蛋白质模拟器概要

2024-09-03

**蛋白质静态坐标**

PDB蛋白质数据库中获得的蛋白质数据是各个蛋白质的静态的原子坐标集合，比如蛋白质1VII，包含有596个原子，从PDB网站下载的pdb文件（1vii.pdb）中主要存储的是596个原子的三维坐标。参考《书稿第2章蛋白质数据库PDB.pdf》。

**蛋白质的运动本质**

依据物理学的认知，蛋白质是一种由多个原子通过化学键组合在一起的生物大分子，具有特定的立体构象，具有特定的生物化学功能。只要环境温度大于绝对0开尔文温度（大约-273摄氏度），蛋白质的每个原子就会处于运动状态，由于化学键的相互作用，运动是受到约束的，在常温状态（25摄氏度为实验室温度，30-37摄氏度为生物体温度），能够维持一定的立体构象，从而具有特定的功能。

**实验观测的局限性**

蛋白质本质上是处于运动状态的，但是目前的实验技术还不能够观测其具体的运动过程。目前有飞秒摄影技术，能够拍摄光子的运动，但是还未见对生物大分子的拍摄报导。

**分子动力学软件**

分子动力学计算系统，起步于80年代，目前比较知名的为amber系统，基于分子动力学的数值计算方法，分析每个原子的受力，计算原子的运动轨迹，给出原子的瞬时坐标，属于计算物理和计算力学的分支。

**qomd蛋白质模拟器**

qomd蛋白质模拟器，是河海大学孟老师课题组开发的一种分子动力学蛋白质模拟软件系统，具有如下特点：

1完全独立自主的系统，计算原理、数据结构、体系结构、代码实现，都是独立自主的，与其他分子动力学系统没有传承关系。

2模拟参数可调的系统，分子动力学模拟系统的关键部分为分子力场的建模和参数设计，qomd系统基于新设计的分子轨道和原子轨道建模，将原子相互作用的电荷力和弹性振子的震动频谱均进行参数化，可以动态调整。可以根据观测目标的要求，进行不同参数条件下的动态模拟。

**蛋白质模拟器运行方式和输出**

是一个基于分子动力学计算，模拟蛋白质原子运动的软件包，初始时刻的各个原子坐标可以设定为与pdb文件（1vii.pdb）原子坐标一致，然后模拟计算每个时刻（飞秒间隔，1飞秒为10的-15秒）每个原子的新的坐标位置，并输出该时刻的坐标。

**动态点云数据**

蛋白质模拟器输出的数据可以看作是一种动态点云数据，与激光扫描器获得的动态点云数据不同之处在于，

1这些数据是软件模拟出来的，可控性强。

2模拟器是参数可调的，模拟数据可以根据需要进行产生。

**神经网络的角度（各位同学的工作）**

在pytorch等成熟神经网络开发平台的基础上，可以设计各种不同类型的神经网络，作为蛋白质动态点云数据的观察器。观察器的目的可以是多种多样的，***这是各位同学的工作方向***。

**模拟器运行参数可调的科学意义**

上述观察器的目的都可以指向一个共同的目标，就是获得一组合适的模拟器运行参数。蛋白质模拟器得到的不是真实物质的真实状况，模拟器本质上是真实世界的近似，为了获得更好的近似效果，（比如蛋白质折叠过程的模拟），需要对模拟器进行不断地调整，或者说，在基本算法和轨道建模的大框架确定的基础上，要根据实际的***模拟效果***，不断地调整模拟参数（电荷系统和频谱系统），使得模拟效果接近真实世界。

**模拟效果是指什么（各位同学的工作）**

先举一例，比如长时间模拟运行的条件下，蛋白质是不是能够维持相对稳定的立体构象，以及相对稳定的立体构象又是如何定义的（这就不是一个很简单的问题，还能牵引出许多别人没有考虑过的新问题），我觉得这都是神经网络要完成的部分。

**用于模拟的蛋白质数据包**

目前先模拟一个单独的蛋白质，暂不考虑两个以上蛋白质之间的相互作用。

1vii蛋白质是一个比较知名的蛋白质（用于研究蛋白质折叠），这个蛋白质的初始坐标数据以及相关的模拟运行参数都打包在一个数据包里（proteinBag\_1VII\_linux.byte）。

