Project 5 FYS4150

Vilde Mari Reinertsen and Kjetil Karlsen

December 10, 2017

Abstract

Molecular dynamics is a numerical method that can be used to study many-particle systems of molecules, clusters and macroscopic systems as solids, liquids and gases, such as in this project. It is used in different science fields, physics, chemistry and biology. In this project we simulated a system of argon atoms by implementing a program with many classes and a complicated class structure. The movement was calculated with the Velocity Verlet method.

We saw how the system was a solid at certain temperatures and melted to a liquid at higher temperatures. We used diffusion through displacement of the atoms and visualized the system in Ovito to find the melting temperature [1]. We found the melting temperature to be around 270-280 K but some low diffusion constants above these temperatures might indicate some strange behaviors in our model. We also found the diffusion constant to be on the order of $\sim 10^{-9}$ cm^2/s for the solid phase and $\sim 10^{-5}$ cm^2/s for the liquid phase, and saw how the total energy stayed conserved while exchanging between potential energy and kinetic energy in the start.

Contents

1 Introduction

Molecular dynamics is a numerical method to simulate the movement of atoms and molecules based on classical Newtonian dynamics. The method can be used to study many-particle systems of molecules, clusters and macroscopic systems as solids, liquids and gases, such as in this project. It is used in different science fields. It can be used in material science with non-relativistic Schrödinger equation to investigate lattice and defect dynamics [2]. In chemistry it can be used to see the behaviour of ideal gases and chemical reactions with hard sphere potential and Lennard Jones potential [3]. It can be used to simulate behavior at membranes in biology, fluctuations of biological marcomolecules and to determine protein structures from NMR [4].

In this project we have a program with many classes and we have used a lot of time to get to know the structure of the classes and the flow of the program. The program simulates a system of 500 atoms and uses Lennard Jones potential to calculated the forces between them. The Velocity Verlet method was used to solve Newtons second law and give the motion of all atoms.

The report first explain the relevant theory. Thereafter, the classes of the program are described and the flow of the program presented. After that, the result of the energy, temperature and diffusion are disclosed together with a discussion of it. At last some concluding remarks are made.

2 Theory

2.1 Crystal structure

There are different ways a solid could arrange itself in order to fill a given volume and minimize the overall energy of the structure. One of the densest is the face-centered cubic (FCC) structure of for instance NaCl. Assuming a hard sphere model of atoms, the FCC has together with the hexagonal close packed (HCP) the theoretically highest density of any structure. The unit cell of FCC is cubic with volume $V_{unitCell} = a^3$ made up by a basis of 4 atoms. These are in positions $a(0,0,0)$, $a(0,0,\frac{1}{2})$, $a(0,\frac{1}{2},0)$ and $a(\frac{1}{2},0,0)$. Due to symmetry, each of these atoms are repeated throughout the structure, as illustrated in figure 2.1. The corner atoms are repetitions of the $a(0,0,0)$ position, and each of these atoms are shared by 8 neighbouring cells. For the side atoms, they are shared only by two unit cells, giving at total of $6 \cdot \frac{1}{2} = 3$ side atoms per unit cell and $8 \cdot \frac{1}{8}$ corner atoms in the unit cell. The density of a unit cell size given in lengths of Angstrom, the density of a FCC cell is $\rho = \frac{4}{a^3}\text{Å}^{-3}$.

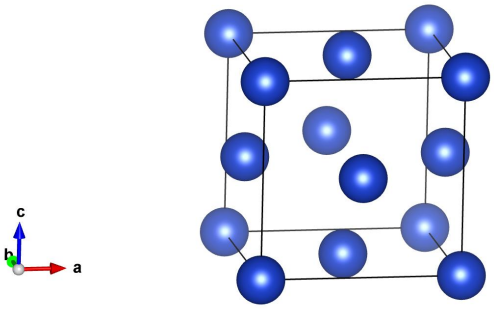

Figure 2.1: Illustration of a fcc structure. Although there are 14 atoms in this figure, the unit cell is made up of 4 atoms. The rest of the atoms appear because of translation symmetry. Illustration is created by the VESTA software [5]

2.2 Lennard Jones potential

A very popular potential used in MD-simulations is the Lennard Jones potential, given in equation 1. It describes the potential energy as a function of distance between two objects i and j, r_{ij}.

$$U(r_{ij}) = 4\epsilon \left[\left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^{6} \right] \tag{1}$$

Through the relation $F(\mathbf{r} = -\nabla U(\mathbf{r})$, it is possible to determine the forces experienced by each atom by every other atom, simply as a function of the distance r_{ij} and relative position between two atoms. The distance r_{ij} is defined as $r_{ij} = |\mathbf{r}_{ij}| = |\mathbf{r}_i - \mathbf{r}_j|$. By investigating each spatial direction separately, a relation for the forces between two atoms i and j arises:

$$F_{ij}^{x} = -\frac{\partial U}{\partial r_{ij}} \frac{\partial r_{ij}}{\partial x_{ij}} \tag{2}$$

$$\frac{\partial r_{ij}}{\partial x_{ij}} = \frac{\mathrm{d}}{\mathrm{d}x_{ij}} \sqrt{(\mathbf{r}_i - \mathbf{r}_j)^2} = \frac{x_{ij}}{r_{ij}} \tag{3}$$

$$\frac{\partial U}{\partial r_{ij}} = 4\epsilon \left[\frac{-12}{r_{ij}} \left(\frac{\sigma}{r_{ij}} \right)^{12} + \frac{6}{r_{ij}} \left(\frac{\sigma}{r_{ij}} \right)^{6} \right] \tag{4}$$

$$F_x(r_{ij}) = 24\epsilon \left[2 \left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^{6} \right] \frac{x_{ij}}{r_{ij}^2} \tag{5}$$

By generalising equation 5 to a three dimensional space one gets a general expression of the force between two atoms i and j:

$$\mathbf{F}(r_{ij}) = 24\epsilon \left[2 \left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^{6} \right] \frac{\mathbf{r}_{ij}}{r_{ij}^2} \tag{6}$$

In figure 2.2 both the potentate and force is illustrated, clearly showing that the repulsion from the $\left(\frac{\sigma}{r_{ij}}\right)^{12}$ term is very strong at low values of r_{ij}. For larger r_{ij} both terms quickly goes towards 0. However, the potential display a minimum before it goes towards zero, determined by σ. For distances smaller than σ, the force is positive and thus repulsive and for $r_{ij} > \sigma$ is shows attractive properties. The forces is never truly 0 before $r_{ij} = \infty$. but for most practical purposes it is mainly the distances $r_{ij} \in [0, 10]$ that contribute to the total force on each atom.

