

**本科生毕业设计[论文]**

**折叠石墨烯层间力对热导率的影响的研究**

院 系 能源与动力工程

专业班级 热动1110班

姓 名 宋琪琛

学 号 U201111701

指导教师 杨诺

2015年 6月 15 日

**学位论文原创性声明**

本人郑重声明：所呈交的论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包括任何其他个人或集体已经发表或撰写的成果作品。本人完全意识到本声明的法律后果由本人承担。

作者签名： 年 月 日

**学位论文版权使用授权书**

本学位论文作者完全了解学校有关保障、使用学位论文的规定，同意学校保留并向有关学位论文管理部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权省级优秀学士论文评选机构将本学位论文的全部或部分内容编入有关数据进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本学位论文属于 1、保密囗，在 年解密后适用本授权书

2、不保密囗 。

（请在以上相应方框内打“√”）

作者签名： 年 月 日

导师签名： 年 月 日

**摘 要**

近年来，科学界和工业界对于具有优异电学、热学性质的石墨烯十分关注。折叠石墨烯作为一种石墨烯的特殊存在形式，具有不同于完美石墨烯的性质。然而目前对于折叠石墨烯的研究仅集中在小尺寸的折叠石墨烯纳米带上，对于更接近实际尺寸的折叠石墨烯则无人研究。由于石墨烯本身具有热导率随尺寸变化的规律，大尺寸的折叠石墨烯的具体性质可能与折叠石墨烯纳米带的性质不同，需要单独研究。

本文通过非平衡分子动力学对无穷大的折叠石墨烯体系进行了研究。得到了折叠石墨烯热导率的尺寸依赖关系，通过外推法得到尺寸无穷大的折叠石墨烯热导率，以表征实际尺寸的折叠石墨烯的热输运能力。得到了热导率对折叠数目依赖关系。同时对于具有不同的层间距的折叠石墨烯的热导率进行了计算，通过统计原子间作用力所传递的热流解释了热导率随着层间距减小而增大的变化规律，并通过声子功率谱密度分析了折叠处对热阻的贡献和层间作用的影响。

**关键词：**折叠石墨烯；非平衡分子动力学；尺寸效应；范德瓦尔斯力；功率谱密度

**Abstract**

Graphene has attracted the interests of researchers both in academia and industry due to its extraordinary electrical and thermal transport properties. Folded graphene, a derivative of graphene, has different properties from pristine graphene arise from its novel structure. However, most of the related studies are focused on folded graphene nanoribbons of small sizes, while folded graphene of realistic sizes is investigated by no one. Given the fact that there exists a strong size effect on thermal conductivity of graphene, the large-area folded graphene may be a different case and thus should be investigated.

Here, we use non-equilibrium molecular dynamics to study thermal transport in large-area folded graphene. The dependence of folded graphene’s thermal conductivity on length and number of folds is obtained and thermal conductivity of infinitely large folded graphene is attained by extrapolation to characterize the thermal properties of folded graphene of realistic sizes. Cases of various inter-plane distances are studied and the impact of van del Waals interaction on thermal conductivity is investigated by directly recording the heat flux by long-range atomic interactions. The contribution to thermal resistance from the curve part in the structure with varying inter-plane distances is studied by calculating the vibration power spectrum.

**Key Words：**folded graphene; non-equilibrium molecular dynamics; size effect; van del Waals interaction; vibration power spectrum

**目 录**

**摘要** Ⅰ

**Abstract** Ⅱ

**1 绪论** 1

1.1 研究背景、目的和意义 1

1.2 国内外研究概况 2

1.3 分子动力学简介 3

1.3.1 模拟过程 4

1.3.2 物理量计算 5

1.3.3 控制模拟进行的条件 5

1.3.4 从模拟中获取声子的行为 7

**2 折叠石墨烯模拟细节** 8

2.1 模拟对象结构及边界条件 8

2.2 模拟的参数设置 9

2.2.1 势函数的形式 9

2.2.2 模拟过程的设置 10

2.3 热导率的计算 10

2.3.1 温度梯度的计算 11

2.3.2 热流的计算 11

**3 模拟结果及讨论** 13

3.1 折叠石墨烯中的温度梯度 13

3.2 无穷大体系的热导率 14

3.3 不同层间距离的折叠石墨烯的热导率 17

3.4 层间力对热导率的影响：从层间热流分析 18

3.5 层间力对热导率的影响：从功率谱分析 19

**4 结论** 22

4.1 总结与展望 22

4.2 可改进的部分 22

**致谢** 23

**参考文献** 24

**附录** 28

S.1 热导率计算的处理细节 28

S.2 声子态密度的计算 29

S.3 程序源代码 29

**1 绪论**

**1.1 研究背景、目的和意义**

煤、石油等化石能源的日渐枯竭迫使人们寻求有效可行的替代方案。以中国为例，煤炭占一次能源消耗的七成。热电厂在发电过程中不仅消耗了大量煤资源，同时产生了大量的废热。内燃机燃烧过程中有60-70%能源以废热的形式被浪费掉[[1](#_ENREF_1)]。中国机动车数量超过2.5亿，这些机动车向环境排放了大量的废热。化工厂中进行的化学过程也伴随着大量废热产生。为了利用这些废热，提高总的能源利用效率，基于热电温差发电的余热回收利用吸引了研究人员的目光。

热电发电技术可以将热能直接转化为电能，并且没有任何污染。当前热电发电技术的主要研究方向是找到热电转换效率较高的材料[[2](#_ENREF_2)]。衡量热电材料的一个重要指标是材料的*ZT*因子，通常*ZT*因子越大，热电转换效率越高。它与材料的塞贝克系数*S*、热导率*κ*、电导率*σ*有关，可以表示为*ZT* = *S*2*σT*/*κ*。由公式可以看出，通过调控材料的输运性质提高*ZT*因子可以提高材料的热电转换效率。

石墨烯是具有*sp2*杂化电子轨道的单层碳原子结构，具有绝佳的电学性质。石墨烯中电子在狄拉克附近的群速度十分大，约8.5 × 105 m/s,使得石墨烯具有卓越的电子输运性质,实验[[3](#_ENREF_3)]证明了石墨烯具有较高的电导率，约5.3× 10-4 ohm−1。通过改变石墨烯的结构带来的量子约束以在石墨烯能带中产生带隙，为将石墨烯应用于下一代微电子和热电装置带来可能[[4](#_ENREF_4)]。石墨烯中的声子平均自由程达到微米级别，使得石墨烯具有优异的热输运性质。通过拉曼光谱实验测量的热导率约2000−5000 W mK−1[[5](#_ENREF_5)],通过电学测量得到石墨烯热导率约为600W mK−1(300K)[[3](#_ENREF_3)]。不同研究小组得到的实验值差异较大。理论计算得到的石墨烯热导率比天然物质中热导率(约2000W mK−1)最高的金刚石还要高[[6](#_ENREF_6)]。对于石墨烯材料的塞贝克系数，实验得到数值的很小，在300K时约8.2 μV*/*K [[7](#_ENREF_7)]。

为了提高石墨烯热电材料的*ZT*因子，设法提高塞贝克系数是一种方法，而调制石墨烯的热导率也是一种方法（本课题专注于这种方法）。过去几年，人们在显微镜下发现了折叠石墨烯的存在[[8](#_ENREF_8)]；随后，研究人员找到了控制形成这种折叠结构的方法[[9](#_ENREF_9)]（如图1-1）。研究表明，在石墨烯结构中创造折叠是一种有效降低石墨烯热导率的方法[[10](#_ENREF_10), [11](#_ENREF_11)]。

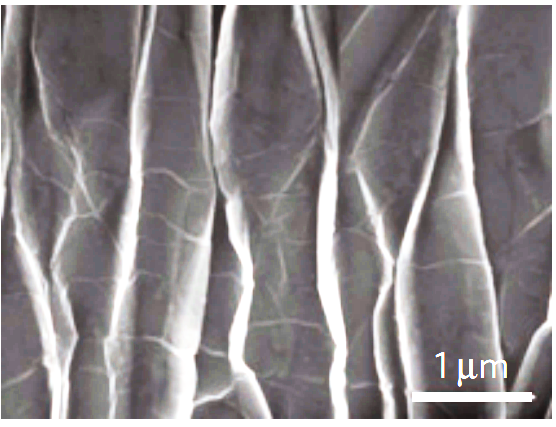


图1-1 利用基底释放应力制备的折叠石墨烯的扫描电子显微镜（SEM）图像[[9](#_ENREF_9)]。

**1.2 国内外研究概况**

对于石墨烯热导率的研究方法主要分为实验和模拟两种方法。多层折叠的石墨烯在结构上与多层石墨烯有相似之处。美国研究人员Jang等人[[12](#_ENREF_12)]通过拉曼测量的方法得到了随着多层石墨烯层数减小而增大的热导率，产生这种现象的原因是层数变小导致了减小了的低能量声子层间耦合和逐渐变弱的层间散射（主要是声子Umklapp散射，指的是三声子散射中，两个声子碰撞产生第三个声子越过第一布里渊区的一种散射）。

目前对折叠石墨烯的研究主要以模拟为主。常见的模拟方法是解声子玻尔兹曼方程、分子动力学等。折叠石墨烯中的弯折部分与碳纳米管的结构有相似性。Lindsay等人[[13](#_ENREF_13)]发现碳纳米管的热输运性质与其曲率相关，并且得到了热导率随着曲率变小先减小后增大并最终趋于常数的规律。他们对此的解释是：曲率较大时，随着曲率减小，半径增大，弹道输运（ballistic transport）的特性越来越不明显，热导率降低；曲率较小时，由于弯曲打破了石墨烯原本有的反射对称性（reflection symmetry）从而使得选择准则（selection rule）更加复杂，曲率越小，声子散射越弱，热导率越高，最终趋于石墨烯的热导率。但是求解声子玻尔兹曼方程的局限性在于只能处理较小的体系；玻尔兹曼方程默认声子输运是满足粒子特性的，在声子平均自由程与系统特征尺寸相近时，则必须考虑声子的波动性。与多层石墨烯、碳纳米管的类比给我们带来了启示：石墨烯结构变异和层间作用对声子散射的贡献。

国内研究人员Yi[[14](#_ENREF_14)]等人利用分子动力学（Molecular Dynamics, MD）分析了具有折叠结构的石墨烯展开变为平面（planar）石墨烯的临界温度，但是没有研究其热输运性质。Xie等人[[15](#_ENREF_15)]通过格林函数（Green’s function）的方法得到了折叠石墨烯的电输运性质。Yang等人[[10](#_ENREF_10)]通过非平衡分子动力学（non-equilibrium molecular dynamics, NEMD）计算了折叠石墨烯纳米带的热导率，并找到了热导率对层间距离以及折叠数目的依赖关系，即层间距离越小，折叠数目越多，热导率越低；通过非平衡格林函数（non-equilibrium Green’s function, NEGF）分析了声子输运情况，并用相图分析了散射增强的原因，即折叠石墨烯的色散关系说明其具有更大的Umklapp散射的概率；然而模拟的体系比较小且结果没有具体呈现出层间力的效应。Zhang等人[[11](#_ENREF_11)]通过与同等表面积的碳纳米管作对比，采用NEMD研究了折叠石墨烯纳米带的热导率，辅以振动本征矢分析（vibrational eigen-mode analysis），探讨了折叠石墨烯边缘对热导率的影响，得到具有不同边界的石墨烯纳米带，热导率相差较大；然而他们的计算并没有考虑多层折叠的情况。

总的来说，采用实验方法对折叠石墨烯热导率的探索很少，而模拟的工作相对丰富。上述采用了分子动力学的研究的一大优势是，能更加直截了当地了解具有结构差异的折叠石墨烯的热输运性质与结构的关系。但几乎所有的模拟都在研究较小的模拟单元（纳米带），对于表征实际尺寸的折叠石墨烯的热导率则无能为力。本课题将通过分子动力学处理相对较大的体系，并采用合理的推论，得到更接近真实尺寸的折叠石墨烯热输运的性质。同时将对层间作用力对于声子输运和热导率的影响作用进行详细探讨，探索通过折叠降低石墨烯热导率的背后机制。

**1.3 分子动力学简介**

经典分子动力学（classical molecular dynamics simulations）是模拟原子间相互作用的时间演化过程的一种计算方法，它主要遵守经典力学中的牛顿第二定律，

 （1-1）

其中，*i*是原子编号，*mi* 是第*i*个原子的质量，**a***i* 是加速度，**a***i* = *d*2 **r***i*/*dt*2 **，F***i*是原子所受作用力。给出原子的初始位置和速度，通过势函数可以求得每个原子所受的力；根据原子受力情况计算加速度就可以计算下一步原子的位置和速度。EMD[[16](#_ENREF_16)]依据处于平衡态时能量在系统中的涨落，记录系统的自相关函数，采用Green-Kubo方法来计算热导率。NEMD则通过在系统中创造非稳态获得系统的物理性质，求热导率的通常做法是通过创造温度梯度和热流。

本次模拟计算采用了NEMD，主要原因是NEMD模拟的是非稳态的输运性质，且折叠石墨烯平面内输运能力最强，即便产生了折叠，热流依然主要沿着层内输运，这个时候利用傅利叶定律定义的计算得到的热导率更能表征这种平面内的输运能力。

**1.3.1 模拟过程**

（1）设置原子初始位置和速度

一般原子的初始位置设置为平衡位置，初始速度按照某种方式赋值。常用的一种方式是按照某一温度*Tinit*下产生遵循高斯分布的随机数，则初始速度是的体系满足初始温度*Tinit*。一般*Tinit*=2*T*0，*T*0是处于平衡态时模拟所处的温度。这样做是因为初始状态势能为零，动能等于内能，随着模拟的进行，势能和动能趋向于相等且各自等于内能一半；取两倍关系符合能量守恒，使得模拟更快达到稳态。

（2）模拟原子的运动

MD的核心部分是势函数，它的选取决定了计算的准确性。力可以通过势函数相对于原子位移的梯度得到，

 （1-2）

*V*是势函数，可能是采用经验势函数，也可能利用密度泛函理论（DFT）计算修正势函数。通过牛顿第二定律得到每个原子的加速度**a***i*。利用Velocity-Verlet算法[[17](#_ENREF_17)]可以得到每一步的原子的速度和位置。具体的计算过程如下，

 （1-3）

 （1-4）

 （1-5）

 （1-6）

**1.3.2 物理量计算**

MD是统计力学方法的一个分支，如果用Γ表示研究体系内各个原子的坐标和动量，用*H*(Γ)表示体系的哈密顿量（Hamiltonian），则一个微正则系综满足*δ*(*H*(Γ)*E*)，*δ*是狄拉克(Dirac)函数。该关系保证系统处于能量为*E*的状态。对于正则系综，温度*T*为常数，分布满足玻尔兹曼(Boltzmann)函数的形式exp(－*H*(Γ)/*k*B*T*)。统计物理指出，物理量可以由系综平均值得到，

 （1-7）

其中，为系统某一时刻一系列动量和位置。 分子动力学满足杜龙—佩蒂特极限（Dulon-Petit limit, *Cp*=*3NAkB*），温度可表示为，，*α*为坐标轴方向。值得注意的是石墨烯的德拜温度很高（平面内模式为2300，Z模式 1287 K）[[18](#_ENREF_18)]，远高于室温，因此室温下的MD模拟是不能正确表征系统的热容的。但是最后计算得到的热导率与实验值相差不大，所以这种MD模拟又是可取的[[19](#_ENREF_19), [20](#_ENREF_20)]。

**1.3.3 控制模拟进行的条件**

（1）边界条件

受制于计算资源所限，MD所能研究的体系是有限的，通常在纳米尺度。有限的系统必然存在边界，且边界的影响常常是显著的。对于固定边界（fixed boundary）处于固定边界的原子动能为零，在整个模拟过程中，边界原子的位置固定不变。对于自由边界（free boundary），处在边界的原子没有被人为施加约束，与边界内部原子没有本质区别，但是通常在边界的原子的邻居数量少于内部原子。对于周期性边界（periodic boundary），处在边界的原子被人为施加了与其他原子的作用。

