Molecular Dynamics Using GROMACS

`Moleculle` `Dynamics` `GROMACS` `Simulation`

Software: GROMACS: http://www.gromacs.org

Manual: http://www.gromacs.org/Documentation/Manual

Platform: Linux

Initial Protein Structure: cns1_model1.pdb

---

1 Check GROMACS

GROMACS location:

> /usr/local/gromacs/bin/

Set environment variable:

```
export PATH=$PATH:/usr/local/gromacs/bin/
#test if environment works
gmx -version
```

Result:

```
Aldert van Buuren  Rudi van Drunen    Anton Feenstra     Sebastian Fritsch
Gerrit Groenhof    Christoph Junghans Peter Kasson       Carsten Kutzner
Per Larsson        Justin A. Lemkul   Magnus Lundborg    Pieter Meulenhoff
Erik Marklund      Teemu Murtola      Szilard Pall       Sander Pronk
Roland Schulz      Alexey Shvetsov    Michael Shirts     Alfons Sijbers
Peter Tieleman     Christian Wennberg Maarten Wolf
and the project leaders:
Mark Abraham, Berk Hess, Erik Lindahl, and David van der Spoel

Copyright (c) 1991-2000, University of Groningen, The Netherlands.
Copyright (c) 2001-2014, The GROMACS development team at
Uppsala University, Stockholm University and
the Royal Institute of Technology, Sweden.
check out http://www.gromacs.org for more information.

GROMACS is free software; you can redistribute it and/or modify it
under the terms of the GNU Lesser General Public License
as published by the Free Software Foundation; either version 2.1
of the License, or (at your option) any later version.

GROMACS:      gmx, VERSION 5.0.7
Executable:   /home/ocean/Softwares/gromacs-5.0.7/build/bin//gmx
Library dir:  /home/ocean/Softwares/gromacs-5.0.7/share/top
Command line:
  gmx -version

Gromacs version:    VERSION 5.0.7
Precision:          single
Memory model:       64 bit
MPI library:        thread_mpi
OpenMP support:     enabled
GPU support:        disabled
invsqrt routine:    gmx_software_invsqrt(x)
SIMD instructions:  AVX_128_FMA
FFT library:        fftw-3.3.3-sse2
RDTSCP usage:       enabled
C++11 compilation:  enabled
TNG support:        enabled
Tracing support:    disabled
Built on:           2019年 01月 07日 星期一 11:24:41 CST
Built by:           root@clab4 [CMAKE]
Build OS/arch:      Linux 4.15.0-43-generic x86_64
Build CPU vendor:   AuthenticAMD
Build CPU brand:    AMD Ryzen 7 1700 Eight-Core Processor
Build CPU family:   23   Model: 1   Stepping: 1
Build CPU features: aes apic avx clfsh cmov cx8 cx16 f16c fma htt lahf_lm misalignsse mmx msr nonstop_tsc pclmuldq pdpe1gb popcnt pse rdrnd rdtscp sse2 sse3 sse4a sse4.
1 sse4.2 sse3
C compiler:         /usr/bin/cc GNU 7.3.0
C compiler flags:    -mavx -mfma4 -mxop    -Wno-maybe-uninitialized -Wextra -Wno-missing-field-initializers -Wno-sign-compare -Wpointer-arith -Wall -Wno-unused -Wunused
-value -Wunused-parameter  -O3 -DNDEBUG -fomit-frame-pointer -funroll-all-loops -fexcess-precision=fast  -Wno-array-bounds
C++ compiler:       /usr/bin/c++ GNU 7.3.0
C++ compiler flags:  -mavx -mfma4 -mxop   -std=c++0x  -Wextra -Wno-missing-field-initializers -Wpointer-arith -Wall -Wno-unused-function  -O3 -DNDEBUG -fomit-frame-poin
ter -funroll-all-loops -fexcess-precision=fast  -Wno-array-bounds
Boost version:      1.55.0 (internal)
```

So, it works.

