Delft University of Technology

Computational Physics
AP3082

Project 1
Molecular Dynamics with Argon

Authors:
Pol de Dalmau Huguet (5414024)
Alberto Gori (5255740)
Matteo De Luca (5388783)

March 21, 2021

Abstract

This study investigates the behaviour of Argon at different conditions and thermodynamic phases with a computer simulation coded in Python. Our simulation enabled us to reproduce the microscopic be compute some observables that we compared with literature results. After running a large number of simulations we conclude that our code can partially reproduce the behaviour of Argon, giving macroscopic observables within one order of magnitude compared to the litterature.

Contents

1 Introduction

Molecular systems are often composed by a very large number of particles, making it impossible to determine most of their properties analytically. Therefore, computer simulations are needed to study the behaviour of such systems. Our case study is the Argon gas, which is a monoatomic noble gas whose weak interactions can be approximated by the Lennard-Jones potential. To implement the computer simulation, we decided to code in Python.

Our goal was to study the system at different conditions and thermodynamic phases. Thus, in our simulation we needed to be able to set the system in equilibrium depending on different parameters such as density, temperature and number of particles. To validate our simulation we also compared it to literature, and to do so we computed some macroscopic quantities of the system such as the pressure and the pair correlation function.

This project was useful to deepen our knowledge on phase transitions and different phases of atomic systems. But mostly it shows that, approximating physical equations, computer simulations can be extremely powerful to simulate systems that would be otherwise impossible to analyze.

2 Methods

2.1 Dimensionless units

The sizes of different quantities in the simulation can vary over several orders of magnitude, leading to complex calculations. In order to simplify the code we used dimensionless units, dividing the original quantities by values that are characteristic of the system [1]. We used $\sigma = 3.405\mathring{A}$ for positions, ϵ such that $\epsilon/k_B = 119.8K$ for the energy, we then derived $(\frac{m\sigma^2}{\epsilon})^{\frac{1}{2}}$ as the natural unit for the time and $\sigma/(\frac{m\sigma^2}{\epsilon})^{\frac{1}{2}} = \sqrt{\frac{\epsilon}{m}}$ for the velocity; we also took the mass of the Argon atoms to be 1 in dimensionless units. Starting from these natural units we found the other quantities in dimensionless units, for example the Lennard-Jones potential $U(r) = 4\epsilon\left(\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right)$ became $\tilde{U}(\tilde{r}) = \frac{U(r)}{\epsilon} = 4(\tilde{r}^{-12} - \tilde{r}^{-6})$ with $\tilde{r} = \frac{r}{\sigma}$ (see Figure 1) and the force corresponding to the Lennard-Jones potential $\tilde{F} = -24\tilde{\mathbf{x}}(2\tilde{r}^{-12} - \tilde{r}^{-6})$ with $\tilde{\mathbf{x}} = \frac{\mathbf{x}}{\sigma}$. Using $\epsilon/k_B = 119.8K$ we used dimensionless temperatures with 119.8K as the natural unit for T. Similarly, we derived all the other quantities in dimensionless units.

2.2 Algorithms for time evolution

To simulate the time evolution of particles, we implemented two different algorithms.
The first was Euler's algorithm, for which the time evolution of the particle is described as

$$\mathbf{x}_i(t_n + 1) = \mathbf{x}_i(t_n) + \mathbf{v}_i(t_n)h \tag{1}$$

$$\mathbf{v}_i(t_n + 1) = \mathbf{v}_i(t_n) + \mathbf{F}(\mathbf{x}_i(t_n))h \tag{2}$$

where h is the time between step t_n and $t_n + 1$ and F is the force given by the Lennard-Jones potential.
The other one is Verlet's algorithm [2] that is a more suitable algorithm for simulations in the

microcanonical ensemble, where energy should be conserved. The time evolution of the particle is therefore described as

$$\mathbf{x}(t+h) = \mathbf{x}(t) + h\mathbf{v}(t) + \frac{h^2}{2}\mathbf{F}(\mathbf{x}(t)) \tag{3}$$

$$\mathbf{v}(t+h) = \mathbf{v}(t) + \frac{h}{2}(\mathbf{F}(\mathbf{x}(t+h)) + \mathbf{F}(\mathbf{x}(t))) \tag{4}$$

In these equations and for the rest of the paper we are working in dimensionless units as we already explained in the previous paragraph.

