# MD总结

石禹龙

终端SSH Shell，输入账号进行连接。

主机：172.21.12.138 sftp://172.21.12.138

端口号：22216

用户名：jawang

密码:P@ssw0rddddc

# linux基础操作：

查看或编辑文件：vi

未修改退出进程：q

修改后不保存退出：q!

修改后保存退出 ：wq

编辑文件夹：cd

编辑上一级：cd ../

强制退出：ctrl+c

打开文件：cat

拷贝文件：cp

拷贝上一级.in文件到此目录下：cp ../\*.in ./

拷贝同级.in文件到此目录下：cp ../同级文件名/\*.in ./

上上级：../../

跨服务器拷贝文件：scp -r jawang@192.168.98.209:~/yls/320k/\* ./ 需输入密码，不会显示。（不行的话，试试将209换成22209，09）

文件改名：mv 原文件名 新文件名

删除文件：rm+文件名

显示文件位置：pwd

后台执行命令：Nohup &（后台运行，用户退出也不受影响）

查看后台进程：top

多行删除：sed -i ‘1,10d’ filename

#文件夹改名mv ace-eth ace-eth1

--------------------------------

**2.VMD观察轨迹**

导入pdb文件；

#换成卡通模式

Graphics - representation - drawing method - NewCartoon-apply

#叠合

Extension -Analysis - RMSD calculator - Align

#画RMSD图

Extension -Analysis - RMSD Trajectory Tool

# 2.蛋白质模拟练习（GROMACS）

## 步骤及文件

1. 产生拓扑文件
2. 定义盒子和溶剂
3. 加离子
4. 能量最小化
5. 平衡
6. 运行MD
7. 分析

文件：

## 产生拓扑文件

#用Pymol的fetch + 蛋白质代号（PBD查到，以3nhe为例）得到蛋白质的pdb文件

mkdir yls #创建文件夹yls

cd yls

#去水(将3nhe.pdb文件导入到filezilla的ysl文件夹下)

grep -v HOH 3nhe.pdb>3NHE\_clean.pdb

gmx\_mpi pdb2gmx -f 3NHE\_clean.pdb -o 3NHE\_processed.gro

#选择力场（年份新，计算精确），选择水模型

AMBER99SB-ILDN protein, nucleic AMBER94 (Lindorff-Larsen et al., Proteins 78, 1950-58, 2010)

TIP3P TIP 3-point, recommended

## 定义盒子和溶剂

gmx.mpi editconf -f 3NHE\_processed.gro -o 3NHE\_newbox.gro -c -d 1.0 -bt cubic

#用溶剂（水）填充盒子

gmx.mpi solvate -cp 3NHE\_newbox.gro -cs spc216.gro -o 3NHE\_solv.gro -p topol.top

## 加离子

gmx\_mpi grompp -f ions.mdp -c 3NHE\_solv.gro -p topol.top -o ions.tpr

#加入Na+和Cl-使得溶剂变成中性。

gmx\_mpi genion -s ions.tpr -o 3NHE\_solv\_ions.gro -p topol.top -pname NA -nname CL -neutral

## 能量最小化

#导入mdp输入参数文件，通过GROMACS MD engine运行能量最小化

wget http://www.mdtutorials.com/gmx/lysozyme/Files/minim.mdp

gmx\_mpi grompp -f minim.mdp -c 3NHE\_solv\_ions.gro -p topol.top -o em.tpr

gmx\_mpi mdrun -v -deffnm em

#GROMACS 能量模型

gmx\_mpi energy -f em.edr -o potential.xvg

## 平衡

#导入mdp参数文件，NVT平衡

wget http://www.mdtutorials.com/gmx/lysozyme/Files/nvt.mdp

gmx\_mpi grompp -f nvt.mdp -c em.gro -r em.gro -p topol.top -o nvt.tpr

gmx\_mpi mdrun -deffnm nvt

#查看温度进程

gmx\_mpi energy -f nvt.edr

# NPT平衡

gmx\_mpi grompp -f npt.mdp -c nvt.gro -r nvt.gro -t nvt.cpt -p topol.top -o npt.tpr

gmx\_mpi mdrun -deffnm npt

#分析压力、密度进程

gmx\_mpi energy -f npt.edr -o pressure.xvg

gmx\_mpi energy -f npt.edr -o density,xvg

## 计算1ns MD

wget http://www.mdtutorials.com/gmx/lysozyme/Files/md.mdp

gmx\_mpi grompp -f md.mdp -c npt.gro -t npt.cpt -p topol.top -o md\_0\_1.tpr

gmx\_mpi mdrun -deffnm md\_0\_1

## 分析部分

#轨迹操作

gmx\_mpi trjconv -s md\_0\_1.tpr -f md\_0\_1.xtc -o md\_0\_1\_noPBC.xtc -pbc mol –center #去水去盒子