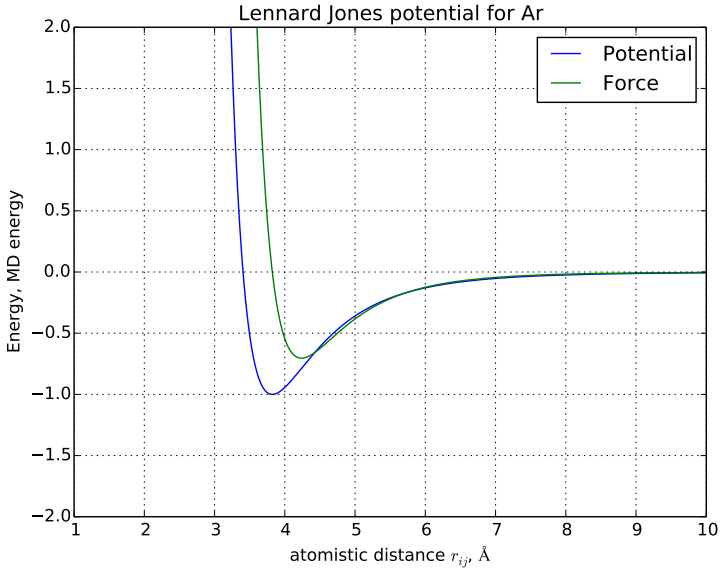

Figure 2.2: Illustration of the Lennard Jones potential. See section 3.4.1 for definition of the MD units

Atom i experience a total force of $F(\mathbf{r}_i) = \sum_{j \neq i} \mathbf{F}(r_{ij})$. Each interaction is only necessary to compute once, as $\mathbf{r}_{ij} = -\mathbf{r}_{ji}$. However, it is necessary to update the force on each atom, according to Newtons third law. Equivalently, the total potential is given as $U_{tot} = \sum_i \sum_{j>i} U(\mathbf{r}_{ij})$.

2.3 The microcanonical ensemble

This subsection is based on M. Jensen [6].

In statistical physics, the microcanonical ensemble is a system with constant volume, particles and total energy. This means that it is an isolated system which does not exchange energy or particles with the surroundings, but are allowed to vary the pressure and temperature. As with any ensemble in statistical physics, the equilibrium is reached when the system is near the most probable state. As Maxwell-Boltzmann statistics governs the probability of each microstate, the equilibrium is dependent on both temperature and energy of the state. In the microcanonical ensemble, temperature T and pressure P are defined through the number of possible microstates Ω as:

$$S = k_B \log \Omega \tag{7}$$

$$\frac{1}{T} = \left(\frac{\partial S}{\partial E}\right)_{N,V} \tag{8}$$

$$p = k_B T \left(\frac{\partial \log \Omega}{\partial V}\right)_{N,E} \tag{9}$$

At equilibrium, every microstate has the same total energy. This leads to the equipartition theorem, stating that it every microstate is equally probable and will be at some point occupied. One example of a microstate not conforming to this is the microstate where all the atoms are located in a sphere of very low radius - giving an infinite potential, see the discussion about the Lennard Jones potential in section 2.2.

Through the equipartition theorem, the following relationship for a 3 dimensional system arises between the total kinetic energy of the system and the temperature:

$$E_K = \frac{3}{2} N k_B T \tag{10}$$

As the temperature at equilibrium is related to the kinetic energy, any system system resembling a real system needs to have kinetic energy >0. The pure lattice points of the FCC lattice has a high degree of symmetry and the force on each atom would for a infinite lattice cancel out to 0. It is therefore necessary to include an initial velocity in order to model the structure at different temperatures, due to the equipartition relation in equation 10. As the total kinetic energy does not provide any information about the velocity of each atom, the probability distribution utilized for the microcanonical ensemble is a good way to initialize the system to progress into a new, random microstate in the next time step.

By including an initial velocity, there is no guaranty that the total momentum $p = \sum_i^N m_i \mathbf{v}_i$ is zero. In order to keep the system fixed in space, the total momentum need to be set to zero, by reducing the momentum of each atom by $m_i v_i - \frac{p}{N}$.

2.3.1 Diffusion and phase transition

Diffusion is a property of any microcanonical ensemble, characterized by the diffusivity D. Through the Einstein relation

$$\langle r^2(t) \rangle = 6Dt \tag{11}$$

the diffusivity is related to the mean square displacement $r_i^2(t) = |r_i(t) - r_i(o)|^2$. The diffusivity is approximately 10^-5 cm^2/s in liquids and 10^-9 cm^2/s for in solids, for example hydrogen in iron [7].

2.4 Density

The density of argon is 0.001784 g/cm^3 at STP when gas and 1.3954 g/cm^3 at the boiling point when liquid [8]. Table 2.1 compare these numbers with the system in this project. Experimentally argon has a melting temperature at 84 K and a boiling temperature at 87 K at 1 atm [8]. The densities implies that even though argon has only 3 degrees in liquid the state at 1 atm, the density of our system is too high for it to transit to gas.

Table 2.1: This table list the calculation of the density and compare the density of liquid argon, argon gas and our system. We used the molar mass of argon, 39.948 g/mol [8].

	calculation:	density $\left[\frac{\#\text{ of atoms}}{\text{cm}^3}\right]$:
Gas:	$\frac{0.001784\ \text{g/cm}^3}{39.948\ \text{g/mol}} = 4.46580{\cdot}10^{-5}\ \frac{\text{mol}}{\text{cm}^3} =$	$2.6893{\cdot}\ 10^{19}$
Liquid:	$\frac{1.3954\ \text{g/cm}^3}{39.948\ \text{g/mol}} = 0.0349304\ \frac{\text{mol}}{\text{cm}^3} =$	$2.10350{\cdot}\ 10^{22}$
This system:	$\frac{4}{a^3} = \frac{4}{5.26^3} = 0.027485\ \frac{\#\text{ of atoms}}{\text{Å}^3} =$	$2.74854 \cdot 10^{22}$

3 Method

3.1 Code structure

3.1.1 Overview of the classes

The code provided for the project were made up of several classes, each with different roles. To a large extent there is a hierarchy in the classes, with *main* being on the top. From this class all main processes are initiated, initial conditions are determined, and the integration loop is run. However, it is the class *atom* that facilitate the object in which to store information about each atom in the calculation. Under follows a quick overview of each class and the main purpose of it.

