CAST - Conformational Search and Analysis Tool

Manual for developers and programmers

DEV MODE! THIS MANUAL IS NOT DESTINED TO BE RELEASED!

Version 3.1
October 24, 2016

WE SHOULD REALLY PUT THE CAST ICON HERE!

Contents

List of Figures

1 Table of Abbreviations

AMBER	Assisted Model Building with Energy Refinement
AMOEBA	Atomic Multipole Optimized Energetics for Biomolecular Applications
CAST	Conformational Analysis and Search Tool
CHARMM	Chemistry at Harvard Macromolecular Mechanics
DFT	Density Functional Theory
DFTB	Density Functional Tight Binding
DOF	degree of freedom
dRMSD	distance root-mean-square deviation
DS	Diversification Search
FEP	Free Energy Pertubation
GAFF	General Amber Force Field
GCC	GNU Compiler Collection
GPU	graphics processing unit
MC	Monte-Carlo
MCM	Monte-Carlo with Minimization
MD	Molecular Dynamics
MOPAC	Molecular Orbital Package
MPI	Message Parsing Interface
NEB	Nudged Elastic Band
NMR	nuclear magnetic resonance
OPLS-AA	Optimized Potentials for Liquid Simulations All-Atoms
PBC	Periodic Boundary Conditions
PCA	Principal Component Analysis
PDF	probability density function
PES	potential energy surface
PME	Particle Mesh Ewald
RMSD	root-mean-square deviation
SAPT-FF	Symmetry Adapted Perturbation Theory based Force Field
SPME	Smooth Particle Mesh Ewald
TS	Tabu Search
US	Umbrella Sampling
VdW	Van-der-Waals
VMD	Visual Molecular Dynamics
WHAM	Weighted Histogram Analysis Method

2 Preface

The CAST allows the accurate treatment of large and flexible (macro-)molecular systems. For the determination of thermally accessible minima CAST offers the newly developed Tabu Search (TS) algorithm[1], as well as Monte-Carlo (MC)[2], Monte-Carlo with Minimization (MCM)[3] and Molecular Dynamics (MD)[4] implementations. For the determination of reaction paths CAST provides the PathOpt[5], the Nudged Elastic Band (NEB)[6] and the Umbrella Sampling (US)[7] approach. Access to free energies is possible through the FEP approach. Along with a number of standard force fields, a newly developed Symmetry Adapted Perturbation Theory based force field (SAPT-FF) is included. Semi-Empirical computations are possible through DFTB+[8] (Density Functional Tight Binding) and Molecular Orbital Package (MOPAC)[9,10] interfaces. For calculations based on Density Functional Theory (DFT), a Message Parsing Interface (MPI) to the GPU accelerated TeraChem[11] program is available. For more information on CAST see [12].

3 Compiling CAST

The CAST source code is not openly distributed. Currently, CAST in principle has no external dependencies. The Particle Mesh Ewald (PME) code however at this point still depends on FFTW libraries and is therefore not enabled by default. If you need to perform simulations with PME, please contact the CAST developers. Compilation was verified using Microsoft Visual Studio 2015 Update 1 on Windows 10 and GCC 5.3 on SuseLinux. A makefile for Linux operating systems is part of the source code repository.

The source code is currently kept on a private GitHub repository. If you feel that you are entitled to access to the repository, please contact the CAST developers (see section 11) and get an invitation.

4 Installation

CAST is usually distributed as a precompiled executable for Linux and Windows. Currently CAST has no external dependencies.

The Windows binary can be used with any Windows operating system starting from Windows 7. No further requirements exist. On Linux, precompiled binaries are statically linked. FFTW 3.4 libraries have to be set in the *PATH* variable of the Linux shell:
LD_LIBRARY_PATH „Path to FFTW lib dir“

5 General Structure and Usage

CAST features several main computation methods which can be combined with force fields, semi-empirical or DFT methods via several interfaces. Input file formatting and available commands are discussed in the following paragraphs.

5.1 Configuration File

A file named „CAST.txt“or „INPUTFILE“ can be used to change the configuration options of CAST. It contains option keywords followed by one or more appropriate values. The keywords are case sensitive. Comments can be included by starting the line with a „#“. The variables are usually of type integer, floating point or boolean (booleans currently being either „0“ and „1“ or plain text „true“ and „false“ without quotation marks).

The following input commands are compulsory for CAST to work and have to be set for every calculation.

Variable	Effect	Default
verbosity	Amount of CAST output	1
cores	Number of OpenMP threads	1
name	Name of input file	none
outname	Name of output files	none
input-type	Format of coordinate file	TINKER

The keyword for the path of the used coordinate file is "name". The variable “cores” controls the number of threads if multi-threated CAST is used (i.e. if CAST has been compiled with OpenMP[13]). “outname” defines the name of the outputfile with information regarding the calculation. The switch controlling how much information CAST will print to the console is named "verbosity". Proper values are positive integral numbers where low numbers indicate less information than higher numbers do. Suggested values for productive use are 1, 2 or 3. In general, lower numbers will yield less obvious output which is more suitable for processing (since there are less superficial descriptors printed). Higher values may slow down program execution but provide detailed insight into the program's execution. The detailed effect of the numerical verbosity setting depends on the chosen task.

An example input for a single point energy calculation with Optimized Potentials for Liquid Simulations All-Atoms (OPLS-AA) Force Field[14,15] is given below:

```
1  verbosity          3
2  name               inputstructure.xyz
3  inputtype          TINKER
4  outname            thisIsTheOutputFile
5  cores              4
6  task               SP
7  interface          OPLSAA
```

A more explicit, commented version of the INPUTFILE containing all parameters is distributed with CAST.

5.2 Force fields

CAST features four different force field implementations:

- Optimized Potentials for Liquid Simulations All-Atoms (OPLS-AA)[14,15]
- Chemistry at Harvard Macromolecular Mechanics (CHARMM)[16]
- Assisted Model Building with Energy Refinement (AMBER)[17]
- Atomic Multipole Optimized Energetics for Biomolecular Applications (AMOEBA)[18,19]

With the exception of the AMOEBA force field, the non-bonded parts of the force fields have been parallelized using the OpenMP programming model. Furthermore, the FEP and Smooth Particle Mesh Ewald (SPME) methods can only be used with OPLS-AA, CHARMM and AMBER. The force fields have to be in a format similar to the one used by the TINKER[20] program. They may be obtained from the TINKER[20] website.

5.2.1 AMOEBA and short range correction

Blalala yaddayaddayadda... insert stuff here.

5.2.2 Smooth particle Mesh Ewald

SPME[21] allows the treatment of full electrostatic in periodic systems. Essential for the use of the SPME are applied periodic boundary conditions. SPME uses the FFTW 3.4 library for the fourier transformations of the charge grid (see Installation details). SPME control features the following parameters:

variable	effect	default
PME	Switches pme on or off 0 = off, 1 = on	0
PMEspline	Spline order used in SPME	5
PMEtreshold	Affects the value of the ewald coefficient	0.00000001

WE NEED TO ADRESS THE FFTW LIB ISSUE!

