Project 1
Molecular Dynamics
FYS-4460

Gullik Vetvik Killie

March 27, 2015

Contents

1 Part 1:

1.1 Task a: Creating a FCC lattice

This task is implemented in the function createFCCLattice()

The noble gas argon should have a stable lattice structure when a solid, by letting the system start in a stable situation we avoid a lot of energy to be infused into the systems temperature due to it minimizing potential energy. The implementation is done by going through several nodes, R_i, put on a grid in the box and then placing 4 atoms around each node. Let c_l be the length between nodes.

$$\mathbf{R}_{ij} = \mathbf{R}_i + \mathbf{r}_j \qquad j = \{1, 2, 3, 4\} \qquad i = \{1, 2, ..., 4\mathrm{N}_{\mathrm{atoms}}\}$$

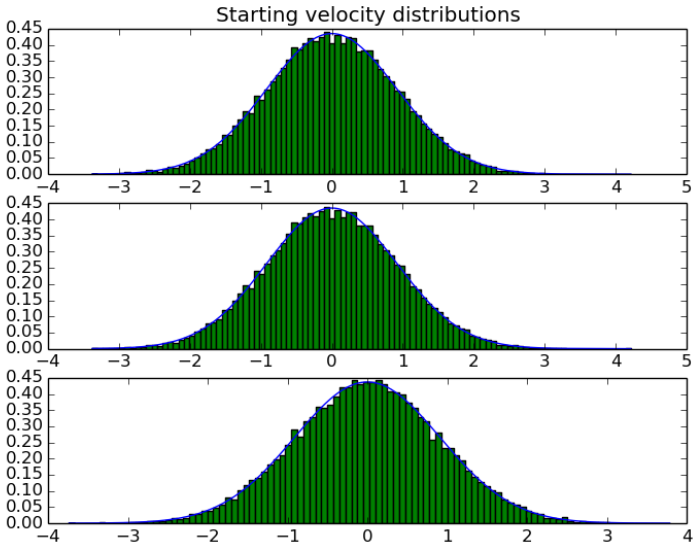


Figure 1: The velocites given to the atoms with a gaussian overlay

$$\mathbf{r}_1 = 0\hat{\mathbf{i}} + 0\hat{\mathbf{j}} + 0\hat{\mathbf{k}}$$
$$\mathbf{r}_2 = \frac{c_l}{2}\hat{\mathbf{i}} + \frac{c_l}{2}\hat{\mathbf{j}} + 0\hat{\mathbf{k}}$$
$$\mathbf{r}_3 = 0\hat{\mathbf{i}} + \frac{c_l}{2}\hat{\mathbf{j}} + \frac{c_l}{2}\hat{\mathbf{k}}$$
$$\mathbf{r}_4 = \frac{c_l}{2}\hat{\mathbf{i}} + 0\hat{\mathbf{j}} + \frac{c_l}{2}\hat{\mathbf{k}}$$

1.2 Task b: Gaussian Velocities

1.3 Task e: Computing forces

We will be using the Lennard-Jones potential to approximate the forces between the molecules, which work quite well, given it's simplicity, for neutral particles, especially noble gases, as we are dealing with in this study [1]. The formula is given below.

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

At short distances the term to the twelfth power dominates and represents a repulsive force, Paulie exclusion principle, while at longer distances the term to sixth power dominates and represents the attractive van der Waal force. The σ is the distance at which the potential is 0, while ϵ is the depth of the well. $\sigma = 3.405$Å and $\varepsilon/k_B = 119.8$K are chosen to fit the physical properties of the system for Argon.
The force felt between the molecules is given by the negative gradient of the potential.

$$\mathbf{F}(r_{ij}) = -\boldsymbol{\nabla} U(r_{ij})$$

The potential only has a nonzero derivative along $\mathbf{r}_{ij}$, the axis between then particles, so it is natural to evaluate the gradient in that coordinate system before projecting it onto the xyz coordinates used by the program.