如图1-2，对于具有周期性边界条件的模型，一个处在边界的原子a不仅与内部原子有作用，同时与跨过整个研究体系的另一个边界的原子b具有相互作用。相当于一个边界原子与一个“虚拟”的原子b’具有相互作用，这个“虚拟”的原子b’的速度、加速度与跨过整个体系的另一个原子b完全相同，但是位置坐标相差一个固定值*L*（模拟系统的某一维度的长度）。这样就避免了边界原子邻居数目少于内部原子的问题。

b’

a

b

*L*

图1-2 周期性边界示意图，虚线表示具有作用力的原子对

（2）热库

NEMD需要在系统中产生温度梯度，热库可以保持这种温度差。热库体现为动力学方程之外的力学作用项。一种常用的一种热库叫做Nosé-hoover热库[[21](#_ENREF_21), [22](#_ENREF_22)]。如图3，以一维原子链为例，处在高温和低温Nosé-hoover热库中的原子[[23](#_ENREF_23)]分别满足，

 （1-8）

式中，下标*H*和*C*分别代表高温和低温热库中的原子，**p***i,*H/C表示原子的动量，**F***i,*H/C表示由势函数计算得到的力，参数*ζi,H/C*满足，

 （1-9）

式中，*θH/C*是原子与热库的耦合强度，*mi,H/C*是原子的质量，*Ti,H/C*。通过公式，

可以记录热库的能量损失/能量获得作为系统热流，其中***f****i,H/C*是热库对应的附加力项。



**TH**

**TC**

图1-3 一维原子链中的相互作用示意图，左侧和右侧分别被施加了热源和冷源

**1.3.4 从模拟获取声子的行为**

（1）声子态密度谱和功率谱

MD模拟的输运过程，实际模拟了声子输运的过程。声子态密度（density of states, DOS）描述了每种能量可能占有的态的数目。某个能量的DOS越大，该能量下可能占据的态越多。DOS为零则表示这个能量下不存在任何声子态。一段连续的DOS为零的能量范围表示了存在一个声子带隙（bandgap）。声子态密度谱能够展现系统中原子振动满足的规律，帮助了解系统的输运性质。

通过计算原子速度的自相关函数γ (t)可以得到DOS，

 （1-10）

对速度自相关函数进行傅利叶变换即可得到归一化的频谱。

Wiener-Khintchine 理论[[24](#_ENREF_24)]证明，对于广义平稳（Wide-sense stationary, WSS）过程，其自相关函数的傅利叶变换是功率谱密度（Power Spectrum）。功率谱密度由速度的傅利叶变换得到，

 （1-11）

其中，是系统的最高频率。而功率谱密度也可以表示为，

 （1-12）

其中，*f* (E)是分布函数。声子满足Bose-Einstein分布，但是经典MD模拟，基于高温假设，因此有，

 （1-13）

（2）声子平均自由程

不论是电子还是声子，其输运能力由载流子的散射决定，没有任何的散射的输运称为弹道输运，这种情况下电导率/热导率的大小不会被约束。声子在输运过程中会发生散射，正是这种散射产生了热阻。通过气体动理论（kinetic theory of gases）可以得到声子输运对应的热导率为 ，其中*n*为单位体积内的载声子浓度，*cv*是比热，*u*是声子群速度，*λ*是声子平均自由程，，*τ*是连续两次碰撞发生的平均时间间隔，被称为弛豫时间（relaxation time）[[25](#_ENREF_25)]，因此可以将平均自由程理解为声子在相邻两次碰撞间输运的平均距离。

**2 折叠石墨烯模拟细节**

**2.1 模拟对象结构及边界条件**

如图2-1，折叠石墨烯的结构是通过在一个等长度的石墨烯上加上若干个弯折得到的。y方向为宽度方向，z方向为垂直于平面层的方向。折叠石墨烯的长度被定义为，L = (n+1) Lplane + n Lcurve，n是折叠的数目，Lplane和Lcurve 分别是平面部分和弯折部分的长度。图2-2 (a)是一个总长为*L*（共300层原子，*L*=36.7 nm）的折叠石墨烯的结构示意图。所有的弯曲部分都是由5层原子组成的，因此Lcurve≡(5+1) ，其中a为石墨烯的晶格常数，a=1.418Å。在未进行模拟时，弯曲部分的原子在侧视图中处在一个同心圆上，自身构成一个圆内接正十二边形的一半。初始时，相邻的平面之间的距离为0.474 nm。

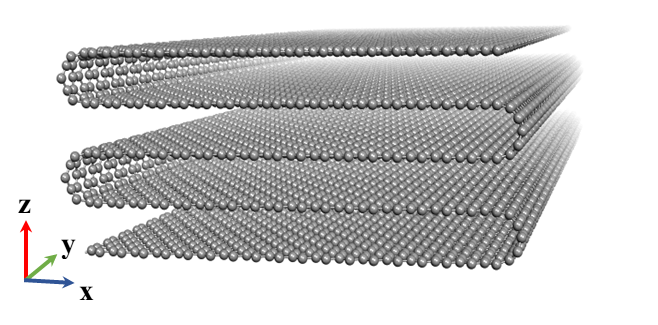


图2-1折叠石墨烯的三维视图

折叠石墨烯上下各有一层基底，以保证石墨烯在z方向没有漂移。同时第一层原子和最后一层原子在x，y方向上被设置了固定边界条件，以防止石墨烯结构出现较大的变形和在x，y方向上漂移。最靠近第一层的五层原子（第2到第6层原子）被施加高温热浴，温度维持在Th = T0 (1+∆)。而最靠近第N层原子的五层原子（第N5到第N1层原子）被施加高温热浴，温度维持在Th = T0 (1∆)。本次模拟T0=300K，∆=0.1。值的注意的是，在这种设置下，沿着*L*的方向温度梯度约为108 K/m，这显然是无法在现实中实现的，但是在这种非真实的模拟设置下得到的结果与实验值依然是吻合的[[19](#_ENREF_19), [20](#_ENREF_20)]，因此依然得到了采用。

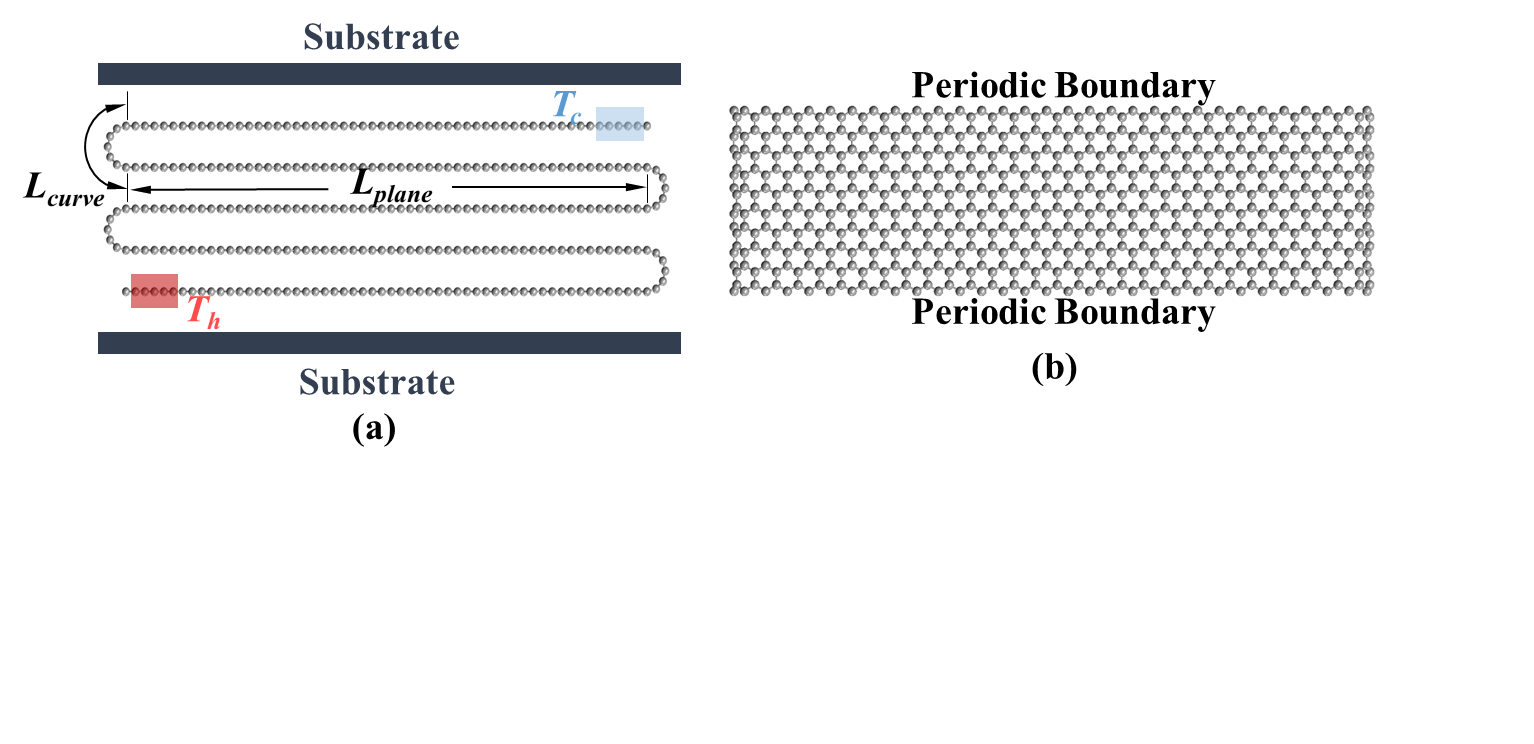
如图2-2(b)，为了获得无限大的折叠石墨烯的性质，宽度方向上（y方向）使用了周期性边界。结构的宽度为21.27 nm（10个C原子）。之所以选取这个宽度，是因为任何大于该宽度的结构的热导率与该宽度下的结构热导率几乎没有区别，即热导率在宽度方向上在21.27 nm时已经收敛。因此，从图2-1中也可以看出，在宽度方向石墨烯是无限延伸的。

图2-2 模拟体系示意图 (a) 弛豫前的折叠石墨烯的结构，Lplane, Lcurve分别为6.75 nm和0.737 nm，总长度L为36.7 nm，折叠数目为4折。靠近两端分别被施加了热源（红色）和冷源（蓝色），上侧和下侧分别有一层基底模拟基底的作用。最两端原子在x, y方向被固定。(b) 宽度为2.13 nm，宽度方向上使用了周期性边界。

**2.2 模拟的参数设置**

**2.2.1 势函数的形式**

MD模拟中，势函数的选择直接决定了模拟结果的精度。本次模拟使用了Morse[[26](#_ENREF_26), [27](#_ENREF_27)]势函数来描述最近邻碳原子之间的成键作用，是一种二体力作用势；使用了简谐余弦势函数（harmonic cosine energy potential）[[28](#_ENREF_28)]来描述中心原子*i*及两个最近邻原子*j*，*k*形成一个夹角*θijk*，这个夹角在平衡位置处等于*θC* = 120*°*，是一种三体作用势；采用了Lennard-Jones 12-6 势函数来描述碳原子间长程作用，例如处于不同平面层原子间的范德瓦尔斯（van der Waals）相互作用[[14](#_ENREF_14)]。综合上述三种势函数，最终采用的势函数形式如下，



 （2-1）

右手边，第一项表示只与最近邻的成键两原子间距离相关的Morse势函数，第二项表示只与原子*i*，*j*，*k*夹角的余弦相关的简谐余弦势函数，第三项表示只与具有范德瓦尔斯力的原子间距离相关的Lennard-Jones势函数。力常数*KCr* = 478.9 kJ mol-1， *KCθ* = 562.2 kJ mol-1， *εCC* = 0.2880 kJ mol-1；平衡位置几何参数 *rc* = 1.418Å，*θC* = 120°；范德瓦尔斯力的参数*σCC* = 0.3407 nm；其他参数*γ* =2.187 Å-1。

**2.2.2 模拟过程的设置**

所有进行的模拟都采用了Verlet算法。时间步长被取为0.5 fs，模拟的步数为6×106步，总的模拟时间为3 ns。在最初的3×105步，上下两个基底的Z方向位置被不断向着压缩内部的折叠石墨烯的方向运动。对于演化过程的每一步，压缩的距离约为10-6 nm，相对于原子的位移（约10-4 nm）较小，避免了大的形变甚至结构中的键断裂。经过这样的压缩过程，折叠石墨烯平面层间的距离从最初的0.474 nm变为0.35 nm。此后，基底的位置不再变化，折叠石墨烯平面层间的距离基本维持在0.35 nm。为了探索不同的层间力的带来的不同的输运性质，进行了层间距离最终被压缩到0.30 nm的一组模拟。

采用了12组不同的初始条件，初始的速度按照麦克斯韦分布分布进行初始化，原子初始位置都在平衡位置。使用MPI程序接口在12个CPU上同时进行计算，最后对12个核心所计算出的结果进行平均。

**2.3 热导率的计算**

NEMD中常采用傅利叶定律计算热导率，其形式如下，

 （2-2）

其中，*J*为系统中的热流，单位为W；是温度梯度，单位为K/m；*A*为横截面积，单位为m2。在这里*A*为宽度与厚度的乘积，一般石墨烯的厚度被定义为0.335 nm（与石墨的层间距一致），因此A=0.713nm2。

**2.3.1 温度梯度的计算**

处在热浴中的原子的温度是恒定的，由于高温和低温热浴施加在系统的两端，因此系统内部将形成一个温度梯度。但是由于采用的Nosé-hoover热库是一种人为的作用，在热源附近经常会形成较大的温度下降（附录图 -1）。为了减小这种影响，选择第六层原子所处的温度为*Th*，第N-5层所处的温度为*Tc*。

 （2-3）

其中，每层原子的温度由1.3.2中的公式得到，即第*i*层的原子温度为，

 （2-4）

式中的1/3表示三个维度的温度的平均。

**2.3.2 热流的计算**

NEMD常常采用通过热源记录热流的方法，热源单位时间所做的功就是系统中的热流。1.3.3中描述了通过热源的记录热流的方法。在热源处记录的单位时间的能量失去为ε1，冷源处记录的单位时间的能量获得为ε2，则热流J=(ε1 +ε2)/2。

注意到，层间的范德瓦尔斯作用可以直接参与热输运，把热量从热源送到冷源。根据公式，可以通过记录范德瓦尔斯力与速度的乘积原子间通过范德瓦尔斯力传递的热流，具体形式如下[[29-31](#_ENREF_29)]，

 （2-5）

其中，Qi🡪j表示稳态时从i原子传递到j原子的能流，**F**ji是原子j受到的原子i施加的的作用力（**F**ji = - **F**ij），**v**i，**v**j分别是第i个原子和第j个原子的速度。i原子和j原子分别处在所研究的热流通过的横截面两侧。如果单独统计范德瓦尔斯力所传递的热流，则只需要令**F**ji为i原子和j原子间的范德瓦尔斯力，则系统中由范德瓦尔斯力传递的能量流为，a，b是横截面两侧的原子数量。

为了研究层间作用对于热输运的影响，可以直接记录由范德瓦尔斯力所传递的热流。如图2-3所示，箭头表示的是通过层间范德瓦尔斯力直接传递的热流，热流从截面一侧传递到另一侧。尽管研究的折叠石墨烯存在总长变化或者折叠变化的情况，但是研究体系具有很强的相似性，因此统一将假想的热流截面选取在第二层平面和第三层平面之间。

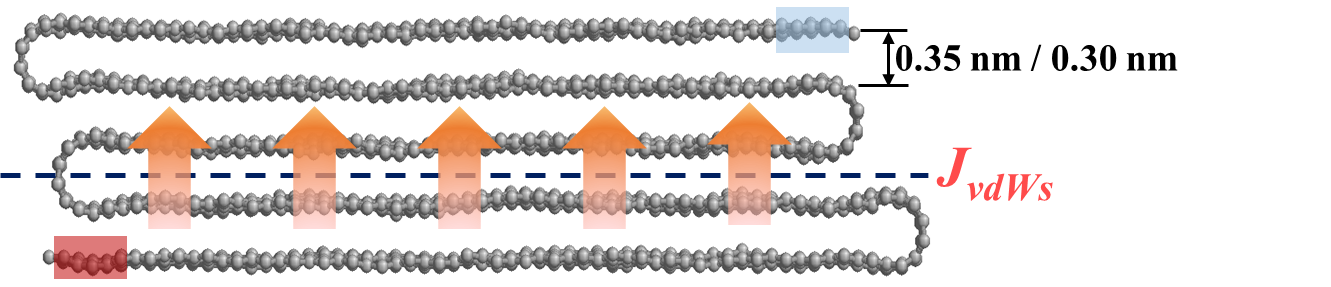


图2-3 通过式2-5记录通过某假想截面的热流的示意图。虚线表示热流通过的截面，处在自下向上第二个平面和第三个平面部分之间。箭头表示通过层间范德瓦尔斯力直接传递的热流JvdWs。对于不同层间距、不同尺寸或折叠数目的情况，虚线位置都取在自下向上第二、三平面部分之间。

