Protocol for Residence Time calculation

In this protocol, we’ll cover all the steps needed to estimate the residence time of a ligand bound to a protein, and hence its koff.

# 1. System preparation

The first step will be to parametrize the ligand-protein system with a forcefield of choice.

# 2. Plain MD simulation in AMBER

Once the solvated system has been prepared, a plain molecular dynamic simulation can be performed in AMBER, to equilibrate the system and also to collect some data that will be needed as input for the following metadynamic simulations.

## 2.1 Creation of GROMACS topology and coordinate files

Unfortunately, currently (as of Sept. 2023) *plumed* is not well suited to work with AMBER, as it will slow dramatically AMBER simulations because:

* it will force a constant sharing of data between the CPU and the GPU
* it’s not possible to run *plumed* with MPI on a higher number of CPU cores as the ones used by AMBER (usually 1).

GROMACS is the common choice when *plumed* has to be used, so we’ll switch to it for metadynamic simulations. Fortunately, *parmed* from AMBER can be used to translate an AMBER topology file to GROMACS format. As starting coordinates for the metadynamic simulations we’ll take the last AMBER restart file from the equilibrated trajectory.

A script that uses *parmed* and *pytraj* from AMBER is provided in the *data* directory (*amber\_to\_gromacs\_conversion.py*). The script also uses some *pytraj* functions to collect some information that will be needed in the setting of the metadynamic simulations. The user is invited to read the script with all its comments to continue the protocol. In the script, the ligand residue name is assumed to be TRM.

# 3. Frequency adaptive metadynamics

## 3.1 Collective variables

To accelerate the unbinding of the drug through the frequency adaptive metadynamics (FaMetaD) simulations, the first step will be to determine the collective variable(s) (CV(s)) that will be used to push the unbinding of the ligand. Such CVs must be able to correctly drive the ligand outside of its binding site, and should well discriminate between the bound/unbound ligand system configurations. The use of more than 4 CVs is not efficient in metadynamic simulations, so one should use as few as possible, keeping in mind that usually, one simple CV is not enough to perform FaMetaD unbinding simulations. All information available about the system we are studying should be used to define a good set of CVs (e.g. if the protein needs to undergo conformational changes to release the ligand, or if some specific amino acid is particularly important for keeping the ligand in place into the binding site, etc.).

The distance between the ligand's center of mass (COM) and the COM of the binding site is a very common CV but, as stated above, usually should be coupled with at least another CV. This second CV should be able for example to discriminate the situation in which the ligand is outside of the binding site while rolling on the surface of the protein, a condition which is not well described by the simple COM distance. One such CV could be the number of water molecules that are found inside a defined radius around the ligand’s COM. This number will intuitively increase as long as the ligand moves further away from the protein.

## 3.2 Plumed file preparation

To define the aforementioned CVs in a plumed input file, we must define the two centers of masses:

bindingsiteCOM: COM ...

ATOMS=’index of the CA atoms of residues within 5 Å from the ligand’

...

ligandCOM: COM ...

ATOMS=’index of the ligand heavy atoms’

...

Where to identify the atoms’ indexes one can use the ambmask tool. Here is an example of the use of ambmask to select the CA atoms:

ambmask -p prmtop -c rst7 -find ":LIGAND<:5&@CA"

Then, we must define a group of atoms containing water’s oxygen atoms, assuming all water molecules are contiguous and that a 3 atoms model was used:

watOgroup: GROUP ATOMS=firstWAToxygen-lastWAToxygen:3

At this point the two CVs can be defined:

COMdistance: DISTANCE ATOMS= bindingsiteCOM, ligandCOM

WATcontacts: COORDINATION ...

GROUPA=ligandCOM

GROUPB=watOgroup

R\_0=8.0

D\_0=3.0 # the function starts to decay after D\_0

...

The number of ligand-water contacts is here computed as , where represent the formation of contact between atom and . The value of is computed via a switching function (with parameters defined by R\_0 and D\_0) as

Finally, the instructions to perform FaMetaD can be defined:

metad: METAD ...

ARG=COMdistance,WATcontacts

SIGMA=’s1’,’s2’

HEIGHT=1.0

BIASFACTOR=15.0

TEMP=300

PACE=500

FREQUENCY\_ADAPTIVE

FA\_MIN\_ACCELERATION=300000

FA\_MAX\_PACE=50000

GRID\_MIN=’gmin1’,’gmin2’

GRID\_MAX=’gmax1’,’gmax2’

FILE=HILLS

ACCELERATION

#ACCELERATION\_RFILE=COLVAR

# The last line should be uncommented if the simulation is a restart from a previous run (for example if ran on a HPC cluster).

...

stopper: COMMITTOR ...

