Homework 2: Molecular Dynamics and the Verlet Algorithm

Kory Beach
Department of Physics, Astronomy, and Applied Physics,
Rensselaer Polytechnic Institute
beachk2@rpi.edu

November 21, 2016

1.1 Introduction

In an age of rapidly improving computational capabilities, it has become increasingly practical and necessary to perform simulations to study phenomena in complex physical systems. Molecular dynamics is a technique that can be used to obtain useful information about hypothetical systems by applying classical Newtonian equations and the principles of statistical mechanics to a large system of interacting particles.

In this report we will discuss the implementation of this technique by means of a Velocity Verlet algorithm to a two-dimensional system of particles interacting with a Lennard-Jones potential. The efficacy and computational implementation of this method, as well as the physical significance of the results, will be explored.

1.2 Methods

1.2.1 The Lennard-Jones Potential

The Lennard-Jones potential an approximate model for a short-range two-particle interaction characterized by counteracting repulsive and attractive terms:

$$V_{\mathrm{LJ}} = 4\varepsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right] \tag{1.1}$$

where r is the inter-particle distance, $-\varepsilon$ is the minimum energy of the interaction, and σ is the empirically obtained distance at which $V_{LJ} = 0$. The first term of this equation describes the extremely strong and extremely short-range Pauli exclusion force, and the second takes the form of a dipole-dipole attraction between particles. Because it is governed primarily by a dipole interaction, the systems that exhibit this behavior are usually composed of the noble gases, as stronger Coulombic forces are not present to dominate dynamics.

If we now consider a system of N interacting particles, the corresponding force between particles two i and j is given by

$$\mathbf{F_{ij}} = \frac{48\varepsilon}{r_{ij}^2} \left[\left(\frac{\sigma}{r_{ij}}\right)^{12} - \frac{1}{2}\left(\frac{\sigma}{r_{ij}}\right)^{6} \right] \mathbf{r_{ij}}, \tag{1.2}$$

meaning that for any given particle i, the total force that it experiences is given by the sum of all contributions by surrounding particles:

$$\mathbf{F_i} = \frac{48\varepsilon}{\sigma^2} \sum_{i \neq j}^{N} (\mathbf{r_i} - \mathbf{r_j}) \left[\left(\frac{\sigma}{r_{ij}}\right)^{14} - \frac{1}{2}\left(\frac{\sigma}{r_{ij}}\right)^{8} \right] \tag{1.3}$$

which by Newton's second law gives an expression for the acceleration $\mathbf{g_i} = \mathbf{F_i}/m$ where m is the mass of one particle.

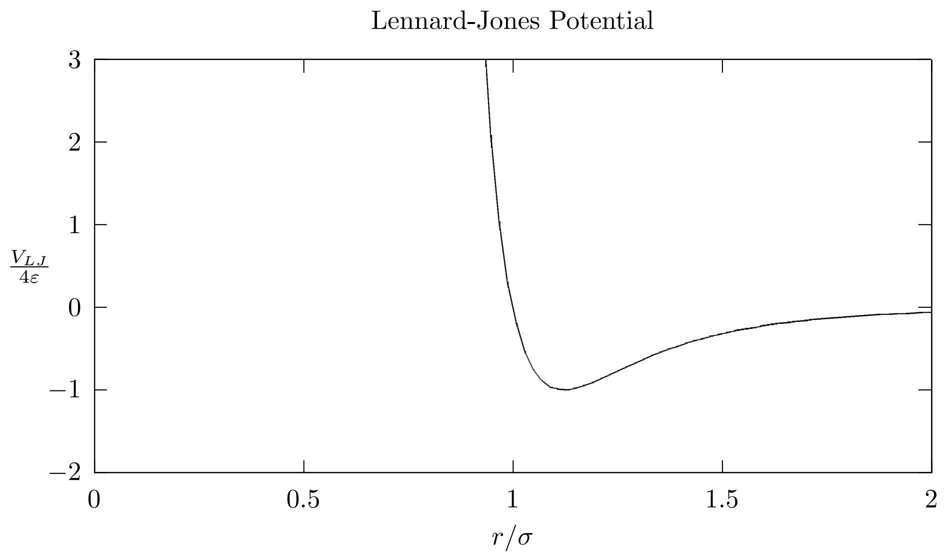

Figure 1.1: Plot of the Lennard-Jones interaction as a function of inter-particle distance.

