VIETNAM JAPAN UNIVERSITY
MASTER PROGRAM IN NANO TECHNOLOGY

NANOMECHANICS LECTURE NOTES

INTRODUCTION ON MOLECULAR DYNAMICS SIMULATION

DR. TIEN QUANG NGUYEN
Specially-Appointed Assistant Professor
Division of Materials and Manufacturing Science,
Graduate School of Engineering, Osaka University
2-1 Yamadaoka, Suita-shi, Osaka 565-0871, JAPAN

Email: quang@mat.eng.osaka-u.ac.jp

Revision Date: Dec 04, 2021

OSAKA UNIVERSITY | 2017-2021

Table of Contents

1. Introduction

The molecular dynamics (MD) method was first introduced by Alder and Wainwright in the late 1950s to study the interactions of hard spheres [1,2]. The next major advance was in 1964, when Rahman performed the first simulation using a realistic potential for liquid argon [3]. And the first molecular dynamics simulation of a realistic system was done by Rahman and Stillinger for liquid water in 1974 [4]. Since then, due to the revolutionary advances in computer technology and algorithmic improvements, molecular dynamics has subsequently become one of the principle tool in many areas of physics, chemistry, biology and materials science. Basically, molecular dynamics is a numerical computation technique that can simulate the behaviour of materials at the atomic scale (microscopic level), including atomic positions and velocities. From these microscopic information, by using statistical mechanics, one can convert to macroscopic observables such as pressure, energy, heat capacities, etc. There are two main families of molecular dynamics methods, which is categorized based on model chosen to represent a physical system. One is classical molecular dynamics and the other is quantum or *ab initio* molecular dynamics. In classical molecular dynamics, molecules are treated as classical objects and ruled by the laws of classical mechanics. While in quantum molecular dynamics, which introduced in the 1980s by Car and Parinello [5], the quantum mechanical effect of the electrons is included in the calculation of energy and forces for the classical motion of the nuclei. Of course the quantum version gives an important improvement over the classical approach. However, they requires more computational resources and the size of simulated system is limited to few hundreds of atoms. For the simulation of systems comprising many thousands (or millions) of atoms, which are usually found in biology and materials science fields, the classical molecular dynamics approach is more practical. And hence, in this chapter, the basic formalism, techniques and applications of classical molecular dynamics will be introduced.

2. Equations of Motion

The classical molecular dynamics method (hereafter, it will be simply called molecular dynamics) is based on Newton's second law. We begin with a system of N particles which is governed by the Newton's equations of motion (EOM):

$$F_i = m_i a_i \tag{2.1}$$

where F_i is the force exerted on particle i, m_i is the mass of particle i and $a_i = d^2 r_i / dt^2$ is the acceleration of particle i.

The computation of the force involves the calculation of the derivative (the gradient) of the interacting potential $U(r_1, r_2, ..., r_N)$, which is a function of the atomic positions r_i of all the atoms in the system:

$$F_i = -\nabla_i U(r_1, r_2, ..., r_N) = -\frac{\partial U}{\partial r_i} \tag{2.2}$$

Combining (2.1) and (2.2) yields:

$$\frac{d^2 r_i}{dt^2} = -\frac{1}{m_i}\frac{\partial U}{\partial r_i} \tag{2.3}$$

The essential task in molecular dynamics simulation is to solve the above equations of motion (second-order differential equation). With a given interacting potential, one can find the trajectories of the particles for an interval time frame, the velocities of the particles and other physical quantities (both micro- and macroscopic). It should be noted that the Newton's equations of motion are time-reversible. That means, if we start at one microscopic state m and proceed to another one n in time Δt, then reversing all of the atomic momenta at n and proceeding according to the Newton's equations for Δt will take us back to the state m along exact trajectory.

The equations of motion can also be developed in a more general formulation for any coordinate system by using Lagrangian or Hamiltonian from classical mechanics.

Let $q = (q_1, q_2, ..., q_N)$ and $p = (p_1, p_2, ..., p_N)$ be generalized atomic coordinates and conjugate momenta, respectively. The Lagrangian formulation is defined in term of kinetic energy, K, and potential energy, U and it is a function of the atomic coordinates and their time derivatives as follows:

$$L(q, \dot{q}) = K(\dot{q}) - U(q) \tag{2.4}$$

Using the Lagrangian, the equations of motion for any coordinate are derived from the Euler-Lagrange equations:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = 0 \tag{2.5}$$

If we consider a system in Cartesian coordinates and define the kinetic energy as:

$$K = \sum_i \frac{m_i v_i^2}{2} = \sum_i \frac{m_i \dot{r}_i^2}{2} \tag{2.6}$$

then the equations of motion become:

$$\frac{d^2 r_i}{dt^2} = -\frac{1}{m_i}\frac{\partial U}{\partial r_i} \tag{2.7}$$

which is exactly the same as the equation (2.3) above.

In Hamiltonian formalism, the Hamiltonian is defined as a function of the atomic coordinates and the conjugate momenta (instead of the coordinate time derivatives):

$$H(q, p) = K(p) + U(q) \tag{2.8}$$

Note that, the Hamiltonian is a Legendre transform of the Lagrangian and it simply returns the total energy of the system. With these generalized coordinates, the equations of motion can be expressed as:

$$\frac{\partial p_i}{\partial t} = -\frac{\partial H}{\partial q_i} \tag{2.9}$$

$$\frac{\partial q_i}{\partial t} = \frac{\partial H}{\partial p_i} \tag{2.10}$$

For Cartesian coordinates, the equations of motion become:

$$\frac{\partial p_i}{\partial t} = -\frac{\partial U}{\partial r_i} \tag{2.11}$$

$$\frac{\partial r_i}{\partial t} = \frac{\partial K}{\partial p_i} \tag{2.12}$$

If the kinetic energy also has the form of the equation (2.6), we can easily derive the equation (2.3) from the equations (2.11) and (2.12).

Basically, the advantage of using the Lagrangian or Hamiltonian formalisms for the equations of motion is that we can treat the simulated system in any coordinate. In fact, they are very powerful methods to solve problems in classical mechanics, quantum mechanics, statistical mechanics, etc.

3. The Integration Algorithms

Due to the complicated nature of the interacting potential, there is no analytical solution to the equations of motion. They must be solved numerically. Several numerical schemes (integrators) are available based on finite difference methods (where time is discretized on a finite grid) such as Verlet algorithm, Beeman algorithm, Gear algorithm, etc. Choosing which algorithm to use, one should consider the following criteria: (1) The algorithm should conserve energy and momentum; (2) It should approximate the true trajectory very well; (3) It should permit a long time-step for integration; and (4) It should be computationally efficient (easy to implement, fast, use little memory,…). Note that, all the above points are not equally important. For instance, as the force calculation is typically the most time-consuming part in molecular dynamics simulation, the efficiency of an integrator is usually of less concern than the ability to use a large time-step. Below, several popular integration algorithms will be described. Before going into details, let assume the position and velocity vectors be r (as above) and v, respectively. All these physical quantities at a future time $t+\Delta t$ can be approximately expressed based on the quantities at the current time t by using Taylor expansions:

$$r(t+\Delta t) = r(t) + v(t)\Delta t + \frac{1}{2!}\frac{dv(t)}{dt}(\Delta t)^2 + O\left[(\Delta t)^3\right] \tag{3.1}$$

$$v(t+\Delta t) = v(t) + \frac{dv(t)}{dt}\Delta t + \frac{1}{2!}\frac{d^2v(t)}{dt^2}(\Delta t)^2 + O\left[(\Delta t)^3\right] \tag{3.2}$$

The Newton's equations of motion is:

$$F(t) = m\frac{dv(t)}{dt} \;\Rightarrow\; a(t) = \frac{dv(t)}{dt} = \frac{F(t)}{m} \tag{3.3}$$

where Δt is the time-step.

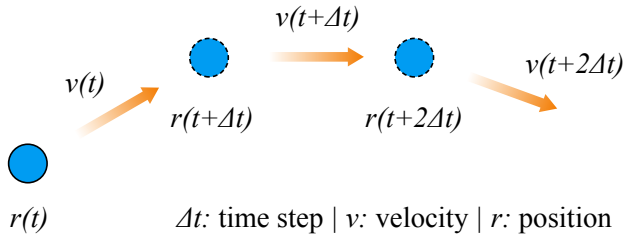

Figure 3.1. Molecular dynamics trajectory from numerical solution

For practical calculations, we should truncate the Taylor expansion at a finite order of time-step Δt. Because of that, there are computation errors associated with the integration schemes. In particular, one can distinguish between truncation errors and round-off errors. The formers are related to the accuracy of the finite different method with respect to the true solution. These errors do not depend on the implementation but the truncation of the Taylor expansion so they are intrinsic to the integration algorithm. The laters are related to errors associated to a particular implementation of the integration algorithm such as the number of digits used in computer floating point arithmetics. Both errors can be reduced by decreasing time-step. For large Δt, truncation error dominates and decreases quickly as Δt is decreased; while for the round-off error, it decreases more slowly with decreasing Δt, and dominates in the small Δt limit.

3.1. Verlet Algorithm

One of the most popular integrators in molecular dynamics simulations is the Verlet algorithm. It was first used in 1791 by Delambre, and has been rediscovered many times since then, most recently by Verlet in the 1960s for use in molecular dynamics [6]. The basic idea is to use two third-order Taylor expansions for the positions, one is a forward time-step and one is is a backward time-step:

$$r(t+\Delta t) = r(t) + v(t)\Delta t + \frac{1}{2!}\frac{dv(t)}{dt}(\Delta t)^2 + \frac{1}{3!}\frac{d^2v(t)}{dt^2}(\Delta t)^3 + O\left[(\Delta t)^4\right] \tag{3.4}$$

$$r(t-\Delta t) = r(t) - v(t)\Delta t + \frac{1}{2!}\frac{dv(t)}{dt}(\Delta t)^2 - \frac{1}{3!}\frac{d^2v(t)}{dt^2}(\Delta t)^3 + O\left[(\Delta t)^4\right] \quad (3.5)$$

Adding (3.4) and (3.5) side-by-side will give:

$$r(t+\Delta t) = 2r(t) - r(t-\Delta t) + \frac{dv(t)}{dt}(\Delta t)^2 + O\left[(\Delta t)^4\right] \quad (3.6)$$

or

$$r(t+\Delta t) = 2r(t) - r(t-\Delta t) + \frac{F(t)}{m}(\Delta t)^2 + O\left[(\Delta t)^4\right] \quad (3.7)$$

We can see that the first and third-order terms from the Taylor expansion cancel out, thus making the Verlet integrator an order more accurate than integration by simple Taylor expansion alone. One problem here is that the velocities are not directly generated. While they are not required to compute the trajectories, they are useful for the calculation of kinetic energy and hence, to verify the conservation of total energy and ensure that the molecular dynamics simulation process is correct. Usually, the velocities can be evaluated from the positions by using:

$$v(t) = \frac{r(t+\Delta t) - r(t-\Delta t)}{2\Delta t} \quad (3.8)$$

As we can see from (3.6) or (3.7), the truncate error in calculated positions is of the order of Δt^4, even if the third derivatives do not appear explicitly. The error associated to velocities is of the order of Δt^2.

The Verlet algorithm flow is as follows:
(1) Given current position and the position at the end of previous step
(2) Compute force at the current position
(3) Compute new position from the present and previous positions, and the present force
(4) Advance to next time step and repeat (1) → (3)

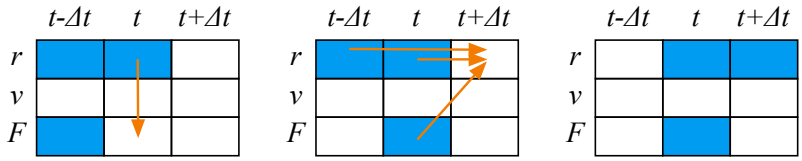

Figure 3.2. Verlet integration flow diagram

The advantage of this algorithm is time-reversible and its simple form (allows to code a program easily). In addition, during the calculation process, we only need to record the two latest positions of particles so the memory requirements are modest. The disadvantage is that the total energy fluctuates heavily although there is no drift in energy. The expression for positions contains two large terms and one small term which may result in numerical problems. In general, the algorithm is of moderate precision.

3.2. Leap-frog Verlet Algorithm

For improvement of the precision of the Verlet algorithm, a half-step 'leap-frog Verlet algorithm' has been proposed. In this algorithm, the velocities are first calculated at time $t + \Delta t/2$; these are used to calculate the positions at time $t + \Delta t$:

$$r(t+\Delta t) = r(t) + v(t+\frac{1}{2}\Delta t)\Delta t \quad (3.9)$$

$$v(t+\frac{1}{2}\Delta t) = v(t-\frac{1}{2}\Delta t) + a(t)\Delta t \tag{3.10}$$

In this modified Verlet algorithm, the calculation of mid-step velocities, $v(t-\Delta t/2)$, is implemented first and the velocities leap over the positions to give the next mid-step values, $v(t+\Delta t/2)$, then the positions leap over the velocities. To calculate the total energy at time t, the velocities can be approximated by:

$$v(t) = \frac{1}{2}\left[v(t+\frac{1}{2}\Delta t) + v(t-\frac{1}{2}\Delta t)\right] \tag{3.11}$$

The leap-frog Verlet algorithm flow is as follows:
(1) Given current position and the velocity at the last half-step
(2) Compute force at the current position
(3) Compute new velocity at the next half-step
(4) Compute new position from the present positions and the next half-step velocity
(5) Advance to next time step and repeat (1) → (4)

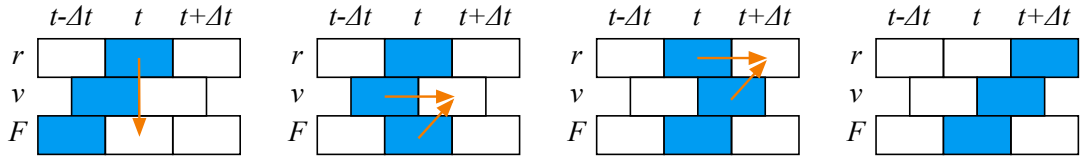

Figure 3.3. Leap-frog Verlet integration flow diagram

Compare to the Verlet algorithm, the leap-frog algorithm differ in what variables are stored in memory and at what times. The advantage of this algorithm is that we do not need to take difference of two large quantities to get a small one (thus no loss of precision) and the velocities are explicitly calculated, however, the disadvantage is that they are not calculated at the same time as the positions.

3.3. Velocity Verlet Algorithm

In order to avoid the asynchrony in time of positions and velocities, an even better implementation of the Verlet algorithm has been proposed which utilizes positions, velocities and accelerations, all at the same time. It is called 'velocity Verlet algorithm'. In this algorithm, the positions and velocities is calculated as follows:

$$r(t+\Delta t) = r(t) + v(t)\Delta t + \frac{1}{2}a(t)(\Delta t)^2 \tag{3.12}$$

$$v(t+\Delta t) = v(t) + \frac{1}{2}\left[a(t) + a(t+\Delta t)\right]\Delta t \tag{3.13}$$

In this scheme, positions, velocities and accelerations at time t are stored. The implementation involves two steps. Firstly, the new positions at time $t+\Delta t$ are calculated from equation (3.12), and the velocities at mid-step are computed from:

$$v(t+\frac{1}{2}\Delta t) = v(t) + \frac{1}{2}a(t)\Delta t \tag{3.14}$$

Secondly, the forces and accelerations at time $t+\Delta t$ are calculated, the new velocities are then obtained from:

$$v(t+\Delta t) = v(t+\frac{1}{2}\Delta t) + \frac{1}{2}a(t+\Delta t)\Delta t \tag{3.15}$$

As we see, the equation (3.13) can be obtained by substituting (3.14) to (3.15).

The velocity Verlet algorithm flow is as follows:
(1) Given current position, velocity and force
(2) Compute new position from the present position, velocity and force
(3) Compute new velocity at the next half-step from the present velocity and force
(4) Compute force at the new position
(5) Compute new velocity at full step
(6) Advance to next time step and repeat (1) → (5)

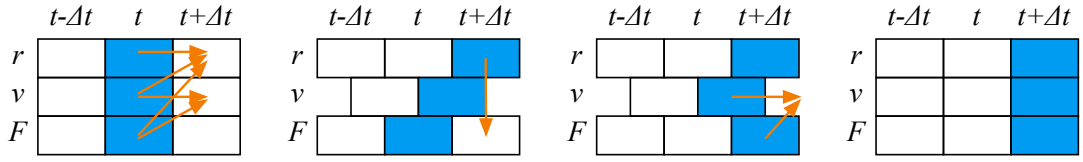

Figure 3.4. Velocity Verlet integration flow diagram

Thus, the total energy calculation in this algorithm is available at time $t + \Delta t$ without an extra step of evaluation of the velocities. The velocity Verlet algorithm inherits the advantages of previous methods with reasonable accuracy. However, it still shows high fluctuations in total energy (but no drift).

3.4. Predictor-corrector Algorithm

Predictor-corrector algorithms belong to another commonly used class of methods for integrating the equations of motion. Among the methods using the predictor-corrector scheme, the most popular one is the Gear predictor-corrector method [7]. The calculation procedure consists of three steps as described below.