2.3 Periodic boundary condition and image convention

In reality, thermodynamic systems consist of numbers of particles of the order of Avogadro's number. Unfortunately, simulations considering such large numbers can not be done within reasonable time and computational power. We instead chose to impose periodic boundary conditions on a box of size $L \times L \times L$. This can be seen as an equivalent case in which we consider only a few particles (compared to Avogadro's number) that are surrounded by other boxes with images of themselves. Then, to compute forces between particles, the minimum image convention is used. This consists of computing the force on a particle by adding the forces exerted on it by all other particles' image that are nearest to said particle [1].

2.4 Harmonic behaviour

Before running simulations with large numbers of particles, it is sensible to simulate a simple case for which the outcome is known, taking advantage of the valley in the Lennard-Jones potential around $r/\sigma \approx r_m$ (see Figure 1), where $r_m = 2^{1/6}\sigma$ which is the radius at which the potential has its minimum. Around this region, the particles are subject to a harmonic potential, hence simple oscillation must be observed. Initializing the system with two particles at rest and an initial relative distance of 1.9, the plot in figure 2b can be obtained. In a first attempt, one should initialize the particles in the center of the box, and make the box large such that the particles do not cross the boundary. Then, in subsequent runs, the particles can be initialized near the boundary. If the behaviour is exactly the same, it is verified that the minimum image convention is functioning as intended.

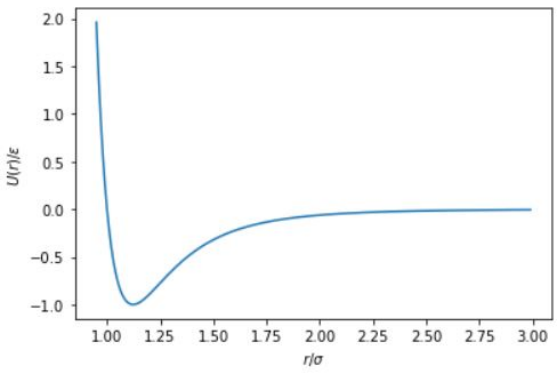
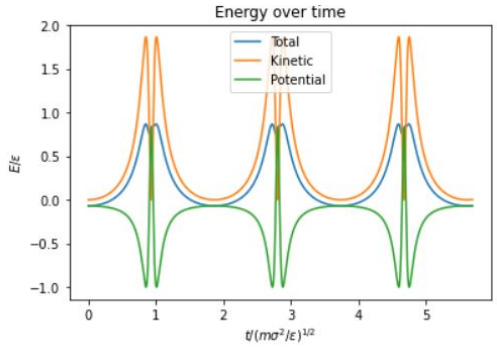
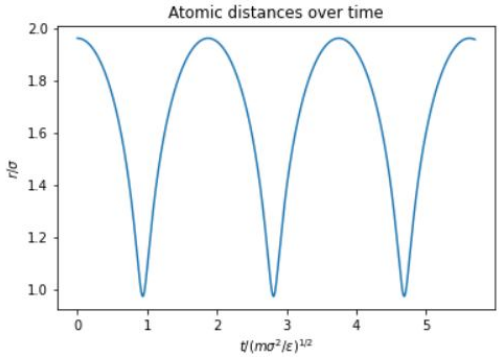


Figure 1: Lennard-Jones potential in natural units as a function of relative distance.



(a) Kinetic, potential and total energy of two particles in normalized units. Total energy is not conserved.



(b) Relative distances in normalized units.

Figure 2: Harmonic behaviour for a system initialized with two particles.

2.5 Initialization with Maxwell-Boltzmann distribution

To reproduce a realistic scenario where the velocities of the particles are determined by the temperature of the system, we initialized them such that they obey a Maxwell-Boltzmann distribution. Then, in order to keep the atoms from drifting, we proceeded to subtract the velocity of the center of mass, so that the average velocity is zero in all directions. With 100000 particles, an initial temperature of T=300K and 20 bins we found the plot of Figure 3, which is compatible with a Maxwell-Boltzmann distribution.

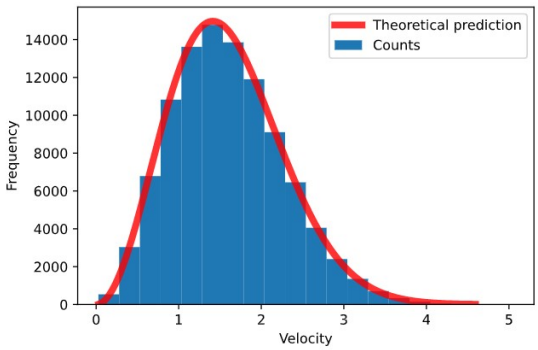


Figure 3: Distribution of the initial velocities of 100000 particles at 300K with 20 bins.

