Conducting Window-based Umbrella Sampling Simulations with NAMD and Analysis with TRAM

Justin Spiriti

August 23, 2022

1 Umbrella Sampling Simulations

1. briefly describe the concept of umbrella sampling, including the form of the umbrella potential and reaction coordinate, at a level suitable for undergrads
2. overview explicit solvent protocol (initial pathway, solvation, minimizing, heating, equilibration, production run)

Conventional molecular dynamics simulations involve simulating the motion of atoms using Newton's laws in a potential given by the force field

$$\begin{aligned} U_{\mathrm{FF}}(\mathbf{x}) \quad = \quad & \sum_{\text{all bonds}} \frac{1}{2} K_b \left(b - b_0\right)^2 + \sum_{\text{all angles}} \frac{1}{2} K_\theta \left(\theta - \theta_0\right)^2 + \sum_{\text{all torsion angles}} K_\phi \left[1 - \cos\left(n\phi + \delta\right)\right] \\ & + \sum_{\text{all nonbonded pairs}} \varepsilon \left[\left(\frac{r_0}{r}\right)^{12} - 2\left(\frac{r_0}{r}\right)^{6}\right] + \sum_{\text{all nonbonded pairs}} \frac{K_{\mathrm{coul}} q_i q_j}{r} \end{aligned} \tag{1}$$

The concept behind umbrella sampling is to add an additional term that forces the simulation to sample in different regions of a space defined by a reaction coordinate. The reaction coordinate is a function of the Cartesian coordinates that defines structural changes in the system. In this case we use the coordinates of the center of mass of the ligand relative to the protein after first aligning the protein-ligand complex via backbone with a reference structure. A harmonic term in terms of this center of mass is added to the force field potential.

$$U(\mathbf{x}) = U_{\mathrm{FF}}(\mathbf{x}) + \frac{k}{2} \left(\mathbf{r}_{\mathrm{CM}} - \mathbf{r}_{\mathrm{CM},0}\right)^2 \tag{2}$$

The harmonic term is centered on a position taken from a previously performed simulation designed to obtain at least one pathway for dissociation; this can be from a steered molecular dynamics simulation, a τ-RAMD simulation, a simulation from the "pathways" method described below, or some other technique.

This attempts to document how I performed the umbrella sampling simulations for PYK2 and the Markov state analysis using TRAM. I have tried to collect together all the scripts I used in the `scripts/` directory. Most of the scripts identify the simulation by a `name` parameter. The name I use is a combination of the name of the protein, the ligand, and the force field used, for example `pyk2-cpd1-amber`.

Needed software includes `tleap` from AmberTools, VMD and NAMD. If compounds need to be parameterized, it will be necessary to use Schrodinger Maestro, Gaussian, and `antechamber` from AmberTools as well.

1. It may be necessary to construct a force field for the compound. This requires performing a quantum chemistry optimization using Gaussian to obtain the electrostatic potential, to which the partial charges can then be fitted using the RESP method. The scripts require a `mol2` file for the compound, which is best obtained by loading a reference PDB file into Schrodinger Maestro, deleting everything except the ligand, making any needed chemical modifications, and saving the result as a `mol2` file. This ensures that atom names in the force field match those in the PDB file. the `set-up-gaussian3` script uses `antechamber` to set up two Gaussian input files. The first performs an optimization at the HF/MIDIX level (MIDIX is a small basis set optimized for reproducing geometries) and the second performs an optimization at the HF/6-31g* level using the output of the first as an initial guess. The `do-gaussian3.qsub` submission script runs both calculations. Finally, the `finish-parameterization` script completes the parameterization, using `antechamber` once again to fit the charges to the electrostatic potential. (The `do-cubes.qsub` and `draw-orbitals.tcl` scripts may be used to draw a picture of the orbitals.) These scripts are in the `scripts/params/` directory.

2. The "pathway" method is a simplified version of the weighted ensemble method designed to find initial pathways for ligand dissociation. It is an alternative to SMD or τ-RAMD for this purpose. It works by running a number of simulations in parallel for a short length of time, then selecting those in which the ligand has moved the most and starting new simulations from them. The `run-pathways.qsub` script is the master script that controls everything. The number of simultaneous simulations is set through the `nseg` variable. It should divide the total number of processors used (set by the `ntasks` option in the script header). The number of new simulations started at each iteration (which should divide the total number of simulations) is set through the `nstart` variable. The total number of iterations is given in the `for iter in `seq 1 200`` statement. The `segment-explicit.inp` and `segment-gb.inp` scripts are NAMD scripts that actually run the MD simulations; `select-new-simulations.py` is a Python script that selects the new simulations and writes this information to a log file.

3. The first step is to choose the centers of the umbrella potentials from among the configurations sampled in a previous SMD or τ-RAMD simulation. For τ-RAMD simulations the `get-com-all` script reads each τ-RAMD trajectory and constructs a file giving the ligand center of mass for each frame in each trajectory (calling VMD with the `get-relative-com3.tcl` script to accomplish its work. The `get-windows.py` script actually selects the windows. This script is important because it selects the spacing and extent of the windows. The parameters are

```
./get-windows.py list-of-coms interval limit > list-of-windows
```

where `list-of-coms` is the list of centers of mass generated by `get-com-all`, `interval` is the minimum distance between centers, and `limit` is the maximum distance for a center from the initial center of mass (which the script takes to be the center of mass in the original list of centers of mass, which comes from first frame of the first trajectory file originally considered). For the PYK2 simulations, the windows were 1.0 Å apart, and the limit was 15 Å. The output of `get-windows.py` is a list containing the window number, name of the trajectory and frame from which each center is selected, and coordinates of the center of mass . The `get-windows` script (which relies on the `align-frame.tcl` script) uses this file to extract the frames, align each one to a reference structure, and write them as individual PDB files in a directory named `frames`. All of the above-mentioned scripts can be found in the `scripts/extract/` directory.