**3 模拟结果及讨论**

**3.1 折叠石墨烯中的温度梯度**

图3-1是在如上述设置的模拟条件下最终形成的温度梯度，横坐标是折叠石墨烯的长度。两侧施加了热浴的部分，温度梯度保持为零。内部原子沿着长度方向形成了温度梯度，依据傅利叶定律，将在长度方向上产生热流。具有弯折的折叠石墨烯中的温度没有呈现像二维结构的完美石墨烯呈现出线性的分布，而是呈现出阶梯状的分布。在弯曲部分（8 nm，15 nm，22 nm，29 nm各自附近），温度梯度的值更大。在水平部分，温度随着长度的增大变化很小，接近水平，表示平面

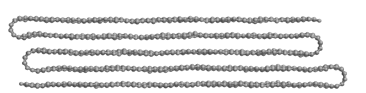


图3-1 折叠石墨烯中沿着长度L的方向的温度梯度，其长度L为36.7 nm，折叠数目为4。点划线表示层间距为0.35 nm的折叠石墨烯，实线表示层间距为0.30 nm的折叠石墨烯，虚线表示相同总长的完美石墨烯。小图是弛豫后的结构，弯曲部分位于8 nm，15 nm，22 nm，29 nm各自附近。

层内的温度变化很小，热导率相对于弯曲部分更大。随着层间距减小，层间作用的增强，这种阶梯状分布更加明显，平面部分的温度差异更小，甚至在局部出现相反的温度梯度。

平面部分中的这种近似水平分布的温度可能源自于折叠结构引入的层间作用。相邻两平面层内各自有下降的温度分布，各自有热流输运。然而层间的范德瓦尔斯力也可以传递热流，且由于相邻平面层的原子比处在同一平面层的原子的温度差异更大，某一层的高温部分和相邻一层的低温部分直接接触，将有一定比例的热流直接从层间传递。由于平面层间热流的存在，平面层内的温度梯度被削弱。当层间距减小，层间作用增加时，层间热流可能增大，使得平面层内的梯度进一步削弱。原先平面层的低温部分可能因为与相邻一层的高温部分接触，温度变得比同层内其他位置高，这解释了相反的温度梯度的出现。下文将利用温度梯度求得热导率，并分析层间作用对热导率的影响来证明上述假设。

**3.2 无穷大体系的热导率**

在NEMD模拟中，模拟得到的结果常常是具有很强的尺寸依赖性。对于尺度较小的系统，热导率通常与尺寸有关，并且热导率的数值被尺寸的大小所限制。这是因为声子的平均自由程收到了系统边界的约束。这种规律被称作卡斯米尔极限(Casimir limit) [[32](#_ENREF_32)]。系统的实际平均自由程可以表示为[[20](#_ENREF_20)]

  （3-1）

如果认为声子群速度vc与尺寸无关，则由1.3.3中式知，

 （3-2）

这里，l∞是无穷大的系统中的声子平均自由程，lsys是系统的尺寸, κ∞是无穷大系统的热导率。由公式可以看出，1/κ ~ 1/lsys。通过模拟具有不同lsys的体系，得到1/κ与1/lsys的关系，用直线拟合所得到的热导率数据，采用外推的方法的到1/lsys为零的情况，即得到了1/κ∞。

对于一维纳米线，lsys通常是纳米线的直径[[33](#_ENREF_33)]。对于二维的薄膜，lsys通常是薄膜的厚度的两倍[[33](#_ENREF_33)]。对于三维系统，lsys通常是模拟体系的尺寸L的四分之一[[20](#_ENREF_20)]。（这里*lsys*没有呈现与维度相关的规律性是因为，讨论一维、二维系统时，使用了面积与体积比表征散射的强弱；讨论三维系统时，则是通过模拟具有不同大小的周期性边界的三维体系得到的结论）。对于我们研究的折叠石墨烯，尽管是三维体系，但是不具有各项同性，不能够简单地直接认为lsys= L/4，因此为了得到无穷大的折叠的热导率，必须找到正确的lsys的表达形式。

如图3-2，热导率的倒数1/κ与Lplane的倒数1/Lplane成线性关系。对于某一个固定总长L的折叠石墨烯，当折叠数目n变大时，由L = (n+1) Lplane + n Lcurve知，Lplane将变小，1/Lplane将变大。不同颜色代表不同总长度L的折叠石墨烯。同一总长度下，n=4，5，6和7，对应着同一种颜色的自左向右四个数据点。对于某一固定长度的折叠石墨烯，折叠数目的增加，Lplane减小时，热导率减小，这与Yang[[10](#_ENREF_10)]的结论一致。相应颜色的直线表示对数据点的线性拟合，这种线性关系表明对于固定总长L的折叠石墨烯，热导率κ ~ Lplane。



图3-2 层间距为0.35 nm的折叠石墨烯热导率的倒数与平面部分尺寸Lplane的倒数的关系。不同颜色的点代表具有不同总长L的折叠石墨烯，L的取值有四个，为36.7 nm, 49.0 nm, 61.3 nm和73.6 nm。对于相同L取值的情况，注意到Lplane的数值的变化是因为折叠的数目发生了变化。

这种线性关系可能是源自于声子自由程被边界所限制。石墨烯的声子平均自由程约102 nm[[34](#_ENREF_34)]。所研究的体系x方向上最大的尺寸不超过15 nm。如果认为此时系统的平均自由程被x方向的边界所约束，那么平面内的声子输运具有弹道输运的性质，热导率随着尺寸增大而线性增大[[34](#_ENREF_34), [35](#_ENREF_35)]。

注意到，对于一个总长L固定不变的石墨烯，Lplane的取值是离散的，依赖于L和n的取值。上述线性关系外推至Lplane趋于无穷大的情况需要L趋于无穷大，而这是模拟所不能实现的。此外，从图中可以明显看出，对于具有相同的折叠数目n的折叠石墨烯，L的取值不同时，热导率是不同的。因此，图3-2揭示了Lplane影响热导率κ的因素之一，但不是唯一因素，总长L也是影响因素之一。

图3-3 黑色空心五边形数据点表示具有相同平面长度Lplane，不同折叠数目的折叠石墨烯热导率，从右向左n取值为2 ~ 7，层间距为0.35 nm。黑色线为数据点的线性拟合，表示了折叠石墨烯热导率的倒数与总长度L的倒数的关系。蓝绿色菱形空心点表示不同长度的完美石墨烯的热导率的倒数，蓝绿色线为数据点的线性拟合，表示二维结构的完美石墨烯热导率的倒数与长度的倒数关系。

由图3-3可以看到，折叠石墨烯的热导率的倒数与总长度L成正比。注意到在这里，L的增大的同时，折叠的数目n也在增大。L越大时，热导率越大。这与上文提到的NEMD模拟中常见的尺寸效应吻合，此时lsys=l。折叠石墨烯的热导率比对应长度的二维结构的完美石墨烯要低。当L趋于无穷大时，完美石墨烯热导率为313 W/mK，这与实验得到的具有基底的石墨烯热导率基本一致[[3](#_ENREF_3)]；折叠石墨烯热导率为241 W/mK。平面长度为6.75 nm的无穷多折叠、无穷长的折叠石墨烯热导率比完美石墨烯降低了约24 %。且当尺寸越小时，降低的越明显，从拟合后的直线来看，最多能降低45 %（L = 21.7 nm时）。

**3.3 不同层间距离的折叠石墨烯的热导率**

折叠石墨烯的热导率的大小对于层数的依赖关系如图3-4所示。图3-4(a)是0.35 nm层间距时，折叠石墨烯的热导率与层数的关系。不同形状的数据点表示不同的总长L的折叠石墨烯的热导率。四条参考线自下而上依次是长度为36.7 nm，49.0 nm，63.3 nm，76.6nm的完美石墨烯的热导率。对于某一个长度的石墨烯，热导低于完美石墨烯的热导率，与上文结论一致，且折叠数目越多，热导率越低。右边是层间距为0.30 nm时，热导率的下降相对于0.35 nm的情况不明显。因此对于具有相同总长度的石墨烯和折叠石墨烯，κplanar > κfolded, 0.30nm > κfolded, 0.35nm。这与[[10](#_ENREF_10)]中热导率随着层间压缩增大而减小的结论是不同的。折叠石墨烯的尺寸比折叠石墨烯纳米带要大很多，出现这种差别可能是层间作用的差别：大面积的折叠石



**(b)**

**(a)**

图3-4 层间距离为 (a) 0.35 nm， (b) 0.30 nm的折叠石墨烯的热导率与折叠的数目的关系，作为参考的水平线是相同总长L的平面结构的完美石墨烯的热导率。

墨烯中的层间范德瓦尔斯力作用虽然弱，但是由于面积较大，依然可以产生较大的层间热流，有利于热的输运。下面将具体对层间热流进行讨论，证明这种假设。

**3.4 层间力对热导率的影响：从层间热流分析**

折叠石墨烯热导率随着层间力增大而增大的结论与纳米带中的结论恰好相反，是因为层间作用对于热输运的贡献变得不可忽视。图3-5表示长度为36.7 nm的折叠石墨烯中，总热流与范德瓦尔斯力传递的热流随着折叠的数目的变化。可以看到，折叠数目越多，系统中的总热流越小，这与上一节中减小的热导率趋势保持一致。层间热流随着折叠数目的增多，呈现大致减小的趋势（层间距为0.35 nm的n=5石墨烯比n=4和n=6的情况都要小，这可能是由于记录层间作用时只选取了一个虚拟截面，导致记录的热流与真正的层间热流有微小差异）。当层间距减小时，层间热流明显增加，直接导致总的热流增大。且层间热流占总热流的比例也明显增加，表示增强了的层间作用使得层间输运对热导率的贡献增加。热流增大意味着有更多的热输运通道存在，即声子发生的散射更少。下面从功率谱密度来分析层间作用的影响。



图3-5 系统内总热流（total）和层间范德瓦尔斯力传递的热流（vdWs）随着层间距离的变化（0.35 nm，0.30 nm），以及随着折叠数目的变化

**3.5 层间力对热导率的影响：从功率谱分析**

声子的功率谱密度也叫振动功率谱，它表示单位频率的声子输运的功率。如果对于某个频率，声子的功率谱密度较大，那么该频率对应的声子占据的态的数目更多[[36](#_ENREF_36)]，如果声子功率谱密度为零，则表示系统中不存在该频率的声子。图3-6是折叠石墨烯中不同位置的原子的功率谱密度，图3-6 (a)、(b)分别是是压缩距离为0.35 nm和0.3 nm的折叠石墨烯平面部分和弯曲部分的功率谱密度。折叠石墨烯的总长为，36.7 nm，折叠数目为4，平面部分为自下向上第二个平面层中的60个原子的数据求平均得到，弯曲部分由自下向上第二个弯折处的60个原子得到。

当层间距离为0.35 nm时，平面部分和弯曲部分的功率谱密度有明显的不同。弯曲部分的低频部分更加平缓，主要分布在0 ~ 20 THz的频率范围内，平面部分的低频部分更加陡峭，主要分布于0 ~ 10 THz的频率范围内，而10 ~ 20 THz对应的功率谱密度则极小，这个范围内弯曲部分的功率谱密度的局部最高值。声子从经过平面/弯曲交界处时，必须通过散射改变自身频率才能通过。由于声子满足Bose-Einstein分布，低频声子对于热输运的贡献更大，因此低频部分的差异对于热阻的增加有着主要贡献。高频部分，尽管平面部分和弯曲部分的功率谱形状上相近，但是尖峰所处的位置有所不同：弯曲部分的尖峰相对于平面部分朝着低频方向移动（红移），这也会导致部分声子发生散射，降低了热导率。因此，具有弯折结构的石墨烯与完美石墨烯相比，因为在弯折/平面交界处存在声子输运的失配，热导率更低，这也解释了图3-1中弯曲部分更陡峭的温度分布。

当层间距离为0.30 nm时，平面部分的功率谱密度曲线与弯曲部分的相比，尖峰所处位置一致，形状大致一致。注意到弯曲部分的功率谱尖峰高度低于平面部分，说明平面部分的声子色散更弱。在7 ~ 10 THz的频率范围内，平面部分的功率谱密度更小，这保证两条曲线的面积积分是一致的。层间距为0.30 nm的情况比起0.35 nm的情况，声子的失配情况更少，系统具有更少的散射，因此热导率稍高。但是，折叠的存在使得散射必然存在，因此热导率依然比完美热导率低。

层间距从0.35 nm减小至0.30 nm，平面部分的变化主要出现在低频0 ~ 10 THz的两个尖峰被压缩变得瘦高，全峰半宽高因此增大，其他位置变化不大。这说明低频部分具有差别，平面部分的热导率也是不同的。弯曲部分的变化则较为



**(b)**

**(a)**

图3-6 (a) 压缩层间距至0.35 nm的折叠石墨烯中，平面部分和弯曲部分的原子的功率谱密度。(b) 压缩层间距至0.30 nm的折叠石墨烯中，平面部分和弯曲部分的原子的功率谱密度。

明显。增强了的层间作用使得低频0 ~ 20 THz的尖峰被压缩至 0 ~ 10 THz内，低频声子的分布更加集中在更低频的位置，且峰的形状也发生了明显改变，由许多分散的峰变为比较明显的、紧凑分布的两个尖峰。增强的层间作用则使高频部分则发生了蓝移，频率大于34 THz时对应的尖峰位置朝着高频移动。

综上，折叠的引入使得平面部分和折叠部分存在声子模式失配，声子散射增加而热导率降低，层间距从0.35 nm变为0.30 nm，层间作用力增强，弯曲部分和平面部分声子失配减少，声子散射减少，折叠对热导率的降低幅度有所减小。因此相同的总长度下，κplanar > κfolded, 0.30nm > κfolded, 0.35nm。这与纳米带中的热导率随层间距减小而增大的结论相反，是因为纳米带中层间作用可以忽略，热导率下降主要来自于折叠对层内输运的影响，折叠石墨烯中的层间作用的贡献更大，而且随着尺寸的增大，折叠对热导率的减小作用也有所削弱。

**4 结论**

**4.1 总结与展望**

利用NEMD模拟折叠石墨烯中的热输运，发现折叠石墨烯表现出了不同于完美石墨烯的性质。折叠的存在降低了热导率。这是因为弯曲部分和平面部分的声子行为不同，折叠/平面交界处将发生明显的散射。当折叠的数目越多时，一方面这种散射发生的更多，另一方面平面部分的长度变小，将限制具有长自由程声子的出现，这两方面都增加了系统的热阻，导致热导率下降。系统总长度越大，这种热导率的减小越来越不明显，这是因为体系变大后，允许的长自由程声子增多，折叠所影响到的声子的比例越来越小。

相邻平面层的通过范德瓦尔斯力的直接输运削弱了各自平面层内的温度梯度，使得平面层内出现接近水平的温度分布。具有不同层间距的折叠石墨烯的热导率不同，层间距越小，热导率越高。当层间距减小时，层间范德瓦尔斯作用增强，层间通过范德瓦尔斯力作用直接传递的热流越多，增强的层间作用提供了更多的热输运的通道。增加的层间力还使得弯曲部分与平面部分的声子行为差异更小，发生的散射减少。这些因素都导致了更高的热导率。

本文发现对于较大面积的折叠石墨烯，相对于完美石墨烯热导率下降并不多。要想用于热电应用，应该尽量使用小尺寸的折叠石墨烯，并创造尽可能多的折叠。层间距离应该避免较小，以减少热在层间直接输运。

**4.2 可改进的方面**

第一，本次模拟只计算了两组层间距不同的情况，要想得到更详细的依赖关系，应该计算其他层间距的情况。第二，模拟出现的效应被归结为长程声子受到影响，但是没有给出证明。依据文献[[31](#_ENREF_31)]，可以通过从MD抽取出折叠石墨烯中声子自由程的信息，得到具有不同自由程的声子对热导率的贡献，证明这种解释。第三，只计算了一种平面部分长度的情况，其他平面部分长度的情况则不得而知。

**致谢**

首先感谢我的导师杨诺教授对我的毕业设计进行的指导。杨诺教授强调学术论文的严谨性，关注细节，同时教会我在学术演讲中讲述专业内容时要注重表达的简洁和清晰。杨诺教授也是我本科阶段二课的导师，他不仅在我的第一段研究经历中提供了无私的支持，保持了严苛的要求，更对我的重大人生选择上提供了中肯的建议。在此表示诚挚感谢！

感谢纳米传热实验室的所有成员，我们每周一次的例会、在实验室的热烈讨论都给予了我切实的提高和对于学术进一步的热爱。特别地，我要感谢钱鑫学长，他的探索精神启发了我走上学术之路，在我申请出国留学时帮助我开拓了视野。感谢金泽林硕士，他充满耐心地回答了我所有的疑问。感谢安盟博士，他与我的合作让我真切体会到团队协作的重要性。

感谢国家超级计算天津中心（National Supercomputer Center in Tianjin）和纳米传热实验室对本文使用的计算资源的支持。

感谢我的特优生导师向军教授，尽管我最终没有选择煤燃烧方面的研究，但他鼓励我找到了自己喜欢的研究方向。

感谢我的室友们，他们对我的作息习惯给予了充分的包容，对我生活给予了关怀。

最后感谢我的父母，他们严格的家教培养了我坚韧的性格，他们对我的包容，对我可能犯错时的提醒，是我成长过程中最宝贵的财富。

**参考文献**

[1] J. Jadhao and D. Thombare, *Review on exhaust gas heat recovery for IC engine*, International Journal of Engineering and Innovation Technology (IJEIT) Volume **2** (2013).