5.3 Energy Interfaces

The energy interfaces are the main parts of CAST for the choice of the underlying computational method. Via the interface the different force fields as well as the interfaces to external programs can be accessed. For semi-empirical calculations an interface to MOPAC (2012) (serial[9] and parallel[10]) is present, for DFT computations the graphics processing unit (GPU) accelerated program TeraChem[11] can be accessed. Both programs are not part of the CAST distribution and the authors are not responsible for access to those programs. MOPAC[9,10] is freely available from the MOPAC website. TeraChem[11] can be purchased at the PetaChem website. The following interfaces and keywords are available:

Interface	Type	Further input
OPLS-AA	internal	parameterfile
AMBER	internal	parameterfile
CHARMM22	internal	parameterfile
AMOEBA	internal	parameterfile
SAPT-FF	internal	parameterfile and Spackman-input
MOPAC	external	MOPAC variables
TeraChem	external	TeraChem input

In contrast to the force fields, the MOPAC and TeraChem interfaces do not need correct force field parameters. The coordinate file has to be in TINKER[20] format, however, it can be generated with arbitrary parameters.

5.3.1 MOPAC

MOPAC is accessed via system call. The MOPAC interface expects the MOPAC executable path to be either „\opt\mopac\MOPAC2012.exe“on Linux (yes, the executable default fileending is .exe even on Linux-Systems) or „C:\Program Files\mopac\MOPAC2012.exe“ on Windows systems by default. The path can be changed using the config-variable „MOPACpath“.
The following keywords are handed over to MOPAC. They are controlling the calculation and they can be adjusted via the „MOPACkey“parameter. If the value of the keyword „MOPACdelete“ is set to 0, CAST will not delete the temporary input and output transfer-files written by CAST and MOPAC.

variable	effect	default
MOPACkey	Input parameters, see MOPAC manual	PM7 MOZYME
MOPACpath	Path to MOAPC executable	see text
MOPACversion	Version of MOPAC (2007, 2012)	MOPAC2012
MOPACdelete	Delete MOPAC temporary files 0=no, 1=yes	1

Sometimes, when performing extended calculations such as MD Simulations, MOPAC calls may fail. For this reason, CAST keeps a failcounter. If more than 1000 MOPAC calls have failed, CAST will abort. This implies that if only a single MOPAC call fails, CAST will continue. A short error message can in this case be found in the CAST output (independent of the verbosity setting).

5.3.2 TeraChem

TeraChem[11] is accessed via MPI[22] when the keyword „TERACHEM“ is specified in the energy interface. All further parameters regarding basis set, functional and so on have to be set in an extra file readable by CAST. CAST then transfers the input parameters to TeraChem via MPI. The syntax for the TeraChem input is such that each string needs to be set in its own line. Otherwise it's identical to the TeraChem input. An example is shown below:

```
1  basis
2  cc-pvdz
3  charge
4  0
5  method
6  b3lyp
7  dftgrid
8  2
9  dftd
10 d2
```

The name for the TeraChem input has to be „*CAST_TERACHEM_OPTIONS.txt*“.

5.4 Coordinates file

5.4.1 TINKER (Standard)

CAST makes use of the TINKER[20] style .xyz files. This format has the advantage of carrying more information in the file than the standard xyz format. Programs that are able to read and/or write TINKER style files are Molden[23], Visual Molecular Dynamics (VMD)[24], Avogadro[25], ChemBioOffice[26], TINKER[20] and Open Babel[27].

A Tinker format file contains a sequence of structures where the first line of each structure contains the number of atoms while the following lines cover one atom each. The next table explicates the composition of the atom lines.

Column	Width	Justification	Miscellaneous
Number	6	R	
Free	2		
Symbol	3	L	
X coordinate in Å	12	R	6 decimal places
Y coordinate in Å	12	R	6 decimal places
Z coordinate in Å	12	R	6 decimal places
Atomtype	6	R	
Bound atoms	6 (each index)	R	multiple values

Note: For alchemical transformations during Free Energy Pertubation simulations, each line may also contain the „IN“ or „OUT“ (case insensitive) keyword at the end, separated by at least one space from the last bound atom.

5.4.2 AMBER

CAST can also to some extent read structures from AMBER files. In order to do this a few options have to be changed in the CAST input file:

- The name of the input file has to be the .prmtop file that contains the atom types and structure information.
- The input file type has to be set to "AMBER".
- The option "amber_mdcrd" has to be activated and set to the name of the .rst file that contains to coordinates of the atoms.
- The energy interface has to be set to "AMBER", too.
- The parameter file has to be the tinker-adapted amber parameter file "amber99.prm" or - if there are gaff atom types in your structure - to "amber99_gaff_big.prm".

The coordinate file should look like this:

- The first line contains either a title or the number of atoms in the structure.
- If the first line is a title the number of atoms is found in the second line.
- The rest of the lines contain the coordinates. In each line (except the last in case of an odd number of atoms) there are the coordinates of two atoms in the order x y z x y z where each number has a width of 12.
- In the last line there might be box parameters also in the format x y z x y z or not.

If you encounter some problems with AMBER files the most probable reason is that some of the atom types in your prmtop-file are not recognized. This has to be corrected in the CAST code.

GAFF *As AMBER is compatible with gaff CAST can also process structures that contain GAFF atom types. All atom types and structure features from the gaff.dat file are also implemented in the "amber99_gaff_big.prm". However not all of them are recognized by CAST. In order to recognize a gaff atom type you have to do the following:*

- *Go to the file "coords_io_AMBER.cc" and add the atom type to the function "amberUtil::toTinkerType". The number that is returned must be the number of the atom type in the .prm file (something between 3000 and 3070).*
- *Add the number of bonding partners of the atom type in place of the last question mark in the corresponding atom-line in the .prm file.*
- *Add the charge of the atom type in place of the question marks in the corresponding charge-line in the .prm file. In case of doubt use the charge of the corresponding amber atom type (i. e. the type with the same symbol but in capital letters). If you want (or need) to use the charges from the .prmtop file you have to divide them by 18.2223 (see* `http://ambermd.org/formats.html#topo.cntrl` *). You can use the script*

"amber_find_atomnumber_and_charge.py" to find the charge. There you have to put the name of your .prmtop file in the first line and after running it you get a file named "charges.txt" with consecutive numbers, atom types and charges that are already divided by 18.2223.

5.5 Tasks

The main computations in CAST are called *tasks*. They are invoked with the configuration file keyword „*task*“ followed by the identifier of the desired computation. On the following pages outline the individual tasks that can be performed by CAST.

6 Specific Tasks

6.1 SP - Single point energy calculation

A single point calculation calculates the potential energy of the respective system. Single point calculations can be run with either using a force field as well as semi-empirical or DFT methods. In case of force field calculations the output is decomposed into the different force field contributions.

The output consists of the abbreviations for the different force field contributions as well as their respective energy in $\frac{kcal}{mol}$. At the end the total potential energy is given in $\frac{kcal}{mol}$.

symbol	energy
B	Bond energy
A	Angle energy
U	Urey-Bradley energy (CHARMM)
ID	Improper dihedrals energy (AMBER, CHARMM)
IT	Improper torsions energy (OPLS-AA)
V	Van-der-Waals energy
C	Charge energy
SOLV	Solvent energy (if implicit solvent is used)
SUM	Total potential energy

6.2 GRAD - Single point energy and gradient calculation

The result of a gradient calculation is the same as the result for a single point calculation with the addition that the gradients of the atoms are also calculated. The output is can be found as a file where the filename specified by the configuration parameter "output" is amended by the keyword "GRAD". This file contains the information about the gradients with respect to atoms, force field contributions and x-, y- and z-coordinates.

