Simulation Methods in Statistical Physics

(FK 8028)

Programming project report 2: Extended System

Submitted by Mikhail Ivanov

Introduction

For this part of the programming project, the periodic boundary conditions (PBC), as well as the initial conditions generator, were introduced to molecular dynamics (MD) code for the Lennard-Jones (LJ) fluid simulations. Most of the previous code in *Python 3* was reused, but all the computationally intensive functions were rewritten with *NumPy* and *Numba* libraries to speed-up MD simulations.

For the initial positions of the LJ particles, a simple cubic lattice with the lattice constant equal to *Rm* ($2^{1/6}\sigma$) with any number of unit cells is generated and all the lattice points are filled with simulated particles. Cubic simulation box length is set to the number of unit cells multiplied by *Rm*. PBC ensure that the simulated system is surrounded by its equivalent copies and no surface effects are present. Thus, the bulk properties of the system may be simulated.

Initial velocities are generated in several steps as follows – firstly, random velocities according to the Maxwell-Boltzmann distribution are assigned to each particle, then the center-of-mass motion is removed, and finally, all the velocities are scaled to the desired temperature.

As the LJ potential is sufficiently short ranged, the minimal image convention is an adequate way to implement PBC. Minimal image convention means that each particle interacts with the closest replica of every other particle in the box, so effectively the particle interacts with all the others, like if it was in the center of the simulation box. Interaction cut-off radius may be interpreted as half of the box length. To visualize the particles in PBC, output coordinates were wrapped (by subtracting or adding simulation box length to each particle's coordinate if it was exceeding the box length or less than zero) before saving them to the trajectory file.

New implemented features were tested and simulation of 1000 argon atoms at the starting temperature of 100 K was carried out for 20 ps with 2 fs time step.

Results

1. Initial conditions generator

1.1. Initial positions

Cubic lattice with *Rm* lattice constant is a good starting configuration for LJ fluids, as the potential energy of such system is very low and there are no overlapping atoms, but one should equilibrate the system long enough before starting computing the system's averages.

To check if the correct initial positions for the cubic lattice are obtained, simulation box containing 1000 Ar atoms (10x10x10 lattice points) was generated, and the structure was visualized with *VMD* software package (*Figure 1* and *2*):

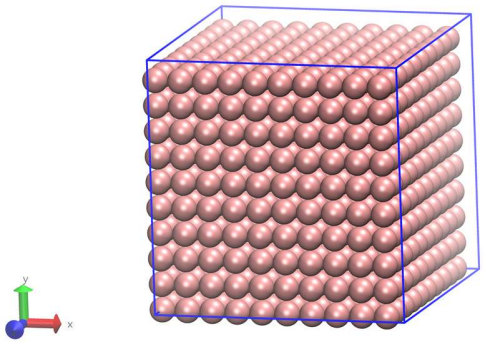

Figure 1 – Starting configuration of 1000 Ar atoms in the simulation box

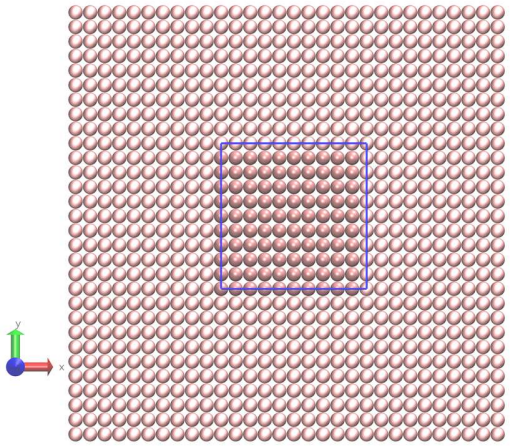

Figure 2 – Simulation box with 1000 Ar atoms, surrounded by its images

From these figures, it is clearly visible that a perfect cubic lattice is generated and that with the PBC, the whole space is filled with its copies.

1.2. Initial velocities

Assigning random velocities according to the Maxwell-Boltzmann distribution to each simulated particle is a very common way of initializing the velocities.