---

2 cns1_model1 MD Simulation

2.1 protein pdb file preprocessing

1 . Remove the crystal waters in the pdb file

Use `grep` to delete these lines very easily:

```
grep -v HOH cns1_model1.pdb > cns1_clean.pdb
```

2.3 Molecule Dynamics Simulation

1 . Transfer pdb to gmx format, which contains:

> - The topology for the molecule.
> - A position restraint file.
> - A post-processed structure file.

```
gmx pdb2gmx -ignh -ff amber99sb-ildn -f cns1_clean.pdb -o cns1_clean.gro -p cns1_clean.top -water tip3p
```

- `-ignh`: Ignore all hydrogen atoms, because if H atoms are present, they must be in the named exactly how the force fields in GROMACS expect them to be. So dealing with H atoms can occasionally be a headache!
- `-ff`: We use Amber99SB-ILDN force field.
- `-f`: Specify the input file name
- `-o`: Specify the newly output GROMCAS file
- `-p`: Specify the newly ouput topology file, including the parameters of all atom-atom interactions.
- `-water`: Specify the mater model. We use TIP3P water model.

2 . Create the simulation box

We use `editconf` operation to create periodic simulation box.

```
gmx editconf -f cns1_clean.gro -o cns1_clean-PB.gro -bt cubic -d 1.2
#-bt dodecahedron (dodecahedron is probable better in other situations)
```

- `-f`: Input protein structure.
- `-o`: Output structure file with the simulation box information.
- `-bt`: Simulation box tyoe, default: rectangle box.
- `-d`: The minimum distance of the protein feom the X, Y, Z direction of the simulation box.

We use -bt option to create the rhombic dodecahedron, because its volume is ~71% of the cubic box of the same periodic distance, thus saving on the number of water molecules that need to be added to solvate the protein. `-d` option specifies the minimum distance of the molecule to the box margin, using nm as the unit. It specifies the size of the box. In theory, of the majority of the systems, `-d` can not be less than 0.9nm, so we use 1.2nm.

> We can check the structure of the protein with the added box. We need to transfer the gro format to pdb format using `editconf` operation

For example, we can convert the file.gro to file.pdb

```
gmx editconf -f file.gro -o file.pdb
```

3 . Energy minimization of the protein molecule in vaccum

We should design the specific .mdp file to specify parameters.

> em-vac-pme.mdp

```
; Definition delivered to preprocessor
define          = -DFLEXIBLE    ; Use flexible water model rather than rigid water model, thus steepest descent method will further minimize the energy

; Model type, Ending control, input control parameter
integrator      = steep         ; Algorithm (steep = steepest descent minimization)
emtol           = 500.0         ; Stop minimization when the maximum force < 1000.0 kJ/mol/nm
emstep          = 0.01          ; Minimization step size
nsteps          = 1000          ; Maximum number of (minimization) steps to perform
nstenergy       = 1             ; Energy writing frequency
```

```
energygrps      = System        ; Writing energy group

; Nearst list, interaction calculation parameter
nstlist         = 1             ; Frequency to update the neighbor list and long range forces
ns_type         = grid          ; Method to determine neighbor list (simple, grid)
coulombtype     = PME           ; Treatment of long range electrostatic interactions
rlist           = 1.0           ;
rcoulomb        = 1.0           ; Short-range electrostatic cut-off
vdwtype         = cut-off       ; Method for calculating van der waals interaction
rvdw            = 1.0           ; Short-range Van der Waals cut-off
constraints     = none          ; Set the restriction of the model
pbc             = xyz           ; Periodic Boundary Conditions in all 3 dimensions
```

Gromacs pre-processor (grompp) can integrate all information including simulated parameter, molecular structure, molecular structure, acquired force field parameter into a single binary file (tpr file), so it requires only tpr file while running `mdrun`.

```
gmx grompp -f em-vac-pme.mdp -c cns1_clean-PB.gro -p cns1_clean.top -o cns1_em-vac.tpr
```

- `-f` option specifies the input parameter file
- `-c` option specifies the input structure file
- `-p` option soecifies the input topology file
- `-o` option specifies the output file which is used for `mdrun`

After that, we get file cns1_em-vac.tpr and parameter file mdout.mdp.

We use `mdrun` intruction to run the energy minimization

```
gmx mdrun -v -deffnm cns1_em-vac
```

- `-deffnm` specifies default file name
- `-v` shows the infomation of the simulation process

After 1001 steps, result is :

```
Energy minimization reached the maximum number of steps before the forces
reached the requested precision Fmax < 500.

writing lowest energy coordinates.

Steepest Descents did not converge to Fmax < 500 in 1001 steps.
Potential Energy  = -6.9249328e+04
Maximum force     =  3.4546157e+03 on atom 755
Norm of force     =  4.1397404e+01

gcq#243: "Let's Unzip And Let's Unfold" (Red Hot Chili Peppers)
```

As a result, we get the record file `cns1_em-vac.log`, Binary full-precision trajectory `cns1_em-vac.trr`, Energy-minimized structure `cns1_em-vac.gro`.

