating on the cell, and the Cartesian positions as variables. The new box $\mathbf{h}'$ can be computed from the original box $\mathbf{h}_0$ ($\mathbf{h}$ at the start of the simulation) by using $\mathbf{h}' = \epsilon \mathbf{h}_0$ with

$$\epsilon = \begin{pmatrix} 1+\epsilon_{xx} & 0 & 0 \\ \epsilon_{xy} & 1+\epsilon_{yy} & 0 \\ \epsilon_{xz} & \epsilon_{yz} & 1+\epsilon_{zz} \end{pmatrix} \qquad \mathbf{h} = \begin{pmatrix} h_{xx} & 0 & 0 \\ h_{xy} & h_{yy} & 0 \\ h_{xz} & h_{yz} & h_{zz} \end{pmatrix} \tag{14.5}$$

The Cartesian coordinate are more convenient for the energy evaluation routines. A linear transfomation relates the position of a particle r_i' within the cell after a homogeneous transformation is applied to its original position r_i in the reference state of the body, i.e. the position at the previous iteration:

$$r_i' = \mathcal{M} r_i \qquad \mathcal{M} = h h_0^{-1} \tag{14.6}$$

The keyword 'Minimization' denotes the minimization in COTA.

```
SimulationType Minimization
```

The generalized first derivative and Hessian matrix used in COTA has the structure shown in Table 14.1. During the minimization the system-configurations are written to the movie directory and the restart-file is updated.

14.2 Minimization

14.2.1 Steepest descent

14.2.2 Conjugent gradient

14.2.3 Snyman's method

Snyman's method is reliable, comparable to conjugate gradient techniques, and very efficient for large minimization problems. Snyman transforms the problem of finding a local solution of the minimum value of a twice continuously differentiable objective function $F(\mathbf{r})$ to an associated dynamic problem: solve the equations of motions $\ddot{\mathbf{r}}(t) = -\nabla F(\mathbf{r}(t))$ subjected to initial conditions $\mathbf{r}(0) = \mathbf{r}_0, \dot{\mathbf{r}}(0) = \mathbf{v}_0 = \mathbf{0}$. In integrating the equations, a simple leap-frog scheme was used, combined with an interfering strategy which ensures that the potential energy and thus F is systematically reduced. The method is originally defined as a leap-frog integration scheme:

$$\mathbf{h}^T = \left\{ \frac{\partial U}{\partial r_1^x}, \frac{\partial U}{\partial r_1^y}, \frac{\partial U}{\partial r_1^z}, \frac{\partial U}{\partial r_2^x}, \frac{\partial U}{\partial r_2^y}, \frac{\partial U}{\partial r_2^z}, \ldots, \frac{\partial U}{\partial r_n^x}, \frac{\partial U}{\partial r_n^y}, \frac{\partial U}{\partial r_n^z}, \frac{\partial U}{\partial \epsilon_{xx}}, \frac{\partial U}{\partial \epsilon_{xy}}, \frac{\partial U}{\partial \epsilon_{xz}}, \frac{\partial U}{\partial \epsilon_{yy}}, \frac{\partial U}{\partial \epsilon_{yz}}, \frac{\partial U}{\partial \epsilon_{zz}} \right\}$$

$$\begin{pmatrix} \begin{pmatrix} \frac{\partial U}{\partial r_1^x \partial r_1^x} & \frac{\partial U}{\partial r_1^y \partial r_1^x} & \frac{\partial U}{\partial r_1^z \partial r_1^x} & \cdots & \cdots & \cdots & \frac{\partial U}{\partial r_n^x \partial r_1^x} & \frac{\partial U}{\partial r_n^y \partial r_1^x} & \frac{\partial U}{\partial r_n^z \partial r_1^x} \\ \frac{\partial U}{\partial r_1^x \partial r_1^y} & \frac{\partial U}{\partial r_1^y \partial r_1^y} & \frac{\partial U}{\partial r_1^z \partial r_1^y} & \cdots & \cdots & \cdots & \frac{\partial U}{\partial r_n^x \partial r_1^x} & \frac{\partial U}{\partial r_n^y \partial r_1^x} & \frac{\partial U}{\partial r_n^z \partial r_1^x} \\ \frac{\partial U}{\partial r_1^x \partial r_1^z} & \frac{\partial U}{\partial r_1^y \partial r_1^z} & \frac{\partial U}{\partial r_1^z \partial r_1^z} & \cdots & \cdots & \cdots & \frac{\partial U}{\partial r_n^x \partial r_1^x} & \frac{\partial U}{\partial r_n^y \partial r_1^x} & \frac{\partial U}{\partial r_n^z \partial r_1^x} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \frac{\partial U}{\partial r_1^x \partial r_n^x} & \frac{\partial U}{\partial r_1^y \partial r_n^x} & \frac{\partial U}{\partial r_1^z \partial r_n^x} & \cdots & \cdots & \cdots & \frac{\partial U}{\partial r_n^x \partial r_n^x} & \frac{\partial U}{\partial r_n^y \partial r_n^x} & \frac{\partial U}{\partial r_n^z \partial r_n^x} \\ \frac{\partial U}{\partial r_1^x \partial r_n^y} & \frac{\partial U}{\partial r_1^y \partial r_n^y} & \frac{\partial U}{\partial r_1^z \partial r_n^y} & \cdots & \cdots & \cdots & \frac{\partial U}{\partial r_n^x \partial r_n^x} & \frac{\partial U}{\partial r_n^y \partial r_n^x} & \frac{\partial U}{\partial r_n^z \partial r_n^x} \\ \frac{\partial U}{\partial r_1^x \partial r_n^z} & \frac{\partial U}{\partial r_1^y \partial r_n^z} & \frac{\partial U}{\partial r_1^z \partial r_n^z} & \cdots & \cdots & \cdots & \frac{\partial U}{\partial r_n^x \partial r_n^x} & \frac{\partial U}{\partial r_n^y \partial r_n^x} & \frac{\partial U}{\partial r_n^z \partial r_1^x} \end{pmatrix} & \begin{pmatrix} \frac{\partial U}{\partial r_1^x \partial \epsilon_{xx}} & \frac{\partial U}{\partial r_1^x \partial \epsilon_{xy}} & \frac{\partial U}{\partial r_1^x \partial \epsilon_{xz}} & \frac{\partial U}{\partial r_1^x \partial \epsilon_{yy}} & \frac{\partial U}{\partial r_1^x \partial \epsilon_{yz}} & \frac{\partial U}{\partial r_1^x \partial \epsilon_{zz}} \\ \frac{\partial U}{\partial r_1^y \partial \epsilon_{xx}} & \frac{\partial U}{\partial r_1^y \partial \epsilon_{xy}} & \frac{\partial U}{\partial r_1^y \partial \epsilon_{xz}} & \frac{\partial U}{\partial r_1^y \partial \epsilon_{yy}} & \frac{\partial U}{\partial r_1^y \partial \epsilon_{yz}} & \frac{\partial U}{\partial r_1^y \partial \epsilon_{zz}} \\ \frac{\partial U}{\partial r_1^z \partial \epsilon_{xx}} & \frac{\partial U}{\partial r_1^z \partial \epsilon_{xy}} & \frac{\partial U}{\partial r_1^z \partial \epsilon_{xz}} & \frac{\partial U}{\partial r_1^z \partial \epsilon_{yy}} & \frac{\partial U}{\partial r_1^z \partial \epsilon_{yz}} & \frac{\partial U}{\partial r_1^z \partial \epsilon_{zz}} \\ \cdots & & & & & \\ \cdots & & & & & \\ \cdots & & & & & \\ \frac{\partial U}{\partial r_n^x \partial \epsilon_{xx}} & \frac{\partial U}{\partial r_n^x \partial \epsilon_{xy}} & \frac{\partial U}{\partial r_n^x \partial \epsilon_{xz}} & \frac{\partial U}{\partial r_n^x \partial \epsilon_{yy}} & \frac{\partial U}{\partial r_n^x \partial \epsilon_{yz}} & \frac{\partial U}{\partial r_n^x \partial \epsilon_{zz}} \\ \frac{\partial U}{\partial r_n^y \partial \epsilon_{xx}} & \frac{\partial U}{\partial r_n^y \partial \epsilon_{xy}} & \frac{\partial U}{\partial r_n^y \partial \epsilon_{xz}} & \frac{\partial U}{\partial r_n^y \partial \epsilon_{yy}} & \frac{\partial U}{\partial r_n^y \partial \epsilon_{yz}} & \frac{\partial U}{\partial r_n^y \partial \epsilon_{zz}} \\ \frac{\partial U}{\partial r_n^z \partial \epsilon_{xx}} & \frac{\partial U}{\partial r_n^z \partial \epsilon_{xy}} & \frac{\partial U}{\partial r_n^z \partial \epsilon_{xz}} & \frac{\partial U}{\partial r_n^z \partial \epsilon_{yy}} & \frac{\partial U}{\partial r_n^z \partial \epsilon_{yz}} & \frac{\partial U}{\partial r_n^z \partial \epsilon_{zz}} \end{pmatrix} \\ \begin{pmatrix} \frac{\partial U}{\partial r_1^x \partial \epsilon_{xx}} & \frac{\partial U}{\partial r_1^y \partial \epsilon_{xx}} & \frac{\partial U}{\partial r_1^z \partial \epsilon_{xx}} & \cdots & \cdots & \cdots & \frac{\partial U}{\partial r_n^x \partial \epsilon_{xx}} & \frac{\partial U}{\partial r_n^y \partial \epsilon_{xx}} & \frac{\partial U}{\partial r_n^z \partial \epsilon_{xx}} \\ \frac{\partial U}{\partial r_1^x \partial \epsilon_{xy}} & \frac{\partial U}{\partial r_1^y \partial \epsilon_{xy}} & \frac{\partial U}{\partial r_1^z \partial \epsilon_{xy}} & \cdots & \cdots & \cdots & \frac{\partial U}{\partial r_n^x \partial \epsilon_{xy}} & \frac{\partial U}{\partial r_n^y \partial \epsilon_{xy}} & \frac{\partial U}{\partial r_n^z \partial \epsilon_{xy}} \\ \frac{\partial U}{\partial r_1^x \partial \epsilon_{xz}} & \frac{\partial U}{\partial r_1^y \partial \epsilon_{xz}} & \frac{\partial U}{\partial r_1^z \partial \epsilon_{xz}} & \cdots & \cdots & \cdots & \frac{\partial U}{\partial r_n^x \partial \epsilon_{xz}} & \frac{\partial U}{\partial r_n^y \partial \epsilon_{xz}} & \frac{\partial U}{\partial r_n^z \partial \epsilon_{xz}} \\ \frac{\partial U}{\partial r_1^x \partial \epsilon_{yy}} & \frac{\partial U}{\partial r_1^y \partial \epsilon_{yy}} & \frac{\partial U}{\partial r_1^z \partial \epsilon_{yy}} & \cdots & \cdots & \cdots & \frac{\partial U}{\partial r_n^x \partial \epsilon_{yy}} & \frac{\partial U}{\partial r_n^y \partial \epsilon_{yy}} & \frac{\partial U}{\partial r_n^z \partial \epsilon_{yy}} \\ \frac{\partial U}{\partial r_1^x \partial \epsilon_{yz}} & \frac{\partial U}{\partial r_1^y \partial \epsilon_{yz}} & \frac{\partial U}{\partial r_1^z \partial \epsilon_{yz}} & \cdots & \cdots & \cdots & \frac{\partial U}{\partial r_n^x \partial \epsilon_{yz}} & \frac{\partial U}{\partial r_n^y \partial \epsilon_{yz}} & \frac{\partial U}{\partial r_n^z \partial \epsilon_{yz}} \\ \frac{\partial U}{\partial r_1^x \partial \epsilon_{zz}} & \frac{\partial U}{\partial r_1^y \partial \epsilon_{zz}} & \frac{\partial U}{\partial r_1^z \partial \epsilon_{zz}} & \cdots & \cdots & \cdots & \frac{\partial U}{\partial r_n^x \partial \epsilon_{zz}} & \frac{\partial U}{\partial r_n^y \partial \epsilon_{zz}} & \frac{\partial U}{\partial r_n^z \partial \epsilon_{zz}} \end{pmatrix} & \begin{pmatrix} \frac{\partial U}{\partial \epsilon_{xx} \partial \epsilon_{xx}} & \frac{\partial U}{\partial \epsilon_{xy} \partial \epsilon_{xx}} & \frac{\partial U}{\partial \epsilon_{xz} \partial \epsilon_{xx}} & \frac{\partial U}{\partial \epsilon_{yy} \partial \epsilon_{xx}} & \frac{\partial U}{\partial \epsilon_{yz} \partial \epsilon_{xx}} & \frac{\partial U}{\partial \epsilon_{zz} \partial \epsilon_{xx}} \\ \frac{\partial U}{\partial \epsilon_{xx} \partial \epsilon_{xy}} & \frac{\partial U}{\partial \epsilon_{xy} \partial \epsilon_{xy}} & \frac{\partial U}{\partial \epsilon_{xz} \partial \epsilon_{xy}} & \frac{\partial U}{\partial \epsilon_{yy} \partial \epsilon_{xy}} & \frac{\partial U}{\partial \epsilon_{yz} \partial \epsilon_{xy}} & \frac{\partial U}{\partial \epsilon_{zz} \partial \epsilon_{xy}} \\ \frac{\partial U}{\partial \epsilon_{xx} \partial \epsilon_{xz}} & \frac{\partial U}{\partial \epsilon_{xy} \partial \epsilon_{xz}} & \frac{\partial U}{\partial \epsilon_{xz} \partial \epsilon_{xz}} & \frac{\partial U}{\partial \epsilon_{yy} \partial \epsilon_{xz}} & \frac{\partial U}{\partial \epsilon_{yz} \partial \epsilon_{xz}} & \frac{\partial U}{\partial \epsilon_{zz} \partial \epsilon_{xz}} \\ \frac{\partial U}{\partial \epsilon_{xx} \partial \epsilon_{yy}} & \frac{\partial U}{\partial \epsilon_{xy} \partial \epsilon_{yy}} & \frac{\partial U}{\partial \epsilon_{xz} \partial \epsilon_{yy}} & \frac{\partial U}{\partial \epsilon_{yy} \partial \epsilon_{yy}} & \frac{\partial U}{\partial \epsilon_{yz} \partial \epsilon_{yy}} & \frac{\partial U}{\partial \epsilon_{zz} \partial \epsilon_{yy}} \\ \frac{\partial U}{\partial \epsilon_{xx} \partial \epsilon_{yz}} & \frac{\partial U}{\partial \epsilon_{xy} \partial \epsilon_{yz}} & \frac{\partial U}{\partial \epsilon_{xz} \partial \epsilon_{yz}} & \frac{\partial U}{\partial \epsilon_{yy} \partial \epsilon_{yz}} & \frac{\partial U}{\partial \epsilon_{yz} \partial \epsilon_{yz}} & \frac{\partial U}{\partial \epsilon_{zz} \partial \epsilon_{yz}} \\ \frac{\partial U}{\partial \epsilon_{xx} \partial \epsilon_{zz}} & \frac{\partial U}{\partial \epsilon_{xy} \partial \epsilon_{zz}} & \frac{\partial U}{\partial \epsilon_{xz} \partial \epsilon_{zz}} & \frac{\partial U}{\partial \epsilon_{yy} \partial \epsilon_{zz}} & \frac{\partial U}{\partial \epsilon_{yz} \partial \epsilon_{zz}} & \frac{\partial U}{\partial \epsilon_{zz} \partial \epsilon_{zz}} \end{pmatrix} \end{pmatrix}$$