[2] D. G. Cahill, P. V. Braun, G. Chen, D. R. Clarke, S. Fan, et al., *Nanoscale thermal transport. II. 2003–2012*, Applied Physics Reviews **1** 011305 (2014).

[3] J. H. Seol, I. Jo, A. L. Moore, L. Lindsay, Z. H. Aitken, et al., *Two-dimensional phonon transport in supported graphene*, Science **328** 213 (2010).

[4] Z. Wang, R. Xie, C. Bui, D. Liu, X. Ni, B. Li, and J. Thong, *Thermal transport in suspended and supported few-layer graphene*, Nano Lett. **11** 113 (2011).

[5] S. Ghosh, I. Calizo, D. Teweldebrhan, E. P. Pokatilov, D. L. Nika, A. A. Balandin, W. Bao, F. Miao, and C. N. Lau, *Extremely high thermal conductivity of graphene: Prospects for thermal management applications in nanoelectronic circuits*, Appl. Phys. Lett. **92** 151911 (2008).

[6] L. E. F. Torres, S. Roche, and J.-C. Charlier, *Introduction to graphene-based nanomaterials: from electronic structure to quantum transport*, Cambridge University Press, 2014.

[7] X. Xu, Y. Wang, K. Zhang, X. Zhao, S. Bae, et al., *Phonon Transport in Suspended Single Layer Graphene*, arXiv:1012.2937 (2010).

[8] Z. Ni, Y. Wang, T. Yu, Y. You, and Z. Shen, *Reduction of Fermi velocity in folded graphene observed by resonance Raman spectroscopy*, Phys. Rev. B **77** (2008).

[9] J. Zang, S. Ryu, N. Pugno, Q. Wang, Q. Tu, M. J. Buehler, and X. Zhao, *Multifunctionality and control of the crumpling and unfolding of large-area graphene*, Nat. Mater. **12** 321 (2013).

[10] N. Yang, X. Ni, J.-W. Jiang, and B. Li, *How does folding modulate thermal conductivity of graphene?*, Appl. Phys. Lett. **100** 093107 (2012).

[11] H. Zhang, T. Zhou, G. Xie, J. Cao, and Z. Yang, *Thermal transport in folded zigzag and armchair graphene nanoribbons*, Appl. Phys. Lett. **104** 241908 (2014).

[12] W. Jang, Z. Chen, W. Bao, C. N. Lau, and C. Dames, *Thickness-dependent thermal conductivity of encased graphene and ultrathin graphite*, Nano Lett. **10** 3909 (2010).

[13] L. Lindsay, D. Broido, and N. Mingo, *Diameter dependence of carbon nanotube thermal conductivity and extension to the graphene limit*, Phys. Rev. B **82** 161402 (2010).

[14] L. J. Yi, Y. Y. Zhang, C. M. Wang, and T. C. Chang, *Temperature-induced unfolding of scrolled graphene and folded graphene*, J. Appl. Phys. **115** 204307 (2014).

[15] Y. Xie, Y. Chen, X. L. Wei, and J. Zhong, *Electron transport in folded graphene junctions*, Phys. Rev. B **86** 195426 (2012).

[16] R. Kubo, *The fluctuation-dissipation theorem*, Reports on Progress in Physics **29** 255 (1966).

[17] W. C. Swope, H. C. Andersen, P. H. Berens, and K. R. Wilson, *A computer simulation method for the calculation of equilibrium constants for the formation of physical clusters of molecules: Application to small water clusters*, J. Chem. Phys. **76** 637 (1982).

[18] V. Tewary and B. Yang, *Singular behavior of the Debye-Waller factor of graphene*, Phys. Rev. B **79** 125416 (2009).

[19] Y. Wang, A. K. Vallabhaneni, B. Qiu, and X. Ruan, *Two-dimensional thermal transport in graphene: a review of numerical modeling studies*, Nanoscale and Microscale Thermophysical Engineering **18** 155 (2014).

[20] P. K. Schelling, S. R. Phillpot, and P. Keblinski, *Comparison of atomic-level simulation methods for computing thermal conductivity*, Phys. Rev. B **65** 144306 (2002).

[21] S. Nosé, *A unified formulation of the constant temperature molecular dynamics methods*, J. Chem. Phys. **81** 511 (1984).

[22] W. G. Hoover, *Canonical dynamics: Equilibrium phase-space distributions*, Phys. Rev. A **31** 1695 (1985).

[23] A. Dhar, *Heat transport in low-dimensional systems*, Adv. Phys. **57** 457 (2008).

[24] D. W. Ricker, *Echo signal processing*, Springer Science & Business Media, 2003.

[25] M. Kaviany, *Heat transfer physics*, Cambridge University Press, 2008.

[26] N. Yang, G. Zhang, and B. Li, *Carbon nanocone: A promising thermal rectifier*, Appl. Phys. Lett. **93** 243111 (2008).

[27] Y. Quo, N. Karasawa, and W. A. Goddard, *Prediction of fullerene packing in C60 and C70 crystals*, Nature **351** 464 (1991).

[28] R. E. Tuzun, D. W. Noid, B. G. Sumpter, and R. C. Merkle, *Dynamics of fluid flow inside carbon nanotubes*, Nanotechnology **7** 241 (1996).

[29] S. Lepri, *Thermal conduction in classical low-dimensional lattices*, Phys. Rep. **377** 1 (2003).

[30] O. Narayan and A. P. Young, *Continuum and lattice heat currents for oscillator chains*, Physical review. E, Statistical, nonlinear, and soft matter physics **80** 011107 (2009).

[31] K. Sääskilahti, J. Oksanen, S. Volz, and J. Tulkki, *Frequency-dependent phonon mean free path in carbon nanotubes from nonequilibrium molecular dynamics*, Phys. Rev. B **91** (2015).

[32] H. Casimir, *Note on the Conduction of Heat in Crystals*, Physica **5** 495 (1938).

[33] Y. F. Zhu, J. S. Lian, and Q. Jiang, *Re-examination of Casimir limit for phonon traveling in semiconductor nanostructures*, Appl. Phys. Lett. **92** 113101 (2008).

[34] E. Pop, V. Varshney, and A. K. Roy, *Thermal properties of graphene: Fundamentals and applications*, MRS Bulletin **37** 1273 (2012).

[35] Z. Rieder, J. Lebowitz, and E. Lieb, *Properties of a harmonic crystal in a stationary nonequilibrium state*, Journal of Mathematical Physics **8** 1073 (1967).

[36] N. Yang, T. Luo, K. Esfarjani, A. Henry, Z. Tian, J. Shiomi, Y. Chalopin, B. Li, and G. Chen, *Thermal Interface Conductance Between Aluminum and Silicon by Molecular Dynamics Simulations*, J. Comput. Theor. NanoS. **12** 168 (2015).

**附录**

**S.1 热导率计算的处理细节**

同等尺寸的完美石墨烯的热导率是计算折叠石墨烯热导率的重要参考。由于石墨烯的热导率计算具有明显的尺寸依赖性，因此必须对不同尺寸的石墨烯进行单独计算。图 给出了不同尺寸下的温度分布。可以看出，由于人为热浴的作用，处于边界的原子的温度分布存在明显的温度跳跃，导致了温度分布的非线性。为了减少这种作用的影响，计算温度梯度时将这部分原子去掉，利用剩余原子的温度数据进行拟合温度梯度。因为去掉的原子数目是不同的，故∆T的取值也是变化的。



图S-1 长度为36.7nm，48.9nm，61.3nm和73.6nm的沿着长度方向温度分布曲线及温度梯度的线性拟合。

**S.2 声子态密度的计算**



图S-2 压缩层间距至0.35 nm，0.30 nm的折叠石墨烯中，平面部分和弯曲部分的原子的功率谱密度。

图S-2是平面部分和弯曲部分的声子态密度，所记录的原子位置与图3-6中功率谱统计的原子完全一致，结构参数也与图3-6所计算的结构弯曲一致。声子态密度通过对自相关函数的傅利叶变换得到。在高温近似下，声子态密度和功率谱密度只相差一个系数kBT。比较图S-2和图3-6也可以发现，声子态密度曲线和功率谱密度曲线形状大致一致。与功率谱密度分析得到的结论也是一致的。主要的区别是在层间距为0.35 nm的情况时，弯曲部分声子态密度在34 THz有明显的尖峰，且尖峰高度最高，而功率谱中的最高尖峰分布在其他位置，这可能是计算的时间不够长或者用于平均的数据少导致的误差。

**S.3 程序源代码**

用来建立折叠石墨烯结构的Fortran90程序：

! Folded Grephene! 包含位置 和0速度，0加速度

module Parameters

! Contains parameters of Folded Graphene

!INTEGER, parameter :: Model\_No = 100 ! 100=homogeneous;1=graded 1/N 2/N ... N/N ; 0=half big -half small

REAL, PARAMETER :: PI=3.141592653589793238462643383279502884197 ! 2= graded mass every layer

INTEGER,PARAMETER :: Case\_No=12 !MPIf90 no of cores for MPI !20140428qichen

CHARACTER\*70 :: SampOut\_base='READ\_pva1\_T300\_LAY000\_00F0\_FG' !BPB11 pos vel acc !20140428qichen

CHARACTER\*80 :: SampOut(Case\_No) !20140428qichen

CHARACTER\*80 :: SampOut\_4 = 'READ\_Neib\_LAY000\_00F0\_FG.dat'

CHARACTER\*70 :: SampOut\_2\_base = 'READ\_3D\_LAY000\_00F0\_FG' !20140428qichen

character\*80 :: SampOut\_2(Case\_No) !20140428qichen

character\*80 :: SampOut\_3 = 'READ\_2D\_LAY000\_00F0\_FG.pdb'

!character\*80 :: SampOut\_1 = 'READ\_tem\_LAY000\_00F0\_FG.dat'

CHARACTER\*80 :: SampOut\_5 = 'READ\_Npl\_LAY000\_00F0\_FG.dat'

CHARACTER\*80 :: SampOut\_6 = 'READ\_Mlist\_LAY000\_00F0\_FG.dat'

CHARACTER\*80 :: SampOut\_7 = 'READ\_BClist\_LAY000\_00F0\_FG.dat' ! 边界粒子的编号

CHARACTER\*80 :: SampOut\_8\_base = 'READ\_Van\_Neib\_LAY000\_00F0\_FG' !范德华力 的 邻居表 包含邻居编号，平移距离，邻居数目!20140608qichen

CHARACTER\*80 :: SampOut\_8(Case\_No)!20140608qichen

CHARACTER\*80 :: SampOut\_9\_base = 'READ\_Sub\_Grp\_pairs\_LAY000\_00F0\_FG' !20150302qichen

CHARACTER\*80 :: SampOut\_9(Case\_No) !20150302qichen

INTEGER :: rdseed(90)=(/127,131,137,139,149,151,157,163,167, 179,181,191,193,197,199,211,223,227,229, &

233,239,241,251,257,263,269,271,277,281, 283,293,307,311,313,317,331,337,347,349, &

353,359,367,373,379,383,389,397,401,409, 419,421,431,433,439,443,449,457,461,463, &

467,479,487,491,499,503,509,521,523,541, 547,557,563,569,571,577,587,593,599,601, &

607,613,617,619,631,641,643,647,653,659,173 /) !MPIf90 hu20130606!MPIf90 !20140428qichen

integer, parameter :: NLAY =300 ! No of layers, length of tube= Nlay\*a\*3^0.5/2; 10lay=1.26nm

integer, parameter :: Ntube1=5 ! armchair 10 10

integer, parameter :: Ntube2=5

integer :: NPL =Ntube1+Ntube2 ! only right for armchair

integer :: N =NLAY\*(Ntube1+Ntube2) ! NLAY\*NPL

integer,parameter ::n\_folded=6 !弯曲的个数

double precision ::r\_van\_1,r\_van\_2 !20141006qichen

INTEGER, DIMENSION(:,:), ALLOCATABLE :: around\_shift

INTEGER, DIMENSION(:), ALLOCATABLE :: around\_flag

integer, dimension(:), allocatable :: Npl\_list ! No of partical per layer

integer, dimension(:), allocatable :: Npl\_mov ! No of partical moved per layer= whcih without 4 neighbor

integer, dimension(:), allocatable :: BC\_list

integer :: N\_BC

integer, parameter :: DIM = 3 ! Dimension

double precision, parameter :: a = 1.418E-10 ! C-C disk 1.458 SWNT 1.418 morse配套应该是1.418

double precision :: offset\_gauss !=1.458E-11 ! 位置偏移最大值 0.0\*a

double precision, parameter :: T = 2.\*300.d0 !100.0d0 ! temperature

double precision, parameter :: kb = 1.38E-23

double precision, parameter :: mass = 0.012E-23/6.023

double precision, parameter :: mass\_ratio = 1

DOUBLE PRECISION,DIMENSION(:),ALLOCATABLE :: mass\_list ! graded nework

double precision :: Aera =Ntube1\*3.\*a\*1.4E-10 ! 1.4\*2PiR, only right for armchair

INTEGER :: Naround=3

integer, dimension(:), allocatable ::Van\_num !20140504qichen

integer, dimension(:,:), allocatable ::Van\_around !20140504qichen

integer, dimension(:,:), allocatable ::Neib\_sec\_near !20141222qichen

integer,dimension(:,:), allocatable ::Van\_around\_shift !20140504qichen

integer, dimension(:), allocatable ::Sub\_grp\_pairs\_num !20150302qichen

integer, dimension(:,:), allocatable ::Sub\_grp\_pairs !20150302qichen

DOUBLE PRECISION,DIMENSION(:,:),allocatable :: pos\_sub\_down !20150302qichen

DOUBLE PRECISION,DIMENSION(:,:),allocatable :: pos\_sub\_up !20150302qichen

INTEGER ,parameter :: Van\_top=600 !20140520qichen 邻居上限

integer,dimension(:),allocatable :: Mark !20140821qichen

DOUBLE PRECISION, DIMENSION(:,:),allocatable :: slp2 !20140821qichen

DOUBLE PRECISION, DIMENSION(:),allocatable :: slp1,slp\_x !20140821qichen

integer,dimension(:),allocatable ::Plane\_No !用来实时存储各个平面和弯曲的编号 !20140821qichen

integer, dimension(:), allocatable ::Plane\_flag !20140504qichen

INTEGER, DIMENSION(:,:), ALLOCATABLE:: around

double precision ::lay\_dist !distance between layers

double precision ::plane\_len !the length of the plane part

double precision :: z\_top !20140517qichen the topest atom's z coordinate

double precision :: z\_low !20140517qichen the lowest atom's coordinate

INTEGER :: N\_plane\_up =0,N\_plane\_down=0 !20150302qichen 这里 N\_plane\_up/down 指的是动态变化的最上/下 平面部分，即受基底作用力的部分

logical :: VelAcc= .TRUE. ! if vel & acc write to file?

