
Atomic modeling of argon

PROJECT 5, FYS-3150

INA K. B. KULLMANN, CANDIDATE NR: 20

Abstract

Goal: Find the melting temperature of the solid.

The aim of this project is to numerically find the critical temperature for the two dimensional Ising model by using the metropolis algorithm. We will first test the implementation of the algorithm carefully, first by comparing with theoretical values calculated for a small system. Then we will see if the algorithm behaves as expected according to our physical intuition for a larger system.

When we have found a estimate for the critical temperature we will compare it to Lars Onsagers analytical result.

All source codes can be found at: <https://github.com/inakbk/molecular-dynamics-fys3150>.

Contents

1	Introduction	3
2	Theory	3
3	Numerical methods	3
3.1	Creating the lattice	3
3.2	The Maxwell-Boltzmann distrubution (giving velocities)	4
3.3	Periodic boundaryconditions (PBCs)	5
3.4	Intergration methods	6
3.5	The Lennard-Jones potential (interaction/forces)	6
3.6	Calculating physical properties of the system	9
4	Results	10
4.1	Testing the implementation of the model and algorithm	10
4.2	Equilibrium time?	10
4.3	Comparing the two integrators	10

1 Introduction

Molecular dynamics (MD) is a computer simulation method used to study atoms and molecule structure and movement. In a MD simulation the atoms or molecules are allowed to interact through a force given by a potential for a given time. This makes it possible to study the systems development over time.

MD is a type of N-body simulation since the simulation often consists of a large number of atoms or molecules. It is therefore possible to use MD to study statistical properties of a large system consisting of N such atoms or molecules. For systems that obey the ergodic hypothesis the evolution of a single molecular dynamics simulation may be used to determine macroscopic thermodynamic properties of the system. This is because the time averages of an ergodic system correspond to microcanonical ensemble averages¹.

Often the main motivation to use Molecular dynamics is that it is not possible to determine properties of the system analytically because of the large number of particles. The main limitation for the numerical simulation is the computer resources available, but also cumulative errors in the numerical integration. The first is solved by applying periodic boundary conditions while the latter is solved by proper selection of algorithms and parameters. In this paper we will have a look at two numerical integration methods; the Euler-Cromer method and the Velocity Verlet integrator.

In this paper we will study the properties of a large system consisting of Argon atoms. And compare with experimental data(?). We will have a constant number of particles, a constant volume and a more or less constant energy (depending on the integrator). We are more interested in the statistical properties of the system than in the individual motion of each of the particles. We want to sample microstates from the microcanonical ensemble (NVE). (?se over avsnittet over?)

The applications of MD is many ranging from chemical physics, materials science and the modelling of biomolecules. What areas of physics can it be used in? Chemistry and biology?(fra oppg) see 'Areas of application and limitations' at https://en.wikipedia.org/wiki/Molecular_dynamics and google

2 Theory

3 Numerical methods

3.1 Creating the lattice

We want to start out the simulation with the atoms in a crystal structure. We will choose the face-centered cubic (FCC) lattice which is a stable structure for the Lennard-Jones potential that will be used. The lattice is build up by unit cells, a group of atoms, so that a larger system can be created by repeating these cells in space. The FCC lattice unit cell is of size

¹https://en.wikipedia.org/wiki/Molecular_dynamics 3.dec 11:25

$b\text{\AA}$ and consists of four atoms with local coordinates

$$\mathbf{r}_1 = 0\hat{\mathbf{i}} + 0\hat{\mathbf{j}} + 0\hat{\mathbf{k}}, \quad (1)$$

$$\mathbf{r}_2 = \frac{b}{2}\hat{\mathbf{i}} + \frac{b}{2}\hat{\mathbf{j}} + 0\hat{\mathbf{k}}, \quad (2)$$

$$\mathbf{r}_3 = 0\hat{\mathbf{i}} + \frac{b}{2}\hat{\mathbf{j}} + \frac{b}{2}\hat{\mathbf{k}}, \quad (3)$$

$$\mathbf{r}_4 = \frac{b}{2}\hat{\mathbf{i}} + 0\hat{\mathbf{j}} + \frac{b}{2}\hat{\mathbf{k}}. \quad (4)$$

where $b = 5.26$ is the lattice constant. We want to construct a larger system with $N \times N \times N$ such unit cells next to each other. The origin of unit cell (i, j, k) is

