In-Class Example: Lysozyme and Water

Run these calculations in your /gscratch!

Provided Files:

- submit_gromacs
- plot-EM.py
- plot-NVT-temp.py
- plot-NPT.py
- plot-RMSD.py
- plot-Rgyration.py
- 1AKI_clean.pdb
- ions.mdp
- minim.mdp
- nvt.mdp
- npt.mdp
- md_prod.mdp

1. Open two separate Beartooth shell access terminals through SouthPass, or your local terminal if you prefer. On one terminal, we will use an interactive node to run GROMACS commands for pre- and post-processing data. On the other terminal, we will submit the calculations to the SLURM queue.

2. Navigate to `/project/biocompworkshop/adavies2`. Copy the directory `inclass-example` to your personal gscratch.

```
cp -r inclass-example /gscratch/<your-username>
```

3. Navigate to your gscratch where you've copied the directory.

4. Move the file `submit_gromacs` to your bin.

```
mv submit_gromacs ~/bin
```

5. If you haven't done so already, you'll need to install certain Python packages to plot the data from GROMACs using the provided Python scripts.

```
module load arcc/1.0 gcc/12.2.0 py-pip/22.2.2

pip install pandas
pip install numpy
pip install seaborn
pip install matplotlib
```

6. On the terminal where you will run an interactive node, run the following command.

   ```
   salloc --account=biocompworkshop --time=2:00:00 --nodes=1 --
   ntasks-per-node=1 --partition=teton
   ```

   You should see your node change from ‘blog2’ or ‘blog1’ to a ‘tXXX’ where the XXX are a series of numbers.

7. Run the following commands in the interactive node.

   ```
   module use /project/biocompworkshop/ukapoor/codes/packages

   module load gromacs/2022.5
   ```

8. Run the file 1AKI_clean.pdb through the pdb2gmx command, which will generate three files: a GROMACS formatted structure file, a topology file which contains the force field information for the lysozyme atoms, and a position restraint file which will be used during the energy minimization step.

   ```
   mpirun gmx_mpi pdb2gmx -f 1aki_clean.pdb -o 1aki_processed.gro -
   water spce
   ```

   This will prompt a selection of force fields. For this tutorial, we will use OPLS-AA, option 15.

9. Next, we must generate the box that will contain our simulation and which we’ll apply periodic boundary conditions to.

   ```
   mpirun gmx_mpi editconf -f 1aki_processed.gro -o 1aki_newbox.gro
   -c -d 1.0 -bt cubic
   ```

10. Now that we have generated a box that has our protein centered in the middle, we will solvate the protein with water. The force field that we will use to describe our water-water interactions is the SCP-E force field which we have specified in the previous pdb2gmx command.

    ```
    mpirun gmx_mpi solvate -cp 1aki_newbox.gro -cs spc216.gro -o
    1aki_solv.gro -p topol.top
    ```

11. The lysozyme protein has a +8 charge. Systems don’t exist at a new charge in reality, so we need to add counter ions that will charge neutralize our system.

    ```
    mpirun gmx_mpi grompp -f ions.mdp -c 1aki_solv.gro -p topol.top -
    o ions.tpr

    mpirun gmx_mpi genion -s ions.tpr -o 1aki_solv_ions.gro -p
    ```

```
topol.top -pname NA -nname CL -neutral
```

This last command, genion, will bring another prompt which determines how the ions are added to the system. In this case, we want to replace solvent atoms with ions, so select 13 “SOL”.

12. Now that we have initialized our system, we need to minimize it to ensure that we begin our MD simulation with a ground-state, or local minima, energy configuration. The following command will generate a binary file, em.tpr, that contains the topology and atomic coordinate information.

    ```
    mpirun gmx_mpi grompp -f minim.mdp -c 1aki_solv_ions.gro -p
    topol.top -o em.tpr
    ```

13. In your separate terminal, which is **not** running an interactive node, run the following command, which will use the submit_gromacs script to submit your energy minimization calculation to the SLURM queue. We are going to use 2 tasks per node and 2 CPUs per task on a GPU for this calculation. It should take ~ 30 seconds to run.

