# Project report

# Introduction

Molecular dynamics is used to model atomic and molecular systems by numerically integrating the equations of motion for the particles. In contrast to Monte Carlo simulations Molecular dynamics models the system as evolving in time. Once the system reaches thermodynamic equilibrium its properties can be decided by the ergodic hypothesis; that the statistical ensemble averages are equal to the time averages of the system.

The goal of the project was to create a molecular dynamics program that can simulate FCC monocrystals by using the Lennard-Jones potential. By changing the parameters of the system such as the Lennard-Jones parameters sigma and epsilon, the lattice constant and initial temperature the program calculates values such as the temperature, the cohesive energy and the specific heat.

The substance is modeled as a bulk with infinite extension in all three directions. The model assumes periodicity, however, which means that only a part of the bulk has to be considered. It is the atoms in this cube-shaped part that the program calculates the positions and movements for.

# User instructions

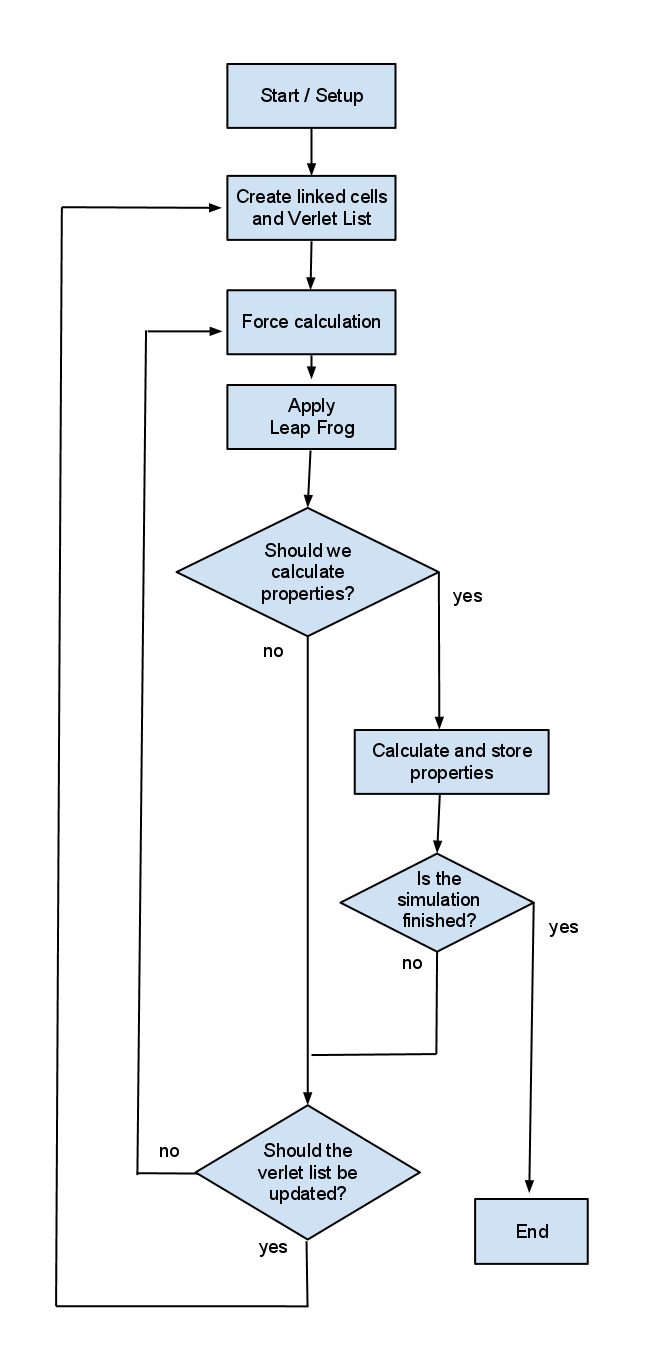
To change the system constants, open the mdmainwin.cpp-file. There are parameter values prepared for four elements: argon, copper, xenon and silver. To select which element to simulate, check for the number above each paragraph of element constants and change it to zero for the elements that should not be simulated and to one for the element that should be simulated. It is also possible to turn off all prepared element constants and write new ones. Below the element constants you can choose how many atoms should be included in the simulation, how many iterations should pass between each measurement, the number of iterations, the inner and outer cutoff radius for the Lennard-Jones potential, what the maximum difference between two consecutive measurements of the potential energy should be in order for the system to be regarded as in equilibrium as well as the parameters for the thermostat and for the filtering function that transforms the measured values into time averages. At the bottom are flags that decide which properties should be calculated. After deciding the system parameters, run the program. A window should emerge with a button that reads “Start simulation”. After pushing that button there is a text reading “Running simulation” followed by a percentage value. When the counter reaches 100 % the simulation is ready and the measured values appear in the window. In the map MD-build-desktop files will appear containing the values. These can be plotted by uploading them to Matlab.

# Data structures

The particles are objects with the properties position, non-modulated relative position, non-modulated relative position when the verlet list was last calculated, velocity and acceleration. The values of position are modulated. This means that the values will be confined to an interval so that the system does not extend outside the system box. The particle indices of the neighbors of all particles are stored in an array. For each particle the indices of the neighboring particles are placed next to each other in the array. In another array the ith element consists of the position in the previous array of the first neighbor of the particle with index i. There is a 3D floating point vector class. In the mdsystem object most of the parameters and variables are stored, such as the material constants sigma and epsilon used in the Lennard-Jones potential, the particle mass, the time length of each iteration, the number of iterations and the vectors containing the measured values of the system properties. A vector which elements are the particles in the system is also included here.

# Flow chart

Below is the flow chart for the main part of the MD-program.