Table 14.1: The structure of the generalized Hessian matrix. The matrix consists of four parts: a) the second derivative of the total energy with respect to particle position ($3N \times 3N$), b) the second derivatives of the total energy with respect to (6×6), and c) the cross terms ($3N \times 6$ and $6 \times 3N$, respectively).

Given $\mathbf{r}_0$, $\mathbf{v}_0$, and a time step $\Delta t > 0$, compute for $k = 0, 1, 2, \ldots$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{v}_k \Delta t \tag{14.7}$$
$$\mathbf{v}_{k+1} = \mathbf{v}_k + \mathbf{f}_{k+1} \Delta t \tag{14.8}$$

where

$$\mathbf{f}_k = -\nabla F(\mathbf{r}_k) = -\nabla F_k \tag{14.9}$$

If the Hessian matrix is positive semidefinite a sufficient condition for descent is

$$\mathbf{v}^T \mathbf{f}_{k+1} > 0 \tag{14.10}$$

A steepest descent step is introduced in the Snyman algorithm in order to try to satisfy this descent at each step:

$$\bar{\mathbf{v}} = \bar{\mathbf{f}}_k \Delta t \tag{14.11}$$
$$\bar{\mathbf{r}} = \bar{\mathbf{r}}_k + \bar{\mathbf{v}} \Delta t \tag{14.12}$$
$$\bar{\mathbf{f}} = -\nabla F(\bar{\mathbf{r}}) \tag{14.13}$$

Snyman considers thee possibilities which may arise in step k to $k+1$:

- $\mathbf{v}^T \mathbf{f}_{k+1} > 0$. The desired descent condition is met and the leap-frog integration is continued.
- $\mathbf{v}^T \mathbf{f}_{k+1} \leq 0$ and $\bar{\mathbf{v}}^T \bar{\mathbf{f}} > 0$. The descent condition is not met along the leap-frog trajectory while it is met along the steepest descent. An intermediate step is then taken with the certainty of achieving descent by successively setting $\mathbf{v}_{k-1} = \mathbf{v}_{k-1}/2$, recomputing $\mathbf{v}_k$ and restarting the step.
- $\mathbf{v}^T \mathbf{f}_{k+1} \leq 0$ and $\bar{\mathbf{v}}^T \bar{\mathbf{f}} \leq 0$. A remedy to avoid this case is to halve the time step and to restart the step with the original velocity $\mathbf{v}_k$.

Note that the particles are never explicitly forced to take a steepest descent step.

Besides being a natural and elegant solution, a molecular simulation package already contains all the available tools to make the Snyman algorithm easy to implement. The algorithm can easily handle large systems and in COTA we have adopted and modified the Snyman method to include rigid units. Other methods would require a constrained minimizer to explicitly guarantee that the extra degrees of freedom q_0, q_1, q_2, q_3 satisfy $q_0^2 + q_1^2 + q_2^3 + q_3^2 = 1$.

14.2.4 Newton-Rapson and quasi-Newton methods

Consider the energy expansion accoording to Eq. 14.1. We can differentiate this equation with respect to dx to ontain dU/dx. To find the displacement vector dx, we set the result to zero (stationary condition) and solve to get

$$dx = -\mathcal{H}^{-1}\mathbf{h} \tag{14.14}$$

if the generalized Hessian matrix is positive definite, taking this step will always lead to a lower energy. However, far away from a minimum there is no garantue this matrix is indeed positive definite. The idea behind quasi-Newton is to start with a positive definite, symmetric approximation to $\mathcal{H}$ (usually the unit matrix) and build up approximations of $\mathcal{H}$ in such a way that the matrix remains positive definite and symmetric.

14.2.5 Baker's method [1]

The Newton-Rapson step is

$$\delta\mathbf{x} = -\mathcal{H}^{-1}\mathbf{h} \tag{14.15}$$

It is convenient to transform the new displacement variables, the gradient and the Hessian to the Hessian eigenvector basis in which the Hessian matrix is diagonalized (by a unitary matrix E of eigenvectors):

$$\mathcal{H} \rightarrow \mathcal{H}' = E^T \mathcal{H} E = \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{pmatrix}, \qquad \delta \mathbf{x}' \rightarrow E^T \delta \mathbf{x}, \qquad \mathbf{h}' = E^T \mathbf{h} \tag{14.16}$$

The inverse of the Hessian $\mathcal{H}^{-1}$ has only diagonal element $(1/\lambda_1, 1/\lambda_2, \ldots, 1/\lambda_n)$ and the components of the Newton step can be written as

$$\delta x_i' = -\frac{\mathbf{h}_i'}{\lambda_i} \tag{14.17}$$

The diagonal step can be transformed back using $\delta \mathbf{x} \rightarrow V \delta \mathbf{x}'$ and we can thus rewrite the Newton step as a sum over the eigenvectors $\mathbf{e_i}$ (called local principle modes) and eigenvalues λ_i of the Hessian matrix

$$\delta \mathbf{x} = -\sum_i \frac{(\mathbf{e_i}^T \mathbf{h})}{\lambda_i} \mathbf{e_i} \tag{14.18}$$

where $\mathbf{e_i}^T \mathbf{h}$ is the component of the gradient along the eigenmode $\mathbf{e_i}$. For zero eigenvalues, the corresponding step component is set to zero. A zero eigenvalue means that for a displacement in the direction of the eigenvector the energy does not change, while a positive and negative value mean an increase and decrease in energy, respectively. Therefore, a true minimum has all positive eigenvalues. A first order saddle point has exactly one negative eigenvalue. The Newton-Rapson steps minimizes along the eigenvectors with positive eigenvalues and maximizes along eigenvectors with negative eigenvalues. Therefore, when the starting configuration is of the correct curvature (the desired number of negative eigenvalues) the Newton-Rapson is a good one to take. In general, the step must be modified to opbtain structure of the desired curvature [1].

A simple remification is to use a shift parameter γ which shift the value of the eigenvalues [2]

$$\delta \mathbf{r} = \sum_i \frac{\mathbf{e_i}^T \mathbf{f}}{\lambda_i - \gamma} \mathbf{e_i} \tag{14.19}$$

In order to find the shift parameter Simons et al. [3] rewrite Eq. **??** as

$$U(\mathbf{r} + \delta \mathbf{r}) - U(\mathbf{r}_0) = \frac{\mathbf{h}^T \delta \mathbf{r} + \frac{1}{2} \delta \mathbf{r}^T \mathcal{H} \delta \mathbf{r}}{1 + \delta \mathbf{r}^T \mathcal{S} \delta \mathbf{r}} \tag{14.20}$$

$$= \frac{\frac{1}{2} \left(\delta \mathbf{r}^T 1 \right) \begin{pmatrix} \mathcal{H} & \mathbf{h} \\ \mathbf{h}^T & 0 \end{pmatrix} \begin{pmatrix} \delta \mathbf{r} \\ 1 \end{pmatrix}}{\left(\delta \mathbf{r}^T 1 \right) \begin{pmatrix} \mathcal{S} & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \delta \mathbf{r} \\ 1 \end{pmatrix}} \tag{14.21}$$

where the diagonal matrix $\mathcal{S}$ can be viewed as a "scaling"-matrix. Setting the derivative $dE/d\delta \mathbf{r} = 0$ leads to the eigenvalue equation

$$\begin{pmatrix} \mathcal{H} & \mathbf{h} \\ \mathbf{h} & 0 \end{pmatrix} \begin{pmatrix} \delta \mathbf{r} \\ 1 \end{pmatrix} = \gamma \begin{pmatrix} \mathcal{S} & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \delta \mathbf{r} \\ 1 \end{pmatrix} \tag{14.22}$$

Note that the dimensionality is $(n+1)$ with n the number of variables to be optimized. Expanding out Eq. 14.22 gives

$$(\mathcal{H} - \gamma \mathcal{S}) \, \delta \mathbf{r} + \mathbf{h} = \mathbf{0} \tag{14.23}$$

$$\mathbf{h}^T \delta \mathbf{r} = \gamma \tag{14.24}$$

Making use of the diagonal representation of the Hessian, taking $\mathcal{S}$ as a unit matrix, Eq. 14.23 rearranges to Eq. 14.19, and substitution of Eq. 14.19 into the diagonal form of Eq. 14.24 gives the following expression for the shift parameter γ

$$\gamma = \sum_i \frac{\left(\mathbf{e_i}^T \mathbf{f}\right)^2}{\gamma - \lambda_i} \tag{14.25}$$

which can be solved by iteration (see Alg. 2).

The shift parameters γ have the following properties:

1. The $(n+1)$ eigenvalues of Eq. 14.22 bracket the n eigenvalues of the Hessian $\gamma_i \leq \lambda_i \leq \gamma_{i+1}$.
2. Near a local minimum, γ_1 is negative and approach zero at convergence.

Using these two properties we see that we can solve Eq. 14.25 using Newton-Rapson. The method will always converge, because the root must be less than zero and is bracketed by the first and second eigenvalue.

Taking a big step leads to an increase likelyness of moving out the region where the Hessian was valid. Therefore it is makes sense to set a maximum tolerance on the size of any calculated $\delta\mathbf{r}$ and scale down $\delta\mathbf{r}$ accordingely if this maximum is exceeded. A maximum stepsize of 0.3 was found to work well in practice [3].