6.3 LOCOPT - Local optimization

CAST is using the L-BFGS algorithm[28] with the More and Thuente line-search[29] for local optimizations if an interface without optimizer is used or additional forces are applied. An example for this is using force fields as the energy interface.
what does additional forces applied mean?
If an interface with included optimizer (MOPAC, TeraChem, Tinker) is used (and if no bias forces are to be applied), the optimization process will be carried

out by those programs and CAST will only retrieve the final geometry and gradient values. The integrated L-BFGS can be adjusted regarding the maximum number of steps performed in the optimization routine via the "BFGSmaxstep" configuration option (default "10000"). The convergence criterion can be altered using "BFGSgrad" (default "0.001").

variable	effect	default
BGFSgrad (float)	Threshold for gradients	0.0001
BFGSmaxstep (integer)	Number of max BFGS steps	10000

6.4 MD - Molecular Dynamics

The MD keyword is used to start a Molecular Dynamics (MD) Simulation[30]. MDs can be performed in NVE, NVT and NPT ensembles. For temperature control a Nose-Hoover thermostat[31,32] is available or the temperature control can be done by direct velocity scaling. Pressure can be controlled by a Berendsen-barostat[33]. If constant pressure is desired, the use of periodic boundary conditions is compulsory. Integration of the equations of motion can be done via Velocity-Verlet or Beeman[34] integration scheme (implementation see [35]). There is also a third integrator named "Beeman_2" available which is a former version of the Beeman integrator. It is not recommended to use this integrator since no one knows what it is doing. Also some options might not be available for the Beeman_2 integrator.

variable	effect	type [default value]
MDsteps	Number of MD steps	integer [10000]
MDintegrator	Type of integrator • 0 = Velocity-Verlet • 1 = Beeman • 2 = Beeman_2	integer [0]
MDveloscale	Remove translation and rotation at every step • 0 = no • 1 = yes	integer [0]
MDthermostat	Nosé-Hoover thermostat • 0 = no • 1 = yes	integer [0]
MDtimestep	Timestep in picoseconds	float [0.001]
MDtrack	Track MD and write output • 0 = no • 1 = yes	integer [0]
MDtrackoffset	Offset for tracked MD output (in frames, every n'th frame will be written)	integer [1]
MDsnap	Number of snapshots	integer [100]
MDsnap_buffer	Number of snapshots saved in memory before written to file	integer [50]
MDsnap_opt	Optimize snapshots with chosen energy interface • 0 = no • 1 = yes	integer [0]
MDheat	Apply Heating • 1st value: snapshot number • 2nd value: temperature at snapshot	integer, float [0 0.0]
MDpress	Enable pressure control • 0 = no • 1 = yes	integer [0]
MDpcompress	i dont know what this is	float [0.000046 (value for water)]

MDpdelay	Barostat delay in picoseconds	float [2.0]
MDptarget	Target pressure in atm	float [1.0]
MDspehrical	Switches for spherical boundary conditions	integer, float, float, float, float, float, float [0, 20.0, 20.1, 10.0, 10.0, 2.0, 4.0] (*details below*)
MDrattle	Switch for Rattle[36] H-bond constraints • 0 = no • 1 = all hydrogen bonds • 2 = specific bonds (*see below*)	integer [0]
MDrattpar	Filename for the parameter file for equilibrium distances of h-bonds	string [none] example or something?
MDrattlebond	Specify certain atom pairs for Rattle[36] when MDRattle = 2 fixation	integer, integer [none]
MDbiased_potential	Switch on and off a biased potential around the active site • 0 = off • 1 = on	integer [0]
MDactive_site	atom numbers of active site (write a new line for every atom)	integer [none]
MDcutoff	inner and outer cutoff for biased potential	integer, integers [none]
MDadjust_by_step	calculate the geometrical center for the active site and the distances to it new for every step • 0 = no • 1 = yes	integer [1]
MDrestart_offset	Offset for restart file writing in frames	integer [0]
MDrefine_offset	Offset for nonbonded list generation	integer [200] wtf is this, clarify
MDresume	Boolean switch for using a restart file to start MD • 0 = no • 1 = yes	integer [0]
MDpre_optimze	Perform Optimization before starting the Simulation • 0 = no • 1 = yes	integer [0]

6.4.1 Heating

Note: The settings for "MDheat" can be applied multiple times. The first number indicates the snapshot number, the second one the desired temperature at this snapshot. If the "MDheat" keyword is specified multiple times heating and cooling can be applied sequentially during a single simulation. CAST reads the "MDheat" keywords from top to bottom and starts with the first one, finishing with the one on the bottom.

```
1   MDheat          0      0.0
2   MDheat       1000    300.0
3   MDheat       5000    300.0
4   MDheat      10000      0.0
5
```

In the above example the simulation starts at 0 Kelvin and is then heated to 300K within the first 1000 steps. The temperature is kept steady for 4000 steps until step 5000 is reached. Now the system is cooled back to 0 Kelvin during steps 5000 to 10000. The user has to take care that enough total steps are performed for the temperature scaling sequence!

There are two possible ways for temperature control in CAST:

If you disable the option "MDthermostat" a simple velocity scaling is used for heating up. After reaching the desired temperature the control mechanism will however be turned off (though it still writes the temperature you expect into the logfile). This can be avoided by using a very low energy gradient for constant temperature for the duration of "constant" temperature. Also setting the start temperature to zero will result in an error because there's a division by zero. So your settings for first heating up and then keeping the temperature constant might be:

```
1   MDheat          0      0.0001
2   MDheat       1000    299.9999
3   MDheat      10000    300
4
```

The other possibility is the Nosé-Hoover thermostat that is used if "MDthermostat" is enabled. It also works together with "MDheat" but it is not completely clear how.

Note: In case you enable logging, the verbosity will be controlled by global verbosity setting. If you, for example, desire a descriptor of columns to be printed at the top of the logfile, chose a verbosity higher than 4.

MD Simulations use the common types of boundary conditions present in CAST. For more information on this, see section 7.

6.4.2 MD with biased potential

By setting the option "MDbiased_potential" from 0 to 1 you can switch on a biased potential. This means that the movement of the atoms depends on the distance of every atom to the active site that is defined by the option "MDactive_site". Atoms near the active site move "normally" whereas the speed of atoms with a larger distance to the active site is scaled down until there's no movement. With the option "MDcutoff" you can modify the strength of the biased potential. The first number is the distance around the active site where no additional potential is applied, the second number is the distance around the active site outside of which there's no movement. During the MD the distances of the atoms to the active site change. If you want to take this into account set the option "MDadjust_by_step" to 1. Then the position of the active site as well as the distances are calculated new for every step. If "MDadjust_by_step" is set to 0 the position of the active site and the distances are calculated only once and stay constant for every atom during the simulation.

If you use the "MDheat" option without switching on the Nosé-Hoover thermostat ("MDthermostat") the temperature (and thus the scaling factor) is only calculated by the velocities for the atoms inside the inner cutoff radius where the movement is not manipulated. The kinetic energy is however calculated for all atoms so they do not fit together directly. The Nosé-Hoover thermostat might not work together well with the biased potential as the temperature and the scaling factor are calculated by the velocities of all atoms after the adjustment according to the biased potential when many of them are zero.