In the present code, the velocity initialization is performed by firstly assigning three random values of v_x, v_y and v_z following the Maxwell-Boltzmann distribution for each particle:

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

Then, center-of-mass velocity vector is obtained by summing all the particles' velocities components and dividing it by the total number of the particles:

$$\vec{V} = \frac{1}{N}\sum_{N}(v_x + v_y + v_z)$$

Center-of-mass velocity vector is subtracted from every particle's velocity vector:

$$\vec{v}_{corrected} = \vec{v} - \vec{V}$$

Finally, all the velocities are scaled to the desired temperature:

$$\vec{v}_T = \vec{v}_{corrected} \cdot \sqrt{\frac{T_{desired}}{T_{calc}}}$$

where T_{calc} is expressed as:

$$T_{calc} = \frac{m\langle v^2 \rangle}{3k}$$

Figure 3 and *Figure 4* show that generated initial velocities are in a good agreement with the Maxwell-Boltzmann distribution – both for velocity component and velocity norm.

2. Periodic boundary conditions

Before starting the simulations, one should make sure that the particles interact correctly through the periodic boundaries. To do that, the total potential energy of 3x3x3 Ar lattice was computed with the code as well as derived theoretically. The test system is shown in *Figure 5* and *Figure 6*. To compute the potential energy by hand, one should consider all the possible interactions in the system. Let us look at the particle in the center of this system (*Figure 7*):

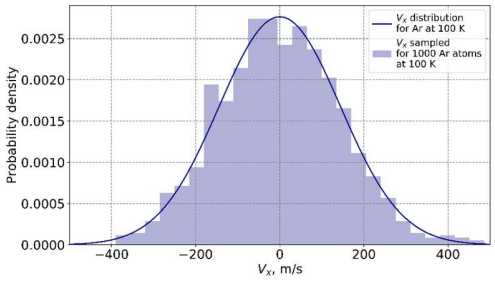

Figure 3 – Velocity component distribution together with the histogram of the velocity component, sampled for 1000 Ar atoms at 100 K

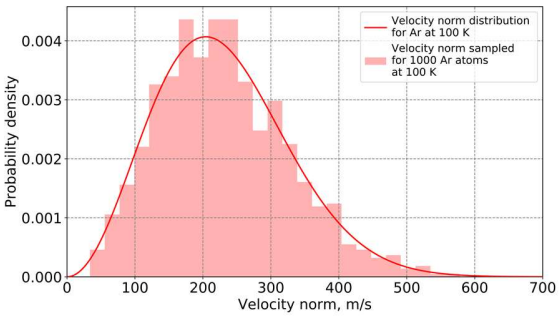

Figure 4 – Velocity norm distribution together with the histogram of the velocity, sampled for 1000 Ar atoms at 100 K

