Tutorial:
Running τRAMD using Gromacs v.1.0

04.2020, updated: 06.2021

Daria B. Kokh, Bernd Doser, Xingyi Cheng, Stefan Richter, Rebecca C. Wade,

Heidelberg Institute for Theoretical Studies (HITS gGmbH), Germany

Email contact: mcmsoft@h-its.org

Content

1. Introduction, general comments and pre-processing

- This tutorial provides guidelines for using the τ-Random Acceleration Molecular Dynamics (τRAMD) procedure to compute relative protein-ligand residence times by generating RAMD trajectories using the GROMACS software. The τRAMD procedure is described in Refs. [2]. RAMD simulations may also be run using GROMACS to simply explore ligand egress routes from a protein binding site using the relevant parts of this tutorial. In RAMD simulations, conventional molecular dynamics (MD) simulations are run with an additional, randomly oriented, force of a constant user-defined magnitude applied to the center of mass of the ligand. The direction of this force is changed during a RAMD simulation according to the extent of movement of the ligand. The RAMD method typically enables ligand egress from protein binding sites to be observed in times of a few nanoseconds. Many RAMD simulations must be run in order to generate an ensemble of ligand egress routes from a given protein binding site.
- The τRAMD procedure was originally implemented using NAMD and is described in a tutorial (https://kbbox.h-its.org/toolbox/tutorials/) where a more detailed description of the method can be found. This GROMACS tutorial for τRAMD is based on the NAMD tutorial. The same protein-ligand systems are simulated. The results are not identical due to different implementations of algorithms in NAMD and GROMACS, due to the use of random numbers in RAMD, and due to numerical error. However, with sufficient sampling, the computed egress routes and ligand residence times agree within numerical error. The implementation of RAMD in GROMACS provides better computing performance with better scalability than the implementation in NAMD. This difference arises because the random force is implemented inside the MD code in GROMACS rather than in a tcl script as for NAMD, for details see Ref.
- We usually use the Amber ff14 and Gaff force fields for the protein and the ligand, respectively, but any other force field can be in principle be used. Keep in mind the limitations of molecular mechanics force fields. For example, the presence of Br or I atoms in a bound compound will quite likely lead to an underestimation of its residence time because of unaccounted polarization effects.
- In this tutorial, the minimization, heating and equilibration of the system are carried out using the Amber software package, but this can be done using the Gromacs software instead.
- In the τRAMD procedure, RAMD simulations must be run starting from a number of different coordinates and velocities (referred to as starting replicas) generated in conventional MD simulations. Although it is possible to select several starting replicas from a single MD trajectory, it is strongly recommended to run several different trajectories to generate starting replicas for RAMD (with different seed numbers and velocities generated from the Maxwell distribution). This will ensure proper sampling of the bound state of the protein-ligand system.
- Good sampling of the bound state (i.e. generation of several starting replicas) and of dissociation routes (i.e. of multiple trajectories starting from each replica) is critical for obtaining reliable results with the τRAMD procedure. The statistical distribution should be checked and the number of replicas and trajectories increased if necessary (see Section "Analysis of simulation results")
- We have tested the use of different types of thermostats in RAMD runs with GROMACS. In the original NAMD implementation of the τRAMD procedure , Langevin dynamics was used with the relaxation time parameter set to tau=1 /ps. For Langevin dynamics in GROMACS (integrator = sd), the relaxation time parameter must be set twice as fast (tau_t=2 /ps; see http://manual.gromacs.org/documentation/2018/user-guide/mdp-options.html). We have found that there are small differences in the ligand dissociation times computed with

GROMACS with the Nosé–Hoover and Berendsen thermostats, for which the relaxation time parameter, tau_t, should be set to 1/ps. These differences are smaller than, or comparable to, the variation of the average dissociation times in different replicas. Moreover, often the slowest replica yields the same dissociation time across simulations with different thermostats, which suggests that the dissociation times are mainly defined by the starting conditions. Figure 1 shows a comparison of the results of τRAMD simulations run under different thermostat conditions with Gromacs with the results of NAMD Langevin dynamics and with experimental data.