2.6 FCC lattice generation

The simulated Argon will need to be initialized with a high number of particles. Choosing random positions within our simulation box puts us at risk of placing two particles too close to one another making the initial potential energy very high, and thus making our system unstable. This motivates us to initialize the atoms in an fcc lattice given that this is the lattice structure for solid Argon [1]. From the geometry of the lattice and the box size, it is easy to determine how many particles can fit inside the volume L^3 at most, that is the number of unit multiplied by four. Four is the number of atoms that fit inside an fcc unit cell, and the number of cells is simply $(L/a)^3$

where L is the box size and a the unit cell size length (not the lattice constant!). In addition to plotting the fcc to verify it is initialized as required (see Figure 4), the function that calculates atomic distances can be used. We find that the maximum relative distance between atoms is $\sqrt{3}L/2$ and the minimum is $a/\sqrt{2}$. We use the example of Figure 4 (L= 100, a=L/4) and see that indeed, the maximum distance: $86.602 = \sqrt{3}L/2$, minimum distance: $17.677 = a/\sqrt{2}$.

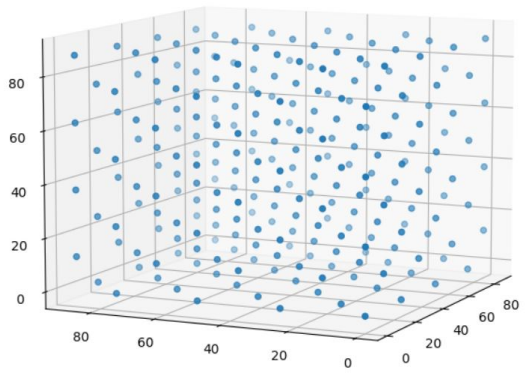


Figure 4: FCC lattice with 256 particles. $L = 100$, $a = L/4$. Note that the axes do not reach the length of 100 since we use the boundary condition.

2.7 Temperature re-scaling

When setting the initial velocities with a Maxwell-Boltzmann distribution we define a temperature T for the system. Anyway, it is not that simple to set a fixed temperature, as the particles will exchange kinetic and potential energy to equilibrate. Since it is hard to predict this behaviour, we can control it by forcing the kinetic energy to have a value corresponding to a certain temperature. This is done by rescaling the velocities of a factor λ which, in dimensionless units, is:

$$\lambda = \sqrt{\frac{(N-1)3T}{\sum_i {v_i}^2}} \tag{5}$$

Thus, we let the system equilibrate for a small period of time after which we start rescaling. The rescaling stops when the system reaches equilibrium. This can be seen when λ is close to 1, so in the code we stop the rescaling if $|l - 1| < 0.01$.

2.8 Computation of observables

With all the steps described above, we had a simulation capable of describing the microscopic behaviour Argon atoms. Our next goal was to compute the macroscopic observables of the system. To do that, we used statistical averages that were implemented averaging over simulation time in the following way:

$$\langle A \rangle = \frac{1}{n - n_0} \sum_{\nu > n_0}^{n} A_\nu \tag{6}$$

where ν labels the timestep of the numerical integration, n is the total number of timesteps and n_0 is the timestep at which we reached equilibrium. In fact, we only start computing the observables after the system has reached a thermodinamical equilibrium, while changing temperature, particle number and density for different simulations. The observables we computed are the pair correlation function and the pressure of the system.

2.8.1 Pair correlation function

Pair correlation function is a measure of how many particles we have per unit volume, as a function of the distance from a reference particle. As explained in section 2.6 we set the initial positions of the Argon atoms in a FCC lattice. This means that if the system is in a solid state we expect the particles to be at a fixed distance, which should be a function of the lattice constant. Instead, for a liquid or gaseous system, the function should be smoother.
The pair correlation function was computed as follows:

$$g(r) = \frac{2V}{N(N-1)} \frac{\langle n(r) \rangle}{4\pi r^2 \Delta r} \tag{7}$$

where V is the volume of the simulation box, N is the number of particles, r is the relative distance between particles and $n(r)$ is the number of particle pairs with respective distances within r and $r + \Delta r$.
It is interesting to analize this function for different densities and temperatures to understand the different thermodinamical phases of the system.