If we only need the molecule dynamics simulation in vaccum, the above steps have got what we want, the `.trr` file, which can be depicted to movies using VMD software. However, if we need simulation in the solvent, water, for example, we have to go further.

4 . Infusing solvent and ion into the box and conduct energy minimization

We use `gmx solvate` to infuse the water into the box, simulating the solvation of the protein molecule.

```
1.  gmx solvate -cp cns1_em-vac.gro -cs spc216.gro -p cns1_clean.top -o cns
    1-b4ion.gro
```

- `-cp` specifies the model of the simulating protein box
- `-cs` specifies the water model to be SPC water model used for infilling
- `spc216` is the general 3-site water structure in GROMACS
- `-p` adjusts the topology file, adding the physical parameters of the corresponding water molecule
- `-o` specifies the output file after filling the water molecule

As a result, we get `cns1-b4ion.gro`.

Then we need to minimize the energy of the system, using another em-sol-pme.mdp file.

> em-sol-pme.mdp

```
define          = -DFLEXIBLE

integrator      = steep
emtol           = 250.0
nsteps          = 5000
nstenergy       = 1
energygrps      = System

nstlist         = 1
ns_type         = grid
coulombtype     = PME
rlist           = 1.0
rcoulomb        = 1.0
rvdw            = 1.0
constraints     = none
pbc             = xyz
```

Likewise, we need to use `grompp` to process the .mdp file, after which we start to conduct `mdrun`.

```
gmx grompp -f em-sol-pme.mdp -c  cns1-b4ion.gro -p cns1_clean.top -o ion.tpr
```

The generated `ion.tpr` is the result, and the output message is as follows:

```
Setting the LD random seed to 4222617473
Generated 2211 of the 2211 non-bonded parameter combinations
Generating 1-4 interactions: fudge = 0.5
Generated 2211 of the 2211 1-4 parameter combinations
Excluding 3 bonded neighbours molecule type 'Protein_chain_A'
Excluding 2 bonded neighbours molecule type 'SOL'

NOTE 3 [file cns1_clean.top, line 117939]:
  System has non-zero total charge: -3.999995
  Total charge should normally be an integer. See
  http://www.gromacs.org/Documentation/Floating_Point_Arithmetic
  for discussion on how close it should be to an integer.



Removing all charge groups because cutoff-scheme=Verlet
Analysing residue names:
There are:   792    Protein residues
There are: 270889      Water residues
Analysing Protein...
Number of degrees of freedom in T-Coupling group rest is 2475219.00
Calculating fourier grid dimensions for X Y Z
Using a fourier grid of 192x192x192, spacing 0.118 0.118 0.118
Estimate for the relative computational load of the PME mesh part: 0.22
This run will generate roughly 68 Mb of data

There were 3 notes

gcq#215: "I'm Not Gonna Die Here !" (Sphere)
```

We can see from the message that the total charge is -3.999995. Since the total charge should be described in integers, there are around -4e charge.

Then, we need to add the ion into the system, using the `ion.tpr` file. The ion we add should neutralise the charge original in the system. Since we have -4e charge in the system, we need to add more negion than cation.

We use `geion` operation to replace some water molecules by ions.

```
1.  gmx genion -s ion.tpr -o cns1-b4em.gro -neutral -conc 0.15 -p cns1_clea
    n.top
```

- `-neutral` option garantee the whole charge in the system equals zero, and the system is neutralized.
- `-conc` option sets the required ion concentration (here we use 0.15M)

- `-g` option specifies the name of the output record file
- `-norandom` option set the ion according to the electric potential, not random (default). However, here we use random set, so we do not use this option

Aftering executing this operation, it appears a consecutive solvent molecule group. Here we choose 13(SOL). Typing `Enter`, the procudure will tell your your solvent molecules have been replaced by Cl ion.