Firstly, from the positions r and their time derivatives up to a certain order at time t, one predicts the same quantities at time $t + \Delta t$:

$$\tilde{q}(t + \Delta t) = Gq(t) \tag{3.16}$$

where G is simply a Pascal's upper-triangular matrix:

$$G = \begin{pmatrix} 1 & 1 & 1 & 1 & \cdots \\ 0 & 1 & 2 & 3 & \cdots \\ 0 & 0 & 1 & 3 & \cdots \\ 0 & 0 & 0 & 1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \tag{3.17}$$

and $q(t)$ is a vector consisting of the coordinates $r(t)$ and their time derivatives components taken from the Taylor expansion of $r(t)$:

$$q(t) = \begin{pmatrix} r(t) \\ (dr(t)/dt)\Delta t \\ (d^2r(t)/dt^2)(\Delta t)^2/2! \\ (d^3r(t)/dt^3)(\Delta t)^3/3! \\ \vdots \end{pmatrix} \tag{3.18}$$

This is called the predictor step. If $q(t)$ is chosen as a $(p+1)$-component vector, then we call it the p-th order Gear method. In molecular dynamics simulations, the fifth-order predictor (Gear5 algorithm) is quite often used. Using the notation:

$$r^{(n)} = \frac{1}{n!}\frac{d^n r}{dt^n}(\Delta t)^n \tag{3.19}$$

and

$$\tilde{r}^{(n)} = \frac{1}{n!}\frac{d^n \tilde{r}}{dt^n}(\Delta t)^n \tag{3.20}$$

the predictor $\tilde{r}_n$ for the Gear5 algorithm can be written as:

$$\begin{pmatrix} \tilde{r}^{(0)}(t+\Delta t) \\ \tilde{r}^{(1)}(t+\Delta t) \\ \tilde{r}^{(2)}(t+\Delta t) \\ \tilde{r}^{(3)}(t+\Delta t) \\ \tilde{r}^{(4)}(t+\Delta t) \\ \tilde{r}^{(5)}(t+\Delta t) \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 & 5 \\ 0 & 0 & 1 & 3 & 6 & 10 \\ 0 & 0 & 0 & 1 & 4 & 10 \\ 0 & 0 & 0 & 0 & 1 & 5 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r^{(0)}(t) \\ r^{(1)}(t) \\ r^{(2)}(t) \\ r^{(3)}(t) \\ r^{(4)}(t) \\ r^{(5)}(t) \end{pmatrix} \tag{3.21}$$

Secondly, using the predicted coordinates $\tilde{q}(t+\Delta t)$, the force $\tilde{F}(t+\Delta t)$ is evaluated by taking the gradient of the interacting potential at the predicted positions. The resulting acceleration from this force in general differs from the predicted acceleration. The difference will form an 'error signal':

$$\Delta a(t+\Delta t) = \frac{\tilde{F}(t+\Delta t)}{m} - \frac{d^2\tilde{r}(t+\Delta t)}{dt^2} \tag{3.22}$$

Finally, the positions and their derivatives are corrected using the 'error signal' as follows:

$$q(t+\Delta t) = \tilde{q}(t+\Delta t) + \frac{1}{2}c\Delta a(t+\Delta t)(\Delta t)^2 \tag{3.23}$$

or

$$q(t+\Delta t) = \tilde{q}(t+\Delta t) + c(\Delta t)^2 \times \frac{1}{2}\left[\frac{\tilde{F}(t+\Delta t)}{m} - \frac{d^2\tilde{r}(t+\Delta t)}{dt^2}\right] \tag{3.24}$$

where c is a constant vector called the correction vector. Basically, there is no unique choice for c. The best choice for the vector c depends on how many derivatives of $r(t)$ are used. For the fifth-order Gear algorithm (Gear5), the correction vector is chosen as follows:

$$c = \begin{pmatrix} 3/20 \\ 251/360 \\ 1 \\ 11/18 \\ 1/6 \\ 1/60 \end{pmatrix} = \begin{pmatrix} 0.1500 \\ 0.6972 \\ 1.0000 \\ 0.6111 \\ 0.1667 \\ 0.0167 \end{pmatrix} \tag{3.25}$$

The p-th order Gear algorithm flow is as follows:

(1) Given current position and its time derivatives up to p order

(2) Compute the predicted coordinates at the next step from the current coordinates and matrix G

(3) Compute force at the next step from the predicted coordinates above
(4) Compute the error signal and the corrected coordinates with the use of vector c
(5) Advance to next time step and repeat (1) → (4)

The Gear predictor-corrector algorithms have very small energy fluctuations (good local energy conservation), but they are not time-reversible and the energy error increases linearly with time (long-term energy drift). As compared to the Verlet algorithm, the Gear algorithm has advantage for small time-steps. In addition, similar to the velocity Verlet algorithm, having the velocities available is convenient for implementing the temperature control via velocities. However, since many higher-order time derivatives of the positions are needed, therefore, they requires more computational resources.

Practically, if the stability in the total energy of the system is required, the velocity Verlet integrator is a good choice. It is widely used and well tested for different systems. On the other hand, if small fluctuations in energy is needed and the simulations are short enough, the Gear5 algorithm can be used. For the improvement of the predictor-corrector methods, the readers can refer to a newer algorithm by Martyna and Tuckerman [8]. This new predictor-corrector algorithm is theoretically well-motivated, time-reversible and comparable to Gear4 in accuracy.

4. Interatomic Potentials

In molecular dynamics simulation, the evolution of the simulated system completely depends on the forces interacting between atoms. These forces are derived from the interacting potential U, which depends on the coordinates of the particles:

$$F_i = -\nabla_i U(r_1, r_2, ..., r_N) \tag{4.1}$$

Thus, to use molecular dynamics method for given material we need to know the potential function form - the rules that are governing interaction of atoms in that material. In classical molecular dynamics, under Born-Oppenheimer approximation, the motion of electrons is assumed to be much faster than that of the atomic nuclei. And hence, the interacting potential does not include any electronic degrees of freedom explicitly, all the electronic effects are incorporated into U so it is called interatomic potential.

The interatomic potential of an interacting system can be expanded in terms of a many-body expansion as follows:

$$U(r_1, r_2, ..., r_N) = \sum_i U_1(r_i) + \sum_{i, j>i} U_2(r_i, r_j) + \sum_{i, j>i, k>j} U_3(r_i, r_j, r_k) + ... \tag{4.2}$$

where U_1 is one-body term and it is only meaningful if the atoms are placed in an external field, U_2 is two-body term or pair potential - the interaction depends only on the spacing between two atoms and is not effected by the presence of other atoms, U_3 is three-body term - which arise when the interaction of a pair of atoms is effected by the presence of the third atom. From this expansion, one usually categorized the interatomic potentials into two classes: pair potential (up to U_2 is considered) and many-body potentials (U_3 and/or higher term are included).

In order to obtain the interacting potential for a particular system, once can assume an analytical form for the potential function and then find the suitable parameters for the potential function which can reproduce a set of theoretical or experimental data through fitting methods. This step is very important and can be technically elaborate. In the past, finding an analytical form was usually done by considering the pairwise interaction between atoms (two-body potential), in which the energy of a pair depends on their relative distance. Nowadays, many-body forms are proposed in the attempt to capture as much as possible the physical and chemical characteristics of bonding. However, due to the vast differences in the electronic structures of materials at different states, it would be too ambitious to find an 'universal' interacting potential which can be used for modelling at all conditions of materials such as bulk, surface/interface, low/high temperatures, etc. The choice of the potential usually depends on the kind of materials, and/or the area of intended application. For instance, high accuracy is typically required in computational chemistry whole the computational speed is often critical in materials science. In this section, several popular forms of interatomic potential will be introduced.

4.1. Lennard-Jones Potential

The most common used two-body potential is the Lennard-Jones (LJ) potential proposed in 1924 by Lennard-Jones [9], where the pairwise interaction between two species A and B separated by distance r is defined as:

$$U_{LJ}(r) = \varepsilon_0 \left[\left(\frac{r_0}{r} \right)^m - \frac{m}{n} \left(\frac{r_0}{r} \right)^n \right] \tag{4.3}$$

Here the parameters m , n are adjustable depending on the elements under consideration and they are usually integers. For instance, in case $(m,n) = (12,6)$, the Lennard-Jones potential approximates very well the interaction in rare gases such as Ar or Kr, while $(m,n) = (8,4)$ is proved to be acceptable for describing metallic systems.

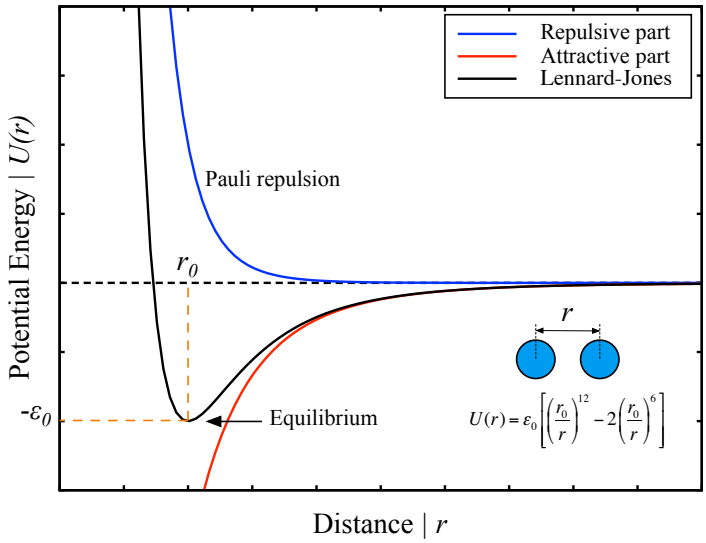

Figure 4.1. Lennard-Jones potential (m=12, n=6)

The potential has a short-range repulsive term $\sim 1/r^m$ and a long-range attractive term $\sim 1/r^n$. The origin of attractive term is due to fluctuating dipoles (van der Waals interactions or dispersion force). Whereas, the choice of repulsive term for short distances has no theoretical justification. Its origin is related to the Pauli principle: when the electronic clouds surrounding the atoms start to overlap, the energy of the system increases abruptly. As seen in equation (4.3), the Lennard-Jones potential has its minimum, $-\varepsilon_0$, at the distance r_0 . Therefore, we can consider these two parameters as the chemical bond strength and the equilibrium bond distance. These parameters can be fitted to reproduce experimental data or accurate first-principles calculations.

Alternatively, the Lennard-Jones potential for $(m,n) = (12,6)$ can be written in the following form:

$$U_{LJ}(r) = 4\varepsilon_0 \left[\left(\frac{\sigma}{r} \right)^{12} - \left(\frac{\sigma}{r} \right)^6 \right] \tag{4.4}$$

This is called standard Lennard-Jones potential. In this form, the parameter σ is the finite distance at which the inter-particle potential is zero. This distance is related to the distance r_0 as:

$$r_0 = 2^{1/6}\sigma \approx 1.122\sigma \tag{4.5}$$

If electrostatic charges are present, we need to add the appropriate Coulomb potentials to the equations (4.3) and (4.4):

$$U_{CL}(r) = \frac{Q_1 Q_2}{4\pi\varepsilon r} \tag{4.6}$$

where Q_1, Q_2 are the charges of the two particles and ε is the permittivity of vacuum. Note that the Coulomb potential has a much slower decay ($\sim 1/r$) at large distances compared to the Lennard-Jones case ($\sim 1/r^6$). The correct handling of long-range interactions is an essential aspect of polyelectrolyte or molten salts simulations.

Due to the computational simplicity, the Lennard-Jones potential is used extensively in computer simulations even though more accurate potentials exist.

4.2. Buckingham Potential

The Buckingham potential was proposed by Buckingham [10] for describing the short-range interaction of two atoms that are not directly bonded as a function of interatomic distance r as follows:

$$U_{BK}(r) = Ae^{-Br} - \frac{C}{r^6} \tag{4.7}$$

where A, B, and C are constants. This potential consists of two parts: repulsive and attractive terms. The attractive part is similar to that of the Lennard-Jones potential for $(m,n) = (12,6)$, while the repulsive part is an exponential function.

An extension of the Buckingham potential for application to ionic systems is the Coulomb-Buckingham potential, where the long-range interaction part is typically handled by the Coulomb potentials:

$$U_{CB}(r) = Ae^{-Br} - \frac{C}{r^6} + \frac{Q_1 Q_2}{4\pi\varepsilon r} \tag{4.8}$$

with the parameters Q_1, Q_2, ε defined previously.

It should be noted that, as $r \to 0$, the repulsive term converges to a constant, while the attractive term diverges. Hence, the Buckingham potential "turns over" as r becomes small. This may be problematic when dealing with a structure with very short interatomic distances, as the nuclei that cross the turn-over distance will become strongly and unphysically bound to one another at a distance close to zero.

4.3. EAM Potential

The embedded atom method (EAM) was suggested by Daw and Baskes [11,12] as a way to overcome some problems of two-body potentials such as reproducing the Cauchy discrepancy between the elastic constants and the correct vacancy formation energy. This potential is basically based on the concept of local density which allows one to account for the dependence of the strength of individual bonds on the local environment. In embedded atom method, the potential energy is written as:

$$E = \frac{1}{2}\sum_{i \neq j} \varphi_{ij}(r_{ij}) + \sum_i F_i(\overline{\rho}_i) \tag{4.9}$$

Here, the first term $\varphi_{ij}(r_{ij})$ is the core-to-core repulsive energy between atom i and j separated by distance r_{ij} and is expressed in a pairwise functional form. The second term $F_i(\overline{\rho}_i)$ corresponds to the energy required to embed atom i into the electron density $\overline{\rho}_i$. Thus, the many-body effect is incorporated into the embedding energy term via the electron density. This electron density can be defined as a linear superposition of contribution of individual neighbor atoms:

$$\overline{\rho}_i = \sum_{i \neq j} \rho_j(r_{ij}) \tag{4.10}$$

where $\rho_j(r_{ij})$ is the contribution to the electron density $\bar{\rho}_i$ at atom i due to atom j at the distance r_{ij}. The embedding energies and pair interactions are usually determined empirically by fitting to various bulk properties. For the metals, the functions are determined by fitting to bulk lattice constant, sublimation energy, elastic constants and vacancy formation energy.

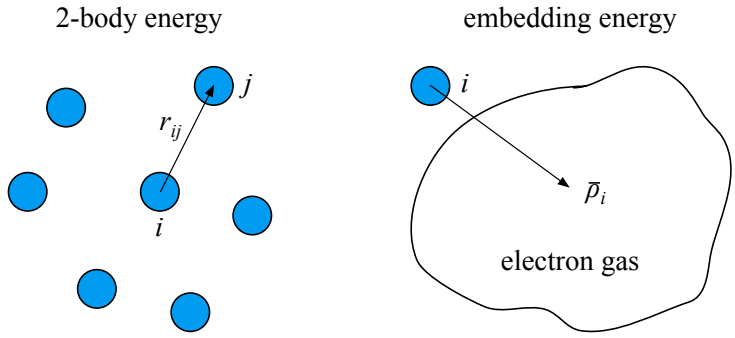

Figure 4.2. Pair interaction and embedding energy

As we can see, both terms in equation (4.9) have a central-force characteristic, i.e., depend on the interatomic distances r_{ij} only and not on bond angles. In the embedded atom method, the potential energy is described explicitly by the two-body interaction, therefore, it is sometimes called as environment-dependent pair potential. The simulations using the embedded atom method potentials are nearly as simple and efficient as with pair potentials (about 2 times slower than pair potentials). However, the lack of explicit three-body terms makes it challenging to design potentials for metals where covalent effects are important.

An example for embedded atom method potentials is described below for iron-hydrogen binary system. For pure iron potentials, the pairwise and embedding energy functions were fitted to the three elastic constants of the BCC iron, in addition to the standard set of properties: (1) First-principles forces in a liquid-like structure; (2) Lattice parameters of BCC and FCC iron at zero Kelvin; (3) Unrelaxed BCC vacancy formation energy; and (4) The energy difference between BCC and FCC crystals. The functional forms of the potentials are as follows [13]:

$$\rho(r) = \sum_{k=1}^{n^\rho} a_k^\rho (r_k^\rho - r)^3 \theta(r_k^\rho - r) \tag{4.11}$$

$$F(\bar{\rho}) = -\bar{\rho}^{1/2} + a^F \bar{\rho}^2 \tag{4.12}$$

$$\varphi(r) = \begin{cases} \dfrac{Z^2 q_e^2}{r} \xi\left(\dfrac{r}{r_s}\right), & r \le r_1 \\ e^{B_0 + B_1 r + B_2 r^2 + B_3 r^3}, & r_1 < r \le r_2 \\ \displaystyle\sum_{k=1}^{n^\varphi} a_k^\varphi (r_k^\varphi - r)^3 \theta(r_k^\varphi - r), & r > r_2 \end{cases} \tag{4.13}$$

$$r_s = 0.88534 \frac{r_B}{2^{1/2} Z^{1/3}} \tag{4.14}$$

$$\xi(x) = 0.1818 e^{-3.2x} + 0.5099 e^{-0.9423x} + \ldots + 0.2802 e^{-0.4029x} + 0.02817 e^{-0.2016x} \tag{4.15}$$

where Z is the atomic number of iron, q_e is the charge on an electron, θ is the Heaviside step function, r_B is the Bohr radius, r_s is the screening length, and $\xi(x)$ is the screening function. In the pair potential form (4.13), a Biersack and Ziegler's universal screened-Coulomb function in the range $r \le r_1$ and a cubic spline function in the range $r > r_2$ were used. In the range $r_1 < r \le r_2$, an exponential function was chosen to interpolate between the universal screening function and the spline function, and ensure the continuity of the pair potential and its first and second derivatives at r_1 and r_2.

