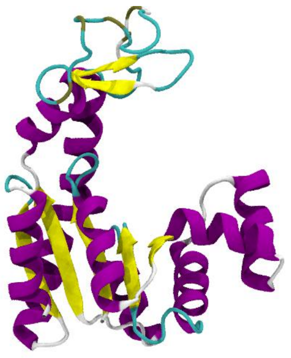
AdKGromacsTutorial Documentation

Release 1.1

Sean Seyler and Oliver Beckstein

Sep 27, 2017

Contents



CHAPTER 1

Objective

Perform an all-atom molecular dynamics (MD) simulation—using the Gromacs MD package—of the apo enzyme adenylate kinase (AdK) in its open conformation in a physiologically realistic environment, and carry out a basic analysis of its structural properties in equilibrium.

CHAPTER 2

Tutorial files

All of the necessary tutorial files can be found on GitHub and can be obtained by cloning the repository:

```
git clone https://github.com/Becksteinlab/AdKGromacsTutorial.git
```

CHAPTER 3

Workflow overview

For this tutorial we'll use Gromacs (version 5.1.3) to set up the system, run the simulation, and perform analysis. An initial structure is provided, which can be found in the `tutorial/templates` directory, as well as the MDP files that are necessary for input to Gromacs. The overall workflow consists of the following steps:

Directory organization

The workflow for setting up, running, and analysing a simulation consists of multiple and rather different steps. It is useful to perform these different steps in separate directories in order to avoid overwriting files or using wrong files.

Create working directories

It is recommended that the following directory structure be used, as the tutorial steps through them sequentially:

```
coord/
top/
solvation/
emin/
posres/
MD/
analysis/
```

Create these directories using:

```
mkdir top solvation emin posres MD analysis
```

Description of directories

`coord` original PDB (structural) files

`top` generating topology files (`.top`, `.itp`)

solvation adding solvent and ions to the system

emin performing energy minimization

posres short MD simulation with position restraints on the heavy protein atoms, to allow the solvent to equilibrate around the protein without disturbing the protein structure

MD MD simulation (typically, you will transfer the `md.tpr` file to a supercomputer, run the simulation there, then copy the the output back to this trajctory)

analysis post-processing a production trajectory to facilitate easy visualization (i.e., using VMD); analysis of the simulations can be placed in (sub)directories under analysis, e.g.

```
analysis/RMSD
analysis/RMSF
...
```

The subdirectories depend on the specific analysis tasks that you want to carry out. The above directory layout is only a suggestion, but, in practice, some sort of ordered directory hierarchy will facilitate reproducibility, improve efficiency, and maintain your sanity.

Important: The command snippets in this tutorial assume the directory layout given above as the workflow depends on each step's being carried out *inside the appropriate directory*. In particular, *relative* paths are used to access files from previous steps. It should be clear from context in which directory the commands are to be executed. If you get a `File input/output error` from **grompp** (or any of the other commands), first check that you are able to see the file by just doing a `ls ../path/to/file` from where you are in the file system. If you can't see the file then check (1) that you are in the correct directory, (2) that you have created the file in a previous step.

Obtain starting structure

Note: The starting structure `coord/4ake_a.pdb` has been provided as part of the tutorial package, so the instructions that follow are optional for this tutorial. However, these steps provide an idea of what may be required in obtaining a suitable starting structure for MD simulation.

1. Download 4AKE the Protein Data Bank (PDB) through the web interface
2. Create a new PDB file with just chain A

   Modify the downloaded PDB file. For a relatively simple protein like AdK, one can just open the PDB file in a text editor and remove all the lines that are not needed.(For more complex situations, molecular modeling software can be used.)

- Remove all comment lines (but keep TITLE, HEADER)
- Remove all crystal waters (HOH) **[#crystalwaters]_**
- Remove all chain B ATOM records.
- Save as `coord/4ake_a.pdb`.