    ```
    submit_gromacs 2 2 em
    ```

14. Once the job is done, we are going to analyze the potential energy from each step of the previous calculation to confirm that we have minimized the system. Return to the terminal with the interactive node.

    ```
    mpirun gmx_mpi energy -f em.edr -o potential.xvg
    ```

    This command will also prompt a response for what data you would like extracted. We will look at the potential energy. Type “10 0” and hit enter.

15. Staying in the interactive node, plot the data using plot-EM.py.

    ```
    python3 plot-EM.py
    ```

    We can look at this image via SouthPass.

16. Now that we have minimized the structure, we will equilibrate the system. First, we want to equilibrate temperature. For this, we will use an NVT ensemble. We will use 2 tasks and 2 CPUs per task with a GPU. This calculation should take about 1 minute. Remember, run the first command in your interactive node and the second command in your secondary terminal **without** the interactive node.

    ```
    mpirun gmx_mpi grompp -f nvt.mdp -c em.gro -r em.gro -p topol.top
    -o nvt.tpr
    ```

```
submit_gromacs 2 2 nvt
```

17. This time, we will analyze the temperature from the output of the NVT run.

    ```
    mpirun gmx_mpi energy -f nvt.edr -o temperature.xvg
    ```

    When prompted, select “16 0”.

    ```
    python3 plot-NVT-temp.py
    ```

    Use SouthPass to look at the plot produced from this command.

18. Now that we’ve equilibrated the temperature, we need to equilibrate the density by way of equilibrating the volume. This will be done with an NPT ensemble. This calculation should take about a minute as well.

    ```
    mpirun gmx_mpi grompp -f npt.mdp -c nvt.gro -r nvt.gro -t nvt.cpt
    -p topol.top -o npt.tpr
    ```

    ```
    submit_gromacs 2 2 npt
    ```

    From the NPT run, we will analyze two things; the pressure and the density of our system over time. The plot-NPT.py script will generate two plots. Again, look at these plots on SouthPass and determine if the properties have equilibrated.

    ```
    mpirun gmx_mpi energy -f npt.edr -o pressure.xvg
    “18 0”

    mpirun gmx_mpi energy -f npt.edr -o density.xvg
    “24 0”

    mpirun gmx_mpi energy -f npt.edr -o volume.xvg
    “23 0”

    python3 plot-NPT.py
    ```

    Why are there large pressure fluctuations? Compare these fluctuations to the fluctuations in volume.

19. Now that our system is minimized and equilibrated, we can run a production run. This will be the run that we will analyze our final properties from. In this simple example, we will look at the root mean squared displacement (RMSD) and the radius of gyration for the protein.

    ```
    mpirun gmx_mpi grompp -f md_prod.mdp -c npt.gro -t npt.cpt -p
    topol.top -o md_prod.tpr
    ```

For this calculation, since it is much longer than the other two (10 ps for NPT, 20 ps for NPT, and 1 ns for the production run), we will use a few more resources to speed up the calculation. We will not use 2 tasks and 5 CPUs per task on the GPU. This should take about 5-6 minutes to complete.

```
submit_gromacs 2 5 md_prod
```

20. To analyze the root mean squared displacement, run the following commands.

```
mpirun gmx_mpi trjconv -s md_prod.tpr -f md_prod.xtc -o
md_prod_noPBC.xtc -pbc mol -center
“1 0”
```

```
mpirun gmx_mpi rms -s md_prod.tpr -f md_prod_noPBC.xtc -o
rmsd.xvg -tu ns
“4 4”
```

```
python3 plot-RMSD.py
```

Again, analyze the image produced.

21. Finally, we will look at the radius of gyration to determine the stability of our protein throughout the simulation.

```
mpirun gmx_mpi gyrate -s md_prod.tpr -f md_prod_noPBC.xtc -o
gyrate.xvg
“1”
```

```
python3 plot-Rgyration.py
```