When working with this potential computationally it is useful to convert to a system of dimensionless units given in terms of the constants of the system. In our case we will take the following conventions:

[energy] $= \varepsilon$	[length] $= \sigma$	[force] $= \varepsilon/\sigma$
[time] $= \sigma\sqrt{m/\varepsilon}$	[velocity] $= \sqrt{\varepsilon/m}$	[temperature] $= \varepsilon/k_B$

meaning that every variable can be replaced by the given unit times that variable. For example, $r \to r\sigma$ and $T \to T\varepsilon/k_B$. When we rewrite our units in this manner, we obtain a dimensionless form for the acceleration of particle i, which will govern the dynamics of the simulation:

$$\mathbf{g_i} = 48 \sum_{i \neq j}^{N} (\mathbf{r_i} - \mathbf{r_j}) \left(\frac{1}{r_{ij}^{14}} - \frac{1}{2r_{ij}^{8}} \right). \tag{1.4}$$

1.2.2 The Velocity Verlet Algorithm

In order to simulate the dynamics of a physical system we must first discretize the equations of motion for the particles in the system. The Velocity Verlet algorithm does this by applying a finite difference approximation to the Newtonian kinematic equations. If we start with the two-point formlua form of the velocity that evolves the velocity with some acceleration over a timestep, we have

$$\mathbf{v_i^{k+1}} = \mathbf{v_i^k} + \Delta t \mathbf{g_i^k} + \mathcal{O}(\Delta t^2) \tag{1.5}$$

where Δt is the timestep between iterations k and $k+1$ and $\mathbf{g_i^k}$ is the instantaneous acceleration felt by particle i at position $\mathbf{r_i^k}$. Since this is an iterative approximation, a slight improvement to the acceleration can be made by taking the average acceleration between iterations:

$$\mathbf{v_i^{k+1}} = \mathbf{v_i^k} + \frac{\Delta t}{2}\left(\mathbf{g_i^k} + \mathbf{g_i^{k+1}}\right) + \mathcal{O}(\Delta t^2). \tag{1.6}$$

Moreover, the positions can be updated upon each iteration by using the Newtonian kinematic equation for constant acceleration (i.e. the instantaneous acceleration over the small timestep Δt),

$$\mathbf{r_i^{k+1}} = \mathbf{r_i^k} + \Delta t \mathbf{v_i^k} + \frac{\Delta t^2}{2}\mathbf{g_i^k} + \mathcal{O}(\Delta t^4). \tag{1.7}$$

With this formulation– namely, Equations 1.4, 1.6, and 1.7 – we have a recipe to evolve an interacting system of particles forward in time. In the algorithm's simplest form, all that it then requires is a set of initial position $\{\mathbf{r^k}\}$ coordinates and an inital distribution of particle velocities $\{\mathbf{v^k}\}$.

1.2.3 Intitialization

Since both Equations 1.6 and 1.7 provide information about iteration $k+1$ based on the velocity and position at step k, it is necessary to provide the particles with sensible initial parameters that allow the algorithm to start.

For the initial particle postions, a simple square lattice in two dimensions is sufficient for initialization. If we consider a system with N particles, where N is ideally a perfect square, we can define a variable $n_{xy} = \sqrt{N}$ to be the number of rows or columns of particles in a box of dimensions LxL. The lattice constant a in this square configuration is then given by L/n_{xy} , which allows us to populate the space in the following manner:

```
for (int i = 0; i < N; i++){
        x[i] = (i%nxy) * a;                 // x-position coordinates
        y[i] = (floor(i / nxy)+1) * a;      // y-position coordinates
}
```

where every index i indicates a different particle in the system and we have used the C++ function "floor" to round down so as to indicate the row on which the particles lie in the box.