$$\mathbf{R}_{i,j,k} = i\hat{\mathbf{u}}_1 + j\hat{\mathbf{u}}_2 + k\hat{\mathbf{u}}_3, \quad (5)$$

where $i = 0, 1, \dots, N_x - 1, j = 0, 1, \dots, N_y - 1, k = 0, 1, \dots, N_z - 1$. The unit vectors of the unit cells are scaled with the lattice constant b so that

$$\hat{\mathbf{u}}_1 = b\hat{\mathbf{i}}, \quad \hat{\mathbf{u}}_2 = b\hat{\mathbf{j}}, \quad \hat{\mathbf{u}}_3 = b\hat{\mathbf{k}}. \quad (6)$$

The coordinates of particle one to four is then given by:

$$\begin{array}{llll} x_1 = 0 + ib & x_2 = \frac{1}{2} + ib & x_3 = 0 + ib & x_4 = \frac{1}{2} + ib \\ y_1 = 0 + jb & y_2 = \frac{1}{2} + jb & y_3 = \frac{1}{2} + jb & y_4 = 0 + jb \\ z_1 = 0 + kb & z_2 = 0 + kb & z_3 = \frac{1}{2} + kb & z_4 = \frac{1}{2} + kb \end{array}$$

Each of the sides of the simulation box has the length $L = bN$ so that the size of the whole system or the volume of the cubic lattice is given by $V = L^3 = (bN)^3$. The density of the system is then:

$$\rho = \frac{M_{tot}}{V} = \frac{4 * N^3 * m}{(bN)^3} = \frac{4m}{b^3}$$

where m is the mass of one atom and is equal for all the atoms. We can see that the density will be constant as long as the mass and the lattice constant is held constant.

should include particle density not mass density! (?)

3.2 The Maxwell-Boltzmann distrubution (giving velocities)

The atoms are usually given velocities according to the Maxwell-Boltzmann distribution so that

$$P(v_i)dv_i = \left(\frac{m}{2\pi k_B T}\right)^{1/2} \exp\left(-\frac{mv_i^2}{2k_B T}\right) dv_i, \quad (7)$$

where m is the mass of the atom, k_B is Boltzmann's constant and T is the temperature. We recognize this as a normal distribution with zero mean and standard deviation $\sigma = \sqrt{k_B T/m}$. This is a good choice because bla bla bla bla (?)

But this will result in a nonzero net momentum in the system which we will remove. The momentum in a given direction for one atom is $p_i = mv_i$ where the mass is m and v_i is the velocity in direction i . The total momentum in one direction, assuming that all particles have the same mass, is given by:

$$p_{tot} = m(v_{i,1} + v_{i,2} + v_{i,3} + \dots + v_{i,n})$$

where n is the total number of atoms in the simulation.

We want to remove this momentum evenly from all the particles. The momentum we want to remove from each particle is then:

$$p_{rm} = \frac{p_{tot}}{n}$$

so that p'_i is the momentum for one particle after removing the total momentum:

$$p'_i = p_i - p_{rm}$$

If we write this out

$$\begin{aligned} mv'_i &= mv_i - \frac{m(v_{i,1} + v_{i,2} + v_{i,3} + \dots + v_{i,n})}{n} \\ \Rightarrow v'_i &= v_i - \frac{v_{i,1} + v_{i,2} + v_{i,3} + \dots + v_{i,n}}{n} \\ &= v_i - \bar{v}_i \end{aligned}$$

We see that to remove the total momentum we only have to subtract the average velocity from each of the particles velocity.

3.3 Periodic boundaryconditions (PBCs)

To avoid problems with boundary effects we will apply periodic boundary conditions. This has a great analogy to 'old' video games such as Snake 2. If the snake head passes through one side of box, it re-appears on the opposite side with the same velocity. In the system of atoms this would mean that if an atom should leave the simulation box at one side it will enter on the opposite side with the same values for the physical parameters as it had before it left. This also implies that an atom at the edge of the box will interact with an atom at the opposite side of the box so that every atom have the same number of 'neighbours'.

MD simulations that use periodic boundary conditions have a large number of unit cells (defined above?), a group of atoms, so that a larger system can be created by repeating these cells in space.