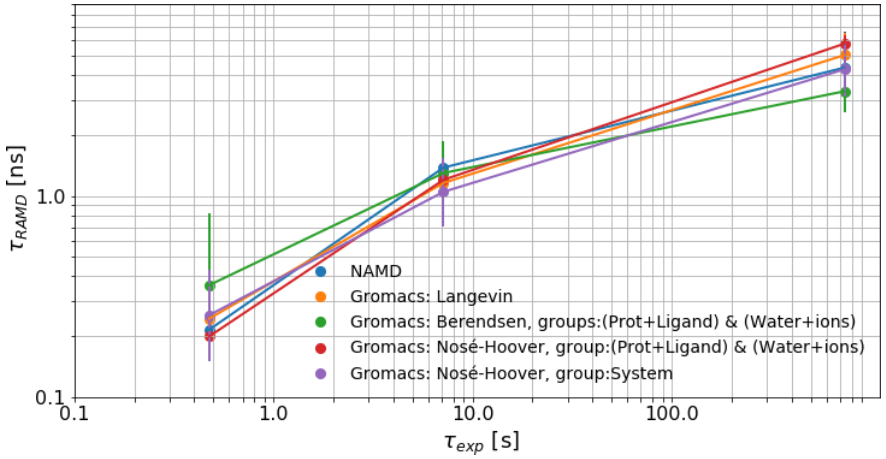

Fig.1 Test of the Gromacs thermostat parameters for three inhibitors of HSP90 (compounds 8, 16, and 20, see Refs. [1,2]): Computed τRAMD residence times are plotted against experimental residence times. Different thermostats were used in Gromacs: Nosé–Hoover (with two group assignments) and Berendsen with a relaxation time of 1 ps^{-1} and a Parrinello-Rahman barostat with a relaxation time of 5 ps^{-1}; Langevin dynamics with a relaxation time of 2 ps^{-1} and no barostat or group splitting. In the NAMD implementation, a Langevin thermostat (relaxation time of 1 ps^{-1}) and a Langevin barostat were used.

3. Required Software and Data

Software

- Gromacs (RAMD version - https://github.com/HITS-MCM/gromacs-ramd)
- Python (v.3.7 or above; libraries: numpy.py; matplotlib, pylab, seaborn, scipy)

2. Organization of the file system for τRAMD simulations

All files needed for this tutorial are available in the gromacs-ramd.tar.gz file. The directories are organized as follows in this tutorial:

```
AMBER/                      contain topology and coordinate files
Gromacs/
      Replica1/             contain conventional MD equilibration trajectories
      Replica2/
            .....
RAMD/
     Replica1/
          TRJ-1             contain RAMD trajectories
          TRJ-2
```

```
        ...
        TRJ-10
    Replica2/
        TRJ-1
        TRJ-2
        ...
        TRJ-10


    .....
```

Important: For each ligand, at least 4 starting replicas must be prepared from conventional MD (equilibration) and, for each starting replica, at least 10 ligand dissociation RAMD trajectories must be simulated.

3. System preparation using AMBER with the Amber (Amber 14ff) and GAFF force fields

○ Generate the Amber topology file and starting coordinates

Follow steps 1 &2 in the NAMD tutorial for τRAMD at : https://kbbox.h-its.org/toolbox/tutorials/estimation-of-relative-residence-times-of-protein-ligand-complexes-using-random-acceleration-molecular-dynamics-ramd/

