REMD simulations in GROMACS

Overview

This tutorial comes in two parts. In the first, the theory behind replica exchange simulations will be briefly described. Then, we will look at how to perform temperature REMD on a small peptide in explicit solvent.

In the directory within the tarball for this tutorial, you will find two subdirectories, each with some files you can use to follow along with the tutorial. If you make a mistake, there's backups of input and output for each stage in the archive subdirectory for each stage. This tutorial assumes you are comfortable with basic GROMACS and UNIX usage, including using cd to move up and down through the directory hierarchy, cp to copy files, and ls to see what you have in the current directory. This tutorial is intended for use with GROMACS 2019.2. You may see minor differences if you use other versions of GROMACS.

Theory of replica exchange simulations

Many molecular simulation scenarios require ergodic sampling of energy landscapes that feature many minima. Often the barriers between minima can be difficult to cross at ambient temperatures over accessible simulation time scales. This means that results are often confounded by the choice of initial conditions, because the simulation never leaves that space.

Replica exchange simulations seek to enhance the sampling in such scenarios by running numerous independent replicas in slightly different ensembles, and periodically exchanging the coordinates of replicas between the ensembles. Typically, the set of replicas is constructed so that one extreme of the set of replicas is the ensemble from which sampling is wanted, and the other extreme is one where barriers can be crossed more easily. So, if

- the attempts to exchange replicas between ensemble produce valid ensembles,
- the probability of exchange attempts succeeding is high enough, and
- the resulting flow of replicas over the ensembles produces good mixing,

then the resulting sampling will be statistically correct and closer to the ergodic ideal.

How can this work? It relies on doing Monte Carlo moves of replicas between ensembles. The probability of observing a replica in a particular state depends on the potential energy and the temperature, i.e. $P(x) \propto \exp^{-\frac{U(x)}{k_B T}}$. The range of potential energies that is observed follows a normal distribution (because of the Central Limit Theorem). If two ensembles are chosen so that the set of states have significant overlap in the resulting potential energy distributions,

then when we observe a state in one ensemble there is appreciable chance that it could have been observed in the other.

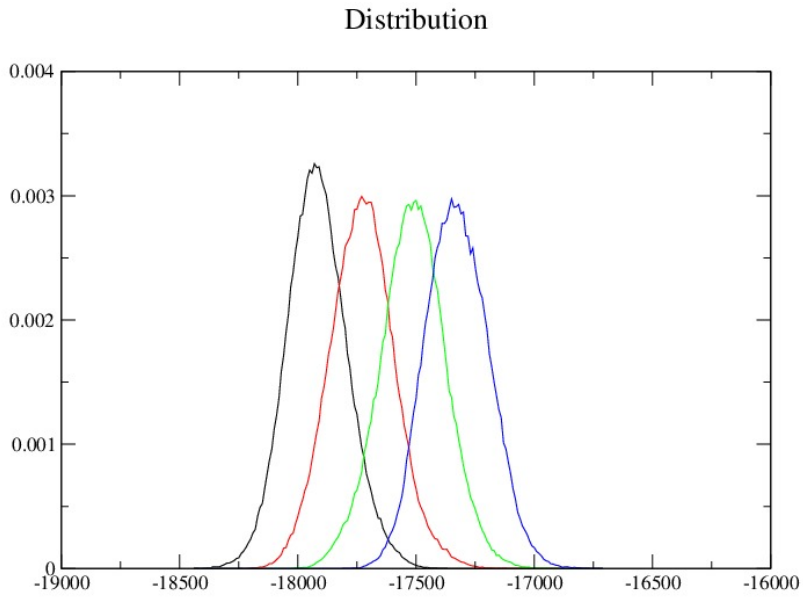

Figure 1: Potential energy distributions of alanine dipeptide replicas.

This allows us to construct a Metropolis-style Monte Carlo move that proposes that we exchange the coordinates found in two replicas based upon the probabilities that they would have been observed in the two ensembles. The exchange is accepted only if a random number has a suitable value. When done correctly, detailed balance is achieved, and the sampling in both ensembles is correct whether or not an exchange took place.

What range of ensembles should be chosen? The method of REMD (also known as parallel tempering) conducts the same simulation over a range of temperatures. We don't have to choose temperatures, but it's convenient and easy to understand. The highest temperature is chosen so that the barriers will be crossed. Replicas wander up and down through temperature space as exchanges are accepted. Barriers are crossed probabilistically at all of the temperatures, but at a higher rate at the higher temperatures. Those states filter down to the lower temperatures if they "belong" to the corresponding ensembles. In this way, the ergodicity of the sampling is enhanced.