Initializing the velocities of the particles is also of importance, for it is necessary to have a statistically valid distribution in order for the dynamics to make sense. For this we can use the Maxwell-Boltzmann velocity distribution, which describes the way in which components of particle probability are distributed in an ideal gas, and which takes the form

$$f_v(v_i) = \sqrt{\frac{m}{2\pi kT}}\exp\left(\frac{-mv_i^2}{2kT}\right) \tag{1.8}$$

where v_i is a single Cartesian component of the velocity vector. By generating pseudorandom numbers, in our case by means of a Box-Muller algorithm by Numerical Recipes [1], within this distribution we can separately assign an initial value for each of the velocity vector components.

1.2.4 Boundary and Interaction Conditions

Once the particle velocities and positions are initialized, it is necessary to set up boundary conditions that accurately reflect the physical system we wish to simulate. For our system of particles interacting with a Lennard-Jones potential we can impose periodic boundary conditions such that the LxL box of particles is repeated infinitely in all directions.

The first challenge when implementing this condition is to make sure that particle number is conserved within the box– that is to say that whenever a particle exits one side of the box it should enter again from the opposite side. This is most easily done by updating the position coordinate after every iteration with a normalizing term that brings the particle to the corresponding position in the original box. In C++, we can apply the following operation after every modification of the position coordinate based on Equation 1.7:

```
x[i] = x[i] - L*floor(x[i] / L);
```

where we have subtracted from the x-coordinate of particle i the integer number of Ls required to bring it back into the center box.

In addition to this position coordinate correction, it is also necessary to account for the fact that, even when two particles are both within the center box, there are two possible interaction distances per axis that must be considered: the separation through the center of the box and the separation through the border. This can be addressed in the following manner:

```
deltax[i][j] = deltax[i][j] - L*round(deltax[i][j] / L);
```

where deltax[i][j] (Δx_{ij}) is initially the distance between the x-coordinates of particles i and j through the center of the box and the C++ function "round" gives 0 if $0 \leq \Delta x_{ij} < L/2$ and 1 if $L/2 \leq \Delta x_{ij} \leq L$, hence choosing the shorter of the two possible interaction distances.

It is worth noting that although it is necessary to calculate the relative separations between particles in the manner shown above, in the interest of reducing computation time, we can choose to omit sufficiently distant pairs when calcuating the acceleration. This can be done by defining some cutoff threshold beyond which the interaction force is set to 0. In the case of the Lennard-Jones potential, it is sufficient to set a cutoff around $r_{cut} = 3\sigma$, since the dipole-dipole interaction is short-range.

1.2.5 Ensembles and Velocity Correction

In order to develop a physically meaningful model for the interaction dynamics of a Lennard-Jones system, we must stop to consider our model in the context of statistical ensembles. Without the presence of an external temperature reservoir, the temperature is allowed to fluctuate in the system but the total energy remains constant. This is known as a microcanonical ensemble, or an NVE ensemble, where the conserved quantity (within an algorithmic range of error)

$$E_{tot} = \sum_i^N \frac{1}{2} m v_i^2 + \sum_{i \neq j}^N V_{LJ}^{ij} \tag{1.9}$$

can be easily computed with the velocities and positions of the particles every few iterations.

A more physically realistic system is one in which energy is not necessarily conserved but the temperature is held constant by an external reservoir. This is known as a canonical (NVT) ensemble, and adjusting our model to such a system requires that we consider the relationship between temperature and kinetic energy in a statistical system. In the thermodynamic system, we can say that in general,

$$E_K = \sum_i^N \frac{1}{2} m v_i^2 = \frac{G}{2} k_B T \tag{1.10}$$

where G is the total number of degrees of freedom, which in our case would be $G = 2N - 4$ because it is a two-dimensional system.