For hydrogen, the potentials is taken from the nickel-hydrogen system. The potential parameters were determined by fitting to the solution energy and migration energy of hydrogen in bulk nickel, the chemisorption energy in the fourfold hollow site of the (100) surface and the threefold hollow site on the (111) surface, the equilibrium height of the hydrogen above the (100) surface, and by requiring that the observed adsorption sites for hydrogen on the (100), (110) and (111) have the lowest classical energies [14-16]:

$$\rho(r) = a_1^{\rho} r^6 \left[e^{-a_2^{\rho} r} + 2^9 e^{-2a_2^{\rho} r} \right] f_c(r) \tag{4.16}$$

$$F(\overline{\rho}) = a_1^F \overline{\rho}^2 + a_2^F \overline{\rho} + \frac{a_3^F \overline{\rho}^{5/3}}{a_4^F + \overline{\rho}} \tag{4.17}$$

$$\varphi(r) = a_1^{\varphi} \left[e^{-2a_2^{\varphi}(r - a_3^{\varphi})} - 2e^{-a_2^{\varphi}(r - a_3^{\varphi})} \right] f_c(r) + a_4^{\varphi} \rho(r) \tag{4.18}$$

where $f_c(r)$ is a cutoff function that forces the pair potential smoothly to zero beyond a certain distance r_c:

$$f_c(r) = e^{1/(r - r_c)} \tag{4.19}$$

The embedding energy was completely determined by the interaction of hydrogen with nickel and not based on the molecular properties of hydrogen. The pair interaction in equation (4.18) was determined by fitting the equilibrium bond energy and bond length of a hydrogen molecule. However, it was not possible to fit the molecular properties of hydrogen exactly, although the bond energy was correctly determined; the bond length was approximately 0.1 Å larger than the experimentally determined value of 0.74 Å. Therefore, the pair energy was re-defined so that the potential energy surface for an isolated hydrogen molecule was described correctly. The modified functional form is as follows:

$$\varphi^*(r) = s(r)\left[E_{mol}(r) - 2F(\rho) \right] + \left[1 - s(r) \right] \varphi(r) \tag{4.20}$$

where $s(R)$ is a switching function which switches the hydrogen pair interaction rapidly from that defined by the embedded atom method in equation (4.18) at distances greater than 0.9 Å to that defined by molecular hydrogen below this value:

$$s(r) = 0.5\left\{ 1 - \tanh\left[15(r - 0.9) \right] \right\} \tag{4.21}$$

In equation (4.20), $E_{mol}(r)$ is the energy of the hydrogen molecule as a function of separation r. It was chosen as:

$$E_{mol}(r) = -2E_b(1 + a^{\dagger})e^{-a^{\dagger}} \tag{4.22}$$

$$a^{\dagger} = \frac{(r - r_0)}{\lambda r_0} \tag{4.23}$$

where $E_b = 2.37$ eV/atom is the molecular bond strength, $r_0 = 0.74$ Å is the equilibrium separation distance, $a^\dagger$ is the measure of the derivative from the equilibrium separation distance, and $\lambda = 0.4899$.

Finally, for the cross-pair potential between iron and hydrogen, it can be chosen to be in the form of a Morse function:

$$\varphi(r) = \left[D_e \left(1 - e^{-\alpha(r-r_e)}\right)^2 - D_e \right] h_c\left(\frac{r - r_c}{h}\right) \tag{4.24}$$

where r_e is the equilibrium bond distance, D_e is the well depth, α controls the 'width' of the potential (the smaller α is, the larger the well), and h_c is the cutoff function which smoothly turns the pair potential to zero beyond a cutoff distance r_c:

$$h_c(x) = \begin{cases} \dfrac{x^4}{1+x^4}, & x \le 0 \\ 0, & x > 0 \end{cases} \tag{4.25}$$

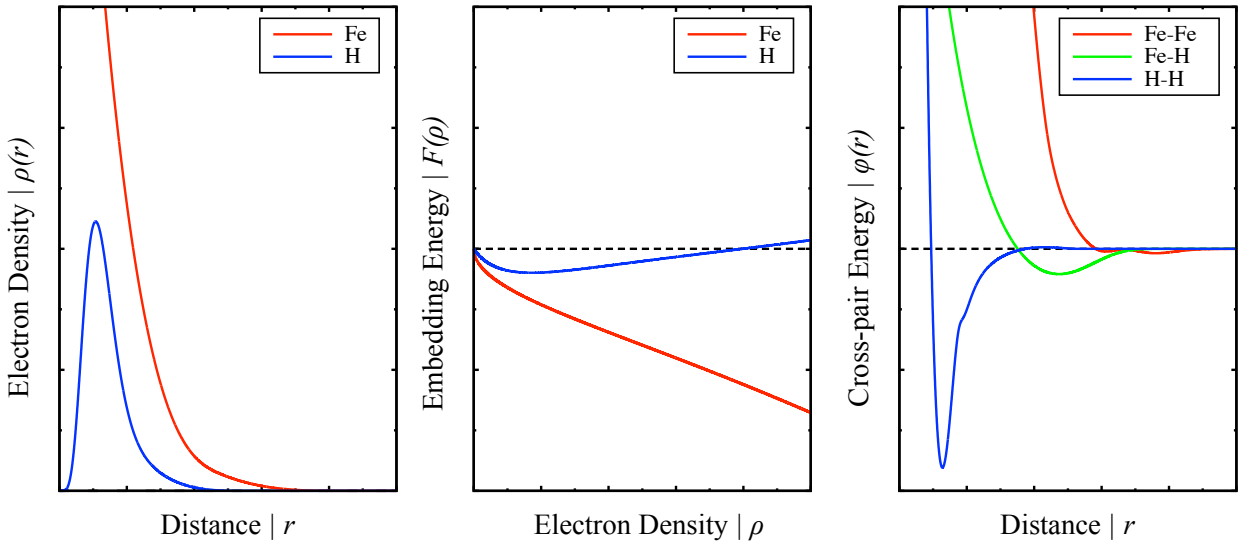

Figure 4.3. Embedded Atom Method potential for Fe-H binary system

4.4. MEAM Potential

The modified embedded atom method (MEAM) was suggested by Baskes [17,18]. The potential energy in the modified embedded atom method formalism is similar to that of embedded atom method, which consists of two parts, a pair potential term representing the electrostatic core-to-core repulsion, and a cohesive term representing the energy the atom gets when it is embedded in the electron density.

$$E = \frac{1}{2}\sum_{i \neq j} \varphi_{ij}(r_{ij}) + \sum_i F_i(\overline{\rho}_i) \tag{4.26}$$

The difference between the modified embedded atom method and the embedded atom method is on the definition of the electron density $\overline{\rho}_i$. In the embedded atom method, the host electron density is given by a linear superposition of spherically averaged atomic densities. For the materials with highly-directional bonds, the embedded atom method formalism shows the inconsistency in elastic constants as compared to the experimental data. To overcome this, in the modified embedded atom method, the host electron density is modified to include angular-dependent contributions (a.k.a depends explicitly on the three-body components).

$$\overline{\rho}_i = \sum_{i \neq j} \rho_j(r_{ij}) + \sum_{j,k \neq i} \left[a_j^1 a_k^1 \cos\theta_{jik} - a_j^2 a_k^2 (1 - 3\cos^2\theta_{jik}) \right] \rho_j(r_{ij}) \rho_k(r_{ik}) \quad (4.27)$$

where θ_{jik} is the angle centred on atom i between atoms j, i, k; and $a_i^{1,2}$ are constants depending on the atom type i. This improvement, however, slows down the speed of simulations (3 to 5 times) as compared to the embedded atom method model.

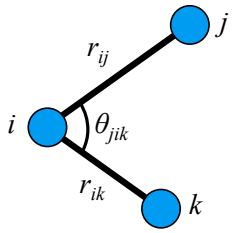

Figure 4.4. Three-body component

A simplified version of modified embedded atom method that employs cubic splines instead of pre-determined analytic functions was developed later by Lenosky [19]:

$$\overline{\rho}_i = \sum_{i \neq j} \rho_j(r_{ij}) + \frac{1}{2} \sum_{j,k \neq i} f_{ij}(r_{ij}) f_{ik}(r_{ik}) g_i(\cos\theta_{jik}) \quad (4.28)$$

Here, $f_{ij}(r_{ij})$ is the radial function between atom i and j, while $g_i(\cos\theta_{jik})$ is the angular function of the three-body angle between atoms j, i, k with central atom i. The two functions are also presented as cubic splines. As we can see, when $f_{ij}(r_{ij}) = 0$ and $g_i(\cos\theta_{jik}) = 0$, the electron density will fall back to the embedded atom method form. The explicit 3-body term is included in the local "electron" density through the dependence on $f_{ij}(r_{ij})$ and $g_i(\cos\theta_{jik})$.

4.5. ADP Potential

The angular-dependent potential (ADP) was suggested by Mishin [20] based on the embedded atom method potential model for investigation of the Fe-Ni system. It is a generalization of the embedded atom method potential for simulating the covalent bonding component which is lack in the embedded atom method. In the angular-dependent potential formalism, the potential energy is written as follows:

$$E = \frac{1}{2} \sum_{i \neq j} \varphi_{ij}(r_{ij}) + \sum_i F_i(\overline{\rho}_i) + \frac{1}{2} \sum_{i,\alpha} (\mu_i^\alpha)^2 + \frac{1}{2} \sum_{i,\alpha,\beta} (\lambda_i^{\alpha\beta})^2 - \frac{1}{6} \sum_i \nu_i^2 \quad (4.29)$$

Here indices i and j enumerate atoms and the superscripts $\alpha, \beta = 1,2,3$ refer to the Cartesian directions. The first term in equation (4.29) represents pair interactions between atoms, the second term is the embedding energy of atom i in the host electron density $\overline{\rho}_i$ induced at site i by all other atoms of the system. Thus, the first two terms constitute the regular embedded atom method potential form. Notice that they have a central-force character. The additional three terms introduce non-central components of bonding via the vectors:

$$\mu_i^\alpha = \sum_{j \neq i} u_{ij}(r_{ij}) r_{ij}^\alpha \quad (4.30)$$

and tensors:

$$\lambda_i^{\alpha\beta} = \sum_{j \neq i} w_{ij}(r_{ij}) r_{ij}^\alpha r_{ij}^\beta \quad (4.31)$$

The quantities ν_i are traces of the λ tensor:

$$\nu_i = \sum_{\alpha} \lambda_i^{\alpha\alpha} \tag{4.32}$$

In equations (4.30) and (4.31), μ_{ij} and w_{ij} are two additional pairwise functions, which depend on the interatomic distance r_{ij}. The quantities μ_i^{α} and $\lambda_i^{\alpha\beta}$ can be thought of as measures of the dipole and quadrupole distortions, respectively, of the local environment of atom i. These terms are especially important in non-centrosymmetric structures such as diamond and HCP and in materials with a negative Cauchy pressure.

In modified embedded atom method, angular-dependent interactions also introduced through dipole, quadrupole and higher-order multipoles. However, the difference between modified embedded atom method and angular-dependent potential is that, in modified embedded atom method, they constitute a part of the tensor electron density whereas in the angular-dependent potential they contribute directly to the total energy.

4.6. Stillinger-Weber Potential

The Stillinger-Weber potential introduced by Stillinger and Weber [21] is one of the first attempts to model a semiconductor with a classical model. This model is based on a two-body term and a three-body term of the potential energy expansion.

$$E = \sum_{i,j>i} U_2(r_i, r_j) + \sum_{i,j>i,k>j} U_3(r_i, r_j, r_k) \tag{4.33}$$

The single-particle potential $U_1(r_i)$ normally describes the wall and external forces to which the system is subject therefore it is absent from the equation (4.33).

The two-body potential was selected as follows:

$$U_2(r) = \begin{cases} \varepsilon A\left(\frac{B}{r^p} - \frac{1}{r^q}\right) e^{(r-a)^{-1}}, & r \le a \\ 0, & r > a \end{cases} \tag{4.34}$$

where A, B, p, q and a are positive parameters. It is very much like a Lennard-Jones potential, only with different exponents and a smooth cutoff distance that smoothly terminates the potential at an interatomic distance $r = a$ without discontinuities in any r derivatives.

The directional bonding is introduced through an explicit three-body term which is the sum of functions of each of the three angles of a triplet, (i, j, k):

$$U_3(r_i, r_j, r_k) = h_{jik} + h_{ijk} + h_{ikj} \tag{4.35}$$

where θ_{jik} is the angle between the ij and ik bonds subtended at vertex i, etc. Provided that both r_{ij} and r_{ik} are less than the previously introduced cutoff a, the function $h_{jik} = h(r_{ij}, r_{ik}, \theta_{jik})$ has the following form ($\lambda, \gamma > 0$):

$$h(r_{ij}, r_{ik}, \theta_{jik}) = \varepsilon\lambda e^{\gamma(r_{ij}-a)^{-1} + \gamma(r_{ik}-a)^{-1}} \left(\cos\theta_{jik} + \frac{1}{3}\right)^2 \tag{4.36}$$

otherwise it vanishes identically.

The original intention of this potential is to describe the bonding in silicon. That means the potential should predict the diamond-like tetrahedral structure with bond angle of $109.47°$ as the most stable atomic configuration. The equation (4.36) satisfies this requirement as we can see that it favors those configurations where $\cos\theta_{jik} = -1/3$ (a.k.a $\theta_{jik} = 109.47°$).

This potential gives a fairly realistic description of crystalline silicon. However, its built-in tetrahedral bias creates problems in other situations: it cannot predict the right energies of the non-tetrahedral polytypes found under pressure; the coordination of the liquid is too low; surface structures are not correct. These are called transferability problems: it is difficult to use the potential under conditions different from those it was designed for.

4.7. Tersoff Potential

The Tersoff potential was developed by Tersoff [22,23] based on the concept of bond order: the strength of a bond between two atoms is not constant, but depends on the local environment, i.e. an atom with many neighbors forms weaker bonds than an atom with few neighbors. It was calibrated firstly for silicon and later for carbon. In this formalism, the potential energy is written as:

$$E = \frac{1}{2}\sum_{i \neq j} f_c(r_{ij})\left[f_R(r_{ij}) + b_{ij} f_A(r_{ij})\right] \tag{4.37}$$

where $f_R(r_{ij})$, $f_A(r_{ij})$ are repulsive and attractive pair potential, respectively:

$$f_R(r_{ij}) = Ae^{-\lambda_1 r_{ij}} \tag{4.38}$$

$$f_A(r_{ij}) = -Be^{-\lambda_2 r_{ij}} \tag{4.39}$$

and $f_C(r_{ij})$ is the cutoff function to restrict the interaction range:

$$f_c(r_{ij}) = \begin{cases} 1, & r_{ij} \le R \\ \frac{1}{2} + \frac{1}{2}\cos\left(\frac{\pi(r_{ij} - R)}{S - R}\right), & R < r_{ij} \le S \\ 0, & r_{ij} > S \end{cases} \tag{4.40}$$

It has to be noticed that the parameters R and S in $f_C(r_{ij})$ are not systematically optimized but are chosen so as to include the first-neighbor shell only for several selected high-symmetry bulk structure (for carbon $R = 1.8$ Å). S is called the cutoff distance of the interaction.

The main feature of this potential is the presence of the b_{ij} term, a bond-order term which includes three-body interactions and angularity:

$$b_{ij} = (1 + \beta^n \zeta_{ij}^n)^{-\frac{1}{2n}} \tag{4.41}$$

where

$$\zeta_{ij} = \sum_{k \neq i,j} f_c(r_{ik}) g(\theta_{jik}) e^{\lambda_3^m (r_{ij} - r_{ik})^m} \tag{4.42}$$

defines the effective coordination number of atom i, i.e. the number of nearest neighbours, taken into account the relative distance of two neighbors $r_{ij} - r_{ik}$ and the bond-angle θ_{jik}.

And the angular function, depends on the angle θ_{jik} between bonds formed by pairs of atoms (i,j) and (i,k) is defined as:

$$g(\theta_{jik}) = \gamma_{ijk}\left(1+\frac{c^2}{d^2}-\frac{c^2}{d^2+(h-\cos\theta_{jik})^2}\right) \tag{4.43}$$

Here, the parameter h is the minimum of the angular function, the parameter d determines how sharp the dependence on angle is, and c expresses the strength of the angular effect.

In Tersoff formalism, the $A,B,\lambda_1,\lambda_2,n,\beta$ parameters are only used for two-body interactions. The m,γ,λ_3,c,d,h parameters are only used for three-body interactions. The R and S parameters are used for both two-body and three-body interactions.

Practically, the original form of the Tersoff potential does not always reproduce correctly the elastic constants and melting point of materials (for instance, diamond silicon). To improve the description of these physical properties, the modified cutoff function proposed by Murty [24] is employed in the new form:

$$f_c(r_{ij}) = \begin{cases} 1, & r_{ij} \le R \\ \frac{1}{2}+\frac{9}{16}\cos\left(\frac{\pi(r_{ij}-R)}{S-R}\right)-\frac{1}{16}\cos\left(\frac{3\pi(r_{ij}-R)}{S-R}\right), & R < r_{ij} \le S \\ 0, & r_{ij} > S \end{cases} \tag{4.44}$$

In line with that, in order to increase the flexibility of the potential, the modification is made for angular function by Kumagai [25] as follows:

$$g(\theta) = c_1 + g_0(\theta)g_a(\theta) \tag{4.45}$$

$$g_0(\theta) = \frac{c_2(h-\cos\theta)^2}{c_3+(h-\cos\theta)^2} \tag{4.46}$$

$$g_a(\theta) = 1 + c_4 e^{-c_5(h-\cos\theta)^2} \tag{4.47}$$

Another improvement over the original form is for simulations of high-energy events, which occur, for example, in sputtering simulations. In this case, the energy expression for r close to zero (the repulsive part) needs to be modified based on the Coulomb potential and the Ziegler-Biersack-Littmark (ZBL) universal screening function. The modified form of the potential is written as:

$$\widetilde{U}_{ij} = f(r_{ij})U_{ij} + \left(1-f(r_{ij}\right)U_{ij}^{ZBL} \tag{4.48}$$

where U_{ij} is the original Tersoff potential and U_{ij}^{ZBL} is the universal repulsive ZBL potential:

$$U_{ij} = f_c(r_{ij})\left[f_R(r_{ij}) + b_{ij}f_A(r_{ij})\right] \tag{4.49}$$

$$U_{ij}^{ZBL} = \frac{1}{4\pi\epsilon_0}\frac{Z_1Z_2e^2}{r_{ij}}\phi\left(\frac{r_{ij}}{a}\right) \tag{4.50}$$

$$a = \frac{0.8854 a_0}{Z_i^{0.23} + Z_j^{0.23}} \tag{4.51}$$

$$\begin{aligned} \phi(x) = 0.1818 e^{-3.2x} + 0.5099 e^{-0.9423x} + \ldots \\ + 0.2802 e^{-0.4029x} + 0.02817 e^{-0.2016x} \end{aligned} \tag{4.52}$$

and $f(r_{ij})$ is the Fermi-like function used to smoothly join the ZBL part with the Tersoff part, such that the equilibrium properties of the original potential are maintained.

$$f(r) = \frac{1}{1 + \mathrm{e}^{-b_f(r - r_f)}} \tag{4.53}$$

Here, the parameter b_f controls the 'sharpness' of the transition and the parameter r_f is the cutoff distance for ZBL potential.