The update of the Hessian matrix can also be done by an update scheme. The update depends on the gradient vector at the old point $\mathbf{h}$, the gradient at the new point $\mathbf{h}'$, and the displacement step $\delta\mathbf{r}$. Two alternate algorithms are the Powell update

$$\mathcal{H}^{\text{new}} = \mathcal{H}^{\text{old}} + \frac{1}{\delta\mathbf{r}^T\delta\mathbf{r}}\left[\mathcal{V}\delta\mathbf{r}^T + \delta\mathbf{r}\mathcal{V}^T - \frac{\left(\mathcal{V}^T\delta\mathbf{r}\right)\left(\delta\mathbf{r}\delta\mathbf{r}^T\right)}{\left(\delta\mathbf{r}^T\delta\mathbf{r}\right)}\right] \tag{14.26}$$

where

$$\mathcal{V} = (\mathbf{h}' - \mathbf{h} - \mathcal{H}\mathbf{h}) \tag{14.27}$$

and the BFGS update

$$\mathcal{H}^{\text{new}} = \mathcal{H}^{\text{old}} + \frac{\mathcal{V}\mathcal{V}^T}{\left(\mathcal{V}^T\delta\mathbf{r}\right)} - \frac{\left(\mathcal{H}^{\text{old}}\delta\mathbf{r}\right)\left(\delta\mathbf{r}^T\mathcal{H}^{\text{old}}\right)}{\delta\mathbf{r}^T\mathcal{H}^{\text{old}}\delta\mathbf{r}} \tag{14.28}$$

with

$$\mathcal{V} = (\mathbf{h}' - \mathbf{h}) \tag{14.29}$$

The BFGS update is generally considered the best choice for a minimum. For a transition state the Powell update is prefered. However, since the computational bottleneck is the diagonalization of the Hessian rather than obtaining the Hessian itself, the best method is to use the recalculated Hessian at each iteration.

if $\lambda_2 \geq$ *small* **then**
  $\gamma = 0$
else
  $\gamma = \frac{\lambda_1+\lambda_2}{2}$
end
repeat
  $f = \gamma - \sum_i \frac{\left(\mathbf{e_i}^T\mathbf{f}\right)^2}{\gamma-\lambda_i}$;
  $f' = 1 + \sum_i \frac{\left(\mathbf{e_i}^T\mathbf{f}\right)^2}{(\gamma-\lambda_i)^2}$;
  $\gamma = \gamma - \frac{f}{f'}$;
until $f < \epsilon$;

Algorithm 2: Baker minimization: finding γ_1 by Newton-Rapson. The method will always converge, because the root must be less than zero and is bracketed by the first and second eigenvalue. The starting value for λ_1 can be chosen halfway the first and the second eigenvalues. Convergence is reached when $f < \epsilon$ with ϵ small.

Set $\delta\mathbf{r} = \mathbf{0}$;
repeat
  Take the step $\mathbf{r}' = \mathbf{r} + \delta\mathbf{r}$;
  Compute the energy U, gradient $\mathbf{h}$, and Hessian-matrix $\mathcal{H}$;
  Diagonalize the Hessian-matrix to obtain the eigenvectors $\mathbf{e_i}$ and eigenvalues λ_i;
  if *no negative eigenvalues and NR is "true"* **then**
    Take the Newton-Rapson step $\delta\mathbf{r} = -\sum_i \frac{\mathbf{e_i}^T\mathbf{h}}{\lambda_i}\mathbf{e_i}$;
  else
    Compute γ_1 by iteration;
    Take the Baker step: $\delta\mathbf{r} = -\sum_i \frac{\mathbf{e_i}^T\mathbf{h}}{\lambda_i - \gamma_1}\mathbf{e_i}$;
  end
  if $|\delta\mathbf{r}| > \delta\mathbf{r}_{max}$ **then**
    Scale down step so that $|\delta\mathbf{r}| = \delta\mathbf{r}_{\max}$;
  end
  Check convergence criteria;
until *Converged and no negative eigenvalues* ;

Algorithm 3: Baker minimization.

14.2.6 Projection operators for constraints

The set of Cartesian coordinates is redundant. Only $3N-6$ coordinates are needed to describe the geometry of N atoms, but the coordinates are of size $3N$. It is important to prevent rotational and translational motions during minimization. The Eckart conditions provide a way of doing this. The full $3N \times 3N$ Hessian in Cartesian coordinates is treated by first projecting out vectors corresponding to translations and initesimal rotations constructed using the Eckart conditions:

$$\mathcal{T} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \vdots & \vdots & \vdots \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \mathcal{R} = \begin{pmatrix} 0 & z_1 & -y_1 \\ -z_1 & 0 & x_1 \\ y_1 & -x_1 & 0 \\ 0 & z_2 & -y_2 \\ -z_2 & 0 & x_2 \\ y_2 & -x_2 & 0 \\ \vdots & \vdots & \vdots \\ 0 & z_N & -y_N \\ -z_N & 0 & x_N \\ y_N & -x_N & 0 \end{pmatrix} \tag{14.30}$$

The translation and rotaional contraints form a set $\mathbf{e}_i, i = 1 \ldots 6$, and the unit vectors $\mathbf{e}_i$ must be orthogonal to each other. A Gram-Schimdt orthogonalization can be carried out and the vectors can be subsequently normalized. The orthgonal projector on the set $\mathbf{e}_i$ can be written as

$$\mathcal{P} = \mathcal{I} - \sum_{i=1}^{m} \mathbf{e}_i \left(\mathbf{e}_i^T\right) \tag{14.31}$$

The matrix $\mathcal{P}$ of size $3N \times 3N$ can be used as a projection operator on first and second derivatives:

$$\mathbf{h}' = \mathcal{P}\mathbf{h} \tag{14.32}$$

$$\mathcal{H}' = \mathcal{P}\mathcal{H}\mathcal{P} \tag{14.33}$$

$$\tag{14.34}$$

The resulting matrix is then diagonalized and eigenvectors with zero eigenvalues rejected.

14.3 Saddle points

14.3.1 Baker's method

14.4 Discussion and examples

Bibliography

[1] J. Baker. An algorithm for the location of transition-states. *J. Comp. Chem.*, 7(4):385–395, 1986.

[2] C. J. Cerjan and W. H. Miller. On finding transition-states. *J. Chem. Phys.*, 75(6):2800–2806, 1991.

[3] J. Simons, P. Jorgensen, H. Taylor, and J. Ozment. Walking on potential-energy surfaces. *J. Phys. Chem.*, 87(15):2745–2753, 1983.

Chapter 15

Vibrational Analysis

15.1 Powder diffraction

15.1.1 Bragg's law

Consider an incident front of waves with parallel propagation vectors which forms an angle θ with the plaens (hkl). The path differences Δ introduced between a pair of waves both before and after they are reflected by the neighboring planes are $\Delta = d_{hkl} \sin\theta$. The total path difference is 2Δ, and the constructive inteference is observed when $2\Delta = n\lambda$, where n is an integer and λ is the wave length of the indicent wavefront. We arrive at Bragg's law:

$$2d_{hkl} \sin\theta_{hkl} = n\lambda \tag{15.1}$$

15.1.2 Peak positions

The interplanar distance is a function of the unit cell parameters and Miller indices, h, k, and l, which fully describe every set of crystallographic planes. The corresponsing formula for the inverse square of the interplanar distance $1/d^2$ is given by

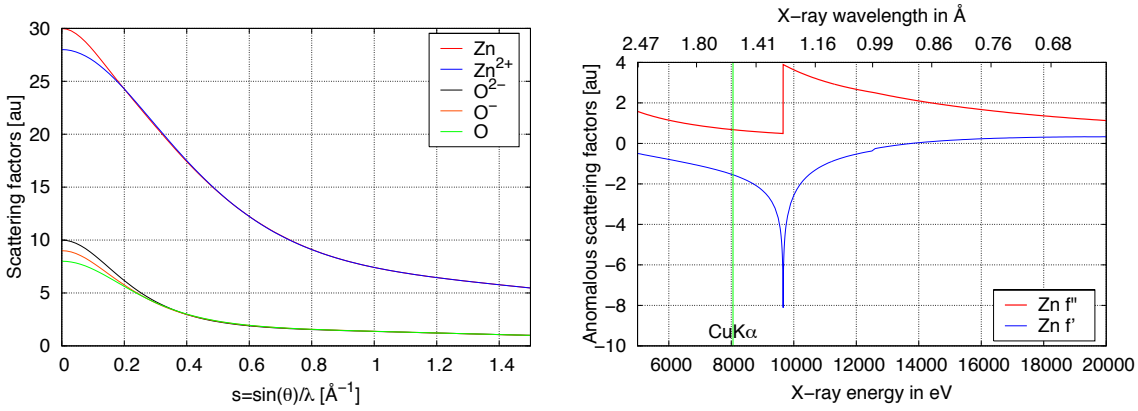
$$\begin{aligned}\frac{1}{d^2} =& \frac{\frac{h^2}{a^2\sin^2\alpha} + \frac{2kl}{bc}(\cos\beta\cos\gamma - \cos\alpha)}{1 - \cos^2\alpha - \cos^2\beta - \cos^2\gamma + 2\cos\alpha\cos\beta\cos\gamma} + \\ & \frac{\frac{k^2}{b^2\sin^2\beta} + \frac{2hl}{ac}(\cos\alpha\cos\gamma - \cos\beta)}{1 - \cos^2\alpha - \cos^2\beta - \cos^2\gamma + 2\cos\alpha\cos\beta\cos\gamma} + \\ & \frac{\frac{l^2}{c^2\sin^2\gamma} + \frac{2hk}{ab}(\cos\alpha\cos\beta - \cos\gamma)}{1 - \cos^2\alpha - \cos^2\beta - \cos^2\gamma + 2\cos\alpha\cos\beta\cos\gamma}\end{aligned} \tag{15.2}$$

In term of the reciprocal lattice

$$\mathbf{a}^* = \frac{\mathbf{b}\times\mathbf{c}}{V}, \qquad \mathbf{b}^* = \frac{\mathbf{c}\times\mathbf{a}}{V}, \qquad \mathbf{c}^* = \frac{\mathbf{a}\times\mathbf{b}}{V} \tag{15.3}$$

the decription significantly simplies:

$$\mathbf{d}^*_{hkl} = h\mathbf{a}^* + k\mathbf{b}^* + l\mathbf{c}^*, \qquad d^*_{hkl} = \frac{1}{d_{hkl}} \tag{15.4}$$



(a) Scattering of zinc and oxygen elements.

(b) Anomalous scattering of a zinc atom.

Figure 15.1: Scattering factors.

The range of Miller indices for a θ-range of $[0, \theta^{\max}]$ is $h = \{-h^{\max}, h^{\max}\}$, $k = \{-k^{\max}, k^{\max}\}$, and $l = \{-l^{\max}, l^{\max}\}$ with

$$h^{\max} = (\text{int}) \frac{2 \sin \theta^{\max}}{\lambda a^*} \qquad a^* = |\mathbf{a}^*| = \sqrt{\left(a_x^*\right)^2 + \left(a_y^*\right)^2 + \left(a_z^*\right)^2} \tag{15.5}$$

$$k^{\max} = (\text{int}) \frac{2 \sin \theta^{\max}}{\lambda b^*} \qquad b^* = |\mathbf{b}^*| = \sqrt{\left(b_x^*\right)^2 + \left(b_y^*\right)^2 + \left(b_z^*\right)^2} \tag{15.6}$$

$$l^{\max} = (\text{int}) \frac{2 \sin \theta^{\max}}{\lambda c^*} \qquad c^* = |\mathbf{c}^*| = \sqrt{\left(c_x^*\right)^2 + \left(c_y^*\right)^2 + \left(c_z^*\right)^2} \tag{15.7}$$

Then, all possible angles of the diffraction peaks can be found by looping over the Miller indices using

$$2\theta_{hkl} = 2 \arcsin \left(\frac{1}{2} \lambda d_{hkl}^* \right) \tag{15.8}$$

with

$$d_{hkl}^* = |\mathbf{d}_{hkl}^*| = \sqrt{\left(ha_x^* + kb_x^* + lc_x^*\right)^2 + \left(ha_y + kb_y^* + lc_y^*\right)^2 + \left(ha_z^* + kb_z^* + lc_z^*\right)^2} \tag{15.9}$$

15.1.3 Peak intensities

The relative peak intensities are given by:

$$I_{hkl} \propto M_{hkl} L_p \left| F_{hkl} \right|^2 \tag{15.10}$$

1. Multiplity of the Bragg plans M_{hkl}
2. The Lorents monochromator polarization factor L_p

$$L_p = \left(\frac{1 + \cos^2 (2\theta)}{\sin^2 \theta \cos \theta} \right)_{hkl} \tag{15.11}$$

3. The scattering factor F_{hkl} for reflection hkl

$$\mathbf{F}_{hkl} = \sum_{i=1}^{N} g_i t_i (s) f_i (s) \exp \left[2\pi \imath \left(hx_i + ky_i + lz_i \right) \right], \qquad s = \frac{\sin \theta_{hkl}}{\lambda} \tag{15.12}$$

where x, y, z of the fractional positions of the N atoms.