Here we will describe two different methods for maintaining an isothermal system– we can either (1) rescale the velocities by a constraining constant derived from Equation 1.10 or (2) we can add a frictional term to the acceleration by means of the Nose-Hoover method.

(1) Rescaling

If we wish to correct the temperature after every few iterations to ensure that the system remains relatively close to isothermal, one method we can use is to multiply the velocity of every particle by some scaling factor λ. If we define our target speed per particle to be the root-mean-squared speed given by the thermodynamic parameters of the system,

$$v_{rms} = \sqrt{\frac{GkT}{m}}, \tag{1.11}$$

then we can make the assumption that for some λ

$$v_{rms} = \lambda \sqrt{\sum_i^N v_i^2} = \lambda \sqrt{v_s} = \sqrt{\frac{GkT}{m}} \tag{1.12}$$

which yeilds an expression for λ:

$$\lambda = \sqrt{\frac{GkT}{m v_s}} \tag{1.13}$$

where v_s is the sum of the squares of all the velocities in the system.

One drawback of this method is that it rests on the initial assumption that the velocity distribution remains relatively uniform, which is not always true in such systems. One potential problem that arises from this method comes from the fact that the same scaling factor is applied to all the velocities. If a single velocity is an outlier that dwarfs those of the other particles, λ will approach zero, causing the rest of the particles to stop moving so as to keep the global temperature constant. Nonetheless, on relatively short timescales, this method is sufficient for examining the dynamics of the system.

(2) The Nose-Hoover Method

Another method of keeping the temperature in check is to include a friction term to the acceleration. In the Nose-Hoover algorithm, we can rewrite the acceleration in the following manner,

$$\mathbf{g_i} = \frac{\mathbf{F_i}}{m} - \eta \mathbf{v_i} \tag{1.14}$$

so that at every step the acceleration is reduced by a term proportional to the velocity by a temperature-dependent frictional parameter η. The Nose-Hoover coefficient is generally given by its time derivative

$$\dot{\eta} = \frac{1}{M_s}\left(\sum_i^N \frac{\mathbf{p_i^2}}{m} - Gk_BT\right) \tag{1.15}$$

where M_s is called the thermal inertial parameter which is tunable and should be on the order of Gk_BT. This formulation of the Nose-Hoover coefficient can be easily incorporated into the Velocity Verlet algorithm by means of a forward difference approximation:

$$\eta^{k+1} = \frac{\Delta t}{M_s}\left(\sum_i^N \frac{\mathbf{p_i^2}}{m} - Gk_BT\right) + \eta^k \tag{1.16}$$

where the inital value of η can be set to 0.

1.3 Implementation

This molecular dynamics simulation of a system of Lennard-Jones particles was implemented using C++ in Visual Studio 2013. Various parameters for the temperature, the number of particles, and the size of the box were used.

One of the significant challenges in the computational implementation was dealing with explosive potentials for instances when, due to the discretization of the timestep, particles could find themselves instantly on top of each other and feeling an extremely powerful repulsive force. This unphysical occurance can lead to velocities rapidly getting out of control and must therefore be carefully addressed.

Two parameters of significance were adjusted to solve this problem: first, a minimum interaction radius threshold of 0.9σ was introduced below which the repulsive forces remained constant; second, the timestep was reduced to $.001\sigma\sqrt{m/\varepsilon}$. With these changes, the explosive potentials no longer occurred because the particles were able to approach each other more continuously, and even when they did perchance overlap, the repulsive forces were curtailed by the cutoff. Because the Lennard-Jones repulsion increases so rapidly at radii below 1.0σ it is a good approximation to set a cutoff at 0.9σ, as the particles should generally never even reach such a threshold in a truly physically accurate system.

Various statistics for the simulation were recorded, including the total energy of the system, the pair correlation function, and the mean square displacement of the particles. All of these statistics, as well as the particle positions were outputted to text files and processed through Gnuplot. The progression of the particles through time could be viewed in .gif form through the "gif" terminal in Gnuplot.

