Day 1: Preparation of helix topology

This tutorial will guide the user through the process of setting up and running pulling simulations necessary to calculate the free energy of the unfolding process for small sequences in different secondary structure elements (i.e alpha helix and hairpin).

The unfolding energy ($\triangle G_{unfold}$) is derived from the potential of mean force (PMF), extracted from a series of umbrella sampling simulations. A series of initial configurations have to be generated, each corresponding to a location wherein the molecule of interest is harmonically restrained at an increasing center-of-mass (COM) distance from a reference point using an umbrella biasing potential. This restraint allows the molecule to sample the configurational space in a defined region along a reaction coordinate between the initial and final state. The windows must allow for slight overlap of the COM positions for proper reconstruction of the PMF curve.

The steps for such a procedure (and the ones utilized in this tutorial) are as follows:

1. Prepare a protein structure for simulation.
2. Generate a series of configurations along a single degree of freedom (reaction coordinate).
3. Run umbrella sampling simulations on each. configuration to restrain it within a window corresponding to the chosen COM distance.
4. Use the Weighted Histogram Analysis Method (WHAM) to extract the PMF and calculate $\triangle G_{unfolding}$.

For more detailed explanations of umbrella sampling see any of the following papers:

J Phys Chem B. 2009 Feb 26;113(8):2234-46.
Biochim Biophys Acta. 2015 May;1850(5):872-877.
Proc Natl Acad Sci U S A. 2005 May 10;102(19):6825-30.

Step One: Prepare the Topology

Here we generate a molecular topology for an umbrella sampling simulation. The system that we are going to work with is a short helical peptide.

1. load the Gromacs software onto your local machine using the following command:

```
module load environments/applications/SERIAL/gromacs-4.6.7-
    singleprecision-plumed-2.1.2
```

2. Download *helix.pdb*, *ions.mdp*, *min.mdp* and *npt.mdp* from my GitHub repository:
   https://github.com/jamieAmacpherson/canes_tutorial

3. Convert the helix peptide from .pdb format to a useable structure format (.gro):

```
pdb2gmx_s -f helix.pdb -ignh -water spce -o helix.gro
```

   When prompted to select the force fields, choose GROMOS96 53A6.

   In addition to converting the structure file to a .gro format, this command will generate a structure topology.

Step Two: Define the Unit Cell

Defining the unit cell for a pulling simulation is not unlike defining the unit cell for any other simulation. There is, however, one major consideration. One must allow enough space in the pulling direction to allow for a continuous pull without interacting with the periodic images of the system.

Run the following:

```
editconf_s -f helix.gro -center 0 0 0 -o helix_000.gro
editconf_s -f helix_000.gro -princ -o  helix_000_p.gro
```

Select group 4 (Backbone)

Open $helix_0 00.gro$ with VMD. Inspect whether the principal axis is parallel to the z-axis. If it is not run the following command to rotate the molecule in order to orientate it along the z-axis:

```
editconf_s -f helix_000_p.gro -rotate 0 90 0 -o helix_000_z.gro
```

Now that our helix is oriented along the z-axis, we can extend the unit cell parameters so that we have enough space to pull the helix apart.

```
editconf_s -f helix_000_z.gro -o helix_newbox.gro -c  -box  3 3
    16
```

Step Three: Adding Solvent and Ions

In molecular simulations of biomolecular systems we must try to reproduce, as accurately as possible, the cellular conditions in which the proteins normally exist. This will make an observations we see more 'physiologically relevant'. To reproduce cellular conditions we will do two things: 1) add water molecules to our simulation and 2) neutralize the system by adding counter-ions to the box.

First, add water molecules to the simulation box:

```
genbox_s -cp helix_newbox.gro -cs spc216.gro -o solv.gro -p
    topol.top
```

Now add neutralize the system by adding counter-ions:

```
grompp_s -f ions.mdp -c solv.gro -p topol.top -o ions.tpr
genion_s -s ions.tpr -o solv_ions.gro -p topol.top -pname NA -
    nname CL -neutral -conc 0.1
```

You will be prompted to select which atoms you would like to replace with the added counter-ions, select 'Water'. Clearly it wouldn't be desirable to remove any protein atoms – water molecules are more dispensable in this instance!

Step Four: Energy Minimization and Equilibration

Prior to running our umbrella sampling simulation we need to first relax the potential energy of the protein system. To do this we will perform an energy minimization using the steepest descent algorithm. This is then followed by a short simulation run in the isothermal-isobaric 'NpT' ensemble.

To perform energy minimization, run the following:

```
grompp_s -f min.mdp -c solv_ions.gro -p topol.top -o min.tpr
mdrun_s -deffnm min -s min.tpr
```

Now generate the binary input file for the NpT ensemble run and run the equilibration:

```
grompp_s -f npt.mdp -c min.gro -p topol.top -o npt.tpr
mdrun_s -deffnm npt -s npt.tpr
```

Data analysis

You will see that one of the files that is generated from both the minimization and equilibration steps ends with *.edr* - this is a file which stores energies of the system over the course of the simulation. Using the following command we can extract energies from the *.edr* file produced by the energy minimization step with the following:

```
g_energy_s -f min.edr -o minimization_energies.xvg
```

n.b. You will be prompted to select which energy types you would like to extract.

In addition to the system energy, we would also like to monitor structural properties of the helix. The following can be used to measure the deviation of helix atoms from their starting structure:

```
g_rms_s -f npt.xtc -s npt.tpr -o rmsd.xvg
```

.xvg files can be loaded using using any graphic software (Matlab, R, gnuplot or xmgrace). For a tutorial on xmgrace see the following:
http://mintaka.sdsu.edu/reu/grace.tutorial.html