2.8.2 Pressure

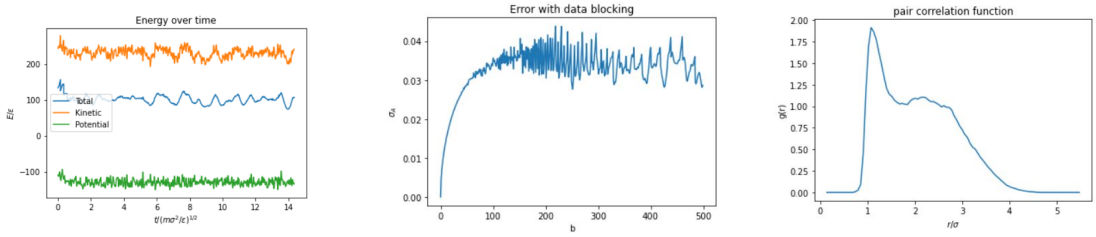
Pressure was computed (in dimensionless units) as follows:

$$\frac{\beta P}{\rho} = 1 - \frac{\beta}{3N} \left\langle \frac{1}{2} \sum_{i,j} r_{ij} \frac{\partial U}{\partial r_{ij}} \right\rangle \tag{8}$$

where β is $\frac{1}{K_b T}$ where K_b is the Boltzmann constant, T the temperature of the system at equilibrium, and U the potential energy. This is a useful observable to compare with literature.

3 Results

Figure 5 below shows the plot of energy, error with data blocking and pair correlation function for a system with 50 particles at T = 3 and density of $\rho = 0.3$. The total energy oscillates over time but it's average stays constant. Also, we could obtain a pressure $P = (1.42 \pm 0.04)$ with the error of 0.4 coming from data blocking. Finally and the pair correlation function shows the behaviour of a gas, as expected from such density and temperature.



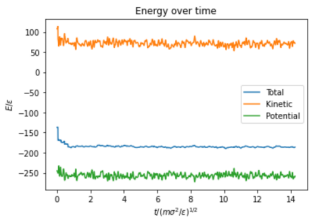
(a) Kinetic, potential and total energy.

(b) Error in the pressure with data blocking. The calculated pressure is $P = 1.417$.

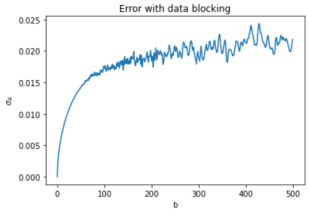
(c) Pair correlation similar to that of a gas. Compare to figure 8.

Figure 5: Simulation results for 50 particles at $T = 3$ and a density of $\rho = 0.3$ in reduced units.

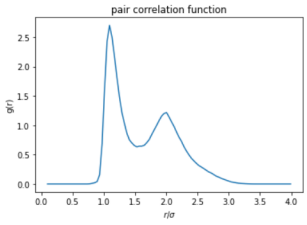
Figure 6 shows the results of a simulation with 50 particles at $T = 1$ (119.8K) and density $\rho = 0.53$. We find that the total energy of the system is approximately conserved and the pair correlation function is compatible with that of a liquid since the second peak is smooth. The calculated pressure is $P = (1.02 \pm 0.02)$ with an error of about 0.02 as we can see from Figure 6b.



(a) Kinetic, potential and total energy.



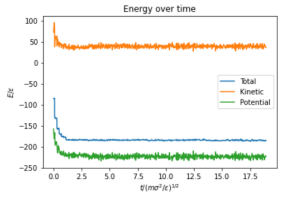
(b) Error in the pressure with data blocking. The calculated pressure is $P = 1.02$.



(c) Pair correlation similar to that of a liquid. Compare to figure 8.

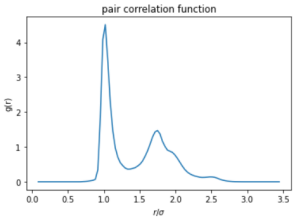
Figure 6: Simulation results for 50 particles at $T = 1$ and a density of $\rho = 0.53$ in dimensionless units.

Figure 7 below shows the plot of energy, error with data blocking and pair correlation function for a system with 50 particles at T = 0.5 and density of $\rho = 1.2$. The total energy barely oscillates over time and it's average stays co1stant. Also, we could obtain a pressure $P = (1.80 \pm 0.01)$ with the error of 0.01 coming from data blocking. Finally, the pair correlation function somewhat shows the behaviour of a solid, but the peaks are not well defined. We can see a third peak emerging compared to the plot for a liquid, but the difference is subtle.



(a) Kinetic, potential and total energy.

