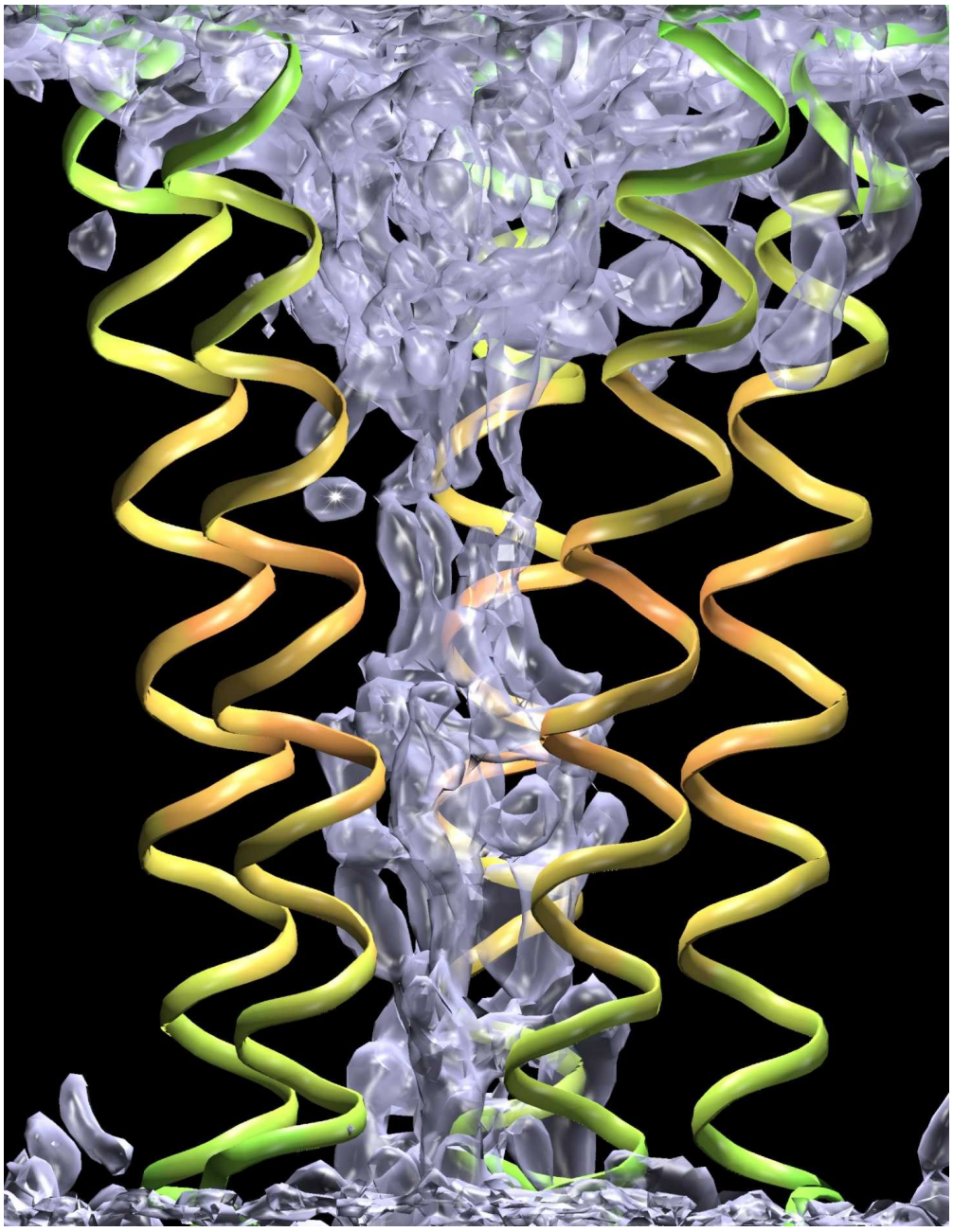
Principles of Gating Mechanisms of Ion Channels



Oliver Beckstein

Principles of Gating Mechanisms of Ion Channels

Oliver Beckstein

Laboratory of Molecular Biophysics and
Merton College, Oxford

Michaelmas 2004

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy at the University of Oxford

D Programs and scripts

This appendix describes some programs that were developed or modified in the course of this work. Where licensing permits, they are made available under the GNU General Public License[319] on the author's website `http://sansom.biop.ox.ac.uk/oliver/software/`.

D.1 Simulation setup

D.1.1 pgeom

Usage

`pgeom [OPTIONS] -R` R_{outer} `-P` R_{P} L_{P} `-M` R_{M} L_{M}

Synopsis

pgeom calculates positions of pseudo-atoms in a model for a transmembrane channel and writes the structure to the file `pore.pdb` in PDB format, and a gromacs topology to `pore.itp`. The pore is constructed from concentric, stacked rings of spherical pseudo atoms. Typically, no bonds are generated because the pore atoms are held at their equilibrium position by harmonic restraints (which are generated with the genpr program, which is part of gromacs).

The user determines the dimensions of the mouth region and the constriction region. The pore profile is linearly interpolated between the radius of the constriction R_{P} and the radius of the mouth R_{M}.

Options

`-h`		show help
`-v`		be verbose (= `-debuglevel 30`)
`-debug`	N	set debuglevel (0…100)
`-spec`	m_{spec}	atomic species for pore wall (see Tab. D.3)
`-showspec`		show hard coded atomic species (see Tab. D.3)
`-o`	`pore.pdb`	`pdb` coordinate file
`-s`	`pore.itp`	`itp` topology file

`-R`	R_{outer}	Outer radius of the model R_{outer}
`-P`	$R\ L$	Pore region: inner radius R and length L
`-M`	$R_{\text{M}}\ L_{\text{M}}$	Mouth region: largest inner radius R_{M} and length L_{M}
`-b`	$d_{\min}\ d_{\max}\ \gamma_{\min}\ \gamma_{\max}$	Option is repeatable. Form bonds with angle γ when interatomic distance d satisfies $d_{\min} \leq d \leq d_{\max}$ (in Å) and angle $\gamma_{\min} \leq \gamma \leq \gamma_{\max}$, $0^\circ \leq \gamma \leq 90^\circ$
`-c`		only write connectivity to output, no bond length or k_{bond}, k_{angle}
`-x`		neither connectivity nor bond length to output (isolated atoms)
`-kB`	k_{bond}	force constant of bonds, in $\text{kJ} \cdot \text{mol}^{-1} \cdot \text{nm}^{-2}$
`-kA`	k_{angle}	force constant of angles, in $\text{kJ} \cdot \text{mol}^{-1} \cdot \text{rad}^{-2}$
`-cc`		center coordinates on cavitybox, not on unitcell

As an experimental feature, the pore volume can be calculated as described in Appendix B; the Lennard-Jones parameters are taken from the GROMACS force field. Setting any of the switches for profile calculation also enables the volume calculations. All the following *lengths* are in *nano metre* (nm) not Ångstrom.