1.4 Results and Discussion

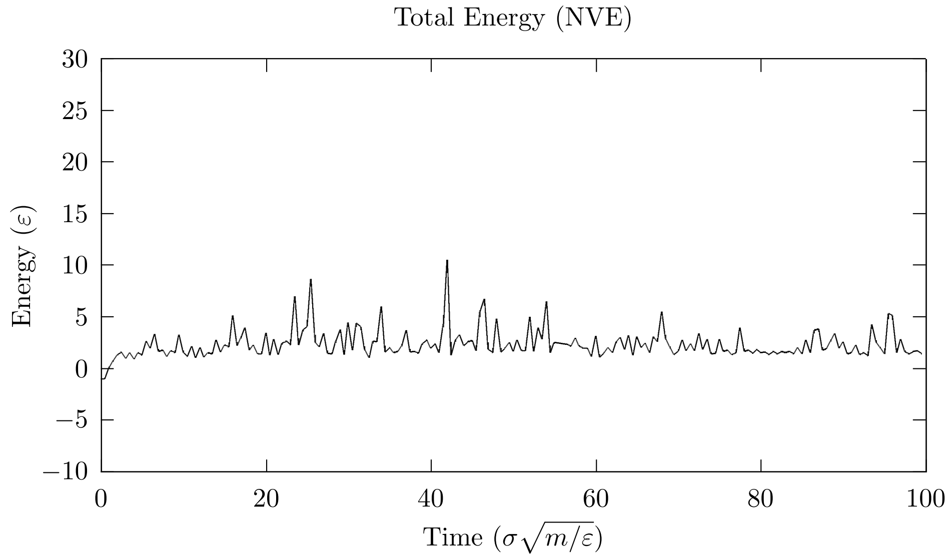

Figure 1.2: Sum of potential and kinetic energy in the microcanonical ensemble for 100 particles.