后续还要打包不同的蛋白质，形成一个数据集。

**蛋白质模拟研究领域的认知进化**

在90年代末期，大约1998年，科学界首次用分子动力学模拟软件模拟了一个小蛋白质（1vii，36氨基酸，596原子）的折叠过程，从展开的肽链状态模拟折叠为大致的球型状态。

模拟蛋白质折叠过程，也是我这些年工作的主要目标，但是为什么又走回到了先做好一个真正独立自主的模拟器这条路上了。

今天太晚了，这个问题以后有时间再说。

**总结**

今天（2024-09-03）先总结一下，模拟器是做好了，但仍然还是个半成品，需要用各位同学设计的神经网络观测运行数据和训练调整模拟器参数，得到全成品的模拟器。

2024-09-05

**蛋白质模拟器代码包**

src\qomdProtein2024.cpp，是模拟器的运行样例。

其中含有打包在obj\libQomd.a中的函数和数据结构，知道如何调用即可。

void procedure\_24102(FILE\* fptr\_writerecord\_test\_output)简介

PROCEDURE\_STEP : proteinBag\_1VII get from byte file <proteinBag\_1VII\_linux.byte>

void \* proteinBag\_1VII是蛋白质1VII的模拟数据包的头地址，所有的运行数据都在这里面，其中的主要数据部分从文件<proteinBag\_1VII\_linux.byte>中读取。

PROCEDURE\_STEP : get qomdRunPara

运行参数InputPara \* qomdRunPara从文件"cppCuda\_run\_para\_linux\_2024.txt"中读取。

PROCEDURE\_STEP : reset qomdRunPara in proteinBag

运行参数InputPara \* qomdRunPara送进蛋白质1VII的模拟数据包proteinBag\_1VII。

PROCEDURE\_STEP : reset CudaGridBlock

设置GPU运行参数，暂时不用，保留模块。

PROCEDURE\_STEP : init osci data before run

在运行模拟器前，还要根据运行参数qomdRunPara初始化一些必要的数据，都在模拟数据包proteinBag\_1VII里面。

PROCEDURE\_STEP : init dvnn vertexList

DvnnAtomicVertexList\* vertexList是一个独立于模拟数据包proteinBag\_1VII的瞬时（当前时刻）原子数据包，可以随时从模拟数据包proteinBag\_1VII中copy数据出来，DvnnAtomicVertexList数据结构是可见的，可以在此基础上进一步写代码。

PROCEDURE\_STEP : init dvnn timePointList

DvnnTimePointList\* timePointList是另一个独立的动态点云（时间变化的）原子数据包，从DvnnAtomicVertexList\* vertexList数据包中copy数据出来。通过函数dvnn\_save\_timePointList\_listData\_to\_byte\_file将数据输出到字节文件"time\_point\_linux.byte"里。该字节文件的读写方法都代码可见。

PROCEDURE\_STEP : qomd\_femto\_iterator set run method

模拟器可以有四种运行方案，由运行参数InputPara \* qomdRunPara确定。

PROCEDURE\_STEP : iterate

模拟器运行模块，循环运行，并且每一步都输出数据到vertexList和timePointList，并且写数据到文件"time\_point\_linux.byte"里。

PROCEDURE\_STEP : output dvnn vertexList

结束模拟并输出数据到文件"dvnn\_vertexList\_2.txt"和"dvnn\_vertexList\_pdb\_2.txt"。

void procedure\_24103()简介

从文件"time\_point\_linux.byte"读取点云数据的样例。

问题1：为什么只会运行procedure\_24102

(FILE\* fptr\_writerecord\_test\_output)函数，怎么样才能运行procedure\_24103()函数？

答：运行入口在文件"QOMD\_run\_para\_linux\_2024.txt"中设置，程序运行该文件中的第一个RUN\_PROCEDURE值。

问题2：在init dvnn vertexList步骤中输出的dvnn\_vertexList\_1.txt是proteinBag\_1VII的瞬时（当前时刻）原子数据吗？既然是当前时刻，为什么运行的结果始终是相同的值？

答：DvnnAtomicVertexList\* vertexList是一个独立于模拟数据包，存储瞬时数据，可以写到文件里，dvnn\_vertexList\_1.txt就是初始的写出数据，运行结束后的写出数据在"dvnn\_vertexList\_2.txt"

问题3：qomd\_femto\_iterator set run method中，RUN\_METHOD : 1 = COUVDW or 2 = ELCCOL，RUN\_MODE : 1 = SER or 2 = PAR，各代表什么意思？