(a) the population factor g_i

(b) atomic scattering $f_i(s)$

$$f_i(s) = c_0^i + \sum_{j=4}^{4} a_j^i \exp\left[-b_j^i s^2\right] \tag{15.13}$$

The scattering factors of various chemical elemenst and ions can be represented as functions of 9 coefficients $c_0, a_1, \ldots, a_4, b_1, \ldots, b_4$, which can be found in the International Tables for Crystallography, vol. C.

(c) anomalous scattering

(d) temperature effect $t_i(s)$

$$t^i(s) = \exp\left[-B^i s^2\right] \tag{15.14}$$

$$t^i(s) = \exp\left[-\frac{1}{4}\left(B_{11}^i h^2 a^{*2} + B_{22}^i k^2 b^{*2} + B_{33}^i l^2 c^{*2} + 2B_{12}^i hka^*b^* + 2B_{13}^i hla^*c^* + 2B_{23}^i klb^*c^*\right)\right] \tag{15.15}$$

15.1.4 Peak shapes

15.2 Infra-red spectra (harmonic approximation)

The potential energy surface of a (periodic) system can be Taylor-expanded

$$U(\mathbf{r}_0 + \Delta\mathbf{r}) = U(\mathbf{r}_0) + \left(\frac{\partial U}{\partial r}\right)^T \Delta\mathbf{r} + \frac{1}{2}\Delta\mathbf{r}^T \frac{\partial U}{\partial r_\alpha \partial r_\beta}\Delta\mathbf{r} + \ldots \tag{15.16}$$

The Hamiltonian and the energy of a 1D harmonic oscillator are given by

$$H_v^h = -\frac{\hbar}{2}\frac{\partial}{\partial E^2} + \frac{1}{2}\frac{\partial^2 U}{\partial E^2}\bigg|_{E=E_0} E^2 \tag{15.17}$$

$$\frac{U_v^h}{hc} = \tilde{w}\left(n + \frac{1}{2}\right) \tag{15.18}$$

15.3 Infra-red spectra (anharmonic approximation)

$$\begin{aligned} U(\mathbf{E}) = U\left(\mathbf{E}^0\right) &+ \sum_{i=1}^{N} \frac{\partial U}{\partial \mathbf{E}_i}\bigg|_{\mathbf{E}=\mathbf{E}^{(0)}} \left(\mathbf{E}_i - \mathbf{E}_i^{(0)}\right) \\ &+ \frac{1}{2!}\sum_{i=1}\sum_{j=1} \frac{\partial^2 U}{\partial \mathbf{E}_i \partial \mathbf{E}_j}\bigg|_{\mathbf{E}=\mathbf{E}^{(0)}} \left(\mathbf{E}_i - \mathbf{E}_i^{(0)}\right)\left(\mathbf{E}_j - \mathbf{E}_j^{(0)}\right) \\ &+ \frac{1}{3!}\sum_{i=1}\sum_{j=1}\sum_{k=1}^{N} \frac{\partial^3 U}{\partial \mathbf{E}_i \partial \mathbf{E}_j \partial \mathbf{E}_k}\bigg|_{\mathbf{E}=\mathbf{E}^{(0)}} \left(\mathbf{E}_i - \mathbf{E}_i^{(0)}\right)\left(\mathbf{E}_j - \mathbf{E}_j^{(0)}\right)\left(\mathbf{E}_k - \mathbf{E}_k^{(0)}\right) \\ &+ \frac{1}{4!}\sum_{i=1}\sum_{j=1}\sum_{k=1}^{N}\sum_{l=1}^{N} \frac{\partial^4 U}{\partial \mathbf{E}_i \partial \mathbf{E}_j \partial \mathbf{E}_k \partial \mathbf{E}_l}\bigg|_{\mathbf{E}=\mathbf{E}^{(0)}} \left(\mathbf{E}_i - \mathbf{E}_i^{(0)}\right)\left(\mathbf{E}_j - \mathbf{E}_j^{(0)}\right)\left(\mathbf{E}_k - \mathbf{E}_k^{(0)}\right)\left(\mathbf{E}_l - \mathbf{E}_l^{(0)}\right) \\ &+ \ldots \end{aligned} \tag{15.19}$$

For most potential analytical first and second derivative can easily be calculated. Having obtained the normal mode accoording to the harmonic approximation a finite difference van be applied where one compute the second derivatives (analytically) for displaced geometries along a single eigenmode $\mathbf{E}_k$.

$$\frac{\partial^3 U}{\partial \mathbf{Q}_i \partial \mathbf{Q}_j \partial \mathbf{Q}_k} = \frac{-k_{ij}^{+2} + 8k_{ij}^{+} - 8k_{ij}^{-} + k_{ij}^{-2}}{12\Delta \mathbf{Q}_k} \tag{15.20}$$

$$\frac{\partial^4 U}{\partial \mathbf{Q}_i \partial \mathbf{Q}_j \partial \mathbf{Q}_k \partial \mathbf{Q}_l} = \frac{k_{ij}^{++} - k_{ij}^{--} + k_{ij}^{-+} + k_{ij}^{+-}}{\Delta \mathbf{Q}_k \Delta \mathbf{Q}_l} \tag{15.21}$$

where k_{ij} is the second derivative at the reference point, k_{ij}^{+} and k_{ij}^{-} correspond to displaced geometries along mode E_k in the positive and negative direction, respectively, $k_{ij}^{++}, k_{ij}^{--}, k_{ij}^{+-}, k_{ij}^{-+}$ means a positive/negative displacement along mode E_k and a positive/negative displacement along mode E_l.

The energy levels of a 1D harmonic oscillator are given by

$$U\left(n\right) = \sum_{i=1}^{N} \tilde{w}_i \left(n_i + \frac{1}{2}\right) + \sum_{i \geq j}^{N} x_{ij} \left(n_i + \frac{1}{2}\right)\left(n_j + \frac{1}{2}\right) + \ldots \tag{15.22}$$

This equation is known as the Dunham expansion. The relationship between the harmonic term x and the derivatives of the potential is given by

$$x = -\frac{5}{48\tilde{w}} \frac{\partial^3 U}{\partial E^3} + \frac{1}{16} \frac{\partial^4 U}{\partial E^4} \tag{15.23}$$

The harmonic frequencies ω depend on the harmonic force constants, the anharmonic constants x can be expressed in terms of cubic and quartic anharmonicities. Typically, the anharmonic corrections to the harmonic frequencies

$$\nu_i - \omega_i = 2x_{ii} + \frac{1}{2} \sum_{j \not= i} x_{ij} \tag{15.24}$$

are approximately 10%-15%.

Chapter 16

Transition State Theory

16.1 Dynamically corrected transition state theory

16.1.1 Theory

Lattice random walk theory

Diffusive motion of particles occurs by a series of discrete steps separated by elastic collisions, localized vibrations, and short shuffles. Diffusion is an irreversible macroscopic process, but is actually comprised of reversible microscopic steps, and may be well described by random walk theory. A random walk is a simple mathematical model for the movement of a particle on a lattice under the influence of some random or stochastic force affecting its direction of motion. It is particularly attractive because in many instances analytical solutions can be worked out for both static as well as dynamic properties. From the internal (crystal) structure a lattice can be constructed that determines the lattice topology and the lattice distances. The dynamics of the random walk is uniquely determined once the jumping frequencies k_i for a lattice direction i are specified. The jump frequency is defined as

$$k = \frac{\langle \text{number of successful hops} \rangle}{\text{unit of time}} \tag{16.1}$$

The total jumping frequency k_{tot} is related to the specific jumping frequencies k_i for a given structure by a summation over the lattice connectivity Z:

$$k_{\text{tot}} = \sum_{i=1}^{Z} k_i \tag{16.2}$$

For a jump to be truly random, each of the possible jump directions is chosen with equal probability, the probability that the new lattice site is empty does not enter into any equation (the particles can overlap). The expected value $\langle \mathbf{r}(t) \rangle = 0$, and the driving force $\nabla \mu = 0$ for a simple regular random walk (necessary for the measurement of the self-diffusivity D_S). However, in real systems, jumps are usually correlated by defined interactions between jumping particles.

Let k_i be the average frequency that a random walker (an atom or molecule) jumps for lattice vector $\boldsymbol{\lambda}_i$, and $\mathbf{r}(t)$ is the position of a particular random walker. The position of a particle (relative to the starting position) after a time t (or $n = k\,t$ hops) will be:

$$\mathbf{r}(t) = \sum_{i=1}^{n} \boldsymbol{\lambda}_i \tag{16.3}$$

where n_i is the number of jumps in lattice direction λ_i on a regular lattice with connectivity Z. In primitive cubic crystals there exists one lattice site per unit cell surrounded by $Z = 6$ neighbors, the lattice vectors

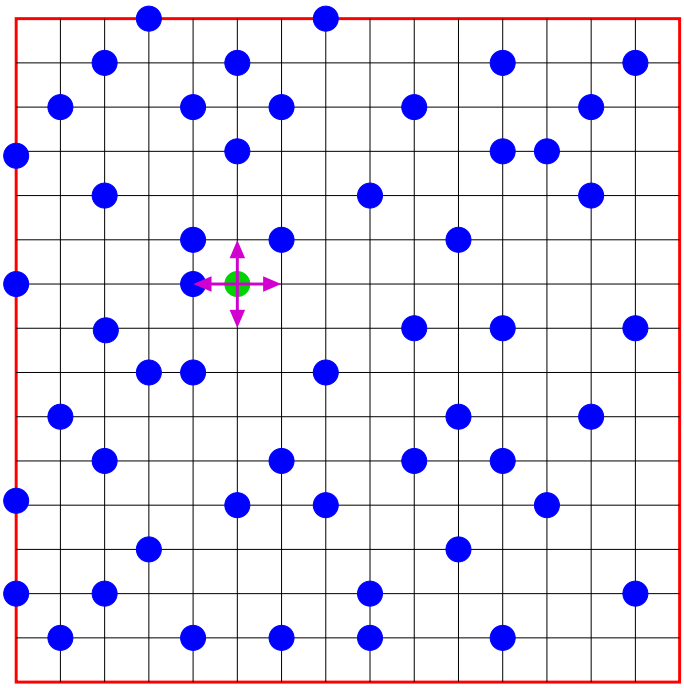


Figure 16.1: A random walk of N particles on a $M \times M$ lattice. One of the particles is chosen at random (the green one). Next, one of the four (up, down, left, right) directions is chosen. Only if the site is empty the jump is allowed. In this example the jump of the green particle to the left would be rejected. Particles leaving the lattice are put back at the other end of the lattice (periodic boundary conditions).

are $\boldsymbol{\lambda}_i = \{\{1,0,0\},\{0,1,0\},\{0,0,1\},\ \{\bar{1},0,0\},\{0,\bar{1},0\},\{0,0,\bar{1}\}\}$. The distance between two particles should increase with time, which is measured by the spread of the distribution $\langle \mathbf{r}\,(t)\rangle$

$$\left\langle \mathbf{r}^2\,(t)\right\rangle = \left\langle \sum_{i=1}^{n} \boldsymbol{\lambda}_i \cdot \boldsymbol{\lambda}_i + 2\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} \boldsymbol{\lambda}_i \cdot \boldsymbol{\lambda}_j \right\rangle \tag{16.4}$$

written as a sum of diagonal and off-diagonal terms.