For more details, see for example the many papers by Sugita Okamoto and Ulrich Hansmann.

Stage 1

Planning an REMD simulation

There are a number of inter-connected issues to consider in designing an REMD simulation:

- What range of temperatures do I want to span?
- How many replicas will I use?
- What exchange probability should I seek?
- How much compute resource can I use?
- How will I generate my starting coordinates for each replica?

In this tutorial, these decisions are made arbitrarily, but for real studies you will want to consult the literature for guidance here. Suggested reading includes papers by David Kofke, Ulrich Hansmann, and Mark Abraham.

Temperatures should often be distributed across the replicas in a geometric progression, i.e. with the same ratio used to scale each temperature from the one below it. If the energy landscapes are sufficiently similar across the temperature space, then this will keep the overlap of the potential energy distributions constant. In turn, this will keep the probability of accepting exchanges constant. Beware, though - in real simulations you will find bottle-necks in temperature space that restrict replica flow. You should be prepared to revise your temperature scheme after some testing.

As you can read in the above literature, an exchange acceptance probability around 0.2 is generally a good idea. Often you will have to experiment with the number of replicas you need to use to span the desired temperature range to achieve about that exchange probability. If you are simulating in the NPT ensemble, you might try this REMD temperature generator to help out.

Here, we will use only four replicas, so that the tutorial can run reasonably. Beware that you might want many hundreds of replicas to span a decent temperature range if you were looking at something like a large protein! The temperatures we will use are 298.00, 308.00, 318.34, and 329.02.

Preparing REMD equilibrations in GROMACS

Normally, gmx mdrun runs a single simulation. For temperature REMD we need to run a number of simulations that can communicate. This is done via the gmx_mpi mdrun −multidir mechanism, which runs N simulations within the same program. This requires an MPI-enabled version of GROMACS (usually installed as gmx_mpi mdrun), and N input .tpr files, each running at a different temperature. Naturally, we will have to equilibrate those to their target temperature before starting replica exchange, and that is most convenient to do with gmx_mpi mdrun −multidir!

You will find directories equil0, equil1, ... equil3 inside the stage1 directory. Each of those contains all the files needed to equilibrate a simulation at the right temperature. You can check the contents of the .mdp files with commands like

```
less equil0/equil.mdp
```

or if you want to check the reference temperatures for the thermostats in each, you can do that efficiently with a command like

```
grep ref-t equil*/equil.mdp
```

Now we want to run grompp in each directory to build the independent .tpr files we need. We could cd to each directory in turn and run gmx grompp, but that gets tedious fast. Since we're humans, we learn to use tools to do boring things for us. The bash shell lets you write a quick loop to do that operation. Here's how that looks

```
(for dir in equil[0123]; do cd $dir; gmx grompp -f equil -c confout; cd ..; done
```

This says to loop over equil0 ... equil3, and for each of those, change into the directory, run gmx grompp and then change back to the parent directory. This can be a tremendous time saver. Now if you do

```
ls equil*
```

you will see that there is a .tpr file in each directory. Now we're ready to roll! Use

```
mpirun -np 4 gmx_mpi mdrun -v -multidir equil[0123]
```

to run the simulation. (Perhaps you will need mdrun_mpi rather than gmx_mpi mdrun on some clusters.) Use ls again, and you will see that each of those directories now has a .log and .edr file corresponding to the run you just did.

The .mdp file you just used only runs a few steps, because you might not have time to wait for an equilibration to actually run on your machine. If you want to do a proper equilibration, edit all the equl.mdp files to set nsteps = 100000. This will run 200 ps of equilibration, so go and have coffee or something while you wait!

Stage 2

Running a REMD simulation in GROMACS

In this stage, we'll take the output from an equilibration run of decent length and use that as input for the REMD simulation proper. Change to the the stage2 directory. In each sim0 ... sim3 sub-directory, you will see the usual collection

of input files, plus the .cpt file that is the CheckPoinT file from an equilibration run. The sim.mdp input file in each directory has a different temperature, and is set up to expect to continue the matching equilibration run. We can see that difference if we run commands like

```
diff ../stage1/equil2/equil.mdp sim2/sim.mdp
```

Also, you can see that we are running a continuation, rather than generating new velocities! The only differences between the simulation .mdp files are in the reference temperatures can be seen with commands like

```
diff sim[12]/sim.mdp
```