，方便VMD查看以及用cpptraj处理

#计算RMSD，相对于晶体结构

gmx\_mpi rms -s md\_0\_1.tpr -f md\_0\_1\_noPBC.xtc -o rmsd.xvg -tu ns

gmx\_mpi rms -s em.tpr -f md\_0\_1\_noPBC.xtc -o rmsd\_xtal.xvg -tu ns

#分析旋转半径

gmx\_mpi gyrate -s md\_0\_1.tpr -f md\_0\_1\_noPBC.xtc -o gyrate.xvg

gmx\_mpi make\_ndx -f em.gro -o index.ndx

#聚类分析，找出最具典型的构型，截断值为0.1

gmx\_mpi cluster -s md.tpr -f md.xtc -g -dist -sz -clid -cl -method linkage -cutoff 0.1

#Select group for least squares fit and RMSD calculation:

#Select a group: 4 backbone

#Select group for output:

#Select a group: 1 all

rm \*#

vi cluster.log

vi clusters.pdb

gmx\_mpi trjconv -s md1.tpr -f md1.xtc -o md1\_noPBC.xtc -pbc mol -center

which sander.MPI

gmx\_mpi rms -h #查看rmsd帮助

#

#

# 3. 蛋白质模拟练习（Amber）

## 建立初始结构及运行计算得到平衡系统

## 运行模拟并获取快照集合

## 3）计算结合自由能并分析结果

mkdir yls

cd yls

tleap

source oldff/leaprc.ff99

ras=loadpdb ras.pdb

raf=loadpdb raf.pdb

com=combine{ras raf} #将两个文件合并

solvatebox com TIP3PBOX 9 #常用的水模型

charge com #平衡电荷，一般为0，若不为0，需用钠或氯离子中和

saveamberparm ras ras.prmtop ras.inpcrd #储存参数

saveamberparm raf raf.prmtop raf.inpcrd

saveamberparm com com.prmtop com.inpcrd

quit #退出

wget <http://ambermd.org/tutorials/advanced/tutorial3/files/ras-raf.prmtop>

……

#得到[ras-raf.prmtop](http://ambermd.org/tutorials/advanced/tutorial3/files/ras-raf.prmtop), [ras-raf.inpcrd](http://ambermd.org/tutorials/advanced/tutorial3/files/ras-raf.inpcrd), [ras.prmtop](http://ambermd.org/tutorials/advanced/tutorial3/files/ras.prmtop), [ras.inpcrd](http://ambermd.org/tutorials/advanced/tutorial3/files/ras.inpcrd), [raf.prmtop](http://ambermd.org/tutorials/advanced/tutorial3/files/raf.prmtop), [raf.inpcrd](http://ambermd.org/tutorials/advanced/tutorial3/files/raf.inpcrd), [ras-raf\_solvated.prmtop](http://ambermd.org/tutorials/advanced/tutorial3/files/ras-raf_solvated.prmtop), [ras-raf\_solvated.inpcrd](http://ambermd.org/tutorials/advanced/tutorial3/files/ras-raf_solvated.inpcrd)文件。利用inpcrd（原子信息）和 prmtop（坐标位置）文件输出为相应的文件，使用ambpdb命令。

ambpdb -p ras.prmtop < ras.inpcrd > ras.pdb

ambpdb -p raf.prmtop < raf.inpcrd > raf.pdb

ambpdb -p com.prmtop < com.inpcrd > com.pdb

#生成了两个新的ras.pdb和raf.pdb文件。

vi min.in heat.in density.in #改变温度

#能量最小化（heating，density）

mpirun -np 16 sander.MPI -O -i min.in -p com.prmtop -c com.inpcrd -o min.out -r min.rst -ref com.inpcrd

mpirun -np 16 sander.MPI -O -i heat.in -p com.prmtop -c min.rst -x heat.mdcrd -o heat.out -r heat.rst -ref min.rst

mpirun -np 16 sander.MPI -O -i density.in -p com.prmtop -c heat.rst -x density.mdcrd -o density.out -r density.rst -ref heat.rst

