1. 查看计算机CPU、内存信息：  
   cat /proc/cpuinfo  
   cat /proc/meminfo
2. 查看Linux类型信息：  
   cat /proc/version  
   cat /etc/issue
3. 打印text.txt文件里包含“1234”字符串的行：  
   cat text.txt | sed -n ‘/1024/p’  
   -n: 只处理匹配的行，默认处理所有的行;  
   p: 执行打印操作，另外d表示删除操作。
4. 用sed进行字符替换：  
   cat filename | sed ‘s/string\_old/string\_new/’ > newfile  
   或者  
   sed ‘s/string\_old/string\_new/’ <filename > newfile
5. 寻找并复制：  
   find . -name “md\*.xvg” **-exec** cp **{}** xvg/ **\;**  
   或者  
   find . -name “md\*.xvg” **| xargs** **-i** cp **{}** xvg/
6. Python里面用eval将对字符串进行求值：  
   a = “[1,2]\n”  
   a  
   “[1,2]\n”  
   b = eval(a)  
   b  
   [1,2]
7. Python里面使用os模块执行Bash相关命令：  
   import os  
   os.system(“ls”)  
   os.listdir(os.getcwd())
8. grep选择不匹配的行  
   grep -v ‘^HETATM’ 6PTI.pdb > 6PTI\_protein.pdb
9. 使用VMD来叠合同一蛋白的两个不同构象，二者必须要有相同数目的原子：  
   set sel0 [atomselect 0 all]  
   set sel1 [atomselect 1 all]  
   set M [measure fit $sel0 $sel1]  
   $sel0 move $M
10. 正则表达式基础：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 常用的元字符 | | 常用的限定符 | | 常用的反义代码 | | 懒惰限定符 | |
| . | 匹配除换行符以外的任意字符 | \* | 重复零次或更多次 | \W | 匹配任意不是字母、数字、下划线或汉字的字符 | \*？ | 重复任意次，但尽可能少重复 |
| \w | 匹配字母、数字、下划线或汉字 | + | 重复一次或更多次 | \S | 匹配任意不是空白符的字符 | +？ | 重复1次或更多次，但尽可能少重复 |
| \s | \s匹配任意的空白符，包括空格，制表符(Tab)，换行符，中文全角空格等 | ？ | 重复零次或一次 | \D | 匹配任意非数字字符 | ?? | 重复0次或1次，但尽可能少重复 |
| \d | 匹配数字 | {n} | 重复n次 | \B | 匹配不是单词开头或结束的位置 | {n,m}? | 重复n到m次，但尽可能少重复 |
| \b | 匹配单词的开始或结束 | {n,} | 重复n次或更多次 | [^x] | 匹配除x以为的任意字符 | {n,} | 重复n次以上，但尽可能少重复 |
| ^ | 匹配字符串的开始 | {n,m} | 重复n到m次 | [^aeiou] | 匹配除aeiou以外的任意字符 |  |  |
| $ | 匹配字符串的结束 | [aeiou] | 匹配任何一个元音字母 |  |  |  |  |
|  |  | [0-9] | 与\d相同，匹配一个数字 |  |  |  |  |

1. TCL 文件操作：  
   set file [open text.txt r] #以只读模式打开文件  
   foreach line [split [read $file] \n] {puts $line} #read会一次性读入所有的内容，包  
    #换行符\n；使用split进行拆分，然  
    #后打印每一行的内容。  
   tell $file #显示当前文件指针所在的位置  
   seek $file 0 #将文件指针放到文件开始处，默认  
    #的origin是start，还有current和end。  
   close $file #关闭文件  
   写文件  
   set chan [open my.log a]  
   set timestamp [clock format [clock seconds]]  
   puts $chan "$timestamp - Hello, World!"  
   close $chan
2. TCL字符串操作：  
   set text “TCL”  
   string range $text 0 2 #返回字符串的前三位“TCL”  
   string equal $text “TCL” #比较两个字符串是否相同，若相同返回  
    #1，不同返回0  
   string trim $text L #返回TC，trim会返回除去首尾的chars  
    #字符串，默认为除去两端的空格
3. 比较两个文件的不同：  
   diff a.txt b.txt  
   cmp a.txt b.txt
4. VMD TCL查看总带电量：  
   set sel [atomselect top all]  
   eval vecadd [$sel get charge]
5. PDC查看账户信息：  
   Lindgren: projinfo [-u guanglin]  
   Ferlin: module add easy→cac members guanglin→cac examine prj.1206-065
6. Use **catdcd** to **reduce** the size of a NAMD trajectory file:  
   catdcd -o sample\_SMALL.dcd -stride 10 sample.dcd  
   This command extracts the trajectory file every 10 frame.  
   Use catdcd to **combine** two trajectory files:  
   catdcd -o sample\_combine.dcd sample1.dcd sample2.dcd  
   Use catdcd to print the number of frames in the trajectory file:  
   catdcd -num sample.dcd  
   Note:  
   catdcd is a pre-combiled script and can be executed directly.
7. 由bash切换到tcsh shell：  
   tcsh 或者/etc/tcsh  
   查看当前shell：  
   echo $0 或者ps | grep $$ | awk ‘{print $4}’  
   注意：  
   echo $SHELL查看的系统默认的shell
8. Redirecting Standard Output and Standard Error  
   Bash allows both the standard output (file descriptor 1) and the standard error output (file descriptor 2) to be redirected to the file whose name is the expansion of word with this construct.  
   There are two formats for redirecting standard output and standard error:  
   **&>word** and >&word  
   Of the two forms, the first is preferred. This is semantically equivalent to  
   >word 2>&1
9. EXCEL把X轴坐标轴放在最下面：设置“主要纵坐标轴”——横坐标轴交叉——坐标轴值——“-4000”就是以（0,-4000）为坐标原点，-4000是这个折线图上Y轴最小值。
10. BASH For循环：  
    #!/bin/bash  
    for ((c=1; c<=5; c++))  
    do  
    echo $c  
    done
11. Kill all running jobs on Ferlin:  
    **spq** -u guanglin | **awk** '{if (NR!=1) print $2}' | **xargs** -i sprelease {}
12. PyMol显示小分子球棍模型：  
    set stick\_radius,0.2  
    set sphere\_scale,0.25
13. PyMol 显示双键和三键