答：COUVDW 库伦力范德华力模型，ELCCOL电荷力碰撞力模型，SER串行模式，PAR并行模式。是模拟器内部的运行模式选择。这个先不必细究，随便选就可以。

问题4：iterate中为什么要输出round和temperature？（WRITE\_FREQ数值越大，running time也会变大）（time\_start的值在WRITE\_FREQ不同时也不同，那又该怎么解释问题2？）

答：输出温度是从宏观角度观测运行是否正常，在300K范围附近波动都属于正常，WRITE\_FREQ用来控制输出温度的频次，**不必每一步都计算温度**。

问题5：输出的dvnn\_vertexList\_2.txt中只有596个原子各自的坐标值，那么怎么体现不同时刻原子的位置坐标？

答：不同时刻原子的位置坐标，由DvnnTimePointList\* timePointList负责提取和输出。

**2024-11-22**

**蛋白质长链模拟折叠**

从长链状态开始，模拟折叠成蛋白质的立体结构，与PDB蛋白质数据库中获得的蛋白质立体结构进行对比。

过程，每一步（飞秒）计算各个振子（四种振子）的组成原子的运动位移，计算所有原子对的电荷力碰撞力位移，然后将原子位移叠加，得到一个步长的原子位移。

特点，时间长，每个步长的运动是计算可靠的。

假设：如果模拟器的各个参数是正确的，长链模拟运行的结果就会与实验数据（PDB蛋白质数据）一致。

**折叠原理基本要点**

（1）四种振子的运动中，bond，angle，plane，三种都是在**单一平衡位置**附近周期性弹性运动，键长、键角都不会有大的变化，其主要作用是通过共价键作用维持蛋白质原子局部相互位置关系。

（2）elccol(或者couvdw)原子对的电荷力碰撞力，是通过电荷斥力和吸力以及近距离碰撞力，负责牵引原子之间的位置关系。

（3）dihedral的二面角是折叠变化的核心，通过上述（1）（2）的作用力和位移，以及dihedral本身的弹性力作用（相对要弱很多，频率小很多），dihedral的二面角不断变化（有些像旋钮一样可以360度旋转变化，有些变化范围受限与plane类似）。

（4）所有的dihedral的二面角的取值状态，确定了蛋白质的立体结构。

**振子能量、能量流转**

（1）四种振子的运动中，bond，angle，plane，三种都是在平衡位置附近周期性弹性运动，决定运动规律的是振子能量和振动频率，振子能量由极限拉伸位置和极限压缩位置确定，而且这个极限位置是变化的，意味着振子的能量是变化的。举个单摆的例子，当外力的推拉作用与单摆运动相互作用时，就可以增加或者减小单摆的极限位置，等于是增加或减少了单摆的能量。

（2）elccol(或者couvdw)原子对的电荷力碰撞力，也有能量的变化，也是一种势能动能的转化。比如正负电荷相吸的两个原子，相当于是具有吸引势能，如果一个飞秒步长计算使得两原子相互靠近了，相当于是势能转化为动能，或者是能量流转给了其他振子。如果一个飞秒步长计算使得相吸的两原子相互离远了，相当于是动能转化为势能，或者是能量从其他振子流转给该原子对。

至于每个计算步长有多少势能动能转化，取决于电荷值，**（猜测）此电荷值也是随温度而变化的**，**温度实际上是系统的总能量的宏观衡量**。【实际计算可以固定每个原子的电荷值，然后设置一个由温度控制的**势能动能转化率因子**。】

（3）dihedral振子也是具有周期性弹性运动特性的，只是其平衡点不是一个，可能是多个，其状态在若干个平衡位置之间转换（可以称为**相变**），也有能量的变化，相对其他而言较小。

（4）综上所述，各种振子的变化本质上来说都是其各自能量驱动的，能量的大小，决定了振子中各个原子的位移大小，如果某个局部区域的各个振子的能量大，则可能产生的结构变化（dihedral**相变**）的概率就比较大。**长链的折叠过程可以认为是能量驱动下的各个dihedral不断相变的过程。**

**能量流转、频谱的关系**

（1）能量是在各个振子和elccol(或者couvdw)原子对之间流转的，而且这种流转是不均匀的，取决于当前的振子之间的空间位置布局，以及更重要的频谱关系。仍然举例单摆（比推秋千），当外力作用（相当于能量流转）与单摆运动同步时（相当于同频或者倍频），可以使得单摆的极限位置快速增大或者减小，既能量流转的效率很高。相反，不同频时，能量流转的效率低。也就是说，**频谱关系决定了能量的流转效率**。