6.5 GOSOL - Global Optimization and Solvation

write something here

6.6 TS, MC - Global Optimization Tasks

CAST can be used to perform Conformational Analysis. To find the lowest possible conformation, either a standard Monte-Carlo-Simulation[2,37] or a Tabu-Search[38–40] based approach can be choosen. The total number of steps for the global optimization routines is set by "Iterations" (default "1000"). CAST will save all minima between E_0 (current lowest energy) and $E_0 + D$ where the value of D is adjusted with the "Erange" keyword (default "0.0"). If the current step does not result in a newly accepted minimum, CAST will select a new starting point. The key "GOfallback" selects either a simple fallback to local / global minimum (value "LAST_GLOBAL", default) or an evolutionary selection algorithm (value "EVOLUTION").

variable	effect	default
Iterations	Number of iterations	1000
Erange	Energy range for output	0.0
GOfallback	Type of fallback	LAST_GLOBAL

6.6.1 Starting point selection

Simple Fallback
This method uses the parameter "GOfallback_limit" to determine how often the program can use a specific minimum as a starting point. If the limit for the last accepted minimum is reached, CAST uses the current "global" minimum instead. If the limit for this minimum has also been reached, CAST stops.

Evolutionary selection
This algorithm selects a new starting point among a limited number of minima N (the limit is set via "GOincluded_minima" and defaults to 10.) A roulette selection algorithm is applied where the fitness of the respective structures is determined based on their energetic rank. A lower L and an upper bound H for the fitness can be specified using "GOfitness_bounds" (default "0.5 1.0") where minimum N has fitness L and minimum 1 has fitness H. The interpolation between those points (1,H) -¿ (N,L) can be controlled by "GOfitness" where the value "LINEAR" (default) denotes linear interpolation while "EXPONENTIAL" activates exponential decay. CAST makes up to 100 attempts to select a minimum which hasn't reached the limit yet and stops if none is found.

variable	effect	default
GOfallback_limit	Number of times a minimum can be used	-
GOincluded_minima	Number of minima to look for new starting point	10
GOfitness_bounds (2 values)	Lower and upper bound for fitness	0.5 1.0
GOfitness	Interpolation between **GOfitness_bounds**	LINEAR

Metropolix Criterion
The global optimization routines in CAST evaluate the energy of a certain conformation (E) using the Metropolis criterion (MEC).

$$R < e^{-\frac{E-E_0}{kT}} \tag{1}$$

with R: Random number ($0 <= R <= 1$) and E: Evaluated energy and E_0: Reference energy and k: Boltzmann constant and T: Temperature

The reference energy E_0 can either be the energy representing the most stable structure (current "global minimum") or the last local minimum energy. The corresponding control option is called "GOmetrolocal" and defaults to 0 (off)

which means that the current global minimum energy is used in the conditional. A value of "1" will make CAST use the "current local minimum" energy (the starting point of the current iteration). If the Metropolis Criterion is not met, the conformation will be discarded. The temperature used is controlled via "Temperature" and defaults to "298.15". It is multiplied by a factor, adjusted via "Tempscale" (default: "1.0").

variable	effect	default
GOmetrolocal	Use global or current local minimum energy as reference; 0 = no, yes = 1	0
Temperature	Temperature in Kelvin	298.15
Tempscale	Multiplication factor for temperature	1.0

6.6.2 MC - Monte Carlo

The MC simulation will move the system randomly across the potential energy surface (PES) and evaluate the reached point either directly or after local optimization. The "MCminimization" option turns minimization on (value "1"; default) or off (value "0").

Move types

CAST can move to the next sampling point during MC in three different ways, controlled via the "MCmovetype" option (default "1"). A value of "2" will make the program carry out the contortion in Cartesian space (where the "MCstep_size" option with a default value of "2.0" will restrict the absolute value of the distortion vector). The value "1" represents direct rotation of randomly selected main dihedral angles (see 1.5). A value of "0" means that the target conformation is not obtained directly by adjusting dihedrals but the conformational change will be achieved by applying quadratic bias potentials on the main dihedrals towards the target conformation during a local optimization process ("MCmax_dihedral" restricts the maximum distortion of a dihedral angle; default "160.0").

MCmovetype value	Move type	Associated options
0	Biased main dihedral optimization	MCmax_dihedral
1	Main dihedral	MCmax_dihedral
2	Cartesian	MCstep_size

The number of distorted dihedrals is selected randomly in case of a "MCmovetype" value of 0 or 1 (biased or direct main dihedral adjustment).

$$N = -log(R) + 1 \tag{2}$$

with R being a random number between 0 and 1 and N being the number of distorted rotated dihedrals.

variable	effect	default
MCminimization	Turn minimization on or off; 0 = off, 1 = on	1
MCmovetype	Movetype to go to next sampling point	1
MCstep_size	Absolute value of distortion vector	2.0
MCmax_dihedral	Maximum distortion of dihedral angle in degre	160.0

6.6.3 TS - Tabu Search

The Tabu Search (TS) process in CAST is essentially an alternating combination of the dimer method[41] and the local optimization process (see section 6.3). If a certain number of TS iterations do not yield a new and acceptable minimum, the Diversification Search (DS) process (MC with minimization is used here) is executed. The option controlling how many steps need to fail before diversification search comes into place is called "TSdivers_threshold" (default "25"). The "TSdivers_iter" option (default "30") represents the number of iterations during one diversification routine. If you want CAST to start with DS iterations instead of the TS iteration you can set "TSmc_first" to "1" (default "0").

variable	effect	default
TSdivers_threshold	Number of failed steps before Diversification Search	25
TSdivers_iter	Number of iterations during diversification	30
TSmc_first	Start with diversification instead of Tabu-Search iterations; 0 = no, 1 = yes	1

6.6.4 Output

The output for verbsity settings of 2 and higher includes 15 columns (with *NA* being the number of currently accepted minima):

- Method (MC, MCM or TS)
- Current iteration
- \ I guess that means blank line, does it?
- Maximum number of iteration
- Index of current minimum in the interval (0, *NA*)

- Current minimum energy (starting point of current step)
- "Transition step" energy (energy after distortion / dimer method without optimization)
- New minimum energy (after optimizing the "transition step" structure).
- Identifier for acceptance (A = accepted; R = rejected)
- Acceptance information
  - ok = new minimum, but not lowest
  - GM = new minimum, lowest
  - energy = rejected because of metropolis criterion
  - broken = Configurational feature of the structure broken
  - tabu = already visited minimum
- Number of accepted minima (*NA*)
- Number of minima within the energy range
- Current temperature used for the metropolis criterion
- Iteration Runtime
- Number of CPU clock ticks required for current iteration

Example:

```
1  MCM; 87/100; 0 -2.0604e+02; -1.8566e+02; -2.0516e02;
2  R (energy) 1 1 (47.04 K, 3.37s (33701 ticks))
3
```

6.7 DIMER - Dimer Method

This task will perform the improved dimer-method for finding transition states[41]. The "dimer size" (magnitude of dimer vector) can be controlled via "DIMERdistance" (default "0.01"). It is possible to adjust the maximum rotational force during the dimer translation. The dimer translation is interrupted and the dimer is rotated into the minimum in case this limit is exceeded. The key controlling this value is called "DIMERtflimit". The maximum number of iterations for the dimer rotation and translation steps can be set in the combined "DIMERmaxit" option, which takes two parameters where the first one limits the number of rotation iterations per translation step while the second one represents the maximum number of translations. The convergence criterion for the dimer rotation (the angle that needs to be undercut) in degrees can be adjusted using "DIMERrotconvergence".