double precision, dimension(:,:), allocatable :: pos ! positions

double precision, dimension(:,:), allocatable :: vel ! positions

double precision, dimension(:,:), allocatable :: acc ! positions

double precision, dimension(:,:),allocatable :: Pos2d\_big ! POS FOR 2D GRAPHITE

double precision, dimension(:,:),allocatable :: temp ! TEMPERATURE

end module Parameters

module SWPotential

DOUBLE PRECISION, PARAMETER :: SW\_A =5.3732037

DOUBLE PRECISION, PARAMETER :: SW\_B =0.50824571

DOUBLE PRECISION, PARAMETER :: SW\_p =4.d0

DOUBLE PRECISION, PARAMETER :: SW\_pp1 =5.d0 !SW\_p+1

DOUBLE PRECISION, PARAMETER :: SW\_q =0.d0

DOUBLE PRECISION, PARAMETER :: SW\_qp1 =1.d0 !SW\_q+1

DOUBLE PRECISION, PARAMETER :: SW\_sa =1.8943619d0

DOUBLE PRECISION, PARAMETER :: SW\_lamda =18.707929

DOUBLE PRECISION, PARAMETER :: SW\_gama =1.2d0

DOUBLE PRECISION, PARAMETER :: SW\_costh =-1./2.d0

DOUBLE PRECISION, PARAMETER :: SW\_sigm =1.26329438E-10 ! 1.418E-10/2^(1/6)

DOUBLE PRECISION, PARAMETER :: SW\_epsl =4.96/(6.24150636309E18) ! J/pair = 4.96 eV/pair

CONTAINS

FUNCTION fun\_f2(r0)

DOUBLE PRECISION :: fun\_f2,r,r0

r=r0/SW\_sigm ! 此处如果用r=r/sw 将会在整个程序里都改变r 即rik

IF (r < SW\_sa) THEN

fun\_f2 = -1.d0\*SW\_A\*( SW\_B\*SW\_p/(r\*\*SW\_pp1) + &

( SW\_B/(r\*\*SW\_p) - 1.d0 )/((r-SW\_sa)\*(r-SW\_sa)) ) \* dexp(1.d0/(r-SW\_sa)) !SW\_B SW\_p SW\_q

ELSE

fun\_f2 = 0.d0

END IF

RETURN

END FUNCTION fun\_f2

FUNCTION fun\_f3(v\_rij0,s\_rij0,v\_rkj0,s\_rkj0,costh)

USE Parameters

DOUBLE PRECISION :: s\_rij0,s\_rkj0,s\_rij,s\_rkj,costh

DOUBLE PRECISION,DIMENSION(DIM) :: fun\_f3,v\_rij0,v\_rkj0,v\_rij,v\_rkj

v\_rij=v\_rij0/SW\_sigm

s\_rij=s\_rij0/SW\_sigm

v\_rkj=v\_rkj0/SW\_sigm

s\_rkj=s\_rkj0/SW\_sigm

IF (s\_rij < SW\_sa .and. s\_rkj< SW\_sa) THEN

fun\_f3(:) = SW\_lamda\*fun\_H1(s\_rij,s\_rkj)\*( SW\_gama\*fun\_H2(costh)\*v\_rij/((s\_rij-SW\_sa)\*(s\_rij-SW\_sa)\*s\_rij) + &

2.d0\*(costh-SW\_costh)\*( v\_rij\*costh/(s\_rij\*s\_rij) - v\_rkj/(s\_rij\*s\_rkj) ) )

ELSE

fun\_f3 = 0.d0

END IF

RETURN

END FUNCTION fun\_f3

FUNCTION fun\_H1(rij,rkj)

DOUBLE PRECISION :: fun\_H1,rij,rkj

fun\_H1 = dexp( SW\_gama/(rij-SW\_sa) + SW\_gama/(rkj-SW\_sa) )

RETURN

END FUNCTION fun\_H1

FUNCTION fun\_H2(costheta)

DOUBLE PRECISION ::fun\_H2, costheta

fun\_H2 = (costheta-SW\_costh)\*(costheta-SW\_costh)

RETURN

END FUNCTION fun\_H2

FUNCTION fun\_V2(r0)

DOUBLE PRECISION :: fun\_V2,r,r0

r=r0/SW\_sigm

IF (r < SW\_sa) THEN

fun\_V2 = SW\_epsl\*SW\_A\*(SW\_B/(r\*\*SW\_p) - 1.d0 )\*dexp(1.d0/(r-SW\_sa))

ELSE

fun\_V2 = 0.d0

END IF

RETURN

END FUNCTION fun\_V2

FUNCTION fun\_V3(v\_rij0,s\_rij0,v\_rkj0,s\_rkj0,costh)

USE Parameters

DOUBLE PRECISION :: fun\_V3,s\_rij,s\_rkj,s\_rij0,s\_rkj0,costh

DOUBLE PRECISION,DIMENSION(DIM) :: v\_rij,v\_rkj,v\_rij0,v\_rkj0

v\_rij=v\_rij0/SW\_sigm

s\_rij=s\_rij0/SW\_sigm

v\_rkj=v\_rkj0/SW\_sigm

s\_rkj=s\_rkj0/SW\_sigm

IF (s\_rij < SW\_sa .and. s\_rkj< SW\_sa) THEN

fun\_V3 = SW\_epsl\*SW\_lamda\*fun\_H1(s\_rij,s\_rkj)\*fun\_H2(costh)

ELSE

fun\_V3 = 0.d0

END IF

RETURN

END FUNCTION fun\_V3

end module SWPotential

!#######################################################################################

!######## MAIN PROGRAME START #############################################

!#######################################################################################

program Positions

!USE DFLIB

use Parameters

use SWPotential

implicit none

integer :: i,j,k,k1,i\_gauss,i\_lay,i\_ver,i\_inl,i\_1,i\_0,i\_nei,i\_big,i\_case ! CNCone !20140428qichen

double precision :: theta0,theta1,theta2 ! CNCone

Logical :: flag1, flag2

double precision :: r0,r1,r2 ! radius of tube

double precision, dimension(:), allocatable :: v0 ! 温度为T的平均速度

double precision, dimension(12) :: gauss ! guass 分布的时候用到

real :: ag,ag\_1

Call Name\_files !BPB11 !BPB15f90 补全文件名! 20140428qichen

CALL CREATE\_2D\_GRAPHITE ! Part 1 计算一个2D Graphite平面

CALL Get\_Npl\_LIST ! Part 3 Get Npl\_list

!CALL PLANE\_TO\_TUBE ! Part 将2D平面坐标转换成TUBE 3D坐标

CALL PLANE\_TO\_CURVE !part

CALL NEIGHBOR\_LIST ! Part 2 计算生成 Neighbor list

CALL Get\_Van\_around\_list !part 生成Van der Waals force neighbor list

CALL Initiating\_substrate !part 生成基底的位置 ！20150302qichen

CALL Get\_Sub\_grp\_pairs ! part to generate the relationship of subtrate atom and sample atom !20150302qichen

CALL Get\_Npl\_Move\_LIST ! Part 4 Get Npl\_mov

CALL Get\_BC\_LIST ! Part 5 Get BClist 边界粒子的编号

!CALL GAUSS\_OFFSET\_POSITION

CALL compute\_masslist ! Part 6 MASS LIST

! Part 8 计算生成 !加速度

allocate( temp(DIM+1,N) )

!do i\_big=1,100

!CALL EVOLVE\_SAMPLE ! Part 10 优化结构

!CALL compute\_temperature ! Part 11 20140428qichen

!write(\*,'(1x,"### AVE Tmp ###",E16.8 ,"### Max Tmp ###",E16.8)') SUM(temp(DIM+1,:))/N, MAXVAL(temp(DIM+1,:)) !20140428qichen

!if( MAXVAL(temp(DIM+1,:)) < 1.d0) then

do i\_case=1,Case\_No !BPB15f90 !20140428qichen

vel(:,:)=0.d0 !20140428qichen

CALL RANDOM\_VELOSITIES(i\_case) ! Part 7 计算生成 和温度 !20140428qichen

temp(:,:)=0.d0 !Nuo20130614

CALL compute\_temperature ! Part 11 20140428qichen

write(\*,'(1x,"### AVE Tmp ###",E16.8 ,"### Max Tmp ###",E16.8)') SUM(temp(DIM+1,:))/N, MAXVAL(temp(DIM+1,:))

! CALL Initial\_momentum(1,N) ! BPB8\_F90 动量归零 !20140428qichen

! CALL compute\_temperature ! Part 11

CALL WRITE\_OUT(i\_case) ! Part 9 ! Write out Data !20140428qichen

print '(a,I10)', '\*\*\*\*\*\* Success Case ', i\_case !20140428qichen

enddo !BPB15f90 !20140428qichen

go to 911

!endif

!enddo

911 deallocate( around)

deallocate( around\_shift)

deallocate( around\_flag)

deallocate( mass\_list )

!deallocate( Npl )

deallocate( pos )

deallocate( vel )

deallocate( acc )

deallocate( Pos2D\_big )

deallocate(temp)

deallocate( Npl\_list )

deallocate( Npl\_mov )

deallocate( BC\_list )

deallocate( Van\_num ) !20140504qichen

deallocate( Van\_around ) !20140504qichen

deallocate( Van\_around\_shift) !20140504qichen

deallocate( Sub\_grp\_pairs) !20150302qichen

deallocate( Sub\_grp\_pairs\_num) !20150302qichen

deallocate( pos\_sub\_down) !20150302qichen

deallocate( pos\_sub\_up) !20150302qichen

deallocate( Neib\_sec\_near ) !20141222qichen

deallocate( slp2 ) !20140821qichen

deallocate( slp1 ) !20140821qichen

deallocate( slp\_x ) !20140821qichen

deallocate(Plane\_No) !20140821qichen

STOP

end program Positions

!#######################################################################################

!######## MAIN PROGRAME END #############################################

!#######################################################################################

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!!Name\_files!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE Name\_files !BPB11 !BPB15f90 !20140428qichen start

use Parameters !20140428qichen

implicit none

INTEGER:: i,j,i1,i2,icase

CHARACTER\*1 :: tmpcha1,tmpcha2

do icase=1,Case\_No

!ASII of '0' =48, '9'=57

IF(icase>99) stop 'CHA'

i1=icase/10

i2=mod(icase,10)

tmpcha1=CHAR(i1+48)

tmpcha2=CHAR(i2+48)

if(LEN\_TRIM(SampOut\_2\_base)+6 > LEN(SampOut\_2(icase)) ) stop 'Len' !20140428qichen

SampOut\_2(icase)=Sampout\_2\_base(1:LEN\_TRIM(SampOut\_2\_base))//'\_'//tmpcha1//tmpcha2//'.pdb' !BPB11 !20140428qichen

if(LEN\_TRIM(SampOut\_base)+6 > LEN(SampOut(icase)) ) stop 'Len'

SampOut(icase)=SampOut\_base(1:LEN\_TRIM(SampOut\_base))//'\_'//tmpcha1//tmpcha2//'.dat' !BPB11 !20140608qichen start

if(LEN\_TRIM(SampOut\_8\_base)+6 > LEN(SampOut\_8(icase)) ) stop 'Len'

SampOut\_8(icase)=SampOut\_8\_base(1:LEN\_TRIM(SampOut\_8\_base))//'\_'//tmpcha1//tmpcha2//'.dat' !BPB11 !20140608qichen end

if(LEN\_TRIM(SampOut\_9\_base)+6 > LEN(SampOut\_9(icase)) ) stop 'Len'

SampOut\_9(icase)=SampOut\_9\_base(1:LEN\_TRIM(SampOut\_9\_base))//'\_'//tmpcha1//tmpcha2//'.dat'

enddo

RETURN

END SUBROUTINE Name\_files !BPB11 !20140428qichen end

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!!Name\_files!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

! Part 1 计算一个2D Graphite平面

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE CREATE\_2D\_GRAPHITE ! Part 1 计算一个2D Graphite平面

!USE DFLIB

use Parameters

implicit none

integer i,j,k

double precision :: alfa ! angle alfa of a in circle whose perimeter is (3a\*Npl/2)

double precision :: r ! radius of tube

double precision :: dist ! distence between layer

double precision, dimension(:), allocatable :: alfa1 ! POSTITIONS of odd layer

double precision, dimension(:), allocatable :: alfa2 ! POSTITIONS of even layer

!integer :: k1,i\_lay,i\_ver,i\_inl,i\_1,i\_0,i\_nei ! CNCone

!double precision :: theta0,theta1,theta2 ! CNCone

!NPL=2\*NAM

!N = NLAY\*NPL

dist=a\*3.0\*\*(0.5)/2.0

!allocate( pos(DIM,N) )

allocate( Pos2d\_big(DIM,N) ) ! polar coordinates on 2D surface;

! Pos2d\_big(N\_big,1) instead of x, Pos2d\_big(N\_big,2) instead of y

allocate( alfa1(Npl) ) ! 奇数层Y

allocate( alfa2(Npl) ) ! 偶数层Y

! get the POSTITIONS of particle in 1st layer circle

alfa1(1)=a/2.0

alfa1(2)=3.0\*a/2.0

do i=3,Npl,2

alfa1(i)=alfa1(i-1)+2.0\*a

alfa1(i+1)=alfa1(i)+a

enddo

! get the POSTITIONS of particle in 2nd layer circle

alfa2(1)=0.0

alfa2(2)=2\*a

do i=3,Npl,2

alfa2(i)=alfa2(i-1)+a

alfa2(i+1)=alfa2(i)+2.0\*a

enddo

! get the POSTITIONS of particle in ALL layer circle

do i=1,Npl

Pos2d\_big(1,i)=0.0 ! Get the coordinates of 1st layer

Pos2d\_big(2,i)=alfa1(i)

Pos2d\_big(3,i)=0.0

Pos2d\_big(1,(i+Npl))=0.0 ! Get the coordinates of 2nd layer

Pos2d\_big(2,(i+Npl))=alfa2(i)

Pos2d\_big(3,(i+Npl))=dist

enddo

! get coordinates from 3rd layer to last layer

do i=2,(Nlay-1),2 ! Layer's No. - 1

if( i==(Nlay-1)) then

do j=1,Npl

Pos2d\_big(1,i\*Npl+j)=Pos2d\_big(1,j) ! odd layers

Pos2d\_big(2,i\*Npl+j)=Pos2d\_big(2,j)

Pos2d\_big(3,i\*Npl+j)=i\*dist

!Pos2d\_big(1,(i+1)\*Npl+j)=Pos2d\_big(1,j+Npl) ! even layers

!Pos2d\_big(2,(i+1)\*Npl+j)=Pos2d\_big(2,j+Npl)

!Pos2d\_big(3,(i+1)\*Npl+j)=(i+1)\*dist

enddo

else

do j=1,Npl

Pos2d\_big(1,i\*Npl+j)=Pos2d\_big(1,j) ! odd layers

Pos2d\_big(2,i\*Npl+j)=Pos2d\_big(2,j)

Pos2d\_big(3,i\*Npl+j)=i\*dist

Pos2d\_big(1,(i+1)\*Npl+j)=Pos2d\_big(1,j+Npl) ! even layers

Pos2d\_big(2,(i+1)\*Npl+j)=Pos2d\_big(2,j+Npl)

Pos2d\_big(3,(i+1)\*Npl+j)=(i+1)\*dist

enddo

endif

enddo

RETURN

END SUBROUTINE CREATE\_2D\_GRAPHITE ! Part 1 计算一个比较大的2D Graphite平面

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

! Part 将2D平面坐标转换成台式3D坐标

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE PLANE\_TO\_TUBE ! Part 3 将2D平面坐标转换成TUBE 3D坐标