set valence, 1

1. awk读取输入记录，一次读取一条记录（行），然后自动将各个记录切分为字段，awk将每条记录内的字段数目，存储到内建变量NF。  
   默认以空白分隔字段，例如空格与制表字符。可以将FS变量设置为一个不同的值，就可以变更awk分隔字段的方式。可以使用-F选项修改分隔字段，-F选项会自动设置FS变量。  
   awk的输入、输出分隔字符用法是分开的，默认awk的输出字段是以一个空格来分隔的，可以设置OFS变量，改变输出字段分隔字符。  
   在命令行里使用-v选项，可以设置awk变量。  
   awk ‘{print $1}’ 打印第1个字段  
   awk ‘{print $1, $NF}’ 打印第1个和最后一个字段，不使用逗号两个字段将打印在一起。  
   awk ‘ NF > 0 {print $0}’打印非空行，$0表示整条记录。  
   awk -F: ‘{print $1, $5}’ /etc/passwd

awk求和：  
cat text.txt | awk '{SUM+=$2} END { print SUM }'

1. sed命令基本用法：  
   使用s命令，要求正则表达式寻找，用替代文本替换匹配的文本。“/”是最常用的定界符，但任何可现实的字符都能作为定界符，通常都会以标点符号作为定界符，如分好、冒号或逗号。  
   在s命令里以g结尾表示全局性，如果没有设置g，sed只会取代第一个匹配的。  
   sed ‘s/:.\*//’ /etc/passwd | sort -u 删除第一冒号之后的所有东西，排序列表并删除重复部分。  
   -n选项修改了sed的默认行为，当提供此选项时，sed将不会自动在操作完成后打印，若在脚本里使用p，则会打印指定的行：  
   sed -n ‘/<HTML>/p’ \*.html 仅显示包含<HTML>的行。  
   可指定打印行的范围，用逗号将地址隔开，符号$指最后一行：  
   sed -n ‘2,$p’ kgl.html 打印第二行到最后一行。  
   sed 在一行中使用多个命令时，一定要加-e 命令。  
   - 描点(anchor)  
   用以标 识 于句子中的位置所在. 常见有:  
   ^: 表示句首. 如 ^abc 表示以 abc 为首的句子.  
   $: 表示句尾. 如 abc$ 表示以 abc 结尾的句子.wqq  
   \<: 表示词首. 如 \<abc 表示以 abc 为首的詞.  
   \>: 表示词尾. 如 abc\> 表示以 abc 結尾的詞.  
   sed ‘/^$/d’ text.txt 删除空行  
   $ sed '/test/d' example-----删除example文件所有包含test的行
2. Linux BASH 命令替换  
   a. 使用反引号：  
   echo `cd ~; ls`  
   b. 使用$(…)的格式  
   echo $(cd ~; ls)
3. Linux单引号和双引号  
   单引号强制Shell将一对引号之间的所有字符都看做起字面上的意义；  
   双引号会确切地处理括起来的文字中的转义字符和变量、算术、命令替换。
4. Linux算术运算及条件测试  
   算术运算：expr和$((…))，expr后面的算术表达式必须以空格分开。  
   条件测试：test和[…], 在[]里面，条件表达式两端空格不能少。  
   例子：  
   i=1  
   while [ “$i” -le 5] 或者test “$i” -le 5  
   do  
    echo i is $i  
    i=`expr $i + 1` 或者i=$((i+1))  
   done
5. gzip是个使用广泛的压缩程序，文件经它压缩过后，其名称后面会多出".gz"的扩展名。  
   解压：  
   gzip -d 或者 gunzip
6. PyMol 显示球棍模型：  
   首先选中要显示的分子，将其显示为sticks和spheres，然后调整stick和sphere的显示比例分别为0.15和0.2.  
   set stick\_radius,0.2  
   set sphere\_scale,0.25
7. NAMD bug:

