6. Analysis

Analysis should normally be performed locally on your workstation, i.e. copy back all the files from the supercomputer to your local directory. However, if you *must* run analysis on saguaro then read the next section on *analysis jobs on saguaro*.

A typical analysis tasks reads the trajectory (XTC) or energy (EDR) file, computes quantities, and produces datafiles that can be plotted or processed further, e.g. using Python scripts. A strength of Gromacs is that it comes with a wide range of tools that each do one particular analysis task well (see the Gromacs manual and the Gromacs documentation).

Analysis jobs

If you have to perform analysis on saguaro you will have to submit an analysis job. You must write your own submission script with the analysis commands included. It is important that you **set the number of nodes to 1** (`#PBS -l nodes=1`) because no analysis tool can make use of multiple nodes and you would simply waste CPU-hours—we get billed for the requested nodes, not for what you actually use!

An analysis submission script could be similar to the following:

```
#!/bin/bash
#PBS -N analysis
#PBS -l nodes=1
#PBS -l walltime=00:10:00
#PBS -A phy598s113
#PBS -j oe
#PBS -o analysis.$PBS_JOBID.out

# host: saguaro
# queuing system: PBS

LIBDIR=/home/obeckste/Library

cd $PBS_O_WORKDIR

# use the serial executables (without MPI)
. $LIBDIR/Gromacs/versions/serial-4.5.5/bin/GMXRC

# example RMSD calculation; replace with your command
printf "Calpha\nCalpha\n" | g_rms -s md.tpr -f md.xtc -n ca.ndx -o rmsd.xvg -fit rot+tran
```

Note that *all required input files* must be present in the directory from which you are submitting.

6.1. Trajectory visualization

If you just look at the output trajectory `md.xtc` in VMD then you will see that the protein can be split across the periodic boundaries and that the simulation cell just looks like a distorted prism. You should *recenter* the trajectory so that the protein is at the center, *remap* the water molecules (ans ions) to be located in a more convenient unitcell representation, and it is often desirable to *RMS-fit* the protein on a reference structure (such as the first frame in the trajectory) to remove overall translation and rotation.

In Gromacs, the trjconv tool can do all these "trajectory conversion tasks". The protocol is to (1) center and remap and (2) to RMS-fit (due to technical limitations in **trjconv** you cannot do both at the same time). **trjconv** asks the user a number of questions that depend on the chosen options. In the command line snippets below, this user input is directly fed to the standard input of **trjconv** with the `printf TEXT | trjconv` "pipe" construct. In order to better understand the command, run it interactively without the pipe construct and manually provide the required information.

1. Center (`-center`) on the *Protein* and remap all the molecules (`-pbc mol`) of the whole *System*:

```
printf "Protein\nSystem\n" | trjconv -s md.tpr -f md.xtc -center -ur compact -pbc mo
```

2. RMS-fit (`-fit rot+trans`) to the protein *backbone* atoms in the initial frame (supplied in the TPR file) and write out the whole *System*:

```
printf "Backbone\nSystem\n" | trjconv -s md.tpr -f md_center.xtc -fit rot+trans -o m
```

Visualize in VMD:

```
vmd ../posres/posres.pdb md_fit.xtc
```

6.2. Observables

A number of interesting quantities and observables [1] can be calculated with Gromacs tools. A selection is shown below but you are encouraged to read the Gromacs manual and the Gromacs documentation to find out what else is available.

> **Note:** The online manual is the documentation for Gromacs 4.6.1 at the moment whereas we are using version 4.5.5. Some of the descriptions may vary slightly. If in doubt, check the help function (`-h` or `man COMMAND`) or the

Gromacs 4.5.6 PDF from the manual section.

Selection of Gromacs analysis tools

g_energy
: basic thermodynamic properties of the system

g_rms
: calculate the root mean square deviation from a reference structure

g_rmsf
: calculate the per-residue root mean square fluctuations

g_gyrate
: calculate the radius of gyration

g_dist, g_mindist
: calculate the distance between atoms or groups of atoms (make a index file with make_ndx to define the groups of interest). **g_mindist** is especially useful to find water molecules close to a region of interest.

  For AdK, look at the distance between the C_α of K145 and I52.

do_dssp
: Use the DSSP algorithm [Kabsch1983] to analyze the secondary structure (helices, sheets, ...).

6.2.1. RMSD

The RMSD is the root mean squared Euclidean distance in *3N* configuration space as function of the time step,

$$\rho^{\mathrm{RMSD}}(t) = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \left(\mathbf{r}_i(t) - \mathbf{r}_i^{\mathrm{ref}}\right)^2}$$

between the current coordinates $\mathbf{r}_i(t)$ at time *t* and the reference coordinates $\mathbf{r}_i^{\mathrm{ref}}$.