Before implementing the canonical (NVT) ensemble, the dynamics of the

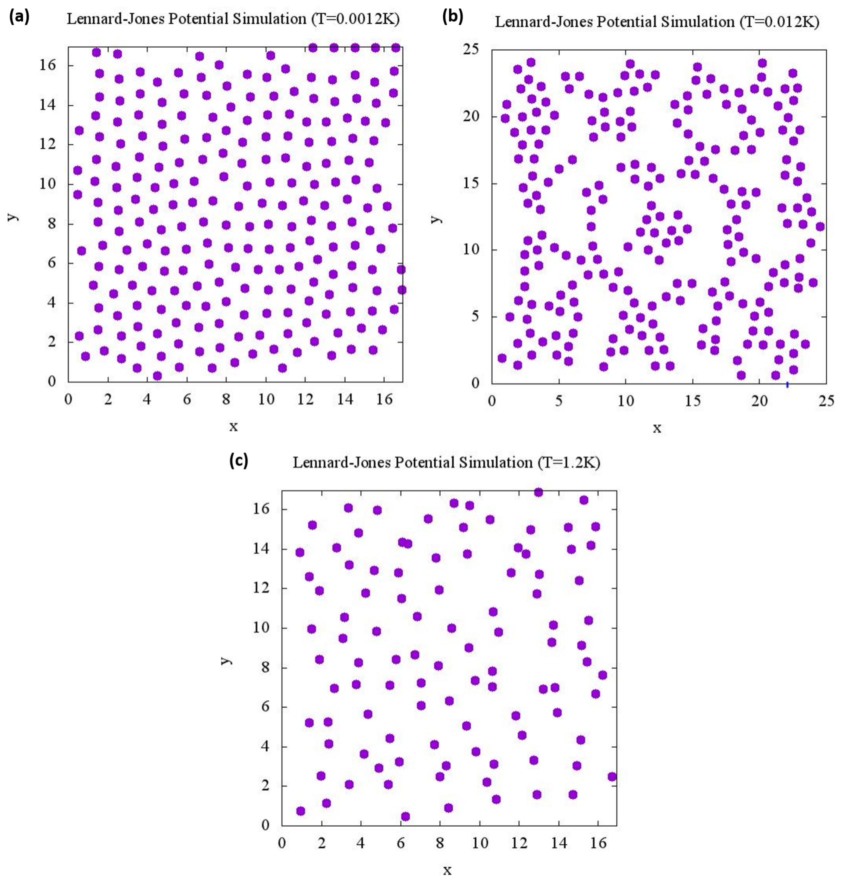

Figure 1.3: Snapshots of (s) solid, (b) liquid, and (c) gas phases of the Lennard-Jones system. Axes are in units of σ. Temperature shown is calculated using known parameters for σ and ε in Argon.

Lennard-Jones system were tested for conservation of energy. As shown in Figure 1.2, the system demonstrated relatively constant energetics over the course of a substantial period of time. The total energy in this plot is calculated at every timestep by adding the kinetic and potential energies of all particles in the system. The fluctuations in the energy can be explained by the fact that the potential energy is calculated up to a cutoff radius, allowing for instantaneous fluctuations in the local density of particles to slightly affect the total energy.

Once the dynamics of the microcanonical ensemble were shown to be functional, the Nose-Hoover algorithm was implemented to fix the temperature, simulating introduction of the system to an isothermal heat bath. Various combinations of temperature and number density were studied, and in the end three distinct phases were observed in the system. As shown in Figure 1.3, after the systems were allowed to evolve into steady states, the particles assumed different phases for different temperatures.

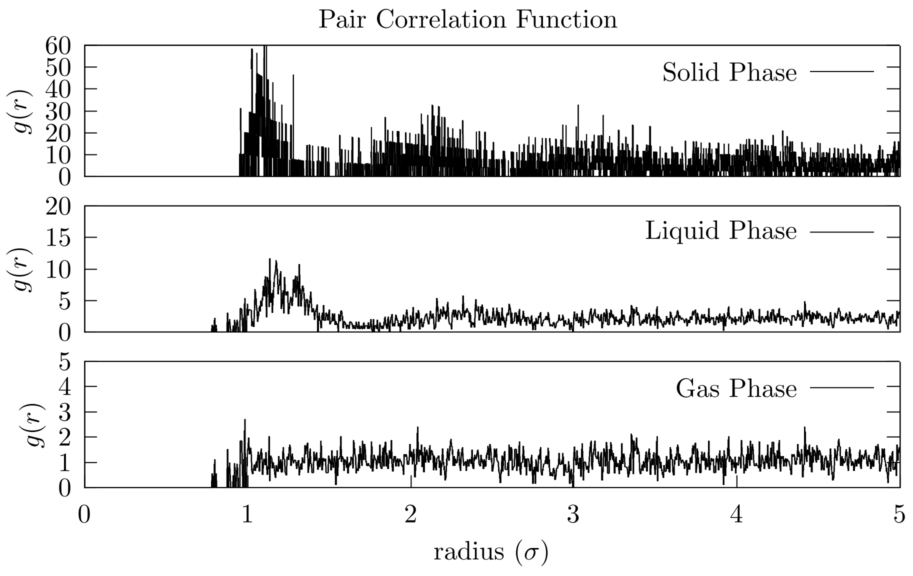

Figure 1.4: Pair correlation ($g(r)$) for solid, liquid and gas phases respectively. The solid phase shows clear periodicity of the lattice while this effect is less prominent in the latter phases. The normalization of $g(r)$ is dimensionless and arbitrary.

Visually, this is obvious in the snapshots shown. In the solid state, the particles assume a sturcture close to a hexagonal closed packing with an inter-atomic distances of about 1.1σ, which corresponds to the energy minimum of the Lennard-Jones potential. In the liquid state, this hexagonal structure is not quite reached, but clustering of the particles occurs. In the gaseous phase, the particles are noticeably dispersed and interact less frequently.