○ Energy minimize, heat and equilibrate the protein-ligand systems

We use AMBER for energy minimization, heating and equilibration of each protein-ligand complex. We apply gradually decreasing restraints on the heavy atoms of the protein and ligand. Here is an example of the preparation pipeline using the Amber package with the calculations performed with the pmemd program. Note that the residue numbers defined in the “restraintmask” should be defined according to the system studied.

```
echo "
# Minimization performed for relaxing the solute : restraint_wt=500
 &cntrl
  imin=1, ntx=1, irest=0, ntrx=1, ntxo=1,
  ntpr=25, ntwr=500, ntwx=0, ntwv=0, ntwe=0,
  ntf=1, ntb=1, dielc=1.0, cut=10.0,
  nsnb=25, igb=0,
  ntr=1, restraint_wt=500, restraintmask='(:1-277 & !@H= & !@Na= & !@Cl=)',
  maxcyc=1500, ntmin=1, ncyc=500, dx0=0.01, drms=0.0001,
  ntc=1,
 /
" > ref_min-500
mpiexec pmemd.MPI -O -i ref_min-500 -p ref.prmtop -c ref.inpcrd -ref ref.inpcrd -o ref-min-500.ou -r ref-min-500.crd -inf ref-min-500.log

echo "
# Minimization performed for relaxing the solute : restraint_wt=100
```

```
 &cntrl
  imin=1, ntx=1, irest=0, ntrx=1, ntxo=1,
  ntpr=25, ntwr=500, ntwx=0, ntwv=0, ntwe=0,
  ntf=1, ntb=1, dielc=1.0, cut=10.0,
  nsnb=25, igb=0,
  ntr=1, restraint_wt=100, restraintmask='(:1-277 & !@H= & !@Na= & !@Cl=)',
  maxcyc=1500, ntmin=1, ncyc=100, dx0=0.01, drms=0.0001,
  ntc=1,
 /
" > ref_min-100
mpiexec pmemd.MPI -O -i ref_min-100 -p ref.prmtop -c ref-min-500.crd -ref ref-min-500.crd -o ref-min-100.ou -r ref-min-
100.crd -inf ref-min-100.log

echo "
# Minimization performed for relaxing the solute : restraint_wt=5
 &cntrl
  imin=1, ntx=1, irest=0, ntrx=1, ntxo=1,
  ntpr=25, ntwr=500, ntwx=0, ntwv=0, ntwe=0,
  ntf=1, ntb=1, dielc=1.0, cut=10.0,
  nsnb=25, igb=0,
  ntr=1, restraint_wt=5, restraintmask='(:1-277 & !@H= & !@Na= & !@Cl=)',
  maxcyc=1500, ntmin=1, ncyc=500, dx0=0.01, drms=0.0001
  ntc=1,
 /
" > ref_min-5
mpiexec pmemd.MPI -O -i ref_min-5 -p ref.prmtop -c ref-min-100.crd -ref ref-min-100.crd -o ref-min-5.ou -r ref-min-5.crd -
inf ref-min-5.log

echo "
# Minimization performed for relaxing the solute : no positional restraints
&cntrl
  imin=1, ntx=1, irest=0, ntrx=1, ntxo=1,
  ntpr=25, ntwr=500, ntwx=0, ntwv=0, ntwe=0,
  ntf=1, ntb=1, dielc=1.0, cut=10.0,
  nsnb=25, igb=0,
  ntr=0,
  maxcyc=1500, ntmin=1, ncyc=500, dx0=0.01, drms=0.0001,
  ntc=1,
 /
" > ref_min
mpiexec pmemd.MPI -O -i ref_min -p ref.prmtop -c ref-min-5.crd -ref ref-min-5.crd -o ref-min.ou -r ref-min.crd -inf ref-
min.log



echo "
# NVT Heating with restrained heavy atoms 50 kcal/mol*A^2
 &cntrl
  imin=0, ntx=1, irest=0, ntrx=1, ntxo=1,
  ntpr=100, ntwx=1000, ntwv=1000, ntwe=1000,
```

```
 ntc=2, ntf=2, ntb=1, dielc=1.0, cut=10.0,
 nsnb=100, igb=0,
 ntr=1, restraint_wt=50.0, restraintmask=':1-277 & !@H= & !@Na= & !@Cl=',
 nstlim=500000,
 t=0.0, dt=0.002,
 ntt=3, gamma_ln=1.0, tempi=10.0, temp0=300.0,
 vlimit=15,
 ntp=0,
 tol=0.00000001,
/
" > ref_heat
mpiexec pmemd.MPI -O -i ref_heat -p ref.prmtop -c ref-min.crd -ref ref-min.crd -o ref-heat.ou -r ref-heat.crd -inf ref-heat.log

echo "
# NPT equilibration with restrained heavy atoms 50 kcal/mol*A^2
 &cntrl
 imin=0, ntx=5, irest=1, ntrx=1, ntxo=1,
 ntpr=100, ntwx=1000, ntwv=1000, ntwe=1000,
 ntc=2, ntf=2, ntb=2, dielc=1.0, cut=10.0,
 nsnb=100, igb=0,
 ntr=1, restraint_wt=50.0, restraintmask=':1-277 & !@H= & !@Na= & !@Cl=',
 nstlim=1000000,
 t=0.0, dt=0.002,
 ntt=3, gamma_ln=1.0, tempi=300.0, temp0=300.0,
 vlimit=15,
 ntp=1, taup=1.0, pres0=1.0, comp=44.6,
 tol=0.00000001,
/
" > ref-equil-NPT-50
mpiexec pmemd.MPI -O -i ref-equil-NPT-50 -p ref.prmtop -c ref-heat.crd -ref ref-heat.crd -o ref-equil-NPT-50.ou -r ref-equil-
NPT-50.crd -inf ref-equil-NPT-50.log -x ref-equil-NPT-50.mdcrd

echo "
# NPT equilibration
 &cntrl
 imin=0, ntx=5, irest=1, ntrx=1, ntxo=1,
 ntpr=100, ntwx=1000, ntwv=1000, ntwe=1000,
 ntc=2, ntf=2, ntb=2, dielc=1.0, cut=10.0,
 nsnb=100, igb=0,
 nstlim=1000000,
 t=0.0, dt=0.002,
 ntt=3, gamma_ln=1.0, tempi=300.0, temp0=300.0,
 vlimit=15,
 ntp=1, taup=1.0, pres0=1.0, comp=44.6,
 tol=0.00000001,
/
" > ref-equil-NPT
```

```
mpiexec pmemd.MPI -O -i ref-equil-NPT -p ref.prmtop -c ref-equil-NPT-50.crd -ref ref-equil-NPT-50.crd -o ref-equil-NPT.ou -r ref-equil-NPT.crd -inf ref-equil-NPT.log -x ref-equil-NPT.mdcrd
```

