A molecular dynamic simulation for a system interacting via Lennard-Jones potential

Smit Chaudhary, Ignacio Fernández Graña, Georgios Sotiropoulos

March 21, 2021

Abstract

Molecular dynamics simulations are key in the study of the properties and structure of complex molecular systems consisting of many particles. In this work we implement a molecular simulation of a simple system of atoms interacting via the Lennard-Jones potential. Argon is an example of such system. To demonstrate the thermodynamic properties of the system, two observables are studied: the pair-correlation function and the pressure. Through these quantities we investigate the system at different phases and compute the first virial coefficient. We also studied compressibility factor for system at various values and cross-refernced with known values from literature.

1 Introduction

Molecular systems normally consist of a very large number of particles with complex interactions between them. This makes it impossible to determine certain properties analytically. Therefore, computer simulations are needed to study the behaviour of such systems. A molecular dynamics simulation allows us to infer macroscopic physical attributes of the system as well as the detailed molecular level information. A relatively easy case study is the Argon gas. Argon is a monoatomic noble gas with weak interactions that can be approximated by the Lenard-Jones potential[5]. The simple single-atom molecular structure of Argon and the Lennard-Jones interaction make it an ideal candidate for molecular simulations. The Lennard-Jones potential has the following form:

$$U(r) = 4\epsilon \left(\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right) \tag{1}$$

The Lennard-Jones potential has a soft repulsive core and a long attractive tail which models the dipole interactions between neutral atoms such as Argon.

The way particles evolve in time is given by Newton's second law of motion for each particle $i = 1, \ldots, N$:

$$m\frac{d^2\mathbf{x}_i}{dt^2} = -\nabla U(\mathbf{x}_i) \tag{2}$$

when N is very large, it is not possible to solve this system analytically and one has to make use of numerical methods to get an approximated solution. Solving this system of equations gives the the positions and velocities for every particle, allowing us to study microscopic and macroscopic quantities of the system. In this paper we will mainly study two thermodynamic quantities: the pair-correlation function and the pressure.

This paper is structured as follows. In Section 2 we present the simulation methods we use and the physical quantities that we study. In section 3 our results are presented and discussed. We conclude our work in section 4.

2 Methods

2.1 Dimensionless Units

Throughout this work we will make use of a system of units known as *dimensionless units*. Any physical quantity can be converted to an equivalent dimensionless quantity by way of a scaling factor. Dimensionless units themselves can always be converted back to the real, physical units at any time. Thus no information is lost by using dimensionless units but there are some important advantages of working in dimensionless units. The primary advantage being we do not really need to handle unnecessary floating point numbers and suffer

due to finite precision of the same. Another advantage being, we get an insight into the physics in terms of the length scales and other typical dimensions at the atomic scales.

In this dimensionless units we express the energies in units of ϵ the positions in units of σ and the time in units of $(\frac{m\sigma^2}{\epsilon})^{1/2}$. Further, we express temperature in the units of $(\frac{\epsilon}{k_B})$ where k_B is the Boltzmann's constant. Once we define these quantities in dimensionless units, all quantities would be dimensionless with appropriate scaling factors. Hereafter, all quantities are being expressed in dimensionless units, unless otherwise specified.

2.2 ODE integrating methods - Verlet's algorithm

There are many algorithm one can use to solve a system of equations like the on presented in equation (2). We have implemented two of them: *Euler's* method and *Verlet's* method. The former is a first-order integration scheme, so the total error is proportional to the step size, while the latter is a second-order scheme. Moreover, Euler's method main problem is that it does not work well for systems where the energy must be conserved, which is our case. Therefore, our experiments have been carried out with the Verlet's method. which is part of a more general class of algorithms called *symplectic algorithms* [**symplectic**]. In this case, if we call $\mathbf{x} = \{x_0, x_1, \ldots, x_N\}$, $\mathbf{v} = \{v_0, v_1, \ldots, v_N\}$ the positions and velocities for every particle, then the solution of equation (2) for a time $t + h$ reads:

$$\begin{aligned} \mathbf{x}(t+h) =& \mathbf{x}(t) + h\mathbf{v}(t) + \frac{h^2}{2}\mathbf{F}(\mathbf{x}(t)) \\ \mathbf{v}(t+h) =& \mathbf{v}(t) + \frac{h}{2}\left(\mathbf{F}(\mathbf{x}(t+h)) + \mathbf{F}(\mathbf{x}(t))\right) \end{aligned} \tag{3}$$

2.3 Setting up simulation parameters

We now know how to simulate the dynamics of the different particles of the system. Nevertheless, we want to study the system at different conditions and thermodynamic phases. Therefore, our simulation must be able to set the system in equilibrium at different parameters. Namely, our control parameters are going to be temperature and density.

