2/13/2020 Week 5 Module 2
Interactive Exercise for Molecular Dynamics

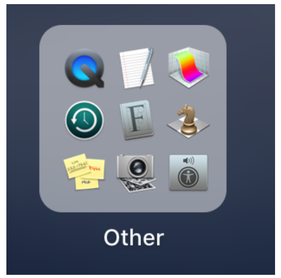
- In this module, you will learn how to
  - navigate the UNIX terminal
  - execute programs with a command line interface (CLI)
  - run a short MD simulation on your desktop machine
  - visualize the simulation with VMD
- In this class session you will also work on preparing your team's target for MD simulation

The UNIX Terminal and Command Line Interface (CLI)

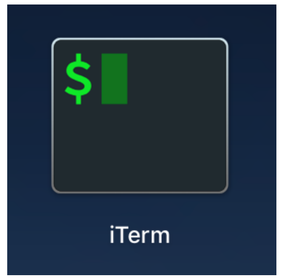
The UNIX Terminal

- Predecessor and very similar to the terminal used in
  - LINUX - many scientific computing clusters
  - Mac OS X - this lab and many of your own laptops/computers
- Useful for
  - accessing programs that run with a command line interface (CLI)
  - automation

- To start the terminal on a Mac:



- I actually usually use another program:



UNIX Terminal Essentials

- Try the following commands
  - echo $SHELL - echo reports the value of a variable. $SHELL is the terminal interface you are using, and will affect the details of all other commands. bash is a popular shell.
  - File operations
    - ls – list files and directories
    - cp – copy files
    - rm – remove files and directories
    - mv – rename or move files and directories to another location
  - Directory management
    - pwd – confirm current directory
    - cd – change directory
    - mkdir – make new directory
    - rmdir – remove directory
- Also see
  - https://www.unixtutorial.org/basic-unix-commands
  - An Introduction to Linux (https://www.youtube.com/watch?v=IVquJh3DXUA). Work with the terminal starts at 3:35

Command Line Interface (CLI)

- Computer software is usually either accessible by a
  - graphical user interface (GUI) or a
  - command line interface (CLI)
- A lot of scientific software is based on a CLI
  - it takes effort to create a GUI
  - a CLI is easier to automate
- Starting a program on the CLI looks like
  - path_to_program/program_name required_argument1 --argument1_name argument1_value
    - path_to_program can be ./ for the local directory or omitted if the program is in the $PATH variable
    - the number of required and optional arguments depends on the program

Running OpenMM

- To actually run OpenMM, you need to invoke a Python script.
- To invoke the script, enter the following commands
  - `cd ~/Desktop/OpenMM', which changes your current directory to the directory where the files are
  - 'ls' lists the contents of the current directory. It isn't 100% necessary but useful to check.
  - `python MD_ubq.py' actually runs the script

```
(openmm) Minh-IIT-MBP2018:[~]: cd ~/Desktop/OpenMM
(openmm) Minh-IIT-MBP2018:[~/Desktop/OpenMM]: ls
1ubq.pdb   MD_ubq.py
(openmm) Minh-IIT-MBP2018:[~/Desktop/OpenMM]: python MD_ubq.py
```