For the one-dimensional lattice, the two-dimensional square lattice, and the three-dimensional cubic lattice, all the jumping frequencies k_i and jump vectors are equivalent. Using the relationship $\boldsymbol{\lambda}_i \cdot \boldsymbol{\lambda}_j = |\boldsymbol{\lambda}_i|\,|\boldsymbol{\lambda}_j| \cos\Delta\phi_{ij}$, where $\Delta\phi_{ij}$ is the angle between the i^{th} and j^{th} jump-vectors. Therefore,

$$\mathbf{r}^2\,(t) = n\,\lambda^2 + 2\left(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} |\boldsymbol{\lambda}_i|\,|\boldsymbol{\lambda}_j| \cos\Delta\phi_{ij}\right) \tag{16.5}$$

$$= n\,\lambda^2 + \sum_{i=1}^{n}\sum_{\substack{j=1\\ j\neq i}}^{n} |\boldsymbol{\lambda}_i|\,|\boldsymbol{\lambda}_j| \cos\Delta\phi_{ij} \tag{16.6}$$

$$= n\,\lambda^2\left(1 + \frac{1}{n}\left\langle \sum_{i=1}^{n}\sum_{j=1}^{n} \cos\Delta\phi_{ij}\right\rangle\right) \tag{16.7}$$

because $\Delta\phi_{ij} = -\Delta\phi_{ji}$. We can write

$$\left\langle \mathbf{r}\,(t)^2\right\rangle = n\,f\lambda^2 = Zk_i\,f\lambda^2 t, \tag{16.8}$$

and apply Einstein's equation

$$\left\langle \mathbf{r}\left(t\right)^{2}\right\rangle = 2dDt \tag{16.9}$$

to yield

$$D = \frac{Z}{2d} f\, k_i \lambda^2 = f\ k_{A\rightarrow B}\lambda^2 \tag{16.10}$$

$$= \frac{1}{2d} f\, k_{\mathrm{tot}} \lambda^2 \tag{16.11}$$

This relates the macroscopic self-diffusivity to the jump frequency k, lattice hop distance λ and correlation factor f. The basic assumption of the random walk model is the quick loss of memory of the molecules between consecutive jumps, i.e. a molecule will proceed with a probability independent of its history. The correlation factor f

$$f \equiv 1 + \frac{1}{n}\left\langle \sum_{i=1}^{n}\sum_{j=1}^{n} \cos\Delta\phi_{ij} \right\rangle \tag{16.12}$$

contains *all* memory-effects, arising from ordering and interparticle interactions.

The regular random walk has no "memory" of the previous step when determining the current one. This feature can be applied to a wide range of physical problems, but there are a number of other interesting problems for which this is not the case. In a persistent walk, the transition (or step) probability depends upon the previous transition, and a particle has a retention to the directional over a certain number of trajectory steps. In order for a simplistic regular lattice model to be valid, the loss of memory is an important condition that has to be satisfied.

Correlations

The collective diffusivity contains all the dynamical correlations. Here, the motion results from the jumps of *different* particles at different times. In contrast, for self-diffusivity the motion results from the jumps of a *tagged* particle at different times. Memory effects [**?**, **?**] have a tendency to decrease D_S with respect to D_C, indicating it is somehow related to the well-known back-correlation mechanism where a diffusing particle has a higher probability to jump backwards than in any other direction, simply because the originating site is guaranteed to be empty. Most of the memory effect arise from ordering and interparticle interactions, the latter giving the leading contribution. Fig. **??** shows the self- and collective diffusivity of methane in LTA-type zeolite at 300K as a function of loading. In the low loading limit both diffusivities converge, because particle-particle interactions vanish. However, at a loading of 7 molecule per cage or higher the correlations are clearly visible. Although in principle conventional MD captures all relevant correlations, the dcTST method by Beerdsen et al. not only captures these correlations correctly but is also suitable for systems with large free energy barriers [**?**]. These correlations, originating from particle-particle interactions, are significant at higher loadings as is evidenced by the large difference between collective and self-diffusivity.

Memory effect are stronger on single particle motion than on the collective motion, where most of the back-correlations cancel out. For a Langmuir gas, where the only interaction is the site exclusion, they cancel out *exactly*. Correlations between successive jumps can be studied by considering *directional* correlations between two jumps separated by m previous jumps by a tagged particle. For a Langmuir gas the factor f reduces in the high loading limit to [**?**]

$$f = \frac{1 + \langle\cos\phi\rangle}{1 - \langle\cos\phi\rangle} \tag{16.13}$$

as a correlation factor for vacancy diffusion, where ϕ is the angle between two consecutive single particle jumps. This equation assumes that the predominant memory contribution comes from the back-correlation between two consecutive single-particle jumps ($m = 1$). Eq. 16.13 is very much related to the end-to-end distance of an isolated, infinitely long, hypothetical model chain comprised of bonds of fixed lengths joined with fixed bend angles [**?**]. Unlike the freely joined case, the fixing of the angles ϕ imposes correlations.

A particle residing in a lattice point once in a while jumps to a neighboring site. If thermalization occurs we call it a *single jump*, otherwise we speak of a *long jump* or a *multi-jump*. These kinetic correlations become important at low loadings and in channel-type structures with smooth walls, e.g. carbon nanotubes. However, for entropy dominated barriers (e.g. methane and ethane in LTA-type zeolites) one can usually neglect kinetic correlations.

Dynamically corrected transition state theory (dcTST) at infinite dilution

At infinite dilution and sufficient dissipation the correlation factor $f = 1$ (there are no memory-effects), and Eq. 16.10 reduces to

$$D = k_i \lambda^2 = k_{AB} \lambda^2 \tag{16.14}$$

The lattice distance λ is fixed and a property that can be obtained from crystal Xray-scattering experiments. Therefore, Eq. 16.14 defines k_i as the hopping rate from lattice point A (in equilibrium) to a neighboring lattice point B (in equilibrium). Note that an attempt of a hop is always successful, and that a particle cannot have a position in between the lattice points, i.e. the jump is instantaneous and discrete. In principle, one could use MD simulations to determine this rate by computing the average residence time that a particle is in a cage. However, such a computation using MD proves cumbersome. Firstly, an A-to-B order parameter has to be defined, secondly, a criteria should be used to distinguish unsuccessful hops on a very short time scale from the successful AB-hop on a much longer time scale (the equilibrium one), and thirdly, very few trajectories will involve motion from exactly point A to point B.

Dynamically corrected transition state theory overcomes these problems by computing precisely what we need: the hopping rate from lattice point A (in equilibrium) to a neighboring lattice point B (in equilibrium). In other words, dcTST is fully compatible and complementary to lattice random walk theory. We consider a system which can be in two stable states, A and B. The reaction coordinate, a parameter that indicates the progress of the diffusion event from region A to region B, is denoted by q. Here, q is a function of the Cartesian coordinates, $\dot{q}$ denotes its time derivative, q^* is the location of the dividing surface, and q_A, q_B are the minima of the free energy corresponding to state A and B, respectively. In general, the reaction coordinate q is a function of the configuration of the whole system, i.e. $q = q(\mathbf{r}_1, \ldots, \mathbf{r}_N)$. However, we can choose q as the position of one of the atoms of the diffusing molecules [**?**]. We introduce two characteristic functions n_A and n_B that measure whether the system is in state A or B. A possible and often used definition is

$$n_A = \theta\left(q^* - q\right), \tag{16.15}$$

$$n_B = \theta\left(q - q^*\right), \tag{16.16}$$

where θ is the Heaviside function $\theta(x)$, which has value zero for $x < 0$ and value of unity for $x \geq 0$. With these definitions the transition rate $k_{A\to B}$ is given by [**?**]

$$k_{A\to B} = \underbrace{\frac{\langle \delta\left(q^* - q\right)\rangle}{\langle \theta\left(q^* - q\right)\rangle}}_{P_{\in A}(q^*)} \times \underbrace{\frac{\langle \dot{q}(0)\delta\left(q^* - q(0)\right)\theta\left(q\left(t\right) - q^*\right)\rangle}{\langle \delta\left(q^* - q(0)\right)\rangle}}_{R(t)}, \tag{16.17}$$

where δ is the Dirac delta function, $\langle n_A \rangle$ is the equilibrium mole fraction of particles in state A, $P_{\in A}\left(q^*\right)$ is the equilibrium probability density of finding the system at the top of the barrier divided by the equilibrium probability of finding it at state A, and where $R\left(t\right)$ is the averaged particle flux at the top of the barrier multiplied by the probability that the system ends up in state B at time t. From detailed balance follows

$$\frac{k_{A\to B}}{k_{B\to A}} = \frac{\langle n_B \rangle}{\langle n_A \rangle} \tag{16.18}$$

where

$$\langle n_A \rangle = \frac{\int_A e^{-\beta F(q)}\, dq}{\int_{A+B} e^{-\beta F(q)}\, dq} \tag{16.19}$$

$$\langle n_B \rangle = \langle 1 - n_A \rangle \tag{16.20}$$

The expression Eq. 16.17 is rigorously correct for arbitrary crossings provided that

- the actual crossing time is negligible compared to the time a particle spends inside the cage, i.e. there is a large separation in time scales. This condition is satisfied when the free energy barrier is much larger than $k_B T$.
- the velocity distribution at the dividing surface is known (The order parameter q is taken to be the position of a particle and therefore $\dot{q}$ is simply the velocity of a particle. In TST is it assumed that the top of the barrier is in equilibrium and hence these velocities follow directly from the Maxwell-Boltzmann distribution).

At infinite dilution the molecules perform a random walk on a lattice spanned by the cage-centers. The transmission rates are easily converted to diffusion coefficients if the jump distance and the number of equivalent diffusion paths are known.

Eq. 16.17 is a product of a static and a dynamic term

- the probability $P_{\in A}(q^*)$ of finding the system at the top of the barrier is a *time-independent* equilibrium quantity and can be computed explicitly

$$\begin{aligned} P_{\in A}(q^*) &= \frac{\langle \delta(q^* - q) \rangle}{\langle \theta(q^* - q) \rangle} \\ &= \frac{e^{-\beta F(q^*)}}{\int_{\text{cage A}} e^{-\beta F(q)}\, \mathrm{d}q}, \end{aligned} \tag{16.21}$$

  where $F(q)$ is the free energy as a function of the diffusion path q.

- the flux $R(t)$ through the dividing surface is a conditional average, namely the product $\dot{q}(0)\,\theta(q(t) - q^*)$, given that $q(0) = q^*$. Using the assumption that the velocities of the atoms follow the Maxwell-Boltzmann distribution, we can estimate from kinetic theory the long time value of $R(t)$ by $\left\langle \frac{1}{2}|\dot{q}(0)| \right\rangle = \sqrt{\frac{k_B T}{2\pi m}}$, where m is the mass of the segments of the particle involved in the reaction coordinate (the total mass of the particle if the center of mass is used or the mass of only one segment if the reaction coordinate is a single segment like the middle bead in a molecule). Transition state theory predicts a crossing rate $k^{\text{TST}}_{A\to B}$ given by

$$k^{\text{TST}}_{A\to B} = \sqrt{\frac{k_B T}{2\pi m}} \frac{e^{-\beta F(q^*)}}{\int_{\text{cage A}} e^{-\beta F(q)}\, \mathrm{d}q}. \tag{16.22}$$

  Calculating TST rate constants is therefore equivalent to calculating free energy differences.

  The TST particle flux estimation $\sqrt{\frac{k_B T}{2\pi m}}$ contains spurious crossings, i.e. some particles that cross the transition state from A in reality would fail to equilibrate in B. The correction $\kappa(t)$ is defined as the ratio between the real rate and the TST expression.

$$\kappa(t) \equiv \frac{k_{A\to B}(t)}{k^{\text{TST}}_{A\to B}} = \frac{\langle \dot{q}(0)\delta(q(0) - q^*)\,\theta(q(t) - q^*) \rangle}{\left\langle \frac{1}{2}|\dot{q}(0)|\delta(q(0) - q^*) \right\rangle}. \tag{16.23}$$

It is the probability that a particle that starts with an initial velocity $\dot{q}$ from the dividing surface will, in fact, cross the barrier, and therefore $\kappa(t)$ corrects for trajectories which cross the transition state from A but fail to equilibrate in B. The numerator in Eq. **??** counts trajectories with a positive, but also with a negative weight. It can be shown that $\lim_{t\to 0^+}\kappa(t) = 1$ and $\lim_{t\to 0^+} k_{A\to B}(t) = k^{\mathrm{TST}}_{A\to B}$. There is a large separation of time scales. The transmissions are completed in a time much less than the time to react, and Eq. **??** will reach a plateau value κ. For classical systems $0 < \kappa \leq 1$ and Eq. 16.22 is corrected as

$$k_{A\to B} = \kappa\, k^{\mathrm{TST}}_{A\to B}. \tag{16.24}$$

Standard Molecular Dynamics (MD) yields the transmission coefficients, a separate MC simulation is used to generate the starting configurations. The reaction coordinate is restricted to the dividing surface q^*. The MC moves involved are translations of the reaction bead in the plane of the dividing surface and complete regrowing of the molecule starting from the restricted bead. Subsequently, the transmission coefficient is calculated by standard MD in the NVE ensemble. The beads are given independent velocities, corresponding on average to the desired temperature, by sampling from the Maxwell-Boltzmann distribution.