#运行MD

vi md.in #使用i键进行插入修改数字，Ctrl+C退出，qw保存退出

heat comp

&cntrl

imin=0, irest=1, ntx=5,

nstlim=3000000, dt=0.001,

ntc=2, ntf=2,iwrap=1,

cut=9.0, ntb=2, ntp=1, taup=2.0

ntpr=1000, ntwx=1000,

ntt=3, gamma\_ln=2.0,

temp0=300.0

/

nohup pmemd.cuda -O -i md.in -p com.prmtop -c density.rst -o md.out -r md.rst -x md.crd &

#top查看后台运行，q退出，k是取消。通过vi md.out 查看剩余时间，若完成，再进行下一步. cd ../编辑上一级文件

vi mmgbsa.in

Input file for running PB and GB

&general

startframe=1000, endframe=1500, interval=10, keep\_files=0,

/

&gb

igb=5, saltcon=0.100

/

1,tleap #重新生成com-nw.prmtop(无水)

> source oldff/leaprc.ff99 导入立场文件

ras=loadpdb ras.pdb

raf=loadpdb raf.pdb

com=combine{ras raf} #将两个文件合并

saveamberparm com com-nw.prmtop com-nw.inpcrd

quit

2, cpptraj < trajin.in #得到md.pdb文件，**利用MMGBSA计算结合自由能**

MMPBSA.py -O -i mmgbsa.in -o FINAL-380k.dat -cp com-nw.prmtop -rp ras.prmtop -lp raf.prmtop -y md.pdb

MMPBSA.py -O -i mmgbsa.in -o FINAL-450k.dat -cp com-nw.prmtop -rp ras.prmtop -lp raf.prmtop -y md.pdb

# 4.算RMSD平均值

cpptraj

parm com.prmtop

trajin md.crd

rms first out rms.dat :242@C,CA,N

run

quit

vi rms.dat

#cat表示打开文件，|表示同时执行两个命令，NR表示行数,$2从第二行开始，最终求得RMSD平均值。

cat rms.dat|awk '{sum+=$2} END {print "average=", sum/NR}’

# 5.生成.pdb文件（.tpr .xtc/.gro）--gromacs

gmx\_mpi trjconv -s md.tpr -f md.xtc -o md.pdb -pbc mol -skip 10 -center

#完成后下载pdb文件，并用VMD查看。

#用gro文件和tpr文件得到pdb文件，gro文件是最后一个构象。

gmx\_mpi trjconv -f em.gro -s em.tpr -o em.pdb -pbc mol -ur compact

# 6.grep提取文件信息--linux

#将多个pbsa(pbsa1, pbsa2,pbsa3…)文件下的MMPBSA.dat中DELTA TOTAL所在行一起输出，-H打印文件名，-n打印行号

grep -Hn 'DELTA TOTAL' pbsa\*/MMPBSA.dat

#若想将结果输出至txt文件，则

grep -Hn 'DELTA TOTAL' pbsa\*/MMPBSA.dat > result.txt

#将多个txt文件中的x行y列数据一起输出至output1.txt中，（可删除$），只是x行，则NR==$x{print}

cat \*.txt| awk 'NR==$x{print $y}' >output1.txt

# 7.计算残基间距离或面积--gromacs

#计算单个残基间距离，例如计算26和282残基间距离

gmx\_mpi distance -f md.part0001.xtc -s md.tpr -select "com of resnr 26 plus com of resnr 282" -oall 26-282.xvg

#计算区间残基平均距离gmx\_mpi distance -f md.part0001.xtc -s md.tpr -select "com of resnr 26 to 28 plus com of resnr 282 to 300" -oall 26-282.xvg

vi 26-282.xvg

gmx\_mpi distance -f md.part0001.xtc -s md.tpr -select "com of resnr 26 plus com of resnr 300" -oall 26-300.xvg