`-volume`		calculate pore volume
`-profile`	`profile.dat`	calculate the profile $R(z)$ in addition to the volume and write to file
`-z1`	z_1	profile between z_1 and z_2
`-z2`	z_2	
`-Rmax`	$R_{\max}$	integrate out to $R_{\max}$ (also use for the total volume integration if `-profile` is set)
`-npoints`	N	number of points per dimension in the integrals
`-T`	T	temperature in Kelvin [`300`]
`-wca`		if set, only use the repulsive part of the Lennard-Jones potential (split after Weeks, Chandler, and Andersen[123])
`-plot`		xfarbe output of the potential in z slices
`-nzplot`	N_{slice}	number of plot slices for `-plot`

Example

The hydrophobic pores of radius R, that were investigated in Chapter 3 were generated with the command line

Table D.3: Hard coded atomic species in pgeom.

m_{spec}	name	symbol	r/nm	q/e
0	methane	CH_4	0.195	0
1	carbon	C	0.140	+0.38
2	oxygen	O	0.150	−0.38

```
pgeom -R 18 -P R 8 -M 10 8 -x -o pore.pdb -s
pore.itp
```

where $R_{\text{outer}} = 1.8$ nm (which is not functionally important because the pore complex is embedded in a membrane mimetic slab), $R_{\text{M}} = 1.0$ nm, $L_{\text{M}} = L_{\text{P}} = 0.8$ nm (apart from the cases in Section 5.3.3 where L_{P} was varied), and the pore radius R. Pores with hydrophilic pore walls were created by adding charges to selected atoms in the topology file `pore.itp`.

D.1.2 prepumbrella.pl

Usage

```
prepumbrella.pl [OPTIONS] --topology=topol.top
--mdp=umbrella.mdp --index=refgroup.ndx
--confstart=startingconfigs.dat --type="SPECIES"
```

Synopsis

Create run input files (`tpr`) for umbrella sampling runs, using gromacs (version 3.2.1-LMB). Give the number of umbrella windows and either the length over which the windows are (equally) distributed or the minimum and maximum coordinate. At the moment this only caters for 1D PMFs, e.g. just along z in a pore or channel.

If only `--length` is given, the windows are created symmetrically to 0 (which is then the centre of mass of the reference index group (type '`com`' or '`com_t0`' in the `ppa`)). If `--min` and `--max` or `--length` and either one of `--min` or `--max` are given, the window centres are offset accordingly (still relative to the reference group's centre of mass). The window width Δz is calculated as length $L = \max z - \min z$ divided by the number of windows N, $\Delta z = L/N$.

Unless overridden from the commandline, the force constant is chosen so that an ion in window i can freely diffuse into an adjacent window $i+1$ or $i-1$. The energetic barrier to this diffusion is given by `--barrier` (and is the energy of the umbrella potential at $z_i + \frac{3}{2}\Delta z$, i.e. the centre of the neighbouring window). Set the temperature to the one set for the simulations.

This script requires all the files needed to run grompp so that it can create the `tpr` files for all the windows and a special list of starting configurations, customarily named `conf.dat`. It creates a `ppa` file using a hard coded template. The grompp `mdp` file should have the proper run length (e.g. 1 ns) and typically neither `trr` nor `xtc` output are required.

Implicit assumptions

- Currently we assume that the direction of the sampled coordinate is $(0, 0, 1)$, i.e. the z-axis.
- the configuration list contains absolut positions
- the `center_of_mass` (com) must be supplied in the configuration list
- max, min, and length are relative to the com of the ref group

Options

Defaults in brackets, for most options abbreviations can also be used.

`--help`	this help
`--help-targets`	list of known target systems (for `--targetsystem`)
`--help-config`	format of the starting configuration file (for `--confstart`); see below
`--version`	version number of prepumbrella.pl (quote when submitting a bug report)
`--verbose`	chatty output
`--debug=INTEGER`	set debug level ($>$ 9 switches off error checks, $>$ 11 grompp is not run at all, only commandline shown)

Input files (all must exist):

`--topology=FILE`	gromacs topology [`topol.top`]
`--mdp=FILE`	grompp input with run parameters [`grompp.mdp`]
`--index=FILE`	index file containing the reference group [`pull.ndx`]
`--confstart=FILE`	list that describes available starting conformations (see below for format) [`conf.dat`]
`--np=INTEGER`	number of processors to write tpr files for [`1`]

Output:

`--Project=DIRECTORY`	place all window directories under `DIRECTORY` [`.`]
`--prefix=STRING`	output `tpr`s are named `STRINGn.nnnn.tpr` (where n.nnnn is the centre of the umbrella window in nm) if the standard `--format` string is used []
`--format=PRINTFSTRING`	pattern to create output filenames. May contain one '`%s`' (for prefix) and one '`%f`' for the umbrella centre (see printf(3)) [`%s%+.4f`]
`--qsuball=FILE`	script that contains `qsub` commands to submit all windows through a queuing system
`--targetsystem=SYSTEM`	create run script(s) for the given target systems; can be given repeatedly to produce more than one type of runscript. Use `--help-targets` for a list of valid `SYSTEM`s and the values that they define (with `--verbose`) [`local`]

Umbrella windows:

`--type=STRING`	name of the particle, eg `Na`, `Cl`, `NA`, ... (as used in the input files)
`--nwindows=INTEGER`	number of umbrella windows [`20`]
`--length=FLOAT`	length to cover with windows, symmetrically to the reference group's centre of mass. Alternatively, set `--min` and `--max`.
`--min=FLOAT`	relative to reference group centre of mass, in nm
`--max=FLOAT`	relative to reference group centre of mass, in nm
`--kUmbrella=FLOAT`	force constant of the harmonic umbrella potential in $\mathrm{kJ \cdot mol^{-1} \cdot nm^{-2}}$; overrides program choice
`--Temperature=FLOAT`	temperature of the simulation in Kelvin [`298.0`]
`--barrier=FLOAT`	barrier to move to the centre of an adjacent window in kT [`1.5`]

Format of the starting configuration file

The configuration file `conf.dat` contains a header section where parameters are set, and a list of possible starting configurations. prepumbrella.pl uses the list of starting configurations to find those which have a particle of interest close

to the centre of an umbrella window. It then looks up the corresponding starting structure file (in `gro` format) from the list and uses it to create binary run input file (`tpr`) by running grompp.

```
; comment (same as in Gromacs mdp files)
; all entries must be on one line
; '@' lines set variables
@ var = value
filename   particle_type  atom_nr   pos_x pos_y pos_z
filename   particle_type  atom_nr   pos_x pos_y pos_z
...
```

`filename`	full path to the `gro` or `pdb` file
`particle_type`	name used in gromacs, eg `Na`, `Cl`, `SOL`
`atom_nr`	atom_id, number in `gro` or index file
`pos_x|y|z`	coordinates of the particle in nm

Required variables Variable assignments are introduced by an @-sign at the beginning of the line; spaces around '=' are mandatory. Required variables must be given in the configuration file.

`center_of_mass =` $x\ y\ z$	center of mass of the reference group

Optional variables If no values are given the program defaults are chosen (given in brackets) and the program issues a warning. A cylindrical confinement potential is only activated if both `RConfine` and `KConfine` are non-zero.