|  |
| --- |
| ------------- Processor 0 Exiting: Called CmiAbort ------------  Reason: FATAL ERROR: DIDN'T FIND vdW PARAMETER FOR ATOM TYPE OCL  FATAL ERROR: DIDN'T FIND vdW PARAMETER FOR ATOM TYPE OCL  解决办法： 将toppar\_water\_ions.str里面的NBFix相关项注释掉。  !KGL SOD OCL -0.075020 3.190 ! For lipid carboxylate groups  !KGL SOD OC2D2 -0.075020 3.190 ! For carb carboxylate groups  这是针对脂和糖的力场参数，需要导入相应的力场。 |

1. Kill all jobs on AKKA:  
   qselect -u $USER | xargs qdel

<http://docs.notur.no/uit/stallo_documentation/faq/how-can-i-kill-all-my-jobs>

1. 用wget下载文件：  
   wget “[http://www.bevanlab.biochem.vt.edu/Pages/Personal/justin/gmx-tutorials/membrane\_protein/Files/ions.mdp” -O ions.mdp](http://www.bevanlab.biochem.vt.edu/Pages/Personal/justin/gmx-tutorials/membrane_protein/Files/ions.mdp)  
   wget “http://www.rcsb.org/pdb/files/2RH1.pdb”
2. Node、CPU、Core、Processor的关系：

|  |
| --- |
| NUMA(Non-Uniform Memory Access，非一致性内存访问)和SMP(Symmetric Multi-Processor，对称多处理器系统)是两种不同的CPU硬件体系架构。  SMP的主要特征是共享，所有的CPU共享使用全部资源，例如内存、总线和I/O，多个CPU对称工作，彼此之间没有主次之分，平等地访问共享的资源，这样势必引入资源的竞争问题，从而导致它的扩展内力非常有限。  NUMA技术将CPU划分成不同的组（Node)，每个Node由多个CPU组成，并且有独立的本地内存、I/O等资源。Node之间通过互联模块连接和沟通，因此除了本地内存外，每个CPU仍可以访问远端Node的内存，只不过效率会比访问本地内存差一些.  在NUMA架构下，CPU的概念从大到小依次是：Node、Socket、Core、Processor。  随着多核技术的发展，我们将多个CPU封装在一起，这个封装一般被称为Socket，而Socket中的每个核心被称为Core。为了进一步提升CPU的处理能力，Intel又引入了HT（Hyper-Threading，超线程)的技术，一个Core打开HT之后，在OS看来就是两个核，当然这个核是逻辑上的概念，所以也被称为Logical Processor，本文简称为Processor。  综上所述，一个NUMA架构的主机可以有一个或者多个Socket，一个多核Socket显然包含多个Core，一个Core如果打开HT则变成两个Logical Processor。Logical processor只是OS内部看到的，实际上两个Processor还是位于同一个Core上，所以频繁的调度仍可能导致资源竞争，影响性能。注意node跟主机不是等同的概念，node只是CPU的分组，是个虚拟的概念，从一个node里面processor的个数就能看出。 |

|  |  |  |
| --- | --- | --- |
| node | |  | | --- | | A fairly technically correct and also fairly ugly definition of a node is: a region of memory in which every byte has the same distance from each CPU. A more common definition is: a block of memory and the CPUs, I/O, etc. physically on the same bus as the memory. |   numactl --hardware  available: 4 nodes (0-3)  node 0 cpus: 0 1 2 3 4 5  node 0 size: 8191 MB  node 0 free: 7788 MB  node 1 cpus: 6 7 8 9 10 11  node 1 size: 8192 MB  node 1 free: 7819 MB  node 2 cpus: 12 13 14 15 16 17  node 2 size: 8192 MB  node 2 free: 7839 MB  node 3 cpus: 18 19 20 21 22 23  node 3 size: 8192 MB  node 3 free: 7837 MB  node distances:  node 0 1 2 3  0: 10 16 16 16  1: 16 10 16 16  2: 16 16 10 16  3: 16 16 16 10  从上面可以看出本机有4个Numa node  ls /sys/devices/system/node/node0  cpu0 cpu2 cpu4 cpulist distance numastat  cpu1 cpu3 cpu5 cpumap meminfo scan\_unevictable\_pages  可见, node0包含0/1/2/3/4/5共6个Processor，这里的CPU指的是logical processor。 |
| socket | 通过/proc/cpuinfo查看，cpuinfo里的physical id描述的就是Socket的编号，可以从中找到本机到底有多少个Socket，并且每个Socket有那几个Processor。  cat /proc/cpuinfo | grep “physical id”  physical id : 0  physical id : 0  physical id : 0  physical id : 0  physical id : 0  physical id : 0  physical id : 0  physical id : 0  physical id : 0  physical id : 0  physical id : 0  physical id : 0  physical id : 1  physical id : 1  physical id : 1  physical id : 1  physical id : 1  physical id : 1  physical id : 1  physical id : 1  physical id : 1  physical id : 1  physical id : 1  physical id: 1  有几个不同的physical id就有几个socket。本机有2个socket，编号为0和1. |
| core | cat /proc/cpuinfo | grep “cpu cores”  cat /proc/cpuinfo | grep “core id”  cpu cores : 12 //一个socket有12个core，  core id : 0, 1, 2, 3, 4, 5 //每个core在socket内的编号  core id可能是不连续的.  本机有2个socket，每个有12个core，所以一共有24个core。 |
| processor | 这里的processor是指logical processor  cat /proc/cpuinfo |grep processor | wc -l  本机共有24个processor。因为core和processor的数目一致，所以本机没有开启HT，若后者是前者两倍，则开启了HT，如果CPU支持HT，OS还会把这个Core看成 2个Logical Processor。 |
| siblings | 每个socket中能有几个processor也可以从siblings字段中获取，siblings是每个socket里面processor的个数。当没有开启HT时，siblings和cpu cores是一致的。 |