$$\begin{aligned}
\mathbf{F}(r_{ij}) &= -4\epsilon \partial_{r_{ij}} \left[\left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^{6} \right] \hat{\mathbf{r}}_{ij} \\
\mathbf{F}(r_{ij}) &= -4\epsilon \left[-\frac{12}{r_{ij}} \left(\frac{\sigma}{r_{ij}} \right)^{12} + \frac{6}{r_{ij}} \left(\frac{\sigma}{r_{ij}} \right)^{6} \right] \hat{\mathbf{r}}_{ij} \\
\mathbf{F}(r_{ij}) &= 24\epsilon \left[2 \left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^{6} \right] \frac{\mathbf{r}_{ij}}{r_{ij}^2} \\
m_i \frac{\partial^2 r_i}{\partial t^2} &= \sum_j 24\epsilon \left[2 \left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^{6} \right] \frac{\mathbf{r}_{ij}}{r_{ij}^2}
\end{aligned}$$

1.4 f: NonDimensionality

Now we introduce some new units to get a nondimensional equation of motion. $\begin{cases} r & \to \sigma r' \\ t & \to \tau t' \end{cases}$

$$\begin{aligned}
m_i \frac{\partial^2 \sigma r_i'}{\partial (\tau t')^2} &= \sum_j 24\epsilon \left[2 \left(\frac{\sigma}{\sigma r_{ij}'} \right)^{12} - \left(\frac{\sigma}{\sigma r_{ij}'} \right)^{6} \right] \frac{\sigma \mathbf{r}'_{ij}}{\sigma^2 r_{ij}'^2} \\
\frac{\partial^2 r_i'}{\partial (t')^2} &= \frac{24\epsilon\tau^2}{m_i \sigma^2} \sum_j \left[2 \left(\frac{\sigma}{\sigma r_{ij}'} \right)^{12} - \left(\frac{\sigma}{\sigma r_{ij}'} \right)^{6} \right] \frac{\mathbf{r}'_{ij}}{r_{ij}'^2}
\end{aligned}$$

Choosing $\tau = \sigma\sqrt{m/\varepsilon}$ then gives the dimensionless equation

$$\frac{\partial^2 r_i'}{\partial (t')^2} = 24 \sum_j \left[2 r_{ij}'^{-12} - r_{ij}'^{-6} \right] r_{ij}'^{-2} \mathbf{r}'_{ij}$$

1.4.1 Algorithm to implement force

The implementation of the force will be along the following steps:

- Find vector between the two particles $\mathbf{r}_{ij} = \mathbf{r}_i' - \mathbf{r}_j$
- Calculate $r_{ij}'^2$, $r_{ij}'^6$ and $r_{ij}'^{12}$ for a particle pair
- Calculate the modified force $24 \sum_j \left[2 r_{ij}'^{-12} - r_{ij}'^{-6} \right] r_{ij}'^{-2} \mathbf{r}'_{ij}$
- Add the force to both particles force account, halves the necessary computations. One positive one negative

N_{atoms}	4	32	108	256	500	864	1372	2048
time (s)	0.003539	0.054251	0.274064	1.22426	3.96325	11.3712	28.3554	62.9434
time (s) with list	-	0.179424	1.74696	2.89621	9.14402	13.3765	33.5662	39.6798

Table 1: This table shows time to compute 100 timesteps with different amount of atoms in the model. The time is increasing much faster than linearly. After the neighborlists are implemented it is slower on very few atoms, but it grows slower and spends less time aabove 2048 atoms.

1.5 h: Neighbour lists

A simple system of 3×3 neighbors, see figure 2a will have all the properties and connections of a larger system, since a neighbor is only connected to the neighbors bordering it. Let us choose the neighbor in the center, 111, and find the neighbors we must go through to find to let it interact with each of it's neighbors once. First we let it interact with the top layer, 2b, and then we note that it is not necessary to let it interact with the bottom layer, since all the cells in that layer will interact with the choosen cell, 111. Then we fill the remainer of the front,

Table 1 shows the time spent computing with different amount of atoms in the system, as the system increases the time is increasing fast and we can see the need to implement neighbor cells. The time is increasing fast because the program has to calculate the forces between all the atoms and that is of the order $\mathcal{O}\big(N^2\big)$. By only calculating the nearby atoms, which will be the ones affecting each other the most, the time taken will mostly increase linearly.