`reference_group = group`	reference group from index file [`protein`]
`reftype = com_t0|com`	center of mass from initial frame or always from the current frame [`com_t0`]
`RConfine =` R	use a confinement potential outside a radius R nm [`0.0`]
`KConfine =` k	and a harmonic force constant k kJ $\cdot$ mol^{-1} $\cdot$ nm^{-2} [`4500.0`]

prepconflist.pl (Appendix D.1.3) can be used for setting up the configuration list from `trr` frames.

Bugs and Limitations

- Runscripts are generated for different target system environments and the specifics are hard coded in the script itself. See the source itself and `--help-targets --verbose`. The target system complex should become configurable through an external XML file.
- Only single ions as pulled groups are tested.
- `conf.dat` should be XML.
- The umbrella `ppa` file should be a template and not hard coded.
- Dirty hack: if there is an index file named exactly '`index.ndx`' in the current directory then it is automatically picked up at the grompp stage; this allows for using non-standard groups in the `mdp` file.

Example

`conf.dat` contains all required variables and a list of starting configurations, with the starting configurations themselves accessible. The index file `refgroup.ndx` contains the index group `protein`, relative to whose centre of mass the umbrella windows are placed. The topology and related files are found two directories above the current one, with the main topology being `chl.top`. The molecular dynamics run input file `umbrella.mdp` is similar to one used for equilibrium simulations with the exception of a shorter run length (typically about 1 ns, no trajectory output, and when starting from structure files with velocities, `gen_vel = no` and `unconstrained-start = yes`). In the example, a Na^+ ion (with the general name `Na`) is to be umbrella sampled. The command line

```
source /opt/gromacs/3.2.1-LMB/i686-pc-linux-gnu/bin/GMXRC
prepumbrella.pl --topology=../../chl.top
--mdp=umbrella.mdp  --index=refgroup.ndx
--confstart=conf.dat  --Project=0R55/Na
--prefix=Na0R55z --type=Na --nwindows=101
--min=-0.9 --max=0.9  --Temperature=300
--targetsystem=workstations --qsuball=submitall.sh
```

will have prepumbrella.pl construct 101 separate input files for mdrun; each of these windows can be run independently from the others thus allowing massively parallel execution on clusters or compute farms. Depending on the target system, run input files for a queuing system are also created, which allow submission of all 101 jobs through execution of a single script, `submitall.sh`.

Note that the script requires a working gromacs environment to create the run input files.

After these jobs have finished, g_wham is run on all result files `*.pdo.gz` as described in Appendix D.2.5.

D.1.3 prepconflist.pl

Usage

```
prepconflist.pl [OPTIONS] file1.gro [file2.gro
...]
```

Synopsis

prepconflist.pl creates the file list which forms the major content of the input file (`conf.dat`) to prepumbrella.pl. It reads all files given on the command line and writes matches to the output file (with `--append` it appends to the output file).

Options

The options can be abbreviated; defaults are given in [].

`--help`	help
`--verbose`	(alias for `DEBUGLEVEL`$= 3$)
`--version`	print version id
`--debug=DEBUGLEVEL`	0 (almost) quiet, < 0 silent, $1 \ldots 3$ verbose, > 5 really debugging stuff (20 max) [`1`]
`--output=FILE`	output gro file [`conf.list`]
`--append`	append to output file (do not overwrite)
`--atomname=STRING`	name of the pulled atom in the gro file [`NA`]
`--z1=FLOAT`	only list starting positions with z coordinate
`--z2=FLOAT`	between z_1 and z_2 (box coordinates in nm).
`--radius=FLOAT`	Also, only list them when they are within a cylinder of radius R which is centered on the center of mass.
`--xcom=FLOAT`	ref. group centre of mass x-coordinate (nm)
`--ycom=FLOAT`	ref. group centre of mass y-coordinate (nm)
`--zcom=FLOAT`	ref. group centre of mass z-coordinate (nm), but this is not used at the moment as the cylinder is always taken to be parallel to the z-axis

Example

First we generate a **selection of equilibrium configurations** of the whole system by dumping snapshots from an equilibrium trajectory of the system.

```
mkdir start
echo -e '0\n'
| trjconv -tu ns -b 2 -sep -s ../../chl_md.tpr
-f ../../chl_lay_water_md.trr -o start/na.gro
```

Here a `trr` file is used, which allows to start with velocities. About 300 frames are typically required. The whole system is required (which is selected by choosing '0'). The index for the **reference group** is created through

```
make_ndx -f ../../chl_md.tpr -o refgroup.ndx
```

Typically, the whole protein is chosen. The **centre of mass** of the reference group can be calculated with

```
g_traj -com -f start/na_1.gro -s ../../chl_md.tpr
-n refgroup.ndx -ox com.xvg
```

and is found as the single entry in `com.xvg`.

Then prepconflist.pl goes through the configurations and generates a list relating the position of a particle type of interest (e.g. ion or water oxygen) to the file where it was found.

```
prepconflist.pl --xcom=2.2 --ycom=2.3
--radius=0.8  --z1=1.7 --z2=6.4 --atomname=Na
--output=conf.list --verbose  start/*.gro
```

To reduce the number of entries one can restrict the volume in which particles are searched for. The volume is a cylinder centered on the centre of mass of the reference group, extending from $z_1 = 1.7$ nm to $z_2 = 6.4$ nm with a radius $R = 0.8$ nm. The resulting `conf.list` can be appended to the header section of a `conf.dat` file (see Appendix D.1.2).

D.2 Analysis

D.2.1 g_count

Usage

```
g_count -f trajectory.xtc -s topol.tpr [OPTIONS]
```

Synopsis

g_count counts the numbers of molecules within a cylindrical region as a function of time. It takes an index file with atomnumbers and chooses the molecules that these atoms belong to (or optionally (with option `-nom`) it simply takes all atoms in the index) and generates an output file with the number of molecules/atoms which are present in the cavity a each time frame.

The output data file (option `-dat`) contains for each time step t the number of particles in the cavity (the pore occupancy $N(t)$), the concentration $c(t)$, and the mass density $\rho(t)$.

column:	1	2	3	4
quantity:	t	$N(t)$	$c(t)$	$\rho(t)$
units:	ps		$\mathrm{mol}\cdot\mathrm{l}^{-1}$	$\mathrm{g{\cdot}cm}^{-3}$

Density and concentration are calculated from the volume of the cavity,

$$V = R_{\text{eff}}^2 \pi (z_2 - z_1) \tag{D.1}$$

where R_{eff} is the apparent accessible radius (corrected with wall atom radius $r_A = \sigma_{AA}/2$ (approximately the van der Waals radius), water radius $r_{\text{water}} \approx 0.14$ nm, and the distance between the centre of a wall atom A (typically, a united atom methane CH_4) and the water oxygen (O_W), $d_{\text{wall-water}} = \sigma_{AO_W}$. σ come from the Lennard-Jones potential of the wall atom and the water oxygen.

$$R_{\text{eff}} = \hat{R} - d_{\text{wall-water}} + r_{\text{water}}; \tag{D.2}$$

$\hat{R}$ is the radius of the pore, measured from the atomic center of a wall atom to the pore axis. The pore radius R (which is quoted throughout this work) is

$$R = \hat{R} - r_A, \tag{D.3}$$

and it is the radius of the cylinder formed by the solvent accessible van der Waals surface.* Thus, the radius correction could be calculated from the force field (i.e. σ) as

$$\delta R = \frac{1}{2}\sigma_{AA} - \sigma_{AO_W} + r_{\text{water}} \tag{D.4}$$

*Because of the atomic roughness of the pore wall it is not strictly correct to speak of a single pore radius—depending on the angle the distance from the pore axis to the surface varies. R is the shortest radial distance in the pore and hence represents a lower bound on the true average pore radius.. See also the discussion in Appendix C.3.