4.8. Machine Learning Potential

(Under construction)

5. Temperature and Pressure Controls

In molecular dynamics simulations, the microscopic quantities such as positions, velocities of particles are produced. However, one often wants to explore the macroscopic properties of a simulated system such as temperature, pressure, etc. through such microscopic properties. To do so, statistical mechanics is used to provide the mathematical expressions that connect the two levels, with the goal is to understand and predict macroscopic phenomena from the motion and distribution of individual particles making up the system. In macroscopic level, the macroscopic state (or thermodynamic state) of a system is defined by parameters set such as the temperature T, the pressure P, and the number of particles N. While, in microscopic level, the microscopic state of a system is defined by particles' positions q, and momenta p. These can be considered as coordinates in a multi-dimensional space called phase space. A single point in phase space describes a microscopic state of the system. A collection of points in phase space satisfying the conditions of a particular thermodynamic state is called an ensemble. In statistical mechanics, common ensembles are: the microcanonical ensemble, the canonical ensemble and the grand canonical ensemble. Here, canonical simply means standard or acceptable and the canonical ensemble therefore holds the central place in statistical mechanics.

NVE Ensemble (microcanonical). The constant-volume, constant-energy ensemble, also known as the microcanonical ensemble. The ensemble is obtained by solving Newton's equations of motion without any temperature and pressure control. It corresponds to an adiabatic process with no heat exchange. In this ensemble, the molecular dynamics trajectory may be seen as an exchange of potential and kinetic energies, with total energy being conserved. However, because of rounding and truncation errors during the integration process, there is always a slight drift in energy.

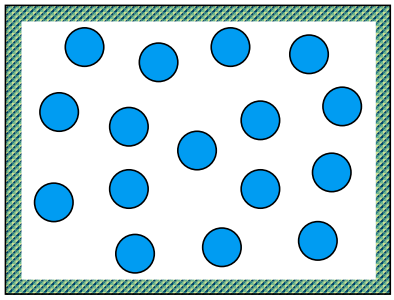

Figure 5.1. Microcanonical Ensemble

NVT Ensemble (canonical). The constant-volume, constant-temperature ensemble, also referred to as the canonical ensemble. The ensemble is obtained by solving Newton's equations of motion with the temperature controlled through a 'heat bath'. The volume is kept constant throughout the run.

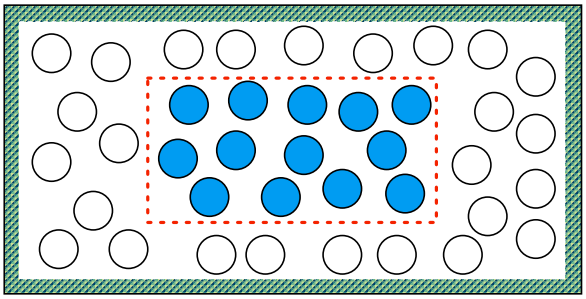

Figure 5.2. Canonical Ensemble

NPT Ensemble (isothermal-isobaric). The constant-pressure, constant-temperature ensemble allows control over both the temperature and pressure. In addition to a 'heat bath', a 'pressure bath' is needed. It corresponds most closely to experimental conditions where the temperature and pressure are typically kept fixed.

μVT Ensemble (grand canonical). The grand canonical ensemble is a generalization of the canonical ensemble where the restriction to a definite number of particles in the system is removed. It describes systems in contact with a 'heat bath' at temperature T and a particle reservoir that maintains the chemical potential μ. The system not only exchanges heat with the 'heat bath' but also exchanges particles with the reservoir. The volume is kept constant throughout the run, but the number of particles and energy fluctuate at thermal equilibrium.

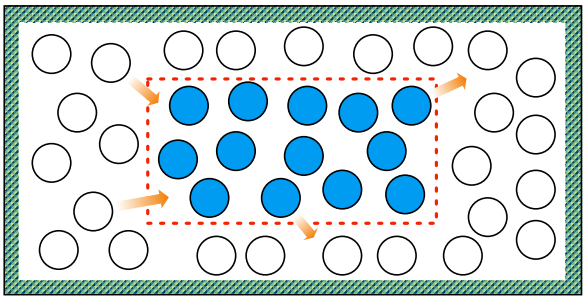

Figure 5.3. Grand canonical ensemble

Among four ensembles mentioned above, the first three ensembles are often used in molecular dynamics simulations. Especially, to compare the simulation results with experiments, the NVT and NPT ensembles are usually used where energy exchange between the system of interest and the environment occurs. For carrying out molecular dynamics simulation under these ensembles, we need to control the temperature and pressure. The way of controlling the temperature is called 'thermostat' while the method of controlling pressure is referred to as 'barostat'. In this section, several techniques for controlling temperature and pressure are shown.

5.1. Temperature

For a system at given temperature T and in equilibrium state, the equipartition theorem states that the average kinetic energy per particle (with 3 degrees of freedom) is related to T as:

$$\left\langle \frac{1}{2}mv^2 \right\rangle = \frac{3}{2}k_B T \tag{5.1}$$

where k_B is the Boltzmann constant.

If the condition of zero or fixed total momentum is imposed in the simulation, the N-particle system will have $N_f = 3N - 3$ number of degrees of freedom. Hence, the temperature in a molecular dynamics simulation can be calculated from the total kinetic energy by:

$$T = \frac{1}{3(N-1)k_B}\sum_i m_i v_i^2 \tag{5.2}$$

Basically, this temperature is different from the true (thermodynamic) temperature because the kinetic energy fluctuates in time. For such reason, it is called 'instantaneous temperature'. In constant temperature molecular dynamics simulations (NVT, NPT,…), the total energy is not conserved because the system is in contact with heat bath. Under the thermal equilibrium condition (with the heat bath), the momentum is distributed according to the Maxwell-Boltzmann distribution:

$$P(p) = \left(\frac{\beta}{2\pi m}\right)^{3/2} e^{-\frac{\beta}{2m}p^2} \tag{5.3}$$

where $\beta = 1/k_B T$. The fluctuations of the kinetic energy is related to the second and fourth moments of momentum distribution. The relative variance of the kinetic energy is given by:

$$\frac{\sigma_{p^2}^2}{\langle p^2\rangle^2} = \frac{\langle p^4\rangle - \langle p^2\rangle^2}{\langle p^2\rangle^2} = \frac{2}{3} \tag{5.4}$$

Here, we used:

$$\langle p^2\rangle = \int p^2 P(p)(4\pi p^2)dp = 4\pi\left(\frac{\beta}{2\pi m}\right)^{3/2}\int p^4 e^{-\frac{\beta}{2m}p^2}dp = \frac{3m}{\beta} \tag{5.5}$$

and:

$$\langle p^4\rangle = \int p^4 P(p)(4\pi p^2)dp = 4\pi\left(\frac{\beta}{2\pi m}\right)^{3/2}\int p^6 e^{-\frac{\beta}{2m}p^2}dp = \frac{15m^2}{\beta^2} \tag{5.6}$$

Since, in simulation, the kinetic energy per particle is used to measure the instantaneous temperature as showed in equation (5.1), hence this temperature fluctuates. The relative variance of the temperature can be written as:

$$\frac{\sigma_T^2}{\langle T\rangle^2} = \frac{\langle T^2\rangle - \langle T\rangle^2}{\langle T\rangle^2} = \frac{1}{N}\frac{\langle p^4\rangle - \langle p^2\rangle^2}{\langle p^2\rangle^2} = \frac{2}{3N} \tag{5.7}$$

Typically, the relative fluctuations in the temperature will be of order $1/\sqrt{N_f}$. To get an accurate estimation of the temperature, we should average over many fluctuations. Usually, for a simulation system with a sufficiently large number of particles, the instantaneous temperature will converge to a roughly constant value.

5.2. Velocity Rescaling

One of the simplest ways to control the temperature in molecular dynamics simulations is the momentum-rescaling method or the velocity rescaling. When we want the system temperature to reach a desired temperature T_0 from the current temperature T , we rescale all the velocities by a factor λ as follows:

$$v_i^0(t) = \lambda v_i(t) \tag{5.8}$$

while the positions of particles are unchanged. The new velocities $v_i^0(t)$ will be used in the next step in the integration scheme of the equations of motion.

By using equation (5.2), the rescaling factor λ can be written as:

$$\lambda = \sqrt{\frac{T_0}{T}} \tag{5.9}$$

In this controlling method, the kinetic energy will remain constant in time, with zero fluctuations. The rescaling procedure totally suppresses any possible fluctuation in the temperature of the system, and hence, it does not provide the correct statistical ensemble as expected from equation (5.7).

5.3. Berendsen Thermostat

The Berendsen thermostat, or also known as the weak-coupling thermostat, was suggested by Berendsen [26]. It is also based on some velocity rescaling, however the rescaling is 'softer' as compared to the original velocity rescaling method. That means, this method assigns a time scale for the updating of the velocities, rather than assuming they are completely rescaled to the target temperature at each time-step.

In this method, a frictional term is added to the equations of motion to describe the weak-coupling to the heat bath:

$$\frac{d^2 r_i}{dt^2} = -\frac{1}{m_i}\frac{\partial U}{\partial r_i} + \gamma\left(\frac{T_0}{T} - 1\right)\frac{dr_i}{dt} \tag{5.10}$$

which then drives the system exponentially toward the desired temperature T_0.

Practically, the rescaling factor of the velocities is given by:

$$\lambda = \sqrt{1 + \frac{\Delta t}{\tau_T}\left(\frac{T_0}{T} - 1\right)} \tag{5.11}$$

Here, Δt is the time-step of simulation and τ_T is a characteristic time to be fixed at the beginning of the simulation. As we can see, when $\tau_T = \Delta t$, equation (5.11) will fall back to equation (5.9) of the original velocity rescaling method. If $\tau_T \to \infty$, no rescaling is done as $\lambda = 1$.

Similar to the velocity rescaling method, the Berendsen thermostat also does not reproduce trajectories consistent with the correct canonical ensemble.

5.4. Andersen Thermostat

The Andersen thermostat introduced by Andersen [27] is another straightforward way to control the temperature of a molecular dynamics simulation. In this method, some particles undergo random 'collisions' with external degrees of freedom. This can be thought as coupling all the atoms in the system to an imaginary external heat bath that imposes the desired temperature. By doing so, the velocities of these particles are changed randomly based on a Maxwell-Boltzmann distribution at the desired temperature T_0:

$$P(v) = \left(\frac{m}{2\pi k_B T_0}\right)^{1/2} e^{-\frac{m}{2k_B T_0}v^2} \tag{5.12}$$

Practically, the stochastic collision procedure in simulation is not necessary to be done every time-step but rather it is established by adopting a collision frequency ν (or collision time $\tau = 1/\nu$) which determine the strength of the coupling of the system to the heat bath. In the limit of an infinitely long trajectory averaged over many heat bath collisions, the Andersen thermostat with non-zero collision frequency ν rigorously

generates the correct canonical ensemble probabilities. However, due to the perturbation of the particles' velocities, i.e., generating discontinuous velocity trajectories, the method messes up the natural dynamics of the system. Therefore, this might not be a good method for study atomistic processes in detail.

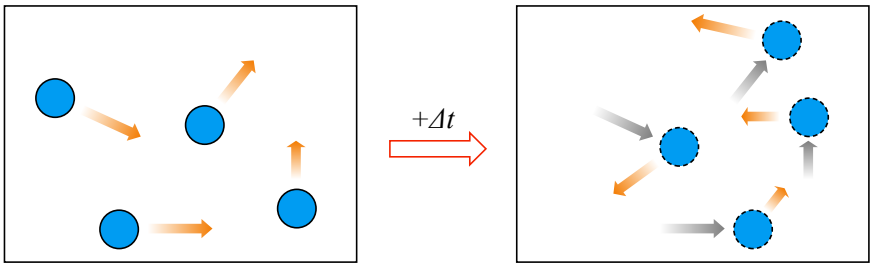

Figure 5.4. The Andersen thermostat

5.5. Nosé-Hoover Thermostat

Aside from the stochastic approach introduced previously by Andersen for controlling temperature, there is a sophisticated method with a deterministic nature, called Nosé-Hoover thermostat. Initially, the approach by Nosé [28] is to extend the Lagrangian of the system in a way that it contains additional, artificial coordinates and velocities. Due to this, the method belongs to the class of method called the extended-system method. Actually, the extended-Lagrangian approach was first introduced by Andersen [27] in the context of constant-pressure molecular dynamics simulations. A simpler formulation of the equations of motion was later proposed simultaneously by Nosé [29] and Hoover [30,31] so this temperature-controlling method is generally referred to as the Nosé-Hoover thermostat.

In this scheme, the physical system is brought in contact with an imaginary external heat bath and exchanges energy with the bath. Nosé's approach was to add two additional degrees of freedom to the system which are the 'position' s and the conjugate 'momentum' p_s of the imaginary heat reservoir. In addition, an effective 'mass' Q associated with s is introduced such that $p_s = Q\dot{s}$. The dotted in s denotes its time derivative.

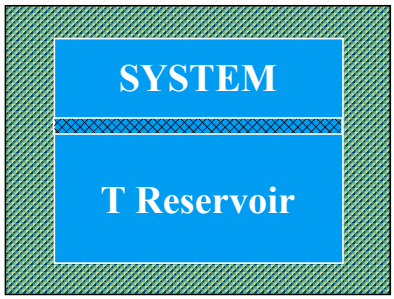

Figure 5.5. The Nosé-Hoover thermostat

The interaction between the physical system and the heat bath is expressed via the scaling of the velocities of the particles:

$$\tilde{v}_i = s\frac{dr_i}{dt} \tag{5.13}$$

Here, $\tilde{v}_i$ is considered as the real velocity of particle i. We can interpret this as an exchange of energy between the physical system and the heat reservoir. The variable s plays a role as the temperature controller and its value depends on the difference between the system temperature T and the desired temperature T_0.

The Lagrangian of the extended system of the N particles and variable s is:

$$L_{Nosé} = \frac{1}{2}\sum_i m_i s^2\left(\frac{dr_i}{dt}\right)^2 - U(r) + \frac{Q}{2}\dot{s}^2 - (N_f + 1)k_B T_0 \ln s \tag{5.14}$$

The third and forth terms in equation (5.14) are the kinetic and potential energies associated with variable s. The parameter N_f is the number of degrees of freedom in the physical system and a logarithmic dependence of the potential on the variable s is essential for producing the canonical ensemble.

The equations of motion of the extended system for r_i and s which are derived from the Lagrangian are the followings:

$$\frac{d^2 r_i}{dt^2} = -\frac{1}{m_i s^2}\frac{\partial U}{\partial r_i} - \frac{2\dot{s}}{s}\frac{dr_i}{dt} \tag{5.15}$$

$$\frac{d^2 s}{dt^2} = \frac{1}{Q}\sum_i m_i s\left(\frac{dr_i}{dt}\right)^2 - \frac{(N_f+1)k_B T_0}{Qs} \tag{5.16}$$

In the extended system (virtual system), the velocities are amplified by a factor s^{-1} compared to the real system velocities, while the atomic coordinates are identical in both systems. Therefore, the timescale in the virtual system is stretched by the factor s as compared to the timescale in the real system. However, the use of an extended system with a stretched timescale is not very intuitive, and the sampling of a trajectory at uneven time intervals is rather impractical for the investigation of the dynamical properties of a system. In fact, the Nosé equations of motion can be reformulated in terms of real system variables (together with real-time sampling) so as to avoid these problems. As mentioned, the real variables $(\tilde{r},\tilde{p},\tilde{t})$ and virtual variables (r,p,t) are related as follows:

$$\tilde{s} = s, \quad \tilde{r}_i = r_i, \quad \tilde{p}_i = p_i / s, \quad \tilde{t} = \int_0^t \frac{dt}{s} \tag{5.17}$$

Hoover has shown that the equations derived by Nosé can be further simplified by introducing the thermodynamic friction coefficient $\zeta = p_s / Q = \dot{s}$. Note that ζ is in the extended system (i.e., a function of the extended system time t). The equivalent quantity in the real system is $\tilde{\zeta} = \zeta = \dot{\tilde{s}} / \tilde{s}$. The equations of motion then become:

$$\frac{d^2 \tilde{r}_i}{d\tilde{t}^2} = -\frac{1}{m_i}\frac{\partial U}{\partial \tilde{r}_i} - \tilde{\zeta}\frac{d\tilde{r}_i}{d\tilde{t}} \tag{5.18}$$

$$\frac{d\tilde{\zeta}}{d\tilde{t}} = \frac{1}{Q}\sum_i m_i\left(\frac{d\tilde{r}_i}{d\tilde{t}}\right)^2 - \frac{N_f k_B T_0}{Q} \tag{5.19}$$

The equation (5.18) is the Newton equation with an additional friction term, which depends on a time-dependent friction coefficient ζ. In equation (5.19), Hoover has replaced the (N_f+1) by N_f for convenience. The equation (5.19) can be rewritten by using equation (5.2) as follows:

$$\frac{d\tilde{\zeta}}{d\tilde{t}} = \frac{N_f k_B T}{Q}\left(1 - \frac{T_0}{T}\right) \tag{5.20}$$

Thus, the variable $\tilde{\zeta}$ no longer changes in time when the instantaneous temperature T is equal to the desired temperature T_0.