We compute the C_α **RMSD** with g_rms with respect to the reference starting structure (the one used for creating the `md.tpr` file). Work in a separate analysis directory:

```
mkdir analysis/RMSD && cd analysis/RMSD
```

First we **create an index file** for the C_α atoms [2]. Use make_ndx to create a

file `ca.ndx` that contains the C_α atoms as an *index group*. Start **make_ndx** and use `md.tpr` as input; the output index file will be `ca.ndx`:

```
make_ndx -f ../../MD/md.tpr -o CA.ndx
```

Use make_ndx interactively by typing the following commands [3]:

```
keep 1
a CA
name 1 Calpha
q
```

(This sequence of commands only retains the “Protein” default selection, then selects all atoms named “CA”, renames the newly created group to “Calpha”, and saves and exits.)

You can look at `CA.ndx` and see all the index numbers listed under the heading `[ Calpha ]`.

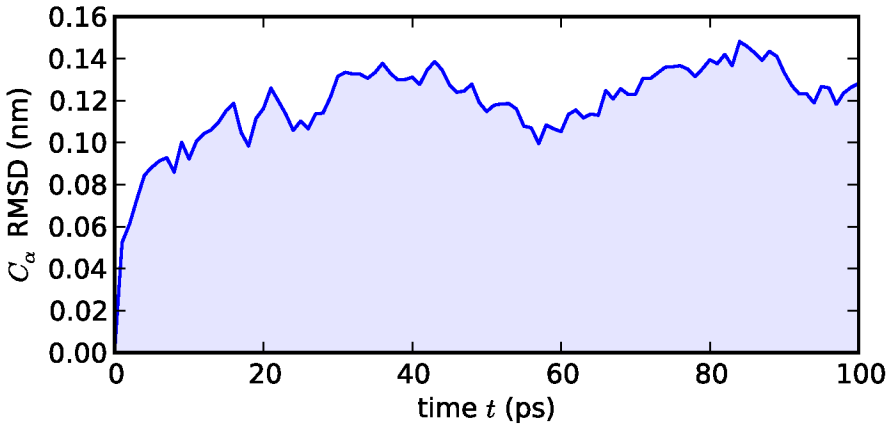
Run **g_rms**, using our newly defined group as the selection to fit and to compute the RMSD:

```
printf "Calpha\nCalpha\n" | g_rms -s ../../MD/md.tpr -f ../../MD/md.xtc -n CA.ndx -o rmsd
```

Note that the units are nm.

Plot the `rmsd.xvg` file as usual (you might want to use `g_rms -xvg none` if you are processing with NumPy/matplotlib):

```
import matplotlib.pyplot as plt
import numpy
t,rmsd = numpy.loadtxt("rmsd.xvg", unpack=True)
fig = plt.figure(figsize=(5,2.5))
ax = fig.add_subplot(111)
fig.subplots_adjust(bottom=0.2)
ax.set_xlabel("time $t$ (ps)")
ax.set_ylabel(r"C$_\alpha$ RMSD (nm)")
ax.fill_between(t,rmsd, color="blue", linestyle="-", alpha=0.1)
ax.plot(t,rmsd, color="blue", linestyle="-")
fig.savefig("rmsd_ca.png", dpi=300)
fig.savefig("rmsd_ca.svg")
fig.savefig("rmsd_ca.pdf")
```



Root mean square distance (RMSD) of the C_α atoms of AdK from the initial simulation frame.

6.2.2. RMSF

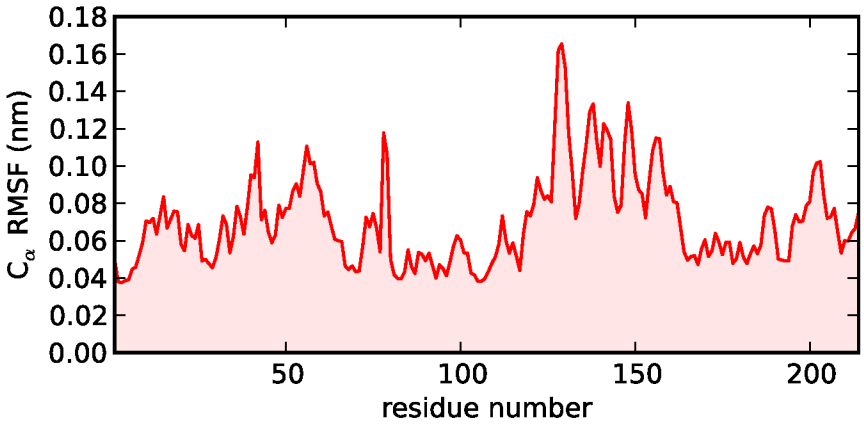
The residue root mean square fluctuation **RMSF** is a measure of the flexibility of a residue. It is typically calculated for the C_α atom of each residue and is then simply the square root of the variance of the fluctuation around the average position:

$$\rho^{\mathrm{RMSF}}_i = \sqrt{\left\langle \left(\mathbf{r}_i - \langle \mathbf{r}_i \rangle \right)^2 \right\rangle}$$