(b) Error in the pressure with data blocking. The calculated pressure is $P = 1.8$.



(c) Pair correlation similar to that of a solid. Compare to figure 8.

Figure 7: Simulation results for 50 particles at $T = 0.5$ and a density of $\rho = 1.2$ in reduced units.

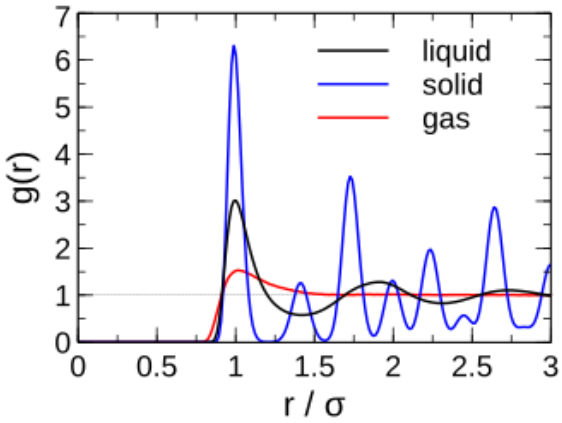


Figure 8: The radial distribution functions of solid (T = 50 K), liquid (T = 80 K), and gaseous argon (T = 300 K). The radii are given in reduced units of the molecular diameter ($\sigma = 3.822$Å). Taken from [3].

Figure 8 shows pair correlations for different phases. The pair correlation plots determined in our simulations are unable to show this many peaks since we simulate a volume containing less unit cells. For this reason, the periodicity is less obvious to see. As a consequence, it is more difficult to draw sound conclusions about what the phase is. One noteworthy observation are the peaks of the pair correlation function matching the lattice constant of the solid argon and multiples of it. Other peaks are to be seen at multiples of $l\sqrt{2}$, where l is the lattice constant. These correspond to particles on the faces and on the corners of the fcc unit cell respecively.

The above measurements and more that were taken can be summarized in the following table:

Table 1: Results for the pressure from this study compared to those from Verlet [2]

ρ	T	This study's $\beta P/\rho$	Verlet $\beta P/\rho$
0.3	3	1.417 ± 0.018	Na
0.53	1	1.02 ± 0.02	Na
1.2	0.5	1.8 ± 0.01	Na
0.5	1.36	1.3 ± 0.03	3.40
0.85	0.591	577 ± 0.04	-0.18
0.85	0.88	1.08 ± 0.02	1.64

4 Conclusions

After running a large number of simulations and comparing the data to the literature, we can conclude that our code can only partially simulate the behaviour of an Argon gas.
The first problem is undoubtedly that the energy inside the system is not perfectly conserved. This also holds for the simple case of two particles and no rescaling of the velocities. In fact, the kinetic energy seems to oscillate to slightly higher peaks than the potential energy, but we were not able to understand how, or why.
To check for possible mistakes we verified by hand, using a small number of particles (usually 4 or 5), that each function returned the expected value. The functions for kinetic and potential energy both returned the expected values. Also, the forces on all the particles summed up to zero for each time step as expected.
One possible source of errors might be that we initialize the velocities with a Maxwell-Boltzmann distribution which makes sense for a large number of particles, while we always simulated with less than 100. This may not explain why energy isn't conserved after equilibrium.

Equilibrium in the simulations could be reached nonetheless, and we were able to compute the pair correlation function and the pressure of the system. They both gave reasonable results, even if they do not match the literature perfectly.
Finally, the performance of the code allowed us to run simulations up to 100 particles and up to 20000 time steps in about 30 minutes. Instead, simulations with 30 particles could be run in just a few minutes. As this was our first time programming in Python, we used numpy arrays containing numpy arrays to store the variables needed for the simulation instead of using a list containing arrays. This is probably the main reason why our simulation is not very fast even if we used numpy functionalities whenever possible, trying to avoid unnecessary for loops, which reduce performance because they check data types at each iteration.

References

[1] J. Thijssen, *Molecular dynamics simulations*, 2nd ed. Cambridge University Press, 2007, p. 197–262.

[2] L. Verlet, "Computer "Experiments" on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules," *Phys. Rev.*, vol. 159, pp. 98–103, Jul 1967. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRev.159.98

[3] Wikibooks, "Molecular simulation/radial distribution functions — wikibooks, the free textbook project," 2020, [Online; accessed 19-March-2021]. [Online]. Available: https://en.wikibooks.org/w/index.php?title=Molecular_Simulation/Radial_Distribution_Functions&oldid=3710016