- *system*:
  Contains all the information about the system. Creates a system of atom-instances in a fcc lattice. It also contains the integration function, linking the system to *VelocityVerlet*. In addition, functions that are relevant to the behaviour of the system, like applying periodic BC's and calculation of forces are found here.
- *atom*:
  Contains information about each atom in the simulation, like starting position, position, velocity and force experienced by each atom. Is mainly a class to store information about each atom, but also contains a function to calculate the initial velocity distribution.

- *VelocityVerlet*:
  Contains simply the velocity verlet algorithm (per timestep). Takes the entire system class as an input variable and updates position, velocity and force on each atom in the system. Uses functions from *system* in order to update forces and utilize periodic BC's.

- *LennardJones*:
  When calling the *calculateForces* function in *system*, it links to this object. Updates forces on each atom in the system and the total potential, through the Lennard Jones potential.

- *statisticssampler*:
  Collect information about the system, like kinetic and potential energy, instantaneous temperature. In addition, also contains a unit test checking energy convergence.

- *vec3*:
  Defining the behaviour of every 3×1 vector utilized in the program. Also contains functions to calculate length of *vec3* objects, cross products, nullifying a *vec3* instance. Every vector in the code is a *vec3* instance.

- *unitconverter*:
  Contains information about constants and units used in the program, in addition to functions converting values between MD-units and SI units.

- *io*:
  Framework to create and store information to a *.xyz* file.

- *random.h*:
  Header file containing a selection of functions from the c++ library *random* and *ctime*.

3.1.2 Program flow

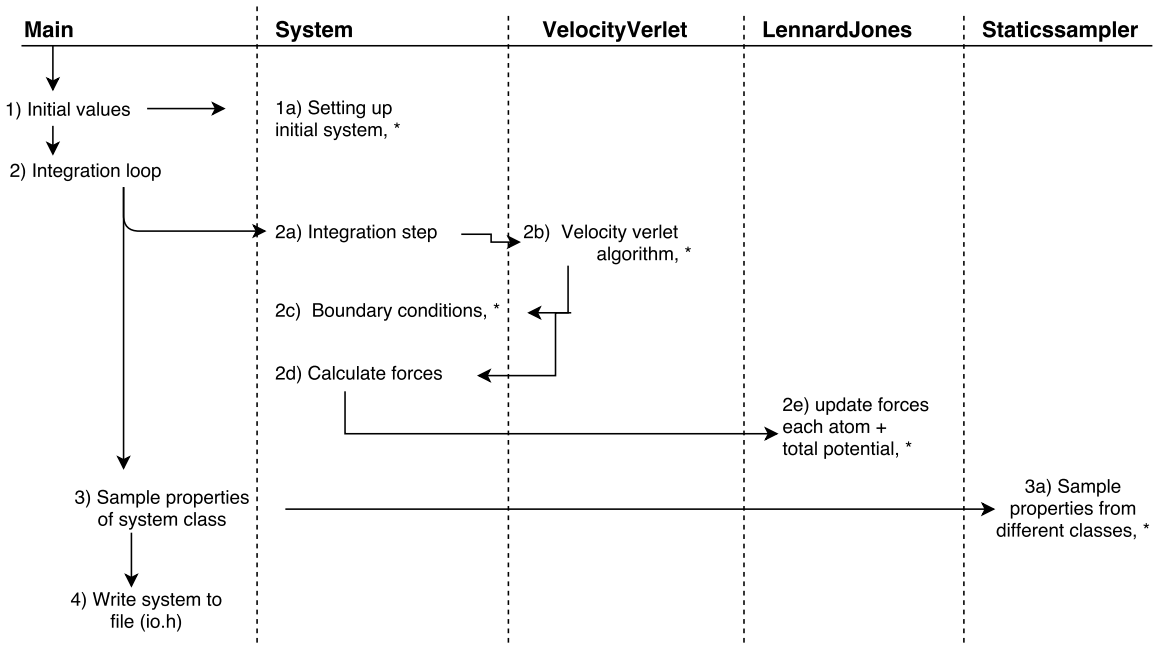

Figure 3.1: Simplified flow chart of the code structure. * indicates that the Atom class was used, see the discussion below

Figure 3.1 illustrates the work flow in the code somewhat simplified. Due to the amount of classes in this project, every time the *atom* class is called is marked by *. As *system* contains several atoms, each time *atom* is called is by a loop over all the atoms in *system*, updating i.e. the position. To some extent, all the classes can be viewed as a blackbox, with *main* controlling the sequence and what is being performed.

Step 1)specifies the initial values, like temperature and number of unit cells, sending them to step 1a, which creates the fcc lattice as specified by main. In addition, it also creates instances of *io* and *StatisticsSampler* and initialize the output files. It also creates the specified amount of *atom*-instances. In order to integrate the system through time, a time loop is performed in *main*. However, each integration step is then micromanaged by *system* through an instance of *VelocityVerlet*, which does the actual integration over every instance of *atom* in *system*. Through the instance "system" the periodic boundary conditions are applied and forces are calculated, see snippet below.