variable	effect	default
DIMERdistance	Controls magnitude of dimer vector	float [0.01]
DIMERtflimit	Controls magnitude of dimer force	float [0.01]
DIMERmaxit	Controls rotation steps per translation and total translation steps	integer, integer [20 100]
DIMERrotconvergence	Convergence criterium for rotation in degree	float [5.0]

6.8 UMBRELLA - Umbrella Sampling

The Umbrella Sampling (US)[42–44] implementation features three reaction coordinates which can be sampled: distances, angles and dihedral angles. Distances can be chosen between any two particles in the system. Bond angles and dihedrals can be defined between any three or four particles in the system. None of the reaction coordinates is limited to existing internal coordinates of the system. The Umbrella Sampling implementation is divided into two parts: equilibration with the applied bias potential and a production run with the applied bias potential. For all reaction coordinates a half harmonic potential is used. The number of steps for equilibration is defined by the „*USequil*“ configuration variable. The number of production steps is defined by the „*MDsteps*“ variable.

variable	effect	default
USequil	number of equilibration steps	integer [0]
USsnap	Offset for snapshots (Snapshot every x steps)	integer [0]
UStorsion	Definition for torsion angle	integer, integer, integer, integer, float, float [none]
USdist	Definition for distance	integer, integer, float, float [none]

The syntax for the definition of the restraints can be seen in the following listing:

```
1  #Torsion restraint    <atom 1> <atom 2> <atom 3> <atom 4> <force>
       <value>
2  UStorsion                1        5        7        9       0.05
       0.0
3  #Distance restraint   <atom 1> <atom 2> <force> <value>
4  USdist                   1        9        10.0     3.0
5
```

Torsion restraints consist of 6 values: the first 4 integers are the index numbers of the atoms which form the dihedral. Indices start at "1". The fifth number is the force constant of the harmonic potential in $\frac{kcal}{deg^2}$ and the last index the desired value of the angle in degree. The distance restraint consist of the two

indices for the atoms, the force constant in $\frac{kcal}{Å^2}$ and the distance in Å. The Umbrella Sampling output is written to a file named "umbrella.txt". The output is formatted for use with the Weighted Histogram Analysis Method (WHAM) program[45,46] for post-processing. For details on the use of WHAM we refer the reader to the corresponding WHAM manual.

6.9 NEB - Nudged Elastic Band Method

write something here

6.10 INTERNAL - Conversion to internal coordinates

write something here

6.11 STARTOPT - I dont even know

write something here

6.12 PROFILE - Repeated Gradient Calculation?

write something here

6.13 FEP - Free Energy Perturbation

Free Energy Pertubation (FEP) allows the alchemical transformation of one moiety into another and the derivation of the free energy change corresponding to the transformation. For FEP calculations the coordinate file has to be slightly modified. CAST uses the dual topology paradigm to define the topology for the chimeric system. In the dual topology paradigm both states of the transformation are present during the simulation. Therefore, the input structure of the starting system has to be extended by the moiety of the final structure. As an example, the transformation of ethane to propane is shown.

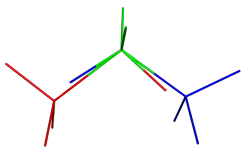

Figure 1: Exemplifying FEP calculations in CAST

The original ethane molecule inputfile is modified by adding another CH3-group + H to one of the ends of the molecule. In the next step the atoms belonging to both the starting and the final step have to be identified (they are marked in green in the figure). Those atoms don't need to be modified in the input file and are always present in the simulation. The atoms marked in blue are the ones belonging to the starting system and are phased "out" during the simulation. All atoms belonging only to the starting point have to be marked with an *IN* in the coordinate file after the bonding partner specification. The atoms belonging to the final state of the system have to be marked with an *OUT* after the bonding partner specification. The modified file is shown in the following:

```
      13
 1  C     0.000000    0.000000    0.000000     26   2   3   4   5    IN
 2  H     0.000000    0.000000    1.089000     1    1  IN
 3  H     1.026720    0.000000   -0.362996     1    1  IN
 4  H     0.513360    0.889165   -0.363000     1    1  IN
 5  C    -0.683537   -1.183920   -0.483333     26   1   6   7   8  12  13
 6  C    -0.683537   -1.183920   -1.933333     26   5   9  10  11  OUT
 7  H    -0.170177   -2.073085   -0.120333     1    5
 8  H    -1.710256   -1.183920   -0.120333     1    5
 9  H    -1.196896   -2.073085   -2.296333     1    6  OUT
10  H    -1.196896   -0.294755   -2.296333     1    6  OUT
11  H     0.343182   -1.183920   -2.296333     1    6  OUT
12  H    -1.539136   -1.480308   -1.088333     1    5  IN
13  H    -0.146459   -0.253674   -0.304136     1    5  OUT
```

Figure 2: Modifies tinker input style for FEP calculations

One condition of the dual topology paradigm is that the atoms only belonging to the start point and end point must not interact with each other during the calculation. CAST takes care of this automatically by excluding all angles, dihedrals or non-bonded interactions involving atom belonging to *IN* or *OUT*.

In order to avoid so called endpoint catastrophies the electrostatic and van der Waals-potentials between incoming and leaving atoms and static ones is modified.[35][47]

The softcore electrostatic potential is:

$$U_{el}(ab) = \lambda_{el} \cdot \frac{q_a \cdot q_b}{\sqrt[6]{R_{ab}^6 + \alpha_{cshift} \cdot (1 - \lambda_{el})}} \tag{3}$$

The modified Lenard-Jones is calculated like this:

$$U_{vdw}(ab) = \lambda_{vdw} \cdot \epsilon_{AB} \left[\frac{R_{ab,0}^{12}}{\left(\alpha_{vshift} \cdot (1 - \lambda_{vdw})^2 \cdot R_{ab,0}^6 + R_{ab}^6\right)^2} - \frac{2 \cdot R_{ab,0}^6}{\alpha_{vshift} \cdot (1 - \lambda_{vdw})^2 \cdot R_{ab,0}^6 + R_{ab}^6} \right] \tag{4}$$

FEP calculations can be modified with various parameters.

variable	effect	default
FEPlambda	Final value for order parameter. Doesn't need to be changed at all	float [1.0]
FEPdlambda	Lambda increment. $FEPdlambda^{-1}$ = number of FEP windows.	float [0.05]
FEPvdwcouple	Controls coupling of VdW interactions (somehow λ_{vdw} is calculated from it)	float [1.0]
FEPeleccouple	Controls coupling of electrostatics (somehow λ_{el} is calculated from it)	float [1.0]
FEPvshift	Value for the VdW shifting parameter in the softcore potential (= α_{vshift} in equation 4)	float [1.0]
FEPcshift	Value for shifting parameter in the electrostatic potential (= α_{cshift} in equation 3)	float [1.0]
FEPequil	Number of equilibration steps in each window	integer [10]
FEPsteps	Number of production steps in each window	integer [10]
FEPfreq	Frequency of FEP output	integer [1000]

FEP calculations produce two output files: "alchemical.txt" and "FEP_Results.txt". "alchemical.txt" contains detailed information about electrostatic and VdW interactions for the current lambda value. Furthermore, the temperature and free energy change is displayed. The file "FEP_Results.txt" contains the total free energy change for the simulation and for each window. The FEP implementation is part of the MD code. Thermostats, barostats boundary conditions and all other parameters needed can be controlled with the corresponding MD variables.