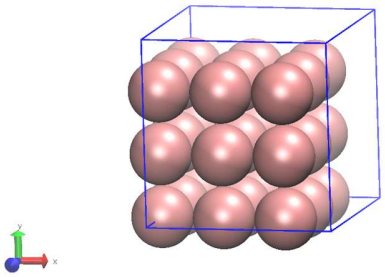

Figure 5 – 3x3x3 Ar lattice in the simulation box

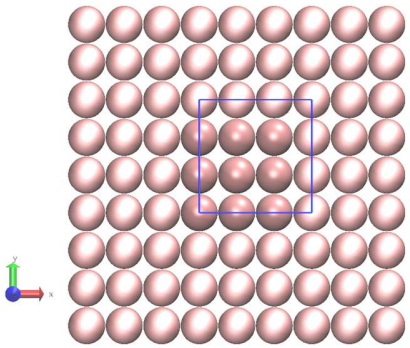

Figure 6 – 3x3x3 Ar lattice surrounded by its images

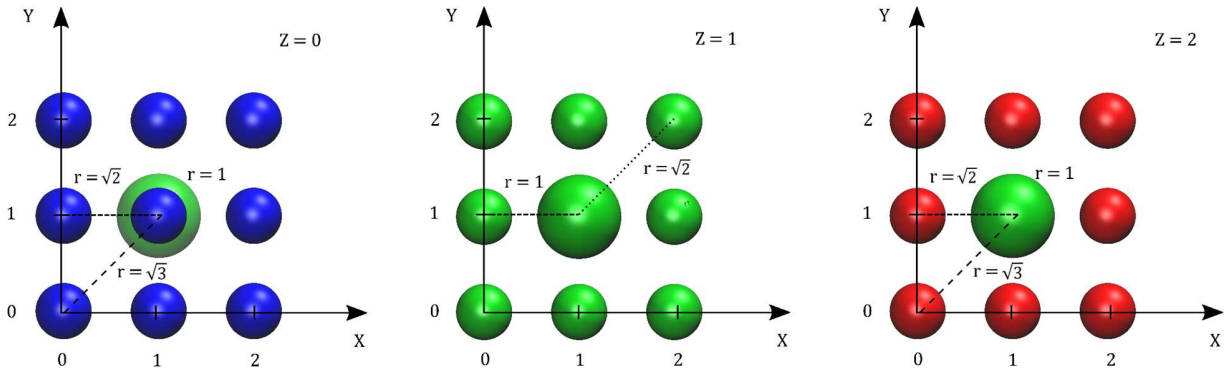

Figure 7 – The diagram showing three z planes of the 3x3x3 Ar lattice. Blue particles reside at z = 0, green at z=1 and red at z=2. Big green particle is the central particle. The diagram shows three different possible values of distances between the particles (hence, three possible interaction energy values) in the system – $r = 1$, $r = \sqrt{2}$ *and* $r = \sqrt{3}$. *Distance unit is Rm* ($2^{1/6}\sigma$).

From *Figure 7*, where the 3x3x3 Ar lattice is shown as three z-planes, it is possible to compute the interaction energies between the central particle and all the other particles. The energies depend on the distance between the particles: 6 particles are separated from the central particle by 1 *Rm*, 12 particles by $\sqrt{2}$ *Rm* and 8 particles by $\sqrt{3}$ *Rm* (26 interaction in total). The first type of interactions (r = *Rm*) yields -ε of potential energy, the second one (r = $\sqrt{2}$ *Rm*) gives ≈-0.234ε and the last one (r = $\sqrt{3}$ *Rm*) ≈-0.073ε. With the PBC we can expect that every particle in the system will behave as if it is in the center of the system, so the only thing to do is to count all the interaction in the system. There 27 particles in the system, so the total number of pairs is 27^2 = 729, but we should include only the total number of unique pairs without self-interaction. We should subtract 27 pairs (self-interaction) from the total number of 729 pairs and divide by two giving the total number of 351 unique pair-wise interactions. From the interactions of central particle with the others we can derive the ratio of different interactions: 6/26 of interactions give -ε of potential energy, 12/26 ≈-0.234ε and 8/26 ≈-0.073ε. Now we can compute the total potential energy of this lattice:

$$V_{total} = 351 \cdot \frac{6}{26} V_{LJ}(Rm) + 351 \cdot \frac{12}{26} V_{LJ}(\sqrt{2}Rm) + 351 \cdot \frac{8}{26} V_{LJ}(\sqrt{3}Rm)$$

$$V_{total} = -81\varepsilon - 37.96875\varepsilon - 7.85185\varepsilon = -126.8206\varepsilon$$

This value is perfectly in line with the total potential energy of 3x3x3 Ar lattice, computed by the program, so we can conclude, that the particles interact correctly through the boundaries.

Another test of the implemented PBC was performed by simulating two Ar atoms in a small simulation box, approaching each other. Their trajectories were visualized with *VMD* (*Figure 8* and *9*) as well as their positions with one periodic copy of each particle along one axis were plotted against the simulation time (*Figure 10*).

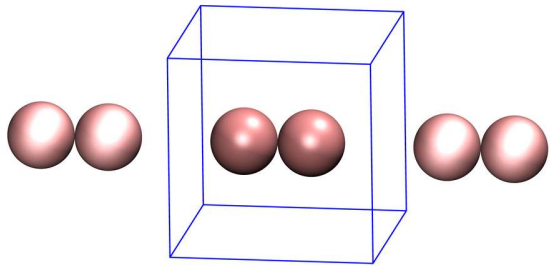

Figure 8 – Two Ar atoms in a small simulation box, interacting with each other in the real space. Two copies of the system are present outside the simulation box.