This approach turns out to be somewhat imprecise and δR was then considered as an adjustable parameter which was determined as described in Section 2.1.4 on page 28 by considering a large pore and matching the volume resulting from the above calculation and an integration of the local density (produced by g_ri3Dc, described in Appendix D.2.3).

The `-track` index file contains the atom_ids of all atoms that were at least for `-tau` τ ps in the cavity (entry `[ *_tracked ]`). The entry `[ *_cavity ]` contains all atoms that were for at least one time frame in the cavity. Entries containing molecule numbers are titled `[ *_molecules ]`.

`-tdat` writes a data file containing all atoms or molecules in the cavity and their total residency time. Format:

column:	1	2	3	4	5
quantity:	atom_nr	molecule_nr	name	total residency time $\mathfrak{T}_{res}$	$\mathfrak{T}_{res}/\mathfrak{T}$
units:				ps	

Defaults If no values are given for the point on the axis (option `-cpoint`) then the center of the initial box is taken. If no radius `-R` R is given, the maximum radius fitting in the box is chosen. z_1 and z_2 default to the full box length in z-direction.

Files

`-f`	`traj.xtc`	Input	Generic trajectory: xtc trr trj gro g96 pdb
`-s`	`topol.tpr`	Input	Structure+mass(db): tpr tpb tpa gro g96 pdb xml
`-o`	`c_numbers.xvg`	Output	xvgr/xmgr file
`-dens`	`c_density.xvg`	Output, Opt.	xvgr/xmgr file
`-conc`	`c_conc.xvg`	Output, Opt.	xvgr/xmgr file
`-dat`	`cavity.dat`	Output	Generic data file
`-tdat`	`c_track.dat`	Output	Generic data file
`-n`	`index.ndx`	Input, Opt.	Index file
`-track`	`c_track.ndx`	Output, Opt.	Index file

Other options

`-h`	bool	`no`	Print help info and quit
`-nice`	int	`0`	Set the nicelevel
`-b`	time	`-1`	First frame (ps) to read from trajectory
`-e`	time	`-1`	Last frame (ps) to read from trajectory
`-dt`	time	`-1`	Only use frame when t MOD dt = first time (ps)
`-w`	bool	`no`	View output xvg, xpm, eps and pdb files

`-m`	bool	`yes`	index contains atoms, but g_count counts the molecules
`-axis`	vector	`0 0 1`	Vector pointing parallel to the pore axis
`-cpoint`	vector	`0 0 0`	Point on the pore axis
`-R`	real	`-1`	Radius of the cylindrical pore cavity
`-z1`	real	`-1`	Confine waters to have a z coordinate larger than z1, and
`-z2`	real	`-1`	smaller than z2
`-tau`	real	`10`	Total time (ps) that a particle has to be in the cavity at least so that it is tracked
`-dR`	real	`0.03`	Correct water-accessible cavity volume

Known problems and caveats

- `-m` behaves differently from the standard usage within the g_* programs—it figures out for itself what the molecules are and does not need *molecule* numbers but *atom_ids*.
- `-m` is the *default* behaviour!
- When counting *ions* you *must* use `-nom`!
- The density is calculated as $\langle M \rangle \, / [R_{\mathrm{eff}}^2 \pi (z_2 - z_1)]$ where $\langle M \rangle$ is the average mass in the pore cavity—so it makes only sense if this approximates the true cavity volume.
- The default volume/radius correction is specific for methane pseudo atoms (ffgmx, ffG43a1) and SPC water.
- The program is only tested with pore axis parallel to z-axis (0,0,1).

Example

Suggested use for water. Create an index file for the water molecules:

```
echo -e "keep 0\ndel 0\nr SOL\nq\n" _
| make_ndx -f in.pdb -o sol.ndx
```

Then analyse the water molecules in a trajectory traj.xtc with

```
g_count -m -f traj.xtc -s topol.tpr -n sol.ndx
```

D.2.2 g_flux

Usage

```
g_flux -f trajectory.xtc -s topol.tpr [OPTIONS]
```

Synopsis

g_flux calculates the flux of molecules (or atoms) through a cylindrical region as a function of time. It takes an index file with atomnumbers and chooses the molecules that these atoms belong to (or optionally (with `-nom`) it simply takes all atoms in the index) and generates an output file with the number of molecules/atoms at each time frame.

Definition of observables

Flux The flux Φ is the number of particles permeating the pore per unit time. It is calculated from the total number of permeating particles M during a simulation of length $\mathfrak{T}$ as

$$\Phi = \frac{M}{\mathfrak{T}}. \tag{D.5}$$

M counts particles permeators in both directions ("up" or "+" and "down" or "−"), i.e. $M = M^+ - M^-$ where $M^+ \geq 0$ and $M^- \leq 0$. The fluxes are

$$\Phi^+ = \frac{M^+}{\mathfrak{T}} \geq 0 \tag{D.6}$$

$$\Phi^- = \frac{M^-}{\mathfrak{T}} \leq 0 \tag{D.7}$$

$$\Phi = |\Phi^+| + |\Phi^-|. \tag{D.8}$$

In equilibrium, the net flux $\Phi^+ + \Phi^-$ is zero because on average there is no net force favouring either direction. In practice, there tends to be a very small bias (of the order of 0.1% of Φ) towards negative fluxes which might be related to the rescaling of the particle positions due to the constant-pressure weak-coupling scheme.

The cumulative flux

$$\mathcal{F}(t) := \int_0^t dt'\, \Phi(t') = \sum_{t'=0}^{t} M(t')$$

is calculated from the instantaneous flux $\Phi(t) = M(t)/\Delta t$ over one time step Δt and counts the number of permeation events up to time t.