Attention! Do not use temperature gradients in this task because the temperature will continue to rise in every MD simulation. So if you say you want to raise the temperature from 0 to 300 K it will do this in the first MD but it will go up to 600 K in the next, then to 900 K and so on.

6.14 ALIGN - Trajectory Alignment

The trajectory alignment task allows the alignment of ensembles of structures, for example structures obtained from molecular dynamics. For translational alignment, the center-of-mass of all structures are aligned to the origin of the coordinate system. Rotational alignment is performed via Kabsch's method[48,49]. This task can furthermore calculate distance measures between the structures

of the ensemble in regard to a reference structure. Regarding the distance metric, one can chose between the (standard) root-mean-square deviation (RMSD), the distance root-mean-square deviation (dRMSD) and the Holm and Sander Score[50]. For a comparison of molecular distance meassures see [51]. If no alignment is performed beforehand the distance measures are calculated among the unaligned snapshots. CAST provides the following optionsfor the task ALIGN:

variable	effect	default
traj_align_translational	Switch translational (= center-of-mass) alignment on or off, false = off, true = on	true
traj_align_rotational	Switch rotational alignment[48, 49] on or off, false = off, true = on	true
traj_print_bool	Switch output of distance measurement on or off, "false" = off, "true" = on	true
align_external_file	Get reference structure from different file then to-be-aligned structures. This is the filename. The file has to be in the same folder.	none
ref_frame_num	Number of reference frame for alignment (if align_external_file is used, this number refers to the therby specified ensemble). First structure is "0"	0
dist_unit	Specifies the distance metric; 0 = RMSD, 1 = dRMSD, 2 = Holm and Sander Score	0
holm_sand_r0	value for Holm and Sander Score's contact cutoff distance in Å	20

6.15 ENTROPY - Conformational and Configurational entropy calculations

For further analysis of Molecular Dynamics ensembles CAST provides the calculation of entropy contributions. Different approaches for entropy calculation can be used. Currently CAST offers calculations according to Karplus[52]and Schlitter[53]. Calculations can be performed in cartesian or internal coordinates for all or only several snapshots. Furthermore, the degrees of freedom (DOFs) can also be truncated to a selection of internals or cartesian coordinates. If internal coordinates are to be used, you need to specify the indentifying integer of the atom where the desired internal coordinate belongs to. Since the building of internal Z-Matrices can differ between different software packages, we suggest you run the task "INTERNAL" to obtain the Z-Matrix CAST will use for your specific molecular system. The following options are provided by CAST:

variable	effect	default
entropy_alignment	Switch alignment of structures according to Kabsch's method[48, 49] on or off, false = off, true = on; see section 6.14	true
entropy_ref_frame_num	Reference frame for alignment	0
entropy_start_frame_num	First frame to be used for entropy calculations	0
entropy_offset	If a value $n \neq 1$ is specified, only every *n'th* frame will be used for entropy calculations	1
entropy_temp	Temperature of the original simulation in K	300.00
entropy_use_internal	Specifies wether internal coordinates should be used, "true" = yes, "false" = "no"	false
pca_use_internal	Specifies wether internal coordinates should be used, "true" = yes, "false" = no	false
entropy_internal_ang	Specify number of atoms (starting with "0") for which the bond angle in radians will be included in the entropy calculations (example: "3-7,8,13,15,79-115")	none
entropy_internal_bnd	Specify number of atoms (starting with "0") for which the dihedral angle will be included in the entropy calculations (example: "3-7,8,13,15,79-115")	none
entropy_trunc_atoms_bool	If cartesian coordinates are used, this specifies wether only those belonging to certain atoms will be used in entropy calculations. Options are "true" = truncation activated and "false" = all atoms will be included.	false
entropy_trunc_atoms_num	If entropy_trunc_atoms_bool = true, specify the integers of those atoms whose coordiantes are to be included (example: "1,3,7,11-45,2,77-99")	none
entropy_method	Specifies the method according to which the entropy will be calculated • 1 = Quasi-Harmonic-Approx., configurational entropy, according to Karplus[52] • 6 = Quasi-Harmonic-Approx., conformational entropy, according to Schlitter[53] (use with cartesian coordiantes only) • 0 = All of the above, executed sequentially	

6.16 PCA - Principal Component Analysis

The Principal Component Analysis tasks performs a principal component analysis on Simulation trajectories. The procedure is split into two tasks, *PCAgen* and *PCAproc*. First, *PCAgen* is run to obtain the coordinates of the input trajectory in their transformed form as Principal Component Analysis (PCA) modes. These are contained in a file titled "pca_modes.dat" which also contains the eigenvalues and eigenvectors of the covariance matrix. For more information on the theoretical background of the use of PCA see [54–58]. The mathematical steps in performing PCA are closely related to those during conformational and configurational entropy calculations, therefore the configuration options of this task are very similar to those of the task ENTROPY:

variable	effect	default
pca_alignment	Switch alignment of structures according to Kabsch's method[48, 49] on or off, false = off, true = on; see section 6.14	true
pca_ref_frame_num	Reference frame for alignment	0
pca_start_frame_num	First frame to be used (couting starts at "0")	0
pca_offset	If a value $n \neq 1$ is specified, only every *n'th* frame will be used	1
pca_read_vectors	Switch "true" or "false". If "true", eigenvectors of the covariance matrix will not be calculated but read from a previously performed PCA file named „pca_modes.dat". This enables the projection of structures onto the PCA modes of other trajectories.	"false"
pca_read_modes	Switch "true" or "false". If "true", PCA Modes of the covariance matrix will not be calculated but read from a previously performed PCA file named „pca_modes.dat". This is for example useful when different histogramming procedures are to be applied.	"false"
pca_use_internal	Should dihedrals instead of cartesian coordinates be used?	"false"
pca_internal_dih	Specify number of atoms (starting with "1") for which the dihedral angle in radians will be included in the dPCA according to task INTERNAL (see [56–58]) (example: "3-7,8,13,15,79-115")	none
pca_ignore_hydrogen	Ignore either all cartesian coordinates of hydrogens or all dihedrals including hydrogen	"false"
pca_trunc_atoms_bool	If cartesian coordinates are used, this specifies weather only those belonging to certain atoms will be used in thePCA. Options are "true" = truncation activated and "false" = all atoms will be included.	false
pca_trunc_atoms_num	If pca_trunc_atoms_bool = true, specify the integers of those atoms whose coordinates are to be included (example: "1,3,7,11-45,2,77-99")	none

variable	effect	default
pca_print_probability_density	Boolean flag ("true" / "false"): Print probability density of generated PCA modes (obtained through internal histogramming).	true
pca_histogram_width	Bin size of one histogram. Number of bins is adjusted accordingly. One can use "pca_histogram_number_of_bins alternatively.	none
pca_histogram_number_of_bins	Number of histogram bins per dimension. Bin size is adjusted accordingly. "pca_histogram_width" can be used alternatively.	none
pca_dimensions_for_histogramming	Specify an integer range identifying the dimensions that will be histogrammed. Example: "1,2": Two dimensional histogramming of the first two PCA Modes. Example: "1-3": Three-Dimensional histogramming of the first three PCA Modes. Example: "1,5,8,11-15" etc.	none