The size of the simulation box must also be large enough to prevent unphysical behaviour. If the box is too small one unit cell (a few atoms) might interact with itself. In the Snake analogy this would mean that the "head" interacts with or bites its own "tail" trough the wall which is allowed in the game, but not very physical. Thus the box size have to be large enough relative to the size of a unit cell, length of the simulation and the desired accuracy².

²https://en.wikipedia.org/wiki/Periodic_boundary_conditions#Practical_implementation:_continuity_and_the_minimum_image_convention

3.4 Intergration methods

During the first implementation of the MD code we have used the Euler-Cromer method to find the next time step for the atoms. (bla bla legge til likninger?)

In MD the Velocity Verlet integrator is usually used. This is because other integration algorithms tend to drift the energy over time, which we want to avoid. The Velocity Verlet algorithm consists of three steps (in addition to computing the forces, see section 3.5)

$$\mathbf{v}(t + \Delta t/2) = \mathbf{v}(t) + \frac{\mathbf{F}(t)}{m} \frac{\Delta t}{2} \quad (8)$$

$$\mathbf{r}(t + \Delta t) = \mathbf{r}(t) + \mathbf{v}(t + \Delta t/2) \Delta t \quad (9)$$

$$\mathbf{v}(t + \Delta t) = \mathbf{v}(t + \Delta t/2) + \frac{\mathbf{F}(t + \Delta t)}{m} \frac{\Delta t}{2}. \quad (10)$$

symplectic integrator bla bla bla?

To measure the drift of the energy we will use the standard deviation of the total energy E :

$$\sigma_E = \sqrt{\langle E^2 \rangle - \langle E \rangle^2}$$

that we will calculate using the Euler-Cromer method and the Velocity-Verlet integrator and compare the results.

Will also compare cumputing time (plot this!?)

say something more about energy drift?

(discussed in the lecture notes.?) see https://en.wikipedia.org/wiki/Verlet_integration#Velocity_Verlet on conserved quantities
bla bla bla

3.5 The Lennard-Jones potential (interaction/forces)

In this project we will use the Lennard-Jones potential which is a mathematically simple model that approximates the interaction between a pair of neutral atoms or molecules. Due to its computational simplicity, the Lennard-Jones potential is used extensively in computer simulations even though more accurate potentials exist³.

The potential calculates the energy between two atoms i and j as

$$U(r_{ij}) = 4\epsilon \left[\left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^6 \right], \quad (11)$$

where $r_{ij} = |\mathbf{r}_i - \mathbf{r}_j|$ is the distance from atom i to atom j , ϵ is the depth of the potential well (units energy) and σ is the distance at which the potential is zero.

The parameters in the potential can be fitted to reproduce experimental data. For argon, optimal values of the parameters are:

$$\frac{\epsilon}{k_B} = 119.8 \text{ K}, \sigma = 3.405 \text{ \AA}. \quad (12)$$

With these parameters the L-J potential reproduces equilibrium thermodynamic properties that are in good agreement with experimental values for argon.

³http://en.wikipedia.org/wiki/Lennard-Jones_potential

In our code, we use the so called MD units (see appendix ??). (finne enheten og verdier til parameterne?)

The force is calculated by taking the negative gradient of the potential

$$\mathbf{F}(r_{ij}) = -\nabla U(r_{ij}), \quad (13)$$

giving the x -component (the other components are calculated the same way)

$$F_x(r_{ij}) = -\frac{\partial U}{\partial r_{ij}} \frac{\partial r_{ij}}{\partial x_{ij}}, \quad (14)$$

where x_{ij} is the x -component of \mathbf{r}_{ij} . Calculating the differensials separately:

