Assignment 3: Introduction to Molecular Simulation using GROMACS

Structural Bioinformatics II (CHEM 5412) Spring 2026
Vincent Voelz
March 14, 2026

Learning Goals

Students will learn how to prepare and equilibrate an all-atom molecular simulation of a peptide using the software package GROMACS, and interpret the results.

Assignment

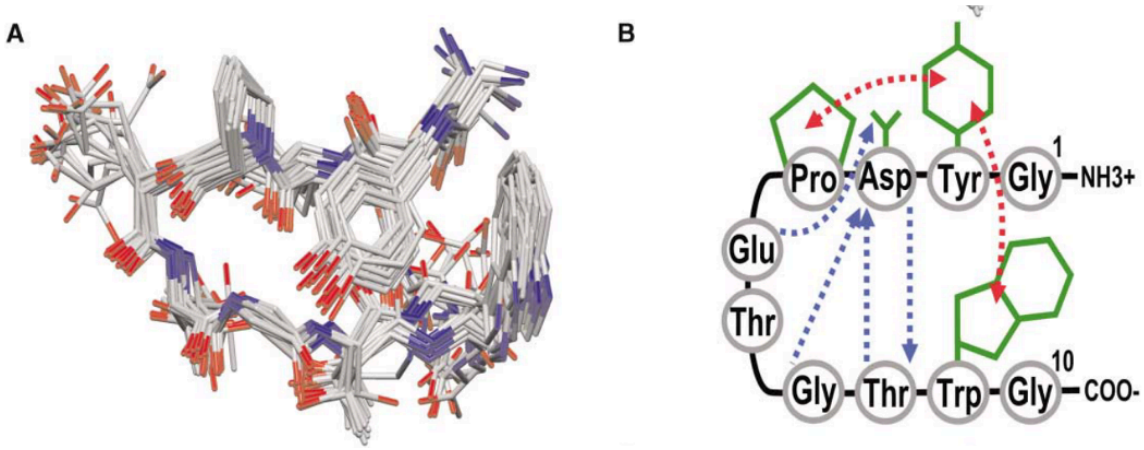
For this assignment, we will perform a short molecular simulation of the beta-hairpin miniprotein *chignolin*.

We will:

1. Prepare the simulation by solvating the protein and adding counterions
2. Minimize the energy of the system, and investigate the results
3. Equilibrate the system, first at constant *temperature* and then at constant *pressure*, and investigate.
4. Visualize the simulated trajectories and report on their molecular motions.

Introduction: The Chignolin Mini-protein

Chignolin is a synthetic 10–residue peptide (GYDPETGTWG) designed to form a stable β-hairpin structure. Because it is extremely small yet still exhibits cooperative folding behavior, it has become a widely used model system for studying protein folding and molecular simulation methods.



Figures taken from Honda et al. (2004). (A) The NMR structural ensemble in PDB:1UAO. (B) Key hydrogen bonding (blue) and hydrophobic (red) interactions observed between residues.

Chignolin was originally designed by Honda, Yamasaki, and coworkers and characterized experimentally using NMR spectroscopy. Despite containing only ten amino acids, it forms a well-defined folded state stabilized by hydrophobic packing and backbone hydrogen bonds. The small size of chignolin makes it ideal for testing simulation methods, benchmarking force fields, and exploring folding dynamics.

In this exercise, we will simulate the NMR structure of chignolin (PDB ID 1UAO) using the molecular dynamics program GROMACS.

Key references

1. Honda, Shinya, Kazuhiko Yamasaki, Yoshito Sawada, and Hisayuki Morii. “10 Residue Folded Peptide Designed by Segment Statistics.” *Structure* 12, no. 8 (2004): 1507–18. https://doi.org/10.1016/j.str.2004.05.022.
2. Honda, Shinya, Toshihiko Akiba, Yusuke S. Kato, et al. “Crystal Structure of a Ten-Amino Acid Protein.” *Journal of the American Chemical Society* 130, no. 46 (2008): 15327–31. https://doi.org/10.1021/ja8030533.
3. Lindorff-Larsen, K., S. Piana, R. O. Dror, and D. E. Shaw. “How Fast-Folding Proteins Fold.” Science 334, no. 6055 (2011): 517–20. https://doi.org/10.1126/science.1208351.