ARG= COMdistance

stride=5000

BASIN\_LL1=’distance at which the ligand is unbound’

BASIN\_UL1=’BASIN\_LL1+50’

FILE=committor.dat

...

PRINT STRIDE=5000 ARG=\* FILE=COLVAR

In the METAD section:

1. The ’s1’ and ’s2’ values are the sigma of the Gaussians deposited for each CV, and a good starting value can be identified by running a 100 ns long classical MD simulation while tracking the CV values. This can be done with the same *plumed* input file used for the FaMetaD simulations, but commenting out the METAD part. After the simulation has been run, half of the standard deviation of each CV can be used as the respective SIGMA value in the FaMetaD simulations.
2. The ’gmin1’,’gmin2’ and ’gmax1’,’gmax2’ values are used to define a grid on which the bias deposited during the simulation will be stored. The grid boundaries must be abundant enough that during the simulation the CVs will never go outside of these boundaries, as in such cases the simulation will halt immediately. Intuition or a preliminary steered molecular dynamic simulation can be used to define the boundary for the grid, keeping in mind that a bigger grid will need more memory to be stored during the simulation.

The COMMITTOR section will automatically stop the simulation when some condition is reached. It can be used here to stop the simulation when the distance between the ligand and the protein is such that the unbinding of the ligand had happened.

The PRINT section will store in a file called COLVAR all the CV values and METAD parameters once every STRIDE step.

## 3.3 Data collection

Having prepared the *plumed* input file, the simulation can be run with GROMACS. Among the standard output files created during the simulation, *plumed* will write in a file named COLVAR the information regarding CVs and metadynamic. With the CVs defined above, the file will have 6 columns, collecting in the order: time, COMdistance, WATcontacts, metad.bias, metad.acc, and metad.pace. The COMdistance and WATcontacts columns can be used to track the advancement of the simulation, while the metad.acc column reports the acceleration factor of the process. If the COMMITTOR was properly defined, the simulation should automatically stop when the ligand finally moves away from the protein. From the visual inspection of the trajectory, we can confirm the unbinding event, and by multiplying the simulation time by the relevant acceleration factor stored in the COLVAR file, the time needed to observe the unbinding event can be obtained. This process must be repeated a sufficient number of times (25 times in our example), to collect enough data to be analyzed to obtain the proper residence time of the ligand.

# 5. Data analysis

Once at least 15 unbinding events are observed by FaMetaD, the data collected can be processed to estimate the residence time of the ligand. However, simply taking the average of the collected data is not advisable, as mean residence time may be susceptibly altered with even one sample falling into the long-time tail of the distribution. Observing that the ligand unbinding is a rare event, we know that the distribution of the dissociation times is expected to be exponential, characteristic of a homogenous Poisson process.

To determine the residence time, we can then proceed as follow:

1. We compute the empirical cumulative distribution function (CDFE), which describes the probability of observing at least one unbinding event within time *t*.
2. We fit the data with the equation of the theoretical cumulative distribution function (CDFT) of a Poisson process

o retrieve the characteristic transition time, , which represents the average residence time of our ligand.

1. We use the two-sample Kolmogorov-Smirnov (KS) test to assess whether the data collected indeed follows a Poisson distribution. In doing so, we check the *null* hypothesis that the sample of transition times extracted from FaMetaD and a large sample of times randomly generated according to the exponential probability distribution reflect the same underlying distribution. The *p-value* represents the probability that the distribution of times extracted from metadynamics is obtained from the theoretical exponential distribution, and a *p-value* greater than 0.05 must be obtained.

Such procedures can be easily implemented, here’s a python example for each step, in which we assume to have imported the library numpy as np, and stored in the variable data the collected unbinding times:

a)

range\_factor = 100

nbins = 10\_000

bins\_time = np.logspace(

np.log10(np.amin(data)/range\_factor), np.log10(np.amax(data)\*range\_factor),

num=nbins, base=10)

hist\_values, \_ = np.histogram(data, bins=bins\_time)

hist\_values = np.append(hist\_values,0)

cdf\_e = np.cumsum(hist\_values)/len(data)

b)

from scipy.optimize import curve\_fit

def homogenous\_poisson\_process\_cdf(t, rate):

# rate = 1/tau

return 1-np.exp(-x\*rate)

popt, \_ = curve\_fit(

homogenous\_poisson\_process\_cdf,

xdata= bins\_time,

ydata=cdf\_e,

p0=1/np.mean(data)

)

tau = 1/popt[0]

c)

from scipy.stats import ks\_2samp

sampling\_from\_theoretical\_distribution = np.random.exponential(

scale=tau,

size=int(len(data)\*1E6))

\_, ks\_p\_value = ks\_2samp(data, sampling\_from\_theoretical\_distribution)

To calculate the standard error of the computed residence time, a bootstrap analysis can be performed, resampling the experimental dataset.