Advanced Lab Course: BK.MDS

(tested for GROMACS 2019.6)

Introduction

Molecular dynamics simulations are the methods to study the dynamics of macromolecules such as proteins, DNA, RNA, lipids or synthetic polymers. The motions of individual atoms of a system are calculated here by numerical integration of Newton's equation of motion on a computer.

The aim of the experiment is to familiarize you with the basics of this method. Here, simulations of a short polymer chain, similar to a protein will be carried out. Simulations will be performed in the computer where motions of all atoms and energy will be recorded as a time trajectory. Various analysis methods will be explored such as the calculation of structural properties, free energies, and interpretation of different interaction energies.

Experiments

MD simulation of a 13-bead polymer chain

MD simulation of the polymer chain

In the first part, we will consider a 13-bead polymer chain as a minimal "toy" model of a protein. We will simulate the dynamics of this polymer chain by molecular dynamics. This system is very simple where "bonded" interactions between neighboring atoms are described by harmonic bending and stretching terms, and "non-bonded" interactions are described by the Lennard-Jones potential. The objective of this task is to learn the basic steps used in MD software such as GROMACS.

For all the simulations, three input files are usually required:

1. Coordinate file (gro/pdb): It contains atomic coordinates of each system to be simulated.
2. Topology file (top/itp): It contains the information about force-field parameters related to atoms and these parameters will be used to calculate potential energy during the simulation.
3. Simulation parameters file (mdp): It contains parameters which will be used to run the simulation such as type of simulation, no. of steps, time step, temperature etc (for more details: http://manual.gromacs.org/current/online/mdp_opt.html).

A. Simulation of folding/unfolding @ 300K

At first, we will simulate the system for 50 ns. Inside the **'mp-aa-fold'** directory, three input files (mdp_300K.mdp, poly-13.pdb and poly-13.top) are provided for this simulation. You may look into these file in any text editor.

Have a look at the starting structure in pymol by typing

pymol poly-13.pdb

Within pymol, a number of visualization options are available in menus on the right hand panel using the buttons “A” “S” “H” “L” and “C”. If you select ‘show cell’ under the “S” button, it will show the simulation box (you may have to zoom out). For this system, the simulated chain just exists in vacuum, without any solvent, but in most protein simulations the box is filled with water.

Next, you can issue the following commands (back in the terminal) to start the simulation:

```
gmx grompp -f mdp_300K.mdp -c poly-13.pdb -p poly-13.top -o topol.tpr -maxwarn 1
gmx mdrun -s topol.tpr -v
```

After simulation, these files will be generated:

traj_comp.xtc = trajectory file to be used for the analysis (Only contains atomic coordinates).

traj.trr = trajectory file containing both atomic coordinate and velocities.

ener.edr = energy file containing various type of energies and other parameters like temperature, pressure etc.

md.log = A log file of the mdrun

state.cpt = Checkpoint file to be used for a restart in case of crash

confout.gro = The final coordinates.

The trajectory of this simulation will be converted to a pdb file, and then visualized with pymol via the following command.

gmx trjconv -f traj_comp.xtc -o traj-aa.pdb -conect -s topol.tpr -dt 10

When trjconv asks for an index group, enter “0.” (The “-dt 10” flag sets the frame spacing to 10ps). Next load the trajectory in pymol with:

pymol traj-aa.pdb

In pymol, you can watch the chain fold and unfold. You can adjust the frame rate under the ‘Movie’ menu at the top of pymol. For convenience, a visualization script called ‘snake.pml’ is provided in the tools folder. You can view the scripts contents with any editor. To run it within pymol, type in the pymol terminal (the box above the 3D model of the molecule):

@../tools/snake.pml

We have simulated this chain at 300 K, and seen a number of transitions between a compact “folded” state and an extended “unfolded” state.

Question 1: In what ways is this polymer chain similar/different from an actual protein solvated in water? How is this simulated process of expansion/collapse similar and different to protein folding?

Question 2: If you had to guess, just from watching the movie, what is the ratio of compact to extended states? Can you estimate a free energy difference from this ratio?

