Molecular Dynamics Simulation and Analysis in a Notebook

Python is the perfect glue language, with big emphasis on readability and an awesome ecosystem of packages for all kind of areas, ranging from web development or machine learning to the most interesting science. In this short tutorial, we will focus on molecular dynamics as provided by the Omnia project.

Omnia is a compilation of diverse packages for chemistry simulations, but its main pillar is OpenMM: “a toolkit for molecular simulation using high performance GPU code”. While written in C++/CUDA/OpenCL, it also offers rich APIs in Python. A good number of several other packages use OpenMM or provide additional functionalities, such as pdbfixer (preparation and cleaning), mdtraj or pytraj (analysis), nglview (visualization), parmed (converters between different MD programs) or openmoltools (a bit of everything, actually).

We will use all of them at some point of the tutorial, outlined here:

- **Part A**
  - Prepare an environment for MDs with conda
  - Download, clean and prepare structures from PDB with pdbfixer
  - Set up an OpenMM MD simulation from scratch
- **Part B**
  - Preview the MD movie in the notebook with nglview
  - MD Analysis with pytraj or mdtraj
  - Custom clustering with the scipy stack

A.1 - Prepare the environment

First, create a new Python 3.5 environment and name it freely. We are using Python 3.5 and not 3.6 because some packages aren't still available for this version.

```
$> conda create -n openmm python=3.6 nomkl ipython jupyter
notebook
```

The following packages will be required along the tutorial:

- openmm
- mdtraj
- pdbfixer
- nglview
- parmed
- openmoltools

However, they are not provided in the standard conda channels like numpy or ipython. They are hosted in the omnia channel, so we need to specify that with the -c switch:

```
$> conda install -c omnia openmm mdtraj pdbfixer parmed
openmoltools
```

nglview is not up-to-date in the omnia channel, but it is in bioconda:

```
$> conda install -c bioconda -c conda-forge nglview ipywidgets

$> jupyter-nbextension enable nglview --py --user
```

That's it! Now, try to import this packages to test if they are correctly installed.

A.2 - Download and prepare your structure

Disclaimer: This tutorial is not meant as a MD crash course, but as an exercise to play with Python capabilities in science. Hence, I am not trying to be very cautious about the MD simulation itself. We will use small structures that will allow us to progress rapidly with a modest computer. Remember, we don't care about the MD, only the setup process!

To run a proper MD, we need to download the structure from some place (PDB, for example), but that's only the first step. Sometimes, the structures are not complete, or include wrong residue names. Some include non-natural residues! Depending on the origin of the files, hydrogens must be added, and depending on which kind of simulation we want to run, we might need to solvate the structure (ie, surround the molecule with a box of waters). While that can be easily attained with Chimera or another GUI programs, we are going to use Python tools... Just because.

Another benefit of learning this procedure is that you could automatize the same pipeline for hundreds of files... :)