In Nosé-Hoover thermostat, the energy of the physical system fluctuates, however, the energy of the system plus the heat bath is conserved. The Hamiltonian is written as:

$$H_{Nosé-Hoover} = \frac{1}{2}\sum_i m_i \left(\frac{d\tilde{r}_i}{d\tilde{t}}\right)^2 + U(\tilde{r}) + \frac{Q}{2}\tilde{\xi}^2 + N_f k_B T_0 \ln s \tag{5.21}$$

It has been proved [28,30] that the Nosé-Hoover equations as well as the Nosé equations of motion sample a canonical ensemble of microstates in the real system. In both methods, some care must be taken in the choice of the fictitious mass Q. Too large values of Q (loose coupling) may cause a poor temperature control. Indeed, the limiting case of the Nosé-Hoover thermostat with $Q \to \infty$ and $\xi(0) = 0$ is molecular dynamics, which generates a microcanonical ensemble. On the other hand, too small values (tight coupling) may cause high-frequency temperature oscillations leading to the same effect.

5. 6. Pressure

From the general formulation of the equipartition theorem (in Hamiltonian formalism) we have:

$$\left\langle p_k \frac{\partial H}{\partial p_k} \right\rangle = \left\langle q_k \frac{\partial H}{\partial q_k} \right\rangle = k_B T \tag{5.22}$$

where q_k is generalized position and p_k is generalized momentum coordinates in the phase space.

Using the equations of Hamiltonian mechanics, these formulae may also be written as:

$$\left\langle p_k \frac{dq_k}{dt} \right\rangle = -\left\langle q_k \frac{dp_k}{dt} \right\rangle = k_B T \tag{5.23}$$

Applying (5.23) in combination with Newton's second law, in cartesian coordinates, if $r_i = (x_i, y_i, z_i)$ and $p_i = (p_i^x, p_i^y, p_i^z)$ denote the position vector and momentum of particle i, and F_i^{TOT} is the total force on that particle, then:

$$\left\langle r_i \cdot F_i^{TOT} \right\rangle = \left\langle x_i \frac{dp_i^x}{dt} \right\rangle + \left\langle y_i \frac{dp_i^y}{dt} \right\rangle + \left\langle z_i \frac{dp_i^z}{dt} \right\rangle = -3k_B T \tag{5.24}$$

Summing over a system of N particle yields:

$$\left\langle \sum_i r_i \cdot F_i^{TOT} \right\rangle = -3Nk_B T \tag{5.25}$$

Note that the total force is composed of two contributions:

$$F_i^{TOT} = F_i + F_i^{EXT} \tag{5.26}$$

The first component is the internal force arising from the interaction with all other particles in the system, while the second one is the external force coming from the interaction with the container's walls.

If all particles are enclosed in a parallelepiped container of sides L_x, L_y, L_z, volume $V = L_x L_y L_z$, we will have:

$$\left\langle \sum_i r_i \cdot F_i^{EXT} \right\rangle = L_x(-PL_yL_z) + L_y(-PL_xL_z) + L_z(-PL_xL_y) + = -3PV \tag{5.27}$$

Combining equations (5.25), (5.26) and (5.27), we obtain:

$$-3Nk_BT = \left\langle \sum_i r_i \cdot F_i \right\rangle - 3PV \tag{5.28}$$

or:

$$PV = Nk_BT + \frac{1}{3}\left\langle \sum_i r_i \cdot F_i \right\rangle \tag{5.29}$$

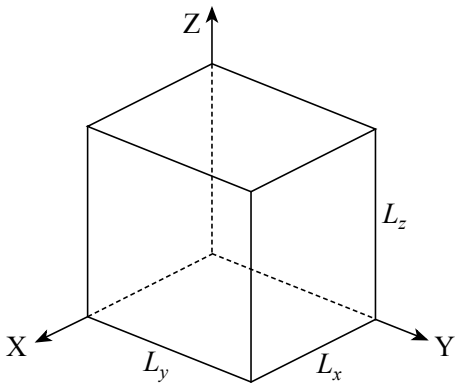

Figure 5.6. Parallelepiped container

This is known as the virial equation, and it constitutes a way to measure pressure in molecular dynamics simulations. Similar to the case of temperature, this pressure is called 'instantaneous pressure'.
In the case of pairwise interactions via a potential $\varphi(r)$, equation (5.29) becomes:

$$PV = Nk_BT - \frac{1}{3}\left\langle \sum_{i \neq j} r_{ij} \frac{\partial \varphi}{\partial r}\bigg|_{r_{ij}} \right\rangle \tag{5.30}$$

5.7. Volume Rescaling

In molecular dynamics simulations, it is often desirable to maintain the simulated systems at both constant and pressure so that it can mimic the experimental conditions. If the simulated system has a fixed boundary, the pressure can not be controlled. Thus, pressure control methods are always accompanied by the changes of the boundary conditions.

The simplest method of controlling the pressure is called volume rescaling, where the instantaneous pressure $P(t)$ is made to equal to the desired pressure P_0 by rescaling the simulation cell dimensions as well as all components of the atom positions at periodic intervals. The pressure will be decreased if the simulation cel is enlarged and vice versa.

If the simulation box has a parallelepiped shape spanned by three vectors $a = (a_1, a_2, a_3)$, $b = (b_1, b_2, b_3)$, $c = (c_1, c_2, c_3)$ we can control the pressure by:

$$h_{\alpha\beta}^0 = h_{\alpha\beta}\left[1 - \varepsilon\left(P_0 - P(t)\right)\right] \tag{5.31}$$

Here, $h = (a, b, c)$ is a 3×3 tensor with indices $(\alpha, \beta = 1,2,3)$ and ε is an appropriate scale factor.

5.8. Berendsen Barostat

The idea of Berendsen barostat is very similar to that of Berendsen thermostat [26] where the pressure is weakly coupled to a 'pressure bath' and the volume is periodically rescaled. With the Berendsen barostat the system is made to obey the equation of motion at the beginning of each time-step:

$$\frac{dP(t)}{dt} = \frac{P^0 - P(t)}{\tau_P} \tag{5.32}$$

where τ_P is the barostat relaxation time constant. The larger τ_P, the weaker the coupling with 'pressure bath'. This is very similar to the characteristic time τ_T in the Berendsen thermostat.

In the isotropic implementation, at each step the simulation cell volume is scaled by a factor η while the coordinates and the cell vectors are scaled by $\eta^{1/3}$:

$$\eta(t) = 1 - \frac{\Delta t}{\tau_P}\gamma\left(P^0 - P(t)\right) \tag{5.33}$$

Here, γ is the isothermal compressibility of the system. The exact value of γ is not critical to the algorithm as it relies on the ratio γ / τ_P. The extension of the isotropic algorithm to anisotropic cell variations is straightforward by applying the scale factors differently for different dimensions.

5.9. Andersen Barostat

The Andersen barostat suggested by Andersen [27] for pressure control is an extended system method, which involves coupling the system to a fictitious 'pressure bath' using an extended Lagrangian and introducing new degrees of freedom. This coupling mimics the action of a piston on a real system. Here the piston is not of the usual cylindrical type that expands or contracts the system along only one direction; instead, this fictitious piston can cause an isotropic expansion or contraction on the system.

In this approach, the real coordinates r of atoms are replaced by the scaled coordinates s:

$$s_i = \frac{r_i}{V^{1/3}} \tag{5.34}$$

where V is the volume of the simulation system with the cell shape that is restricted to be cubic. In addition, a new constant parameter is introduced to control the volume of the simulation cell, which is the 'mass' Q of the imaginary piston.

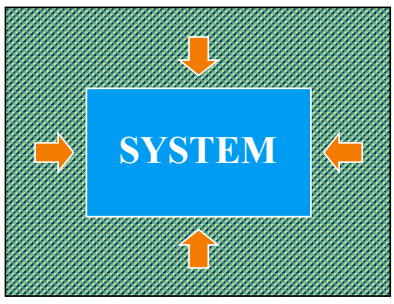

Figure 5.7. The Andersen barostat

The new Lagrangian in the scaled system is then written as follows:

$$L_{Andersen} = \frac{1}{2}\sum_i m_i V^{2/3}\left(\frac{ds_i}{dt}\right)^2 - U(V^{1/3}s) + \frac{Q}{2}\dot{V}^2 - P^0 V \tag{5.35}$$

As we can see, the volume V of the simulation cell itself is taken as a fictitious dynamical variable just like the 'position' variable s in the Nosé-Hoover thermostat (not to confuse with the scaled coordinates in this section). This variable is considered as the coordinate of the piston. The third term in equation (5.35) is the kinetic energy for the motion of V, and the forth terms represents the potential energy associated with V and an external pressure P^0 acting on the piston.

In the scaled system, the equations of motion which are derived from the Lagrangian are the followings:

$$\frac{d^2 s_i}{dt^2} = -\frac{1}{m_i V^{1/3}} \frac{\partial U}{\partial r_i} - \frac{2}{3} \frac{\dot{V}}{V} \frac{ds_i}{dt} \tag{5.36}$$

$$\frac{d^2 V}{dt^2} = \frac{1}{Q} \frac{1}{3V} \sum_i \left(m_i V^{2/3} \left(\frac{ds_i}{dt} \right)^2 - V^{1/3} s_i \frac{\partial U}{\partial r_i} \right) - \frac{P^0}{Q} \tag{5.37}$$

Now, in additional to the equation (5.34), we define a correspondence for the momenta between the scaled system and the original system:

$$p_i = \pi_i / V^{1/3} \tag{5.38}$$

where the variable π is the momentum conjugate to s:

$$\pi_i = \frac{\partial L_{Andersen}}{\partial \dot{s}_i} = m_i V^{2/3} \dot{s}_i \tag{5.39}$$

Using the correspondences in (5.34) and (5.38), the equations of motion (5.36) and (5.37) can also be written as follows:

$$\frac{dr_i}{dt} = \frac{p_i}{m_i} + \frac{1}{3} \frac{\dot{V}}{V} r_i \tag{5.40}$$

$$\frac{dp_i}{dt} = -\frac{\partial U}{\partial r_i} - \frac{1}{3} \frac{\dot{V}}{V} p_i \tag{5.41}$$

$$\frac{d^2 V}{dt^2} = \frac{1}{Q} \frac{1}{3V} \sum_i \left(\frac{p_i^2}{m_i} - r_i \frac{\partial U}{\partial r_i} \right) - \frac{P^0}{Q} \tag{5.42}$$

By using the definition of pressure from the equation (5.29), the equation (5.42) can be reformulated as:

$$\frac{d^2 V}{dt^2} = \frac{P(t) - P^0}{Q} \tag{5.43}$$

It means that the cell motion is triggered by the deviation of the internal pressure from the external pressure.

Finally, for the choice of the fictitious 'mass' of the piston, Q, a low 'mass' will result in rapid box size oscillations, which are not damped very efficiently by the motions of the molecules. While a large 'mass' will give rise to a slow adjustment of the volume, i.e., the pressure. In the limit where $Q \to \infty$ and $dV/dt = 0$ initially, these equations of motion become equivalent to the dynamical equations for normal molecular dynamics.

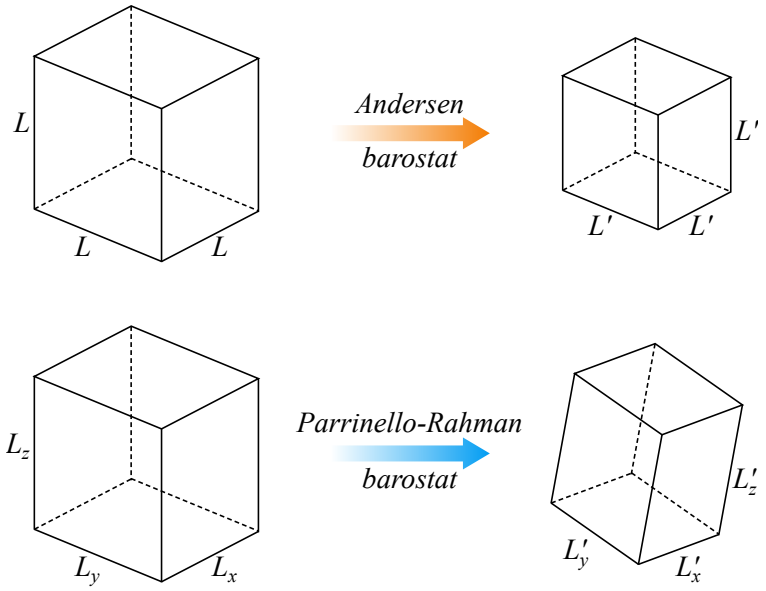

Figure 5.8. Difference between the Andersen barostat and the Parrinello-Rahman barostat

6. Acceleration Techniques on Simulation

The goal of molecular dynamics simulations is to provide information about the macroscopic properties of atomic or molecular systems. However, it is inevitably restricted by the computational power. Although the computing power has increased steadily over the past decade, most simulations probe the structural and thermodynamical properties of a system with the size ranging from a few hundred to a few million particles. Clearly speaking, although this number is large, it is still far below the thermodynamical limit (~Avogadro number 10^{23}). As the number of particles in the system becomes larger, longer simulation time is required. Moreover, since the typical time scale for atomic motions is of the order of a picosecond or less, we should use values of the order of a femtosecond for the time-step to keep the numerical errors small. Therefore, it may take may take millions of iterations to follow the motion of a single particle for just a microsecond. Under these circumstances, many time-saving methods have been developed to accelerate the simulations. In this section, we shall discuss several computational techniques that are of great practical importance for the molecular dynamics method.

6.1. Potential Cutoff

Generally, the original functional forms of the interacting potentials, for instance the Lennard-Jones potential in equation (4.3), have an infinite interaction range. Therefore, in order to reduce the calculation cost in molecular dynamics simulation, the potential functions, except for the long-range interaction such as the Coulomb interaction, are usually designed to have a cutoff radius r_c which is the maximum atomic distance to the central atom in which the surrounding atoms affect on the central atom. If the atomic force has a long-range interaction (electrostatic), this problem is very serious and the complicated methods like the Ewald summation scheme, which will not be discussed in this chapter, is needed to perform the calculation effectively. Back to the case of short-range interaction, the termination of the potentials function must satisfy several requirements: (1) Potential (or force) should remain minimally perturbed by modifying function at small distances; (2) Modified potential (or force) must remain a smooth function. This requirement is crucial because any violation may result in catastrophic instability of the integration of equation of motions due to sudden variation of forces; and (3) Because energy is conserved in standard molecular dynamics simulations (NVE ensemble), modification of potentials should not lead to noticeable energy drift. The error that results when we ignore interactions with particles at larger distances can be made arbitrarily small by choosing r_c sufficiently large. Often used methods to modify the potential are the followings:

Simple Truncation. The most simple implementation is to ignore all interactions beyond r_c . In this case, the potential that is simulated is:

$$U^{\mathrm{mod}}(r)=\begin{cases} U(r), & r \le r_c \\ 0, & r > r_c \end{cases} \tag{6.1}$$

where $U(r)$ is the original potential. Generally, this simple truncation creates a discontinuity in the potential, which then results in the infinite force at r_c. Because the truncation violates at least the second requirement above, it is not suitable for a molecular dynamics simulation. However, if the potential function is already close to zero at r_c, the numerical error associated with the truncation is small and might be acceptable. In addition, it can be used in Monte Carlo simulations. In that case, one should be aware that there is an 'impulsive' contribution to the pressure due to the discontinuous change of the potential at r_c.

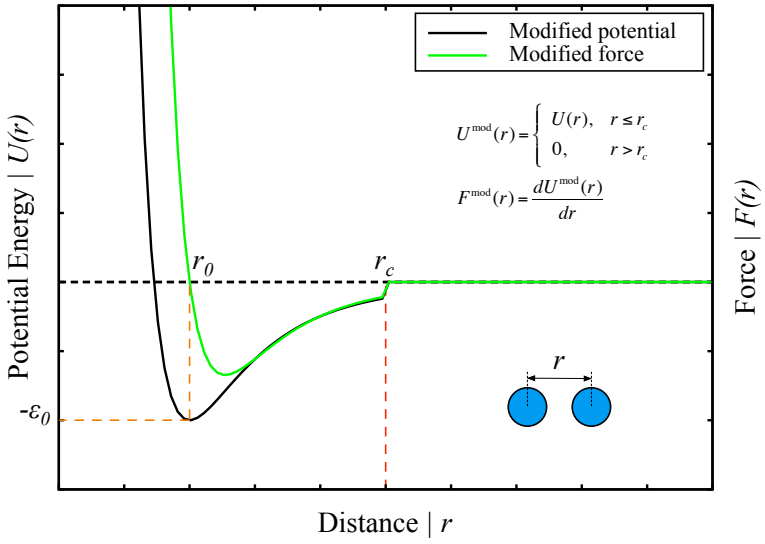

Figure 6.1. Simple truncation of Lennard-Jones potential

Truncation and Shift. The discontinuity of the potential in the simple truncation method can be smeared out by changing the form of the potential function slightly, although this must be done carefully since it is the potential that defines the model:

$$U^{\mathrm{mod}}(r)=\begin{cases} U(r)-U(r_c), & r \le r_c \\ 0, & r > r_c \end{cases} \tag{6.2}$$

In this case, the modified potential vanishes at the cutoff radius and the continuity is ensured, however, the force is not affected and still has discontinuity at the cutoff distance. In order to ensure zero force value and eliminate the discontinuity at the cutoff point in the force functions, another term can be added to the potential. The modified function form then becomes:

$$U^{\mathrm{mod}}(r)=\begin{cases} U(r)-U(r_c)-\left.\dfrac{dU(r)}{dr}\right|_{r_c}(r-r_c), & r \le r_c \\ 0, & r > r_c \end{cases} \tag{6.3}$$

As we can see, this form satisfies:

$$U(r)\big|_{r_c}=0 \tag{6.4}$$

and:

$$\left.\frac{dU(r)}{dr}\right|_{r_c} = 0 \tag{6.5}$$

The discontinuity now appears in the gradient of the force, not on the force itself. The force goes smoothly to zero at the cutoff r_c, removing problems in energy conservation and any numerical instability in integrating the equations of motion. Due to the continuity in the potential and the force, it can be used in molecular dynamics simulations, and no 'impulsive' corrections to the pressure is needed.