4. Preparation of Gromacs input files from equilibrated AMBER topology & coordinate files

○ Generate the AMBER restart file for each simulated system

You will need AMBER topology (*.prmtop) and input restart (*.rst7) files (restart file *rst7 can be obtained from a coordinate file *crd using cpptraj) generated after energy minimization, heating, and a short equilibration (see, for example, the procedure described in the previous section) .

Here is an example of the input file for cpptraj to transfer *crd file to *rst7:

```
parm ref.prmtop
trajin ref-equil-NPT.crd
trajout ref-equil.rst7
run
```

○ Generate the Gromacs topology and coordinate files

Launch Python and use the ParmEd library (http://parmed.github.io/ParmEd/html/index.html) to prepare Gromacs topology (*.top) and *.gro files as follows:

```
import parmed as pmd
amber = pmd.load_file('ref.prmtop', 'ref-equil.rst7')
amber.save('gromacs.top')
amber.save('gromacs.gro')
```

5. System equilibration using GROMACS

○ Generate the index file containing the definition of subsystems

First, the index.ndx file with appropriate subsystems (called “groups” in Gromacs) needs to be generated using the interactive programe *make_ndx of* Gromacs:

gmx make_ndx -f gromacs.gro -o index.ndx

Subsystems can be: all membrane lipids, ‘Protein_ligand’, and ‘Water_and_ions’ .

For example, for the HSP90-inhibitor (INH) system, the subsystems are ‘Protein_INH’ and ‘Water_and _ions’: if the protein is group 1 and the ligand is group 13, then they can be combined as follows:

```
Group the protein and the ligand (INH in the case of HSP90) by typing in their corresponding numbers then quit selection:
1 | 13
```

```
q
```

○ Perform the first equilibration step (using the Berendsen thermostat)

The input files needed for this step are: gromacs gromacs.gro, index.ndx & gromacs.top

The input file, gromacs.mdp, should be prepared as in the example for HSP90 shown below:

gromacs.mdp

```
integrator              = md
comm-mode               = Linear
nstcomm                 = 100
comm_grps               = System
tinit                   = 0.000
dt                      = 0.002
nsteps                  = 10000000
nstxout                 = 25000
nstvout                 = 25000
nstlog                  = 10000
nstenergy               = 10000
nstxtcout               = 5000
nstfout                         = 25000
compressed-x-precision = 1000
xtc_grps                = SYSTEM
pbc                     = xyz
rlist                   = 1.10
coulombtype             = PME
cutoff-scheme           = Verlet
fourierspacing          = 0.12
pme_order               = 4
ewald_geometry          = 3d
ewald-rtol              = 1e-5
ewald-rtol-lj           = 1e-5
optimize_fft            = yesi
vdw-type                = Cut-off
vdw-modifier            = Potential-shift
rvdw-switch             = 0.0
rvdw                    = 1.00

tcoupl                  = Berendsen
tc_grps                 = Protein_INH  Water_and_ions
tau_t                   = 1.0   1.0
ref_t                   = 300 300

Pcoupl                  = no
gen_vel                 = yes
gen_temp                = 300
gen_seed                = 173529
continuation = no
constraints             = h-bonds
constraint_algorithm  = lincs
lincs_order             = 4
lincs_iter              = 1
lincs_warnangle         = 60
DispCorr                = EnerPres
```