Permeation time The residency time $\tau_{\mathrm{p},i}(t)$ for a pore-permeating particle i that entered the pore time at t_i' and exited at t is

$$\tau_{\mathrm{p},i}(t) = t - t_i'. \tag{D.9}$$

The average permeation time $\tau_p(t)$ for all $M(t)$ particles that permeated the pore between t_i' and $t > t_i'$ is calculated as

$$\tau_p(t) = \frac{1}{M(t)} \sum_{i=1}^{M(t)} (t - t_i'),$$

i.e. at each time the length of the permeation events that completed at this step are averaged. The cumulative average permeation time is

$$\langle \tau_p(t) \rangle = \frac{\sum_{t'=0}^{t} M(t')\tau_p(t')}{\sum_{t'=0}^{t} M(t')}$$

and can be used to assess the convergence of the mean permeation time

$$\langle \tau_p \rangle \equiv \langle \tau_p(\mathfrak{T}) \rangle . \tag{D.10}$$

Algorithm

The algorithm to detect the permeating particles can be described as "check on exit." If 'O' denotes the space outside of the pore and 'I' the inside, then only trajectories O→I→O are counted as complete transitions through the pore; I→O→I are periodic boundary trajectories. Any enter event for a particle is recorded; on a later exit it is determined if it left the pore volume on the side opposite to the influx event (a permeation event) or on the same side (not a successful permeation).

In the following, the pore is assumed to be oriented parallel to the z-axis. The faces of the cylindrical pore are at $z_1 < z_2$.* For any given molecule i at position $\mathbf{x} = (x, y, z)$ compare the position at time step t and at the previous step $t - 1$.

1. A molecule **crosses a boundary** at z_m ($m = 1, 2$) if

   $$\min\big(z(t-1), z(t)\big) < z_m < \max\big(z(t-1), z(t)\big)$$

   is true.

2. Given a boundary crossing at step t, determine the type of the crossing (enter or exit event). The type is recorded as the "divergence" D

   $$D = \begin{cases} +1 & \text{if } z_1 < z(t) < z_2 \quad \text{(O→I, } \textit{entering}\text{)} \\ -1 & \text{if } z(t) < z_1 \ \vee \ z(t) > z_2 \quad \text{(I→O, } \textit{exiting}\text{)} \end{cases}$$

*A generalisation to pore axes pointing along other vectors than $(0, 0, 1)$ is straightforward but was not necessary in the cases discussed here.

3. Determine the direction u of the crossing, relative to the pore axis $\mathbf{p}$

$$u = \mathrm{sgn}\big([\mathbf{x}(t) - \mathbf{x}(t-1)] \cdot \mathbf{p}\big),$$

where $u = +1$ signifies a flow "up" and $u = -1$ a flow "down."

4. Characterise the crossing event, and determine permeation events:

Influx event $D = +1$ Only trajectories O→I→O can represent permeation events, i.e. they always start with an influx event, so for the particle i that crossed a boundary at t store the influx event $(i, t, u(t))$.

Efflux event $D = -1$ An efflux event signals a permeation event if the particle exits through the face opposite to the one it entered a previous time t'. Compare the direction of the efflux $u(t)$ with the one of the influx $u(t')$, which was stored as the influx event $(i, t', u(t'))$.

$$u(t) = u(t') \Rightarrow \text{ permeation event.}$$

Then the recorded event $(i, t', u(t'))$ is erased.

Output

The `-dat` file contains instantaneous values at each time step t whereas the `-cdat` data file records the corresponding cumulative values up to t,

column:	1	2	3	4	5	6
`-dat`:	t	$\Phi^+(t) + \Phi^-(t)$	$\Phi^+(t)$	$\Phi^-(t)$	$\Phi(t)$	$\tau_\mathrm{p}(t)$
`-cdat`:	t	$\mathcal{F}^+(t) + \mathcal{F}^-(t)$	$\mathcal{F}^+(t)$	$\mathcal{F}^-(t)$	$\mathcal{F}(t)$	$\langle \tau_\mathrm{p}(t) \rangle$
units:	ps	ps^{-1}	ps^{-1}	ps^{-1}	ps^{-1}	ps

The index file `-ncr` lists the indices of all molecules that permeated the pore successfully. The data file `-res` contains a histogram of permeation times τ_p.

Options

Files

`-f`	`traj.xtc`	Input	Generic trajectory: xtc trr trj gro g96 pdb
`-s`	`topol.tpr`	Input	Structure+mass(db): tpr tpb tpa gro g96 pdb xml
`-n`	`index.ndx`	Input, Opt.	Index file
`-o`	`flux.xvg`	Output	xvgr/xmgr file
`-dat`	`flux.dat`	Output	Generic data file
`-cdat`	`cflux.dat`	Output	Generic data file
`-ncr`	`crossed.ndx`	Output	Index file
`-res`	`residency.dat`	Output	Generic data file

Other options

`-h`	bool	`no`	Print help info and quit
`-nice`	int	`0`	Set the nicelevel
`-b`	time	`-1`	First frame (ps) to read from trajectory
`-e`	time	`-1`	Last frame (ps) to read from trajectory
`-dt`	time	`-1`	Only use frame when t MOD dt = first time (ps)
`-w`	bool	`no`	View output xvg, xpm, eps and pdb files
`-m`	bool	`yes`	index contains atoms, but g_flux counts the molecules
`-axis`	vector	`0 0 1`	Vector pointing parallel to the pore axis
`-cpoint`	vector	`0 0 0`	[hidden] Point on the pore axis
`-z1`	real	`-1`	Confine waters to have a z coordinate larger than z1, and
`-z2`	real	`-1`	smaller than z2

Known problems and limitations

- `-m` behaves different from the standard usage within the `g_*` programs—it figures out for itself what the molecules are and does not need molecule numbers but atom_ids.
- `-m` is the *default* behaviour!
- Despite appearance it only makes sense to specify a pore axis approximately parallel to the z-axis because we only really base the definition of the boundaries on z coordinates.

Example

Suggested use for water: Create an index file for the water molecules:

```
echo -e "keep 0\ndel 0\nr SOL\nq\n"
| make_ndx -f in.pdb -o sol.ndx
```

Then run

```
g_flux -m -f traj.xtc -s topol.tpr -n sol.ndx
-cpoint 2.3.  2.3 2 -z1 1.9 -z2 2.7
```

and analyse the output in `flux.dat`, `cflux.dat`, and `residency.dat`.

D.2.3 g_ri3Dc

Usage

```
g_ri3Dc -f trajectory.xtc -s topol.tpr [OPTIONS]
```

Synopsis

g_ri3Dc places a 3D grid into a simulation box and counts the number of molecules (typically water) in each cell over a trajectory. The rectangular grid is large enough to encompass a cylinder of given radius and height (specify radius R, lower and upper z, and a point on the pore axis (which is always parallel to the z-axis). If no parameters are given for the cylinder, a cylinder is fit into the simulation box. The cylinder (and thus the grid) is fixed. The spatial resolution can be given either as one number for all three dimensions (cartesian coordinates; unit is nm) or separately as three (*Note*: (1) It is recommended to use at least the same grid spacing in x and y (which is compatible with a cylindrical symmetry along z)). (2) The exact dimensions of the grid (radius etc) are re-adjusted to integral numbers of the grid spacing.) At the end, a density map (occupancy time over total simulation time $\mathfrak{T}_{\text{cell}}/\mathfrak{T}$) is written to the grid file.

The output is the number density, averaged over the trajectory. Read this file into a_ri3Dc, the grid analysis program, and produce radial distribution functions, density plots, pore profiles etc. The grid data is written as a binary file (in the machine independent xdr format). a_ri3Dc has an option to write it out as ascii text.