（2）结合前述，长链的折叠过程可以认为是能量驱动下的各个dihedral不断相变的过程，而不同的dihedral获得能量的大小和次序是不同的，取决于频谱关系的和当前的局部布局，先获得高能量的dihedral先实现相变，后获得高能量的dihedral后实现相变。而这种各个dihedral相变的次序，实际上就相当于是蛋白质的折叠路径，不同的折叠路径决定了折叠的最终立体构型。

**动态的能量流转网络**

（1）按照上述机制，建立与模拟器相配套的能量流转网络，可以揭示蛋白质折叠的机理。

（2）这个能量流转网络可以是一个有学习能力的神经网络，对各种尺度的局部布局和频谱关系以及能量流转规律进行总结，形成一些**折叠规则**。

（3）这些规则可以用来加速模拟器的运行，也就是说，不必每个飞秒步长都需要计算，可以利用这些折叠规则跨越一些模拟步骤，加快模拟折叠过程。

**利用能量流转网络加速的可行性**

（1）在模拟折叠过程中，各个dihedral的相变是折叠过程的关键节点性事件，而其他非dihedral相变的过程都是对相变事件的弛豫过程或者称为适应过程，而这些弛豫过程的模拟其实是可以跨越的。

（2）而各个dihedral相变可以由能量流转网络进行预测。

**当前的工作**

（1）先将模拟器中**能量计算**和**能量转移**的功能完善。这个很快就可以搞好。

**2025-01-06**

接前文：“先将模拟器中**能量计算**和**能量转移**的功能完善”。这个部分初步可以运行了。

**运行文件选项：QOMD\_run\_para\_linux\_2024.txt**

（1）1vii从长杆模型开始（长链状态），模拟折叠过程

RUN\_PROCEDURE | 24115 | | run 1vii\_rod

（2）1vii的PDB蛋白质数据，模拟运动

RUN\_PROCEDURE | 24105 | | run 1vii

**参数文件选项：cppCuda\_run\_para\_linux\_2024.txt**

PROTEIN\_NAME | 1VII

| not used, preserve, do not modify

蛋白质名，保留，未使用

GPU\_DEVICE\_NAME | 970

| not used, preserve, do not modify

显卡名，保留，未使用

DATA\_DIR | /home/m1/qomd2022/1VII/

| not used, preserve, do not modify

路径名，保留，未使用

RUN\_START | 1

| 1 = START or 0 = CONTINUE, used, do not modify

选项：重新开始模拟/继续上一次的模拟（此功能需要输出保存现场的中继文件）（此功能未完成，但是此功能实现比较简单，以前版本就有这个功能，可以很快搞好），

PARA\_ELEC | 2

| 1 = use qomd(old fixed para), 2 = use hhmo(new adjustable para), add 2024-10-70

选项：电荷参数（qomd旧参数，固定不可调）/电荷参数（hhmo新参数，可调），

PARA\_FREQ | 2

| 1 = use qomd(old fixed para), 2 = use hhmo(new adjustable para), add 2024-10-70

选项：频谱参数（qomd旧参数，固定不可调）/频谱参数（hhmo新参数，可调），

RUN\_MODE | 2

| 1 = SER or 2 = PAR or 3 = CUDA, could be modified, 1 or 2

选项：串行/并行/GPU（此功能未完成），

RUN\_METHOD | 2

| 1 = COUVDW or 2 = ELCCOL or 3 = only\_4\_osci, could be modified, 1 or 2 or 3

选项：非键计算（COUVDW旧方法）/非键计算（ELCCOL新方法）/不计算非键（仅做测试用），

解释：虽然COUVDW仍然可以运行，但是我们后面基本不用这种方式了（暂时不打算进一步研究这个COUVDW模式了）。

RUN\_GPU | 1

| 1 = one gpu or 2 = two gpus, not used, preserve, do not modify

选项：用一个显卡/用两个显卡，（此功能未完成）

RUN\_ROUNDS | 10000

| form 100 to 100000, used, could be modified

参数：模拟步数

WRITE\_FREQ | 1

| write frequence, for compute temperature, could be modified

参数：输出温度等评估值的频度

WRITE\_MEDIUM\_FREQ | 10000

| write medium data files and pdb file, to protect run data when power off for servers, not used, preserve, do not modify