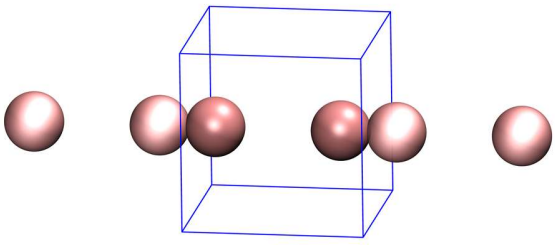

Figure 9 – Two Ar atoms in a small simulation box, interacting with each other through the boundaries

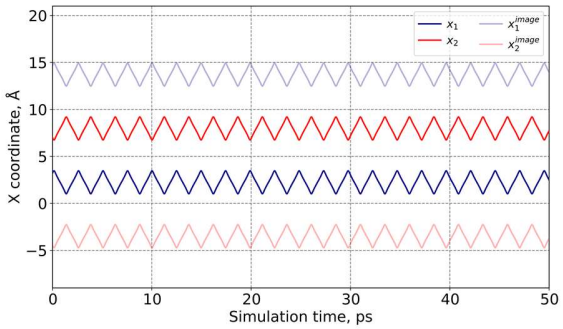

Figure 10 – The positions of two Ar atoms and one periodic copy of each particle against the simulation time. The interactions occur alternately in the real space and through the boundaries.

Wrapping the particles' trajectories (by adding or subtracting the box length to each particle's coordinate so that all them stay in the simulation box and do not drift away from it) is also important when one need to visualize the system's dynamics. The same set up from the previous test is used to test the wrapping algorithm, but both particles were moved closer to the boundary so that one of them would cross it during the oscillations. Their trajectories and positions are shown on *Figures 11-13*:

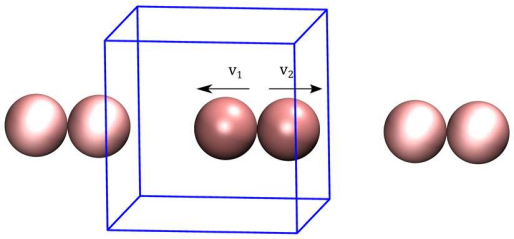

Figure 11 – Initial positions of two oscillating argon atoms and their periodic images. Velocity vectors are shown above the particles.

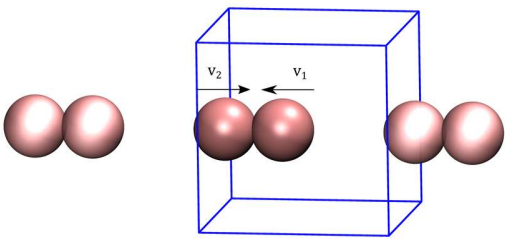

Figure 12 – Second particle has moved through the boundary and collides with the first one from the left edge of the box. Velocity vectors are shown above the particles.

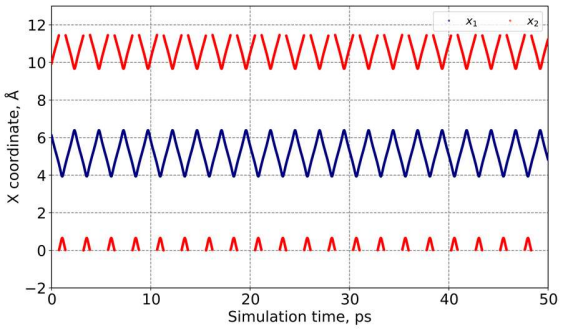

Figure 13 – The positions of two oscillating argon atoms. The second particle moves through the periodic boundaries repeatedly.

These two tests also prove that the periodic boundaries are implemented correctly.