Options

Files

`-f`	`traj.xtc`	Input	Generic trajectory: xtc trr trj gro g96 pdb
`-s`	`topol.tpr`	Input	Structure+mass(db): tpr tpb tpa gro g96 pdb xml
`-n`	`index.ndx`	Input, Opt.	Index file
`-grid`	`gridxdr.dat`	Output	Generic data file

Other options

`-h`	bool	`no`	Print help info and quit
`-nice`	int	`0`	Set the nicelevel
`-b`	time	`-1`	First frame (ps) to read from trajectory
`-e`	time	`-1`	Last frame (ps) to read from trajectory
`-dt`	time	`-1`	Only use frame when t MOD dt = first time (ps)
`-w`	bool	`no`	View output xvg, xpm, eps and pdb files
`-m`	bool	`yes`	index contains atoms, but g_ri3Dc counts the molecules
`-cpoint`	vector	`0 0 0`	Point on the central symmetry axis of the grid
`-R`	real	`0`	Maximum radius that should be contained in the grid
`-z1`	real	`0`	Center grid between z1, and ...
`-z2`	real	`0`	z2 (these boundaries are kept fixed!)

`-delta`	vector		
		`0.02 0.02 0.02`	Spatial resolution in X, Y, and Z (in nm)
`-dtweight`	bool	`yes`	[hidden] Weigh counts with the time step (yes) or count all equally (no)
`-subtitle`	string		Some text to add to the output graphs

Caveats and known limitations

- The program guarantees to use the user supplied grid spacing. If the other dimensions are incommensurable with Delta they are changed to comply.
- If you want radial distribution functions (yes, you do!) always use Delta[XX] == Delta[YY] to keep the cylindrical symmetry
- *z*-axis is the only allowed axis (and this will probably not change in the future)
- `-m` behaves different from the standard usage within the `g_*` programs – it figures out *for itself* what the molecules are and does not need *molecule* numbers but *atom_ids*.
- `-m` is the *default* behaviour. It works nicely with a SOL index file but more complicated solvents are untested.
- For **ions** you haved to use the `-nom` option!
- The XDR file is not compressed.
- Even the developer mistypes the name frequently

Example

Suggested use for water. Create an index file for the water molecules with

```
echo -e "keep 0\ndel 0\nr SOL\nq\n" \
| make_ndx -f in.pdb -o sol.ndx
```

Then run

```
g_ri3Dc -m -f traj.xtc -s topol.tpr -n sol.ndx
-delta 0.05 -grid grid.dat
```

(`-m` is the default but is shown here because the usage is different from the standard gromacs analysis programs). The 3D density is in file `grid.dat` and can be analysed with a_ri3Dc (see Appendix D.2.4).

D.2.4 a_ri3Dc

Usage

```
a_ri3Dc -grid gridxdr.dat [OPTIONS]
```

Synopsis

a_ri3Dc analyses a 3D grid produced by g_ri3Dc. The 2D projections are written in a format suitable for the fast 2D density plotter xfarbe.[223] It also writes a parameter file 'XFarbe' which can be used automatically by setting the environment variable 'XAPPLRESDIR=.', and then running xfarbe, e.g. 'xfarbe rzp.dat ' All geometry options (radius, z1, z2) that are not set and appear as '0' are set from the grid dimensions once it is read in. The -dump option converts a binary grid file into ascii txt. The -plt option produces a binary density file for gOpenMol, which you might have to rename to xxx.plt. vmd since version 1.8.2 can also read plt files and render them. The average density in the test cylinder is printed on std out, using the set units.

Output

density z-averaged	$n(x,y) = {L_z}^{-1} \int dz\, n(x,y,z)$
density y-averaged	$n(x,z) = {L_y}^{-1} \int dy\, n(x,y,z)$
density x-averaged	$n(y,z) = {L_x}^{-1} \int dx\, n(x,y,z)$
density radially averaged	$n(r,z) = (2\pi)^{-1} \int d\phi\, n(r,\phi,z)$
pore profile	$(z, R_G(z), R_G(z)/R)$
radial distribution function (rdf)	$n(r) = (2\pi\ L_z)^{-1} \int d\phi\, dz\, n(r,\phi,z)$
axial distribution function (zdf)	$n(z) = (L_x L_y)^{-1} \int dx\, dy\, n(x,y,z)$

$n(r)$ and $n(z)$ are averaged densities (normalisation is straightforward in order to turn them into 'real' probability distributions). The local density axial distribution function -lzdf averages over all occupied grid cells per z-slice and divides by an effective area which is determined from the 'radius of gyration' of the density

$$R_G^2(z) := \frac{2 \int_0^{2\pi} d\phi \int_0^{R'} dr\, r^3\, n(r,\phi,z)}{\int_0^{2\pi} d\phi \int_0^{R'} dr\, r\, n(r,\phi,z)} \tag{2.48}$$

as explained in Section 2.1.4. This radius is a good approximation to the pore profile. The density itself is in the lzdf.xvg file.

For diagnostic purposes one can also plot the radial distributions of the unoccupied cells (holes in the grid) in order to find suitable grid spacings.

Options

Files

-grid	gridxdr.dat	Input		Generic data file
-profile	profile.xvg	Output		xvgr/xmgr file
-xyp	xyp.dat	Output		Generic data file
-xzp	xzp.dat	Output	Opt.	Generic data file
-yzp	yzp.dat	Output	Opt.	Generic data file
-rzp	rzp.dat	Output		Generic data file
-rdf	rdf.xvg	Output		xvgr/xmgr file
-zdf	zdf.xvg	Output		xvgr/xmgr file
-lzdf	lzdf.xvg	Output		xvgr/xmgr file
-hxyp	hxyp.dat	Output	Opt.	Generic data file
-hrzp	hrzp.dat	Output	Opt.	Generic data file
-hrdf	hrdf.xvg	Output	Opt.	xvgr/xmgr file
-dump	gridasc.dat	Output	Opt.	Generic data file
-plt	plt.dat	Output	Opt.	Generic data file

Other options

-h	bool	no	Print help info and quit
-nice	int	0	Set the nicelevel
-R	real	0	Maximum radius that should be contained in the grid
-z1	real	0	Center grid between z1, and ...
-z2	real	0	z2 (these boundaries are kept fixed!)
-delta	vector		
		0.02 0.02 0.02	[hidden] Spatial resolution in X, Y, and Z for resampling (in nm)
-minocc	real	0	[hidden] The occupancy of a cell must be larger than this number so that it is counted as occupied when calculating the volume, effective radius and local density axial distribution -lzdf. This is given in the chosen units (see -unit).
-subtitle	string		Some text to add to the output graphs
-mirror	bool	yes	mirror the radial projection P(r,z) to create the impression of a full view of the pore
-unit	enum	unity	divide number density (in nm^3) by 1, the density of SPC water at 300K and 1 bar, in mol/l or in Angstrom^3. Allowed values: unity, SPC, molar or Angstrom
-xfarbe-maxlevel	real	48.4841	xfarbe will plot 15 equally space level up to this density (the unit must be the same as for the -unit option!) Default is 1.5 SPC bulk.