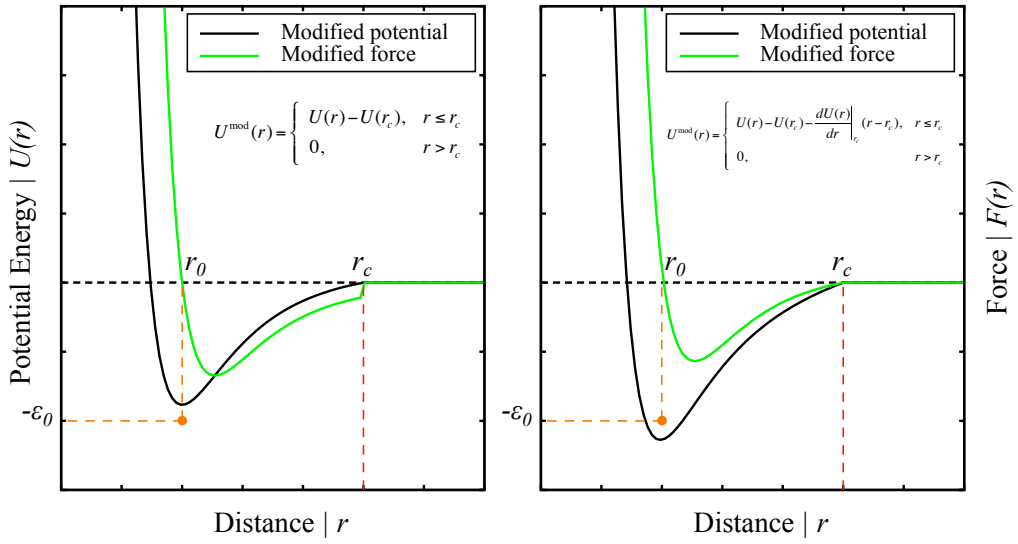

Figure 6.2. Truncation and shift of Lennard-Jones potential

Switching. Another approach is to multiply to the original potential functions by a switching function $g(r)$ on the interval $[r_s, r_c]$:

$$U^{\mathrm{mod}}(r) = g(r)U(r) \tag{6.6}$$

The switching function must satisfy the continuity conditions at r_s and r_c, which are:

$$g(r)\big|_{r_s} = 1 \tag{6.7}$$

$$\left.\frac{dg(r)}{dr}\right|_{r_s} = \left.\frac{d^2g(r)}{dr^2}\right|_{r_s} = 0 \tag{6.8}$$

$$g(r)\big|_{r_c} = 0 \tag{6.9}$$

$$\left.\frac{dg(r)}{dr}\right|_{r_c} = \left.\frac{d^2g(r)}{dr^2}\right|_{r_c} = 0 \tag{6.10}$$

Here, r_s is the starting distance where the termination starts and r_c is the distance where the potential totally vanishes.

One example is the cutoff function found in the Tersoff potential:

$$g(r)=\begin{cases}1, & r\le r_s\\ \dfrac{1}{2}+\dfrac{1}{2}\cos\left(\dfrac{\pi(r-r_s)}{r_c-r_s}\right), & r_s<r\le r_c\\ 0, & r>r_c\end{cases} \tag{6.11}$$

Other examples for $g(r)$ on the interval $[r_s, r_c]$ might be a polynomial function of fifth order:

$$g(r)=\sum_i c_i (r-r_c)^i \tag{6.12}$$

where c_i is polynomial coefficients determined from the conditions (6.7), (6.8), (6.9) and (6.10); or an exponential form:

$$g(r)=e^{\frac{(r-r_s)^3}{(r-r_s)^2-(r_s-r_c)^2}} \tag{6.13}$$

By using the switching function, the interacting potential will gradually reduce to zero in the interval $[r_s, r_c]$ and there is no discontinuity in both the potential and the force.

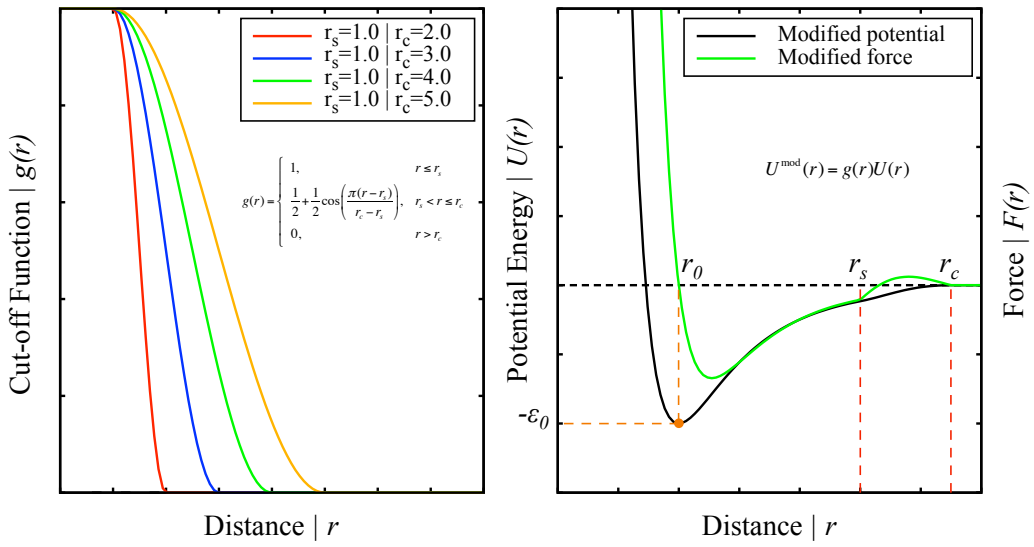

Figure 6.3. Switching of Lennard-Jones potential

6.2. Book-keeping Method

The main part of molecular dynamics simulation is dealing with the interactive forces between atoms. Cutoff procedures alone does not save much computational time, because they still require calculation of the distances between all the pairs of atoms to make an appropriate modification of the potential. Therefore, the computations still scale as $O[N^2]$, where N is the number of atoms. If we only consider the short-range interaction in the simulation, which is very common for most metallic systems, a conventional method called book-keeping method (a.k.a. Verlet lists) [6] can be used effectively to save the simulation time.

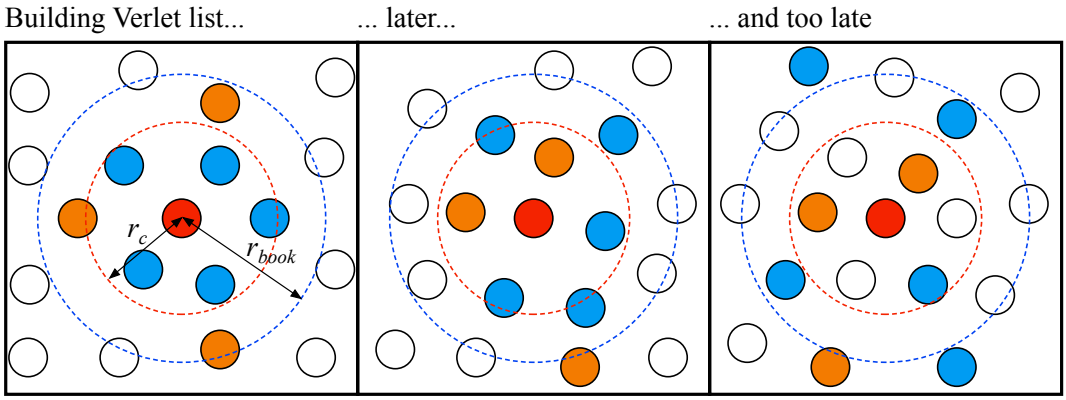

Figure 6.4. The book-keeping method

The concept of the book-keeping method is as follows: each atom has the list (book-keeping list or Verlet list) of its neighboring atoms which reside within the cutoff sphere of radius:

$$r_{book} = r_c + r_{margin} \tag{6.14}$$

where r_c is the interaction range and r_{margin} is the safety zone or the skin distance. This 'book-keeping' list is regularly updated after a certain number of time-steps N_{book}. In the following N_{book} integration steps, only the interactions between atoms in the list are checked in the force calculation routine. After every N_{book} steps, the list is updated by recalculating the distances between all atomic pairs. Because of that, one does not have to run $N(N-1)/2$ times loop for calculation of forces at every time-step. Updating the list still takes a time of $O[N^2]$, however, the time for usual force calculation can be reduced to $O[N]$.

The cutoff radius of the book-keeping list r_{book} should be long enough for other atoms outside of the list which have crossed the safety zone r_{margin} not to come into the interaction range r_c until the next update time. Thus, the minimum requirement for r_{book} is:

$$r_{book} = r_c + N_{book}\Delta t v_{max} \tag{6.15}$$

where v_{max} is the maximum velocity of the atoms. The choice of list cutoff distance r_{book} is a compromise: larger lists will need to be reconstructed less frequently, but will not give as much of a saving on cpu time as smaller lists. This choice can easily be made by experimentation.

6.3. Linked-cell Method

The book-keeping method above only speeds up the simulation if the number of atoms in the Verlet list much smaller than the total number of atoms. For larger system ($N > 1000$ or so, depending on the potential range), another method for reducing the $O[N^2]$ scaling in the calculation of forces becomes preferable. It is the so-called 'linked-cell' method (a.k.a. cell lists) [34]. In this method, the cell lists are generated by dividing the simulated system into cubic cells of the size $r_{cell} \geq r_c$. For each cell, the list of atoms which belong to a given cell is created. In the course of simulation, the calculation of interactions for an atom in a given cell is restricted only to the current cell and its neighboring cells.

In the linked-cell method, the number of atoms to be considered during the calculation of forces is several times larger than that of the book-keeping method. However, every procedure in the program, such as distributing the atoms to their appropriate cells and searching neighboring atoms will be finished within the $O[N]$ scaling, and less memory resources are required than in the book-keeping method. Practically, the powerful approach is to create first cell lists and then generate book-keeping lists based on existing cell lists. By this way, it allows to reduce the overall scaling from $O[N^2]$ to $O[N]$.

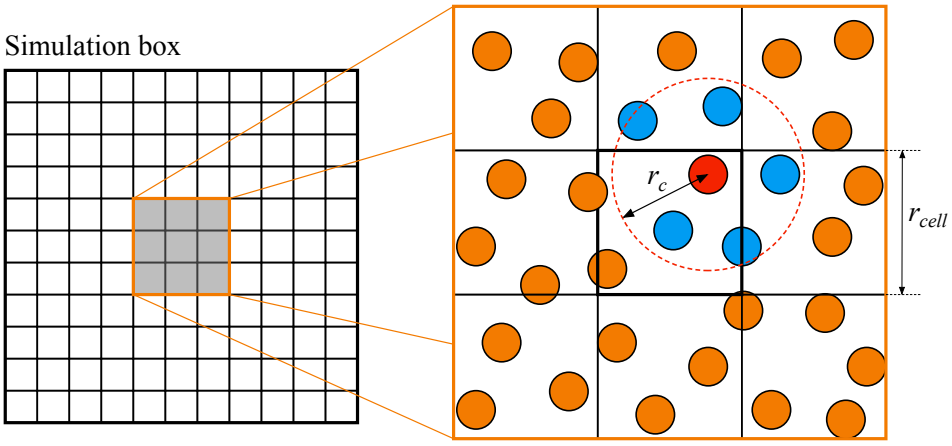

Figure 6.5. The linked-cell method

6.4. Periodic Boundary Conditions

In most case, the purpose of simulation using molecular dynamics is to investigate the properties of an infinite molecular system at given structural conditions such as the concentration of some species in bulk compound. Simulation of that infinite system is impossible without any confinement to the size of the system. In order to overcome such length scale problem, it is essential to choose boundary conditions that mimic the presence of an infinite bulk surrounding the N -particle system. To do so, all atoms are confined in a box, and then replicate the box, together with the atoms in the box, identically in three directions. The box containing the particles is treated as the primitive cell of an infinite periodic lattice of identical cells. As a consequence, an atom leaving the box to the right via one boundary can be identified with an atom entering the box from the left at the opposite boundary. In addition, a given atom will interact with all other atoms not only in the primitive cell but also in all other replicated cells. For example, if we assume that all interatomic interactions are pairwise additive, then the total potential energy of the particles in any one periodic box is:

$$E_{tot} = \frac{1}{2}\sum_{l,m,n}\sum_{i,j} U\left(\left|r_{ij} + la + mb + nc\right|\right) \tag{6.16}$$

where a , b , c are the vectors corresponding to the edges of the primitive cell and l , m , n are arbitrary integer numbers specifying the position of the box among all replicated boxes.

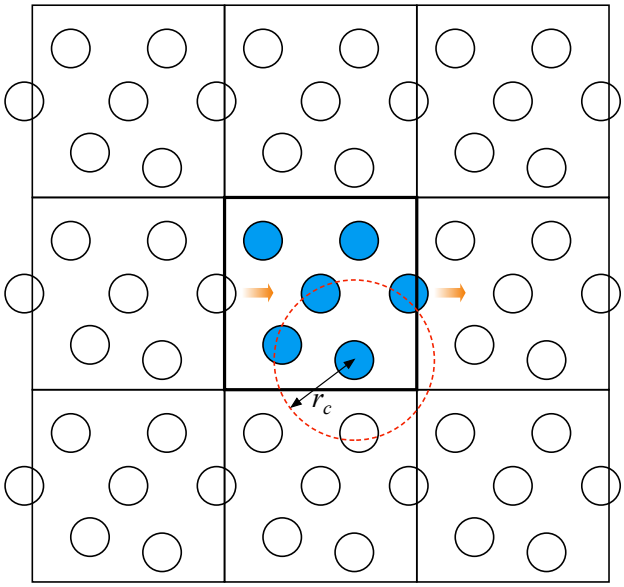

Figure 6.6. The periodic boundary conditions

Practically, we often deals with short-range interactions and the cutoff in the non-bonded interactions limit the sum over l, m, n in equation (6.16) to nearest replicas. Finally, although the use of periodic boundary conditions proves to be surprisingly effective, it is important to bear in mind that the method may lead to spurious correlations not present in a truly macroscopic bulk system. Also, special attention must be paid to the case where the potential range is not short, for instance, the charged and dipolar systems, because long-range Coulomb interaction exists.

7. Molecular Dynamics Simulation Examples

Molecular dynamics evolves a finite-sized molecular configuration forward in time, in a step-by-step fashion. Basically, the simulation process can be done following the steps below:

(1) Set initial conditions: positions, velocities, time-step, interatomic potential type, boundary conditions,…
(2) Update neighbor list if necessary: book-keeping, linked-cell,…
(3) Calculate forces from interatomic potential
(4) Solve the equations of motion: Verlet, Gear,…
(5) Perform temperature and pressure controls
(6) Calculate the desired physical quantities
(7) Update time: $t \rightarrow t + \Delta t$
(8) If the current step exceeds the maximum number of step then advance to (9), otherwise repeat (2) → (8)
(9) Calculate final results and end the simulation

Practically, the initial settings for molecular dynamics simulations can be chosen based on some rules and/or tricks as follows:

Initial positions. For crystalline solids, the atoms are usually placed according to the lattice structure such as BCC, FCC, HCP,… If simulation conditions correspond to the liquid phase, one can either introduce small deviations from the regular lattice positions or give the particles some random initial velocities. For molecules, atoms should be placed in idealized or approximate geometries that satisfy the known bond lengths and angles. Finally, in the case of atoms, they can be placed randomly in the simulation box. However, one should check every distance between two atoms to ensure that the atoms should not be too close to each other or being overlapped. In any case, it is highly recommended to geometrically optimize the initial structure to bring the system to a nearby local minimum. This can remove any overlap present in the system, and sometimes can fix up bond lengths and angles.

Initial velocities. An initial velocity, including its magnitude and direction, must be assigned to each particle. Typically, one can pick random velocities for the particles either from a random uniform distribution or from the Maxwell-Boltzmann distribution corresponding to the temperature T (in K) of the particle ensemble. For the later, the distribution function is as follows:

$$f(v_i^\alpha) = \sqrt{\frac{m_i}{2\pi k_B T}} e^{-\frac{m(v_i^\alpha)^2}{2k_B T}} \tag{7.1}$$

where the superscripts $\alpha = x, y, z$ refer to the Cartesian directions. This is simply the Gaussian function (or normal distribution):

$$\psi(x)_{\mu,\sigma} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x-\mu}{2\sigma^2}} \tag{7.2}$$

with $\mu = 0$, $x = v_i^\alpha$ and $\sigma = \sqrt{k_B T / m_i}$.

In practice, we assign each velocity component of each particle separately:

$$v_i^\alpha = \sqrt{\frac{k_B T}{m_i}} u_G \tag{7.3}$$

where independent Gaussian ($\mu = 0$, $\alpha = 1$) random numbers u_G are obtained from Gaussian random number generator.

Note that, if we only pick the initial velocities based on the equation (7.3), it can lead to a non-zero net momentum of the system, which then create a systematic translational drift of the simulated system. To avoid this, one can choose the initial velocities so that the following conditions is satisfied:

$$\sum_{i=1}^{N} v_i^{\alpha} = 0 \tag{7.4}$$

In practice, the procedure is to find the net momentum:

$$P = \sum_{1=1}^{N} m_i v_i \tag{7.5}$$

Then subtract from each of the atomic velocities so as to remove the net momentum:

$$v_i \leftarrow v_i - \frac{1}{N}\frac{P}{m_i} \tag{7.6}$$

Time-step. This is a critical parameter in the simulation. Too small time-step gives correct result but makes the simulation inefficient and take a very long time to finish. Whereas, too large time-step can make simulation unstable with the total energy rapidly increasing with time (exploding). This is because the particles might be brought too close to each other and even nearly overlapping, which leads to large sudden changes in kinetic energies. Conventionally, based on experience, the time-step is chosen to be a constant throughout the simulations and it should be at least an order of magnitude smaller than the fastest time scale in the system.