Next, we will analyze the energetics of this process using g_energy.

```
gmx energy -f ener.edr -o energy-potential.xvg
```

and select "5" for the potential energy (and "0" to stop the selection). Look at the result with:

```
xmgrace energy-potential.xvg
```

Next, use g_energy to extract these additional energy components: bond angle bending ("G96Angle"), bond stretching ("bond") and non-bonded Lennard-Jones ("LJ-(SR)") energy. You can view combinations (2 or more) of plots together with, for example:

```
xmgrace energy-angle.xvg energy-LJ.xvg
```

The default color sequence in xmgrace is black/red/green/blue, so this color sequence will match the order you input files to xmgrace.

Question 3: In this simple system, the total potential energy is the sum of the angle bending, bond stretching, and non-bonded "LJ" terms. Are any of these energies correlated, anticorrelated or uncorrelated. Are the trends consistent with the folding/unfolding behavior you saw in pymol?

Question 4: From looking at 'energy-potential.xvg', estimate the difference in potential energy between the folded and unfolded states. Is the change in potential energy similar to your previous guess for the change in free energy? Given this, what can you infer about the change in entropy between the folded and unfolded states?

We have seen how the energetics and structure of the model protein are related. Now we will compute the radius of gyration (R_g) as a quantitative measurement of structure. For this, type

```
gmx gyrate -f traj_comp.xtc -s topol.tpr
```

and enter "0" at the prompt. Have a look at the output

```
xmgrace gyrate.xvg
```

The radius of gyration resolves the transitions between the compact and extended states much more clearly than the potential energy. For comparison, you can plot gyrate.xvg and energy-potential.xvg together with xmgrace. Because the potential energies are much larger in magnitude, you can scale them down within the xmgrace program using a tool in the menu: data→transformations→geometric transforms.

Question 5: Does the radius of gyration allow you to distinguish additional states besides fully extended ("unfolded") and fully compact ("folded")?

To proceed, we need to extract just the first two columns out of gyrate.xvg. For this use the provided script

```
../tools/xvg_trim.sh gyrate.xvg > rg.xvg
```

Next lets have a look at a histogram (probability distribution) to see if an intermediate state is evidenced in the radius of gyration data:

```
gmx analyze -f rg.xvg -dist hist-rg.xvg -bw 0.025
```

Question 6: From looking at hist-rg.xvg, what is your revised guess (still a ballpark estimate) for the free energy difference between states?

Question 7: Was there a structure you saw watching the trajectory with pymol that is likely to explain the intermediate values in the histogram (R_g around 0.85 nm)?

Question 8: Aside from discretization error, what information is lost in going from the time series rg.xvg to the probability distribution hist-rg.xvg? (hint: equilibration)

We have just seen that the radius of gyration is a good structural "order parameter" for distinguishing the different states of the system. Next, we will calculate the free energy explicitly as a function of the radius of gyration. In general, if we have, from our simulation, a probability distribution function p(ξ) for some order parameter ξ, then the free energy is $G(\xi) = -k_B T \ln(p(\xi)) + G_0$, where G_0 is an arbitrary additive constant. We can use a script to take the logarithm of the probability distribution, giving us the free energy 'pmf-rg.xvg':

```
../tools/pdf2pmf.py -f hist-rg.xvg -T 300 --units kJ/mol > pmf-rg.xvg
```

We now have a 1D free energy profile, also called a potential of mean force (PMF), that captures the folded, partially-folded and unfolded states (F-P-U, for short).

Question 9: The full configuration space for the 13 atoms is high dimensional, so doesn't it make more sense to compute the free energy in the full Cartesian coordinate space? What are the advantages and disadvantages of projecting all configurations onto 1D, as was done here?

Question 10: The free energy, projected onto a single coordinate R_g, suggests the F-P-U sequence of states. Is this sufficient evidence that folding and unfolding always pass through the intermediate "P" state?

Question 11: The (statistical) uncertainty in our computed free energy depends on that of the probability distribution p(ξ). How might you estimate the statistical uncertainty for this PMF? Is the uncertainty uniform? Or is it higher/lower in different places.