Use the `CA.ndx` file from the *RMSD* calculation with g_rmsf:

```
mkdir analysis/RMSF && cd analysis/RMSF
printf "Calpha\n" | g_rmsf -s ../../MD/md.tpr -f ../../MD/md.xtc -n ../RMSD/CA.ndx -o rms
```

A plot of ρ^{RMSF}_{i} versus residue number *i* shows the regions of high flexibility as peaks in the plot. Note that a 100-ps simulation might be too short to obtain a meaningful RMSF profile.



Root mean square fluctuation (RMSF) of the C_α atoms of AdK.

Comparison with crystallographic B-factors

You can compare the RMSF to the isotropic atomic crystallographic B-factors, which are related by [Willis1975]

\[B_{i} = \frac{8\pi^2}{3} (\rho^{\mathrm{RMSF}}_{i})^2\]

(In this case you would want to calculate the RMSF for all heavy (i.e. non-hydrogen) atoms. You don't need to build and use a separate index file file: simply choose the default group "Protein-H" ("protein without hydrogens")).

> **Note:** Gromacs RMSF are in units of nm and B-factors are typically measured in $Å^2$.

It is straightforward to write Python code that calculates the B-factor from the RMSF in `rmsf.xvg` and it is also easy to extract the B-factor ("temperatureFactor") from columns 61-66 in the ATOM record of a PDB file.

6.2.3. Distances

Distances can be a very useful observable to track conformational changes. They can often be directly related to real experimental observables such as NOEs from NMR experiments or distances from cross-linking or FRET experiments.

Here we calculate a simple distance between two C_α atoms as an approximation to the distance between two chromophores attached to the correspoding residues isoleucine 52 (*I52*) and lysine 145 (*K145*) used in a FRET experiment [Henzler-Wildman2007].

First we need to create an index file containing the two groups:

```
mkdir -p analysis/dist/I52_K145 && cd analysis/dist/I52_K145
make_ndx -f ../../../MD/md.tpr -o I52_K145.ndx
```

Use interactive commands like the following [4]:

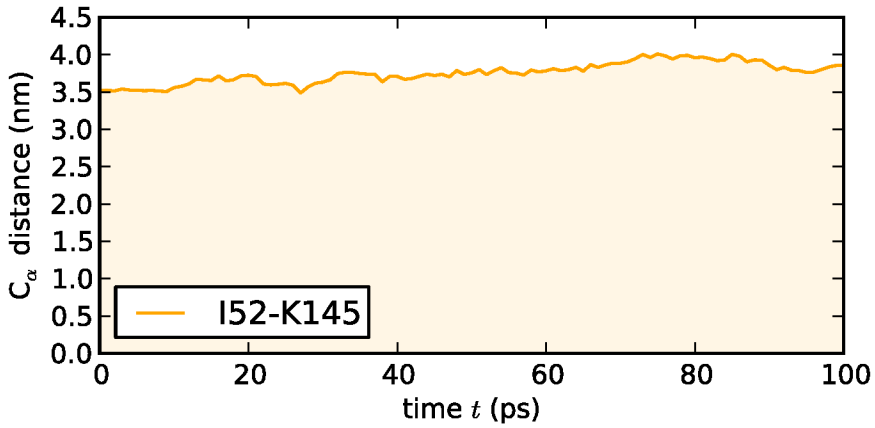
```
keep 0
del 0
r 52 & a CA
name 0 I52
r 145 & a CA
name 1 K145
q
```

to generate the index file `I52_K145.ndx`.

The run g_dist and compute the distance between the two atoms:

```
printf "I52\nK145\n" | g_dist -s ../../../MD/md.tpr -f ../../../MD/md.xtc -n I52_K145.ndx
```

The `dist.xvg` file contains the distance in nm for each time step in ps.



Timeseries of the distance between the C_α atoms of I52 (NMP domain) and K145 (LID domain).

(You can also use the centered and fitted trajectory `md_fit.xtc` as an input instead of `md.xtc` to make sure that the distance calculation does not contain any jumps due to periodic boundary effects, or use g_mindist.)

See also: [Beckstein2009] for a discussion of FRET distances in AdK.