Execution:

gmx grompp -f gromacs.mdp -c gromacs.gro -o gromacs0.tpr -n index.ndx -p gromacs.top -maxwarn 5

gmx mdrun -s gromacs0.tpr -ntmpi 1 -ntomp16 -maxh 24

- ### Perform the second equilibration step (using the Nose-Hoover thermostat and the Parrinello-Rahman barostat) to generate starting replicas

The input files needed for this step are: gromacs1.mdp, state.cpt, gromacs0.top, index.ndx & gromacs.top

The parameter file (gromacs.mdp) for this step should be prepared as in the example shown below.

Equilibration replicas should be generated in the directories: Replica1, Replica2, etc. (see Section **Organization of the file system for tauRAMD simulations**) in this step, with the output coordinates from the first equilibration step as input and using a Maxwell distribution of velocities by specifying the parameter "gen_vel = yes" for each replica (I.e. to start new trajectories).

Gromacs1.mdp

```
integrator              = md
comm-mode               = Linear
nstcomm                 = 100
comm_grps               = System
tinit                   = 0.000
dt                      = 0.002
nsteps                  = 10000000
nstxout                 = 50000
nstvout                 = 50000
nstlog                  = 2000
nstenergy               = 2000
nstxtcout               = 1000
nstfout                         = 50000
compressed-x-precision = 1000
xtc_grps                = SYSTEM
pbc                     = xyz
rlist                   = 1.10
coulombtype             = PME
cutoff-scheme           = Verlet
fourierspacing          = 0.12
pme_order               = 4
ewald_geometry          = 3d
ewald-rtol              = 1e-5
ewald-rtol-lj           = 1e-5
optimize_fft            = yesi
vdw-type                = Cut-off
vdw-modifier            = Potential-shift
rvdw-switch             = 0.0
rvdw                    = 1.00

tcoupl                  = Nose-Hoover
tc_grps                 = Protein_INH  Water_and_ions
tau_t                   = 0.8   0.8
ref_t                   = 300 300

Pcoupl                  = Parrinello-Rahman
pcoupltype              = isotropic
tau_p                   = 2
compressibility         = 4.5e-5
ref_p                   = 1
```

```
;gen_vel              = yes
gen_vel              = no
;gen_temp             = 300
;gen_seed             = 173529
continuation = no
constraints          = h-bonds
constraint_algorithm    = lincs
lincs_order          = 4
lincs_iter           = 1
lincs_warnangle      = 60
DispCorr             = EnerPres
```

Execution:

gmx grompp -f gromacs1.mdp -t state.cpt -c gromacs0.tpr -o gromacs1.tpr -n index.ndx -p gromacs.top -maxwarn 5

gmx mdrun -s gromacs1.tpr -ntmpi 1 -ntomp 16 -maxh 24

6. RAMD simulations using GROMACS

Generate ten trajectories from each starting replica in this step. The input parameter file (gromacs_ramd.mdp) should be prepared as in the example shown below.

The parameters highlighted in red define, respectively, the magnitude of the random force applied on the ligand, and the maximum distance between the centers of mass (COM) of the protein and the ligand. Each of these parameters should be defined by the user according to the system studied. The parameter 'ramd-seed' will be used to set a random direction for the random force applied on the ligand.

The following parameters have to be set:

- The magnitude of the random force in RAMD simulations. A reasonable starting value is 585 kJ/mol nm = 14 kcal/mol Å. The shortest RAMD residence time (this is not the ligand dissociation time in a single trajectory!) for a set of analyzed trajectories should be longer than 100 ps. If it is shorter, the magnitude of the random force should be decreased. The magnitude of the random force should be kept constant over the set of protein-ligand complexes studied.
- The distance between the ligand and protein COMs in the bound and unbound (i.e. when the ligand is considered to be dissociated) states: ramd-max-dist. This distance must be longer than the distance from the COM of the ligand in the bound complex to the protein surface. If this condition is satisfied, a difference of 10-20 Å does not make much difference, since the unbound ligand moves quickly when it has left the protein and the random force is applied.
- Name of "Protein" and "ligand" subsystems: ramd-receptor and ramd-ligand.

