Guest Lecture, Smith College, CS 334, BioInformatics
16 October 2008
GROMACS, MD Tutorial
Filip Jagodzinski

1 Introduction

GROMACS (**GRO**ningen **MA**chine for **Ch**emistry **S**imulation) is a molecular dynamics (MD) simulation package originally developed at the University of Groningen. The highly optimized code makes GROMACS the fastest program for molecular simulations to date.

The source code for GROMACS is freely available at `http://www.gromacs.org`; there you will also find complete documentation as well as a trouble-shooting guide. A wiki page is also available at `http://wiki.gromacs.org`.

GROMACS is invoked via several command-line options; there is no user-interface, per sae. GROMACS is a program for the Unix environment. Please visit the GROMACS website for a complete list of functions that you can invoke from the command line. Detailed information about all of the routines that are used in this tutorial can be found online. A simple Google^{TM}search will suffice. The following GROMACS routines are invoked in this tutorial: `pdb2gmx, editconf, genbox, grompp, mdrun` and `ngmx`.

The contents of this tutorial assume that you are working on the beowulf computer cluster at `csc.smith.edu`. GROMACS has been installed on the beowulf cluster, and the library of GROMACS functions are all available; you do not have to install GROMACS. Throughout this tutorial, text that is intended to be typed at the console is written within a box, such as:

```
command intended for command prompt
```

The following files are being provided, and it is assumed that they are in a directory `md_tutorial`. If they are located in a different directory, you'll need to adjust the tutorial accordingly whenever a reference is made to `md_tutorial`:

- em.mdp - parameter file for the energy minimization MD step
- pr.mdp - parameter file for the position restrained MD step
- md.mdp - parameter file for the main MD step
- 1hhp.pdb - a PDB file
- 1ako.pdb - a PDB file
- 1bhs.pdb - a PDB file

2 Summary of Tutorial

This tutorial is composed of the following sections:

1. Choosing a protein

2. GROMACS preliminaries - is it installed?
3. Creating a topology file for pre-processing
4. Creating and solvating a simulation box
5. Energy minimization - the solvent
6. Position restrained MD - filling in the spaces
7. Creating topology file for MD run
8. Performing the MD simulation
9. Viewing the MD trajectory files
10. Viewing the MD Simulation Results

3 MD Tutorial

3.1 Choosing a protein; using PyMOL

You first must select a protein (or protein fragment) on which you want to perform MD. PyMOL (`http://pymol.org/`) is a freely available molecular viewing program that has been installed on the beowulf cluster; you'll use PyMOL to view several proteins, one of which you'll select for your MD simulation.

The protein data bank (PDB, `www.pdb.org`) is a repository of X-ray crystallographic data of proteins. Figure 3.1 lists the proteins that have been retrieved from the PDB and which have been saved to the beowulf cluster.

When you have logged into your Linux account, navigate to `md_tutorial` directory; there you will see the saved PDB files as well as three files `em.mdp, md.mdp`, and `mdout.mdp` that are used in this tutorial. Be sure to log into your Unix account, and not your Windows account. Next, invoke PyMOL from the command prompt:

```
pymol
```

When you start pymol, a GUI interface will open. Select `File -> open` from the user menu, and select and open that PDB file that you want to view and for which you'll eventually perform an MD simulation. When you open and view the protein, use the mouse to rotate the protein, to zoom in and out, to perform measurements, etc. PyMOL offers a wide range of features, similar to many other protein viewing programs; explore these features at your leisure. While you are using PyMOL, take some time to ask some questions about the protein that you are viewing. For example:

- What are the prominent features of the protein?
- Where do you think the protein will exhibit motion (if any)?
- Are certain parts of the protein "clumpy", while others less so?

PDB ID	Protein Size	Description
1bhs	284 Residues	Human estrogenic dehydrogenases play a crucial role in the last steps of the synthesis of estrogens. 1bhs is the human estrogenic 17β-hydroxysteroid dehydrogenase.
1ako	268 Residues	Exonucleases are enzymes that cleave nucleotides one at a time from an end of a polynucleotide chain. Exonucleases are involved in many cell processes, including the clipping of the RNA primer region during DNA synthesis. 1ako is the exonuclease from *escherichia coli*, a bacteria.
1hhp	99 residues	HIV1 Protease is a protease that hydrolyzes HIV viral polyproteins into functional protein products that are essential for viral assembly and subsequent activity of HIV.

Figure 1: The 3 PDB files have been selected specifically for this lab because of their appropriate length, which is conducive to an MD simulation that can be performed in a short amount of time.