In the Bennett-Chandler approach it is sufficient to assign the barrier position q^* inside the barrier region. The result of the scheme does not depend on the specific location, although the statistical accuracy does. If the dividing surface is not at the top of the barrier the probability of finding a particle will be higher than at the optimal q^*, but the fraction of the particles that actually cross the barrier will be less than predicted by transition state theory.

Importance-sampled MD at infinite dilution

The approach of $\kappa(t)$ to its plateau value can be quite slow [**?**]. Moreover, in the case of diffusive barrier crossings the transmission coefficient is quite small and as a consequence many trajectories have to be generated for an accurate value of κ. The Bennett-Chandler approach becomes inefficient for systems with low transmission coefficients because the scheme employs the noisy θ-function to detect what state the system is in [**?**]. The scheme can be improved by constructing a more continuous detection function. More importantly, using the free energy we can compensate approximately for the effect of the free energy barrier. This leads to a more or less uniform tagged-particle density distribution over the entire range of q. However, only trajectories starting in the barrier region yield relevant information and therefore a weighting function $w(q)$ is applied, restricting the sampling to the barrier region.

A general expression from transition state theory for the rate of hopping from region A to region B over a barrier is [**?**]:

$$k_{A\to B} = -\left\langle \dot{q}(0)\, n_A(t)\, \frac{\partial \chi(q(0))}{\partial q} \right\rangle \tag{16.25}$$

where $\chi(q)$ is a dimensionless function describing the initial distribution function

$$\rho(q, t=0) = \rho_{\mathrm{eq}}(q)\,\chi(q) \tag{16.26}$$

The initial distribution $\chi(q)$ can be approximated well by the steady-state distribution determined from the Fokker-Planck equation

$$\chi(q) = \frac{1}{\langle n_A\rangle}\left[1 - \frac{\int_{q_A}^{q} e^{\beta F(q')}\, dq'}{\int_{q_A}^{q_B} e^{\beta F(q')}\, dq'}\right] \tag{16.27}$$

and varies rapidly with q in the barrier region and slowly elsewhere, so that $\chi'(q)$ selects initial configurations in the barrier region

$$\frac{\partial \chi(q)}{\partial q} = -\frac{1}{\langle n_A\rangle}\frac{e^{\beta F(q)}}{\int_{q_A}^{q_B} e^{\beta F(q)}\, dq} \tag{16.28}$$

We choose

$$n_A(q) = 1 - \frac{\int_{q_A}^{q} e^{(a-1)\beta F(q')}\, dq'}{\int_{q_A}^{q_B} e^{(a-1)\beta F(q')}\, dq'} \tag{16.29}$$

$$w(q) = e^{a\beta F(q)} \tag{16.30}$$

$$\pi(q) \propto e^{(a-1)\beta F(q)} \tag{16.31}$$

where $a > 0$ is a biasing parameter, leading to

$$k_{A\to B} = \frac{1}{\langle n_A \rangle} \frac{\left\langle \int_0^\infty \dot{q}(t)\,\dot{q}(0)\,\frac{w(q(t))}{w(q(0))}\,\frac{e^{-\beta F(q(t))}}{e^{-\beta F(q(0))}}\, dt \right\rangle_\pi}{\int_{q_A}^{q_B} e^{\beta F(q)}\, dq \int_{-\infty}^{\infty} e^{-\beta F(q)}\, dq} \tag{16.32}$$

Although Eq. 16.32 can be considered a TST method using a more continuous "detector" function than the noisy θ-function, it can also be viewed as an MD method in a different, more convenient ensemble $\pi(q) \propto w(q)e^{-\beta F(q)}$. Starting configurations are sampled in the π-ensemble and subsequently a weighted velocity autocorrelation is computed. Note that the dynamics is still generated using the microcanonical ensemble (conventional MD). Importance-sampled MD is especially applicable to systems with erratic free energy landscapes, e.g. multiple barriers of possibly different heights.

Dynamically corrected transition state theory at nonzero loading

The extension of dcTST to finite loading is nontrivial. Conventional methods use a hierarchical approach to compute elementary hopping rates $k^{\text{iso}}_{A\to B}$ between *isolated* cages A and B for use in a subsequent kMC scheme to obtain diffusion coefficients. However, the fundamental question is whether it is *possible* to compute an elementary hopping rate $k^{\text{iso}}_{A\to B}$, in which the contributions of other cages are separated from the contribution of only the cages A and B. Let us consider the class of window/cage-type systems (e.g. methane in LTA) where the barriers are *entropical* in nature. At nonzero loading a molecule hopping from A to B induces a vacancy. While in principle a particle originating from any of the surrounding cages could fill the vacancy, hierarchical approaches will allow only a molecule from B to return to A (e.g. by blocking all windows except the window between cages A and B). The fundamental assumption of kMC (no two jumps can occur at the same time) artificially suppresses these correlated jumps, and we are not aware of a scheme that result in effective kMC hopping rates that regain those correlations.

Beerdsen et al. [**?**] proposed a method to compute diffusivity values directly in systems with high free energy barriers (e.g. cage/window-type zeolites). Here, long time, large distance memory effects are negligible, because once a molecule jumps thermal equilibration takes place and next-nearest cage correlations are rare. It is therefore sufficient to include correlations *during* the jump across the barrier. Hence, we compute

$$D(c) = \frac{1}{6} k^{\text{eff}}_{A\to B}(c)\,\lambda^2 \tag{16.33}$$

$$k^{\text{eff}}_{A\to B}(c) = f(c)\, k_{A\to B}(c) \tag{16.34}$$

where c denotes the loading in molecules per unit cell, or mol/kg. But rather than attempting to compute $k_{A\to B}(c=0)$ or $k^{\text{iso}}_{A\to B}$ from a molecular simulation and the correlation factor $f(c)$ from a coarse-grained kinetic Monte-Carlo method, Beerdsen et al. compute $k^{\text{eff}}_{A\to B}(c)$ *directly* from a molecular simulation.

> $k^{\text{eff}}_{A\to B}(c)$ is the hopping rate of a single tagged particle at an average loading c from cage A to cage B under the influence of an external field exerted caused by the molecular sieve *and* the other $N-1$ particles.

By including the nearest neighboring cages, all relevant short-time correlations are properly captured, including the dominant short-time back-correlation effects due to particle-particle interactions. Correlations at much longer times than $1/k$ are negligible in cage/window-type systems. The computation once again consists of two parts

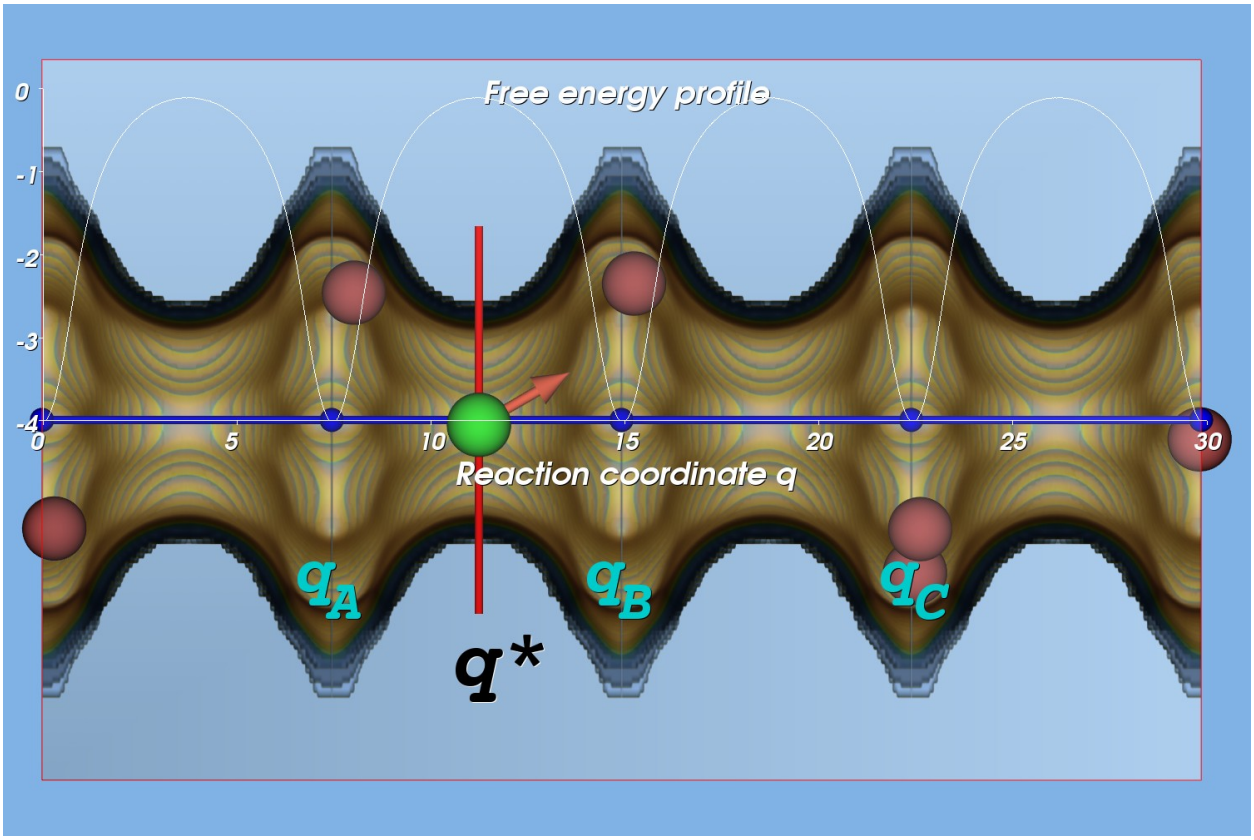


Figure 16.2: A typical snapshot of a tagged methane particle (green color) in LTL-type zeolite restrained to the barrier q^* surface at an average loading of 3 methane molecules per unit cell (there are two parallel channels per unit cell) at 300K. Four unit cells of 7.474 Å in length are shown. The constrictions are caused by the 12-T-membered rings, which form free energy barriers impeding diffusion. The free energy profile in dimensionless units at this average loading is plotted in white, where the reaction coordinate is chosen parallel to the channel direction. If the free energy barriers are high enough, diffusion can be considered a hopping process from minima to minima (q_A, q_B, q_C etc).

- The probability density $P_{\in A}(q^*)$ of finding the system at the top can be computed explicitly by computing free energy profiles making use of Eq. 16.21. During an separate MC simulation in the NVT-ensemble at the desired loading we measure the free energy $F(q)$ by using Histogram Sampling (HS). In the HS method, a histogram is made of the particle positions, mapped on the reaction coordinate. From the histogram a free energy profile is computed, by using

$$\beta F(q) = -\ln \langle P(q) \rangle \,. \tag{16.35}$$

  At conditions where conventional MC is still feasible, all particles can be considered equivalent and all contributions can be used.

  When displacement of particles is impeded by high free energy barriers, conventional HS becomes infeasible. A single tagged particle can be biased to achieve improved statistics by using importance sampling. As a biasing potential the Widom Particle Insertion (WPI) profile can be used. WPI uses a probe particle that is inserted at random positions, to measure the energy required for or obtained by insertion of the particle in the system. This energy is mapped onto the reaction coordinate q, using

$$\beta F(q) = -\ln \left\langle e^{-\beta \Delta U} \right\rangle_N , \tag{16.36}$$

to produce a free energy profile, where $\left\langle e^{-\beta\Delta U}\right\rangle_N$ is the average Boltzmann factor over all positions in the slice perpendicular to the reaction coordinate. A "ghost particle" is used as the measuring probe, but the other particles in the system do not feel its presence. At higher loadings, WPI is known to give erroneous results [**?**, **?**] and therefore the WPI method is not used to compute F(q) directly, but rather to estimate the biasing function when needed.

- The particle flux $R(t)$ through the dividing surface can be computed from the fraction of particles starting on top of the barrier that successfully reach cage B. The other particles present in the system influence this fraction. Starting configurations are generated with one particle constrained to the dividing surface and $N-1$ particles moving around (see Figs. 16.2 and 16.3). These configurations are then used to compute the particle flux in unconstrained NVE-MD simulations, starting with velocities sampled from a Maxwell-Boltzmann distribution at the desired temperature.