Overview of GROMACS

GROMACS (GROningen MAchine for Chemical Simulations) is a widely used open-source software package for performing **molecular dynamics (MD) simulations** of biomolecules.

At its core, GROMACS performs simulations by:

1. **Defining a molecular topology**
    - atoms, bonds, charges, and force-field parameters
2. **Computing forces using a molecular mechanics force field**
    - bonded interactions (bonds, angles, torsions)
    - nonbonded interactions (electrostatics and van der Waals)
3. **Integrating Newton's equations of motion**
    - Numerical integration of $F = ma$ to propagate the system forward in time.

A typical MD workflow involves:

1. Preparing a topology and simulation box
2. Adding solvent and ions
3. Energy minimization
4. Temperature equilibration (NVT)
5. Pressure equilibration (NPT)
6. Production simulation

In this exercise we will perform the first **five steps** of this process.

Useful links

- https://www.gromacs.org
- GROMACS 2026.1 documentation
- User guide - GROMACS 2026.1 documentation

Procedure

For the following steps, please refer to the Colab notebook that can be found on the course GitHub: https://github.com/vvoelz/chem5412-spring2026 .

There should be a link on the main README.md page (and in `/notebooks`) to open the Assignment 3 notebook in Google Colab. Or you can use this link: https://colab.research.google.com/github/vvoelz/chem5412-spring2026/blob/main/notebooks/04_molecular_simulation.ipynb

Step 1 — Generate the Molecular Topology

The first step is to convert the PDB structure into a **GROMACS topology**. I have already sliced out only the first MODEL of the NMR structure, and saved it as `1UAO_model1.pdb`

The command `pdb2gmx`:

- assigns atom types and charges
- adds missing hydrogen atoms (in this case, since 1UAO is an NMD structure it has hydrogens, but we will ignore them and re-add them)
- builds the molecular topology
- generates a coordinate file in **GROMACS `.gro` format**

We will use the **Amber99SB-ILDN** protein force field with **TIP3P water**.

```
$ gmx pdb2gmx -f 1UAO_model1.pdb -p topol.top -o chignolin.gro -ff
amber99sb-ildn -ignh -heavyh -water tip3p
```

Important outputs:

- `topol.top` → system topology
- `chignolin.gro` → coordinates in GROMACS format

The `*.gro` file format stores atomic coordinates

The `.gro` file stores **atomic coordinates and velocities** in a format native to GROMACS.

- Similar in purpose to a PDB file but without all the metadata.
- Contains atom names, residue names, and positions
- Often used as **input coordinates for the next stage of a simulation**

Some key points to remember:

1. The first line of the `*.gro` is the **title** (here, some colorful phrase chosen randomly by GROMACS)
2. The second line of the `*.gro` is the number of atoms

```
$ head -6 chignolin.gro
S  C  A  M  O  R  G
  138
    1GLY      N    1  -0.678  -0.142   0.420
    1GLY     H1    2  -0.764  -0.134   0.470
    1GLY     H2    3  -0.658  -0.239   0.404
    1GLY     H3    4  -0.604  -0.102   0.474

$ tail -5 chignolin.gro
   10GLY    HA2  135  -0.025  -0.392   0.431
   10GLY      C  136  -0.188  -0.503   0.501
```

```
   10GLY    OC1  137  -0.295  -0.491   0.558
   10GLY    OC2  138  -0.129  -0.609   0.488
   1.39835   1.24466   1.06869
```

The `*.top` file stores the molecular topology

The topology describes the **chemical identity and force field parameters** of the system.