参数：输出中继文件的频度，为了中断运行（或意外停机）后继续运行，配合RUN\_START选项0，（此功能未完成）

BOND\_DEVI\_RATIO | 0.2

| double bond\_deviation\_ratio = 0.0435; qomd run para, used, do not modify

参数：BOND振子的最大偏移比例。“此参数可以小范围修改”，“此参数影响温度”

解释：BOND振子的“最大偏移量”由“平衡位值”和这个“最大偏移比例”共同确定。

按照0.2取此参数，比如平衡位（两原子距离）是1.0，则“最大偏移量”为0.2，则“最大（拉伸）量”为1.2，“最大（压缩）量”（压缩）为0.8。

同时确定一个“最小偏移量”，取“最大偏移量”的0.5（此值内置固定），

按照0.2取此参数，比如平衡位（两原子距离）是1.0，则“最小偏移量”为0.1，则“最小（拉伸）量”为1.1，“最小（压缩）量”（压缩）为0.9。

ANGLE\_DEVI\_RATIO | 0.2

| double angle\_deviation\_ratio = 0.0435; qomd run para, used, do not modify

参数：ANGLE振子的最大偏移比例。“此参数可以小范围修改”，“此参数影响温度”

解释：同上。

PLANE\_DEVI\_RATIO | 0.2

| double plane\_deviation\_ratio = 0.0435; qomd run para, used, do not modify

参数：PLANE振子的最大偏移比例。“此参数可以小范围修改”，“此参数影响温度”

解释：同上。

DIHE\_LIMIT\_RATIO | 0.0435

| double dihedral\_ratio\_LimitDeviation = 0.0435; qomd run para, used, do not modify

参数：dihedral振子的“谷底”最大偏移比例。“此参数暂时不改”。

解释：这个解释比较复杂，暂缓。

DIHE\_FLATTOP\_RATIO | 0.9

| double dihedral\_ratio\_FlattopDeviation = 0.9; qomd run para, used, do not modify

参数：dihedral振子的“峰顶”偏移比例。“此参数暂时不改”

解释：这个解释比较复杂，暂缓。

BOND\_ENFORCE\_RATIO | 0.3

| double bond\_normal\_inner\_regressive\_enforce\_ration = 0.3; qomd run para, used, could be modified

参数：BOND振子的回位操作比例参数。“此参数可以小范围修改”，“此参数影响温度”

解释：“强制回位操作”：当振子的状态超出了“极限偏移量”（参考参数BOND\_DEVIMODI\_RATE的解释）时，执行拉回操作，这个“拉回操作的具体量值”由“超出量”结合这个“回位操作比例参数”共同确定。

ANGLE\_ENFORCE\_RATIO | 0.3

| double angle\_normal\_inner\_regressive\_enforce\_ration = 0.3; qomd run para, used, could be modified

参数：ANGLE振子的回位操作比例参数。“此参数可以小范围修改”，“此参数影响温度”

解释：同上。

PLANE\_ENFORCE\_RATIO | 0.3

| double plane\_normal\_inner\_regressive\_enforce\_ration = 0.3; qomd run para, used, could be modified

参数：PLANE振子的回位操作比例参数。“此参数可以小范围修改”，“此参数影响温度”

解释：同上。

BOND\_DEVIMODI\_RATE | 0.1

| double bond\_limit\_deviation\_modify\_rate = 0.5; qomd run para, used, could be modified

参数：BOND振子的“极限偏移量”修改算法比例参数。“此参数可以小范围修改”，“此参数影响温度”

解释：前面提到的“最大偏移量”是一个固定值，含义是该振子的偏移量（拉伸态或压缩态）都不能超过这个值，而每一步模拟过程中都可以在“最大偏移量”和“最小偏移量”范围内修改一个参数称为“极限偏移量”，

既满足条件：“最大偏移量”>“极限偏移量”>“最小偏移量”

这个“极限偏移量”可以表示当前该振子所具有的能量，是不断变化的，相当于是能量的流转（参考前面“振子能量流转”的解释），而这个比例参数（BOND\_DEVIMODI\_RATE）是用来控制这个“极限偏移量”修改算法的。这个“极限偏移量”也决定下一步模拟的振动位移量，也就是对温度有影响。

ANGLE\_DEVIMODI\_RATE | 0.1

| double angle\_limit\_deviation\_modify\_rate = 0.5; qomd run para, used, could be modified