gromacs_ramd.mdp

```
integrator           = md
comm-mode               = Linear
nstcomm              = 100
```

```
comm_grps            = System
tinit                = 0.000
dt                   = 0.002
nsteps               = 20000000
nstxout              = 5000
nstvout              = 5000
nstlog               = 5000
nstenergy            = 5000
nstxtcout            = 5000
nstfout                      = 5000
compressed-x-precision = 1000
xtc_grps             = SYSTEM
pbc                  = xyz
rlist                = 1.10
coulombtype          = PME
cutoff-scheme        = Verlet
fourierspacing       = 0.12
pme_order            = 4
ewald_geometry       = 3d
ewald-rtol           = 1e-5
ewald-rtol-lj        = 1e-5
optimize_fft         = yes
vdw-type             = Cut-off
vdw-modifier         = Potential-shift
rvdw-switch          = 0.0
rvdw                 = 1.00
tcoupl               =  Nose-Hoover
tc_grps              = Protein_INH Water_and_ions
tau_t                = 1.0 1.0
ref_t                = 300 300
Pcoupl               = Parrinello-Rahman
pcoupltype           = isotropic
tau_p                = 2
compressibility      = 4.5e-5
ref_p                = 1
gen_vel              = no
continuation  = no
constraints          = h-bonds
constraint_algorithm = lincs
lincs_order          = 4
lincs_iter           = 1
lincs_warnangle      = 60
DispCorr             = EnerPres


ramd                 = yes
ramd-seed            = 98XX ; XX will be replaced by the trajectory number
ramd-force           = 585  ; = 14kcal/mol A
ramd-eval-freq       = 50 ; number of steps in each RAMD cycle
ramd-r-min-dist      = 0.0025      ; nm
ramd-force-out-freq     = 10
ramd-max-dist        = 4.0       ; nm
ramd-receptor        = Protein
ramd-ligand          = INH
```

Use the following scripts to set up the simulations and store them in the respective RAMD replica directories together with the gromacs_ramd.mdp file (note that this script assumes that the file structure is as given in Section “2. Organization of the file system for τRAMD simulations”). The job_gromacs-ramd.sh script automatically generates directories for the simulated trajectories and

submits the gromacs_ramd_mdrun.sh. The lines marked "#to be edited←---" must be edited for the particular job submission environment used.

Submit_20_jobs_gromacs-ramd.sh

```
#!/bin/bash
#SBATCH -N 1   #to be edited ←------------------------
#SBATCH -t 24:00:00  #to be edited ←------------------------
#SBATCH -B 2:8:1  #to be edited ←------------------------
#SBATCH -n 4  #to be edited ←------------------------
#SBATCH --threads-per-core=1  #to be edited ←------------------------
#SBATCH -p skylake-deep.p   #to be edited ←------------------------
#SBATCH –gres=gpu:1  #to be edited ←------------------------
module add  gromacs/gromacs-2019.3-ramd-1.0-rc2-skylake-deep  # to be edited ←------------------------


num=1 # replica number, change for each replcia
for name in {1..20}  # number of trajectories to be generated
  do
     mkdir TRJ$num-$name
     cd TRJ$num-$name
     ln -s ../../Gromacs/Replica$num/gromacs$num.tpr .
     ln -s ../../Gromacs/Replica$num/state.cpt .
     ln -s ../../Gromacs/Replica$num/gromacs.top topol.top
     ln -s ../../Gromacs/Replica$num/index.ndx .
     sed "s/XX/$name/"  ../gromacs_ramd.mdp > gromacs_ramd.mdp  # set a random seed
     gmx grompp -f gromacs_ramd.mdp  -c  gromacs$num.tpr -o gromacs_ramd.tpr -t state.cpt -n index.ndx -maxwarn  2

    gmx mdrun -s gromacs_ramd.tpr -ntmpi 1 -ntomp 16 -maxh 24 > gromac.output
     cd ..
  done
```