Listing 1: Snippet showing the main part of velocity verlet algorithm as applied in *VelocityVerlet*

```
double dthalf = dt*0.5;
for(Atom *atom : system.atoms()) {
 atom->velocity += dthalf*atom->force/(atom->mass());
 atom->position += atom->velocity*dt;
 //atom->position += atom->velocity*dt/atom->mass();
}
system.applyPeriodicBoundaryConditions();
system.calculateForces();

```

```
10 for(Atom *atom : system.atoms()) {
11  atom->velocity += dthalf*atom->force/(atom->mass());
12  }
```

The sampling of properties are done in step 3, by applying the "sample" function in *StatisticsSampler*, 3a. This function updates the private variables of *StatisticsSampler*, describing properties like temperature and kinetic energy. This is done by calling several specific *StatisticsSampler* - functions sampling each property individually. The last step, step 4, writes .xyz-file through the *io* instance "movie" in *main*. In addition, the information stored in *StatisticsSampler* is saved in a separate output file. This should be done in *io* for the most logical flow, but we found it simplest to do this through *StatisticsSampler*.

3.2 Periodic Boundary Conditions

In numerical calculations it is not possible to calculate on a infinite lattice of atoms. As an infinite lattice have neighbouring atoms in every direction for every atom, an atom can not move into vacuum. By applying periodic boundary conditions (PBC's) to the position of the lattice, every atom moving out of the predefined supercell is put on the other side of the box. In one dimension, this is expressed as $r_{xi} = r'_{xi} - L$ if $r_{xi} > L$, with L being the supercell size. This ensures that the density is constrained and the volume of the supercell remain constant. Applying periodic boundary condition make it possible to simulate an infinitely big system with few atoms, making the results relevant for real systems which contain $\sim 10^{23}$ atoms. The real systems do not contain infinitely many atoms, but the infinite case is a better approximation than just 500 atoms as would be the case in this project if we did not apply periodic boundary conditions.

A surface experience different potential compared to bulk, as there are no neighbouring atoms in one of the directions, giving rise to different physics at the surface compared to the bulk. As infinite lattices does not have a surface, the surface effect needs to be minimized. By implementing the minimum image convention these effects can be reduced. This convention states that if a distance r_{ij} between two atoms i and j is larger than $\frac{L}{2}$, the distance between them is $r'_{ij} = r_{ij} - L$. Thus an atom near an edge experiences forces of approximately equal magnitude to an atom near the centre of the supercell.

Trodde vi trengte image-greia fordi vi har periodic boundary conditions og et atom som er på andre siden av unit cella er da egentlig rett ved siden av (burde vel egentlig være begge deler, men når den er på andre siden så er den så langt borte at det ikke har noe å si). Så det som står her er egentlig et argument for periodic boundary conditions og image-greia er noe man må legge til for at periodc bounadry conditions funker som det skal?

When we calculated the diffusivity we used the mean square displacement. To find the true displacement like we did not have periodic boundary conditions, we needed to keep track of how the atoms had moved back and forth due to the conditions. We

saved a vector, for every atom, containing the length of the supercell, with the right sign, in the direction it had moved, updated it in the function for periodic boundary conditions and subtracted it when calculating the displacement.

The PBC's and minimum image convention models an infinite supercell, by ignoring surface effects. However, it is not a true representation of an infinite cell. A perfect infinite cell consists of an infinite number of particles with random initial velocities, while this project's finite supercell will only have a finite number of atoms, with sudo-random[1] initial velocities. These atoms interact, leading to a stronger codependence of the neighbours for the finite case. By increasing the supercell this effect will decrease.

3.3 Velocity Verlet

This project used the Velocity Verlet algorithm in order to integrate the partial differential equations which arises from the newtons second law. The Velocity Verlet method can be derived from a Taylor expansion around the position $r(t + \Delta t)$. However, for the velocity there exist a similar Taylor series: $v_{i+1} = v_i + h\dot{v}_i + \frac{h}{2}\ddot{v}_i + O(h^3)$. Unfortunately, there is no expression for $\ddot{v}_i$, but it can be approximated by $h\ddot{v}_i \simeq \dot{v}_{i+1} - \dot{v}_i$. Adding this to the expression for v_{i+1} and doing some simple algebra one get the final expression for the velocity:

$$v_{i+1} = v_i + \frac{h}{2}\left(\dot{v}_{i+1} + \dot{v}_i\right) + O(h^3) \tag{12}$$

Looking a bit closer at equation 12, there is a immediate problem. In order to calculate v_{i+1} one need to already know v_{i+1}. This problem can be solved by updating the velocity in two steps, which gives the following algorithm:

$$v_{i+\frac{1}{2}} = v_i + \frac{h}{2}a_i \tag{13}$$

$$x_{i+1} = x_i + v_{i+\frac{1}{2}} \tag{14}$$

$$a_{i+1} = \frac{F_{i+1}}{m} \tag{15}$$

$$v_{i+1} = v_{i+\frac{1}{2}}. + \frac{h}{2}a_{i+1} \tag{16}$$

3.4 Bugs and unit tests

As the structure of the program was provided, a large array of unit tests were not necessary. However, two simple test became very useful. The first, checking that the total momentum was set to 0 provided useful input, as the initial version of "removeTotalMomentum" did not take the changing unit cell size into account. Secondly, the unit test for the conservation of energy revealed several mistakes. One example of this

[1] If a computer is used to generate the random initial velocities, repeated periodically. For the theoretical infinite lattice one usually assumes true random initial conditions.

is that we counted the potential energy per atom in interacting pair, instead of each pair. A more serious bug were found in *velocityVerlet* through this unit test. Line 5 in snippet 1 shows the original, first position update in the algorithm. Here the velocity of each atom is divided by the atom mass, which has no place in the relationship $\frac{dv}{dt} = x(t)$. Most likely this is due to a copy and paste error of the line above, simply substituting the force with velocity.

3.4.1 Units

In order to avoid round off errors and produce results with direct physical meaning, a specific set of units were used in the program. These molecular dynamics units are presented in the project description, and are:

$$\begin{aligned} 1 \text{ unit of mass} &= 1 \text{ a.m.u} = 1.661 \times 10^{-27}\text{kg}, && (17)\\ 1 \text{ unit of length} &= 1.0 \text{ Å} && (18)\\ 1 \text{ unit of energy} &= 1.651 \times 10^{-21}\text{J}, && (19)\\ 1 \text{ unit of temperature} &= 119.735\text{K}. && (20)\\ 1 \text{ unit of force} &= 1.65313 \cdot 10^{-11} N && (21)\\ & && (22) \end{aligned}$$

4 Result and Discussion

4.1 Initial temperature for melting

When looking at the simulation in Ovito, we found that the melting point seems to be around an initial temperature of 600 K [1]. We added the displacement of the atoms to the movie-file, so we could color code the displacement of the atom, to make it easier to see the melting. Figure 4.1 shows the supercell when it is clearly solid and Figure 4.2, Figure 4.3 and Figure 4.4 shows the development from the ordered initial FCC structure to the melted state. The red color represent a displacement of 6 Å or more, the blue a displacement near 0 Å and the green is a medium displacement, around 3 Å.