Fig. 16.3 shows a snapshot of ethane at an average loading of 4 molecules per cage at 750 K in LTA-type zeolite. The lattice, formed by the cage centers, is the three-dimensional cubic lattice. For this snapshot cage B contains more molecules than cage A, and the barrier-molecule has a high probability of recrossing to cage A. The time-dependent transmission coefficient will reach a plateau value κ. However, because during a successful hop cage A donated a particle, while cage B received an additional particle, there is a slightly higher probability for the particle to return to A on a time scale *larger* than the thermalization time. However, for our systems this effect is negligible. Note that during the computation none of the windows is blocked and simultaneous jumps (e.g. from cage C to cage A, and cage D to cage B) are allowed.

The extension of the importance-sampled MD method Eq. 16.32 to non-zero loading is similar. The method can be summarized as follows.

- The free energy profiles at the desired average loading are measured as described above.
- The free energy minima q_A and q_B, and the corresponding hopping lattice are identified.
- A biasing profile $w(q)$ is constructed ranging from q_A to q_B using Eq. 16.30.
- Starting configurations are sampled in the interval q_A to q_B with the bias potential $w(q)$ operating on the tagged particle, leaving the others $N-1$ free to move (unbiased).
- These starting configurations of N particles are integrated using MD for short times $t_{\max}$, and Eq. 16.32 is evaluated. The time $t_{\max}$ is chosen such that the integral appearing in Eq. 16.32 has converged. The trajectories are stopped after $t_{\max}$ time has elapsed or when $q < q_A$ or when $q > q_B$.

As mentioned, in general, the reaction coordinate q is a function of the configuration of the whole system. For dcTST simulations at a certain loading, we choose the reaction coordinate as the position of one of the atoms of the *tagged* molecule [**?**]. Although it can not be excluded that better reaction coordinates exists, for physical reasons our choice seems optimal. The diffusion mechanism is divided in two parts. The first is a static term, corresponding to locations of preferable adsorption sites and estimations of free energy barriers in between, the latter (or actually the inverse of the transmission coefficient: the recrossing) corresponds to collision frequencies, which generally increase with loading. As such the dcTST method is able to explain different diffusion regimes over loading, and provides insight into the mechanisms behind an increase or decrease in diffusivity with loading [**?**].

From hopping rates to diffusion coefficients

A lattice is an infinite periodic array of points. In principle the lattice is completely described by three basis vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}$, and the angles between these vectors α, β and γ. The lattice can be mapped into itself by translation operations and by other symmetry operations. The Wigner-Seitz cell enclosing a point is the region of space that is closer to that lattice point than to any other lattice point. There is only one lattice point in the primitive unit cell thus formed. The symmetry of the unit cell is used to classify structures and how they fill space. In two dimensions there are only five lattices, called Bravais lattices, that fills space

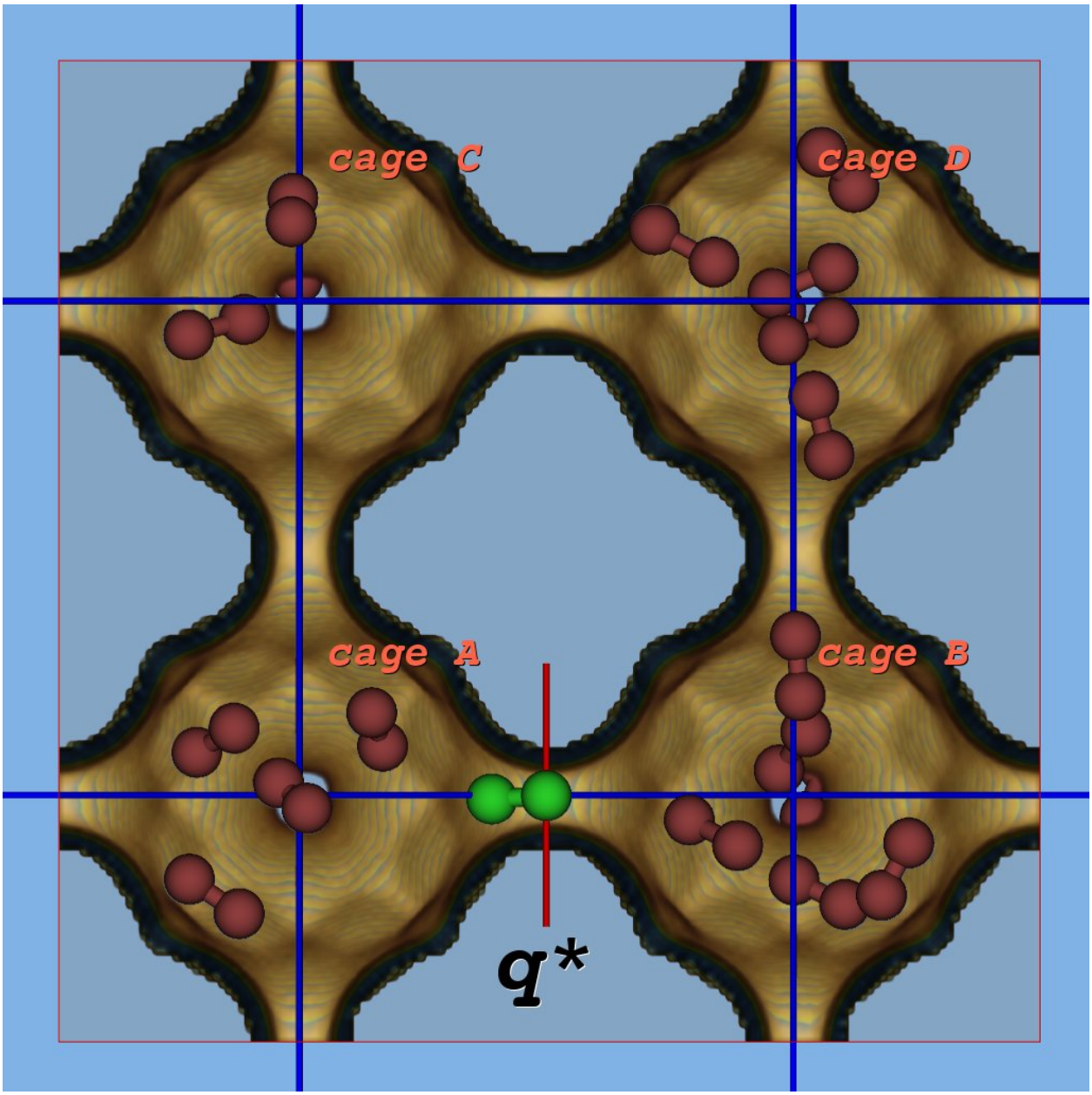


Figure 16.3: A typical snapshot of ethane ($CH_3 - CH_3$) in LTA-type zeolite at an average loading of 4 molecules per cage at 750K, constraining one tagged molecule at the dividing surface q^*. The hopping events are coarse-grained on a lattice spanned by the cage-centers.

and defined by how you rotate the cell content and get the same cell back, and if there are any mirror planes within the cell (the square, hexagonal, rectangular, center rectangular, and oblique lattices). In three dimensions there are seven lattice systems (triclinic, monoclinic, orthorhombic, tetragonal, cubic, trigonal, hexagonal). The general lattice is triclinic, all others are derived by putting constraints on the triclinic lattice. Note that we now have different lattice types: primitive (one lattice point), body-centered (two lattice points), face-centered (four lattice points), side-centered (two lattice points).

For use in the dcTST method, it turns out that often lattices are needed of two or more different types,

i.e. a quasiperiodic nonrandom assembly of two or more types that fill space. These lattice are not Bravais lattices. In analogy to crystallographic lattices, one may fill space by tessellation or tiling a finite number of proto-tiles to fill space without gaps or overlap. These Wigner-Seitz cell (or Voronoï cell) cells form the lattice tiling.

A random walk is a simple mathematical model for the movement of a particle on a lattice under the influence of some random or stochastic force affecting its direction of motion. Let k_i be the average frequency that a random walker (an atom or molecule) jumps for lattice vector $\boldsymbol{\lambda}_i$, and $\mathbf{r}(t)$ is the position of a particular random walker. The total jumping frequency k is related to the specific jumping frequencies k_i for a given structure by a summation over the lattice connectivity Z:

$$k = \sum_{i=1}^{Z} k_i \tag{16.37}$$

In the limit of infinite dilution there are no interparticle correlations and the particles perform a random walk on a lattice spanned by for example the cage-centers. For Bravais lattices, the transmission rates are then easily converted to self-diffusion coefficients by for example

$$D_S = \frac{1}{2d} k\lambda^2 \tag{16.38}$$

where d denotes the spatial dimensionality, for the one dimensional lattice, two dimensional square and hexagonal lattice, and for the three dimensional simple cubic lattice, respectively.

For non-Bravais lattices, the diffusion coefficient can be easily evaluated numerically for any nonrandom quasiperiodic tessellation, but often it is possible to find analytical expressions. Consider an hop from A to C with an intermediate state B. We have

$$\frac{P_{B\to A}}{P_{B\to C}} = \frac{k_{B\to A}}{k_{B\to C}} \tag{16.39}$$

$$P_{B\to A} + P_{B\to C} = 1 \tag{16.40}$$

and therefore

$$P_{B\to C} = P_{B\to A}\frac{k_{B\to C}}{k_{B\to A}} \tag{16.41}$$

$$= (1 - P_{B\to C})\frac{k_{B\to C}}{k_{B\to A}} \tag{16.42}$$

$$\tag{16.43}$$

from which $P_{B\to C}$ can be solved

$$P_{B\to C} = \frac{k_{B\to C}}{k_{B\to A} + k_{B\to C}} \tag{16.44}$$

The hopping rate from A to C is the hopping rate from A to B times the probability $P_{B\to C}$ to go from B to C

$$k_{A\to C} = k_{A\to B} P_{B\to C} \tag{16.45}$$

In general a serial combination of hops is then described by

$$k_{A\to C} = \frac{k_{A\to B} k_{B\to C}}{k_{B\to A} + k_{B\to C}} \tag{16.46}$$

Eq. 16.46 proves convenient for finding the relation between hopping rates and diffusion coefficient on non-Bravais lattices. De relation between $k_{A\to B}$ and $k_{B\to A}$ is given by

$$\frac{k_{A\to B}}{k_{B\to A}} = \frac{\langle n_B \rangle}{\langle n_A \rangle} \tag{16.47}$$

where $\langle n_A \rangle = 1 - \langle n_B \rangle$ is the equilibrium mole fraction of particles in state A. For a symmetric barrier $\langle n_A \rangle = \langle n_B \rangle$ and therefore $k_{A \rightarrow B} = k_{B \rightarrow A}$.

In ERI-type lattice the diffusion in the xy-plane occurs isotropicly on an hexagonal lattice

$$D_{xy} = \frac{1}{4} k_{xy} \lambda_{xy}^2 \tag{16.48}$$

with λ_{xy} the lattice displacement distance, and k_{xy} the corresponding hopping rate. The z-diffusion is dependent on the hopping in the xy plane. The lattice displacement vector $\boldsymbol{\lambda}_z$ is orthogonal to $\boldsymbol{\lambda}_r$, and using Eq. 16.46 plus the symmetry of the lattice, we find

$$D_z = \frac{1}{2} \frac{k_{xy}\, k_z}{k_{xy} + k_z} \lambda_z^2 \tag{16.49}$$

To convert the hopping rate in CHA-type zeolites to a diffusion coefficient we note that the lattice is slightly distorted from a cubic lattice. The orientational averaged diffusion coefficient is *not* affected in CHA-type lattices but the individual components are, albeit that the distortion effect for the CHA-type lattice is negligibly small (smaller than 2%). Therefore diffusion in CHA-type zeolite can be considered isotropic in practice.

16.1.2 Free energy profiles

Free energy profiles can obtained using various methods. The 'FreeEnergy' method uses Widom particle insertion and maps the values onto a one-dimensional reaction coordinate. The mapping can be controlled using 'FreeEnergyMappingType'.