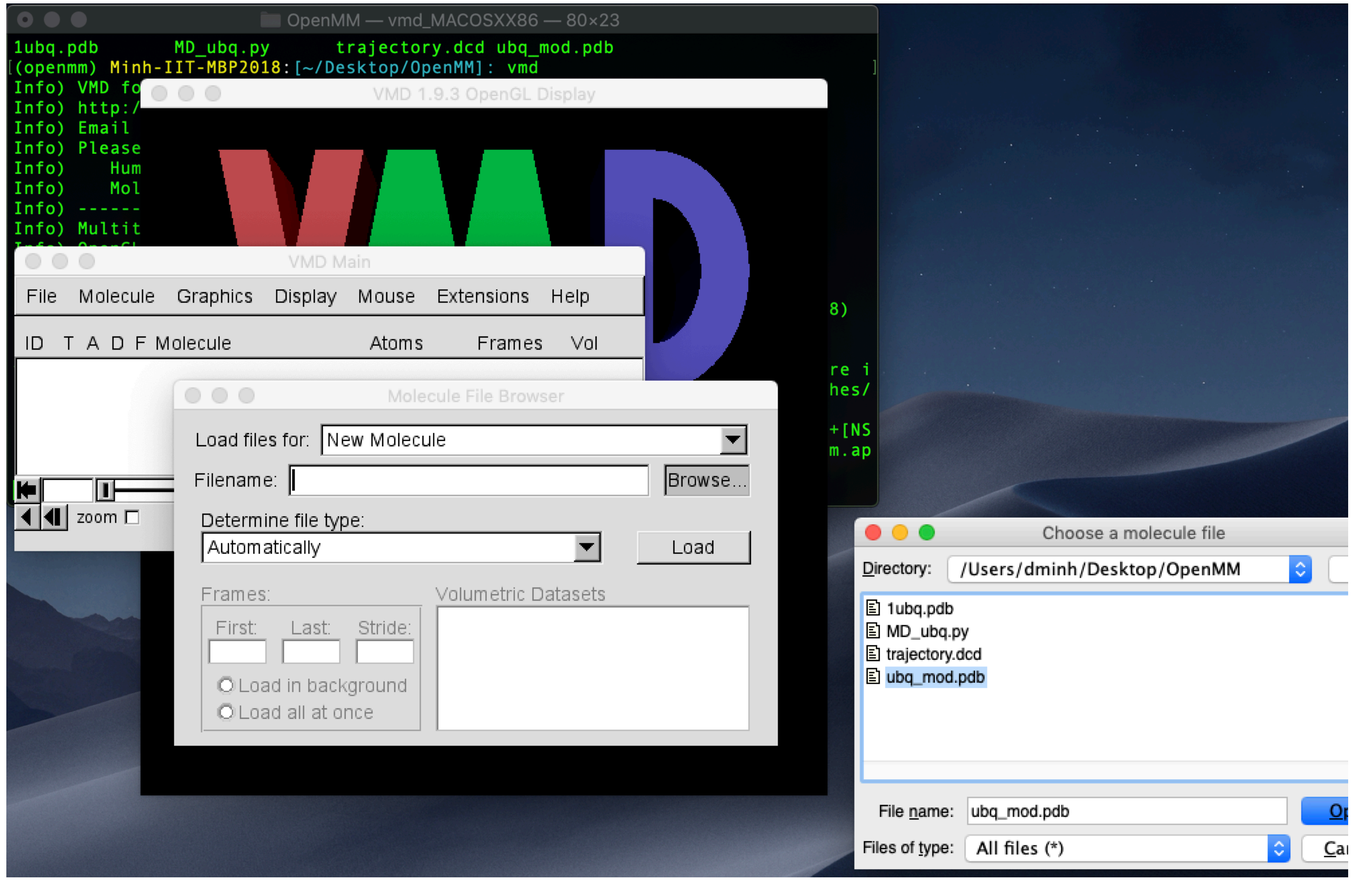
Input files can also be downloaded from github:
https://github.com/daveminh/Chem456/tree/master/static_files/tutorials/ubq-md

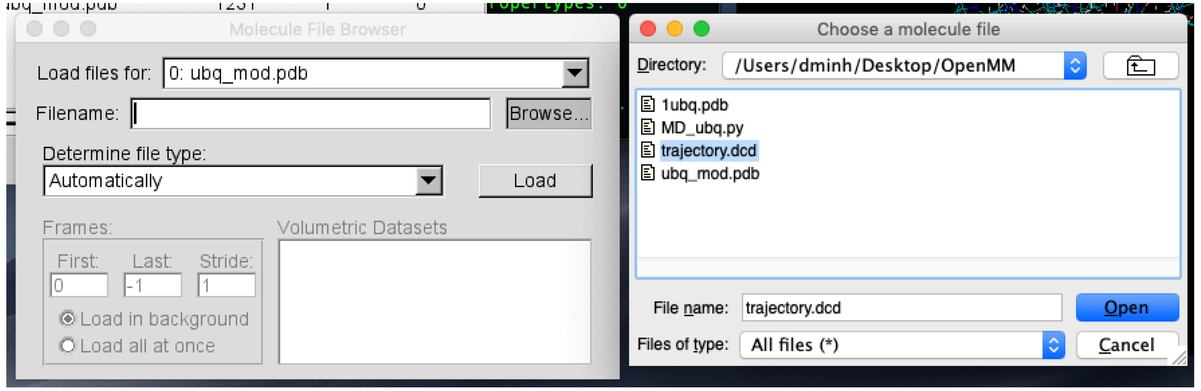
- When you run the script, you will see some output to the console. This is mostly from the StateData reporter
  - Notice that the potential energy goes slightly up. This is because we started with a minimized structure.
- If you type `ls' again, you will see that the simulation created two files
  - ubq_mod.pdb - the model that includes hydrogen but not water
  - trajectory.dcd - the actual simulation data

```
(openmm) Minh-IIT-MBP2018:[~/Desktop/OpenMM]: python MD_ubq.py
Minimizing...
Running Production...
#"Progress (%)" "Step"  "Time (ps)"     "Potential Energy (kJ/mole)"    "Temperature (K)"       "Speed (ns/day)"
Time Remaining"
10.0%   100     0.20000000000000015     -12936.42476230517      183.51751922475094      0       --
20.0%   200     0.4000000000000003      -12646.180945641867     208.17988992042834      66.3    0:02
30.0%   300     0.6000000000000004      -12391.382319417527     222.92731255729387      63.9    0:01
40.0%   400     0.8000000000000006      -12397.931712869551     252.91336736415292      63.2    0:01
50.0%   500     1.0000000000000007      -12164.42101471391      261.4505010540196       63.7    0:01
60.0%   600     1.2000000000000008      -11959.584803213324     267.99278996281504      63.9    0:01
70.0%   700     1.400000000000001       -11810.354813818332     278.89717139432463      64.1    0:00
80.0%   800     1.6000000000000012      -11819.44354535209      280.87644275577924      64.5    0:00
90.0%   900     1.8000000000000014      -11761.649241585736     281.12817290253514      64.6    0:00
100.0%  1000    2.0000000000000013      -11685.676062540104     281.4727076092732       64.9    0:00
Done!
(openmm) Minh-IIT-MBP2018:[~/Desktop/OpenMM]: ls
1ubq.pdb        MD_ubq.py       trajectory.dcd ubq_mod.pdb
(openmm) Minh-IIT-MBP2018:[~/Desktop/OpenMM]:
```