$$\begin{aligned} \frac{\partial U}{\partial r_{ij}} &= \frac{\partial}{\partial r_{ij}} 4\epsilon \left[\left(\frac{\sigma}{r_{ij}} \right)^{12} - \left(\frac{\sigma}{r_{ij}} \right)^6 \right] = 4\epsilon \left[\sigma^{12} \frac{\partial}{\partial r_{ij}} r_{ij}^{-12} - \sigma^6 \frac{\partial}{\partial r_{ij}} r_{ij}^{-6} \right] \\ &= 4\epsilon \left[-12\sigma^{12} r_{ij}^{-13} - (-6)\sigma^6 r_{ij}^{-7} \right] = \frac{24\epsilon}{\sigma} \left[\left(\frac{\sigma}{r_{ij}} \right)^7 - 2 \left(\frac{\sigma}{r_{ij}} \right)^{13} \right] \\ r_{ij} &= |\mathbf{r}_i - \mathbf{r}_j| = |x_i \hat{\mathbf{i}} + y_i \hat{\mathbf{j}} + z_i \hat{\mathbf{k}} - x_j \hat{\mathbf{i}} + y_j \hat{\mathbf{j}} + z_j \hat{\mathbf{k}}| = |(x_i - x_j) \hat{\mathbf{i}} + (y_i - y_j) \hat{\mathbf{j}} + (z_i - z_j) \hat{\mathbf{k}}| \\ &= \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} = \sqrt{x_{ij}^2 + y_{ij}^2 + z_{ij}^2} \\ \frac{\partial r_{ij}}{\partial x_{ij}} &= \frac{\partial}{\partial x_{ij}} \sqrt{x_{ij}^2 + y_{ij}^2 + z_{ij}^2} = \frac{1}{2r_{ij}} \cdot 2x_{ij} = \frac{x_{ij}}{r_{ij}} \end{aligned}$$

gives

$$\begin{aligned} F_x(r_{ij}) &= -\frac{24\epsilon}{\sigma} \left[\left(\frac{\sigma}{r_{ij}} \right)^7 - 2 \left(\frac{\sigma}{r_{ij}} \right)^{13} \right] \cdot \frac{x_{ij}}{r_{ij}} \\ &= \frac{24\epsilon}{\sigma^2} \left[2 \left(\frac{\sigma}{r_{ij}} \right)^{14} - \left(\frac{\sigma}{r_{ij}} \right)^8 \right] x_{ij} \end{aligned}$$

The force in y and z direction can be calculated the same way giving:

$$\begin{aligned} \mathbf{F} &= F_x \hat{\mathbf{i}} + F_y \hat{\mathbf{j}} + F_z \hat{\mathbf{k}} \\ &= \frac{24\epsilon}{\sigma^2} \left[2 \left(\frac{\sigma}{r_{ij}} \right)^{14} - \left(\frac{\sigma}{r_{ij}} \right)^8 \right] (x_{ij} \hat{\mathbf{i}} + y_{ij} \hat{\mathbf{j}} + z_{ij} \hat{\mathbf{k}}) \end{aligned}$$

(another section???)

In section 3.4 we see that to calculate the next time step for the particles we need to calculate the forces. This is very time consuming because we have to sum over all pairs of particles to find the force for one particle. Therefore a lot of time can be saved if the code for the calculation of forces is as efficient as possible.

A better numerical expression for the force is

$$\begin{aligned} \mathbf{F}(r_{ij}) &= \frac{24\epsilon}{\sigma^2} \left(\frac{\sigma}{r_{ij}} \right)^2 \left(\frac{\sigma}{r_{ij}} \right)^6 \left[2 \left(\frac{\sigma}{r_{ij}} \right)^6 - 1 \right] (x_{ij} \hat{\mathbf{i}} + y_{ij} \hat{\mathbf{j}} + z_{ij} \hat{\mathbf{k}}) \\ &= 24\epsilon \sigma^6 r_{ij}^{-2} r_{ij}^{-6} (2\sigma^6 r_{ij}^{-6} - 1) \end{aligned}$$