!USE DFLIB

use Parameters

implicit none

integer :: i

double precision :: r0 ! radius of tube

double precision :: theta0 ! 位置对应的角度

allocate( pos(DIM,N) )

allocate( vel(DIM,N) )

allocate( acc(DIM,N) )

pos=0.0d0

vel=0.0d0

acc=0.0d0

! Pos2d\_big(1,:)(2,:)变化 (3,:)不变

r0 = NPL\*a\*(3.D0)/(4.d0\*PI) ! 切面圆的直径 3^(1/2)\*dc\_c\*(M^2+MN+N^2)^(1/2) /2PI

do i=1,N

theta0 = Pos2d\_big(2,i)/r0 ! 弧长Pos2d\_big(2,i)求角度

pos(3,i)=DCOS(theta0)\*r0 !20140425 qichen 3 & 1 swap

pos(1,i)=DSIN(theta0)\*r0 !20140425 qichen 3 & 1 swap

enddo

pos(2,:)=Pos2d\_big(2,:) !20140425 qichen 3 & 1 swap

RETURN

END SUBROUTINE PLANE\_TO\_TUBE ! Part 3 将2D平面坐标转换成台式3D坐标

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!Part 得到每一层的原子数!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE Get\_Npl\_LIST ! Part 计算生成 Npl\_list

!USE DFLIB

use Parameters

implicit none

integer :: i,j

double precision :: Lw,Hg

allocate(Npl\_list(Nlay))

Npl\_list(:)=0.d0

Lw=-0.1\*a-0.5\*sqrt(3.d0)\*a

Hg=0.1\*a-0.5\*sqrt(3.d0)\*a

searchlay: do i=1,Nlay

Lw=Lw+0.5\*sqrt(3.d0)\*a

Hg=Hg+0.5\*sqrt(3.d0)\*a

searchall:do j=1,N

if( (Pos2d\_big(3,j)>Lw) .and. (Pos2d\_big(3,j)<Hg)) Npl\_list(i)=Npl\_list(i)+1

enddo searchall

enddo searchlay

END SUBROUTINE Get\_Npl\_LIST ! Part 计算生成 Npl\_list

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!!Part 将平面弯折!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE PLANE\_TO\_CURVE

!use DFLIB

use Parameters

implicit none

integer :: i,j,k

integer,parameter :: nR=6 !内接多边形的边的条数(每边长度按sqrt(3.d0)\*a计算)，推荐nR为奇数的两倍，如6，10,14,

double precision :: R ! radius of CURVE

double precision :: theta1 ! 对应的角度

double precision :: dist

INTEGER,dimension(:),allocatable :: icenter !记录弯曲中心的lay's no.

integer :: bot !没有参与弯曲的、依然在水平面内的最后一个粒子行编号

integer :: top !参与弯曲的、曲面内最后一个粒子行编号，行指的是i

double precision :: R\_tp1,R\_tp2 ,dd , Coscos!20140903 qichen

allocate(pos(DIM,N))

allocate(vel(DIM,N))

allocate(acc(DIM,N))

allocate(icenter(20)) ! 至多有20个褶

allocate(plane\_flag(N)) !20140504qichen !20140517qichen

pos=0.0d0

vel=0.0d0

acc=0.0d0

icenter=0 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!i有奇数偶数之分

plane\_flag=0 !20140504qichen !20140517qichen

dist=0.5\*a\*3.0\*\*(0.5)

R= (3.0\*\*(0.5))\*a /(4.d0 \*DSIN(PI/dble(2\*nR))) !20140903qichen !修正以后，侧视图是正多边形

lay\_dist=2.d0\*R !20140429qichen

k=1 !底层以及1st fold

icenter(k)=(Nlay-1-n\_folded\*nR)/(n\_folded+1)+nR/2+1 !Npl/(n\_folded+1)-2 20140328 !找到中心点的层数,按No.1lay去选择

bot=icenter(k)-nR/2 !bot=(icenter(1)-nR/4)\*Nlay

top=icenter(k)+nR/2 !top=(icenter(1)+nR/4+1)\*Nlay

plane\_len=Pos2d\_big(3,Npl\*bot)-Pos2d\_big(3,1) !20140429qichen 2D中3对应3D中1

pos(2,:)=Pos2d\_big(2,:) !20140903qichen

do i=0,bot-1

do j=1,Npl

pos(3,i\*Npl+j)=Pos2d\_big(1,i\*Npl+j) !20140425 qichen 3 & 1 swap

pos(1,i\*Npl+j)=Pos2d\_big(3,i\*Npl+j) !20140425 qichen 3 & 1 swap

plane\_flag(i\*Npl+j)=1 !20140517qichen

enddo

enddo !draw the atom which still remains in the plane(i=odd)

do i=bot,top-1

do j=1,Npl

theta1= (Pos2d\_big(3,i\*Npl+j)-Pos2d\_big(3,bot\*Npl))\*PI/(dble(nR)\*dist) ! 弧长Pos2d\_big(2,i)求角度

pos(3,i\*Npl+j)= R-DCOS(theta1)\*R+pos(3,bot\*Npl) !20140425 qichen 3 & 1 swap

pos(1,i\*Npl+j)=DSIN(theta1)\*R+pos(1,bot\*Npl) !20140425 qichen 3 & 1 swap

if (i==top-1)then !20140923qichen

plane\_flag(i\*Npl+j)=2 !20140517qichen !20140923qichen

else

plane\_flag(i\*Npl+j)=0 !20140923qichen

endif

enddo

enddo !draw the atom of the curve

!20140903qichen

R\_tp1= sqrt(dot\_product(pos(:,(i-2)\*Npl)-pos(:,(i-1)\*Npl),pos(:,(i-2)\*Npl)-pos(:,(i-1)\*Npl)))

write(\*,'(G20.10)') R\_tp1 \* 1E10

R\_tp2= sqrt(dot\_product(pos(:,(i-2)\*Npl)-pos(:,(i-3)\*Npl),pos(:,(i-2)\*Npl)-pos(:,(i-3)\*Npl)))

write(\*,'(G20.10)') R\_tp2 \* 1E10

dd=sqrt(dot\_product(pos(:,(i-1)\*Npl)-pos(:,(i-3)\*Npl),pos(:,(i-1)\*Npl)-pos(:,(i-3)\*Npl)))

write(\*,'(G20.10)') dd \* 1E10

Coscos = (R\_tp1\*R\_tp1+ R\_tp2\*R\_tp2 - dd\*dd) / (2.0d0\*R\_tp1\*R\_tp2)

write(\*,'(G20.10)') 180\* ACOS(Coscos) /PI

!20140903qichen

do i=top,Nlay-1

do j=1,Npl

pos(1,i\*Npl+j)=pos(1,top\*Npl)-dble(i+1-top)\*dist !20140425 qichen 3 & 1 swap

pos(3,i\*Npl+j)=pos(3,top\*Npl) !20140425 qichen 3 & 1 swap

plane\_flag(i\*Npl+j)=2 !20140517qichen

enddo

enddo !draw the atom which form the upper plane

do k=2,n\_folded

if(mod(k,2)==0)then

icenter(k)=(Nlay-1-n\_folded\*nR)/(n\_folded+1)+nR+icenter(k-1) !Npl/(n\_folded+1)+icenter(k-1) 20140328 !找到中心点的层数,按No.1lay去选择

bot=icenter(k)-nR/2 !bot=(icenter(1)-nR/4)\*Nlay

top=icenter(k)+nR/2 !top=(icenter(1)+nR/4+1)\*Nlay

do i=bot,top-1

do j=1,Npl

theta1= (Pos2d\_big(3,i\*Npl+j)-Pos2d\_big(3,bot\*Npl)) \*PI/(dble(nR)\*dist) ! 弧长Pos2d\_big(2,i)求角度

pos(3,i\*Npl+j)= R-DCOS(theta1)\*R+pos(3,bot\*Npl) !20140425 qichen 3 & 1 swap

pos(1,i\*Npl+j)=pos(1,bot\*Npl)-DSIN(theta1)\*R

if (i==top-1)then !20140923qichen

plane\_flag(i\*Npl+j)=k+1 !20140517qichen !20140923qichen

else

plane\_flag(i\*Npl+j)=0 !20140923qichen

endif

enddo

enddo !draw the atom of the curve

do i=top,Nlay-1

do j=1,Npl

pos(1,i\*Npl+j)=pos(1,top\*Npl)+dble(i+1-top)\*dist !20140425 qichen 3 & 1 swap

pos(3,i\*Npl+j)=pos(3,top\*Npl) !20140425 qichen 3 & 1 swap

plane\_flag(i\*Npl+j)=k+1!20140517qichen

enddo

enddo

else

icenter(k)=(Nlay-1-n\_folded\*nR)/(n\_folded+1)+nR+icenter(k-1) !Npl/(n\_folded+1)+icenter(k-1) !找到中心点的层数,按No.1lay去选择

bot=icenter(k)-nR/2 !bot=(icenter(1)-nR/4)\*Nlay

top=icenter(k)+nR/2 !top=(icenter(1)+nR/4+1)\*Nlay

do i=bot,top-1

do j=1,Npl

theta1= (Pos2d\_big(3,i\*Npl+j)-Pos2d\_big(3,bot\*Npl))\*PI/(dble(nR)\*dist) ! 弧长Pos2d\_big(2,i)求角度

pos(3,i\*Npl+j)= R-DCOS(theta1)\*R+pos(3,bot\*Npl) !20140425 qichen 3 & 1 swap

pos(1,i\*Npl+j)=pos(1,bot\*Npl)+DSIN(theta1)\*R !20140425 qichen 3 & 1 swap

if (i==top-1)then !20140923qichen

plane\_flag(i\*Npl+j)=k+1 !20140517qichen !20140923qichen

else

plane\_flag(i\*Npl+j)=0 !20140923qichen

endif

enddo

enddo !draw the atom of the curve

do i=top,Nlay-1

do j=1,Npl

pos(1,i\*Npl+j)=pos(1,top\*Npl)-dble(i+1-top)\*dist !20140425 qichen 3 & 1 swap

pos(3,i\*Npl+j)=pos(3,top\*Npl) !20140425 qichen 3 & 1 swap

plane\_flag(i\*Npl+j)=k+1!20140517qichen

enddo !20140425 qichen 3 & 1 swap

enddo !20140425 qichen 3 & 1 swap

endif !20140425 qichen 3 & 1 swap

enddo !20140425 qichen 3 & 1 swap

RETURN

END SUBROUTINE PLANE\_TO\_CURVE

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

! Part 2 计算生成 Neighbor list

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE NEIGHBOR\_LIST ! Part 4 计算生成 Neighbor list

!USE DFLIB

use Parameters

implicit none

integer :: i\_0,i\_1,KL,KH,i,i\_nei,i\_x,i\_y

double precision :: r0,r1,r2

double precision, dimension(DIM) :: pos\_tmp

allocate(around\_shift(Naround\*DIM,N))

allocate(around\_flag(N))

allocate(around(Naround,N))

around\_shift=0

around\_flag=0

around=0

r1=a\*0.8d0

r2=a\*1.2d0

cyc41:do i\_1 = 1, N

KL=1 !KL = MAX(1,i\_1-Nlay-1 ) ! 搜索下限!20140524qichen

KH= N!KH = MIN(N,i\_1+Nlay+1 ) ! 上限 !20140524qichen

i=i\_1

i\_nei=0

cyc43a:do i\_0 =KL,KH

r0 = sqrt( dot\_product(pos(:,i)-pos(:,i\_0),pos(:,i)-pos(:,i\_0)) )

if( (r0 > r1) .and. (r0 < r2) )then

i\_nei=i\_nei+1

if(i\_nei > Naround)then

write(\*,'(1x,"### i\_neighbor > Naround ###",I10)') i

stop

endif

around(i\_nei ,i) = i\_0

endif

enddo cyc43a

!平移距离是3\*a\*Nlay/4

if(i\_nei < Naround)then

around\_flag(i\_1)=1

! if(i\_1 <= NPL)then !20140429qichen mistake start

! KL=N-NPL+1

! KH=N

! elseif(i\_1> N-NPL)then

! KL=1

! KH=NPL

! else

! continue

!endif !20140429qichen mistake end

KL=1 !KL = MAX(1,i\_1-Nlay-1 ) ! 搜索下限 !20140429qichen !20140524qichen

KH=N !KH = MIN(N,i\_1+Nlay+1 ) ! 上限 !20140429qichen !20140524qichen

cyc43b:do i\_0= KL,KH

do i\_x=-1,1,2

if(mod(Npl,2)==0)then

pos\_tmp(2)=pos(2,i\_1)+dble(3\*i\_x\*Npl/2)\*a

else

pos\_tmp(2)=pos(2,i\_1)+dble( i\_x \*( ( 3\*(Npl-1) / 2+ 3 ) ) )\*a

endif

pos\_tmp(3)=pos(3,i\_1)

pos\_tmp(1)=pos(1,i\_1)

r0 = sqrt( dot\_product(pos\_tmp(:)-pos(:,i\_0), pos\_tmp(:)-pos(:,i\_0) ) )

if( (r0 > r1) .and. (r0 < r2) )then

i\_nei=i\_nei+1

if(i\_nei > Naround)then

write(\*,'(1x,"### i\_neighbor > Naround ###",I10)') i\_1

stop

endif

around(i\_nei ,i\_1) = i\_0

if(mod(Npl,2)==0)then

around\_shift((i\_nei-1)\*DIM+2,i\_1)=-3\*i\_x\*Npl/2 !20140425 a mistake

else

around\_shift((i\_nei-1)\*DIM+2,i\_1)=-1\* i\_x \* ( 3\*(Npl-1) / 2+ 3 ) !20140425 a mistake

endif ! i\_1 和i\_0 平移符号相反

endif

enddo

enddo cyc43b

endif

if(i\_nei < Naround )then

write(\*,'(1x,"### i\_neighbor < Naround ## I = ###",I10)') i\_1

!stop

endif

enddo cyc41

z\_low=pos(3,1) !20140517qichen

z\_top=pos(3,N) !20140517qichen

! do i=1,Npl

! around\_flag(i)=0 !20140425 qichen

! enddo

! do i=N-Npl+1,N

! around\_flag(i)=0 !20140425 qichen

! enddo

!cyc51:do i=1,N !20140504qichen start !搜索在平面内的原子

! if ((pos(1,i)>plane\_len) .or. (pos(1,i)<0))then

! continue

! else

! plane\_flag(i)=1

! endif

! end do cyc51 !20140504qichen end

RETURN

END SUBROUTINE NEIGHBOR\_LIST ! Part 4 计算生成 Neighbor list

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

! Part 计算生成 Npl\_mov 每一层拥有4个neighour 的粒子个数

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE Get\_Npl\_Move\_LIST ! Part 计算生成 Npl\_mov

!USE DFLIB

use Parameters

implicit none

integer :: i,j,k,cont

double precision :: r

allocate(Npl\_mov(Nlay))

do i=1,Nlay

cont=0

k=sum( Npl\_list(1:i) ) - Npl\_list(i)

do j=1,Npl\_list(i)

if( around(Naround,k+j) /= 0) cont=cont+1

enddo

Npl\_mov(i)=cont

enddo

RETURN

END SUBROUTINE Get\_Npl\_Move\_LIST ! Part 4 计算生成 Neighbor list

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

! Part 计算生成 BC\_list 没有4个neighour 的粒子

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE Get\_BC\_LIST ! Part 计算生成 BC\_LIST

!USE DFLIB

use Parameters

implicit none

integer :: i,j,k,cont

double precision :: r

allocate(BC\_LIST(N))

cont=0

do i=1,N

if( around(Naround,i) == 0)then

cont=cont+1

BC\_list(cont)=i

endif

enddo

N\_BC= cont

RETURN

END SUBROUTINE Get\_BC\_LIST ! Part 4 计算生成 Neighbor list

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

! Part 计算 Van De Waal force's list !!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE Get\_Van\_around\_list

!USE DFLIB

use Parameters

implicit none

integer ::i=0,j=0,k=0,i\_nei=0,i\_nei\_sec=0,mk=0,l1=0,l2=0,l3=0 !20140821qichen !20141222qichen

double precision :: N\_temp !20141019qichen

double precision ::r0 !,r1,r2 ！20141006qichen

double precision,dimension(DIM) ::pos\_tmp

allocate(Van\_num(N))

allocate(Van\_around(Van\_top,N)) !sample: (5,7,8,9 ...) 5 , 7,8,9 ...instead of the number of the atom's neighbor.