System type	Type of motion	Time step (fs)
Atoms	Translation	5 ÷ 10
Rigid molecules	Translation and rotation	5.0
Flexible molecules, rigid bonds	Translation, rotation and torsion	2.0
Flexible molecules, flexible bonds	Translation, rotation, torsion and vibration	0.5 ÷ 1.0
Lennard-Jones system	Mixed	0.001 (dimensionless)

Table 7.1. Recommended time-step for simulation

Interatomic potential. The determination or choice of interaction potential lies at the heart of molecular dynamics simulations and it greatly depends on type of the system of interest. A good potential must lead to correct simulation of observed physical phenomena. Quantitatively, calculated properties must correspond to experimentally measured effects, up to the error inherent in the numerical and experimental results. In addition, the evaluation of interaction forces (potential's derivatives) is the most time-consuming part therefore the potential functional forms must be chosen carefully so as to reduce the computation time but still retain acceptable accuracy. In practice, the potential can be re-used from literatures, provided that it can reproduce the physical properties of the simulated materials. Popular database for interatomic potential can be found at http://www.ctcms.nist.gov/potentials/. If the potential does not exists, it must be constructed before any molecular dynamics simulations are being carried out.

Bond type	Potential	Speciality
Ionic	Buckingham	Inert gases, ionic systems
Van de Waals	Lennard-Jones	Inert gases, organic materials
Covalent	Stillinger-Weber	Silicon (liquid phases), semiconductor
	Tersoff-Brenner	Carbon-based materials, hydrocarbons
	REBO/AIREBO	Carbon-based materials
	ReaxFF	Transition-metal-catalyzed, hydrocarbon reactions
Metallic	EAM	Metals and alloys
	Sutton-Chen	Transition metal clusters
	Finnis-Sinclair	Transition metals, heterogeneous systems
Mixed	MEAM	EAM extended to non-metallic systems
	ADP	EAM extended to include angular-dependence term
	COMB	Heterogeneous surfaces/interfaces

Table 7.2. Interatomic potentials corresponding to bond types and their specialities

In order to construct or develop the interatomic potential, one can use some optimization methods such as genetic algorithm, simulated annealing, conjugated gradient,… to fit the potential to experimental data or first-principles calculation results.

Physical property	Atom-level property
Crystal structure	Balance of atomic forces
Cohesive energy	Potential energy at the equilibrium atom positions
Elastic constants	Elastic distortions of unit cell
Equation of state	Compression or expansion of material

Table 7.3. Some important physical quantities for potential fitting

Indeed, the development of an interatomic potential is not a straightforward process. The common process of making potential consists of: (1) Selection of a potential framework and functional form that can capture the underlying physics of interatomic interaction in the simulated materials. The functional form should contain a sufficient number of adjustable parameters so that the potential can be fitted to reproduce a selected set of material properties; (2) Construction of fitting database which can include experimental and/or theoretical data. The physical quantities in the database often reflect which material behaviors the developer is most interested in; (3) Optimization of potential parameters by minimizing the sum of squares of the deviations between the reference values in fitting database and those calculated from the potential. The function to be minimized is the so-called target function, and is defines as follows:

$$Z(s) = \frac{1}{m}\sum_{k=1}^{m} w_k \left[\xi_k(s) - \xi_k^0\right]^2 \tag{7.7}$$

where m is number of configurations being fitted, s is set of parameters, ξ_k^0 are the reference values of configuration k, $\xi_k(s)$ are the corresponding values calculated from the potential, and w_k are the weights attached to each configuration to include the contribution level of the configuration to $Z(s)$. Indeed, evaluation of the target function actually the most time-consuming part in potential development; (4) Verification of the final potential to ensure the transferability characteristic. This is key issue in developing a good interatomic potential because this characteristic shows the ability to accurately describe or predict a material's property that is not included in the fitting database.

7.1. Introduction to LAMMPS

In order to carry out simulations using molecular dynamics, we need to have a computer program to implement the molecular dynamics flowchart stated earlier. For a simple problem, one can easily construct a simple program using FORTRAN or C/C++,… to integrate the equations of motion and perform temperature/pressure controls. However, for complicated problems, making a reliable and re-useable molecular dynamics code is time-consuming. Instead of having to explicitly write the code for the system of interest, we can simply use existing packages which already incorporated the numerical methods needed for simulating the system. One of the well-known program being used currently is the Large-scale Atomic/ Molecular Massively Parallel Simulator (LAMMPS) developed by Plimpton [35] and co-workers using C++ programming language and distributed under the GNU Public License (GPL). This classical molecular dynamics code can model atomic, polymeric, biological, metallic, granular, and coarse-grained systems using a variety of force field and boundary conditions. Extensive information and user manual can be found at http://lammps.sandia.gov. In this section, we will briefly introduce about LAMMPS.

Typically, LAMMPS uses five kinds of files, some are always required and some are optional:

Input file. The input file specifies the details of the system and the integrator, e.g. what kind of interactions are there, what are the numeric constants used in these interactions, what kind of integration should be used, what kind of statistical ensemble should be performed, etc. Each line in the input file is considered as a command, each command causes the program to take some action. Most of the commands have default settings, we only need to use the command if we wish to change the default.

Structural file. The configuration file contains the details of the initial configuration such as simulation box, atomic positions, bonding structure, etc. Note that it is not always necessary to have this file present. Using the input file it is another possible to generate initial configurations, thus eliminating the need of a configuration file.

Potential file. The potential file contains the interatomic potential data which governs the interaction of the simulated system such as parameters in Tersoff potential formulation, tabular data of the pairwise functions, the embedding functions, the density functions in the embedded atom method, etc. For some systems such as Lennard-Jones, Buckingham,… it is not necessary to have this file. In these cases, the potential parameters can be specified in the input file.

Log file. The log file keeps track of some useful quantities like energy and temperature and also contains the warnings and errors thrown during execution.

Output file. This file contains snapshots of the system on times specified by the user during the simulations. There are different types of output files and one can have more than one output file per simulation run.

LAMMPS is designed to efficiently integrate the equations of motion for a system of interacting particles. To keep it simple and running fast, many of the tools needed to pre- and post-process the data for the simulation is not included in the program kernel. Most users pre- and post-process the input and output files with their own analysis tools or re-format them for input into other programs, including visualization programs. For high-quality visualization, users can use the following packages:

OVITO. A scientific data visualization and analysis software for the output data from particle-based simulation models in materials science and related disciplines.
Website: http://www.ovito.org/
Support OS: Windows, Macintosh, Linux
License: Open source (GNU Public License)

VMD. A molecular visualization program for displaying, animating, and analyzing large biomolecular systems using 3D graphics and built-in scripting.
Website: http://www.ks.uiuc.edu/research/vmd/
Support OS: Windows, Macintosh, Linux
License: Open source (UIUC Open Source License)

VESTA. A 3D visualization program for structural models, volumetric data such as electron/nuclear densities, and crystal morphologies.
Website: http://jp-minerals.org/vesta/
Support OS: Windows, Macintosh, Linux
License: Free of charge for non-commercial use (VESTA License)

ParaView. A 3D program for data analysis and visualization with batch processing capabilities
Website: http://www.paraview.org/
Support OS: Windows, Macintosh, Linux
License: Open source (BSD License)

CrystalMaker. A program for building, displaying, manipulating and animating all kinds of crystal and molecular structures.
Website: http://www.crystalmaker.com/
Support OS: Windows, Macintosh
License: Commercial

7.2. Example 1: Harmonic Oscillator

In this section, the NVE simulation of two particles connected by a Hookean spring will be shown. The interaction potential between two particles is in the form of Hooke's law:

$$U(x) = \frac{1}{2}k(x - x_0)^2 \tag{7.8}$$

For this examples, the spring constant k is set to 2 and the equilibrium length x_0 of the spring is equal to 0. The initial extension of the spring is 1. Note that, in LAMMPS, the usual $1/2$ factor is included in k.

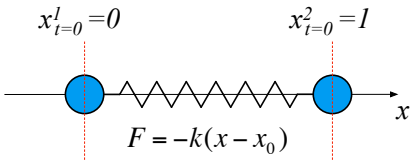

Figure 7.1. Harmonic oscillator

Configuration file (ex1_harm.dat):

```
2 atoms                                          # Number of atoms
1 atom types                                     # Number of atom types
1 bonds                                          # Number of bonds
1 bond types                                     # Number of bond types
-10 10 xlo xhi                                   # Boundaries of the box
-10 10 ylo yhi
-10 10 zlo zhi

Atoms                                            # Atoms: atom-ID molecule-ID atom-type x y z

1 1 1 0.0 0.0 0.0
2 1 1 1.0 0.0 0.0

Bonds                                            # Bonds: ID type atom1 atom2

1 1 1 2
```

Input file (ex1_harm.in):

```
# Molecular Dynamics Simulations with LAMMPS
# Example 1: Harmonic oscillator
# ----------------------------------------

atom_style bond                                 # Spring between the particles
atom_modify sort 0 0.0                          # Turn off the sorting to avoid an error...
read_data ex1_harm.dat                          # Read the initial configuration
bond_style harmonic                             # Use harmonic potential for the bond
bond_coeff 1 1.0 0.0                            # The coefficients for the harmonic bond
mass 1 1.0                                      # Set the mass of all atoms
velocity all set 0.0 0.0 0.0                    # Set the velocity of all atoms
fix nve_fix all nve                             # Perform NVE simulation
compute Kenergy all ke                          # Calculate the total kinetic energy
compute Venergy all pe                          # Calculate the total potential energy
variable K equal c_Kenergy                      # Store the energies in three variables
variable V equal c_Venergy
variable E equal c_Kenergy+c_Venergy
variable x1 equal x[1]                          # Store the positions of the two particles
variable x2 equal x[2]
variable dt equal 0.005                         # Variable containing the timestep
variable t equal step*${dt}                     # Variable containing the current time
timestep ${dt}                                  # Set the timestep for integration
thermo 50                                       # Frequency print out thermo information
fix en_print all print 10 "$t $K $V $E" &       # Print for NVE simulation
    file energy.dat screen no &
    title "#Time Kin Pot Tot"
fix pos_print all print 10 "$t ${x1} ${x2}" &
    file position.dat screen no title "#Time x1 x2"
dump coords all atom 10 dump.harm               # Export atomic positions for visualization
dump_modify coords sort id
run 1500                                        # Run the simulation
print "Simulation complete"                     # Simulation finished
```

Tasks:

(1) Run NVE simulation and plot the positions of particles as a function of time. We will see that the two particles harmonically oscillate.

(2) Plot the kinetic energy, the potential energy and the total energy versus time. See if the total energy is conserved as expected from NVE simulation.

(3) Change spring constant and see the change of the kinetic and potential energies.

7.3. Example 2: Lennard-Jones Fluid

In this example, we are going to simulate a system of N particles of mass $m = 1$ that are placed randomly inside a cubic box of width $L = 10$. We can calculate the density of particles via the formula:

$$\rho = \frac{N}{V} = \frac{N}{L^3} \tag{7.9}$$

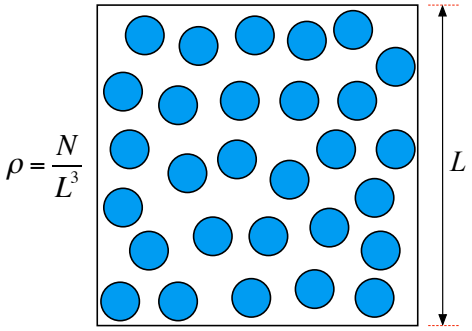

Figure 7.2. Lennard-Jones fluid

The interaction of particles is governed by a Lennard-Jones potential with truncation and shift at the cutoff distance $r_c = 2.5$:

$$U_{LJ}(r) = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \tag{7.10}$$

$$U(r) = \begin{cases} U_{LJ}(r) - U_{LJ}(r_c), & r \le r_c \\ 0, & r > r_c \end{cases} \tag{7.11}$$

Note that, for this kind of potential, we are using the Lennard-Jones unit system in LAMMPS, where the two parameters σ and ε are set to 1. The units for other physical quantities will be specify from these two parameters. Due to the random arrangement of the initial positions of all particles, they might get too close to each other or even overlap. Which may cause extremely high forces between particles, and in turn, lead to missing particles. Therefore, we need to optimize the system slightly to push particles away from each other. For NVE simulation, the input file for this example is as follows:

Input file (ex2_fluid.in):

```
# Molecular Dynamics Simulations with LAMMPS
# Example 2: Lennard-Jones Fluid
# ----------------------------------------

units lj                                          # Unit type: Lennard-Jones
dimension 3                                       # Specify boundary conditions
boundary p p p
atom_style atomic                                 # Define classical particles
variable N equal 300                              # Variable containing N, density, box size
variable dens equal 0.3
variable L equal (${N}/${dens})^(1.0/3.0)
region box block 0 ${L} 0 ${L} 0 ${L} units box   # Specify simulation box
create_box 1 box                                  # Initiate box with one atom type
create_atoms 1 random ${N} 12345 NULL             # Create atoms randomly in simulation box
mass 1 1.0                                        # Set the mass of all atoms
pair_style lj/cut 2.5                             # Use Lennard-Jones potential for interaction
pair_coeff * * 1.0 1.0 2.5
pair_modify shift yes                             # Shift the potential
minimize 1e-2 1e-2 10000 100000                   # Optimize structure to avoid overlapping
min_style cg                                      # Minimize using conjugate gradient method
neighbor 0.3 bin                                  # Build neighbor list
neigh_modify delay 5                              # Frequency to rebuild neighbor list
variable T0 equal 2.0                             # Variable containing the current temperature
velocity all create ${T0} 12345 &                 # Create initial velocities
         mom yes rot yes dist gaussian
fix nve_fix all nve                               # Perform NVE simulation
compute Kenergy all ke                            # Calculate the total kinetic energy
compute Venergy all pe                            # Calculate the total potential energy
variable K equal c_Kenergy                        # Store the energies in three variables
variable V equal c_Venergy
variable E equal c_Kenergy+c_Venergy
variable dt equal 0.005                           # Variable containing the timestep
variable t equal step*${dt}                       # Variable containing the current time
reset_timestep 0                                  # Specify the timestep for integration
timestep ${dt}
thermo 50                                         # Frequency print out ensemble data
thermo_style custom step temp pe ke etotal        # Specify what to print to screen & log file
fix en_print all print 10 "$t $K $V $E" &         # Print for NVE simulation
    file energy.dat screen no &
    title "#Time Kin Pot Tot"
dump coords all atom 10 dump.liquid               # Export atomic positions for visualization
dump_modify coords sort id
run 1500                                          # Run the simulation
print "Simulation complete"                       # Simulation finished
```

Tasks:

(1) Run NVE simulation with time-step $\Delta t = 5 \times 10^{-3}$ and particle density $\rho = 0.3$ at temperature $T = 2$. Plot the total energy as a function of time and see if it is conserved as expected.

(2) Turn on a Nosé-Hoover thermostat for $T = 2$, using an NVT in stead of NVE and repeat the simulation. Check the total energy again and compare the kinetic energy in this simulation to the kinetic energy calculated from the expression $\langle K \rangle = 3Nk_BT/2$ obtained from the equipartition theorem.

(3) Change the particle density within the range $0 < \rho \leq 0.2$. Plot the pressure P of the system as a function of the density and compare with the pressure calculated from the ideal gas law $P = \rho k_B T$.

7.4. Example 3: Crack Growth in 2D Lennard-Jones Crystal

In this example, we are going to model tensile pull and see the crack growth in a pre-cracked 2-dimension Lennard-Jones crystal with hexagonal lattice. The construction of the crystal with a initial crack on the left side can be totally done using LAMMPS built-in commands.

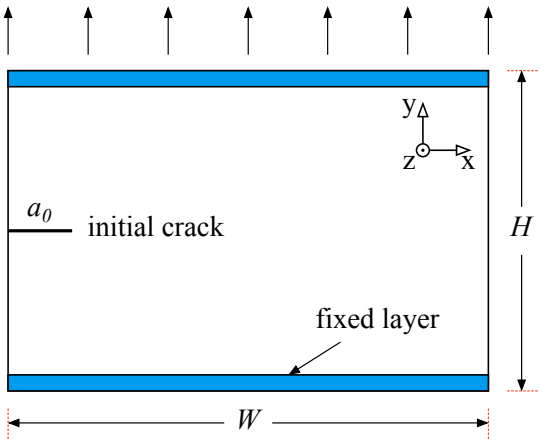

Figure 7.3. Crack growth in 2D Lennard-Jones crystal

The interaction potential is in a form of a Lennard-Jones potential with the truncation and shift not only for the potential but also for the forces at the cutoff distance $r_c = 2.5$ (Lennard-Jones units):

$$U_{LJ}(r) = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \tag{7.12}$$

$$U(r) = \begin{cases} U_{LJ}(r) - U_{LJ}(r_c) - \left.\dfrac{dU_{LJ}(r)}{dr}\right|_{r_c}(r - r_c), & r \leq r_c \\ 0, & r > r_c \end{cases} \tag{7.13}$$

The input file for this example is as follows:

Input file (ex3_crack.in):

```
# Molecular Dynamics Simulations with LAMMPS
# Example 3: Crack Growth in 2D LJ Crystal
# ----------------------------------------

units lj                                        # Unit type: Lennard-Jones
dimension 2                                     # Specify boundary conditions
boundary s s p
atom_style atomic                               # Define classical particles
lattice hex 0.93                                # Specify lattice structure
```

```
region box block 0 100 0 40 -0.25 0.25              # Specify simulation box
create_box 5 box                                    # Initiate box with five atom types
create_atoms 1 box                                  # Create atom type 1 in simulation box
mass * 1.0                                          # Set the mass of all atoms
pair_style lj/sf 2.5                                # Use Lennard-Jones potential for interaction
pair_coeff * * 1.0 1.0
region 1 block INF INF INF 1.25 INF INF             # Define groups of atoms for upper and lower
group lower region 1
region 2 block INF INF 38.75 INF INF INF
group upper region 2
group boundary union lower upper
group mobile subtract all boundary                  # Define group of mobility atoms
region leftupper block INF 20 20 INF INF INF
region leftlower block INF 20 INF 20 INF INF
group leftupper region leftupper
group leftlower region leftlower
set group leftupper type 2                          # Set each group to atom types
set group leftlower type 3
set group lower type 4
set group upper type 5
compute new mobile temp                             # Create initial velocities
velocity mobile create 0.01 12345 temp new
velocity upper set 0.0 0.6 0.0
velocity mobile ramp vy 0.0 0.6 y 1.25 38.75 sum yes
fix 1 all nve                                       # Perform NVE simulation
fix 2 boundary setforce 0.0 0.0 0.0
timestep 0.003                                      # Specify the timestep for integration
thermo 200                                          # Frequency print out ensemble data
thermo_modify temp new                              # Determine how thermo temp is calculated
neighbor 0.3 bin                                    # Build neighbor list
neigh_modify delay 5                                # Frequency to rebuild neighbor list
neigh_modify exclude type 2 3                       # Exclude interaction of types 2 & 3
dump coords all atom 10 dump.crack                  # Export atomic positions for visualization
dump_modify coords sort id
run 5000                                            # Run the simulation
print "Simulation complete"                         # Simulation finished
```

Tasks:

(1) Run NVE simulation with time-step $\Delta t = 3 \times 10^{-3}$ and ramp velocity (pull rate) within the range $0 < v \leq 0.6$. Use OVITO or any visualization software to see how the crystal cracks.