The force is mostly ignorable outside a distance of 3σ, see figure 3.

- Divide box into smaller boxes with $l > 3\sigma$, and put the cell layer class between the system and atoms.

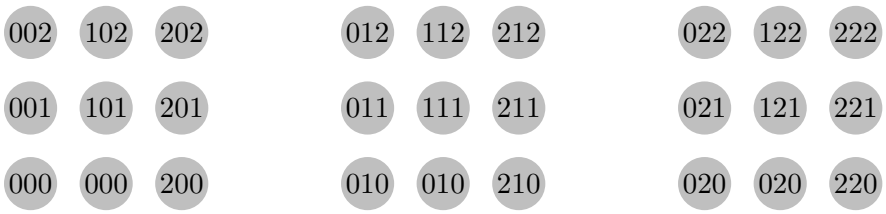
  $$\text{system} \rightarrow \text{cellbox} \rightarrow \text{atoms}$$

- For each box go through all the boxes neighboring boxes. Boxes should be stored in three lists as (i, j, k) and the neighboring boxes to box (i, j, k) are $(i-1, j-1, k-1), (i-1, j-1, k)$ and going through all the combinations to $(i+1, j+1, k+1)$
- Each atom needs an additional box assignment property
- Calculate the forces for all the atoms belonging to the neighboring boxes

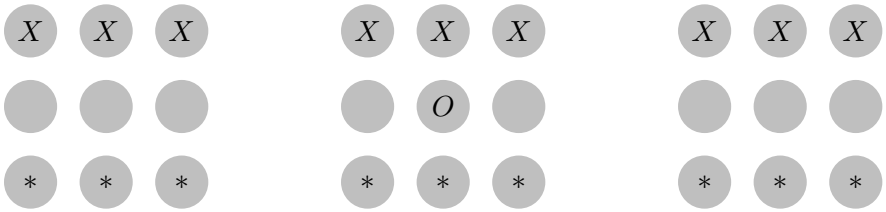
1.6 Task i: Uniform velocity distribution and central limit theorem

From the equipartition theorem, (1) we can find out which mean squared velocity the atoms to have to have at a certain temperature, and which range the uniform velocity distribution should be between to produce the temperature.

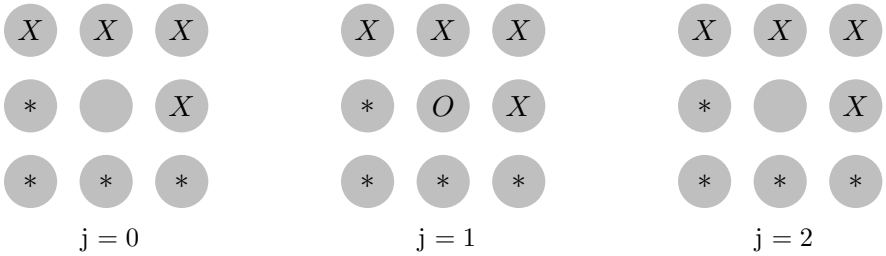
$$\langle E_k \rangle = \frac{2}{3} k_B T \tag{1}$$



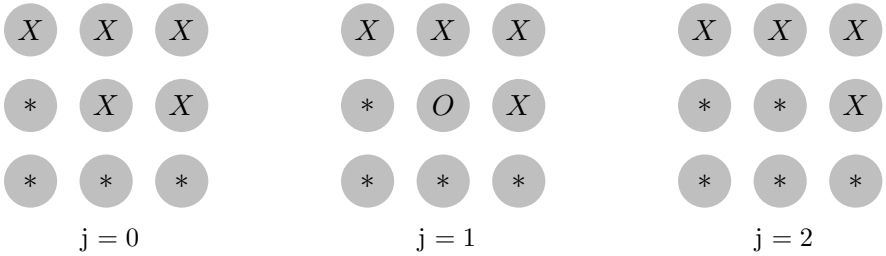
(a) All the neighbors in a 3×3 system divided up into three slices with j constant. The numbers are position in an ijk-grid.