Now let's use gmx grompp to prepare the input .tpr files, again using bash to do monkey work for us

```
(for dir in sim[0123]; do cd $dir; gmx grompp -f sim -c confout -t equil-state;
```

This time we told gmx grompp to use the state from the equilibration .cpt file. OK, it's time to run the REMD simulation with

```
mpirun -np 4 gmx_mpi mdrun -v -multidir sim[0123] -replex 100
```

This run will take a bit longer than the equilibration run, but is still only a "toy" run. We triggered the use of REMD with the −replex flag, which also specified the number of MD integration steps that should take place between exchange attempts. There's a bit of disagreement in the literature about how long that should be. Waiting around 1 ps seems wise in view of how long it takes to make an independent observation of the potential energy, so with a 2 fs time step in the .mdp file, around 500 steps between exchanges will be OK. Above, we did exchanges more often, just for the purposes of getting things to happen during the tutorial.

So, what happened? Let's look at one of the four .log files that were written, with

```
less sim1/md.log
```

and search through it for the output peculiar to replica exchange. Fortunately, all of that is prefixed with the same string, so we can search for that. Enter /Repl and press return you can skip to the first bit of that. Pressing n will then move on to the next occurence. 1 Shift−g will take you back to the top of the file, too.

You can see that mdrun did some checking that you gave it a consistent set of input .tpr files. Further on you might see output like

```
Replica exchange at step 100 time 0.2
Repl 0 <-> 1  dE_term =  3.505e+00 (kT)
Repl ex  0    1    2    3
Repl pr   .03       .03
```

Here we see that at step 100 (which was the first attempt), the time was 0.2 ps. The potential energy of the state in ensemble 1 was about 3.5 kT higher than that of the state in ensemble 0, which meant that an exchange would only succeed about 3% of the time. Since this is the .log file for simulation 1, we can't see the details for the exchange attempt between simulations 2 and 3, but if we went to their .log files, we would see.

The next exchange attempt at step 200 in my run was a bit different (yours will be different again)

```
Replica exchange at step 200 time 0.4
Repl 1 <-> 2  dE_term = -1.420e+00 (kT)
Repl ex  0    1 x  2    3
Repl pr         1.0
```

At replica-exchange steps, GROMACS alternates between two disjoint sets of replica pairs, so that each replica attempts an exchange with both of its neighbours once every two attempts. This time, replica 2 had a potential energy that was lower than replica 1, so the exchange is certain to succeed. Do read up on the Metropolis criterion if you need more information here.

Down near the bottom of the .log file you can see some summary statistics where the average exchange probabilities and number of exchange events are tabulated. In a real simulation, you might see that the distribution is not very uniform despite having run for several nanoseconds, and if so you might want to adjust your temperature scheme. Check out the literature for details.

The REMD statistics are accumulated only over each run, and are not stored in the .edr or .cpt files, and can be lost if you use .log file appending. So if you think you will want to post-process all your REMD statistics from a run in multiple stages, you will have to manage preserving all your .log files intact.

This REMD simulation was also far too short to show anything useful.

De-multiplexing an REMD trajectory in GROMACS

There are two ways the programmer could implement REMD. At exchange attempts the replica simulations could exchange coordinates, or they could exchange temperatures. Other MD packages tend to exchange temperatures; GROMACS exchanges coordinates. This means that the trajectory written to a single file belongs to the ensemble described by the matching .tpr file. Often this is convenient, because you only want to look at the ensemble at your lowest temperature. Because of the exchanges, that trajectory will no longer be continuous - some totally different structure will appear every time an exchange happened. This will mean the time evolution of observables like RMS deviation over time from some other structure will show abrupt jumps.

There are some times you want a trajectory that has continuous coordinates, despite the "jumps" in ensemble space. This means chopping up each trajectory file into pieces and pasting them back together in the right order. The Perl script demux.pl installed with GROMACS will analyse a single REMD .log file and writes two special .xvg files that describe the transformation matrices for the trajectory time series from one form to the other, and back. trjcat −demux can use these files to do the cut-and-paste on the trajectory files for you.

Options for other investigations

- Solvated systems - many more degrees of freedom, so need more hardware to run the simulation, and more replicas to span a given temperature range
- REST (Replica Exchange with Solute Tempering) available as a plugin to GROMACS

Conclusion

Here we've scratched the surface of the possibilities of replica-exchange. There are lots of fancy ways to change the physics of the replicas from which you don't want to sample, in order to enhance the sampling at the one you do want to sample, including actually changing the Hamiltonian!