Density: The simulation must be done on a finite volume in order to be computable. For this reason we will restrict ourselves to a three dimensional box containing a number of particles with periodic boundary conditions. We follow the minimal image convention, i.e., we will take only the force due to the nearest image of other particles. We just consider the closest of all 'image partners' (plus the original) of every atom for calculating interaction. For a constant number of particles, tweaking the dimensions of the simulation box allows us to precisely set a density. Note that as an initial placement the particles are distributed homogeneously in the box arranged in an fcc lattice (see figure 1).

Temperature Temperature is directly connected to the average kinetic energy of all particles over time. Initializing the system according to Maxwell-Boltzmann distribution [5] will satisfy this average behavior over all particles but not over time. Upon thermalization this average kinetic energy may have changed though due to energy exchanges between potential and kinetic energy. To attain the desired input temperature we need to then suitably rescale the velocities of all particles, which is equivalent to adding or subtracting energy to the system. In figure 2 the rescaling effect on total energy is clearly visible right after initialization. Iteratively following this procedure we are able to set our system to the desired temperature. Note that the desired temperature is only approximately achieved within a certain tolerance, so deviations may be present.

2.4 Observables calculation

In order to study the thermodynamic behaviour of the system, we study two different observables: the pair correlation function and the pressure. The pair correlation function describes how density varies as a function of distance from a reference particle. Observables are measured by taking its time average over a significant number of time steps while the system is in equilibrium. For a quantity A, this time average is computed as:

$$\langle A \rangle = \frac{1}{n - n_0} \sum_{\nu > n_0}^{n} A_\nu \tag{4}$$

where n_0 is the number of time steps needed to get to equilibrium and n the total number of simulation steps.

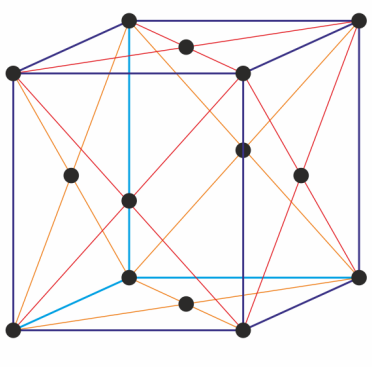

Figure 1: FCC lattice unit box[2]

The pair correlation function is defined as:

$$g(r) = \frac{2V}{N(N-1)} \frac{\langle n(r) \rangle}{4\pi r^2 \Delta r} \tag{5}$$

where $\langle\rangle$ represents aveage over time. The pressure can be computed as:

$$\frac{\beta P}{\rho} = 1 - \frac{\beta}{3N} \left\langle \frac{1}{2} \sum_{i,j} r_{ij} \frac{\partial U}{\partial r_{ij}} \right\rangle \tag{6}$$

Lastly, in order to measure the error of the observables we make use of the Pearson correlation coefficient [7]. For a general observable A, measured in a finite time, this reads:

$$\chi_A(t) = \frac{(N-t)\sum_n A_n A_{n+t} - \sum_n A_n \times \sum_n A_{n+t}}{\sqrt{(N-t)\sum_n A_n^2 - \left(\sum_n A_n\right)^2} \sqrt{(N-t)\sum_n A_{n+t}^2 - \left(\sum_n A_{n+t}\right)^2}} \tag{7}$$

for $1 \leq n \leq N-t$, with N being the number of simulation steps and t the current simulation step. If we fit $\chi_A(t)$ to an exponential $e^{-\frac{t}{\tau}}$, being τ the correlation time, then the error will be given by:

$$\sigma_A = \sqrt{\frac{2\tau}{N} \left(\langle A^2 \rangle - \langle A \rangle^2\right)} \tag{8}$$

3 Results

3.1 Correctness checks

In order to check the validity of our simulations, we check the results both in a a qualitative sense against our physical intuition and in quantitative sense against literature. The first and most vital thing that needs to be demonstrated is total energy conservation, as we are simulating a closed system. Figure 2 shows the energy for a simulation with $\rho = 0.5$ $T = 1.36$. One can see the total energy fluctuating due to rescalings and staying constant once equilibrium is reached. Thus we are sure that the simulations do not violate the conservation of energy.

Furthermore, we may compare some of the results obtained in our simulations with the values in the literature. In table 1 a comparison is presented between the values of the compressibility factor $\beta p/\rho$ for different conditions as obtained by Verlet[6] and as produced in our work.

One can check that the higher the temperature, the more distant are our values with respect to the literature. This makes sense, as the higher the temperature the higher the velocities are, and thus our time discretization will perform worse, specially when having few particles.