`-holes`	bool	`no`	[hidden] calculate all hole distribution functions, using default filenames
`-radnr`	int	`0`	[hidden] block-average the first rad_ba_nr radial bins...
`-radstep`	int	`1`	[hidden] ...in blocks of rad_ba_step

Known limitations

- The radial bin width DeltaR is fixed to (Delta[XX]+Delta[YY])/2). In any case one should never have different bin widths in X and Y.
- There are still a few hidden options of questionable usefulness. Resampling (=changing Delta) is not implemented yet.
- gOpenMol plt binary file comes out with wrong suffix
- note: `-minocc` also influences `-lzdf`

D.2.5 g_wham

Usage

```
g_wham [OPTIONS] file1.pdo.gz file2.pdo ...
```

Synopsis

g_wham is an analysis program that implements the Weighted Histogram Analysis Method (WHAM).[174,175] It is intended to analyze `.pdo` files generated by mdrun with umbrella sampling to create a potential of mean force (PMF).

Data files (`pdo`) may be gzipped and are simply listed on the command-line. Only frames between the begin time `-b` and the end time `-e` are included, giving the opportunity to discard an initial equilibration phase (-1 means 'no limit'). The program will discard any data that is outside of the interval $[\min z; \max z]$. The program will output the true lowest and highest coordinate values after completion, so these values can be used the next time or alternatively with `-noprof` only min and max are calculated in a first pass.

Output:

The first column in both data files is the position z along the sampled coordinate in nm. The histogram contains the data for each individual window. The profile `-o` contains the unbiased distribution $Z(z)$ and the PMF $G(z)/kT = -\ln Z(z)$:

column:	1	2	3
quantity:	z	$Z(z)$	$G(z)$
units:	nm		kT

Files

`-o`	`profile.xvg`	Output	xvgr/xmgr file
`-hist`	`histo.xvg`	Output	xvgr/xmgr file

Other options

`-h`	bool	`yes`	Print help info and quit
`-nice`	int	`19`	Set the nicelevel
`-b`	time	`-1`	First frame (ps) to read from trajectory
`-e`	time	`-1`	Last frame (ps) to read from trajectory
`-tu`	enum	`ps`	Time unit: `ps`, `fs`, `ns`, `us`, `ms`, `s`, `m` or `h`
`-min`	real	`0`	Minimum coordinate in profile
`-max`	real	`0`	Maximum coordinate in profile
`-bins`	int	`100`	Number of bins in profile
`-prof`	bool	`yes`	Calculate profile, when 'no' only calculate min and max
`-temp`	real	`298`	Temperature
`-flip`	bool	`no`	Combine halves of profile
`-tol`	real	`1e-05`	Tolerance

History

g_wham is part of the gromacs package. It was broken in version 3.2.1. It was modified to allow processing of output from umbrella sampling data from mdrun and the format version was incremented to 3.2, breaking any backward compatibility. The new version can read data files in uncompressed and GNUzipped format. It allows sub-selections of the data through the `-b` and `-e` flags. The code has been cleaned up and modularised, making future enhancements for higher dimensional WHAM easier.

Bugs and Limitations

- This version of g_wham reads `pdo` files with version number 3.2. It will *not* read version 3.0 (produced by the mdrun binary from the official gromacs 3.2.1 distribution).
- Only 1D PMF along a box axis implemented.
- Symmetric profiles (`-flip`) do not work.

Example

The first 100 ps of data are discarded as equilibration (this is necessary but one should check for each system by calculating PMFs for different time windows, using the `-b` and `-e` options; the equilibration phase can be significantly longer than the data collection, e.g. 2.5 ns equilibration and 0.5 ns data). First the minimum and the maximum z values need to be found because they are need for setting up the histogram. The `-noprof` option reads in all data and prints the limits on standard output:

```
find .  -name '*.pdo*'
| xargs g_wham -noprof -temp 300 -b 100
```

With these limits (in this example, -0.9 nm $\leq z \leq 0.9$ nm) the WHAM procedure is carried out with

```
find .  -name '*.pdo*'
| xargs g_wham -temp 300 -b 100 -min -0.9
-max 0.9 -tol 1e-5 -bins 500 -o pmf.xvg -hist
histogram.xvg
```

The tolerance for the self-consistent solution of the WHAM equations is important. 10^{-3} kT is not enough. 10^{-5} kT is the minimum that resolves a test-input step function (generated with fakepmf, see Appendix D.3.2). It is recommened to plot the PMF for different values of the tolerance and check its convergence. The PMF is found in the output file `pmf.xvg`, and the histograms from all input files are found in `histogram.xvg`. They can be displayed with xmgrace,

```
xmgrace -block pmf.xvg -bxy 1:3
xmgrace -nxy histogram.xvg
```

D.3 Trajectory generation

D.3.1 Confinement in mdrun

Synopsis

mdrun from version 3.2.1 of gromacs was modified in the parts concerned with umbrella sampling in order to make it work together with the WHAM analysis program g_wham and to add an experimental cylindrical confinement potential. The output format was changed (current version of the format is now 3.2 which is incompatible with the previous version 3.0 but can be read by the modified version of g_wham). mdrun and g_wham are now versioned 3.2.1-LMB.

Umbrella sampling is activated with the `-pi` option. An input file `pull.ppa` must be provided, which controls all aspects of umbrella sampling, an index file which defines the group that is sampled, and a standard run input `tpr` file. Umbrella sampling can be performed in parallel.

Confinement potential

A cylindrical confinement potential was added. Outside the pore region a sampled particle is not bounded in the plane perpendicular to the umbrella potential. Thus the 1D PMF is not well defined and any results from WHAM are meaningless.[169,170] With a confinement potential, starting from outside a confinement radius R_C in addition to the umbrella potential, it should be possible to obtain a defined PMF. The flat-bottomed confinement potential has the form

$$U_C(\mathbf{x}) = \tfrac{1}{2}\, k_C\, \Theta[\rho(\mathbf{x}) - R_C]\, (\rho(\mathbf{x}) - R_C)^2, \tag{D.11}$$

where $\rho(\mathbf{x})$ denotes the distance from the line along which the umbrella windows are distributed (typically, just the z-axis), and k_C is the harmonic force constant.

Input file format

The input `ppa` file for umbrella sampling needs to comply to the following example format (lines starting with '`;`' are comments and are ignored by the parser).

```
; Input for umbrella sampling
; version: Gromacs 3.2.1-LMB

verbose   = yes
runtype   = umbrella       ; umbrella sampling
pulldim   = N N Y          ; restrict in Z dimension
reftype   = com_t0         ; compute center of umbrella potential
                           ; relative to initial COM of
reference_group = Protein  ; the reference group

; Umbrella sampling
; group which is subjected to the umbrella potential
; (see the index file):
group_1   = NA_-0.337
K1        = 3909.50;            ; kJ / (mol nm^2)
Pos1      = 0.000 0.000 -0.337  ; centre of the umbrella potential
; confinement potential: flat-bottomed cylinder
```

```
;                         centered on ref_com
KConfine = 4500.000000 ; harmonic force constant, kJ/(mol*nm^2)
RConfine = 1.500       ; radius past which the potential is
                       ; switched on (nm)
```

Up to four group sections (`group_1` to `group_4`) can be added. A confinement potential is only switched on if both `RConfine` and `KConfine` are non-zero; `RConfine` defaults to `0`, thus leaving out these variables will automatically run without confinement. The index group mentioned in `group_1` must be defined in the accompanying index file.