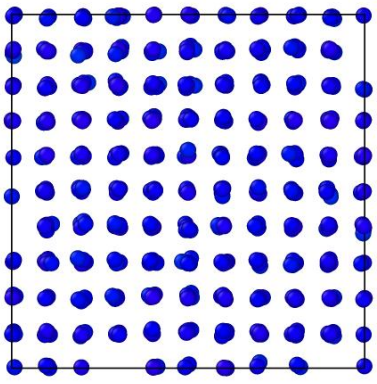

Figure 4.1: This is a snap shot after $3.0 \cdot 10^{-11}$ seconds of the supercell of argon with an initial temperature of 100 K. The structure is clearly solid, with some lattice vibration due to nonzero

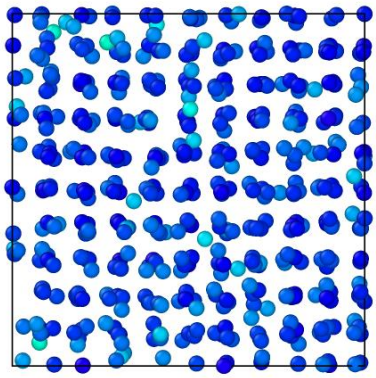

Figure 4.2: This is a snap shot after $1.2 \cdot 10^{-13}$ seconds of the supercell of argon with an initial temperature of 600 K. The structure looks solid in the beginning. The picture is form Ovito [1].

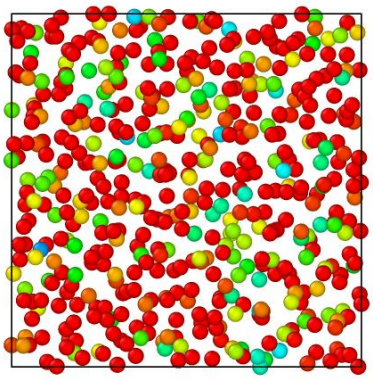

Figure 4.4: This is a snap shot after $5.5 \cdot 10^{-11}$ seconds of the supercell of argon with an initial temperature of 600 K. The structure has melted almost all atoms are displaced by 6 Å or more. The picture is form Ovito [1].

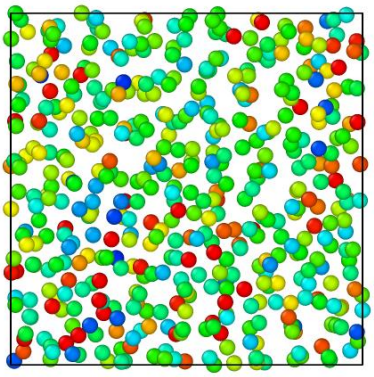

Figure 4.3: This is a snap shot after $2.3 \cdot 10^{-11}$ seconds of the supercell of argon with an initial temperature of 600 K. The structure is not solid, the displacement has increased. The picture is form Ovito [1].

4.2 Real melting temperature

The system starts in a very ordered state, so the potential energy is very low. The atoms were given random velocities from the Maxwell-Boltzmann distribution, so in the first time step the atoms have moved away from this high symmetric low potential state and the potential energy increases very much. Because the total energy has to be conserved, the kinetic energy start at a maximum and decreases a lot to compensate. The drop in kinetic energy gives a drop in temperature because of the relation between them (see Equation 10.). Equation 10 is only accurate in thermal equilibrium, considering that, the initial temperatures are not that relevant. Figure 4.5 shows the development of the temperature with time. The temperature is approximately half the initial temperature in equilibrium.

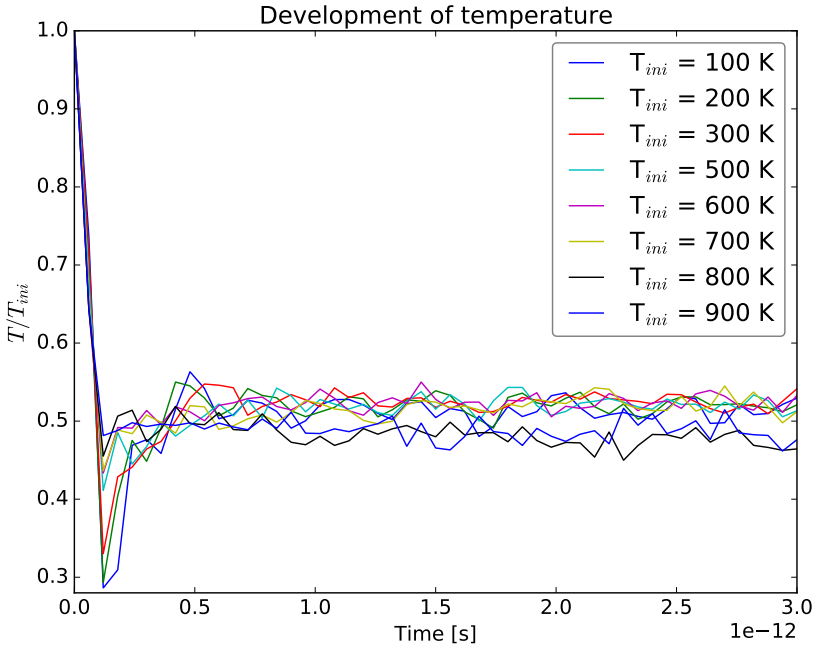

Figure 4.5: This is a plot of the ratio between the initial temperature and the temperature calculated from the kinetic energy (see Equation 10) to show how it decreases to an equilibrium which is approximately half the initial temperature.

As a consequence of the temperature development, we chose to use the average of temperature form the kinetic energy at the last 10 % of the time ($6 \cdot 10^{-12}$ s) to find the temperature at equilibrium.