and for the potential energy

$$U(r_{ij}) = 4\epsilon \sigma^6 r_{ij}^{-6} (\sigma^6 r_{ij}^{-6} - 1)$$

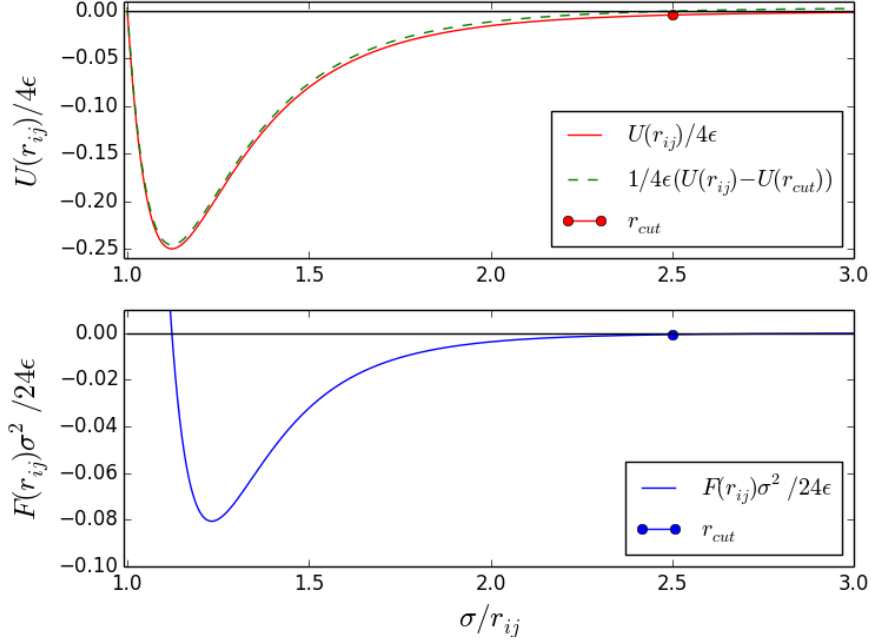


Figure 1: **Top:** The Lennard-Jones potential (red), or the potential energy $U(r_{ij})$ between two atoms i and j plotted against the distance between atom i and j , r_{ij} . The point $(r_{cut}, U(r_{cut}))$ is marked with a red dot. The shifted potential function is also plotted in green. **Bottom:** The force (blue) between two atoms i and j plotted against the distance between atom i and j , r_{ij} . The point $(r_{cut}, F(r_{cut}))$ is marked with a blue dot.

so that the powers of r_{ij} is calculated as few times as possible, r_{ij}^{-6} can be calculated just once for each loop (define?).

Another smart way to make a more efficient code when calculating the forces is to realize that in the limit $r_{ij} \rightarrow \infty$ both the force and the potential energy is zero. In figure 1 the potential energy (red) and the force (blue) plotted against the distance between two atoms r_{ij} . We can see that the energy and the force reaches zero for large values of r_{ij} . The point for which the potential energy and the force is nearly zero, r_{cut} , is marked with a dot in the figure. A good approximation is therefore to stop calculating forces and energies for distances larger than r_{cut} and set all forces and energies for $r_{ij} > r_{cut}$ to zero.

The implementation of r_{cut} will save the algorithm a lot of time and still be a good approximation since the force and the potential energy is esentially zero anyway. The only problem is that the potential energy function will be discontinuously at the point r_{cut} when the energy is set to zero after r_{cut} . A solution to this is to shift the potential function slightly so that the energy is zero at r_{cut} and still set to zero for $r_{ij} > r_{cut}$. The shifted potential function is marked with green in figure 1. This will only shift the values for the potential and total energy, but it will not affect the physics of the problem. The statistical properties of the system will not be affected by the shift in energy.

The minimum image convention (due to calculation of forces) When an atom leave the simulation box it must re-enter the box on the opposite side. An atom at the edge of the simulation box must also interact with an atom at the other side of the box 'through' the wall, or interact with the image of the simulation box. But this atom is also contained inside the box so the atoms interact twice, once through the wall and once from across the box. To avoid counting this interaction twice we will apply the minimum image convention. An atom will interact with the other atoms only once and then choose the image (or the simulation box) that gives the minimum distance⁴:

```

if (periodic_x) then
  dx = x(j) - x(i)
  if (dx > x_size * 0.5) dx = dx - x_size
  if (dx <= -x_size * 0.5) dx = dx + x_size
endif

```

(include drawing??)

This leads to two possible strategic choices: (A) 'fold back' particles into the simulation box when they leave it, or (B) let them go on into the other side of the box?

....ahh, this is relevant for computing forces/potentials! An atom which has passed through one face of the simulation box should re-enter through the opposite face-or its image should do it. Evidently, a strategic decision must be made: Do we (A) 'fold back' particles into the simulation box when they leave it, or do we (B) let them go on (but compute interactions with the nearest images)? The decision has no effect on the course of the simulation, but if the user is interested in mean displacements, diffusion lengths, etc., the second option is preferable.

3.6 Calculating physical properties of the system

The total potential energy V in the system is computed by summing over all pairs of atoms (counting each pair only once)

$$V = \sum_{i>j} U(r_{ij}). \quad (15)$$

where $U(r_{ij})$ are the Lennard-Jones potential given in section 3.5. The kinetic energy is defined as

$$E_k = \sum_{i=1}^{N_{\text{atoms}}} \frac{1}{2} m_i v_i^2, \quad (16)$$

where m_i and v_i is the mass and the speed of atom i . The total energy of the system is then $E = V + E_k$. (which we will study if is conserved or not, see σ_E ??)