Next, we will consider a different order parameter, the end-to-end distance (between atoms 1 and 13) of our toy protein. We will follow the similar steps to calculate a time series and then make a free energy (PMF)

profile. Using the provided index file 'index-pairs.ndx' compute the end-to-end distance and extract the first two columns by calling

```
gmx distance -f traj_comp.xtc -s topol.tpr -n index-pairs.ndx -oall distance.xvg

../tools/xvg_trim.sh distance.xvg > dist.xvg
```

For gmx distance dist, enter "0" (end-to-end distance) and Ctrl+D to start the analysis. Next, generate a histogram of end-to-end distance and invert it to get the free energy with the commands

```
gmx analyze -f dist.xvg -dist hist-dist.xvg -bw 0.1

../tools/pdf2pmf.py -f hist-dist.xvg -T 300 --units kJ/mol > pmf-dist.xvg
```

Question 12: Judging by the 1D free energy profiles, is the end-to-end distance a better or worse order parameter than R_g?

For a better comparison, we will combine rg.xvg and dist.xvg to get a 2D picture of the system's conformational space. Zip the files together, using the provided script, and then view the results

```
../tools/xvg_zip.sh rg.xvg dist.xvg > xy-rg-dist.xvg

xmgrace xy-rg-dist.xvg
```

To improve the view, double click on the data and then in the 'symbol properties' box set type to circles and size to 10 (or 20) and in the 'line properties' box set type to None. This is not the free energy, but the density of points which is closely related.

Question 13: What are the main features of this 2D plot? Given the distribution of points, are direct F-U transitions likely or is the F-P-U sequence more likely? Can you resolve any other metastable states besides F,P, and U?

Next, zip together the R_g and the potential energy you computed earlier and have a look

```
../tools/xvg_zip.sh rg.xvg energy-potential.xvg > xy-rg-potential.xvg

xmgrace xy-rg-potential.xvg
```

Question 14: Approximate the change in potential between the folded and unfolded states from this plot. Use this value and the change in free energy from the free energy profile $G(R_g)$ to estimate the change in entropy between states (it is fine to compute $T\Delta S$). Is the sign of $T\Delta S$ consistent with your intuition? What do you expect to happen if you heat up or cool down the system?

B. Temperature dependence of folding

In the next step, we will repeat the previous protocol, but at reduced and elevated temperatures. The aim is to see the effect that temperature has on stabilizing and destabilizing different structures. Enter the 'mp-bb-temp' folder (which has the same starting files as before). To start, lets simulate at 270 K instead of 300 K. For this, rename the mdp file to mdp_270K.mdp. Next, open the file in an editor and set the 'ref_t' to equal 270. After you run grompp to prepare the run input (tpr) file,

```
gmx grompp -f mdp_270K.mdp -c poly-13.pdb -p poly-13.top -o topol.tpr -maxwarn 1
```

it is suggested that you make a subfolder (e.g. "run-270") in which to store the tpr file, run the simulation and carry out the analysis. (semicolons ';' below allow multiple shell commands to go on the same line)

```
mkdir run-270; mv topol.tpr run-270; cd run-270
```

```
gmx mdrun -s topol.tpr -v
```

Next, to speed up data processing you should make an analysis script that runs the analyses from the previous section. This is faster and less error-prone than typing the same commands into the command line over and over. To get you started, a script called 'do-analysis.sh' is provided. Open it in a text editor and have a look. From inside the run-270 folder, run the script with

```
../do-analysis.sh
```

and inspect the output files. Add the necessary commands to this script so that it computes the free energy for the radius of gyration, and test that it works. Once it works, you can test some different temperatures.

Question 15: How do the free energy profiles compare between 270 K and 330 K? (note that the vertical offsets for the free energy profiles are arbitrary) What does this suggest about the entropy and enthalpy of the states (F-P-U)? What happens to the F state at 400 K?

Have a look at the rg.xvg files at the different temperatures.

Question 16: How do the folding and unfolding rates (timescales) compare between 270 K and 330 K? Is 50ns a long enough simulation at 270 K for the free energy to converge?

C. Effect of chain bending stiffness on folding

Next, we will look at the influence of the chain bending stiffness on folding and unfolding the model protein. It is reasonable to expect that a very stiff chain will favor extended conformations and a soft chain will prefer compact conformations.