allocate(Van\_around\_shift(Van\_top,N)) !sample: (0,0,1,0,5,5.....) 15 stands for 15 neighbors 0,0,1 instead of the coordinate of the first neighbor 0,5,5 the second,....

allocate(Neib\_sec\_near(9,N)) !to record each atom's second nearest neib no. 20141222qichen

allocate(slp2(Nlay-2,2)) !20140821qichen

allocate(slp1(Nlay-1)) !20140821qichen

allocate(slp\_x(Nlay-1)) !20140821qichen

allocate(Mark( 2\*n\_folded)) !20140821qichen

allocate(Plane\_No(N)) !20140821qichen

Van\_num(:)=0

Van\_around(:,:)=0

Van\_around\_shift(:,:)=0

Neib\_sec\_near=0

pos\_tmp(:)=0.d0

slp\_x=0.d0 !20140821qichen

slp1=0.d0 !20140821qichen

slp2=0.d0 !20140821qichen

Mark= 0 !20141226qichen

mk=1 !20140821qichen

Plane\_No=0 !20140821qichen

r\_van\_1=3.0E-10 !关于r\_van\_1的探讨: 次近邻r=a\* 3^0.5= 2.449 故r\_van\_1>2.449 !20141006qichen

r\_van\_2=8.875E-10 !20140520qichen 为了搜到更多邻居 !20141006qichen

!qichen20140821 start

cycv21: do l1=1,Nlay-1

slp\_x(l1) = (sum( pos(1,l1\*Npl+1:(l1+1)\*Npl))- sum( pos(1,(l1-1)\*Npl+1:l1\*Npl)) ) /dble( Npl )

slp1(l1)= ( sum( pos(3,l1\*Npl+1:(l1+1)\*Npl))- sum( pos(3,(l1-1)\*Npl+1:l1\*Npl)) ) / (slp\_x(l1)\*dble(Npl))! 0.5 \* a \* 3^0.5 = 1.228E-10 slp1 表示空间梯度

enddo cycv21

cycv22: do l2=1,Nlay-2

slp2(l2,1)= slp1(l2+1)-slp1(l2)

IF ( (slp2(l2,1)<0.1d0) .and. (slp2(l2,1)> -0.1d0) ) then

slp2(l2,2)=1.d0 ! 这里的0.3是用来鉴别空间梯度变化幅度的一个值，小于0.3代表是平面，大于0.3代表是弯曲部分

else

slp2(l2,2)= 0.d0

endif

enddo cycv22

cycv3: do l3=4,Nlay-5

if (((slp2(l3,2)+slp2(l3+1,2)+slp2(l3+2,2)+slp2(l3+3,2))==0).and.((slp2(l3-1,2)\*slp2(l3-2,2)\*slp2(l3-3,2)) >0 ))then

Mark(mk)=l3

mk=mk+1

elseif (((slp2(l3-3,2)+slp2(l3-2,2)+slp2(l3-1,2)+slp2(l3,2))==0).and.((slp2(l3+1,2)\*slp2(l3+2,2)\*slp2(l3+3,2)) >0 )) then

Mark(mk)=l3+2

mk=mk+1

endif

enddo cycv3

cycv41: do l1=1,(Mark(1)+1)\*Npl !20140903qihcen

Plane\_No(l1)=1

enddo cycv41

cycv42: do l1=1,n\_folded -1

do l2=Mark(2\*l1-1)+1,Mark(2\*l1)-3

!Van\_num( l2\*Npl+1: (l2+1)\*Npl )=0

Plane\_No(l2\*Npl+1: (l2+1)\*Npl)=0

enddo

do l3=Mark(2\*l1)-2,Mark(2\*l1+1)

Plane\_No( l3\*Npl+1:(l3+1)\*Npl)=l1+1

enddo

enddo cycv42

cycv43: do l2= Npl\*(Mark(2\*n\_folded-1)+1)+1,Npl\*(Mark(2\*n\_folded)-2)

!Van\_num( l2\*Npl+1: (l2+1)\*Npl )=0

Plane\_No(l2)=0

enddo cycv43

cycv44: do l1=Npl\*( Mark(2\*n\_folded) -2)+1 ,N

Plane\_No(l1)= n\_folded + 1

enddo cycv44

if ( sum( Plane\_No(:)-Plane\_flag(:))/=0 ) print\*,' WARNING: Please check the code again for distinguish the curve and the plane ' !20140428qichen

!qichen20140821 end

cycv1:do i=1,N

!if(Plane\_No(i)==0)goto 131 !20141226qichen

i\_nei=0

cycv10: do j=1,N !20140830qichen

!if(j==i)goto 138

!if((Plane\_No(j)==0).or.(abs(Plane\_No(i)-Plane\_No(j))/=1))goto 138 !20141226qichen

cycv101:do k=-1,1

pos\_tmp(2)=pos(2,i)+dble(3\*k\*Npl/2)\*a !20140830qichen

pos\_tmp(1)=pos(1,i)

pos\_tmp(3)=pos(3,i)

r0 = sqrt( dot\_product(pos\_tmp(:)-pos(:,j),pos\_tmp(:)-pos(:,j)) )

if((r0<R\_van\_2).and.(r0>R\_van\_1)) then !( abs(j-i)/Npl>10).and.() !20141006qichenthen !20140830qichen .and.(abs(j/Npl-i/Npl)>3)

i\_nei=i\_nei+1

Van\_around(i\_nei,i)=j

Van\_around\_shift(i\_nei,i)= -1\*3\*k\*Npl/2 !20140425 a mistake !20141222qichen

goto 138

endif

enddo cycv101

138 continue

enddo cycv10 !20140830qichen

Van\_num(i)=i\_nei !20140830qichen

! 131 continue !20141226qichen

enddo cycv1

!20150302qichen start to get the amount of atoms on the top plane and the lowest plane

N\_plane\_down = 0 !20150120qichen

N\_plane\_up = 0 !20150120qichen

do i=1,N !20140517qichen

if(Plane\_No(i)==1)then

N\_plane\_down=N\_plane\_down+1 !20140517qichen

endif

if(Plane\_No(i)==n\_folded+1)then

N\_plane\_up=N\_plane\_up+1 !20140517qichen

endif

enddo !20140517qichen

N\_plane\_up=N\_plane\_up+10\*Npl !20150302qichen

N\_plane\_down=N\_plane\_down+10\*Npl !20150302qichen

!20150302qichen start

RETURN

END SUBROUTINE Get\_Van\_around\_list

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

! Part 5 计算生成 高斯位置偏移 常数 offset\_gauss 便宜最大值（暂时不用）

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE GAUSS\_OFFSET\_POSITION ! Part 5 计算生成 高斯位置偏移 常数 offset\_gauss 便宜最大值

!USE DFLIB

use Parameters

implicit none

integer :: i,j,i\_gauss

double precision, dimension(12) :: gauss ! guass 分布的时候用到

real :: ag,ag\_1

offset\_gauss=0.\*a

do i=1+Npl,N-Npl ! CNCone

do j=1,3

do i\_gauss=1,12

CALL RANDOM(ag)

gauss(i\_gauss)=ag

enddo

pos(j,i)=pos(j,i)+(sum(gauss)/6-1)\*offset\_gauss

enddo

enddo

RETURN

END SUBROUTINE GAUSS\_OFFSET\_POSITION ! Part 5 计算生成 高斯位置偏移 常数 offset\_gauss 便宜最大值

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!! Part 6 MASS LIST !!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE compute\_masslist ! graded nework ! Part 6 MASS LIST

use Parameters

INTEGER :: i,j,k

allocate( mass\_list(N) )

mass\_list=1.d0

RETURN

END SUBROUTINE compute\_masslist ! graded nework ! Part 6 MASS LIST

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

! Part 7 计算生成

! 产生随即速度（高斯分布）

! angle1: 产生随即角度（0，180）之间存入angle1(N)

! 要符合线形密度分布 正比于sin(angle1):实现方法 当（0，1）随机数 ag\_1

! 属于[0，sin(angle1)] 则选择这个angle1;如果大于sin(angle1)则丢弃

! angle2: 产生随即角度（0，360）之间 存入angle2(N)

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE RANDOM\_VELOSITIES(i\_case) ! Part 7 计算生成 !20140428qichen

!USE DFLIB

use Parameters

implicit none

integer :: i,j,k,k1,i\_gauss,i\_lay,i\_case ! CNCone !20140428qichen

double precision, dimension(:), allocatable :: v0 ! 温度为T的平均速度

double precision, dimension(12) :: gauss ! guass 分布的时候用到

real(4) :: ag,ag\_1

double precision, dimension(:), allocatable :: angle1

double precision, dimension(:), allocatable :: angle2

allocate( angle1(N))

allocate( angle2(N))

allocate( v0(N))

v0(:) = sqrt(DIM\*T\*kb/(mass\*mass\_list(:)) ) ! DIM

CALL SEED(rdseed(i\_case) ) !MPIf90 ! 为随机数作准备 !20140428qichen

!CALL SEED(13846 ) ! 为随机数作准备 !20140428qichen

!CALL RANDOM\_SEED

!do k=1,Nlay-2

! do j=1,Npl

! i= j + k\*Npl

do i=1+Npl,N-Npl ! CNCone

!

100 CALL RANDOM(ag) !!!!! 实现angle1的非均匀分布，线形分布

angle1(i)= ag\*180.0D0

CALL RANDOM(ag\_1)

if( ag\_1 >= dsind(angle1(i)) ) goto 100

CALL RANDOM(ag)

angle2(i)=ag\*360.0D0

!下面几行产生高斯分布的随机数

! (sum(gauss)-6)/6+1=(sum(gauss)/6) 是遵循gauss分布的随机数 属于(0,2)平均值1

! ( ((sum(gauss)/6)-1) \*2/100+1 )\* 实现 1+/-2%

do i\_gauss=1,12

CALL RANDOM(ag)

gauss(i\_gauss)=ag

enddo

vel(1,i)=( ((sum(gauss)/6)-1) \*2/100+1 )\*v0(i)\*dsind(angle1(i))\*dcosd(angle2(i))

do i\_gauss=1,12

CALL RANDOM(ag)

gauss(i\_gauss)=ag

enddo

vel(2,i)=( ((sum(gauss)/6)-1) \*2/100+1 )\*v0(i)\*dsind(angle1(i))\*dsind(angle2(i))

do i\_gauss=1,12

CALL RANDOM(ag)

gauss(i\_gauss)=ag

enddo

vel(3,i)=( ((sum(gauss)/6)-1) \*2/100+1 )\*v0(i)\*dcosd(angle1(i))

end do

deallocate( v0)

deallocate( angle1)

deallocate( angle2)

do i=1,N

if(around(Naround,i)==0 ) vel(:,i)=0.d0

enddo

RETURN

END SUBROUTINE RANDOM\_VELOSITIES ! Part 7 计算生成

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

! Part ! make momentum of mass centre = 0

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE Initial\_momentum(nstart,nend) ! BPB3\_F90 ! BPB8\_F90 20140428qichen start

use Parameters !20140428qichen

implicit none

INTEGER :: i,j,k,lis, i1,i2,i3,nstart,nend

DOUBLE PRECISION :: moment\_tmp(DIM)

moment\_tmp(:)=0.d0

do i=1,size(moment\_tmp)

moment\_tmp(i)=sum(vel(i,nstart:nend)\*mass\_list(nstart:nend) )

enddo

moment\_tmp(:)=moment\_tmp(:)/dble(nend-nstart+1)

do i=1,size(moment\_tmp)

do j=nstart,nend

vel(i,j)=vel(i,j)-moment\_tmp(i)/mass\_list(j)

enddo

enddo

do i=1,size(moment\_tmp)

moment\_tmp(i)=sum(vel(i,nstart:nend)\*mass\_list(nstart:nend))

enddo

RETURN

END SUBROUTINE Initial\_momentum ! BPB3\_F90 20140428qichen end

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

! Part 9 ! Write out Data

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE WRITE\_OUT(i\_case) ! Part 9 ! Write out Data !20140428qichen

use Parameters

implicit none

integer :: i,i\_case !20140428qichen

character\*80 :: char\_format !20150302qichen

! Write out position velosity acceleration

open(unit=2,file=Sampout(i\_case),status='replace', & !20140428qichen

action='write',err=700)

write(2,'(A1,L2,I6)',err=900) '%',VelAcc,N ! The '%' signals to ignore this line to the BallRoom program producing an image from the simulation

do i=1,N

write(2,'(1X,3E23.15)',err=900) pos(:,i)

enddo

do i=1,N

write(2,'(1X,3E23.15)',err=900) vel(:,i)

enddo

do i=1,N

write(2,'(1X,3E23.15)',err=900) acc(:,i)

enddo

close(unit=2)

! Write out the Van der Waals force neighbor list !20140504qichen start

open(unit=2,file=Sampout\_8(i\_case),status='replace', &

action='write',err=700)

write(2,'(1X,I16,2E23.15)',err=900) Van\_top,r\_van\_1,r\_van\_2!20141006qichen 把搜索半径也输出 !20140520qichen 输出允许最大的van der waals 力的距 “邻居”数

write(char\_format,\*)2\*Van\_top+2

char\_format=adjustl(char\_format)

char\_format="(1x,"//trim(char\_format)//"I8)"

do i=1,N

write(2,char\_format,err=900) Plane\_No(i),Van\_num(i),Van\_around(:,i),Van\_around\_shift(:,i) !I8前面的数应该等于2+Van\_top+DIM\*Van\_top !20140821qichen

enddo

do i=1,N

write(2,'(1X,9I8)',err=900) Neib\_sec\_near(:,i) !20141222qichen for searching second nearest neighbors

enddo

close(unit=2) !20140504qichen start

! Write out Substrate sample neighbor relationship !20150302qichen

open(unit=2,file=sampout\_9(i\_case),status='replace',&

action='write',err=910)

write(char\_format,\*)2\*Van\_top+1

char\_format=adjustl(char\_format)

char\_format="(1x,"//trim(char\_format)//"I8)"

do i=1,N\_plane\_up+N\_plane\_down

write(2,char\_format,err=900) Sub\_grp\_pairs\_num(i),Sub\_grp\_pairs(:,i) !20140425 !20140504qichen

enddo

close(unit=2)

if(i\_case==1)then !BPB12f !20140428qichen

open(unit=2,file=Sampout\_3,status='replace', &

action='write',err=700)

write(2,'("ATOM")',err=900)

do i=1,N

!write(2,200,err=900) i, Pos\_big(:,i)\*1.0E10,1.00, &

write(2,200,err=900) i, Pos2D\_big(:,i)\*1.0E10,1.00, &

temp(DIM+1,i) \*300. /maxval(temp(DIM+1,:) )

enddo

close(unit=2)

! Write out Neighbor list

open(unit=2,file=Sampout\_4,status='replace', &

action='write',err=700)

write(2,'(1X,I16)',err=900)N

write(2,'(1X,I16)',err=900)NPL

write(2,'(1X,I16)',err=900)Nlay

write(2,'(1X,E23.15)',err=900)lay\_dist!20140429qichen

write(2,'(1X,E23.15)',err=900)plane\_len!20140429qichen

write(2,'(1X,I16)',err=900)N\_folded !20140517qichen

write(2,'(1X,E23.15)',err=900)z\_low !20140517qichen

write(2,'(1X,E23.15)',err=900)z\_top !20140517qichen

! write(2,'(1X,E23.15)',err=900)Aera

do i=1,N

write(2,'(1X,14I8)',err=900) around(:,i),around\_shift(:,i),plane\_flag(i),around\_flag(i) !20140425 !20140504qichen

enddo

close(unit=2)

! Write out 2 PDB files（图型文件raswin.exe） Data

open(unit=2,file=Sampout\_2(i\_case),status='replace', &

action='write',err=700)

write(2,'("ATOM")',err=900)

do i=1,N

write(2,200,err=900) i, pos(:,i)\*1.0E10, 1.00, &

temp(DIM+1,i) !\*300. !/maxval(temp(DIM+1,:) )

enddo

close(unit=2)

! Write out Npl

open(unit=2,file=Sampout\_5,status='replace', &

action='write',err=700)