We will also calculate an estimate of the temperature through the equipartition theorem⁵

$$\langle E_k \rangle = \frac{3}{2} N_{\text{atoms}} k_B T. \quad (17)$$

⁴https://en.wikipedia.org/wiki/Periodic_boundary_conditions#Practical_implementation:_continuity_and_the_minimum_image_convention

⁵See for example D. Schroeder's *An Introduction to Thermal Physics* for details. better reference?

We can use this to define an *instantaneous* temperature

$$T = \frac{2}{3} \frac{E_k}{N_{\text{atoms}} k_B}. \quad (18)$$

The diffusion constant of the system can be used to measure the melting temperature. We use the Einstein relation that relates the so called mean square displacement (MSD) to the diffusion constant

$$\langle r^2(t) \rangle = 6Dt, \quad (19)$$

where D is the diffusion constant, t is time. The mean square displacement for atom i is calculated as

$$r_i^2(t) = |\mathbf{r}_i(t) - \mathbf{r}_i(0)|^2, \quad (20)$$

where $\mathbf{r}_i(t)$ is the position of atom i at time t .

The diffusion constant is then given by

$$D = \frac{\langle r^2(t) \rangle}{6t} =$$

(for a given temperature (T_i ?).)

Atoms in a solid will not diffuse (move?) much, so a solid will have a diffusion constant close to zero. A plot of the diffusion constant as a function of temperature will therefore reveal the melting temperature because the solid melts when the diffusion constant is non zero.

4 Results

4.1 Testing the implementation of the model and algorithm

4.2 Equilibrium time?

4.3 Comparing the two integrators

In the top plot in figure 2 we can see the standard deviation of the total energy σ_E plotted versus the timestep dt for the Euler-Cromer and Velocity Verlet integrator. The units on the axis are MN unist explained in section (?do it? explain in caption?) We see that the drift in the energy is significantly larger for the Euler-Cromer method than for the Velocity Verlet integrator and that it increases with increasing dt for the Euler-Cromer method. For the Velocity Verlet integrator the choice of dt does not matter much, the energy drift, or the value for σ_E is small for all values of dt .

In the bottom plot in figure 2 we can see the execution time plotted versus the timestep for the Euler-Cromer and Velocity Verlet integrator. We see that there is not much difference in the execution time between the two integrators and that the dependence on the value of dt seems rather random.

Both plots in figure 2 were obtained with 5 unit cells in each direction, initial temperature $T_i = 2.5 \approx 300$ K, lattice constant $b = 5.26\text{\AA}$ and 1000 timesteps in the simulation. Similar plots are obtained when running with 10 unit cells in each direction.