Generate a solvated protein system

Generate a topology

Using the modified PDB file (chain A of 4AKE with crystal waters removed), generate a topology file for the CHARMM27 force field together with the TIP3P water model using the pdb2gmx tool:

```
cd top
gmx pdb2gmx -f ../coord/4ake_a.pdb -o protein.pdb -p 4ake.top -i protein_posre.itp -
↪water tip3p -ff charmm27
```

Note: Total charge -4.000e (in the next step we will add ions to neutralize the system; we need a net-neutral system to properly handle electrostatics)

Solvate the protein

Adding water

Create a simulation box with editconf and add solvent with solvate:

```
cd ../solvation
gmx editconf -f ../top/protein.pdb -o boxed.pdb -c -d 0.5 -bt dodecahedron
gmx solvate -cp boxed.pdb -cs spc216 -p ../top/4ake.top -o solvated.pdb
```

Attention: In order to reduce the system size and make the simulations run faster we are choosing a very tight box (minimum protein-edge distance 0.5 nm, `-d 0.5`); for simulations you want to publish this number should be 1.2...1.5 nm so that the electrostatic interactions between copies of the protein across periodic boundaries are sufficiently screened.

solvate updates the number of solvent molecules ("SOL") in the topology file (check the `[ system ]` section in `top/system.top`)[2].

Adding ions

Ions can be added with the genion program in Gromacs.

First, we need a basic TPR file (an empty file is sufficient, just ignore the warnings that **grompp** spits out by setting `-maxwarn 10`), then run **genion** (which has convenient options to neutralize the system and set the concentration (check the help!); **genion** also updates the topology's `[ system ]` section if the top file is provided[2]; it reduces the "SOL" molecules by the number of removed molecules and adds the ions, e.g. "NA" and "CL").

```
touch ions.mdp
gmx grompp -f ions.mdp -p ../top/4ake.top -c solvated.pdb -o ions.tpr
printf "SOL" | gmx genion -s ions.tpr -p ../top/4ake.top -pname NA -nname CL -neutral␣
↪-conc 0.15 -o ionized.pdb
```

[2] The automatic modification of the top file by **solvate** and **genion** can become a problem if you try to run these commands multiple times and you get error messages later (typically from **grompp**) that the number of molecules in structure file and the topology file do not agree. In this case you might have to manually delete or adjust the corresponding lines in :file"*system.top* file.

The final output is `solvation/ionized.pdb`. Check visually in VMD (but note that the dodecahedral box is not represented properly.)

Energy minimization

In order to remove "clashes" (i.e. close overlaps of the LJ cores) we perform an *energy minimization*: Instead of a MD simulation we use an algorithm to change the coordinates in such a way as to reduce the total potential energy.

Set up and generate the run file

First, we will copy a file from the templates folder (provided in this tutorial) that tells Gromacs MD program *how* to do energy minimization:

```
cp ../templates/em.mdp .
```

Tip: Have a look at the MDP file to get a feel for what kinds of settings can be adjusted to suit one's needs. Individual parameters are explained in more detail in mdp options.

The `*.mdp` file contains the settings that dictate the nature of the simulation. For energy minimization, we will use the simple *steepest descent* minimizer (`integrator = steep` in `em.mdp`, which runs in parallel). Use grompp (the GROMacs PreProcessor) to generate the run input file (TPR) from the run parameter file (MDP), coordinate file (the solvated system with ions; PDB), and the topology (TOP):

```
cd ../emin
gmx grompp -f em.mdp -c ../solvation/ionized.pdb -p ../top/4ake.top -o em.tpr
```

Perform energy minimization

The energy minimization is performed with **`mdrun`** but by using the appropriate `integrator` option in the Run control options in the MDP file it has been instructed to do a energy minimization:

```
gmx mdrun -v -s em.tpr -deffnm em -c em.pdb
```

Ideally, the maximum force *Fmax* (gradient of the potential) should be < 1e+03 kJ mol^{-1} nm^{-2} (but typically anything below 1e+05 kJ mol^{-1} nm^{-2} works). See the screen output or the `em.log` file for this information.