3.2 GROMACS preliminaries - is it installed?

Before you can proceed, you need to make sure that all of the routines in GROMACS are available and that they are in your path. While logged into your lab Unix account, invoke the following:

```
which mdrun
```

The `which` command searches all of the directories in your system path variable for the one parameter that you provide, which in this case is the executable `mdrun`, which is one of the functions in GROMACS. Thus, if the `mdrun` routine is available, then it means that you have access to GROMACS. When you invoke the `which` Unix command, the system will either locate `mdrun` and provide you with the directory where it can be found, else the Unix system will prompt you with nothing. If when you issue the command, you receive `/usr/local/gromacs/bin/mdrun`, or something similar, then GROMACS is in your path. If instead the system does not return anything, you must add GROMACS to your path by invoking the following:

```
set path = ($path /usr/local/gromacs/bin)
```

Then, again invoke the `which` command to make sure that GROMACS is in your path.

Once you've confirmed that the GROMACS routines are available for you to use, you'll want to copy the PDB file that you want to simulate in GROMACS to a common name that coincides with the name of the protein that is used in the tutorial. The copy, `cp`, command in the following example copies the file `1HHP.pdb` to `protein.pdb`, but for the tutorial, replace the name `1HHP.pdb` with the name of the protein whose motion you want to simulate. Because GROMACS produces many intermediate files, it is suggested that you create a new directory, `mdrun1`, and copy all of the required files into that directory. That way, you'll be sure not to overwrite the original text files that are in your `md_tutorial` directory:

```
cp 1HHP.pdb protein.pdb
```

```
mkdir mdrun1
```

```
cp * mdrun1/.
```

Finally, cd into the `mdrun1` directory:

```
cd mdrun1
```

3.3 Creating a topology file for pre-processing

To run an MD simulation, you need to prepare several files. These files are what you use to indicate to GROMACS exactly how you want to perform the simulation (for how long), and how you want to set up the simulation environment (should the protein be in water? in alcohol?, etc).

First you need to create a molecular topology file that is used by GROMACS. A topology file (.top extension) is generated by the program *pdb2gmx*. The only input for the *pdb2gmx* program is your `protein.pdb` file. Because most pdb files that have been calculated from X-ray crystallography experiments do not contain all hydrogen atoms, the `pdb2gmx` program will also add hydrogens to your protein. The output file which contains the structure of the protein when hydrogen atoms are added is a GROMACS structure file (.gro extension.).

```
pdb2gmx -f protein.pdb -o protein.gro -ff G43a1 -p protein.top -i posre.itp
```

When you invoke a GROMACS command, you'll notice many things flash across the screen. These are indicators that the program that you have invoked is running. Often-times, the information that is printed to the screen will detail exactly what was performed. If an error occurs, it will be displayed on the screen. You may want to look over the output to get a sense of what exactly occurs when you invoke a GROMACS routine. If you are prompted to choose the force field, enter `0` (the numbre zero).

3.4 Creating and solvating a simulation box

Biological structures such as proteins are not just "floating" around in space, and so you must solvate your peptide in a box of water; this will make the simulation a bit more "realistic". You'll use the program `editconf` to define a box in which you'll place your protein (and in which the MD calculations will be performed), and `genbox` is the program that is used to add water around your protein.

The `genbox` program reads the protein structure file and an parameter input file containing the sizes of the desired water box. The output of `genbox` is a GROMOS structure file of a peptide solvated in a box of water. The `genbox` program also changes the topology file (.top extension) to include water. First use the program `editconf` to define the right box size for your simulation.

```
editconf -f protein.gro -o protein.gro -d 0.85
```

Where the 0.5 is a numerical indicator that is used to determine the minimum distance that any atom must be from the edge of the simulation box. Thus, you are making sure that the protein is wholly encased in the box, with a certain amount of "padding" for good measure. Next, create the

simulation box:

```
genbox -cp protein.gro -cs -o protein_b4em.gro -p protein.top
```

where the `-o` option designates the output file, and the `-p` option designates the topology file that will be output.

3.5 Energy minimization - the solvent

So, at this moment you've "dumped" water into your simulation box. Was this done correctly? Did you violate any laws of nature? Well, the function `genbox` adds water molecules indiscriminately, so there's a good chance that there exist several water molecules that are too close to each other, or might even overlap or occupy nearly the same space, which is clearly not allowed. These close interactions result in very high repulsive energies. If you were to start a MD simulation right now, the system would be unstable because of these high energies.

The standard procedure to remove these close contacts is Energy Minimization (EM). Energy minimization perturbs the coordinates of individual atoms until the very high repulsive energies are eliminated (or at least reduced as much as possible). In a way, EM "shakes" the entire system until close contacts are reduced; this, in a way, is what happens in nature – a system tends towards a lowest energy state.

Before you can start the EM step, you need to pre-process all the input files with the GROMACS preprocessor function `grompp`, which pre-processes the topology file (.top), the structure file (.gro) and a parameter file (.mdp) resulting in a binary topology file (.tpr extension). This binary topology file contains all information for a simulation (in this case an energy minimization). The file `em.mpd` is a text file that contains the parameters for the EM step of this simulation. First take a look at the parameter file to see what you will be instructing GROMACS to do:

```
more em.mdp
```

Notice the `dt` and `nsteps` variables. What do you think they mean? The `dt` variable designates the amount of time that should elapse between each new calculation of the energies and motions of each atom, while `nsteps` designates the number of total steps in the simulation.