(b) Let the center cell interact with the top layer. 'X' represents a cell that the center cell, 'O' has interacted with. '*' represents a cell that will interact with the center cel, 'O' if it undergoes the same scheme as the center cell 'O'.



(c) Then we let the center cell interact with the rest of the front, i-direction.



(d) Then we let the center cell interact with side, j-direction and all the neighboring cells to cell 111 is interacted with, or interacts with cell 111 when all the cells are run trough.

Figure 2: A schematic explanation of the algorithm to go through all the neighboring cells of a neighbor cell.

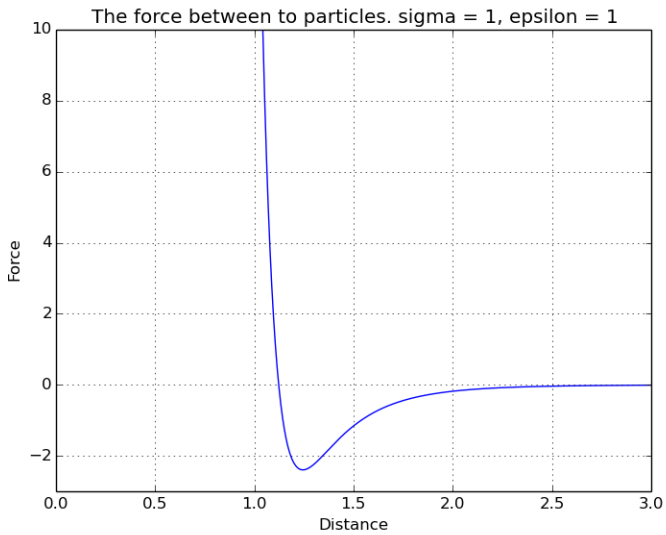


Figure 3: A picture over the force felt between two particles due to the Lennard-Jones potential. At approximately 3 σ of length the magnitude of the force closes to zero and we can ignore the contribution of atoms further away from each other.

If we consider this at an instantenous time

$$\frac{mv^2}{2} = \frac{2}{3}k_B T \tag{2}$$

Since $m = k_B = 1$ in our unit system, and the velocity has three dimensions

$$v^2 = \frac{T}{3} \tag{3}$$

If the uniform distribution runs from $-a \rightarrow a$, then it will have a standard deviation of $\sigma = \frac{a-(-a)}{(12)^{1/2}} \rightarrow a = \sigma(3)^{1/2}$

$$Var(v) = \langle v^2 \rangle - \langle v \rangle^2 = \sqrt{3}T = \sigma \tag{4}$$

$$a = T\sqrt{3} \tag{5}$$

1.7 Task o: Radial distribution function

Not completely done yet, need to divide radial distrbution function by area of shells and stuff, waiting with it for a while.

1.8 Task g: Implementing a thermostat

After initializing the system, it will after while stabilize with a different temperature than it was initialized in. So we will need to implement a thermostat and let that run for a while to get the system to maintain that temperature that we want to measure statistical properties at. In