Inorder to run energy minimization by `mdrun`, we have to use the `grompp` operation again.

```
gmx grompp -f em-sol-pme.mdp -c cns1-b4em.gro -p cns1_clean.top -o cns1_em-sol.tpr
```

Then, we run energy minimization:

```
gmx mdrun -v -deffnm cns1_em-sol
```

We can see that after 5001 steps, it reaches convergence.

```
Energy minimization reached the maximum number of steps before the forces
reached the requested precision Fmax < 250.

writing lowest energy coordinates.

Steepest Descents did not converge to Fmax < 250 in 5001 steps.
Potential Energy  = -1.4573758e+07
Maximum force     =  1.5033184e+04 on atom 5593
Norm of force     =  2.1465935e+01

NOTE: 6 % of the run time was spent in domain decomposition,
      14 % of the run time was spent in pair search,
      you might want to increase nstlist (this has no effect on accuracy)
```

Note: `genion` uses NaCl in default. If users want any other negion and cation, they can use `-pname` (cation) and `-nname` (negion) to specify the name of the ions. Users can refer tothe `ion.itp` file to see how to call different ions

5 . Position-restricted pre-equalization simulation

We now need to construct position-restricted pre-equalization simulation, that is to say, we fix the position of the protein atoms and keep the solvent molecules moving, just like soaking the protein

molecules into the water, enabling the further equilibration of the system.

We need to perform two stages of position-restricted pre-equalizations: 100ps NVT and 100ps NPT, with the temperature of 310 K, which is similar to the experiment room temperature. It is appropriate to simulate in 310 K, since it is similar to the body temperature.

Likewise, we need another .mdp file called `nvt-pr-md.mdp` file.

```
define                  = -DPOSRES ; Tell GROMACS run position restricted siulation

integrator              = md
dt                      = 0.002 ; step length, the unit of which is ps, here we use 2fs
nsteps                  = 50000 ; number of steps, the simulation time is nsteps*dt

; output control parameter
nstxout                 = 500   ; Writing frequency, nstxout=500 and dt=0.002, so it writes out every 1 ps
nstvout                 = 500   ; Velocity saving frequency
nstenergy               = 500   ; Energy saving frequency
nstlog                  = 500   ; log file output frequency
energygrps              = Protein Non-Protein

;
nstlist                 = 5
ns_type                 = grid
pbc                     = xyz
rlist                   = 1.0

;
coulombtype             = PME
pme_order               = 4
fourierspacing          = 0.16
rcoulomb                = 1.0   ; unit is nm
vdw-type                = Cut-off
rvdw                    = 1.0

;
tcoupl                  = v-rescale
tc-grps                 = Protein  Non-Protein
tau_t                   = 0.1      0.1
```

```
ref_t                    = 300      300
;
DispCorr                 = EnerPres
;
pcoupl                   = no
;
gen_vel                  = yes
gen_temp                 = 300
gen_seed                 = -1
;
constraints              = all-bonds
continuation             = no
constraint_algorithm     = lincs
lincs_iter               = 1
lincs_order              = 4
```

Also, we need to use `grommp` and then `mdrun` again.

```
gmx grompp -f nvt-pr-md.mdp -c cns1_em-sol.gro -p cns1_clean.top -o cns1_nvt-pr.tpr
#gmx mdrun -deffnm cns1_nvt-pr
```

If mdrun requires so much time, users can use nohup to run the program background.

```
nohup gmx mdrun -deffnm cns1_nvt-pr &

#If you want to check the state of the program
tail -n 25 cns1_nvt-pr.log
```

NPT simulation uses `npt-pr-md.mdp` ass the parameter file

```
define                   = -DPOSRES

integrator               = md
dt                       = 0.002
nsteps                   = 50000

```

```
nstxout                  = 500
nstvout                  = 500
nstfout                  = 500
nstenergy                = 500
nstlog                   = 500
energygrps               = Protein Non-Protein

nstlist                  = 5
ns-type                  = Grid
pbc                      = xyz
rlist                    = 1.0

coulombtype              = PME
pme_order                = 4
fourierspacing           = 0.16
rcoulomb                 = 1.0
vdw-type                 = Cut-off
rvdw                     = 1.0

Tcoupl                   = v-rescale
tc-grps                  = Protein  Non-Protein
tau_t                    = 0.1      0.1
ref_t                    = 300      300

DispCorr                 = EnerPres

;
Pcoupl                   = Parrinello-Rahman
Pcoupltype               = Isotropic
tau_p                    = 2.0
compressibility          = 4.5e-5
ref_p                    = 1.0
refcoord_scaling         = com

gen_vel                  = no
constraints              = all-bonds
continuation             = yes
constraint_algorithm     = lincs
lincs_iter               = 1
lincs_order              = 4
```

We use the result .gro file as the input of the nvt simulation.

```
gmx grompp -f npt-pr-md.mdp -c cns1_nvt-pr.gro -p cns1_clean.top -o cns
```

```
1_npt-pr.tpr
nohup gmx mdrun -deffnm cns1_npt-pr &
#tail -n 25 cns1_npt-pr.log
```