3. MD simulation of 10x10x10 Ar lattice

3.1 Simulation

As a final test of the new features, MD simulation of 1000 Ar atoms at the starting temperature of 100 K was carried out for 20 ps with 2 fs time step. Only each 10^{th} step was saved to the output files. In the previous report, it was shown, that 2 fs time step is enough to avoid any significant energy conservation errors, but the test was performed for a few particles only. However, the new data from *Figure 14* suggests that total energy is conserved even for a thousand of particles in the simulation:

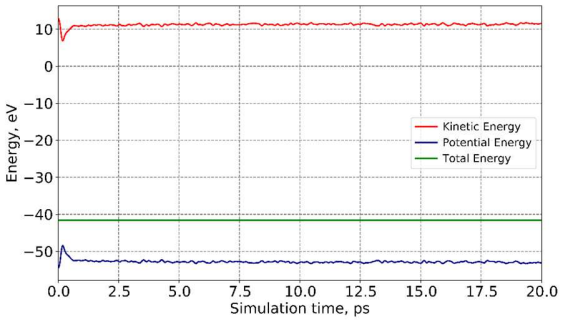

Figure 14 – Kinetic, potential and the total energy of the simulated system of 1000 Ar atoms at the starting temperature of 100 K against the simulation time

One can notice relatively small spikes of kinetic and potential energies during the first 1 ps of the simulation, but after that, both kinetic and potential energies are quite stable. Similar behavior is observed for the system's temperature (*Figure 15*):

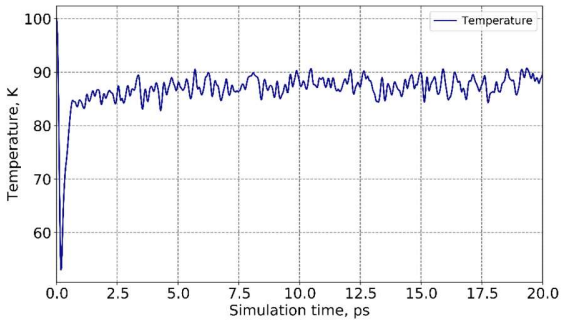

Figure 15 – Temperature of the system against the simulation time

Initial velocities are scaled to give the desired temperature, but it changes rapidly at the start of the simulation, as the system tries to equilibrate itself. After a few picoseconds, the temperature stabilizes at approximately 87 K.

At the last MD step (t = 20 ps) the system seems to be in the liquid state (*Figure 16*):

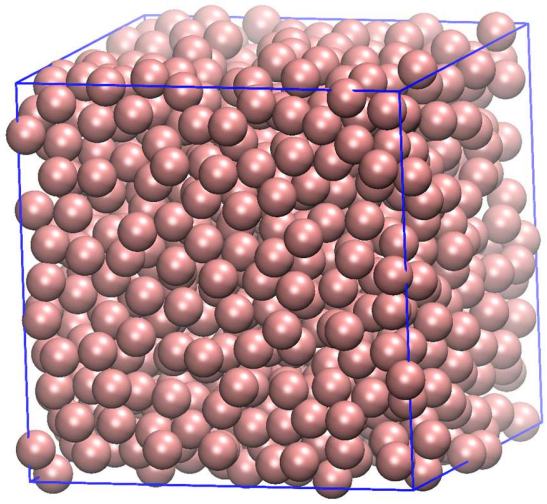

Figure 16 – The simulated system at t = 20 ps. It appears to be in the liquid state

The output trajectory was saved in *xyz* format, and a movie was made from it using *VMD*.

3.2 Performance

Using *NumPy* (handles matrix operations) and *Numba* (compiles the *Python* functions) gives a considerable speed-up – on the order of 200-300 times faster calculations than with *pure Python*. Additionally, using *NumPy* arrays instead of *Python* lists saves memory usage, as *NumPy* arrays need 4-5 times less memory to store the same amount of data. Moreover, *Numba* provides the possibility to parallelize the compiled functions. Attempt was made to parallelize force calculation function, but only 1.35x speed-up was achieved using 4-core CPU against the single-threaded MD, which certainly is not very effective, but still faster than running on one CPU core. On my home PC with AMD Ryzen 3 1300X quad-core CPU, it took about 29 minutes to do the MD simulation of 1000 Ar atoms for 20 ps with 2 fs time step.

Conclusion

Newly added features (initial conditions generator and periodic boundary conditions) are shown to be working properly and together with the code optimization (using *NumPy* and *Numba*) it is now possible to use the present code to simulate the dynamics of the Lennard-Jones fluids.

Appendix

The code and the MD movie are provided separately via Athena.