Now you are ready to invoke `grompp`:

```
grompp -f em -c protein_b4em -p protein -o protein_em
```

Now that you've generated the binary topology file, you can start the EM step. The program which performs the EM step is called `mdrun`. In fact all simulations are performed by `mdrun`.

As the EM step is running, watch the output of the program. The first number (from left to right) is the number of the iteration step. The second number is the step size, which gives an indication of the change in the system. The third number is the potential energy of the system. This number starts at a high value and rapidly drops down, and converges, to a large negative value. The lower the energy of the system, the more stable it is, and the fewer unfavorable interactions there exist. Thus, if the energy of the system is going down, it is a sign that those close water contacts are

being removed.

```
mdrun -nice 4 -s protein_em -o protein_em -c protein_b4pr -v
```

3.6 Position restrained MD - filling in the spaces

Once all close contacts are removed from the system, you'll want to do molecular dynamics of the water molecules, and keep the peptide fixed. This is called position restrained (PR) MD, and the purpose is to let all water molecules equilibrate around the peptide in order to fill holes, crevices, etc. which were not filled by the `genbox` program.

First you'll need to pre-process the input files to generate a binary topology file. The input files are the topology file, the structure file (output of the EM), a parameter file, and an index file.

The parameter file (.mdp extension) contains all information about the PR MD step, including step size, number of steps, temperature, etc. This Parameter file also tells GROMACS what kind of simulation should be performed (for example EM, PR-MD or MD). The parameter file `pr.mpd` has been provided for your. Again, view the contents of the file to see what parameters you'll be setting:

```
more pr.mdp
```

Again, what do you think `nsteps`, `dt`, and `ref_t` mean? Now invoke the `grompp` program:

```
grompp -f pr -c protein_b4pr -r protein_b4pr -p protein -o protein_pr
```

Once the GROMACS file has been created, you can start the PR MD. Note that in this example the simulated time (1ps) is too short to equilibrate your system completely; during actual molecular dynamics simulations that are performed in laboratories, the PR MD step might be allowed to run for a long time, upwards to several days, just to make sure that all of the water molecules have filled any gaps and that the system has been equilibrated.

```
mdrun -nice 4 -s protein_pr -o protein_pr -c protein_b4md -v
```

When the PR MD step concludes, a detail of what calculations were performed is printed to the screen.

3.7 Creating topology file for MD run

Now your complete system is finally ready for the main MD step. Again you need to pre-process the input files by using the `grompp` program to generate the binary topology file (.tpb/.tpr extension). Again, the parameter file has been included for you; take a look at it and notice all of the variables that you'll be using:

```
more md.mdp
```

Now invoke the preprocessing program:

```
grompp -f md -c protein_b4md -p protein -o protein_md
```

3.8 Performing the MD simulation

Now you can start the MD simulation. As the simulation is running, you can gauge its progress by looking at the step number that the simulation is currently on (if you did not alter the `nsteps` variable in the `md.mdp` file, the total number of steps in your simulation will be 20,000, which, if every step is 2 femtoseconds, will produce a simulation that is 40 ps in length):

```
mdrun -nice 4 -s protein_md -o protein_md -c protein_after_md -v
```

Again, look at the data for the calculations that were performed. What is a "flop"? Which of the calculations assumed the largest amount of processor resources? Why?

3.9 Viewing the MD trajectory files

Now that MD has concluded, take a look at the directory to see what intermediate files were generated:

```
ls -l
```

Notice that some of the files are quite large, namely, `protein_md.trr`, which is the trajectory file of the entire simulation. A trajectory file is a record of the atoms in a system and their displacement during the course of the simulation. So, think of the trajectory file as a series of snapshots, sequentially concatenated.

3.10 Viewing the MD Simulation Results

Thus, if you were to "flip" through the snapshots, you have a movie! GROMACS comes with a simple program that allows us to view a trajectory file, and hence see the computed motion of the protein:

```
ngmx -f protein_md -s protein_md
```

Once `ngmx` is open, you'll be prompted with an option to indicate what atoms of the simulation you want to view. If you choose "system", all of the water molecules as well as the protein will be displayed. If you want to view only the backbone of your protein, select "backbone". You can always change which atoms you want to view by selecting `filter` from the `display` option.

To see all of the trajectories and to see the simulation of the motion of the protein, select the `animate` option from the `display` display menu. You'll be presented with the standard "play", "fast forward", "rewind", etc. options. Using these buttons, you can view the entire simulation else progress through it step-by-step.

4 Concluding remarks & questions

Is the simulation what you expected it to be? Does your protein "move" as you expected? Why all of the "jiggling"? Can we be sure that the simulation is "valid"? That it is "accurate"?

And, now that you know how to run GROMACS, you may want to run a simulation that is a bit longer than the few nanoseconds that we performed in this tutorial. Also, you may want to perform a simulation on a larger protein. But be warned, because longer simulations on larger proteins can take days, weeks, months, or even YEARS of computation time!