4.3 Energy

Figure 4.6 shows the kinetic, the potential and the total energy of the system initially and how it reaches an equilibrium, where it fluctuate with a small amplitude. The figure shows how the potential starts in a minimum and the kinetic in a maximum. The total energy seems to be well converged.

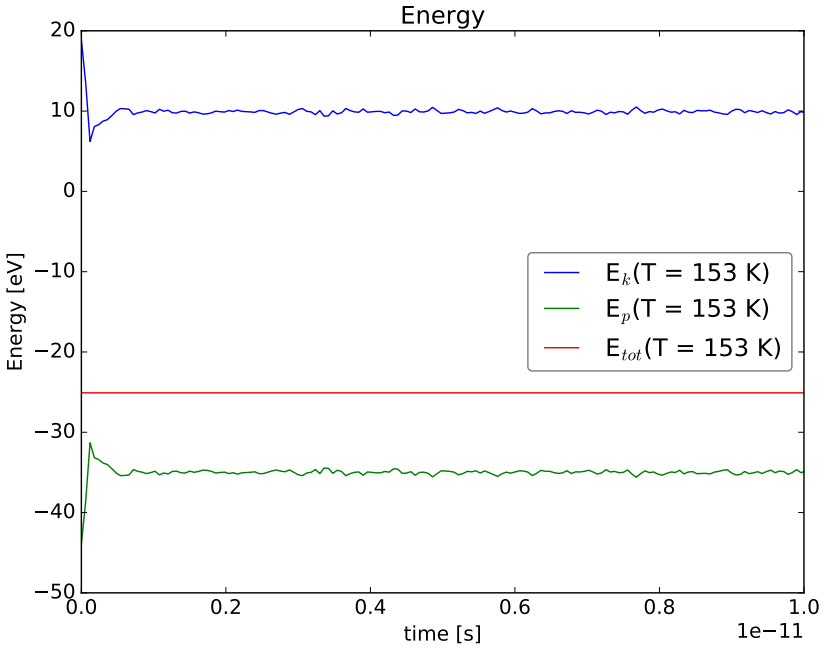

Figure 4.6: This is a plot of the energy at initial temperature 300 K and equilibrium temperature 153 K. The plot shows how the potential energy starts in a minimum and the kinetic in a maximum, and how the total energy is well conserved even though both the kinetic and potential fluctuates.

4.4 Diffusion constant

Figure 4.7 is a plot of the mean square displacement against temperature. According to the Einstein relation it should be a linear plot with slope $6D$, where D is the diffusion constant. We did a linear regression on the data and extracted the diffusion constant for different temperatures. The temperatures in the legend are the equilibrium temperatures.

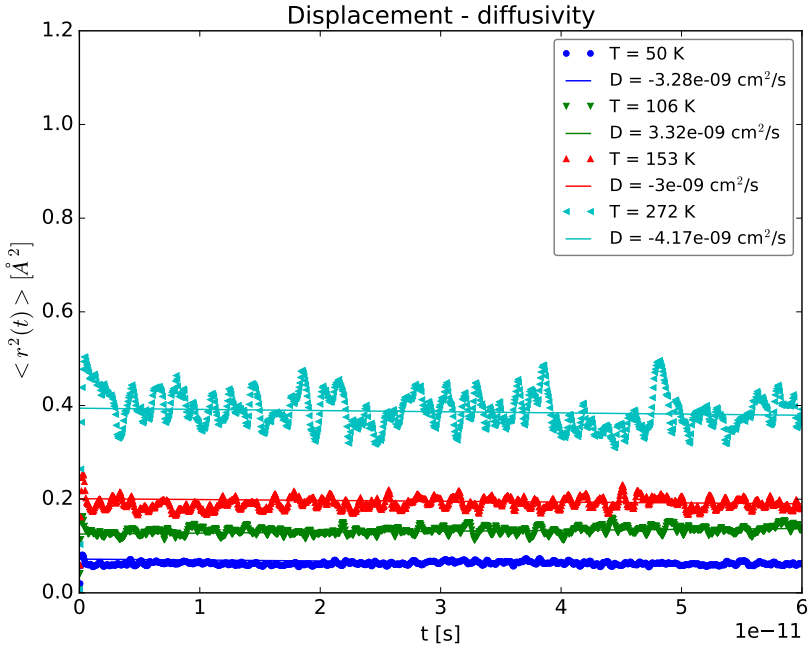

Figure 4.7: This is a plot of Einstein relation (see Equation 10), the diffusion constant is extraced from the slope of the linear regression. The temperatures are below the melting point.

Figure 4.8 is a plot of the mean square displacement against temperatures above the melting temperature. Because most of the data series has a kink and Equation 11 shows a linear relation, we chose to use the last half of the data to do the linear regression and extract the diffusion constant form that. We assumed that the equilibrium had to be reached and that it was reached after $3 \cdot 10^{-11}$ seconds.

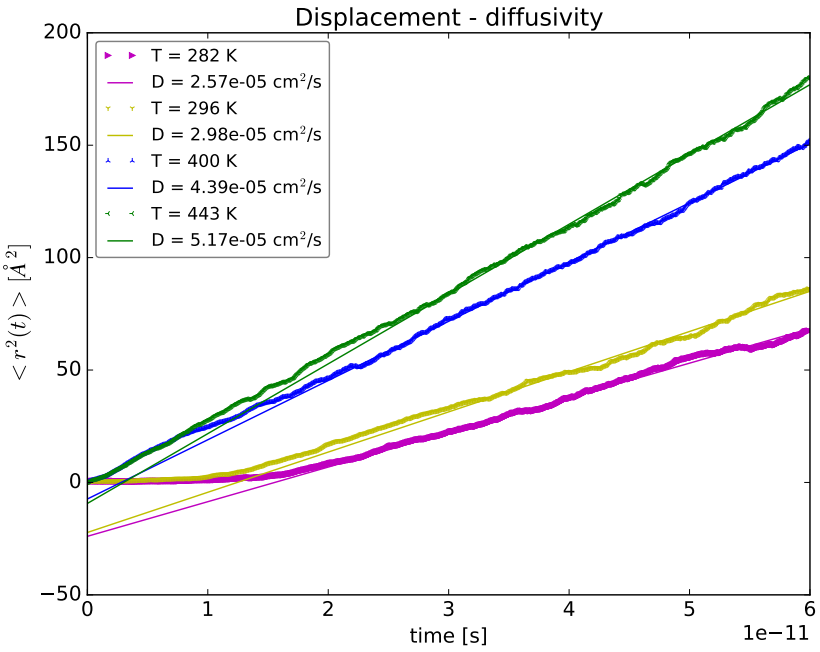

Figure 4.8: This is a plot of Einstein relation (see Equation 11), the diffusion constant is extraced from the slope of the linear regression. The temperatures are above the melting point.