Tip: The final frame of minimization (the structure in `em.pdb`) can be used as the input structure for further minimization runs. It is common to do an initial energy minimization using the efficient steepest descent method and further minimization with a more sophisticated method such as the *conjugate gradient* algorithm (`integrator = cg`) or the Newton-like *Broyden-Fletcher-Goldfarb-Shanno* (`integrator = l-bfgs`) minimizer. For details, see Run control options in the MDP file.

Position-restrained equilibration

We first perform a short MD simulation with harmonic position restraints on the heavy protein atoms. This allows the solvent to equilibrate around the protein without disturbing the protein structure. In addition, we use "weak coupling" temperature and pressure coupling algorithms to obtain the desired temperatue, $T = 300$ K, and pressure, $P = 1$ bar.

Set up and generate the run file

We must first tell Gromacs *how* to perform our equilibration run in the same way that we did for the energy minimization step. This step requires the `top/protein_posres.itp` file with the default value for the harmonic force constants of 1000 kJ mol^{-1} nm^{-2}. The position restraints are switched on by setting the `-DPOSRES` flag in the `posres.mdp` file (see mdp options).

Create the run input (TPR) file, using the energy minimized system as the starting structure:

```
cd ../posres
cp ../templates/posres.mdp .
gmx grompp -f posres.mdp -o posres.tpr -p ../top/4ake.top -c ../emin/em.pdb -maxwarn 2
```

The mdp file contains cut-off settings that approximate the native CHARMM values (in the CHARMM program).

Weak (Berendsen) coupling is used for both temperature and pressure to quickly equilibrate. The protein and the solvent (water and ions) are coupled as separate groups. Gromacs provides a range of groups automatically (run `make_ndx -f TPR` to see them) and we use the groups `Protein` and `non-Protein` (these particularly groups work since roughly Gromacs 4.5.3). If the standard groups do not work then you will have to create the groups yourself using `gmx make_ndx -f TPR -o md.ndx` (which would save them in a file `md.ndx`) and supply it to `gmx grompp -n md.ndx`.

Perform equilibration

Run the position restraints equilibration simulation:

```
gmx mdrun -v -stepout 10 -s posres.tpr -deffnm posres -c posres.pdb
```

(If this is too slow on your workstation, submit to saguaro using 8 cores.)

Attention: Here the runtime of 10 ps is too short for real production use; typically 1 to 5 ns are used.

Generate a centered trajectory in the primary unitcell

In order to visually check your system, first create trajectory with all molecules in the primary unitcell (`-ur compact`; see also below the more extensive notes on trajectory-visualization):

```
echo "System" | gmx trjconv -ur compact -s posres.tpr -f posres.xtc -pbc mol -o␣
↪posres_ur.xtc
```

Note: If you don't have a **`vmd`** command available on the command line then launch VMD, load the `emin/em.pdb` file (*File → New Molecule...*), highlight your molecule 1 ("em.pdb") and load the `posres/posres_ur.xtc` trajectory into your molecule 1", *File → Load Data Into Molecule*. You should see that the first frame (from the energy minimization) looks as if the water is in a distorted box shape whereas all further frames show a roughly spherical unit cell (the rhombic dodecahedron).

Equilibrium molecular dynamics

Set up the production run

As usual, we must tell Gromacs what it will be doing using grompp before we can perform our production simulation. Since we want to start our run where we left off (after doing equilibration), we prepare the TPR input file based on the last frame of the position-restraints with grompp:

```
cd ../MD
cp ../templates/md.mdp .
gmx grompp -f md.mdp -p ../top/4ake.top -c ../posres/posres.pdb -o md.tpr -maxwarn 3
```

The `md.mdp` file uses different algorithms from the position-restraints for the temperature and pressure coupling, which are known to reproduce the exact *NPT* ensemble distribution.