4. Each of the extracted frames needs to be solvated, placing water molecules in a box around the protein and adding sodium chloride to make the system neutral and to give a salt concentration of 150 mM. This is accomplished using the `solvate-all` script which takes the following parameters:

   ```
   ./solvate-all name cpd nwindow
   ```

   The `cpd` parameter is the specific name of the ligand (for example `cpd1`); the script uses this to construct the names of the frcmod and lib parameter files that it will use (`cpd.frcmod` and `cpd.lib` respectively). The script calls tleap from AMBER with the `tleap-explicit-template.in` script, processing PDB files in the `frames` directory and placing solvated PDB files and corresponding inpcrd and prmtop files in the `data` directory. The `sed` statement in the script modifies each PDB file so that it can be processed by tleap, by relabeling hydrogen atoms in the amino group at the N-terminus and placing a `TER` marker between the atoms of the protein and atoms of the ligand. It must be modified to do these correctly for a new protein otherwise tleap will give errors. The `solvate-all` script also calls the `setup-ref.awk` script, which marks all non-hydrogen atoms in the protein and ligand with a 1.0 in the B-factor column (this is important for the restraints during the heating and equilibration phase), creating a "reference structure" that is placed in the `../ref` directory. All of these scripts can be found in the `scripts/solvate/` directory.

5. The systems need to be minimized, heated, and equilibrated. This is done with harmonic restraints on the heavy atoms of the protein and ligand, so that the water equilibrates to the protein-ligand complex and not the other way around. The systems are minimized in five phases with harmonic restraints on the heavy atoms of the protein and ligand and with force constants that descend in factors of 10 from 10^4 kcal/mol Å^2 to 1 kcal/mol Å^2; each phase consists of 1000 steps of minimization. Heating then takes place over 600 ps with the harmonic restraints at 1 kcal/mol Å^2, and with the temperature increasing by 10 K every 20 ps. Equilibration takes place over 400 ps in five phases with the harmonic restraints of 0.75, 0.5, 0.25, 0.1 and 0 kcal/mol Å^2. The NAMD scripts that perform these tasks are `pyk2-umbrella-minimize-amber.inp`, `pyk2-umbrella-heat-amber.inp`, and `pyk2-umbrella-equil-amber.inp`. The `pyk2-umbrella-heat.qsub` script may be used to submit to the Foundry queue (or modified for another cluster). These scripts can be found in the `scripts/min-heat-equil/` directory.

6. Finally, the production run can begin. It is necessary to run the `fix-ref.awk` script on each of the reference structures in order to re-mark each backbone atom in the PDB file with a 1.0 in the B-factor column. The main NAMD script is `pyk2-run-umbrella.inp` and the definition of the umbrella potential is in `colvar3.inp`. The force constant k for the umbrella potential is given by the `forceConstant` setting in the script. The MD simulation is run in 1-ns sections coordinated by the `do5` script. This script calls NAMD to run a section of dynamics, then strips the water molecules from the trajectory file for that section to save disk space. The master control script is `pyk2-run-umbrella-foundry.qsub`. The overall length of the simulation can be adjusted via the loop in this script. The These scripts are located in the `scripts/umbrella/` directory.

7. The trajectory segments need to be joined together to form complete trajectories. The `join-amber` script performs this. The center of mass of the ligand in each frame also needs to be determined. This is done using the `get-relative-com3.tcl` (which is slightly different from the one in the `scripts/extract/` directory).

8. Finally the free energy surface can be derived using WHAM. The source code for the WHAM program is `wham3d-1d3.f90` (it may be compiled with any Fortran compiler before use). The `create-control-file` script generates a "control file" that is the primary input for the WHAM program. The first line of this control file (which is output by `create-control-file`) contains several parameters, including the total number of data points for each window, the resolution of the grid for the three-dimensional histograms, starting and ending times, and the maximum number of WHAM iterations and convergence tolerance. The control file also contains the force constant for the umbrella potential, which must match what was specified in `colvar3.inp`. The program produces a DX file containing the three-dimensional free energy surface that may be visualized using VMD. This is done by importing it into VMD as a volumetric data set alongside the reference structure used for alignment, then adding a representation with the drawing style set to Isosurface and the surface style set to Solid Surface. The level of the contour may be set through the Isovalue box. (The WHAM program also outputs the values of $\ln Z_i/Z_0$; however there might be a minor bug that causes these to come out with the wrong sign.

2 TRAM scripts

These are in the `scripts/tram/` directory.

1. `tica-lag-time-validation-unbiased.py` – performs TICA analysis.

2. `do-cluster.py` – K-means clustering in TICA space

3. `get-ligand-com.py` – determine ligand center of mass in NumPy format for the benefit of the other scripts

4. `pyemma-tram-unbiased2.py` – construct the TRAM model

5. `plot-flux-colored2.py` – calculate and plot dissociation rates as a function of distance cutoff between unbound and bound states

6. `plot-flux-colored-pcca4.py` – construct macrostates from the TRAM model using PCCA and select representative frames

3 Sample data

The `scripts/sample-data/pyk2-cpd1-amber-tramd-solvated.zip` file contains `prmtop` files and starting structures for the solvated systems corresponding to the PYK2-compound 1 system. The structures need to have their occupancy and B-factor columns marked using either the `setup-ref.awk` or `fix-ref.awk` scripts.