7. Analysis of simulation results

○ Extraction of dissociation times from Gromacs simulation output

When the set maximum distance between the centers of mass of the protein and the ligand is reached, the Gromacs-ramd execution will be stopped. The output file will contain a line starting with "GROMACS will be stopped after….". This line can be used to extract the dissociation time in each trajectory. This can be done for a set of trajectories generated from a single replica using the "grep" command, for example:

grep "GROMACS will be stopped after" TRJ1/*output > times_1.dat

here is an example of the file containing dissociation times for one replica:

```
ETRJ1-10/out:==== RAMD ==== GROMACS will be stopped after 691950 steps.
ETRJ1-11/out:==== RAMD ==== GROMACS will be stopped after 610200 steps.
```

```
ETRJ1-12/out:==== RAMD ==== GROMACS will be stopped after 155450 steps.
ETRJ1-13/out:==== RAMD ==== GROMACS will be stopped after 1067100 steps.
ETRJ1-14/out:==== RAMD ==== GROMACS will be stopped after 305550 steps.
ETRJ1-15/out:==== RAMD ==== GROMACS will be stopped after 1288150 steps.
ETRJ1-1/out:==== RAMD ==== GROMACS will be stopped after 1892500 steps.
ETRJ1-2/out:==== RAMD ==== GROMACS will be stopped after 755000 steps.
ETRJ1-3/out:==== RAMD ==== GROMACS will be stopped after 690000 steps.
ETRJ1-4/out:==== RAMD ==== GROMACS will be stopped after 542500 steps.
ETRJ1-5/out:==== RAMD ==== GROMACS will be stopped after 119000 steps.
ETRJ1-6/out:==== RAMD ==== GROMACS will be stopped after 962500 steps.
ETRJ1-7/out:==== RAMD ==== GROMACS will be stopped after 687500 steps.
ETRJ1-8/out:==== RAMD ==== GROMACS will be stopped after 408200 steps.
ETRJ1-9/out:==== RAMD ==== GROMACS will be stopped after 261650 steps.
```

○ Getting relative dissociation times for each starting replica

The time when 50% of the simulated trajectories have dissociated is defined as the effective RAMD residence time. To get better statistics, we use bootstrapping for each replica, which yields the mean and the standard deviation of the distribution of the effective RAMD residence times. The residence time distribution is expected to converge to a Gaussian distribution. If it does not converge, one can improve the statistics by running more trajectories. One can also use the Kolmogorov-Smirnov, KS, test (see Fig.5 in Supporting Information of Ref.[2]) to assess the quality of the statistics by the distance D between the Poisson distribution (with the computed τ) and the cumulative density function. One can use the Python script (https://github.com/DKokh/tauRAMD) to compute the residence time from each replica. This script can be also used for analysis of trajectories generated by NAMD software.

○ Checking statistics

A typical output of the tauRAMD.py script run for one protein-ligand complex with three replicas and 15 trajectories simulated for each replica is illustrated in Fig. 2. The KS test shows the maximum deviation of the simulated data from the Poisson distribution. Its value is usually between 0.1 and 0.2. [2]

Improving the statistics for a particular replica by increasing the number of trajectories is possible, but usually does not lead to a significant change in the residence time. In contrast, if the uncertainty of the final residence time is too large, the simulation of trajectories starting from additional replicas is usually helpful.