1. Gromacs相关：  
   (1)重启任务（接着跑）：  
   mdrun **-s md1.tpr -cpi md1.cpt** -deffnm md1  
   (2)延长任务：  
   tpbconv -s md1.tpr -extend 10000 -o md2.tpr (延长的单位是ps)  
   mdrun -s md2.tpr -cpi md1.cpt -deffnm md2  
   (3)合并轨迹：  
   trjcat -f md1.xtc md2.xtc -o md1\_2.xtc  
   (4)处理轨迹：  
   trjconv -s md1.tpr -f md1.xtc -o md1\_noPBC.xtc -pbc mol -ur compact  
   trjconv -s md1.tpr -n index.ndx -f md1.xtc -o md1\_cluster\_center.xtc -pbc cluster -center

trjconv -s md1.tpr -n index.ndx -f md1.xtc -o md1\_noPBC\_nowater\_skip.xtc -pbc mol -ur compact -skip 2  
(5)提取去水的结构：  
editconf -f md1.gro -n index.ndx -o md1\_protein\_POPC.gro

(6)从轨迹文件提取snapshot：

trjconv -s md1.tpr -f md1\_noPBC.xtc -o md1\_44392.gro -dump 44392 -pbc mol -ur compact

去除周期性边界条件：

trjconv -s pull.tpr -f md3.gro -o md3\_noPBC.gro -pbc mol -ur compact

trjconv -s pull.tpr -f md3.gro -o md3\_noPBC.center.gro -pbc mol -ur compact -center

trjconv -s pull.tpr -f md3.gro -o md3\_noPBC.center.gro -pbc mol -ur compact -fit translation

(7)计算RMSD:

g\_rms -s md1.tpr -f md1\_2\_noPBC.xtc -o md1\_2.rmsd.xvg -tu ns  
(8)计算残基的**RMSF**:  
g\_rmsf -f md1\_noPBC\_50ns.xtc -s md1.tpr -res -fit -oq MIT1\_ RMSF\_oq\_res.pdb -o MIT1\_RMSF\_oq\_res.xvg  
(9)计算Area per lipid:  
 g\_energy -f ener.edr -o box.xvg (选择17 Box-X和18 Box-Y, 以0结束)  
 perl area.pl box.xvg  
(10)计算Lateral diffusion (MSD).  
g\_msd -s npt2.tpr -f npt2.xtc -n index.ndx -lateral z -o msd\_P1.xvg  
(11)计算Membrane thickness  
P1\_selection.dat:  
top\_P1 = name P1 and (z > 5);  
bot\_P1 = name P1 and (z < 5);

top\_P1;

bot\_P1;  
g\_select -s npt2.tpr -f npt2.xtc -on top\_bot\_P1.ndx -sf P1\_selection.dat  
g\_dist -s npt2.tpr -f npt2.xtc -n top\_bot\_P1.ndx  
或者：

g\_density -s nvt2.tpr -f nvt2.xtc -n index.ndx -o density\_P1.xvg

(12) Extract a tpr file for a subset of the original tpr file.

tpbconv -s md3.tpr -n MIT1\_POPC.ndx -o md3\_MIT1\_POPC.tpr

1. VMD TCL选择技巧：

(1). Single quotes are used to include spaces and other non-alphanumeric characters.  
resname ‘A 1’

name ‘O5\*’

name ‘O5\’’

(2). Double quotes around a string are used to specify a regular expression search.

Select all atoms with a name starting with C: name “C.\*”

Here, “.” matches any character except the new line character, “\*” means repeating 0 or more times.