CAST will, after performing PCA, perform histogramming of PCA modes as desired by the user to get an approximation to the free energy landscape of the molecular system. Usually, one would want to histogram the first two PCA modes to obtain a two-dimensional approximation to the free energy landscape. An example for the configuration flags for the most common application of (cartesian) PCA through CAST is:

```
1    pca\_alignment                          true
2    pca\_ref\_frame\_num                      0
3    pca\_start\_frame\_num                    0
4    pca\_read\_vectors                      false
5    pca\_read\_modes                        false
6    pca\_print\_probability\_density          true
7    pca\_histogram\_width                   0
8    pca\_histogram\_number\_of\_bins          100
9    pca\_dimensions\_for\_histogramming     1,2
```

The output of the histogramming is placed in a file called "pca_histograms". For the above example of two-dimensional histogramming, the output can be easily plotted using gnuplot[59]. Example options for gnuplot are given below:

```
1  set terminal png
2  set output ``pca_hist.png''
3  set pm3d map
```

```
4  splot ''pca\_histograms''
```

From this plot, one can identify basins of the potential energy surface. To obtain the structures corresponding to those basins, the second task *PCAproc* may be used. It reads the file *pca_modes.dat* and the original input ensemble to extract structures corresponding to desired regions of the histogrammed PCA Modes. Input options for this task are the following:

variable	effect	default
proc_desired_start	Float values, each one corresponding to the start of a desired region in one dimension (Example:" -5.0, -1.0")	none
proc_desired_stop	Float values, each one corresponding to the stop of a desired region in one dimension (Example:" 5.0, 6.0")	none

6.17 ADJUST - something something

write something here

6.18 GRID - jajajajjajaja blub

write something here

6.19 PATHOPT - I dont even know

write something here

6.20 PATHSAMPLING - Repeated Gradient Calculation?

write something here

6.21 REACTIONCOORDINATE - REACTIONCOORDINATE?

write something here

6.22 REMOVE_EXPLICIT_WATER

This task will remove all explicit water from the input trajectory and write out the truncated coordinate. This may be useful to prepare data for tasks such as ENTROPY, PCAgen or ALIGN. In detail, the truncation is performed by detecting oxygen atoms which are only bound to two hydrogen atoms.

7 Boundary Conditions

CAST features two types of boundary conditions: spherical and periodic. Spherical boundary conditions can be applied with an energy interface if desired, periodic boundaries are limited to the force field interfaces.

7.1 Spherical boundaries

Spherical boundaries apply a harmonic potential on particles which drift farer away from the geometric center of the simulation than a certain threshold. There are 7 input parameters in total which are defined in a single line in the input file.

keyword	param1	param3	param4	param5	param6	param7	
MDspherical	active	Radius1	Radius1	Force1	Force2	Exp1	Exp2
MDspherical	1	30.0	0.0	10.0	0.0	2	0

Effects of the input variables in detail:

variable	effect	default
Active	Switches spherical boundaries on or off; 0 = off, 1 = on	0
Radius1	Distance for the first potential in Å	none
Radius2	Distance for the second potential in Å	none
Force1	Force constant for inner radius	none
Force2	Force constant for outer radius	none
Exp1	Exponent for inner potential	None, only 2 and 4 are reasonable
Exp2	Exponent for outer potential	None, only 2 and 4 are reasonable

There are two potentials that can be applied: a standard harmonic potential if only the variables with index 1 are used and if the variables with index 2 are also used, a second harmonic potential with a different radius can be applied. Force constants are given in $\frac{kcal}{mol}$. Negative force constants push atoms away from the center, positive force constants push it toward the center.

7.2 Periodic boundary conditions

Periodic boundary conditions can be used to simulate periodic systems. The basic idea is to choose the smallest possible unit cell in the systems as the simulated system and virtually copy it indefinitely in all directions to generate an infinite system. In the actual simulation only the conditions of the initial unit cell are calculated. The infinity is generated by the fact, that particles that leave the simulation box reenter from the opposite side. Periodic Boundary Conditions (PBC) are usually used for the simulation of solvated macromolecules or other mixtures, as well as bulk gases, liquids and crystals. Crucial parameters for the correct behavior of the boundary conditions are the size and shape of the unit cell as well as the chosen cutoff radius. Periodic boundaries are enabled in a single line in the input file with the keyword "Periodics".

keyword	param1	param2	param3	param4
periodics	active	x-dim	y-dim	z-dim
periodics	1	10.0	10.0	10.0

Effects of the input variables in detail:

variable	effect	default
active	Switch periodics on or off, 0 = off, 1 = on	0
x-dim	Box-dimension in x-direction in Å	0
y-dim	Box-dimension in y-direction in Å	0
z-dim	Box-dimension in z-direction in Å	0

8 Coding within the CAST Framework

Lorem Ipsum here be text it the future

Once CAST4 is up and running we should put some information on the code here.

9 What to do if something goes wrong

If you encounter unexpected behavior when using CAST, you may contact the developers for support. In order to properly categorize the issue you may be encountering, consider running CAST again with *verbosity* set to a very high number (for example *500*, or, if this produces too much data, *20*) and sending us the text-output as well as the generated files.

Even though the CAST source code is currently not available to the public, we provide a dedicated *github* repository for providing feedback to users and resolve issues. It may be found at `https://github.com/djmuw/cast_feedback`. If you do not want to use the *github* repository, you can also write us an E-Mail.

10 How to cite CAST

If you have used CAST to perform calculations, you should always cite the following main paper:

C. Grebner, J. Becker, D. Weber, D. Bellinger, M. Tafipolski, C. Brückner, B. Engels, *Journal of Computational Chemistry* **2014**, *35*, 1801-1807.

11 Contact and Support

cast@chemie.uni-wuerzburg.de we should really acquire this mail adress, better than not having it.

Working Group Prof. Dr. Bernd Engels
Institute for Physical and Theoretical Chemistry
Julius-Maximilians University Wuerzburg
Emil-Fischer-Strasse 42
97074 Wuerzburg
GERMANY

Technical Support via:
`https://github.com/djmuw/cast_feedback`
or
cast@chemie.uni-wuerzburg.de