!write(2,'(1X,I16)',err=900)Model\_No

write(2,'(1X,I16)',err=900)N

write(2,'(1X,I16)',err=900)Nlay

!write(2,'(1X,E23.15)',err=900)Area

do i=1,Nlay

write(2,'(1X,I16)',err=900) Npl\_list(i)

enddo

do i=1,Nlay

write(2,'(1X,I16)',err=900) Npl\_mov(i)

enddo

close(unit=2)

! Write out BC\_list

open(unit=2,file=Sampout\_7,status='replace', &

action='write',err=700)

!write(2,'(1X,I16)',err=900)Model\_No

write(2,'(1X,I16)',err=900)N

write(2,'(1X,I16)',err=900)N\_BC

do i=1,N\_BC

write(2,'(1X,I16)',err=900) BC\_list(i)

enddo

close(unit=2)

! Write out Mass list

open(unit=2,file=Sampout\_6,status='replace', &

action='write',err=700)

! write(2,'(1X,I16)',err=900)Model\_No

! write(2,'(1X,I16)',err=900)Miss\_angle

write(2,'(1X,I16)',err=900)Nlay

do i=1,N

write(2,'(1X,F16.10)',err=900) mass\_list(i)

enddo

close(unit=2)

endif !20140428qichen

RETURN

200 FORMAT ('ATOM ',I5,1X,' C ',' ','NTI',1X,' ',' 1',' ',3X,3F8.3,2F6.0,' ',' ',' ',' ') !20140425

700 continue

print\*,'Write\_Sample: WARNING: cannot open ',SampOut(i\_case),' for write' !20140428qichen

close(unit=2)

stop

900 continue

print\*,'Write\_Sample: WARNING: error when writing ',SampOut(i\_case) !20140428qichen

close(unit=2)

stop

910 continue

print\*,'Write\_Sample: WARNING: error when writing ',SampOut\_9(i\_case) !20140428qichen

close(unit=2)

stop

END SUBROUTINE WRITE\_OUT

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

! Part 10 ! 优化结构

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

subroutine Evolve\_Sample ! Part 10 ! 优化结构

use Parameters

!use Simulation\_Control

!use Statistics

implicit none

INTEGER :: step, step\_big, i\_t, i, j,d,I\_k ,SUM\_L ,SUM\_L2, N\_SIDE,i\_fft ,l1,l2

DOUBLE PRECISION :: ene\_kin\_aver,ene\_pot\_aver,ene\_tot\_aver,temperature, sum\_a,&

SUM\_TL,SUM\_T

!DOUBLE PRECISION, DIMENSION(:,:),ALLOCATABLE:: zita\_minus, zita\_plus

INTEGER :: Nsteps= 2\*\*2 !12

DOUBLE PRECISION :: deltat= 4.E-16

DOUBLE PRECISION :: rou != 1.E14 !阻尼系数

DOUBLE PRECISION, DIMENSION(DIM,N):: fri\_for

rou= 0.D0 !1.E13

fri\_for=0.d0

time: do step=1,Nsteps

do i=1,DIM

fri\_for(i,:)= -1.d0\*vel(i,1:N)\*rou

enddo

pos(:,1:N ) = pos(:,1:N) + deltat\*vel(:,1:N) &

+ 0.5d0\*(deltat\*deltat)\*( acc(:,1:N) + fri\_for(:,1:N) )

vel(:,1:N) = vel(:,1:N) + 0.5d0\*deltat\*( acc(:,1:N) + fri\_for(:,1:N) )

acc=0.d0

call Compute\_Forces

vel(:,1:N) = vel(:,1:N) + 0.5d0\*deltat\*( acc(:,1:N) + fri\_for(:,1:N) )

enddo time

RETURN

end subroutine Evolve\_Sample

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

subroutine Compute\_Forces

use Parameters

use SWPotential

!use Statistics

implicit none

INTEGER :: i,j1,j01,j02

DOUBLE PRECISION,DIMENSION(DIM):: F2\_all ! store 2-doby-force

DOUBLE PRECISION,DIMENSION(Naround):: For\_sca ! Force scalar ! store 2-doby-force

DOUBLE PRECISION,DIMENSION(Naround,DIM):: For\_vec ! Force vector ! store 2-doby-force

DOUBLE PRECISION,DIMENSION(DIM,Naround):: derta\_i

DOUBLE PRECISION,DIMENSION(DIM,3):: For\_ang

INTEGER,DIMENSION(Naround) :: k\_i

DOUBLE PRECISION,DIMENSION(Naround) :: Rik ! store 2-doby-force

DOUBLE PRECISION,DIMENSION(DIM,Naround,Naround):: derta\_kk

DOUBLE PRECISION,DIMENSION(Naround,Naround) :: cosalfa,Rkk

For\_sca = 0.d0 ! store 2-doby-force

For\_vec = 0.d0 ! store 2-doby-force

derta\_kk = 0.d0

cosalfa = 0.d0

Rkk = 0.d0

**!$omp parallel do &**

**!$omp default(shared) &**

**!$omp private(i,j1,j01,j02) &**

**!$omp private(F2\_all,Rkk,cosalfa) &**

**!$omp private(For\_sca,For\_vec,derta\_i,For\_ang,k\_i,Rik,derta\_kk)**

cyc2: do i = 1, N ! cyc2 calculate particles except boundary layer particles

if(around(Naround,i) /= 0)then ! fixBC

! 初始化变量

cyc20: do j1=1,Naround ! store 2-doby-force

k\_i(j1) = around(j1,i)

if( k\_i(j1)==0)then ! EMPTY NEIGHBOR

derta\_i(:,j1) = 1.d0 ! 设得很大 使得2体力截至函数=0

Rik(j1) = 1.d0

else

derta\_i(:,j1) = pos(:,i)-pos(:,k\_i(j1))

Rik(j1) = sqrt(dot\_product(derta\_i(:,j1) ,derta\_i(:,j1) ))

endif

enddo cyc20

cyc21: do j01=1,Naround-1 ! create half triangle Matrix: derta\_kk,Rkk,Cosalfa

do j02=j01+1,Naround

if(k\_i(j01)==0 .or. k\_i(j02)==0)then ! boundary particle

derta\_kk(:,j01,j02) =0.d0

Rkk(j01,j02) =0.d0

Cosalfa(j01,j02) =0.d0

derta\_kk(:,j02,j01) =0.d0

Rkk(j02,j01) =0.d0

Cosalfa(j02,j01) =0.d0

else

derta\_kk(:,j01,j02) =pos(:,k\_i(j01))-pos(:,k\_i(j02))

Rkk(j01,j02) = sqrt(dot\_product(derta\_kk(:,j01,j02) ,derta\_kk(:,j01,j02) ))

Cosalfa(j01,j02) = (Rik(j01)\*Rik(j01) + Rik(j02)\*Rik(j02) - Rkk(j01,j02)\*Rkk(j01,j02)) / (2.0d0\*Rik(j01)\*Rik(j02))

derta\_kk(:,j02,j01) = derta\_kk(:,j01,j02)

Rkk(j02,j01) = Rkk(j01,j02)

Cosalfa(j02,j01) = Cosalfa(j01,j02)

endif

enddo

enddo cyc21

!SW start

! 2-doby-force

cyc22: do j1=1,Naround ! store 2-doby-force

For\_sca(j1) = -1.d0\*SW\_epsl\*fun\_f2(Rik(j1))/SW\_sigm

For\_vec(j1,:) = For\_sca(j1)\*derta\_i(:,j1) /Rik(j1)

enddo cyc22 ! store 2-doby-force

F2\_all = sum(For\_vec,dim=1) ! store 2-doby-force

Acc(:,i) = Acc(:,i) + F2\_all / (mass\*mass\_list(i))

! langevin

! flux1(i) = flux1(i) + 0.5d0\*( dot\_product(derta\_i(:,1),fluxdir ) \*dot\_product( vel(:,i), For\_vec(1,:)) &

! + dot\_product(derta\_i(:,2),fluxdir ) \*dot\_product( vel(:,i), For\_vec(2,:) ) &

! !+ dot\_product(derta\_i(:,3),fluxdir ) \*dot\_product( vel(:,i), For\_vec(3,:) ) &

! + dot\_product(derta\_i(:,Naround),fluxdir) \*dot\_product( vel(:,i),For\_vec(Naround,:) ) ) ! store 2-doby-force

! 3-doby-force

cyc23: do j01=1,Naround-1

do j02=j01+1,Naround

!NotBC: if( (Rik(j01) < L\_u) .and. (Rik(j02)<L\_u ) )then ! 边界粒子的空健 不被考虑

NotBC: if( k\_i(j01)/=0 .and. k\_i(j02)/=0 )then ! 边界粒子的空健 不被考虑

!SW start

For\_ang(:,1) = -1.d0\*SW\_epsl\*fun\_f3(derta\_i(:,j01),Rik(j01),derta\_i(:,j02),Rik(j02),Cosalfa(j01,j02)) /SW\_sigm

For\_ang(:,2) = -1.d0\*SW\_epsl\*fun\_f3(derta\_i(:,j02),Rik(j02),derta\_i(:,j01),Rik(j01),Cosalfa(j01,j02)) /SW\_sigm

For\_ang(:,3) = - For\_ang(:,1) - For\_ang(:,2)

!SW end

Acc(:,k\_i(j01)) = Acc(:,k\_i(j01))+For\_ang(:,1) / (mass\*mass\_list(k\_i(j01)))

Acc(:,k\_i(j02)) = Acc(:,k\_i(j02))+For\_ang(:,2) / (mass\*mass\_list(k\_i(j02)))

Acc(:,i) = Acc(:,i) +For\_ang(:,3) / (mass\*mass\_list(i))

! langevin

! flux2(k\_i(j01))=flux2(k\_i(j01)) - dot\_product(derta\_i(:,j01),fluxdir)\*dot\_product(vel(:,k\_i(j01)),For\_ang(:,1))

! flux2(k\_i(j02))=flux2(k\_i(j02)) - dot\_product(derta\_i(:,j02),fluxdir)\*dot\_product(vel(:,k\_i(j02)),For\_ang(:,2))

endif NotBC

enddo

enddo cyc23

endif ! fixBC

end do cyc2

**!$omp end parallel do**

return

end subroutine Compute\_Forces

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

! Part 11 温度纪录

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE compute\_temperature ! Part 11

!USE DFLIB

use Parameters

implicit none

integer :: i,j,k,k1,i\_gauss,i\_lay ! CNCone

!! 动能 温度

temp =0.d0

j=DIM+1

do i= 1, DIM

temp(i,:) = temp(i,:)+ vel(i,:) \* vel(i,:)

temp(j,:) = temp(j,:)+ vel(i,:) \* vel(i,:)

end do

temp = temp\* mass/2.0d0

do j=1,DIM

temp(j,:) = 2.0d0 \* temp(j,:)\*mass\_list(:) /kb

enddo

temp(DIM+1,:) = 2.0d0 \* temp(DIM+1,:)\*mass\_list(:) /(kb \* DIM)

RETURN

END SUBROUTINE compute\_temperature ! Part 11

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE Get\_Sub\_grp\_pairs !Part 基底原子邻居列表 !20150302qichen

!USE DFLIB

use Parameters

implicit none

integer ::i=0,j=0,k=0,i\_nei=0

double precision ::r0

double precision,dimension(DIM) ::pos\_tmp

allocate(Sub\_grp\_pairs(2\*Van\_top,N\_plane\_up+N\_plane\_down)) !20150302qichen

allocate(Sub\_grp\_pairs\_num(N\_plane\_up+N\_plane\_down)) !20150302qichen

Sub\_grp\_pairs\_num(:)=0

Sub\_grp\_pairs(:,:)=0

pos\_tmp=0.d0

do i=1,N\_plane\_down

i\_nei=0

pos\_tmp(1)=pos\_sub\_down(1,i)

pos\_tmp(3)=pos\_sub\_down(3,i)

do j=1,N

do k=-1,1

pos\_tmp(2)=pos\_sub\_down(2,i)+dble(3\*k\*Npl/2)\*a !20140830qichen

r0 = sqrt( dot\_product(pos\_tmp(:)-pos(:,j),pos\_tmp(:)-pos(:,j)) )

if((r0 < r\_van\_2).and.(r0>r\_van\_1)) then

i\_nei=i\_nei+1

Sub\_grp\_pairs(i\_nei,i)=j

Sub\_grp\_pairs(i\_nei+Van\_top,i)= -3\*k\*Npl/2

endif

enddo

enddo

Sub\_grp\_pairs\_num(i)=i\_nei

enddo

do i=N\_plane\_down+1,N\_plane\_down+N\_plane\_up

i\_nei=0

pos\_tmp(1)=pos\_sub\_up(1,i-N\_plane\_down)

pos\_tmp(3)=pos\_sub\_up(3,i-N\_plane\_down)

do j=1,N

do k=-1,1

pos\_tmp(2)=pos\_sub\_up(2,i-N\_plane\_down)+dble(3\*k\*Npl/2)\*a !20140830qichen

r0 = sqrt( dot\_product(pos\_tmp(:)-pos(:,j),pos\_tmp(:)-pos(:,j)) )

if((r0 < r\_van\_2).and.(r0>r\_van\_1)) then

i\_nei=i\_nei+1

Sub\_grp\_pairs(i\_nei,i)=j

Sub\_grp\_pairs(i\_nei+Van\_top,i)= -3\*k\*Npl/2

endif

enddo

enddo

Sub\_grp\_pairs\_num(i)=i\_nei

enddo

END SUBROUTINE Get\_Sub\_grp\_pairs !20150302qichen

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

! Part 生成基底原子坐标

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

SUBROUTINE Initiating\_substrate !20150120qichen start

!USE DFLIB

use Parameters

implicit none

integer ::i=0,j=0,ntemp=0

double precision,dimension(2,N\_plane\_up)::abtemp

DOUBLE PRECISION, PARAMETER :: Sigma = 3.3407d-10

DOUBLE PRECISION, PARAMETER :: Z\_balance = Sigma\*(2.d0)\*\*(1.d0/6.d0) !20140517qichen!20140622qichen equilibrium distant [m]

allocate(pos\_sub\_down(DIM,2\*N\_plane\_down))

allocate(pos\_sub\_up(DIM,2\*N\_plane\_up))

pos\_sub\_down=0.d0

pos\_sub\_up =0.d0

pos\_sub\_down(3,:) = z\_low-Z\_balance

pos\_sub\_up(3,:) = z\_top+Z\_balance

do i=0,1

do j=1,Npl\_list(1)

pos\_sub\_down(1:2,i\*Npl\_list(1)+j)=pos(1:2,i\*Npl\_list(1)+j)

pos\_sub\_up (1:2,i\*Npl\_list(1)+j)=pos(1:2,N-(i\*Npl\_list(1)+j)+1)

enddo

enddo

ntemp=(size(pos\_sub\_down,dim=2))/Npl\_list(1)-1

do i=2,ntemp

do j=1,Npl\_list(1)

pos\_sub\_down(2,i\*Npl\_list(1)+j)=pos\_sub\_down(2,(i-2)\*Npl\_list(1)+j)

pos\_sub\_down(1,i\*Npl\_list(1)+j)=pos\_sub\_down(1,(i-2)\*Npl\_list(1)+j)+2.0\*(pos\_sub\_down(1,Npl\_list(1)+1)-pos\_sub\_down(1,1))

enddo

enddo

ntemp=(size(pos\_sub\_up,dim=2))/Npl\_list(1)-1

do i=2,ntemp

do j=1,Npl\_list(1)

pos\_sub\_up(2,i\*Npl\_list(1)+j)=pos\_sub\_up(2,(i-2)\*Npl\_list(1)+j)

pos\_sub\_up(1,i\*Npl\_list(1)+j)=pos\_sub\_up(1,(i-2)\*Npl\_list(1)+j)+2.0\*(pos\_sub\_up(1,Npl\_list(1)+1)-pos\_sub\_up(1,1))

enddo

enddo

abtemp=pos\_sub\_up(1:2,1:N\_plane\_up)- pos(:,N:N-N\_plane\_up+1:-1)

abtemp=pos\_sub\_down(1:2,1:N\_plane\_down)- pos(:,1:N\_plane\_down)

RETURN

909 continue

print\*,'## FATAL, POSSIBLE CAUSES:1.errors of N\_plane\_up/down readin 2. errors of pos readin '

stop

RETURN

END !20150120qichen end