Output file format

The output `pdo` file has a header which is versioned with the output format number, currently 3.2. Data follows after the last header line #####, with the first column being the time in ps and subsequent columns the deviation from the umbrella centre of each of the `N_pullgroups` groups (in nm).

```
# UMBRELLA      3.2
# pulldim  0 0 1
# nSkip 1
# reference_group 'Protein'
# N_pullgroups 1
# Group 1 'NA_-0.337'  Pos -0.337000  K 3909.500000
#####
0.000000        0.003102
0.002000        0.002794
0.004000        0.002398
...
2999.996094     -0.029982
2999.998047     -0.029695
3000.000244     -0.029211
```

D.3.2 fakepmf

Purpose

Instead of deducing a potential of mean force from umbrella sampling the aim is to generate data from a known PMF. Then these data can be processed by g_wham (or any other unbiasing code) to recreate the original PMF. Comparing the initial PMF and the recreated one allows assessment of the unbiasing procedure.

The output is a "time series" of $\xi = z - z_i$ values as if sampled by a particle restrained to an umbrella window at position z_i, given the underlying PMF $W(z)$. The simplest approach is to randomly draw z values from the underlying statistical distribution $p(z)$ [Eq. (D.12)]. Although subsequent values drawn will be uncorrelated—in contrast to values from a dynamical trajectory (where there is strong correlation between close-by time steps)—the data are suitable as test input for unbiasing methods like the weighted histogram analysis method (WHAM).

Usage

```
fakepmf [OPTIONS]
fakepmf --zeta W_0 [OPTIONS]
```

Synopsis

fakepmf creates data files that mimic an umbrella sampling run of a known PMF, which corresponds to a probability distribution of (modified) Gaussians whose width depends on the force constant k of the umbrella potential and the underlying PMF. The default is a flat PMF but if `--zeta` W_0 is selected, a step function of height W_0 is simulated. The number of samples is determined by a fake run length (in ps) and a step size of 2 fs.

The output files have the same format as produced by mdrun (version 3.2.1-LMB) with umbrella sampling and can be directly fed into g_wham. Flat PMFs are stored in files with name `FLAT_z`z_i`.pdo.gz` whereas files with a step function PMF are named `STEP_z`z_i`.pdo.gz`.

Probability distributions

If the functional form of the PMF $W(z)$ is known then the probability distribution for the position of a particle z is

$$p_i(z) = \mathcal{Z}^{-1} \exp\left[-\beta\left(\tfrac{1}{2}k\,(z - z_i)^2 + W(z)\right)\right] \tag{D.12}$$

where k is the strength of the harmonic umbrella restraint, $\xi = z - z_i$ the deviation of the particle from the centre z_i of the i-th umbrella window, and $\mathcal{Z}^{-1}$ the normalisation

$$\mathcal{Z} = \int_{-\infty}^{+\infty} d\xi\, \exp\left[-\beta\left(\tfrac{1}{2}k\,\xi^2 + W(\xi + z_i)\right)\right]. \tag{D.13}$$

For a **flat PMF**

$$W(z) = W_0 = \text{const} \tag{D.14}$$

all p_i are Gaussians, centered at z_i with width $\sqrt{1/k\beta}$. The normalisation is trivial and the probability distribution for z becomes

$$p_i^{\text{flat}}(z; W_0) = \sqrt{\frac{\beta k}{2\pi}} \exp\left[-\tfrac{1}{2}\,\beta k\,(z - z_i)^2\right]. \tag{D.15}$$

For a **step function** of height W_0 at position z_0

$$W(z) = W_0\,\Theta(z - z_0) \tag{D.16}$$

the normalisation is

$$\mathcal{Z}^{\text{step}} = \sqrt{\frac{\pi}{2\beta k}}\left(1 + \text{erf}[\sqrt{\tfrac{1}{2}\,\beta k}(z_0 - z_i)] + e^{-\beta W_0}\ \text{erfc}[\sqrt{\tfrac{1}{2}\,\beta k}(z_0 - z_i)]\right)$$

and the probability distribution

$$p_i^{\text{step}}(z; z_0, W_0) = \frac{1}{\mathcal{Z}^{\text{step}}} \exp\left[-\beta\left(\tfrac{1}{2}\,k\,(z - z_i)^2 + W_0\,\Theta(z - z_0)\right)\right]. \tag{D.17}$$

Algorithm

The task is to draw values z according to a probability distribution $p_i(z)$ (D.12). The standard method is to use a uniform deviate x on the interval $0 < x < 1$ and transform it to yield a non-uniform deviate z obeying (D.12).[162] The random generator `ran1()` from Press et al.[162], p280 produces uniform deviates. For the flat PMF (D.14) the probability distribution (D.15) is a Gaussian, and the Gaussian deviate is easily obtained through a *Box-Muller* transformation (Ref. 162, p289) because the inverse of the cumulative distribution is available. The step function distribution Eq. (D.17) is more complictated but can be sampled using the rejection method (Ref. 162, p290). A comparison function $f(z)$ is required for which f-distributed deviates can be easily calculated and which envelopes the target distribution $p(z) \equiv p^{\text{step}}(z; z_0, W_0)$. The procedure is

1. to produce a f-distributed deviate z from a uniform deviate, then
2. to compare a second uniform deviate $0 < y < 1$ to the ratio $p(z)/f(z)$ and accept z as a p-distributed deviate if $y < p(z)/f(z)$.

The comparison function f was chosen as a non-normalised Gaussian

$$f(z) = 2\pi \max[p(z_0), p(z_i)] \exp[-\tfrac{1}{2}\beta k\,(z - z')^2], \text{ with}$$
$$z' = \begin{cases} z_0, & \text{if } p(z_0) \geq p(z_i) \\ z_i, & \text{if } p(z_0) < p(z_i) \end{cases}.$$

$f(z)$ is guaranteed to envelope $p(z)$ for all positions of the umbrella centre z_i, $1 \leq i \leq N$ and the position of the step barrier z_0 (see also Fig. 2.2 on page 49 for an illustration of $p(z)$).

Options

`-h,`	`--help`		this help
`-L,`	`--length`	L	Length of sampled stretch in nm [`1.9`]
`-N,`	`--nwin`	N	Number of windows [`101`]
`-k,`	`--forceconst`	k	Force constant in $\mathrm{kJ \cdot mol^{-1} \cdot nm^{-2}}$ [`7700.0`]
`-t,`	`--tsim`	$\mathfrak{T}$	'Simulation' time in ps, determines number of samples as the 'stepsize' is fixed at 2 fs [`1000.0`]
`-T,`	`--temperature`	T	temperature in K [`298.0`]
`-z,`	`--zeta`	W_0	height of step function in kT