参数：ANGLE振子的“极限偏移量”修改算法比例参数。“此参数可以小范围修改”，“此参数影响温度”

解释：同上。

PLANE\_DEVIMODI\_RATE | 0.1

| double plane\_limit\_deviation\_modify\_rate = 0.5; qomd run para, used, could be modified

参数：PLANE振子的“极限偏移量”修改算法比例参数。“此参数可以小范围修改”，“此参数影响温度”

解释：同上。

RATIO\_ELCCOL\_R\_WALL | 0.7

| 0.7, qomd run para, used, do not modify

参数：碰撞墙的位置参数

解释：ELCCOL为电荷力/碰撞力的相互作用计算，所有的非键两原子之间的相互作用可以认为是，当距离足够近时，执行碰撞计算，当距离超出碰撞作用范围时，执行电荷力计算。而这个“碰撞作用范围”由两个值设定，“碰撞墙位置”和“接触位置”，可以类比为两个有弹性的皮球，“碰撞墙位置”相当于皮球挤在一起的极限位置，“接触位置”相当于皮球仅仅是挨在一起。“接触位置”的设定由两个范德华距离（类似于原子半径）相加得到，而“碰撞墙位置”由这个参数“RATIO\_ELCCOL\_R\_WALL”结合“接触位置”计算得到。所以这个参数也是个经验值，可以修改。“此参数影响温度”

RATIO\_ELCCOL\_COL1 | 0.2

| 0.5, qomd run para, used, could be modified

参数：碰撞力的计算参数1，

解释：当前非键两原子之间的距离比“碰撞墙位置”还要小，执行回弹计算的比例参数，这个参数也是个经验值，可以修改。“此参数影响温度”

RATIO\_ELCCOL\_COL2 | 0.1

| 0.3, qomd run para, used, could be modified

参数：碰撞力的计算参数2，

解释：当前非键两原子之间的距离比“碰撞墙位置”大，但是小于“接触位置”，执行回弹计算的比例参数，这时的回弹力要小于前面的情况，这个参数也是个经验值，可以修改。“此参数影响温度”

RATIO\_ELCCOL\_ELC | 0.02

| 0.05, qomd run para, used, could be modified

参数：电荷力的计算参数，

解释：当前非键两原子之间的距离比“接触位置”大，可以认为没有回弹力作用，仅仅计算电荷力作用，这个比例参数对温度影响很大（因为电荷力计算是主要的折叠驱动力），这个参数也是个经验值，可以修改。“此参数影响温度”

**2025-01-06**

**评估参数**

struct EvaluatePara {

//////////////////////////////////////////////////////////////////

// temprature

//////////////////////////////////////////////////////////////////

double t\_DeltaR      ;

解释：总温度，应该控制在实验室温度300k左右（摄氏27度），或者生物体温度308k左右（摄氏35度）

double t\_DeltaRElcCol;

解释：由ElcCol计算产生的温度。

double t\_DeltaRCouVdw;

解释：由CouVdw计算产生的温度。

//////////////////////////////////////////////////////////////////

// Bond

//////////////////////////////////////////////////////////////////

double LimitDeviationLevel\_Bond;

解释：前面提到条件公式：“最大偏移量”>“极限偏移量”>“最小偏移量”，每个振子的“极限偏移量”是不断变化的，为了综合掌握这个信息，计算一个实时评估参数“极限偏移量平均水平LimitDeviationLevel\_Bond”，将所有的Bond振子的“极限偏移量”在最大最小值之间的平均位置统计一下。

double Mode\_1\_Level\_Bond;

解释：所有的Bond振子中，执行“强制回位操作”的统计比例。每一步的振子位移计算方法分为两种：“强制回位操作”和“一般震荡操作”。

double Max\_ratio\_terminal\_Bond;

int    Max\_ratio\_terminal\_Bond\_idx;

double Min\_ratio\_terminal\_Bond;

int    Min\_ratio\_terminal\_Bond\_idx;

解释：对于每一个Bond振子，完成一步计算后，其当前的实际状态（两原子距离）与平衡位置（固定值，来源于统计）的比值，记为ratio\_terminal\_Bond，然后统计所有的Bond振子，找到最大的（Max\_ratio\_terminal\_Bond），并记下其Bond振子编号（Max\_ratio\_terminal\_Bond\_idx），找到最小的（Min\_ratio\_terminal\_Bond），也记下其Bond振子编号（Min\_ratio\_terminal\_Bond\_idx），