It includes:

- atom types
- charges
- bonded interactions (bonds, angles, dihedrals)
- references to force-field parameter files
- number of solvent molecules and ions

In this exercise the topology is stored in `topol.top`

Step 2 — Define the Simulation Box

Next we place the peptide into a **periodic simulation box**.

Periodic boundary conditions allow us to simulate a small box that represents an infinite bulk system.

The command `editconf`:

- defines the box shape
- sets the minimum distance between the protein and the box edges.

Here we create a **cubic box** with a **1 nm margin** around the peptide.

```
$ gmx editconf -f chignolin.gro -bt cubic -d 1.0 -o chignolin_box.gro
```

Step 3 — Add Explicit Water

We now fill the simulation box with **explicit solvent molecules**.

The `solvate` command:

- adds TIP3P water molecules
- updates the topology file automatically.

```
$ gmx solvate -cp chignolin_box.gro -p topol.top -o chignolin_solvated.gro
```

Step 4 — Add Counterions

Real biological systems contain dissolved salts. Human physiological salt concentrations in blood and cytosol are about 150 mM. Many of the experiments in the Honda et al. 2004 paper used phosphate buffer (to control the pH) and 100 mM NaCl. So, we will plan to add counterions to approximate a 100 mM concentrations

We also need to add counterions to **neutralize** the net charge of the system. (Periodic simulation of a box with a net charge can be dangerously inaccurate!) Honda et al (2004) performed experiments at pH 5.5. At this pH, the protonation state of the peptide gives it a net charge of –2 (because Asp (D) and Glu (E) have deprotonated carboxylate groups).

```
                                 - -
   Free amino at N-term -->  +GYDPETGTWG-  <-- free carboxylate at C-term
```

So, we should expect to add two more cations (Na+) than anions (Cl-).

To add **Na^+ and Cl^- ions** to neutralize the system and create a salt concentration of **0.1 M**, we first we generate a temporary run input file:

```
$ gmx grompp -f mdp/em.mdp -c chignolin_solvated.gro -r
chignolin_solvated.gro -p topol.top -o em.tpr -v -maxwarn 2
```

Then we replace some water molecules with ions:

```
$ gmx genion -s em.tpr -p topol.top -o chignolin_solvated_ions.gro -conc
0.100 -neutral
```

When prompted, choose the **SOL** group so that ions replace water molecules.

Step 5 — Energy Minimization

Before running dynamics, we must remove steric clashes and relax the system. Energy minimization finds a nearby **local minimum of the potential energy surface**.

For this and future steps, will we utilize the `grompp` GROMACS pre-processor, followed by the `mdrun` command. The parameters specifying the calculations to perform are given in the `*.mdp` file:

```
$ cat mdp/em.mdp
; Parameters describing what to do, when to stop and what to save
integrator      = steep     ; Algorithm (steep = steepest descent minimization)
nsteps          = 10000       ; Maximum number of (minimization) steps to perform
nstxout         = 50
emtol           = 100.0

; Parameters describing how to find the neighbors of each atom and how to calculate
the interactions
nstlist         = 1             ; Frequency to update the neighbour list and long
range forces
cutoff-scheme   = Verlet
rlist           = 1.2           ; Cut-off for making neighbour list (short range
forces)
coulombtype     = PME           ; Treatment of long range electrostatic interactions
rcoulomb        = 1.2           ; long range electrostatic cut-off
vdw-type        = cutoff
vdw-modifier    = force-switch
rvdw-switch     = 1.0
rvdw            = 1.2           ; long range Van der Waals cut-off
pbc             = xyz           ; Periodic Boundary Conditions
DispCorr        = no
```