Run the simulation

CPU run

If your workstation has a decent number of cores or if you simply don't mind waiting a bit longer you can also run the simulation as usual with

```
gmx mdrun -v -stepout 10 -s md.tpr -deffnm md -cpi
```

This will automatically utilize all available cores. The `-cpi` flag indicates that you want Gromacs to continue from a previous run. You can kill the job with `CONTROL-C`, look at the output, then continue with exactly the same command line

```
gmx mdrun -v -stepout 10 -s md.tpr -deffnm md -cpi
```

(Try it out!). The `-cpi` flag can be used on the first run without harm. For a continuation to occur, Gromacs needs to find the checkpoint file `md.cpt` and all output files (`md.xtc`, `md.edr`, `md.log`) in the current directory.

GPU run

We can also try utilizing the GPU(s) available on the workstation. We use a modified MDP file that contains settings compatible with GPU-based Gromacs simulations to generate a new TPR file, which is used to perform the simulation:

```
gmx grompp -f mdgpu.mdp -p ../top/4ake.top -c ../posres/posres.pdb -o mdgpu.tpr -
↪maxwarn 3
gmx mdrun -v -stepout 10 -s mdgpu.tpr -deffnm mdgpu -cpi
```

On machines equipped with a well-matched CPU and GPU, a GPU-accelerated Gromacs run can be around 3 to 5 times faster than a CPU-only run on the same machine. Take a look at the Gromacs page on GPU acceleration for more information.

Trajectory visualization

Analysis are normally performed locally on a workstation, i.e. copy back all the files from the supercomputer to your local directory.

A typical analysis tasks reads the trajectory (XTC) or energy (EDR) file, computes quantities, and produces data files that can be plotted or processed further, e.g. using Python scripts. A strength of Gromacs is that it comes with a wide range of tools that each do one particular analysis task well (see the Gromacs manual and the Gromacs documentation).

Keeping the protein in one piece

If you just look at the output trajectory `md.xtc` in VMD then you will see that the protein can be split across the periodic boundaries and that the simulation cell just looks like a distorted prism. You should *recenter* the trajectory so that the protein is at the center, *remap* the water molecules (and ions) to be located in a more convenient unitcell representation.

We will use the trjconv tool in Gromacs to center and remap our system.

Tip: **`trjconv`** prompts the user with a number of questions that depend on the selected options. In the command line snippets below, the user input is directly fed to the standard input of **`trjconv`** with the `printf TEXT | trjconv` "pipe" construct. In order to better understand the command, run it interactively without the pipe construct and manually provide the required information.

Center (`-center`) on the *Protein* and remap all the molecules (`-pbc mol`) of the whole *System*:

```
printf "Protein\nSystem\n" | gmx trjconv -s md.tpr -f md.xtc -center -ur compact -pbc␣
↪mol -o md_center.xtc
```

Pinning down a tumbling protein

It is often desirable to *RMS-fit* the protein on a reference structure (such as the first frame in the trajectory) to remove overall translation and rotation. In Gromacs, the trjconv tool can also do more "trajectory conversion tasks". After (1) centering and remapping the system, we want to (2) RMS-fit (due to technical limitations in **`trjconv`** you cannot do both at the same time).

RMS-fit (`-fit rot+trans`) to the protein *backbone* atoms in the initial frame (supplied in the TPR file) and write out the whole *System*:

```
printf "Backbone\nSystem\n" | gmx trjconv -s md.tpr -f md_center.xtc -fit rot+trans -
↪o md_fit.xtc
```

Check our modified trajectory

Visualize in VMD:

```
vmd ../posres/posres.pdb md_fit.xtc
```

Analysis

For analyzing MD trajectories in many common formats (including the XTC, TRR, TNG, etc. used by Gromacs) using Python, have a look at the MDAnalysis Python library