//////////////////////////////////////////////////////////////////

// Angle

//////////////////////////////////////////////////////////////////

double LimitDeviationLevel\_Angle;

double Mode\_1\_Level\_Angle;

double Max\_ratio\_terminal\_Angle;

int    Max\_ratio\_terminal\_Angle\_idx;

double Min\_ratio\_terminal\_Angle;

int    Min\_ratio\_terminal\_Angle\_idx;

解释：类似Bond

//////////////////////////////////////////////////////////////////

// Plane

//////////////////////////////////////////////////////////////////

double LimitDeviationLevel\_Plane;

double Mode\_1\_Level\_Plane;

double Max\_ratio\_terminal\_Plane;

int    Max\_ratio\_terminal\_Plane\_idx;

double Min\_ratio\_terminal\_Plane;

int    Min\_ratio\_terminal\_Plane\_idx;

解释：类似Bond

//////////////////////////////////////////////////////////////////

// ElcCol

//////////////////////////////////////////////////////////////////

int Num\_elccol\_move\_ALL ;

int Num\_elccol\_move\_NONE;

int Num\_elccol\_move\_COL1;

int Num\_elccol\_move\_COL2;

int Num\_elccol\_move\_ELC ;

int Num\_elccol\_move\_OTHE;

解释：ElcCol三种操作（COL1、COL2、ELC）的数量统计

//////////////////////////////////////////////////////////////////

// Dihedral

//////////////////////////////////////////////////////////////////

int Num\_Dihedral\_ALL;

int Num\_valley\_change ;

解释：“dihedral**相变”**的数量统计

};

**2025-01-09**

**当前运行数据，每一步运算后，都可以输出到四个列表，**

DvnnForceBondList \* dvnnBondList

DvnnForceAngleList \* dvnnAngleList

DvnnForceDihedralList\* dvnnDihedralList

DvnnForcePlaneList \* dvnnPlaneList

// copy current data

qomd\_shell3\_copy\_current\_from\_proteinBag\_dvnnBondList(dvnnBondList, (void\*) proteinBag\_1VII);

qomd\_shell3\_copy\_current\_from\_proteinBag\_dvnnAngleList(dvnnAngleList, (void\*) proteinBag\_1VII);

qomd\_shell3\_copy\_current\_from\_proteinBag\_dvnnDihedralList(dvnnDihedralList, (void\*) proteinBag\_1VII);

qomd\_shell3\_copy\_current\_from\_proteinBag\_dvnnPlaneList(dvnnPlaneList, (void\*) proteinBag\_1VII);

**数据结构解释：**

struct DvnnRecord{

此数据结构用于记录当前振子（适用于bond,angle,plane）的数据

int    Mode;

// 0 ; 初始值init

// 1 : 强制拉回computeHarmonic\_get\_normal\_delta\_d\_regressive\_enforce\_constraint

// 2 : 一般震荡computeHarmonic\_get\_normal\_delta\_d\_syntonic

Mode记录当前振子的操作模式，

以下四个值为振子的状态变化值，嵌入到三种数据结构情况：

（bond：两原子间距，单位，埃10-10米）

（angle：三原子夹角，单位，弧度角）

（plane：星型四原子平面角度，单位，弧度角）

一个计算步骤，每个振子的状态变化由两种因素构成，

1来自于振子内部的弹性震荡，（强制拉回和一般震荡）

2来自于振子外部的相互作用，（各个振子之间的力作用）

double Start   ; // 开始值

double Target  ; // 内部目标值，内部的弹性震荡的计算预测值

double Arrive  ; // 内部实际达到值，内部的弹性震荡的实际计算值

double Terminal; // 最终状态值，外部扰动效果叠加后的最终值

};

struct DvnnForceBond {

//////  links

int VertexI;              // 原子编号array index of VertexI, index of the data in List of DvnnAtomicVertexList

int VertexJ;              // 原子编号array index of VertexJ, index of the data in List of DvnnAtomicVertexList

//////  fixed parameters

double req;               // 两原子距离的平衡值r\_equilibrium, bottom of the valley

double qomdFreq;          // 旧频谱数据frequency : wave number (1/cm), 2024-10-02

double hhmoFreq;          // 新频谱数据frequency : wave number (1/cm), 2024-10-02

//////  fixed parameters

double MaxDeviation;      // 最大偏移量add 2025-01-08, maximum deviation, fixed para

double MinDeviation;      // 最小偏移量add 2025-01-08, minimum deviation, fixed para

//////  variant parameters

double LimitDeviation;    // 极限偏移量add 2025-01-08, current deviation, variant para