6 . Final simulation

After equalization, we are able to conduct the final simulation

Still, we need another .mdp file

> npt-nopr-md.mdp

```
integrator               = md
dt                       = 0.002
nsteps                   = 500000 ; 1 ns

nstxout                  = 1000
nstvout                  = 1000
nstfout                  = 1000
nstenergy                = 1000
nstlog                   = 1000
energygrps               = Protein Non-Protein

nstlist                  = 5
ns-type                  = Grid
pbc                      = xyz
rlist                    = 1.0

coulombtype              = PME
pme_order                = 4
fourierspacing           = 0.16
rcoulomb                 = 1.0
vdw-type                 = Cut-off
rvdw                     = 1.0

Tcoupl                   = v-rescale
tc-grps                  = Protein  Non-Protein
tau_t                    = 0.1      0.1
ref_t                    = 300      300

DispCorr                 = EnerPres

Pcoupl                   = Parrinello-Rahman
```

```
Pcoupltype                = Isotropic
tau_p                     = 2.0
compressibility           = 4.5e-5
ref_p                     = 1.0

gen_vel                   = no

constraints               = all-bonds
continuation              = yes
constraint_algorithm      = lincs
lincs_iter                = 1
lincs_order               = 4
```

Coding:

```
gmx grompp -f npt-nopr-md.mdp -c cns1_npt-pr.gro -p cns1_clean.top -o cns1_npt-nopr.tpr
nohup gmx mdrun -deffnm cns1_npt-nopr &
#tail -n 25 cns1_npt-nopr.log
```

In order to save disk space, users can use `trjconv` operation to compress `.trr` trace file to `xtc` file, which is calculated faster in analysis. Also we should use `-pbc nojump` option to ensure all the atoms in the box.

```
gmx trjconv -f cns1_npt-nopr.trr -s cns1_npt-nopr.tpr -o cns1_npt-nopr.xtc -pbc nojump -ur compact -center
#Choose 0 in the two choice options. If the xtc file size is too large, choose 1 in thsese 2 options.
```

Note: If the resulting .trr file is to big, which is more than 10GB and even 20GB, you'd better alter the npt-nopr-md.mdp, changing the writing frequency (nstxout, nstvout, nstfout, nstenergy, nstlog) from 100 to 5000 or even 10000.

2.4 Simulation result analysis

Once we get the `.xtc` trace file, we'd like to analyse it.

2.4.1 Make an animation of the MD process

Instead of using `VMD` software, we choose Pymol alternatively, since latest VMD only support 32-bit program, which means that the max number of memory addresses is 2GB. When we used VMD, we could not even be able to operate 300M `xtc` trace file together with `.gro` structural file. Additionally, the movie made from Pymol is better than from VMD, and the UI is more friendly.

Firstly, we need to convert the `xtc` file and `gro` file together to `pdb` file, which can be processed by Pymol.

```
trjconv -s cns1_npt-nopr.gro -f cns1_npt-nopr.xtc -o movie.pdb
```

Then open `movie.pdb` in Pymol.

```
1. Pre-review the movie of the frames
PyMOL> mplay
Type "mstop" to stop the animation
2. Saving frames in PNG format
PyMOL> mpng frame
It will produce .npg files in Pymol directory
```

Then, use gif convertion software of linux operation to convert the npg files to gif format.

In linux, the operation is:

```
convert -delay 10 -loop 0 frame*.png movie.gif
```

`-delay 10`: the time interval between frames
`frame*.png`: the frames which compose the animation

In `VideoMach` software, load the `npg` format files and export the media in `gif` format.

2.4.2 Plot the parameter fluctuation

Use `gmx g_energy` operatio to extract the data

```
gmx energy -f cns1_npt-nopr.edr -o cns1_enrg-npt.xvg
```