12 Bibliography

References

1. C. Grebner, J. Becker, S. Stepanenko, B. Engels, *Journal of Computational Chemistry* **2011**, *32*, 2245–2253.
2. N. Metropolis, S. Ulam, *Journal of the American Statistical Association* **1949**, *44*, 335–341.
3. Z Li, H. A. Scheraga, *Proceedings of the National Academy of Sciences* **1987**, *84*, 6611–6615.
4. W. F. van Gunsteren, H. J. C. Berendsen, *Angewandte Chemie International Edition in English* **1990**, *29*, 992–1023.
5. C. Grebner, L. P. Pason, B. Engels, *Journal of Computational Chemistry* **2013**, *34*, 1810–1818.
6. H. JONSSON, *Classical and Quantum Dynamics in Condensed Phase Simulations* **1998**, *385*.
7. S. Kumar, J. M. Rosenberg, D. Bouzida, R. H. Swendsen, P. A. Kollman, *Journal of Computational Chemistry* **1992**, *13*, 1011–1021.
8. B. Aradi, B. Hourahine, T. Frauenheim, *The Journal of Physical Chemistry A* **2007**, *111*, PMID: 17567110, 5678–5684.
9. J. Stewart, English, *Journal of Molecular Modeling* **2013**, *19*, 1–32.
10. J. D. C. Maia, G. A. U. Carvalho, J. Carlos Peixoto Mangueira, S. R. Santana, L. A. F. Cabral, G. B. Rocha, *Journal of Chemical Theory and Computation* **2012**, *8*, PMID: 26605718, 3072–3081.
11. I. S. Ufimtsev, T. J. Martinez, *Journal of Chemical Theory and Computation* **2009**, *5*, PMID: 26631777, 2619–2628.
12. C. Grebner, J. Becker, D. Weber, D. Bellinger, M. Tafipolski, C. Brückner, B. Engels, *Journal of Computational Chemistry* **2014**, *35*, 1801–1807.
13. OpenMP Architecture Review Board, OpenMP Application Program Interface Version 3.0, **May 2008**.
14. D. S. M. William L. Jorgensen, J. Tirado-Rives, *Journal of the American Chemical Society* **1996**, *118*, 11225–11236.
15. G. A. Kaminski, R. A. Friesner, J. Tirado-Rives, W. L. Jorgensen, *The Journal of Physical Chemistry B* **2001**, *105*, 6474–6487.
16. K. Vanommeslaeghe, E. Hatcher, C. Acharya, S. Kundu, S. Zhong, J. Shim, E. Darian, O. Guvench, P. Lopes, I. Vorobyov, A. D. Mackerell, *Journal of Computational Chemistry* **2010**, *31*, 671–690.
17. W. D. Cornell, P. Cieplak, C. I. Bayly, I. R. Gould, K. M. Merz, D. M. Ferguson, D. C. Spellmeyer, T. Fox, J. W. Caldwell, P. A. Kollman, *Journal of the American Chemical Society* **1995**, *117*, 5179–5197.
18. J. W. Ponder, C. Wu, P. Ren, V. S. Pande, J. D. Chodera, M. J. Schnieders, I. Haque, D. L. Mobley, D. S. Lambrecht, J. Robert A. DiStasio, M. Head-Gordon, G. N. I. Clark, M. E. Johnson, T. Head-Gordon, *The Journal of Physical Chemistry B* **2010**, *114*, PMID: 20136072, 2549–2564.
19. Y. Shi, Z. Xia, J. Zhang, R. Best, C. Wu, J. W. Ponder, P. Ren, *Journal of Chemical Theory and Computation* **2013**, *9*, PMID: 24163642, 4046–4063.
20. J. W. Ponder, TINKER - Software Tools for Molecular Design, via `http://www.dasher.wustl.edu/ffe` (last accessed July 25, 2015).

21. U. Essmann, L. Perera, M. L. Berkowitz, T. Darden, H. Lee, L. G. Pedersen, *The Journal of Chemical Physics* **1995**, *103*, 8577–8593.
22. M. P. Forum, MPI: A Message-Passing Interface Standard, tech. rep., Knoxville, TN, USA, **1994**.
23. G. Schaftenaar, J. Noordik, *Journal of Computer-Aided Molecular Design*, *14*, 123–134.
24. W. Humphrey, A. Dalke, K. Schulten, *Journal of Molecular Graphics* **1996**, *14*, 33–38.
25. M. Hanwell, D. Curtis, D. Lonie, T. Vandermeersch, E. Zurek, G. Hutchison, *Journal of Cheminformatics* **2012**, *4*, 17.
26. PerkinElmer, ChemOffice Professional, via `https://www.cambridgesoft.com/Ensemble_for_Chemistry/ChemOffice/ChemOfficeProfessional/` (last accessed Jan 13, 2016).
27. N. O'Boyle, M. Banck, C. James, C. Morley, T. Vandermeersch, G. Hutchison, *Journal of Cheminformatics* **2011**, *3*, 33.
28. J. D. Head, M. C. Zerner, *Chemical Physics Letters* **1985**, *122*, 264 –270.
29. J. J. More, D. J. Thuente, *ACM Trans. Math. Softw.* **Sept. 1994**, *20*, 286–307.
30. D. Rapaport, *The Art of Molecular Dynamics Simulation*, Cambridge University Press, **2004**.
31. S. Nosé, *The Journal of Chemical Physics* **1984**, *81*, 511–519.
32. W. G. Hoover, *Phys. Rev. A* **1985**, *31*, 1695–1697.
33. H. J. C. Berendsen, J. P. M. Postma, W. F. van Gunsteren, A. DiNola, J. R. Haak, *The Journal of Chemical Physics* **1984**, *81*, 3684–3690.
34. D. Beeman, *Journal of Computational Physics* **1976**, *20*, 130 –139.
35. J. Becker, *Development and Implementation of new Simulation Possibilities in the CAST program package*, Würzburg, **2015**.
36. H. C. Andersen, *Journal of Computational Physics* **1983**, *52*, 24 –34.
37. N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, E. Teller, *The Journal of Chemical Physics* **1953**, *21*, 1087–1092.
38. F. Glover, *Computers and Operations Research* **1986**, *13*, Applications of Integer Programming, 533 –549.
39. F. Glover, *ORSA Journal on Computing* **1989**, *1*, 190–206.
40. F. Glover, *ORSA Journal on Computing* **1990**, *2*, 4–32.
41. A. Heyden, A. T. Bell, F. J. Keil, *The Journal of Chemical Physics* **2005**, *123*.
42. G. M. Torrie, J. P. Valleau, *Chemical Physics Letters* **1974**, *28*, 578 –581.
43. G. Torrie, J. Valleau, *Journal of Computational Physics* **1977**, *23*, 187 –199.
44. J. Kästner, *Wiley Interdisciplinary Reviews: Computational Molecular Science* **2011**, *1*, 932–942.
45. S. Kumar, J. M. Rosenberg, D. Bouzida, R. H. Swendsen, P. A. Kollman, *Journal of Computational Chemistry* **1992**, *13*, 1011–1021.
46. A. Grossfield, WHAM: the weighted histogram analysis method, via `http://membrane.urmc.rochester.edu/content/wham` (last accessed Jan 13, 2016).
47. C. Chipot, A. Pohorille, *Free Energy Calculations*, Springer Berlin Heidelberg, Berlin, Heidelberg, **2007**.
48. W. Kabsch, *Acta Crystallographica Section A* **1976**, *32*, 922–923.
49. W. Kabsch, *Acta Crystallographica Section A* **1978**, *34*, 827–828.

50. L. Holm, C. Sander, *Journal of Molecular Biology* **1993**, *233*, 123–138.
51. S. Wallin, J. Farwer, U. Bastolla, *Proteins: Structure Function and Bioinformatics* **2003**, *50*, 144–157.
52. I. Andricioaei, M. Karplus, *The Journal of Chemical Physics* **2001**, *115*, 6289–6292.
53. J. Schlitter, *Chemical Physics Letters* **1993**, *215*, 617–621.
54. R. Bro, A. K. Smilde, *Anal. Methods* **2014**, *6*, 2812–2831.
55. H. Abdi, L. J. Williams, *Wiley Interdisciplinary Reviews: Computational Statistics* **2010**, *2*, 433–459.
56. A. Altis, P. H. Nguyen, R. Hegger, G. Stock, *The Journal of Chemical Physics* **2007**, *126*, 244111.
57. Y. Mu, P. H. Nguyen, G. Stock, *Proteins: Structure Function and Bioinformatics* **2005**, *58*, 45–52.
58. F. Sittel, A. Jain, G. Stock, *The Journal of Chemical Physics* **2014**, *141*, 014111.
59. T. Williams, C. Kelley, many others, Gnuplot 4.4: an interactive plotting program, `http://gnuplot.sourceforge.net/`, **2010**.