(3). As in perl, there is a “match” operator “=~”

segname =~ “VP[1-4]” (matches VP1, VP2, VP3, VP4)

(4). within <number> of <selection>: selects all atoms within the specified distance (in Å) from a selection, including the selection itself.

same <keyword> as <selection>: finds all the atoms which have the same “keyword” as the atoms in the selection.

Useful example:  
**same residue as (protein within 5 of resname UDPG)**

(5). Select the heavy atoms in TCL:

**set heavy [atomselect top “protein and not name \“H.\*\””**]

1. PyMol里面选择原子或残基：  
   select bb, name c+o+n+ca   
   Create an atom selection named "bb" including all atoms named "C","O","N", or "CA" (Case insensitive.)  
   select ions, name N  
   select nterm, resi 1-3 is equivalent to select nterm, resi 1+2+3  
   select res, resn asp+glu+asn+gln (Case insensitive)
2. 在Ferlin上运行NAMD  
   加载所需的模块：  
   module add i-compilers/12.0.5  
   module add easy  
   module add mpi  
   module add openmpi/1.4.3-intel-12.0.0  
   module add tcl/8.3.5
3. VMD叠合两个蛋白并计算RMSD:  
   mol new MIT1.pdb  
   mol new MIT2.pdb  
   set MIT1\_backbone [atomselect 0 “name CA C N O”]  
   set MIT2\_backbone [atomselect 1 “name CA C N O”]  
   set M [measure fit $MIT2\_backbone $MIT1\_backbone]  
   set MIT2\_all [atomselect 1 all]  
   $MIT2\_all move $M  
   set rmsd [measure rmsd $MIT2\_backbone $MIT1\_backbone]  
   puts $rmsd
4. Schrodinger Host/Processors(CPUs)/Jobs:

|  |
| --- |
| **Host option menu**  The Host menu displays all the hosts defined in the $SCHRODINGER/schrodinger.hosts file, with the number of CPUs available on the host in parentheses. To run the selected job on a remote host, choose the host from this menu.  **CPUs text box**  Specify the number of CPUs to use to run the job. Absent if the job cannot be distributed or run in parallel. The number of CPUs (or cores) actually used is limited by the number of licenses available for the type of job.  **Separate job into *N* subjobs text box**  For jobs that can be distributed over multiple CPUs (and multiple hosts), specify the number of subjobs to split the job into. This number can be larger than the number of CPUs used, and it is often a good idea to choose a number that is several times the number of CPUs for optimal load balancing. |

1. VMD中将分子中心平移到坐标原点  
   set all [atomselect top all]  
   $all moveby [vecinvert [measure center $all]]
2. Perl的Sort函数：

Perl sort function uses two operators: cmp and <=>. So you can sort a list either in an alphanumerical or a numerical order. For this you can use the cmp (string comparison operator) or <=> (the numerical comparison operator). Please recall how this two operators work:

cmp returns -1, 0 or 1 depending on whether the left argument is stringwise less than, equal to, or greater than the right argument

<=> returns -1, 0 or 1 depending on whether the left argument is numerically less than, equal to, or greater than the right argument

Please note that if you explicitly use the cmp and <=> comparisons operators, it matters if $a or $b is on the left or right side of the operator. For instance, if you use $a <=> $b the list will be sorted in an **ascending** numerical order and if you use $b <=> $a the list will be sorted in a **descending** numerical order.

The way sort works in Perl is, that it goes over every two elements of the original array; In every turn it puts the value from the left side into the variable $a, and the value on the right side in the variable $b. Then it calls a comparison function. That "comparison function" will return -1 if the content of $a should be on the left, 1 if the content of $b should be on the left, or 0 if it does not matter as the two values are the same.

1. Linux的发邮件命令（**mail**）：

文本文件：

echo “This is content.” | mail -s “This is title.” [kwonglynn@gmail.com](mailto:kwonglynn@gmail.com)

mail -s “This is title.” < text.txt

发送附件：

(echo "Job submitting problem on Abisko"; **uuencode** NVT3C\_1340268.err NVT3C\_1340268.err) | mail -s "Job submitting problem on Abisko" kwonglynn@theochem.kth.se

1. Linux安装软件包：

安装RPM软件包：

rpm -ivh gcc-3.44-2.i386.rpm --prefix=$HOME/software

安装二进制文件(Source file):