[算面积]

gmx\_mpi distance -f md.part0001.xtc -s md.tpr -select "com of resnr 25 plus com of resnr 63" -oall 25-63.xvg

gmx\_mpi distance -f md.part0001.xtc -s md.tpr -select "com of resnr 63 plus com of resnr 302" -oall 63-302.xvg

gmx\_mpi distance -f md.part0001.xtc -s md.tpr -select "com of resnr 25 plus com of resnr 302" -oall 25-302.xvg

paste 25-63.xvg 25-302.xvg 63-302.xvg|awk '{print($2,$4,$6)}' > input.dat

python s.py

#s.py内容：

import numpy as np

#import pandas as pd

# from scipy.stats.stats import pearsonr

# data=pd.read\_csv('input-abc.dat',sep=' ')

def square(a,b,c):

p=0.5\*(a+b+c)

q=p\*(p-a)\*(p-b)\*(p-c)

return np.sqrt(q)

#f=open('test2.dat','w+')

#f.write(square(a,b,c))

fh = open("input-abc.dat")

fw = open("result.txt",'w')

for line in fh:

a,b,c=map(float,line.strip().split())

result = square(a,b,c)

fw.write("%s\n"%result)

print( "finished")

paste 25-63.xvg result.txt|awk '{print($1,$3)}' > area-ildn.dat

#将area-ildn.dat用origin打开，作图观察面积-时间分布情况

vi 26-300.xvg #第一列为时间，第二列为距离。删除前几行文字，直至第一行为数据行，同理处理其他xvg。

paste 26-300.xvg 26-282.xvg|awk '{print($2,$4)}' > input.dat

vi input.dat

#查看input.dat时，按D删除上面几行直至两列数字处，:wq保存退出

#按距离从大到小排列，上面是最大值，shift+g 最下面是最小值

sort -rn -k1 input.dat|less

sort -rn -k2 input.dat|less

# 8.绘制能量图

承上：

sort -rn -k2 input.dat|wc -l #报出总行数

vi rmsdrgpop.f #记录距离数值上限与下限，在rmsdrgpop.f中修改距离范围

gfortran --free-form rmsdrgpop.f #得到a.out文件，chmod +x a.out授予权限

./a.out

#输入总行数以及间隔

#25001

#0.05

#0.05

ls

#得到output.pop文件，距离1，距离2，能量值。可将此文件下载到本地，拖到origin，将能量值设置为Z轴，选中X，Y，Z进行plot，XYZcontour

#取出最大口袋图

awk '$1>0.9 && $2>0.7 {print($2)}' input.dat > test.dat #筛选出距离1大于0.9，距离2大于0.7的构象

vi test.dat #找出距离最大的距离

vi 26-282.xvg #找出最大距离对应的时间

#用：/+数字，可快速查找，可得到时间为523732.000，输出为pdb，用pymol打开。

gmx\_mpi trjconv -f md-nw.xtc -s md.tpr -pbc mol -ur compact -b 523732.000 -e 523732.000 -o 523732.pdb

# 9.伞取样（for）--gromacs

准备文件：npt.gro smd\*.dat md.mdp

制作mdp文件：

#for语句批量处理

for i in `seq 0 31`;do sed -i '4,4d' smd.${i}.dat;done #将序号0-31共32个smd.dat删去第四行

#再在这32个文件依次添加下面引号内内容

sed -i '4i\restraint-distance: RESTRAINT ARG=dist KAPPA=6000 AT=2.64' smd.0.dat

……

sed -i '4i\restraint-distance: RESTRAINT ARG=dist KAPPA=6000 AT=0.40' smd.31.dat

for i in `seq 0 31`;do gmx\_mpi grompp -f md.mdp -c npt.gro -p topol.top -o smd${i}.tpr -maxwarn 1;done #由md.mdp生成每一个伞取样的smd.tpr文件。

nohup mpirun -np 32 gmx\_mpi mdrun -v -deffnm smd -plumed smd.dat -multi 32 -replex 1000 &

查看文件smd0.gro完成时间：

ls -lhtr smd0.gro

-----------------------------

for i in `seq 0 27`;do cp smd.${i}.dat md.${i}.dat;done #类似于改名

#??