These phases can be further probed by measuring the pair correlation functions and the mean square displacements of the systems. The pair correlation, which is a measure of how far away on average pairs of particles are with respect to each other, is shown in Figure 1.4, revealing a clear difference in behavior between the three phases. It is important to note that for pair correlation calculations, the system must be allowed to evolve for a sufficiently long time such that any inital periodicity of the starting configuration has been "forgotten" by the dynamics of the system. In the Lennard-Jones systems simulated, the solid

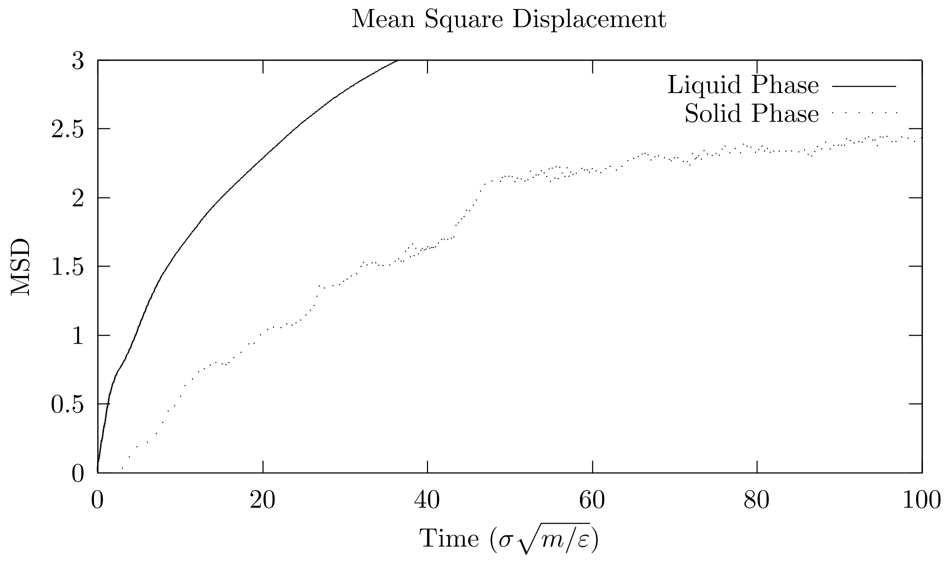

Figure 1.5: Mean square displacement (MSD) in units of σ for sold and gas phases.

phase shows clear periodicity in the lattice, with separation between the peaks at about 1.1σ, as expected based on visual analysis of the system. Similarly, the peaks in the liquid phase show some periodicity for closer-range pairs, but this dies off much more quickly than the highly structured solid phase. In the gaseous phase, there is a slight peak around the Lennard-Jones potential minimum, indicating that interacting particles were slightly more likely to exist in that energy valley, but since the temperature was so high, the kinetic energies of the particles vastly dwarfed the potential for any periodicity in the system.

In addition to the pair correlation, the mean square displacement (MSD), which is a measure of the average displacement of the particles in a given system as a function of time, gives the expected results for the solid and gas phases. As shown in Figure 1.5, the solid phase levels off as the system stabilizes as a function of time, and the MSD of the liquid phase increases roughly linearly with time after an initial nonlinear increase.

1.5 Conclusion

While this analysis of the dynamics of Lennard-Jones systems is far from conclusive, we have shown that a molecular dynamics simulation in a canonical ensemble with temperature moderated by either velocity rescaling or the Nose-Hoover algorithm accurately predicts the possibility of three different phases of matter at different temperatures. These results were confirmed both by visually examining the dynamics of the system, and by examining pair correlation and MSD statistics.

Bibliography

[1] Press, William H. 1992. Numerical recipes in C: the art of scientific computing. Cambridge: Cambridge University Press.

[2] Meunier, Vincent. Advanced Computational Physics Lecture Slides; Lectures 5 and 6: Molecular Dynamics