tar -zxvf software.tar.gz

cd software

./configure --prefix=$HOME/software

make

make install

1. VMD专题：

|  |  |
| --- | --- |
| 测键长（两个原子的距离） | **measure bond atom\_list [options]**  Returns the distance of the two specified atoms. The atoms are specified in form of a list of atom indexes.  The following options can be specified:  -molid <default molid>  -frame <frame>  -first <frame>  -last <frame>  Example:  **measure bond {3 5}**―Returns the bond energy between atoms 3 and 5 of the current frame of the top molecule.  **measure bond {3 5} molid 1 frame all**―Returns the bond energy between atoms 3 and 5 of molecule 1 for all frames.  **measure bond {3 {5 1}} molid 0 first 7**―Returns the bond energy between atoms 3 of molecule 0 and atom 5 of molecule 1. The value is computed for all frames between the seventh and the last frame of molecule 0. |
| 获取碱性氨基酸的名字 | **lsort -unique [$basic get resname]** |
| 生成只包含蛋白的轨迹文件 | **catdcd -o eq\_pro.dcd -i protein.ind eq01.dcd**  Takes only the coordinates corresponding to the indices in 'protein.ind' from eq01.dcd and writes them to eq\_pro.dcd  -i <indexfile>  index file should contain the (zero-based) indices of the atoms in the dcd files whose coordinates are to appear in the output file. The indices should be ASCII text, separated by whitespace.   |  | | --- | | vmd protein\_water.psf protein\_water.pdb  VMD> set p [atomselect top "protein"]  VMD> $p writepdb protein.pdb  VMD> $p writepsf protein.psf  VMD> exit |  |  | | --- | | set f\_prot [open protein.ind w]  set prot [atomselect top protein]  puts $f\_prot [$prot get index]  flush $f\_prot  close $f\_prot | |
|  |  |

1. EUCB (NAMD trajectory analysis-RMSF calculation)

eucb is a free C++ program for analysis of molecular dynamics trajectories in .dcd format (CHARMM/XPLOR/NAMD).

Download and Installation:

<http://stavrakoudis.econ.uoi.gr/stavrakoudis/?menu=eucb>

module load gcc/4-centos6

Usage:

<http://195.130.120.154/wiki/index.php/Eucb:rmsf>

|  |
| --- |
| eucb -psf protein.psf -dcd protein.dcd -rmsf -seq C:10-20  eucb -psf protein.psf -dcd protein.dcd -pdb protein -rmsd backbone3 -seq A,B |

1. Amber ptraj轨迹处理相关

为了视觉方便，防止系统移动幅度过大，或防止系统移出盒子

|  |
| --- |
| 隐性水（非周期性）：**nscm = 1000 (default)**  Flag for the removal of translational and rotational center-of-mass (COM) motion at regular intervals (default is 1000). For non-periodic simulations, after every NSCM steps, translational and rotational motion will be removed. The only reason to even reset the coordinates is to prevent the molecule from diffusing so far away from the origin that its coordinates overflow the format used in restart or trajectory files.  显性水（周期性）：**iwrap = 1**  If iwrap = 1, the coordinates written to the restart and trajectory files will be "wrapped" into a primary box. This means that for each molecule, its periodic image closest to the middle of the "primary box" will be the one written to the output file. This often makes the resulting structures look better visually, but has no effect on the energy or forces. Performing such wrapping, however, can mess up diffusion and other calculations. If iwrap = 0, no wrapping will be performed, in which case it is typical to use ptraj as a post-processing program to translate molecules back to the primary box. For very long runs, setting iwrap = 1 may be required to keep the coordinate output from overflowing the trajectory and restart file formats, especially if trajectories are written in ASCII format instead of NetCDF. |

ptraj使用

格式：ptraj MIT1.prmtop < ptraj.in > ptraj.out

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1. 从轨迹中提取snapshot：   |  | | --- | | **trajin** ../md1.mdcrd 1000 1000  **center @CA mass origin**  **image origin center familiar**  **trajout** md1\_1000.rst **restart** |   2. 处理restart文件的格式，使其便于查看：   |  | | --- | | trajin md1.rst  center @CA mass origin  image origin center familiar  trajout md1\_center\_image.rst restart |   3. 将restart文件转换成轨迹文件，查看PBC盒子   |  | | --- | | trajin md1.rst  center :1-141 mass  image center familiar (注意不要使用origin，否则盒子会移位！)  trajout md1\_center\_image.mdcrd trajectory |     4. 处理轨迹，计算RMSD，除去水和离子等   |  | | --- | | trajin md1.mdcrd  center @CA mass origin  image center familiar  rms first out MIT1\_ MD1\_CA.rmsd @CA  strip :WAT  strip :Na+  trajout MIT1\_MD1\_dry\_nice.mdcrd nobox |   5. 处理REMD轨迹，提起某一温度下的轨迹   |  |  | | --- | --- | | trajin remd.c2.mdcrd.001 remdtraj remdtrajtemp 300.00  trajout remd.c2.300K.mdcrd   |  | | --- | | To read an Amber REMD trajectory at a particular temperature from an ensemble of replica trajectories, specify the **remdtraj remdtrajtemp** keywords followed immediately by the replica temperature (reptemp, default 0.0), and then files will be searched and the final trajectory found at that particular temperature will be used. Note that in this case, trajin assumes the Amber REMD files are named according to the convention that the end of the filename includes “.N”, “.N.gz” or “.N.bz2” where N is the number of replicas ending specification of the replica number (i.e., NNN = 000, 001, ...). An example call is “trajin remd.000.gz remdtraj remdtrajtemp 300.0” which will search all existing files at that path location named remd.N.gz for the final replica at 300.0 degrees. Filenames with an appended .Z or .gz or .bz2 are also recognized and treated appropriately, with the exception of compressed Amber NetCDF files (and .Z compressed Amber REMD files).  Finally, please note that the coordinates must match the names/ordering of the molecular information file (prmtop) previously read in. | | |