gmx\_mpi solvate -cp box.gro -cs -o box\_water.pdb -p topol.top

gmx\_mpi grompp -f em.mdp -p topol.top -c box\_water.pdb -o ions.tpr -maxwarn 1

gmx\_mpi genion -s ions.tpr -p topol.top -o box\_ions.pdb -neutral -conc 0.1

gmx\_mpi grompp -f em.mdp -p topol.top -c box\_ions.pdb -o em.tpr -maxwarn 1

gmx\_mpi mdrun -ntomp 32 -v -deffnm em

gmx\_mpi grompp -f npt.mdp -c em.gro -p topol.top -o npt.tpr -maxwarn 1

gmx\_mpi mdrun -ntomp 32 -v -deffnm npt

---------------------------

# 10.聚类分析--gromacs

#先计算三个残基组成的面积，记为area.xvg

gmx\_mpi trjconv -f md-100ns.xtc -s md.tpr -dropunder 0.50 -o t1.pdb -pbc mol -ur compact -drop area.xvg #删去area.xvg中面积小于0.50的构象

gmx\_mpi cluster -h #查看帮助

gmx\_mpi cluster -f t1.pdb -s md.tpr -cl -cutoff 0.2

vi cluster.log #查看共有几个聚类，选择其中构象数目最多的。

vi clusters.pdb #留下选中的MODEL，删去其他的聚类。用pymol打开，查看此构象

# 11.续跑分子动力学--gromacs

ls -lhtr \* #查看文件完成时间

mkdir test #新建文件夹

cd test

gmx\_mpi convert-tpr -s ../md1.tpr -extend 150000 -o md.tpr #在md1.tpr结束时间上继续跑150ns，得到新的md.tpr文件。

nohup gmx\_mpi mdrun -s md.tpr -cpi ../md.cpt -noappend & #-cpi续跑

top

vi md.part0002.log #查看时间节点，验证是否成功

# 12.分子探针MD

#在pachmol软件上生成混合体系（准备小分子的pdb文件）

#USP2水溶液环境中添加乙酸和异丙醇小分子

mkdir ace-eth

cd ace- eth

vi protein.pdb

gmx\_mpi pdb2gmx -f protein.pdb -ignh -o conf.pdb

vi topol.top

vi ligand-all.pdb

cat ligand-all.pdb >> conf.pdb

vi conf.pdb

gmx\_mpi editconf -f conf.pdb -o box.gro -c -d 1.0 -princ

gmx\_mpi solvate -cp box.gro -cs -o box\_water.pdb -p topol.top

gmx\_mpi grompp -f em.mdp -p topol.top -c box\_water.pdb -o ions.tpr -maxwarn 1

gmx\_mpi genion -s ions.tpr -p topol.top -o box\_ions.pdb -neutral -conc 0.1

gmx\_mpi grompp -f em.mdp -p topol.top -c box\_ions.pdb -o em.tpr -maxwarn 1

gmx\_mpi mdrun -ntomp 32 -v -deffnm em

gmx\_mpi grompp -f npt.mdp -c em.gro -p topol.top -o npt.tpr -maxwarn 1

gmx\_mpi mdrun -ntomp 32 -v -deffnm npt

gmx\_mpi grompp -f md.mdp -c npt.gro -p topol.top -o md.tpr -maxwarn 1

nohup gmx\_mpi mdrun -v -deffnm md -noappend &

top

vi ligand-all.pdb history #？？

top

gmx\_mpi trjconv -f md.trr -s md.tpr -o tmp.gro -pbc mol -ur compact <<eof

1

#生成有配体的pbd文件：

gmx\_mpi trjconv -f md.part0001.xtc -s md.tpr -b 863830.000 -e 863830.000 -n index.ndx -o 863830.pdb -pbc mol -ur compact -center # -ur compact更加紧凑。选择有配体和受体的groop。

#用gro文件和tpr文件得到pdb文件，gro文件是最后一个构象。

gmx\_mpi trjconv -f em.gro -s em.tpr -o em.pdb -pbc mol -ur compact

# 13.做b-factor图（RMSF）--gromacs

#先计算rmsf值，反映蛋白质振动情况

gmx\_mpi rmsf -s md.tpr -f md.part0001.xtc -o rmsf\_2.xvg #后缀-res，则残基与rmsf一一对应

#将rmsf\_2.xvg和有关的pdb（conf.pdb）文件下载至桌面，用UltraEdit打开这两个文件，将rmsf\_2.xvg中的每个蛋白原子rmsf值替换掉conf.pdb最后一列均为0的值，再将conf.pdb文件用pymol打开，点击b-factor查看。