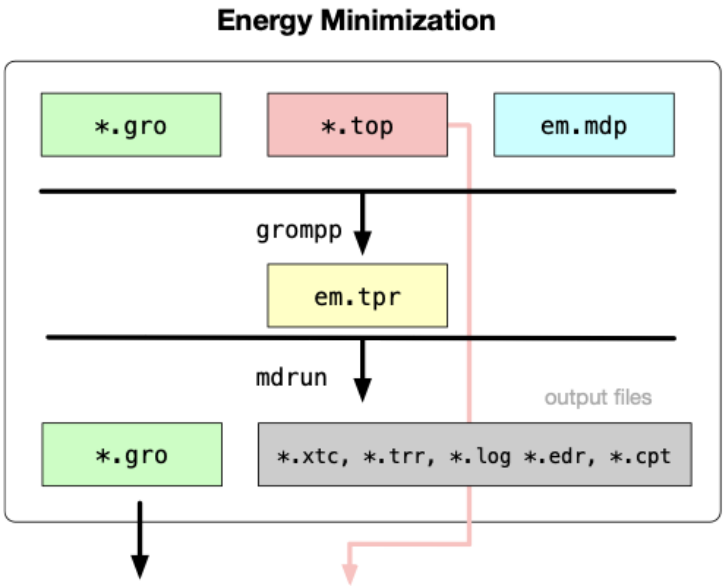
The `.tpr` is a portable run input file

Together with the structure (`.gro`) and topology file (`.top`), `grompp` outputs a binary file called a `*.tpr` file that is the input to the integration engine `mdrun`. It contains everything needed to run a simulation.

It bundles together:

- the molecular topology
- coordinates
- simulation parameters from the `.mdp` file
- constraints and integrator settings

Because it contains all required information, the `.tpr` file is the **primary input for `gmx mdrun`**.



The output files from `gmx mdrun` can then be used for future equilibration steps.

```
$ gmx grompp -f mdp/em.mdp -c chignolin_solvated_ions.gro -r
chignolin_solvated_ions.gro -p topol.top -o em.tpr -v -maxwarn 2

$ gmx mdrun -v -s em.tpr -c em.gro -e em.edr
```

Outputs:

- `em.gro` → minimized structure
- `em.edr` → energy file containing simulation energies

The `.edr` file stores thermodynamic quantities recorded during the simulation, including energies

Typical values include:

- potential energy
- kinetic energy
- temperature
- pressure
- box volume

These values are extracted using `gmx energy`.

Plotting the Energy Minimization

Information in the GROMACS energy file (`.edr`) is converted to a text file using `gmx energy`.

```
$ echo "Potential\n0" | gmx energy -f em.edr -o potential.xvg
```

The `*.xvg` file is human readable, and there is code in the Colab notebook (also in `plot_energy_minimization.py`) to plot the potential energy as a function of minimization step.

Step 6 — Temperature Equilibration (NVT)

Next we equilibrate the system at a constant **temperature (300 K)** while keeping the **volume fixed**.

This ensemble is called **NVT** (constant number of particles, volume, and temperature).

```
$ gmx grompp -f mdp/nvt.mdp -c em.gro -r em.gro -p topol.top -o nvt.tpr -v -maxwarn 2
$ gmx mdrun -v -s nvt.tpr -c nvt.gro -g nvt.log -e nvt.edr -o nvt.trr
```

`.trr` and `.xtc` — Trajectory files

Trajectory files store the **time evolution of the atomic coordinates**.. Two common formats exist:

`.trr`

- full precision trajectory
- includes coordinates, velocities, and forces
- larger file siz

`.xtc`

- compressed trajectory
- coordinates only
- much smaller files

In this exercise we output `nvt.trr` and `nvt.trr`. These contain the motion of all atoms during equilibration.

Step 7 — Pressure Equilibration (NPT)

Finally we equilibrate the system at constant **temperature and pressure**.