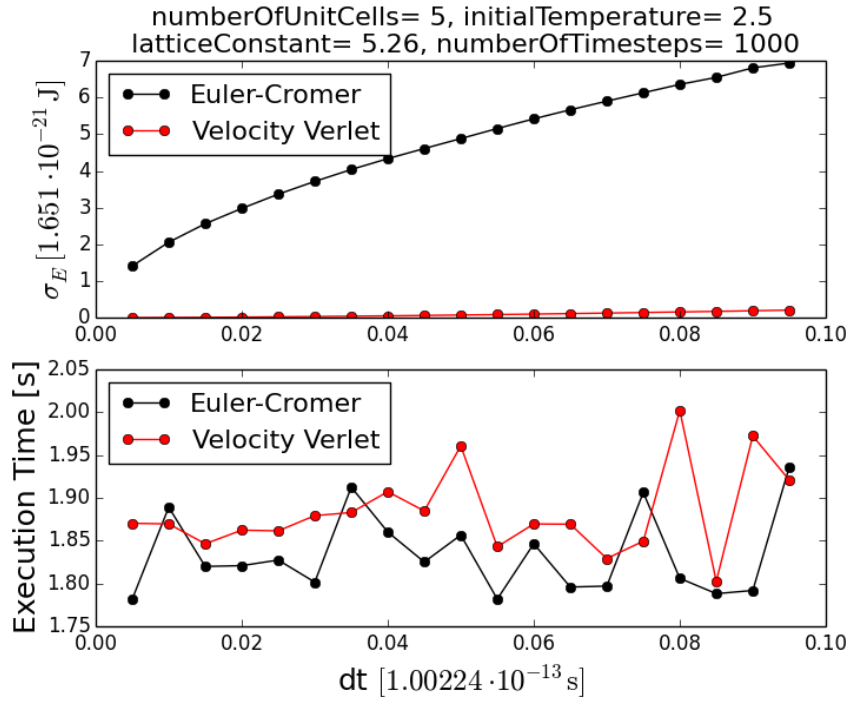


Figure 2: **Top:** The standard deviation of the total energy σ_E plotted versus the timestep dt for the Euler-Cromer and Velocity Verlet integrator with 5 unit cells in each direction, initial temperature $T_i = 2.5 \approx 300 \text{ K}$, lattice constant $b = 5.26 \text{ \AA}$ and 1000 timesteps in the simulation. **Bottom:** The execution time plotted versus the the timestep dt for the Euler-Cromer and Velocity Verlet integrator with 5 unit cells in each direction, initial temperature $T_i = 2.5 \approx 300 \text{ K}$, lattice constant $b = 5.26 \text{ \AA}$ and 1000 timesteps in the simulation.

In figure 3 we see the execution time plotted versus the number of unit cells in each direction for the Euler-Cromer and Velocity Verlet integrator. We see that for small number of unit cells the execution time is about the same for both integrators, they are equally fast. For 15 unit cells in each direction the two integrators deviate slightly in the execution time, but the execution time is quite large, $t_{exe} \approx 800\text{s} \approx 13\text{min}$ so that the deviation in the execution time between the integrators are small compared to the execution time. Thus the choice of integrator does not affect the execution time much and we should choose the integrator that gives the best physical results. The best physical results are when the drift in the energy or σ_E is as small as possible, we should therefore use the Velocity Verlet integrator. The further calculations are done with the Velocity Verlet integrator.

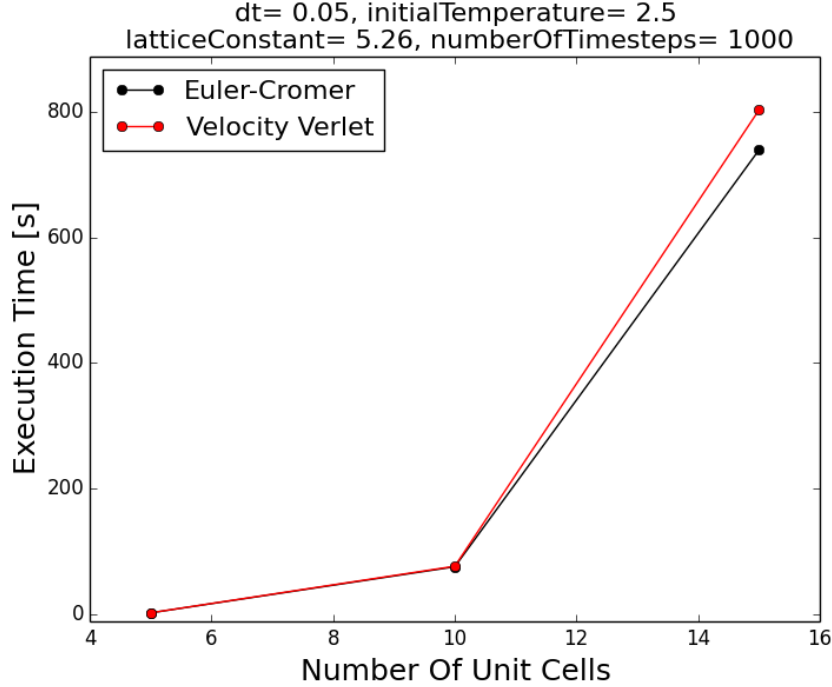


Figure 3: The execution time plotted versus the number of unit cells in each direction for the Euler-Cromer and Velocity Verlet integrator with initial temperature $T_i = 2.5 \approx 300$ K, lattice constant $b = 5.26\text{\AA}$, $dt = 0.05 \approx 5 \cdot 10^{-15}\text{s}$ and 1000 timesteps in the simulation.

number of unit cells is also a measure of how many particles..include?