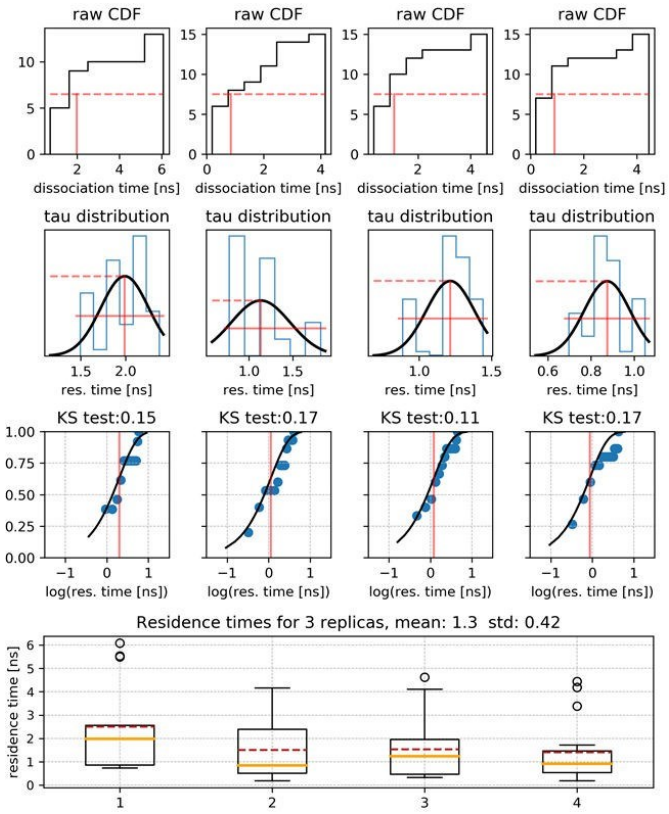

***Fig. 2**: Figure schematically illustrating the τRAMD protocol for computing the residence time of one protein-ligand complex using three replicas and generating 15 dissociation trajectories for each replica: **(first raw)** Cumulative distribution functions for the set of RAMD dissociation times for the three replicas the red line indicates the simulation time at which dissociation was observed in half of the trajectories (effective residence time for the raw data). **(second raw)** Distribution function of the effective residence times obtained after bootstrapping of the raw data along with the corresponding Gaussian distribution (black line), mean (τ_{repl}) and half-width are marked by red lines. **(third raw)** The Poisson cumulative distribution function (CDF, black line), $P = 1 - exp(-t/\tau_{repl})$, is compared with the empirical cumulative density function (ECDF, blue points) obtained from the raw data; τ_{repl} is indicated by the red line and the results of the Kolmogorov-Smirnov, KS, test are quantified by the supremum of the distance D between the Poisson and empirical CDFs. **(fourth raw)** Bar plots of the relative residence times obtained from each replica, the boxes extend from the lower to the upper quartile values of the data, the whiskers show the range of the data, outliers are shown by points, median and mean are shown by orange and dashed red lines, respectively. The residence time averaged over the replicas is shown in the title of the plot.*

○ Getting the final relative residence time

The final relative residence time of a ligand can be estimated as the mean residence time over all replicas. If the variation of the residence times of different replicas exceeds the computational uncertainty, it is recommended to run more replicas.

8. Modification of the protein-ligand system

If needed, one can modify the structures of interest, e.g. by deletion or addition of ligands using either ParmEd or cpptraj.

- To delete a ligand (defined by the residue name) using Python:

```
import parmed as pmd
p = pmd.load_file('ref.prmtop', 'ref-equal.rst')
s  = p['!:INH']
s.save('ref_noligand.prmtop')
#####**Note**: box parameters are not saved in the new prmtop file, but can be added - just search for the "%BOX" and
replace zeros by numbers from the original topology file (ref.prmtop)
```

or cpptraj:

```
parm ref.prmtop
trajin ref-equal.rst
strip ("strip :INH")
trajout ref-equal_noligand.rst
parmstrip ("strip :INH")
parmwrite out ref_noligand.ptmtop
```

- To add a ligand:

Prepare a mol2 file of the ligand with the correct coordinates and generate the topology (ligand.prmtop) and coordinate (ligand.inpcrd) files for the ligand using tleap. Add the ligand to the system using the ParmEd library in Python:

```
import parmed as pmd
p = pmd.load_file('CHL.prmtop','CHL.rst7') # load system
s1 = pmd.load_file('ligand.prmtop','ligand.inpcrd') # load ligand
b = p+s1
b.save('assembly_new.prmtop')
b.save('assembly_new.rst7')
```

References

(1) Kokh, D. B.; Kaufmann, T.; Kister, B.; Wade, R. C. Machine Learning Analysis of TRAMD Trajectories to Decipher Molecular Determinants of Drug-Target Residence Times. *Front. Mol. Biosci.* 2019, *6*, 36.

(2) Kokh, D. B.; Amaral, M.; Bomke, J.; Grädler, U.; Musil, D.; Buchstaller, H. P.; Dreyer, M. K.; Frech, M.; Lowinski, M.; Vallee, F.; et al. Estimation of Drug-Target Residence Times by τ-Random Acceleration Molecular Dynamics Simulations. *J. Chem. Theory Comput.* 2018, *14* (7), 3859–3869.

(3) Kokh, D. B.; Doser, B.; Richter, S.; Ormersbach F.; Chen, X.; Wade, R. C. Computational Workflow for Exploring Ligand Dissociation from Macromolecule: Efficient Random Acceleration Molecular Dynamics Simulation and Interaction Fingerprints Analysis of Ligand Trajectories. *Manuscript*, in preparation.