This ensemble allows the box size to fluctuate until the correct **density** is reached.

```
$ gmx grompp -f mdp/npt.mdp -c nvt.gro -r nvt.gro -p topol.top -o npt.tpr -v -maxwarn
$ gmx mdrun -v -s npt.tpr -c npt.gro -g npt.log -e npt.edr -o npt.trr
```

Plotting Pressure and Volume over time

Convert the energy file:

```
$ echo "Pressure\n0" | gmx energy -f npt.edr -o pressure.xvg
$ echo "Volume\n0" | gmx energy -f npt.edr -o volume.xvg
```

There is code to plot this data in the Colab notebook (also in `plot_pressure_volume_equil.py`)

Visualizing the Trajectory Data

To better understand the dynamics of the peptide, we can visualize the trajectory. There are two recommend ways:

1. Visualizing the Trajectory Data in Colab with py3Dmol
2. Visualizing the Trajectory Data with VMD (Visual Molecular Dynamics)

Visualizing the Trajectory Data in Colab with py3Dmol

To better understand the dynamics of the peptide, we can visualize the trajectory directly inside the notebook.

Follow the directions in the Colab notebook to display the **NPT-equilibrated trajectory** of chignolin.

Visualizing the Trajectory Data in Colab with VMD

VMD (Visual Molecular Dynamics) is a free molecular visualization tool that is optimized for fast rendering of molecular simulation trajectory data. You can download it as an app on your personal computer here:

- https://www.ks.uiuc.edu/Research/vmd/

To load the trajectory:

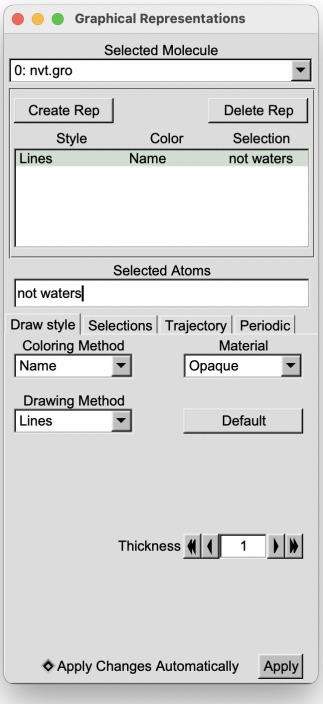
1. First load a the `*.gro` structure, using **File** > **New Molecule …** Choose the output of the NVT step, `nvt.gro` to load in.
2. Then, in the same menu, make sure you have “Load files for ..” with your `nvt.gro` structure selected, and load the `npt.trr` trajectory (the NPT trajectory)

NOTE: You can also load in `*.xtc` files as trajectories. The main concern is that the number of atoms in the `.gro` file needs to match the number of atoms in `*.xtc` file, which typically is only the protein coordinates. In case, you can make a protein-only `*.gro` file using `gmx editconf`.

Once the trajectory is loaded, you can play the movie with the controls at the bottom the window.

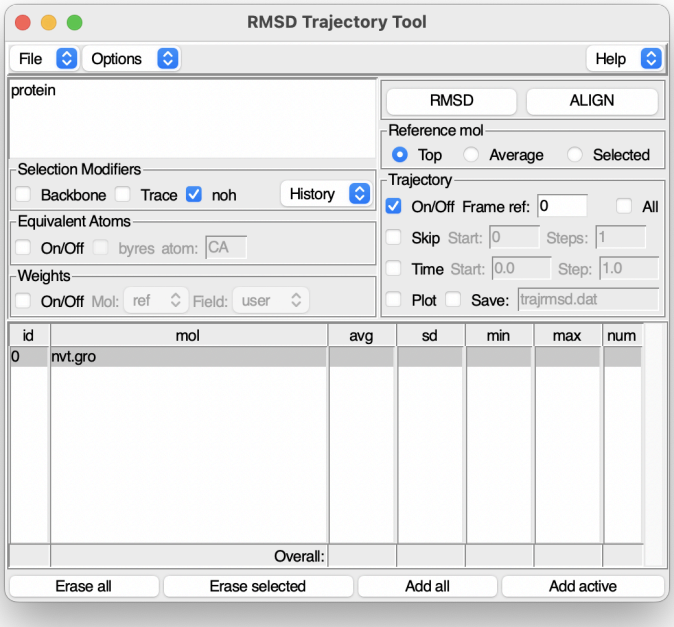
Some VMD Tips and Tricks:

- The protein may be hard to see with all the explicit water. To hide the waters, select the menu **Graphics** > **Representations …** and in the “Selected Atoms” area type `not waters` and Click Apply.