- 'FreeEnergyMappingType X,Y,or Z'
  All positions in the unit-cell are mapping to X,Y, and Z, respectively.
- 'FreeEnergyMappingType A,B,or C'
  All positions in the triclinic unit-cell are mapping to A,B, and C, respectively.
- 'FreeEnergyMappingType CageToCage' The free energy is mapped from to volumes of cage A and cage B. The cages are read from the directory 'structures/zeolites/centers/' using the file corresponding to the framework name. Every Widom insertion is tracked to the closest cage, and if the position is closest to either cage A or cage B the value is used, otherwise it is counted with a value zero. The mapping is an orthogonal projection onto the line connecting the centers of cage A and cage B.

```
SimulationType                  FreeEnergy

# Recrossing/Sampling parameters
=============================================
FreeEnergyMappingType           A
WidomParticleInsertionComponent 0
```

16.1.3 Transmission coefficient

Sampling configurations

Starting configuration can be sampled using standard Monte Carlo, an example 'simulation.input' would be:

```
SimulationType                  MC
NumberOfCycles                  100000

# Recrossing/Sampling parameters
```

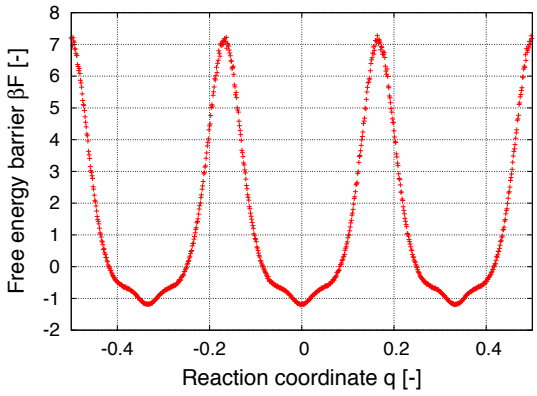


Figure 16.4: The free energy profile of methane in 3x3x3 CHA (monoclinic) at 600K using the 'FreeEnergyMappingType A'-mapping.

```
=============================================
BarrierPosition                  0.5     0.25    0.25
PutMoleculeOnBarrier             yes

WritedcTSTSnapShotsToFile        yes
WritedcTSTSnapShotsEvery         100

Components:
==========================
component 0 MoleculeName                     ethane
            MoleculeDefinition               TraPPE
            StartingBead                     0
            TranslationProbability           1.0
            TranslationNormal                1 0 0
            TranslationDirection             yz
            RegrowInPlaceProbability         1.0
            CreateNumberOfMolecules          0
```

The 'BarrierPosition' defines the location of the barrier in fractional coordinates. The statement 'PutMoleculeOnBarrier yes' generates a molecules with it starting bead at the barrier position. Note that no other molecules need to be generated for this component. The movement of the molecule needs to be restricted such that the tagged molecule never leaves the barrier plane. in this case only translation in the xy-plane is performed and the molecule is regrown in-place, i.e. the starting bead remains unchanged. The transmission coefficient is temperature dependent because the distribution of sampled configuration is temperature dependent. So for each temperature a separate sampling simulation needs to be performed. The statement 'WritedcTSTSnapShotsToFile yes' causes COTA to output sampled configuration snapshots to the file 'dcTST_starting_configurations/System[0]/configurations.data'. For dcTST at infinite dilution the sampling generates independent configuration after a small amount of cycles. So 'WritedcTSTSnapShotsEvery 100' would be sufficient. At finite loading, a second component of the same type of molecule needs to be defined which movement is unrestricted. The 'WritedcTSTSnapShotsEvery' needs to be increased because correlations between successively generated snapshots decay more slowly.

The file 'dcTST_starting_configurations/System[0]/configurations.data' looks like:

```
#NumberOfAdsorbateMolecules:1
#NumberOfCationMolecules:0
```

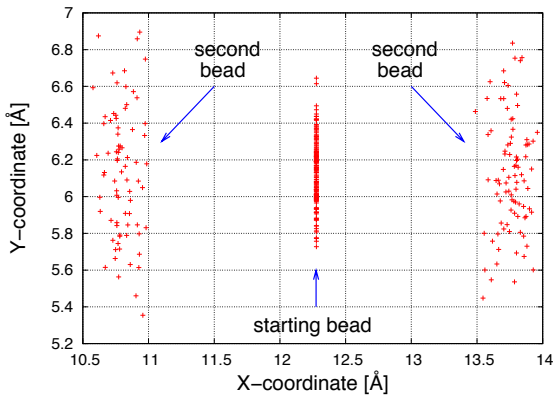


Figure 16.5: Sampling dcTST configurations: ethane at 600K in LTA. The unit cell size of LTA is 24.555, the barrier here is $\{\frac{1}{2}, \frac{1}{4}, \frac{1}{4}\}$. The first bead of ethane is restricted to the barrier plane. The second bead is found away from the plane on both sides, restricted only by the harmonic bond potential ($r_0 = 1.54$ Å).

```
#NumberOfFrameworkAtoms:576
#Barrier: 12.277500 6.138750 6.138750
12.277500 6.093801 5.848286 0.0 0.0 0.0
13.854630 5.946754 6.073181 0.0 0.0 0.0

#NumberOfAdsorbateMolecules:1
#NumberOfCationMolecules:0
#NumberOfFrameworkAtoms:576
#Barrier: 12.277500 6.138750 6.138750
12.277500 6.229654 6.163357 0.0 0.0 0.0
13.730617 6.480124 6.580316 0.0 0.0 0.0

.................
```

It first lists the amount of adsorbates, cations, and framework atoms. Next it defines the barrier position for this particular configuration. For a flexible framework the position of the barrier can be slightly different for each sampled configuration due to center of mass drift and movement of the framework. All four lines are preceeded by the '#' sign that is interpreted by gnuplot as comment. The positions in the file can therefore easily be viewed using gnuplot to show the sampled points. It is important to check that indeed the starting bead remains in the TST-plane, see Fig. 16.5.

Generating trajectories

The transmission coefficient is computed by

$$\kappa = \frac{\langle \dot{q}(0)\,\delta(q(0) - q^*)\,\theta(q(t) - q^*)\rangle}{\left\langle \frac{1}{2}\,|\dot{q}(0)|\,\delta(q(0) - q^*)\right\rangle} \tag{16.50}$$

Here, $\dot{q}(0)$ is the velocity of the reaction coordinate, e.g. the starting bead, at time 0. The starting velocities for all beads are drawn from a Maxwell-Boltzmann distribution. The term $\delta(q(0) - q^*)$ simply means that all starting positions are placed on top of the barrier q^*. The term $\theta(q(t) - q^*)$ is 1 when at time t the reaction coordinate is right of the barrier, and 0 otherwise. Basically, the procedure is to place a particle on top of the barrier, give it a velocity from the Maxwell-Boltzmann distribution pointing to cage B and add the velocity and θ-terms to the averages. In COTA the algorithm is slightly more advanced then this, because not only the simulation pointing to B is performed, but at the same time also the same with all

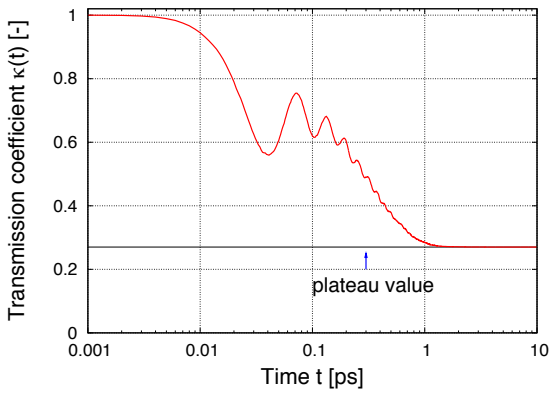


Figure 16.6: Transmission coefficient: ethane at 600K in LTA.

velocities inversed. Only when the first ends up in B, and the latter in A is the contribution positive. A negative contribution is obtained when the opposite occurs: velocity pointing to B ends up in A, and velocity pointing to B ends up in A. The advantage of this method is that the other two situations result in *exact* cancellation.

For efficiency reasons, e.g. when sampling is quite expensive, it might be useful to use one sampled configuration several times using different initial velocities. This can be controlled using 'NumberOfVelocities'. The termination of the trajectories can be controlled with 'MaxBarrierTime', the maximum time of the trajectory in picoseconds, and 'MaxBarrierDistance', the distance in Å from the minimum free energy barrier to the minimum free energy valley. An example 'simulation.input' for ethane in LTA:

```
SimulationType                  BarrierRecrossing

# Recrossing/Sampling parameters
================================================
BarrierPosition                 0.5     0.25    0.25
BarrierNormal                   1 0 0
MaxBarrierDistance              6.13875
MaxBarrierTime                  10.0
NumberOfVelocities              10
```

COTA will generate a file 'TransmissionCoefficient/System[0]/kt.data' with column 1 the number of the iterations, column 2 the time in picoseconds, and column 3 the value of Eq. 16.50. An example is plotted in Fig. 16.6 for ethane in LTA. The transmission coefficient starts at a value 1. The reaction coordinate is initialized with a velocity pointing to region B and therefore a molecule is always found in region B after a single integration step. Some of the trajectories will end up in region A lowering the transmission coefficient, but eventually a plateau value will be reached. This plateau value is the value of interest. Note the oscillatory behavior due the harmonic potential between second bead and the reaction coordinate, i.e. the first bead.

16.1.4 Conversion of hopping rates to diffusion coefficients

Obtaining a TST hopping rate

The 'FreeEnergy/FreeEnergy_methane.dat_0' file (here for methane with a starting bead of 0) can edited to a form which COTA can read to compute the TST hopping rate

$$k_{A\to B}^{TST} = \sqrt{\frac{k_B T}{2\pi m}} \frac{e^{-\beta F(q^*)}}{\int_{\text{cage}A} e^{-\beta F(q)}\, dq} \tag{16.51}$$

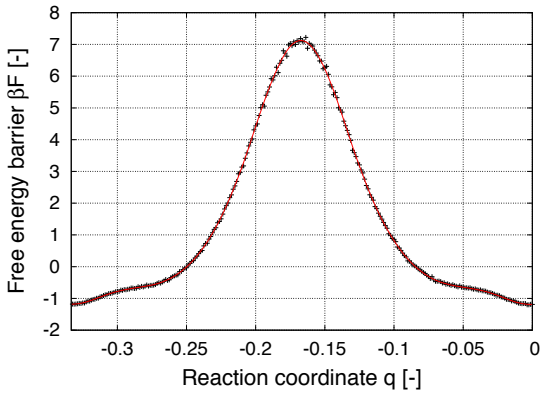


Figure 16.7: Spline fit through the free energy data.

A header needs to be added to the file defining the integration area etc.

```
# 800
# -0.5 -0.16666666666666666 0.16666666666666666666
# -0.33333333333333333333 0.0 9.36632
# -0.5 -0.16666666666666666
.............
```

The first line is the number of data points in the file. The second line expects q_A^*, q_B^*, and q_C^*. The third line state g_A and g_B, i.e. the positions of the minimum free energy valleys, followed by the lattice distance λ in Å, and the last line contains the left and right integration boundary for cage A. COTA fits a spline through the data (See Fig. 16.7). The spline fitting eliminates some of the noise and creates a smooth and continuous function. All operations, e.g. integration, are performed on the spline and not on the scattered data itself. Reading of the profile data and the creation of the spline is done like

```
SimulationType                  MC
NumberOfCycles                  0

Components:
==========================
component 0 MoleculeName                    methane
            MoleculeDefinition              TraPPE
            StartingBead                    0
            BiasingProfile                  Profile.dat
```

where 'BiasingProfile' is the filename of the profile data. Note that this biasing potential can also be used like this for Umbrella sampling and the Ruiz-Montero method. Alternatively, one can use programs like Mathematica to perform the evaluation of Eq. 16.51.

The output is a file named for example 'BiasingSpline_methane_0.dat' (depending on the molecule and the starting bead), and contains a header like

```
# Mass reaction bead:    16.042460000000 [au]
# |v|=Sqrt(k_B T/(2.0*PI*Mass)):   157.308562792211 [m/s]
# P(q*) dq:      484995.816114 [1/m]
# k^TST= |v| P(q*) dq, i.e. the TST hopping rate:      76293994.7931 [1/s]
# D^TST (cubic):    1.150032837e-10 [m^2/s]
```

from which k^{TST} can be extracted. The file contains the spline data, column 1: the x-coordinate, column 2: the spline evaluation βF, column 3: $\exp(-\beta F)$, and column 4, 5, 6 contains the first, second, and third derivative, respectively.

Part IV

Utilities

Chapter 17

Visualization

17.1 Making movies

17.1.1 Combining pictures into a movie

```
ffmpeg -f image2 -i %3d.jpg -sameq movie.mpg
```