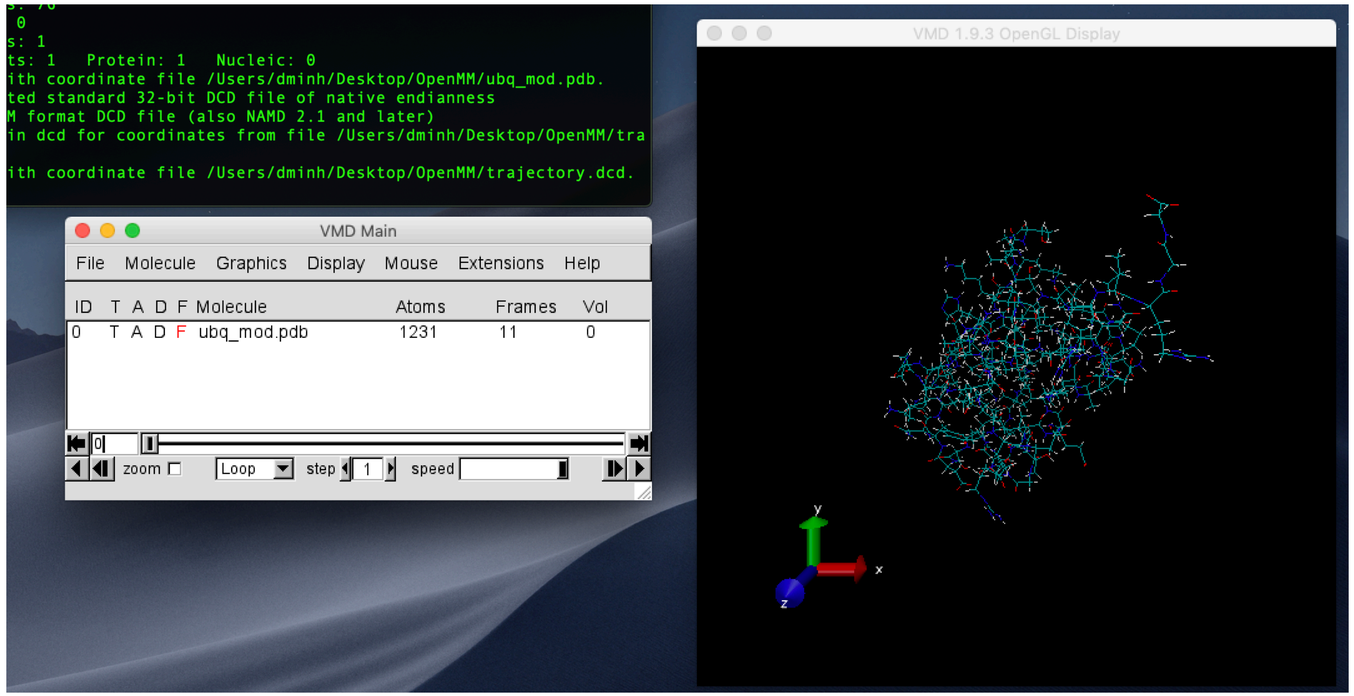
Visualizing Results

- To visualize the molecular dynamics trajectory
  - open vmd
  - create a new molecule based on “ubq_mod.pdb”
  - load files “trajectory.dcd” as a file for the new molecule





- Now look at the “VMD Main” window
  - “Frames” reads 11 instead of 1
  - You can use the bottom of the window to start a movie and control where we are
- There probably won’t be much going on in your movie, but all of molecular dynamics is just like this, just rinse and repeat



Some complications

- Most simulations are not going to be this easy to set up
  - Crystal structures may have missing atoms, which often can be added by homology modeling
  - The added hydrogens should depend on the pH you're trying to model
  - You need to do extra work for non-standard residues, cofactors, and ligands
  - Most of the time it is better to simulate with water
  - For certain systems, you will need to build a membrane
- You usually need much longer simulations to get publishable results
  - GPU computing provides significant acceleration
  - Calculations will usually be done remotely
    - XSEDE supercomputer
    - Google/Amazon cloud
- We will help each group with all of these