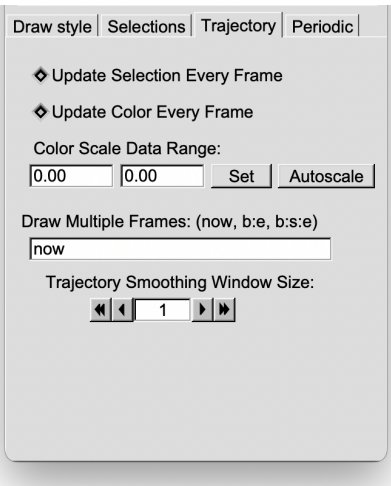


- The chignolin peptide will likely be diffusing around through the simulation box, since we have not constrained its motion. You can RMSD-align every frame with respect to the

first frame using **Extensions > Analysis > RMSD Trajectory Tool**. Make sure “protein” is the atom selection in the window, and click “Align” in the top right



- For a smoother movie, you can visualize the trajectory with sliding-window frame averaging. Choose **Graphics** > **Representations …** then click on the “Trajectory” tab on the bottom half of the window. At the bottom, select a “Trajectory Smoothing Window Size” of 1 frame or more:



There are many other tips and tricks to visualize trajectory data. Please see the VMD documentation and Tutorials here:
https://www.ks.uiuc.edu/Research/vmd/current/docs.html#tutorials

Questions to discuss in your report

1. After completing the system preparation (after Step 4), look at the bottom of your `topol.top` file (in the Colab notebook, use `` `! tail topol.top` ``) `chignolin_solvated_ions.gro`
    a. How many water molecules are in your simulation system?
    b. How many Na+ ions and how many Cl- ions have been added to the system?

2. Look at the bottom of your `chignolin_solvated_ions.gro` file (in the Colab notebook, use `` `! tail chignolin_solvated_ions.gro` ``) In the last line – there should be three numbers; let's call them L_1, L_2 and L_3) These are the box lengths in units of nm = 10^{-9} m. (The three numbers should be identical since it's a cubic PBC box, i,e. $L_1 = L_2 = L_3$)
    a. Use the number of Na+ ions from 1b (let's call it N) and the volume of the box ($V = L_1 \times L_2 \times L_3$) to calculate the concentration of ions in molar units (M = mol/L). **Is the calculated concentration it close to our desired concentration of 100 mM = 0.1 M?**

       Hint: Convert N to units moles by multiplying by $1/(6.022 \times 10^{23}\ \text{mol}^{-1})$, and convert the volume to units liters (L) using the identity 1 L = 1 dm^3, in other words:

       $V = (L_1 \times L_2 \times L_3\ \text{nm}^3)\ (1\ \text{m}^3 / 10^{-27}\ \text{nm}^3)\ (1\ \text{L} / 0.001\ \text{m}^3)$

3. Using the plotting code provided after Step 7, make plots of the pressure over time, and the volume over time. Include this in your report.

    a. After inspecting these plots, which do you think is better for monitoring convergence of the simulations – the instantaneous ***pressure*** or the ***volume*** ? Why?

4. Visualize the NPT trajectory using one of the methods above (Py3Dmol or VMD), and include a screenshot.
    a. Although this is a very short trajectory (200 ps), you should be able to observe some bond rotations. Describe some specific motions you observe.