1. PyMol做高清图（Ray-Tracing）

|  |  |  |
| --- | --- | --- |
| Ray-tracing produces the highest quality molecular graphics images. PyMOL is the first full-featured molecular graphics program to include a high-speed ray-tracer.  All images (ray−traced or not) can be saved in PNG format using the "png" command.  Usage   |  | | --- | | ray [width,height [,renderer [,angle [,shift ]]]  **angle** and **shift** can be used to generate matched stereo pairs  **width** and **height** can be set to any non-negative integer. If both are set to zero than the current window size is used and is equivalent to just using **ray** with no arguments. If one is set to zero (or missing) while the other is a positive integer, then the argument set to zero (or missing) will be scaled to preserve the current aspect ratio. |   Example:   |  | | --- | | ray 3000,2400  png filename.png  For better quality maps with a white background.  set ray\_trace\_fog,0  set ray\_shadows,0  set antialias,1  ray 1600,1200  png img.png | |

1. VMD-Measure the distance between two groups of atoms (COM distance measuring)

For a standalone structure:

|  |
| --- |
| set prot [atomselect top protein]  set lipid [atomselect top “resname POPC”]  set COM\_prot [measure center $prot weight mass]  set COM\_lipid [measure center $lipid weight mass]  set distance [**veclength** [**vecsub** $COM\_prot $COM\_lipid]] |

For a trajectory:

|  |
| --- |
| set prot [atomselect top protein]  set lipid [atomselect top “resname POPC”]  set nf [molinfo top get numframes]  ##################################################  # Loop over all frames. #  ##################################################  set outfile [open distance.dat w]  for {set i 0} {$i < $nf} {incr i} {  puts "frame $i of $nf"  $prot frame $i  $lipid frame $i  set COM\_prot [measure center $prot weight mass]  set COM\_lipid [measure center $lipid weight mass]  set distances($i) [veclength [vecsub $COM\_prot $COM\_lipid]]  puts $outfile "$i $distance($i)"  }  close $outfile |

1. Linux BASH括号的作用：

|  |  |
| --- | --- |
| 圆括号(), (()) | <http://mywiki.wooledge.org/ArithmeticExpression>  Arithmetic in BASH is integer math only. You can't do floating point math in Bash.  There are several ways to tell Bash to treat numbers as integers instead of strings, and to do basic arithmetic operations on them. The first is to use the let command.  Note that each arithmetic expression has to be passed as a single argument to the let command, so you need quotes if there are spaces or globbing characters, thus:  Division in Bash is integer division, and it truncates the results, just as in C.  In addition to the let command, one may use the (( )) syntax to enforce an arithmetic context. If there is a $ (dollar sign) before the parentheses, then a substitution is performed. White space is allowed inside (( )) with much greater leniency than with let, and variables inside (( )) don't require $ (because string literals aren't allowed).  (( )) without the leading $ is not a standard sh feature. It comes from ksh and is only available in ksh, Bash and zsh. $(( )) substitution is allowed in the POSIX shell. As one would expect, the result of the arithmetic expression inside the $(( )) is substituted into the original command.  (( var = 78 )) Integer arithmetic  var=$(( 20 + 5 )) Integer arithmetic, with variable assignment  (( var++ )) C-style variable increment  **Other:**  ( command1; command2 ) Command group executed within a subshell  Array=(element1 element2 element3) Array initialization  result=$(COMMAND) Command substitution, new style  Command substitution allows the output of a command to replace the command itself. Command substitution occurs when a command is enclosed as follows:  $(command)  or  `command` |
| 方括号[], [[]] | <http://mywiki.wooledge.org/BashFAQ/031>  [ ("test" command) and [[ ("new test" command) are used to evaluate expressions. [[ works only in Bash, Zsh and the Korn shell, and is more powerful; [ and test are available in POSIX shells.  No WordSplitting or glob expansion will be done for [[ (and therefore many arguments need not be quoted). This makes [[ easier to use and less error-prone.  Parentheses in [[ do not need to be escaped.  [[ -f $file1 && ( -d $dir1 || -d $dir2) ]]  [ -f "$file1" -a \( -d "$dir1" -o -d "$dir2" \) ]  As a rule of thumb, [[ is used for strings and files. If you want to compare numbers, use an ArithmeticExpression.  When should the new test command [[ be used, and when the old one [? If portability to POSIX or the BourneShell is a concern, the old syntax should be used. If on the other hand the script requires BASH, Zsh or KornShell, the new syntax is much more flexible.  **Other:**  Array[1]=element1 Array initialization  [a-z] Range of characters within a Regular Expression |
| 花括号{} | The ‘$’character introduces parameter expansion, command substitution, or arithmetic expansion.  The basic form of parameter expansion is ${parameter}. The value of parameter is substituted. The braces are required when parameter is a positional parameter with more than one digit, or when parameter is followed by a character that is not to be interpreted as part of its name.  **${parameter/pattern/string}**  The pattern is expanded to produce a pattern just as in filename expansion. Parameter is expanded and the longest match of pattern against its value is replaced with string. If string is null, matches of pattern are deleted and the / following pattern may be omitted.  **Other:**  { command1; command2; . . . commandN; } Block of code  {a..z} Extended brace expansion  {} Text replacement, after find and xargs |