# 14.计算SASA可溶剂化表面积—gromacs

**（如计算目标C22的SASA值）**

#先做index.ndx文件，输入文件为有关的pdb文件，选择groop时，按q直接保存退出，就得到了完整的index.ndx文件。

ls -lhtr \*.pdb #可列出所有的pdb文件，并提示其大小，创作时间

gmx\_mpi make\_ndx -f conf.pdb -o index.ndx

vi md.gro #查看C22所在处原子编号，为312-322

22CYS N 312 -0.065 0.574 5.398 0.4958 -0.4448 0.3117

22CYS H 313 0.014 0.580 5.335 1.8706 -1.3949 1.8394

22CYS CA 314 -0.202 0.576 5.359 -0.3091 0.0387 0.3132

22CYS HA 315 -0.251 0.656 5.416 0.1098 -0.0676 0.8309

22CYS CB 316 -0.208 0.609 5.206 -0.7941 0.2495 -0.6416

22CYS HB1 317 -0.313 0.601 5.176 -0.6607 -1.9533 -0.6896

22CYS HB2 318 -0.156 0.538 5.142 -3.9300 -0.6633 -2.4139

22CYS SG 319 -0.145 0.774 5.179 -0.2625 0.6394 0.1975

22CYS HG 320 -0.023 0.734 5.217 -0.3273 1.3306 1.1730

22CYS C 321 -0.271 0.439 5.393 0.2792 0.0425 0.3460

22CYS O 322 -0.379 0.435 5.449 0.5984 0.6129 -0.0404

vi index.ndx #在末端insert添加目标残基原子序列

[cys]

312 313 314 315 316 317 318 319 320 321 322

:wq

gmx\_mpi sasa –h #查看所需文件

gmx\_mpi sasa -f md.part0001.xtc -s md1.tpr -n index.ndx -o area.xvg

#选择[cys]所在选项

vi area.xvg

#计算第二列SASA值平均值

cat area.xvg | awk '{sum2+=$2;count++}END{print sum2/count}'

#取SASA最大的构象进行分析

sort -rn -k2 area.xvg|less

gmx\_mpi trjconv -f md.part0001.xtc -s md.tpr -b 863830.000 -e 863830.000 -n index.ndx -o 863830.pdb -pbc mol -ur compact –center # -ur compact更加紧凑。选择有配体和受体的groop。

# 15.分子动力学—gromacs

#准备好md.mdp, npt.gro文件

#选力场

gmx\_mpi trjconv -f npt.gro -s md.tpr -o npt.pdb -pbc mol -ur compact

#得到conf.pdb

gmx\_mpi pdb2gmx -f npt.pdb -ignh -o conf.pdb

【#若有残基命名不符合力场规则，则查看相应力场并修改残基名称。

which gmx\_mpi

ls

cd ~/software/gmx514-plumed/share/gromacs/top

vi charmm36-jul2017.ff

vi merged.hdb

#若同一个残基有多个形态名称，则修改后并下载conf.pdb查看该残基是否与正常情况一致。】

gmx\_mpi editconf -f conf.pdb -o box.gro -c -d 1.0 -princ

gmx\_mpi solvate -cp box.gro -cs -o box\_water.pdb -p topol.top

vi em.mdp

gmx\_mpi grompp -f em.mdp -p topol.top -c box\_water.pdb -o ions.tpr -maxwarn 1

gmx\_mpi genion -s ions.tpr -p topol.top -o box\_ions.pdb -neutral -conc 0.1

gmx\_mpi grompp -f em.mdp -p topol.top -c box\_ions.pdb -o em.tpr -maxwarn 1

gmx\_mpi mdrun -ntomp 32 -v -deffnm em

vi npt.mdp

gmx\_mpi grompp -f npt.mdp -c em.gro -p topol.top -o npt.tpr -maxwarn 1

gmx\_mpi mdrun -ntomp 32 -v -deffnm npt

rm \*#

gmx\_mpi make\_ndx -f npt.gro -o index.ndx

#q

vi npt.pdb #PyMol查看感兴趣的残基，再在npt.pdb中找出相应的原子序号

vi index.ndx

#在index.ndx中建立[Pocket]组，加入感兴趣的原子序号，并将非[Pocket]的原子序号补充加入[non-protein]

vi md.mdp

gmx\_mpi grompp -f md.mdp -c npt.gro -p topol.top -o md.tpr -maxwarn 1 -n index.ndx