To start, make a new folder (e.g. 'mp-cc-stiffness'), and copy into it the usual starting files (mdp_300K.mdp, poly-13.pdb and poly-13.top). Whereas last time we edited the mdp file to change the simulation

temperature, now we will change the topology (top) file. Open poly-13.top and scroll down to the [angles] section. The sixth column is the spring constant, k_{angle}, currently 35 kJ/mol, the parameter we will change. Note that the function for the angle bending potential used takes a form as $V_{angle} = k_{angle}/2\ (\cos(\Theta)-\cos(\Theta_0))^2$. Close the file and rename it 'poly-13-k35.top' and make copies 'poly-13-k10.top' and 'poly-13-k45.top'. Edit the new files to have k_{angle}=10 kJ/mol and k_{angle}=45 kJ/mol respectively.

Set up and run md simulations for k_{angle}=10, 35 and 45 kJ/mol. It is advised to make separate run folders for each case. You can also copy the previous simulation files for k_{angle}=35 kJ/mol instead of re-running the simulation.

Copy your analysis script into this folder and use it to compute free energy profiles in terms of the radius of gyration.

Question 17: How do the free energy profiles compare for k_{angle}=10, 35 and 45 kJ/mol? Why is the trend similar or different to the trend in the previous section changing temperatures?

We have seen that the softer chain (k_{angle}=10 kJ/mol) prefers compact "folded" states with a low radius of gyration. Inspect this trajectory with pymol to see what the compact state looks like. The tool g_angle can help distinguish these compact states. First copy the file 'index-trips.ndx' locally, from the tools folder. This file contains triplets of atom indices used to compute the bond angles. Then run

```
gmx angle -f traj_comp.xtc -n index-trips.ndx
```

and look at the output 'angdist.xvg.' Repeat this for each of the other simulations and compare the distributions for different values of k_{angle}. This shows the distribution of angles for the full course of the trajectory.

Question 18: Is the simulation with k_{angle}=10 kJ/mol sampling the same compact 'pretzel' conformation observed in earlier cases? Is R_g a useful order parameter to distinguish different compact states?

D. An entropic spring

Here we will change the properties of our model protein so it behaves like an 'entropic spring.' Before running more simulations, a short exercise introduces the idea of the entropic spring.

Question 19: Consider a short chain, N=3 atoms, where the atoms can only occupy sites of a 2D square lattice and the bond length is fixed to the lattice spacing (bonds cannot go diagonally across a square). The atoms are hard spheres, meaning they cannot occupy the same site and have no attraction to each other. Assuming the first two atoms have a fixed position, how many possible states are there? What are the relative probabilities of different end-to-end distances? What about N=4 and N=5? Using the previous formula for free energy, $G(\xi) = -k_B T \ln(p(\xi)) + G_0$, to compute the free energies as a function of end-to-end distance for N=4 and N=5.

To simulate this situation, start by making a new folder (e.g. 'mp-dd-espring'), and copy into it the usual starting files (mdp_300K.mdp, poly-13.pdb and poly-13.top). We are going to hack the topology file to switch off (or drastically reduce) the Lennard-Jones attraction energy, making the atoms behave nearly as repulsive hard spheres. Open the top file and add a new atomtype (say P1). Under the [nonbond_params] section, set the P1 P1 Lennard-Jones parameters, C6 / C12, to be 0.22e-02 kJ·nm^6/mol and 0.22e-04 kJ·nm^{12}/mol respectively. Also change each of the atom types (in the [atoms] section) to P1 from P4.

Question 20: For this LJ potential, what are epsilon (energetic well depth) and sigma (atom-atom distance at the point where E=0)? You can use the g_tool called g_sigeps here.

Next, we will switch off the angle bending energy by setting k_{angle}=0. The easiest way to do this is to just comment out all the lines of the [angles] section, by adding a semicolon ';' to the start of each line. Now the potential energy is just the sum of the LJ interactions (which are small) and bond stretching energies. The protein is free to coil into any conformation as long as the bonds stay the same length and the atoms do not overlap.

Run an MD simulation using this new topology and look at the probability distribution (g_analyze) of the end-to-end distance.

Question 21: Assuming the probability distribution is Gaussian, how does the free energy formula above simplify? Is a Hookean (harmonic) spring a reasonable model for the computed free energy? What is the spring constant? (hint: 'g_analyze -dist' outputs a histogram and prints its variance)