1. Steered Molecular Dynamics (COM Pulling)

|  |  |
| --- | --- |
|  | http://www.ks.uiuc.edu/Training/Tutorials/namd/namd-tutorial-unix-html/img169.png |
|  | \fbox{ \begin{minipage}{.2\textwidth} \includegraphics[width=2.3 cm, height=2.... ...e too high or the measured force will be dominated by noise. } \end{minipage} } |
|  | \fbox{ \begin{minipage}{.2\textwidth} \includegraphics[width=2.3 cm, height=2.... ...perform long enough simulations to see the unfolding pathway.} \end{minipage} } |
| Note: | **It is very important that you do not apply extremely fast pulling rates or extremely strong force constants, which can seriously deform elements of your system.** |
| Constraints: | **No constraints should be applied to the system when doing the pulling simulation, otherwise the system would have deformation.** |
|  |  |

1. 作业任务查看

|  |  |  |
| --- | --- | --- |
|  | PBS | SLURM |
| 查看工作目录 | qstat -f JobID | scontrol show job JobID |
| 查看账号下的所有任务 | projinfo | tail -n 9 | awk '{print $1}' | xargs -i qstat -u {} | squeue -A ProjectID |
| 按用户名查询 | qstat -u UserName | squeue -u UserName |
| 删除某用户的所有任务 | squeue -h -u UserName | awk ‘{print $1}’ | xargs -i scancel {} | qdel $(qselect -u UserName)  or  sqelect -u UserName | xargs -i qdel {} |

1. **Xmgrace指定对某一列作图：**

|  |
| --- |
| To pick and choose columns from a matrix-type file:  x y1 y2 y3 y4 ...  x y1 y2 y3 y4 ...  Plot col 2 vs. 1 and col 4 vs. 1 in one graph:  **xmgrace -block file -bxy 1:2 -bxy 1:4** |

例子：

xmgrace -block PAP1\_COM\_distance.xvg -bxy 1:5

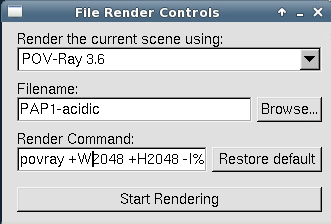
1. EASY任务（Povel, Ferlin）查看任务所在节点:

spusage | grep guanglin

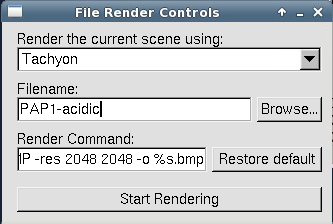
spusage | grep 032314572127

1. VMD做高清图：

1. pov +W2048 +H2048 -I%s -O%s.png -D -X +A +FN



2. “/software/apps/vmd/1.9.1/lib/tachyon\_LINUXAMD64” -aasamples 12 %s -format BMP -res 2048 2048 -o %s.bmp



1. Linux图片格式转换：

convert -resize 1024x1024 name.bmp name.tif (数字之间为字母x)

1. R保存图片：

|  |
| --- |
| Saving graphs:  It is possible to output graphs created in R in many different formats including: Portable Network Graphics (PNG), Windows Bitmap (BMP), postscript (ps) and JPEG. The functions that output the file are named after the format, thus the **jpeg** function produces a JPEG file, the **png** function produces a PNG file, the **bmp** function produces a bmp file, the **pdf** function creates a pdf file and the **postscript** function produces a postscript file. The procedure for saving the graphs to a different format is to first create a file using the one of the functions for outputting plots; then perform all the plot functions to achieve the final graph; and then close the graphics device by using the **dev.off** function. The options **h** and **w** control the size (in pixels) of the height and width of the graph being saved.  The following example creates portable network graphics file of a histogram of var1 in a file called hist.png.  **png("hist.png")**  **hist(var1)**  **dev.off()**  Here is an example creating a pdf file of a boxplot of var1 in a file called boxplot.pdf using the **pdf** function.  **pdf("boxplot.pdf")**  **boxplot(var1)**  **dev.off()** |

R改变坐标轴数字大小：

plot(pc, col = grps, cex.axis=1.8, cex.lab=1.8)