nohup gmx\_mpi mdrun -v -deffnm md -noappend &

vi md.part0001.log

# 16.补足原子信息—amber

tleap

source oldff/leaprc.ff03

pro=loadpdb 3nhe.pdb

saveamberparm pro protein.prmtop protein.inpcrd

quit

生成pdb

ambpdb -p protein.prmtop < protein.inpcrd > protein.pdb

# 17.分子对接—薛定谔

22209,22149上有薛定谔软件，maestro

# 18.残基相关分析—bio3d

vi trajin.in

# trajin.in内容，得到.dcd文件

parm 1.pdb

trajin md-100ns.pdb 1 100000 15

trajout 1.dcd

#

head -8000 md-100ns.pdb > 1.pdb

vi 1.pdb

cpptraj < trajin.in #执行trajin.in

#得到1.pdb和1.dcd

R语言处理数据得到残基相关分析图

library(bio3d)

dcd <- read.dcd("C:/Users/sissichan/Desktop/1.dcd")

pdb <- read.pdb("C:/Users/sissichan/Desktop/1.pdb")

ca.inds <- atom.select(pdb, elety="CA")

##ca.inds <- atom.select(pdb, "backbone")

xyz <- fit.xyz(fixed=pdb$xyz, mobile=dcd, fixed.inds=ca.inds$xyz, mobile.inds=ca.inds$xyz)

dim(xyz) == dim(dcd)

###rmsd

rd <- rmsd(xyz[1,ca.inds$xyz], xyz[,ca.inds$xyz])

plot(rd, typ="l", ylab="RMSD", xlab="Frame No.")

points(lowess(rd), typ="l", col="red", lty=2, lwd=2)

hist(rd, breaks=40, freq=FALSE, main="RMSD Histogram", xlab="RMSD")

lines(density(rd), col="black", lwd=3)

###rmsf

rf <- rmsf(xyz[,ca.inds$xyz])

plot(rf, ylab="RMSF", xlab="Residue Position", typ="l")

###pca

pc <- pca.xyz(xyz[,ca.inds$xyz])

plot(pc, col=bwr.colors(nrow(xyz)) )

hc <- hclust(dist(pc$z[,1:2]))

grps <- cutree(hc, k=2)

plot(pc, col=grps)

p1 <- mktrj.pca(pc, pc=1, b=pc$au[,1], file="pc1.pdb") #输出的pc1.pdb用pymol打开，点b-factor图，保存为gif或mpeg动态图。

p2 <- mktrj.pca(pc, pc=2,b=pc$au[,2], file="pc2.pdb")

###dccm

cij<-dccm(xyz[,ca.inds$xyz])

plot(cij,col.regions=bwr.colors(15))

###

pymol.dccm(cij, pdb, exefile="D:/pymol/PyMOL/PymolWin.exe",type="launch")

#软件启动pymol，查看残基相关分析，找相关系数最正和最负的两组图。

# 19.济南超算

<https://60.208.139.60/>

40.0.0.22 22 SSH gfssimm swetestuesr1

#每个目标化合物跑100ns 2D-REMD，分十次，每次10ns，以防中途停止。

mkdir test3

cd test3

cp ../ test2/\*mpi ./

cp ../test2/\*.sh ./

# tail -1 run.1.sh 查看该命令作用

vi run1.sh #修改参数，从test2继续10ns进程，需在test3修改run1.sh文件

1, $s/test1/test2/g #g表示全部修改

1, $s/mdnew/md/g #test2的文件为md\*

./run1.sh 4 #用4个核，若报错，ctrl+C 终止该项目

rm –rf \*# #删除备份文件

vi md.sh #查看采用了多少个副本，注意N（核）与n（节点）的区别。必要时可调整计算时间

bqueues –l q\_sw\_share #可查看剩余多少核，idle nodes number is

nohup ./md.sh 192 & #./表示执行命令，运算核数需为N的偶数倍数。

bjobs #类似于top，查看后台进程，需杀死后台文件：bkill+序号

vi md12.part0001.log #可查看具体的开始时间节点

下一步建立test4

进程终止后继续进行22209：

mkdir continue/

nohup gmx\_mpi mdrun -v -deffnm md -noappend -cpi ../md.cpt -s ../md.tpr &

--------------------------------------------------------------------------------------