# The main loop

The program has a main loop that maintains the program flow and makes sure all the main parts of the program are called in the right order. This routine makes sure that the simulation steps forward in time, that the desired graphics are plotted in the window and that the widgets handler (Qt) gets the time it needs to handle all the events it wants to handle.

# Start/Setup

In the beginning, the size of the MD box, the time step *Δt*, the number of the particle *N*, and the number of the time steps are decided, followed by the initialization soon after that. In the initialization the atom positions are assigned according to the lattice. The total momentum is set to zero. The velocity distribution is randomized and then rescaled so that the system has the correct temperature.

# Create linked cells and Verlet list

In reality all particles in the system interact with each other. The model only takes into account interactions between particles within a given distance of each other however since it would be too demanding to include all pair interactions in the computer simulation and since a particle is mostly influenced by nearby particles. This distance is called the inner radius. The program has to keep track of which particles are within the inner radius of each other and in order to not make it necessary to check the distances between all the particles every iteration another distance called the outer radius is introduced. Data on which particles are within the outer radius of each other, ie which particles are neighboring each other, are stored and there is no need for an update until enough iterations have passed to make it possible for particles that were originally outside the outer radius of each other to have made it within the inner radius. This data is stored in the Verlet lists. In the first Verlet list the particle indices for the neighbors to the particle with index number 1 are included followed by the particle indices for the neighbors to the particle with index number 2 and so on. In the second Verlet list the i:th element gives the position in the first Verlet list for the index of the first neighbor of the i:th particle. To avoid checking the distance between all the particles when the Verlet lists are updated the system is divided into cubic cells and data on which particles belong to which cell is stored in the Linked cell list. In the first linked cell list the particle indices for the particles in cell 1 are included followed by the particle indices for the particles in cell 2 and so forth in a similar fashion to how the first Verlet list is constructed. Similarly the i:th element in the second linked cell list gives the position in the first linked cell list for the index of the first particle in the i:th cell.

When using the Verlet list method the lists will be updated with different intervals depending on how thick the skin is i.e. what the difference is between the inner and outer radius. It will also depend on how fast the particles are moving. For example, faster particles means shorter interval between updates. This means that if one keeps track of how far the particles has moved it can be decided when to update, which means that the list only is updated when it actually needs to be.

# Force calculation

The Lennard-Jones potential for a particle pair is

where r is the distance between the two particles. Epsilon and sigma are material constants. This can be The potential energy for the neighboring particle pairs are set to zero when they are farther apart than the outer radius and according to the Lennard-Jones formula when the particles are within the outer radius of each other.

In order to avoid discontinuity of the potential energy it is shifted to be zero at the cut-off distance.

# Apply Leap Frog

In this step, the previous position, velocity and acceleration are used to calculate the new position and velocity. This is done by the leap-frog algorithm. The instantaneous temperature at iteration n is here calculated as:

# Calculate and store properties

Since we have the velocity and the potential energy in our data structure thus we can calculate the kinetic and total energy. Temperature, pressure, mean square displacement and other properties will also be calculated. In order to provide the user with smoother plots the values presented will then be the mean values over a certain iteration interval.

The followings are the properties we calculated in our project:

Kinetic energy / Temperature

K is kinetic energy and T is temperature.

Potential energy

The Lennard-Jones potential for a particle pair is used to calculate the potential energy,

The potential is subtract by cutoff energy in order to make the potential continuous at the cut off distance. The cutoff energy value can be obtained using the following formula.

The distant particle pairs with negligible potential energy () are filtered out in order to save the CPU time without losing accuracy.

Total energy

The total energy is the sum of the kinetic energy and the potential energy.

Internal pressure

The particles are treated as ideal gases with a correction term related to the sum of the products of the force and the distance.

Where P is the pressure; V is the volume of primary simulation cell. The 1/6 factor is due to the fact that, to avoid double counting.

Specific Heat

Here, we used another formula instead of the one on the lecture slides and the formula we used gives accurate results.

Where N is the number of the particle and Kb is Boltzmann constnt.

Mean square displacement (MSD)

MSD is a property

where

Self-Diffusion coefficient

The Einstein expression for an isotropic fluid at equilibrium connects MSD and the self-diffusion coefficient:

# Results

The following section will contain the results that have been obtained from the simulations.

## Specific heat

The specific heat was calculated using the formula mentioned above. The specific heat is usually given at room temperature, but since the program is best suited for simulating solid phases, the specific heat for the noble gases has been measured at low temperature, when the gases are in solid phase.

The elements that have been simulated are:

- Silver and copper both at room temperature

-Xenon at 100 K

-Argon at 60 K.

The specific heat for silver and copper are easily obtained from tables since the specific heat is well known at room temperature. Specific heat for argon and xenon at low temperatures are trickier to find, but a few sources have been found (see references) to compere the data obtained. In the following table the results are compared with known data. The simulated values are averages over several simulations and the simulation length has been 50 000 fs with a time step of 1 fs. Longer simulations have been conducted without any noticeable improvement.

**Table 1**

|  |  |  |  |
| --- | --- | --- | --- |
| Element | Simulated values, J/(g\*K) | Tabled values, J/(g\*K) | Temperature, K |
| Silver | 0.233 | 0.233 | 293 |
| Copper | 0.385 | 0.386 | 293 |
| Xenon | 0.188 | 0.179 | 100 |
| Argon | 0.607 | 0.55 | 60 |

**Notice the very good values for silver and copper.**

## Kinetic energy, Potential energy and Total energy

# **References**

<http://www.springerlink.com/content/k328237200233456/fulltext.pdf>

<http://www.springerlink.com/content/p2875753h4661128/fulltext.pdf>