It will appears some options of the data we would like to extract:

```
Select the terms you want from the following list by
selecting either (part of) the name or the number or a combination.
End your selection with an empty line or a zero.
-------------------------------------------------------------------
  1  Angle            2  Proper-Dih.      3  Improper-Dih.    4  LJ-14
  5  Coulomb-14       6  LJ-(SR)           7  Disper.-corr.    8  Coulomb-(SR)
  9  Coul.-recip.    10  Potential        11  Kinetic-En.     12  Total-Energy
 13  Temperature     14  Pres.-DC         15  Pressure        16  Constr.-rmsd
 17  Box-X           18  Box-Y            19  Box-Z           20  Volume
 21  Density         22  pV               23  Enthalpy        24  Vir-XX
 25  Vir-XY          26  Vir-XZ           27  Vir-YX          28  Vir-YY
 29  Vir-YZ          30  Vir-ZX           31  Vir-ZY          32  Vir-ZZ
 33  Pres-XX         34  Pres-XY          35  Pres-XZ         36  Pres-YX
 37  Pres-YY         38  Pres-YZ          39  Pres-ZX         40  Pres-ZY
 41  Pres-ZZ         42  #Surf*SurfTen    43  Box-Vel-XX      44  Box-Vel-YY
 45  Box-Vel-ZZ                           46  Coul-SR:Protein-Protein
 47  LJ-SR:Protein-Protein                48  Coul-14:Protein-Protein
 49  LJ-14:Protein-Protein                50  Coul-SR:Protein-non-Protein
 51  LJ-SR:Protein-non-Protein            52  Coul-14:Protein-non-Protein
 53  LJ-14:Protein-non-Protein            54  Coul-SR:non-Protein-non-Protein
 55  LJ-SR:non-Protein-non-Protein        56  Coul-14:non-Protein-non-Protein
 57  LJ-14:non-Protein-non-Protein        58  T-Protein
 59  T-non-Protein                        60  Lamb-Protein
 61  Lamb-non-Protein
```

Here, we'd like to extract the energy and temperature data, so we type `10 11 12 13` and key in `Enter` twice to execute the operation.

```
Last energy frame read 500 time 1000.000

Statistics over 500001 steps [ 0.0000 through 1000.0000 ps ], 4 data sets
All statistics are over 5001 points

Energy                      Average   Err.Est.       RMSD  Tot-Drift
-------------------------------------------------------------------------------
Potential                -1.6266e+07        550    4475.16   -3429.22  (kJ/mol)
Kinetic En.              2.91022e+06         37    2822.23    77.3609  (kJ/mol)
Total Energy             -1.33558e+07        540       5516   -3351.75  (kJ/mol)
Temperature                 299.995     0.0038   0.290924 0.00797177  (K)
```

The result will show in `cns1_enrg-npt.xvg`, and we can draw the picture using `Microsoft Office Excel`.

2.4.3 Alignment to the initial structure

Firstly, we need to convert the `cns1_npt-nopr.pro` to pdb file.

```
1.  gmx editconf -f cns1_npt-nopr.gro -o cns1_npt-nopr.pdb
```

Use `align` operation in Pymol to align the two structures

Aftering loading the structures into pymol software, we should remove the `waters` and `ions` of the MD result structure.

In pymol, on the right bar of cns1_npt-nopr, click `action|remove waters` to remove waters, and alert the selecting state of `Residue` on the bottom right to `Molecule`.

Click on the protein molecule and it appears an object called (sele), and click `action|extract object` and it created an object called obj01. Rename it to `cns1_MD`.

Load the initial structure file cns1_model1.pdb and we can align the two structures.

```
1.  align cns1_MD, cns1_model1
```

After alignment, we can get the RMSD value in the console.

If we want to make a movie of the 360-degree-view about the alignment, we need to use pymol console.

```
1.  PyMOL> mset 1 x360
2.  This command creates a movie with 60 frames
3.  PyMOL> util.mroll 1,360
4.  This command rotates the protein molecule 360 deg in 360 frames
5.  #the number of the frames can control the speed of the rotation, more
    frames means lower speed.
6.  PyMOL> mplay
7.  Type "mstop" to stop the animation
8.  2. Saving frames in PNG format
9.  PyMOL> mpng frame
```

It saves 360 frames `png` file in the python36 directory, use `VideoMach` to generate `gif` file.

2.4.4 Calculation of RMSD fluctuation

Use `gmx rms` to calculate the RMSD between the backbone after molecular dynamics simulation and the initial backbone.

```
gmx rms -s cns1_npt-nopr.tpr -f cns1_npt-nopr.xtc -o cns1-bkbone-rmsd.xvg
```

> Choose option 4 (backbone) when it alerts.

Use `gmx rmsdist` to calculate the RMSD of the distance fluctuation betwwen the atoms.

```
gmx rmsdist -s cns1_npt-nopr.tpr -f cns1_npt-nopr.xtc -o cns1_distrmsd.xvg
```

Likewise, use `Excel` to draw the picture.