The diffusion constants from Figure 4.7 and Figure 4.8 were plotted against temperature in Figure 4.9. The diffusivity makes a jump at around 300 K, implying that the melting temperature is around 300 K. The values before the jump is around 10^{-9} cm^2/s (see Figure 4.7) which match the values for solids and the values after the melting matches the values for diffusion in liquids. This might suggest that it is a transition from solid to liquid.

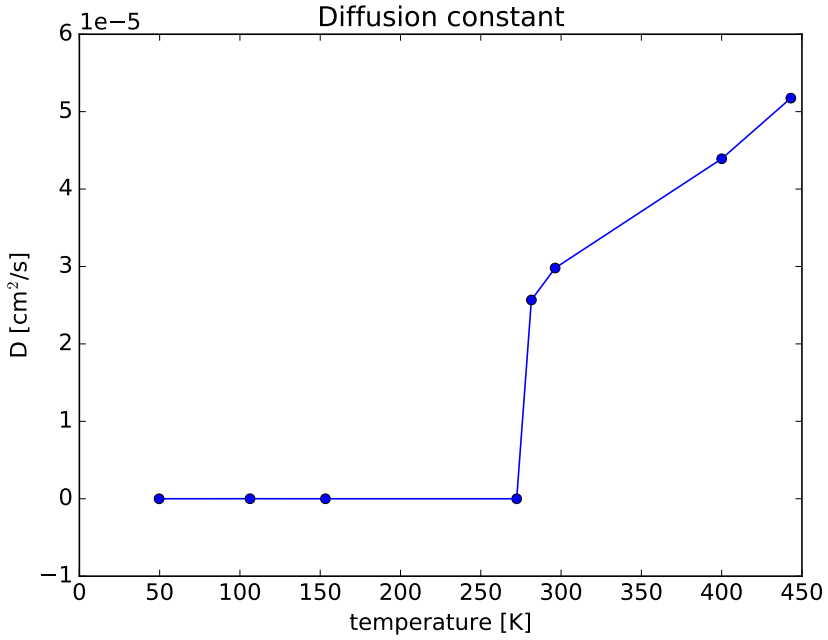

Figure 4.9: This is a plot of the diffusion constant with respect to temperature. The values are taken from the slope of the linear regression of the mean squared distance versus time because of Einsteins relation (see Equation 11). The plot shows how the diffusion constant increases drastically after the melting point around 300 K.

The highest temperature with the relatively low diffusion constant is at $T = 272$ K and the highest with the high diffusion constants are at $T = 282$ K. That implies that the melting temperature is in the interval 272 K to 282 K. The system behave strangely around the melting temperature though, as Figure 4.10 shows, so it was difficult to set a specific melting point.

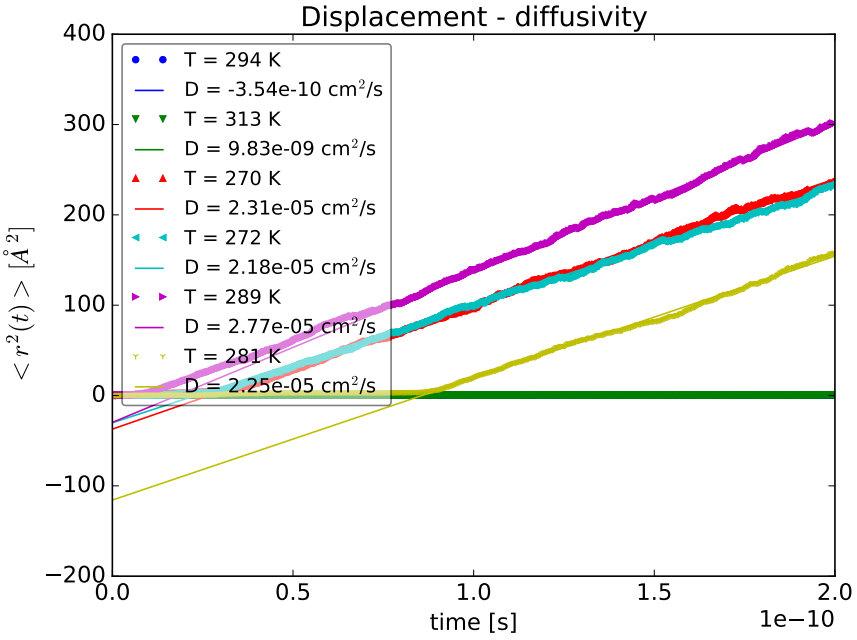

Figure 4.10: This shows the mean square displacement near the melting temperature showing some weird result, that the highest temperatures have the smallest slopes and then also the lowest diffusion constants. These runs were done for a longer time because of the temperature development around the melting point (see Figure 4.12).

The experimental melting temperature of argon is 84 K at 1 atm pressure, and thus a lot smaller than our result. The reason might be that the pressure is not 1 atm. Since we do not calculate the pressure, it is difficult to check our result with phase diagrams plotted from experiments. Phase transitions are dependent on both temperature and pressure, so our result might be accurate for another pressure.

4.5 Equilibrium temperature

Figure 4.11 shows the temperature development with different initial temperatures. The temperature development shows that the system with temperatures around melting point do not reach a proper equilibrium in the same timespan as the others. There is a shift after some time, making the results strange. Figure 4.12 shows the temperature development of system with initial temperatures that result in temperatures around the melting point. The program was run for a longer time span to get better diffusion constants (see Figure 4.10) because of the shift in temperature.

