Hyia, just thought I’d write the pipeline so far up a bit. Especially towards the end, things might be somewhat incomplete. When you walk into bugs or things don’t work please do try fixing it yourself, but really *do not* spend *too* much time on them. Just come ask me if I came across them, and I probably know the easy fix. There is some merit to being able to find things out yourself, but not to the point that you’re stuck for hours on end. Stackoverflow and Google also have a lot of very useful resources.

**How to get an MD simulation to run:**

*Files in attachments:*

* tleapOrders
* ParmEd\_Amb2Gmx.py
* Minim.mdp
* Nvt.mdp
* Npt.mdp
* Md.mdp
* Production.sh
* Production\_DB.sh

*Setting up Linux using WSL*

In order to run an MD simulation it is recommended to install a Linux subsystem on your laptop. Most softwares/packages are developed with the Linux OS in mind, so work best there. Sometimes you come across a programme which works easier on Windows. With WSL you have the power to choose the OS in which you want to install a programme. Usually computational related programmes (GROMACS, prody, python packages) work easier on Linux and visual programmes (such as VMD or PyMOL) work easier on windows. Instructions on how to install WSL can be found here: <https://learn.microsoft.com/en-us/windows/wsl/install>

With the Linux OS you will interact with your laptop using the command-line. There are many resources that teach basic command-line skills, F.e. <https://swcarpentry.github.io/shell-novice/index.html>. If you need something specific ask me or ask Google. This is something you learn best by doing!

*Installing Ambertools23, ParmEd, GROMACS*

We advise to install the programmes using anaconda environments. Some people are also a big fan of pip/python virtual environments, but these are not as widely applicable to different softwares as anaconda. Once you have your Linux subsystem you can install miniconda3 on it (or it is already installed, not sure).

To run MD we have to parameterize the atoms of the protein we are interested in. The chemical identity of each atom depends on its environment. Force fields are developed parameterizations for specific biomolecules such as proteins, nucleic acids, lipids, and so on. We use Amber to parameterize our molecules since it has quite some of these force fields incorporated. I used Ambertools23 specifically, but they release a new one each year (so now Ambertools24 is available). Their documentation is nicely written, so you could read the first chapter to get a sense of the thing: <https://ambermd.org/doc12/Amber24.pdf>. There is much information on force fields and their combinations. However we generate the actual simulations in GROMACS (<https://manual.gromacs.org/2024.3/index.html>). We talked about a couple of reasons, but I realize the biggest one is that parallelization and GPU-acceleration happens under-the-hood, making your life a lot easier. To convert Amber files to GROMACS format we use ParmEd (<https://github.com/ParmEd/ParmEd>), which is a python library.

So what you need is properly functioning versions of Ambertools/Amber, ParmEd and GROMACS.

*Parameterization in Ambertools23*

Once you have your protein PDB you need to decide on which force fields to use. Use the Amber documentation and/or Google as inspiration. If you don’t know, or can’t find something let me/Nina know. Then you write a file for *tleap*, the internal programme that actually does the parameterization. An example is found as *tleapOrders* in the attachments. Here you specify the force fields to load, load in the PDB, solvate in a box of water, add ions and save/generate a topology (.top) and coordinate (.crd) file.

It might be that the parameterization is not succesful. 9 out of 10 times these are a few unrecognized H-atoms, which you can remove from the pdb, and retry. Amber will automatically fill in the missing H-atoms. If the errors are a bit more elaborate try Googling/solving them yourself first. If you don’t manage let me/Nina know and we’ll help you. Please don’t get stuck for hours on end before asking us, ask us sooner. But also don’t ask everything without having Googled a bit yourself first.

*Conversion in ParmEd*

Then you run a python file *ParmEd\_Amb2Gmx.py* to convert the topology and coordinate files to GROMACS format (.top and .gro). These files are what you use for subsequent minimization and equilibration.

*Minimization, equilibration, production in GROMACS*

To minimize, equilibrate and produce the systems you use the following scripts *minim.mdp, nvt.mdp, npt.mdp, md.mdp*. Coupled with the topology and structure file that you generated you can now have GROMACS perform the calculations. I created two bashscripts that show the lines needed (one for local use and one for Delftblue use) 🡪 *production.sh* and *production\_DB.sh*. You could also run the lines in a shell script one by one on your command line (scripting is very nice though, at some point I can explain how to). Most of these files I got from <http://www.mdtutorials.com/gmx/lysozyme/index.html> a tutorial that might be nice to go through! But not if you’re already getting your system to run properly. More GROMACS tutorials if you’re feeling like it: <http://www.mdtutorials.com/gmx/>

*Post-processing and analysis:*

We haven’t discussed these yet, lets do so when the time comes.