“最大偏移量”和“最小偏移量”初始设定后不变。

前面提到的满足条件：“最大偏移量”>“极限偏移量”>“最小偏移量”

每个振子的“极限偏移量”在最大和最小之间不断变化的，表示该振子的能量变化。

//////  variant parameters

DvnnRecord Record;        // 解释在前面

};

struct DvnnForceAngle {

这里的解释类似于bond

//////  links

int VertexI;              // array index of VertexI, index of the data in List of DvnnAtomicVertexList

int VertexJ;              // array index of VertexJ, index of the data in List of DvnnAtomicVertexList, center point

int VertexK;              // array index of VertexK, index of the data in List of DvnnAtomicVertexList

//////  parameters

double thetaeq;           // 三原子夹角平衡值，注意这里单位是角度theta\_equilibrium, bottom of the valley, Degree

double qomdFreq;          // frequency : wave number (1/cm), 2024-10-02

double hhmoFreq;          // frequency : wave number (1/cm), 2024-10-02

//////  fixed parameters

double MaxDeviation;      // add 2025-01-08, maximum deviation, fixed para

double MinDeviation;      // add 2025-01-08, minimum deviation, fixed para

//////  variant parameters

double LimitDeviation;    // add 2025-01-08, current deviation, variant para

//////  variant parameters

DvnnRecord Record;        // 注意这里单位是弧度

};

struct DvnnForceDihedral {

这里的解释类似于bond

//////  links

int VertexI;              // array index of VertexI, index of the data in List of DvnnAtomicVertexList

int VertexJ;              // array index of VertexJ, index of the data in List of DvnnAtomicVertexList

int VertexK;              // array index of VertexK, index of the data in List of DvnnAtomicVertexList

int VertexL;              // array index of VertexL, index of the data in List of DvnnAtomicVertexList

//////  parameters

double qomdFreq;          // frequency : wave number (1/cm), 2024-10-02

double hhmoFreq;          // frequency : wave number (1/cm), 2024-10-02

//////  variant parameters

DvnnRecordDihedral Record;        // add 2025-01-08, current deviation, variant para

};

struct DvnnRecordDihedral{

这专门用于dihedral的数据记录

对于二面角结构，实际上是0~360度旋转，形成两个或者三个Valley（波峰波谷结构），从一个Valley变化到另一个Valley，既完成一次角度相变，这个相变是蛋白质构象变化的关键。

Start   ; // 开始值

Target  ; // 内部目标值，内部的弹性震荡的计算预测值

Arrive  ; // 内部实际达到值，内部的弹性震荡的实际计算值

Terminal; // 最终状态值，外部扰动效果叠加后的最终值

int ValleyStart   ; Valley编号值

int ValleyTarget  ;

int ValleyArrive  ;

int ValleyTerminal;

double PhiStart   ; Phi二面角值（注意这里单位是弧度）

double PhiTarget  ;

double PhiArrive  ;

double PhiTerminal;

};

struct DvnnForcePlane {

这里的解释类似于bond

//////  links

int VertexI;              // array index of VertexI, index of the data in List of DvnnAtomicVertexList

int VertexJ;              // array index of VertexJ, index of the data in List of DvnnAtomicVertexList

int VertexK;              // array index of VertexK, index of the data in List of DvnnAtomicVertexList

int VertexL;              // array index of VertexL, index of the data in List of DvnnAtomicVertexList

//////  parameters

double phieq;             // 星型四原子平面角度平衡值，注意这里单位是角度（并且固定180度）phi\_equilibrium, same as gamma2 in GvnnForcePlane;

                              // Degree : Gamma2 is equal to 180 DEG always.

double qomdFreq;          // frequency : wave number (1/cm), 2024-10-02

double hhmoFreq;          // frequency : wave number (1/cm), 2024-10-02

//////  fixed parameters

double MaxDeviation;      // add 2025-01-08, maximum deviation, fixed para

double MinDeviation;      // add 2025-01-08, minimum deviation, fixed para

//////  variant parameters

double LimitDeviation;    // add 2025-01-08, current deviation, variant para

//////  variant parameters

DvnnRecord Record;        // 注意这里单位是弧度

};