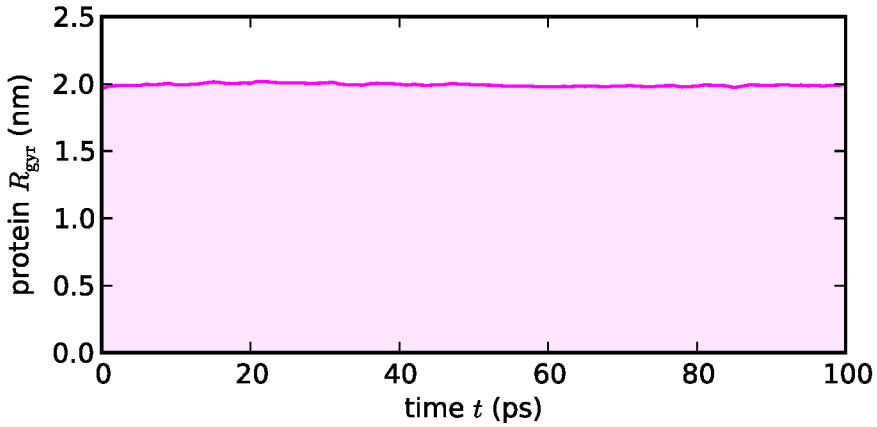
6.2.4. Radius of gyration

The radius of gyration measures the compactness of a protein structure.

$$R_\mathrm{gyr}^2 = \frac{1}{M}\sum_{i=1}^{N} m_i(\mathbf{r}_i - \mathbf{R})^2$$

where $M = \sum_{i=1}^{N} m_i$ is the total mass and $\mathbf{R} = N^{-1}\sum_{i=1}^{N} \mathbf{r}_i$ is the center of mass of the protein consisting of N atoms.

The Gromacs tool g_gyrate can be used to compute it.



Timeseries of the radius of gyration computed for the whole protein.

6.2.5. Secondary structure

Installation of DSSP

In order to use the Gromacs do_dssp command you need to install an additional program, the **dsspcmbi** executable that performs the DSSP algorithm [Kabsch1983]:

- put the provided `bin/MacOSX/dsspcmbi` Mac OS X executable into your `~/opt/bin` directory on your workstation:

  ```
  cp bin/MacOSX/dsspcmbi ~/opt/bin
  ```

  (If you are using your own Linux machine you can try using the `bin/Linux/dsspcmbi` or you will need to download the DSSP source code ("DSSPold") yourself and compile it.)

- set the environment variable `DSSP` in your `~/opt/bin/GMXRC.bash` Gromacs startup file to tell the Gromacs tool **do_dssp** where to find **dsspcmbi**:

```
export DSSP=$HOME/opt/bin/dsspcmbi
```

(You may also set `DSSP` it in your `~/.profile` bash startup file.)

You only have to install **dsspcmbi** once.

do_dssp analysis

Install DSSP as described above. Look at the options of do_dssp.

Footnotes

[1] "Observable" is used in the widest sense in that we know an estimator function of all or a subset of the system's phase space coordinates that is averaged to provide a quantity of interest. In many cases it requires considerable more work to connect such an "observable" to a true experimental observable that is measured in an experiment.

[2] Actually, we don't need to create the index group for the C_α atoms ourselves because Gromacs automatically creates the group "C-alpha" as one of many default groups (other are "Protein", "Protein-H" (only protein heavy atoms), "Backbone" (N CA C), "Water", "non-Protein" (i.e. water and ions in our case but could also contain other groups such as drug molecule or a lipid membrane in more complicated simulations), "Water_and_ions". You can see these index groups if you just run **make_ndx** on an input structure or if you interactively select groups in **trjconv**, **g_rms**, ...

However, making the "Calpha" group yourself is a good exercise because in many cases there are no default index groups for the analysis you might want to do.

[3] In scripts you can pipe all the interactive commands to make_ndx by using t `printf ... | make_ndx` trick:

```
printf "keep 0\ndel 0\na CA\nname 0 Calpha\nq\n" | make_ndx -f ../../MD/md.tpr -o CA.n
```

This will accomplish the same thing as the interactive use described above.

[4] Note that one has to be careful when selecting residue ids in **make_ndx**. It is often the case that a PDB file does not contain all residues, e.g. residues 1–8 might be unresolved in the experiment and thus are missing from the PDB file. The file then simply starts with residue number 9. Gromacs, however, typically *renumbers residues so that they start at 1*. Thus, in this hypothetical case, a residue that might be referred to in the

literature as “residue 100” might actually be residue 92 in the simulation (\(N^\mathrm{sim}_\mathrm{res} = N^\mathrm{PDB}_\mathrm{res} - (\mathrm{min} N^\mathrm{PDB}_\mathrm{res} - 1)\)). Thus, if you wanted to select the C_α atom of residue 100 you would need to select `r 92 & a CA` in **make_ndx**.