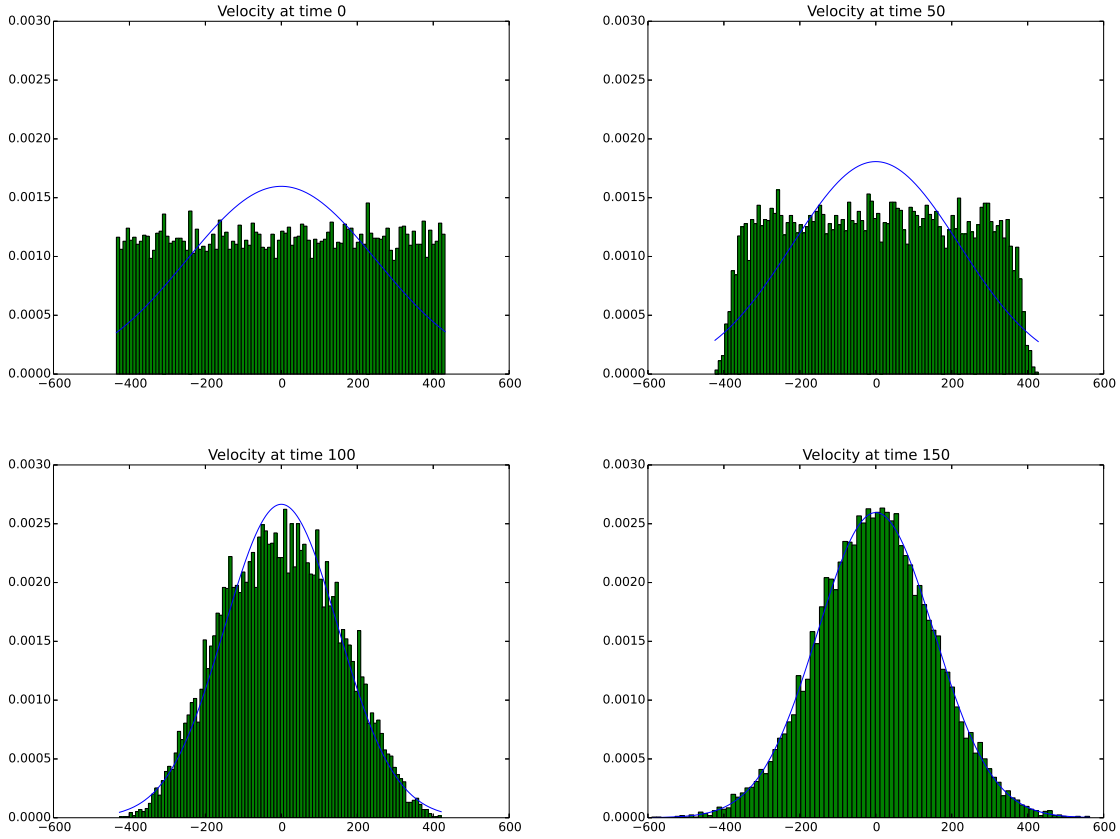


Figure 4: Four figures of the distribution of velocities in the x-direction first given an uniform distribution developing into a gaussian distribution, the line is the best gaussian fit.

this project we will implement a Berendsen thermostat, that works by first calculating a scalar value ,γ, for how much the temperature of the system is different from the wanted temperature, see equation (6), where γ is the adjustment scalar, Δt is the timestep, T_Bath and T is the measured and wanted temperatures, τ is a relaxation factor.

$$\gamma = \sqrt{1 - \frac{\Delta t}{\tau}\left(\frac{T_{Bath}}{T} - 1\right)} \tag{6}$$

Then all of the individual atoms velocities are multiplied with the factor γ and the temperature is increased, or decreases, to get closer to the wanted temperature.

A Unit scheme

We want to use a unit system so all the numbers are computed on a unity scale, this ensures that there will be no overflow and it is easy to check that the number are approximately correct.

The program uses a more natural set of units which let's Boltzmann's constant be 1.

$$1 \text{ mass unit } = 39.948 \text{ a.m.u } = 39.948 \times 1.661 \times 10^{-27} \text{ kg} \tag{7}$$

$$1 \text{ length unit } = 3.405 \text{ Å} = 3.405 \times 10^{-10} \text{ m} \tag{8}$$

$$1 \text{ energy unit } = 1.651 \times 10^{-21} \text{ J} \tag{9}$$

$$1 \text{ temperature unit } = 119.735 \text{ K} \tag{10}$$

All the other units are then expressed in term of these units.

Boltzmann constant translates between temperature and energy, in SI-units it is $k_B = 1.381 \times 10^{-23}$ J K^{-1}, which in the previously defined units becomes

$$k_B = 1.381 \times 10^{-23} \left(\text{ J K}^{-1}\right) \left(\frac{\text{energy unit}}{1.651 \times 10^{-21} \text{ J}}\right) \left(\frac{119.735 \text{ K}}{\text{temperature unit}}\right) \tag{11}$$

$$k_B = 1 \, \frac{\text{energy unit}}{\text{temperature unit}} \tag{12}$$

References

[1] Wikipedia `http://en.wikipedia.org/wiki/Lennard-Jones_potential`.