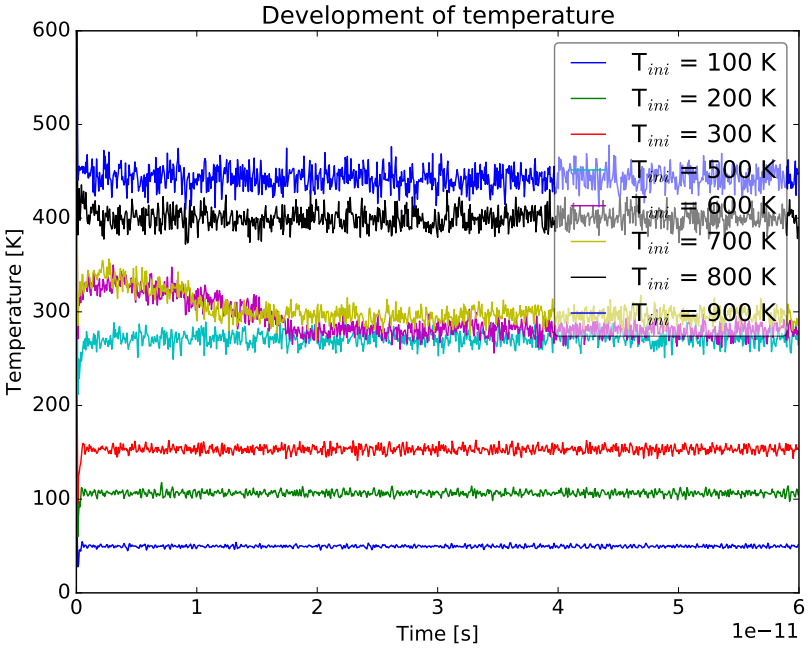

Figure 4.11: This is a plot of the temperature development of systems with different initial temperatures. The system with initial temperatures that end up around the melting temperatures (600 K and 700 K) makes a shift after a while.

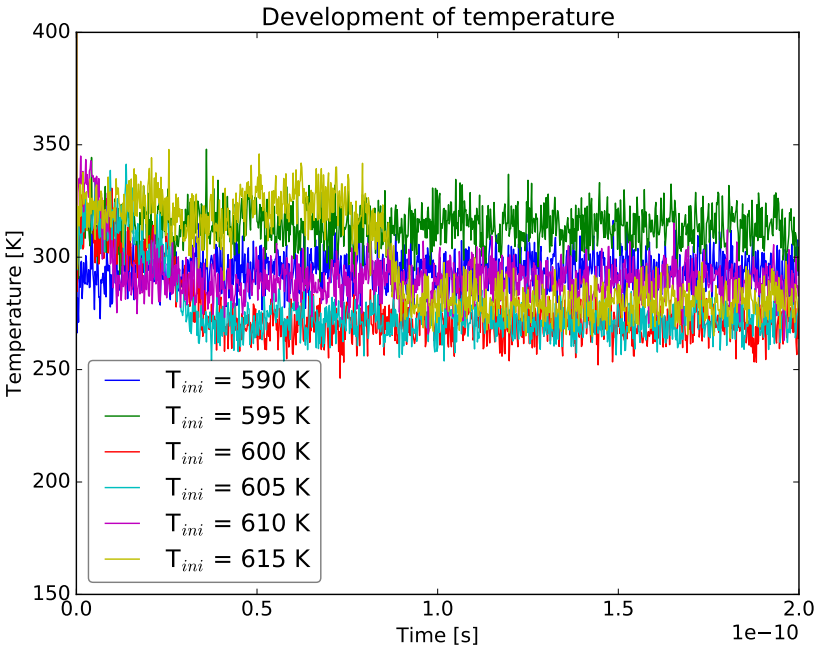

Figure 4.12: This is a plot of the temperature development of systems with different initial temperatures around the melting temperature. This might explain the strange numbers around the melting temperature.

5 Conclusion

Molecular dynamics is a numerical method that can be used to study many-particle systems of molecules, clusters and macroscopic systems as solids, liquids and gases. In this project we simulated a system of argon atoms by implementing a program with many classes and a complicated class structure. Some theory around diffusion and temperature calculations from kinetic energy via the equipartition theorem were presented. The class structure was explained and the advantages of periodic boundary conditions were discussed. We used Ovito and saw how the system was a solid at certain temperatures and melted to a liquid at higher temperatures [1] . We used diffusion through displacement of the atoms and visualized the system in Ovito to find the melting temperature.

We found the melting temperature to be around 270-280 K but some low diffusion constants above these temperatures might indicate some strange behaviors in our model or just consequences of randoms seeds in the initial velocity. We also found the diffusion constant to be on the order of $\sim 10^{-9}$ cm^2/s for the solid phase and $\sim 10^{-5}$ cm^2/s for the liquid phase. The sudden change in diffusivity was how we found a temperature interval for the melting point. We also saw how the total energy stayed conserved while exchanging between potential energy and kinetic energy in the start, implying that the Velocity Verlet method was conserving the energy.

Further on we could also calculate the pressure and maybe change it, to see how the melting point is dependent on the pressure. Then we could check the melting point result in this project with experimental values. By changing the density, we could probably see the phase transition to gas from liquid as well.

References

[1] A. Stukowski. Visualization and analysis of atomistic simulation data with OVITO - the Open Visualization Tool, 2010.

[2] Martin O Steinhauser and Stefan Hiermaier. A review of computational methods in materials science: Examples from shock-wave and polymer physics. *International Journal of Molecular Sciences*, 10(12):5135–5216, 12 2009.

[3] J. Gorecki and J. Gryko. Molecular dynamics simulation of a chemical reaction. *Computer Physics Communications*, 54(2):245 – 249, 1989.

[4] Martin Karplus and Gregory A. Petsko. Molecular dynamics simulations in biology. *Nature*, 347:631 EP –, 10 1990.

[5] K. Momma and F. Izumi. VESTA: a three-dimensional visualization system for electronic and structural analysis, 2008.

[6] Morten Hjorth-Jensen. Computational physics: Lecture notes fall 2015. Department of Physics, University of Oslo, 8 2015. Chapter 12 and 13.

[7] Mass diffusivity. https://en.wikipedia.org/wiki/Mass_diffusivity. Accessed: 2017-12-07.

[8] Argon. https://en.wikipedia.org/wiki/Argon. Accessed: 2017-12-07.

Appendix