(2) Change the pull rate and repeat the simulation. Check the growth of the crack. Try to visualize (and make a movie from a seri of images) the strain on atoms in the system while pulling.

7.5. Example 4: Melting/Cooling of A Nano Particle

In this example, we are going to simulate the melting of a spherical nano gold particle placed inside a cubic box of edge length $L = 8.16$ nm. The particle of diameter $d = 3.25$ nm is generated from a perfect gold bulk structure (FCC structure, $a = 4.0782$ Å).

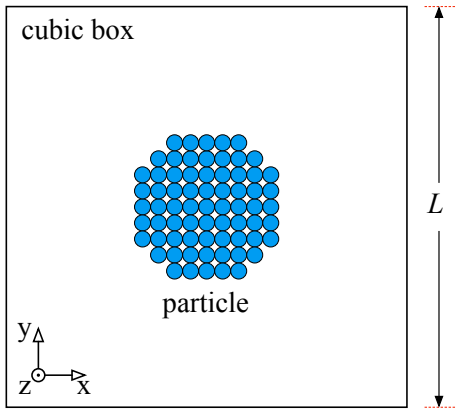

Figure 7.4. Melting/Cooling of A Nano Particle

We use the embedded atom method potential for the interaction of gold atoms from file *Au_u3.eam* which can be found in the potential folder of LAMMPS source code. Note that the cutoff distance for the embedded atom method potential is pre-specified in the potential file. The input file for this example is as follows:

Input file (ex4_melt.in):

```
# Molecular Dynamics Simulations with LAMMPS
# Example 4: Melting of A Nano Particle
# ----------------------------------------

units metal                                          # Unit type: Metal
dimension 3                                          # Specify boundary conditions
boundary p p p
atom_style atomic                                    # Define classical particles
lattice fcc 4.0782                                   # Specify lattice structure (FCC Au)
region box block 0 20 0 20 0 20                      # Specify simulation box
create_box 1 box                                     # Initiate box with one atom type
create_atoms 1 box                                   # Create atom type 1 in simulation box
mass 1 196.97                                        # Set the mass of all atoms equal to Au mass
region sph sphere 10.0 10.0 10.0 4.0 side out        # Create a sphere in the middle of the box
delete_atoms region sph                              # Remove atoms outside the sphere
pair_style eam                                       # Use EAM potential for interaction
pair_coeff * * Au_u3.eam
neighbor 3.0 bin                                     # Build neighbor list (skin distance 3.0Å)
neigh_modify every 20 delay 0 check no               # Frequency to rebuild neighbor list
minimize 1e-8 1e-8 10000 100000                      # Optimize initial structure
min_style cg                                         # Minimize using conjugate gradient method
velocity all create 200.0 12345 &                    # Create initial velocities
         mom yes rot yes dist gaussian
reset_timestep 0                                     # Specify the timestep for integration
timestep 0.001
thermo 5000                                          # Frequency print out ensemble data
thermo_style custom step pe ke etotal temp vol press # Specify what to print to screen & log file
fix nvt_fix1 all nvt temp 200.0 200.0 0.1            # Perform NVT simulation (thermal relaxation)
fix 2 all momentum 10 linear 1 1 1 angular
fix 3 all recenter INIT INIT INIT
run 50000                                            # Run the simulation to reach equilibrium
unfix nvt_fix1
dump coords all atom 1000 dump.melt                  # Export atomic positions for visualization
dump_modify coords sort id
variable N equal "count(all)"                        # Save data to analyze
variable T equal temp
variable K equal ke
variable V equal pe
variable E equal etotal
fix vket_ave all ave/time 1000 10 10000 &
    v_T v_V v_K v_E ave window 2 file vket.dat
compute rdist all rdf 400                            # Get the radial distribution function
fix rdist_fix all ave/time 1000 10 10000 &
    c_rdist[*] ave window 2 &
    mode vector file rdist.dat
fix nvt_fix2 all nvt temp 200.0 300.0 0.1            # Perform NVT simulation (first annealing)
run 50000                                            # Run the simulation to melt
unfix nvt_fix2
fix nvt_fix3 all nvt temp 300.0 1800.0 0.1           # Perform NVT simulation (second annealing)
run 500000                                           # Run the simulation to melt
unfix nvt_fix3
print "Simulation complete"                          # Simulation finished
```

Tasks:

(1) Increase the temperature from 300 K to 1800 K at a rate equal to 3×10^{12} K/s. Plot the total energy as a function of temperature then estimate the melting temperature for the particle. Show the radial distribution function for the initial and final states.
(2) Change the diameter of the particle and see if the melting temperature change.
(3) Repeat the same analysis for a cooling process from 1800 K to 300 K at different cooling rates. See if the particle can be crystallized from the melted structure. If so, estimate the solidification temperature.

Note that the radial distribution function (RDF) or pair correlation function describes how density varies as a function of distance from a reference particle. It is a measure of the probability density of finding particle i at a given distance from particle j, and written as follows:

$$g_{ij}(r) = \sum_{i \neq j}^{N} \delta\left(r - \left|r_i - r_j\right|\right) \tag{7.14}$$

The radial distribution function is usually determined by calculating the distance between all particle pairs and binning them into a histogram. It can also be determined experimentally, by radiation scattering techniques such as X-ray diffraction, neutron diffraction or by direct visualization for large enough (micrometer-sized) particles via traditional or confocal microscopy. In this example, we are going to use this quantity as a measure for the order of molecular systems. For a crystalline solid, it would just be a set of delta spikes.

7.6. Example 5: Sputtering of Carbon onto Si(100)

In this example, the sputtering of carbon atoms onto silicon (100) surface at room temperature $T = 298$ K is going to be presented. The silicon substrate has a diamond structure with the lattice constant of 5.431 Å.

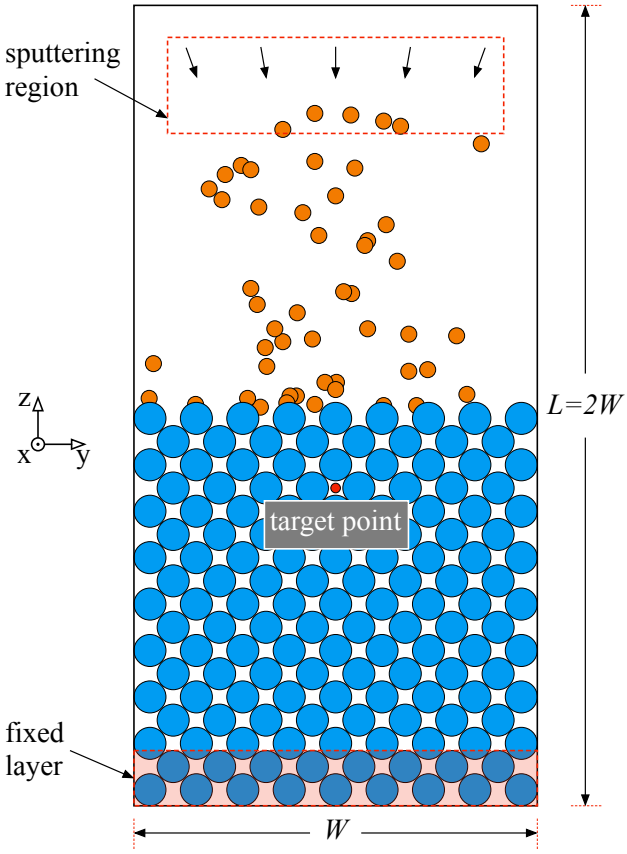

Figure 7.5. Sputtering of Carbon onto Si(100)

The interactions of silicon-silicon, silicon-carbon and carbon-carbon are described by the Tersoff potential without ZBL universal screening. The potential file *SiC_1989.tersoff* can be obtained from the potential folder of LAMMPS source code.

The bottom of the silicon slab is made rigid by setting the force on atoms in two bottom layer to zero so as to avoid the drift of the sample. Before inserting carbon atoms, the silicon substrate is relaxed and thermalized. The sputtering rate (rate of insertion of carbon) is set to 5×10^{15} atoms/s. Note that, we need to ignore lost atoms during the simulation because the sputtering atoms might fly out of simulation box. Below is the input file for this example:

Input file (ex5_sputter.in):

```
# Molecular Dynamics Simulations with LAMMPS
# Example 5: Sputtering of Carbon onto Si(100)
# ----------------------------------------

units metal                                          # Unit type: Metal
dimension 3                                          # Specify boundary conditions
boundary p p f
atom_style atomic                                    # Define classical particles
lattice diamond 5.431                                # Specify lattice structure (Diamond Si)
region whole block 0 6 0 6 0 12                      # Specify simulation box with two atom types
create_box 2 whole
region lbox block 0 6 0 6 0 6                        # Define slab region in simulation box
create_atoms 1 region lbox                           # Create silicon atoms on slab region
region sput block 0.5 5.5 0.5 5.5 10.0 11.5          # Define sputtering region in simulation box
region bot block INF INF INF INF 0.0 0.5             # Make the bottom of the silicon slab rigid
group fbot region bot
group other subtract all fbot
mass 1 28.09                                         # Set the masses of all atoms (Si and C)
mass 2 12.01
pair_style tersoff                                   # Use Tersoff potential for interaction
pair_coeff * * SiC_1989.tersoff Si C
neighbor 0.5 bin                                     # Build neighbor list
neigh_modify delay 1 check yes                       # Frequency to rebuild neighbor list
fix freeze fbot setforce 0.0 0.0 0.0                 # Set force on atoms at bottom region to zero
min_style cg                                         # Relax the SiC substrate
minimize 1e-8 1e-8 10000 10000
reset_timestep 0                                     # Specify the timestep for integration
timestep 0.001
thermo 5000                                          # Frequency print out ensemble data
thermo_style custom step pe ke etotal temp vol press # Specify what to print to screen & log file
dump coords all atom 1000 dump.sputter               # Export atomic positions for visualization
dump_modify coords sort id
velocity other create 298.0 12345 &                  # Create initial velocities for other atoms
         mom yes rot yes dist gaussian
fix nvt_fix all nvt temp 298.0 298.0 10.0            # Perform NVT simulation (thermal relaxation)
run 20000                                            # Run the simulation to reach equilibrium
fix sput_fix all deposit 500 2 500 12345 &           # Setup sputtering of carbon on silicon
    region sput near 1.2 vz -0.5 -1.5 &
    target 16.0 16.0 28.0 units box
thermo_modify lost ignore flush yes                  # Ignore lost atoms (fly out)
run 100000                                           # Run the simulation to sputter
unfix sput_fix
unfix nvt_fix
print "Simulation complete"                          # Simulation finished
```

Tasks:

(1) Run NVT simulation with time-step $\Delta t = 1$ fs and sputtering rate is set to 5×10^{15} atoms/s to visualize the sputtering process. Measure the carbon thickness.
(2) Increase the temperature and/or change the sputtering rate, repeat the simulation and see what happens to the sputtering process.
(3) Change the orient the lattice along different crystallographic planes and sputter. What happens? Is exposed surface important?

References

[1] B. J. Alder and T. E. Wainwright, *J. Chem. Phys.* 27 (1957) 1208
[2] B. J. Alder and T. E. Wainwright, *J. Chem. Phys.* 31 (1959) 459
[3] A. Rahman, *Phys. Rev.* A136 (1964) 405
[4] F. H. Stillinger and A. Rahman, *J. Chem. Phys.* 60 (1974) 1545
[5] R. Car and M. Parrinello, *Phys. Rev. Lett.* 55 (1985) 2471
[6] L. Verlet, *Phys. Rev.* 159 (1967) 98
[7] C. W. Gear, *Math. Comp.* 21 (1967) 146
[8] G. J. Martyna and M. E. Tuckerman, *J. Chem. Phys.* 102 (1995) 8071
[9] J. E. Lennard-Jones, *Proc. R. Soc. Lond. A* 106 (1924) 463
[10] R. A. Buckingham, *Proc. R. Soc. Lond. A* 168 (1938) 264
[11] M. S. Daw and M. I. Baskes, *Phys. Rev. B* 29 (1984) 6443
[12] S. M. Foiles, M. I. Baskes, and M. S. Daw, *Phys. Rev. B* 33 (1986) 7983
[13] M. I. Mendelev, S. Han, D. J. Srolovitz, G. J. Ackland, D. Y. Sun, and M. Asta, *Philos. Mag.* 83 (2003) 3977
[14] S. M. Foiles, M. I. Baskes, C. F. Melius, and M. S. Daw, *J. Less-Common Met.* 130 (1987) 465
[15] J. E. Angelo, N. R. Moody, and M. I. Baskes, *Model. Simul. Mater. Sci. Eng.* 3 (1995) 289
[16] M. I. Baskes, X. Sha, J. E. Angelo, and N. R. Moody, *Model. Simul. Mater. Sci. Eng.* 5 (1997) 651
[17] M. I. Baskes, *Phys. Rev. Lett.* 59 (1987) 2666
[18] M. I. Baskes, J. S. Nelson, and A. F. Wright, *Phys. Rev. B* 40 (1989) 6085
[19] T. J. Lenosky, B. Sadigh, E. Alonso, V. V. Bulatov, T. D. Rubia, J. Kim, A. F. Voter, and J. D. Kress, *Modeling Simul. Mater. Sci. Eng.* 8 (2005) 825
[20] Y. Mishin, M. J. Mehl, and D. A. Papaconstantopoulos, *Acta Mater.* 53 (2005) 4029
[21] F. Stillinger and T. A. Weber, *Phys. Rev. B* 31 (1985) 5262
[22] J. Tersoff, *Phys. Rev. B* 37 (1988) 6991
[23] J. Tersoff, *Phys. Rev. B* 39 (1989) 5566
[24] M. V. R. Murty and H. A. Atwater, *Phys. Rev. B* 51 (1995) 4889
[25] T. Kumagai, S. Izumi, S. Hara, S. Sakai, *Comp. Mater. Sci.* 39 (2007) 457
[26] H. J. C. Berendsen, J. P. M. Postma, W. F. van Gunsteren, A. DiNola, and J. R. Haak, *J. Chem. Phys.* 81 (1984) 3684
[27] H. C. Andersen, *J. Chem. Phys.* 72 (1980) 2384
[28] S. Nosé, *Mol. Phys.* 52 (1984) 255
[29] S. Nosé, *J. Chem. Phys.* 81 (1984) 511
[30] W. G. Hoover, *Phys. Rev. A* 31 (1985) 1695
[31] W. G. Hoover, *Phys. Rev. A* 34 (1986) 2499
[32] M. Parrinello and A. Rahman, *Phys. Rev. Lett.* 45 (1980) 1196
[33] M. Parrinello and A. Rahman, *J. Appl. Phys.* 52 (1981) 7182
[34] R. W. Hockney, S. P. Goel, and J. W. Eastwood, *J. Comp. Phys.* 14 (1974) 148
[35] S. Plimpton, *J. Comp. Phys.* 117 (1995) 1
[36] D. Frenkel and B. Smit, *Understanding Molecular Simulations*, Academic Press (2002)
[37] M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids*, Clarendon Press (1991)
[38] D. C. Rapaport, *The Art of Molecular Dynamics Simulations*, Cambridge University Press (1995)
[39] M. Shimono, *Molecular Dynamics*, Chapter 17, Part E in *Springer Handbook of Materials Measurement Methods*, Handbook, H. Czichos, T. Saito, L. Smith (Eds.), Springer (2006)
[40] F. Ercolessi, *A Molecular Dynamics Primer*, International School for Advanced Studies SISSA/ISAS, Trieste (1997) - http://www.fisica.uniud.it/~ercolessi/md/md.pdf
[41] E. Carlon, M. Laleman and S. Nomidis, *Computational Physics: Molecular Dynamics Simulations*, Lecture Notes (2016) - http://itf.fys.kuleuven.be/~enrico/Teaching/molecular_dynamics_2015.pdf

Class schedule (10 hours):

Lecture 1 (2 hours): Equations of Motions, Integration Algorithms

Lecture 2 (2 hours): Molecular Dynamics Ensembles, Temperature and Pressure Controls

Lecture 3 (2 hours): Interatomic Potential, Acceleration Techniques

Lecture 4 (2 hours): Introduction to LAMMPS, Example

Lecture 5 (2 hours): More LAMMPS Examples