3.2 Pair correlation function

Pair correlation function is a measure of how many particles per unit volume we get from a particle as a function of the distance from this reference particle. As noted in sec 2.3, we initialise our system on an FCC lattice. If the system is in solid phase, one expects the particles to be relatively fixed in space and the pair correlation

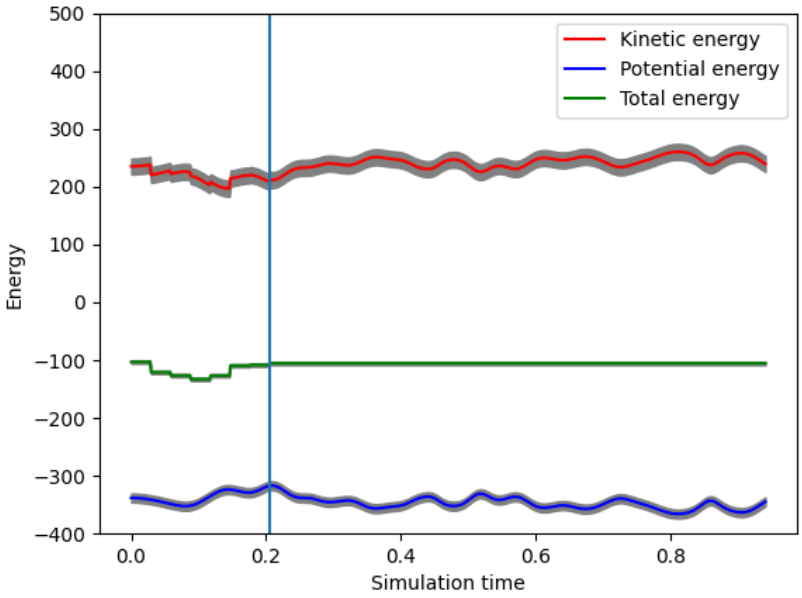

Figure 2: Energy over time for $\rho = 0.5$, $T = 1.36$ in dimensionless units. The vertical line represents the point where equilibrium is reached and the rescaling is not done anymore, thus the total energy stays constant. The shaded region denotes the values within the error interval.

		Verlet's paper	Our implementation
ρ	T	$\beta p/\rho$	$\beta p/\rho$
0.88	1.095	3.44	2.72 ± 0.47
0.75	1.304	1.61	1.913 ± 0.034
0.65	2.557	2.14	1.96 ± 0.26
0.65	0.900	-0.74	-0.690 ± 0.037
0.35	1.448	0.40	0.672 ± 0.085

Table 1: Comparison between the values obtained in [6] with 864 atoms and our results. Our simulations were done with 108 atoms with a 10^{-3} timestep.

function would exhibit peaks at certain distances based on the lattice constant. On the other hand, for a system in liquid and gaseous phase, one would find the pair correlation function to be rather smooth.

According to values given in [5] the Argon system is in solid phase for $\rho = 1.2$ and temperature $T = 0.5$, in liquid phase for $\rho = 0.8$ and $T = 1$ and in gas phase for $\rho = 0.3$ and $T = 3$. Figure 3 shows the pair correlation function obtained under the mentioned values of ρ and T for solid, gas and liquid phase. The simulations were done with 108 atoms and a time step of 10^{-3} units. The results validate the qualitative argument laid out earlier about the smoothness of the function for different phases.

Additionally, one can also look at the positions of the peak in solid phase. In a solid, the particles are relatively immobile and only jiggle about their position. We denote the lattice constant of the FCC lattice as l. The closest pair of particles are those such that one is at the corner of the lattice and the other is at one of the corresponding face. For reference see figure 1. The closest points are the ones at the end and middle of the red (or orange) lines. They are $\frac{l}{\sqrt{2}}$ apart. The next closest pair would be those along the same edge of the lattice. This corresponds to those at the ends of the purple (or blue) in figure 1. These points are distance l units away. One could further see that the next closest pair is the particles along the face diagonal. Those are $l\sqrt{2}$ distance apart. The next closest particle to the one at a corner would be the particle at the face of the adjacent cube. This is $\frac{l\sqrt{10}}{2}$ units apart. One can go an and note the particle at the body diagonal as the next closest particle. This would be at $l\sqrt{3}$ units away.

Figure 3 aptly exhibits these peaks for the solid phase. One can also see that for larger distance the pair correlation function dies out in figure 3. The pair correlation function actually does not go to zero for distances about 3 times the lattice constant since we do expect to have particles such at such distances for all phases.

But in our simulation, the dimension of the simulation region is about three times the lattice constant. Thus, the dying tail in figure 3 is the artefact of the finite simulation region and not physically accurate.

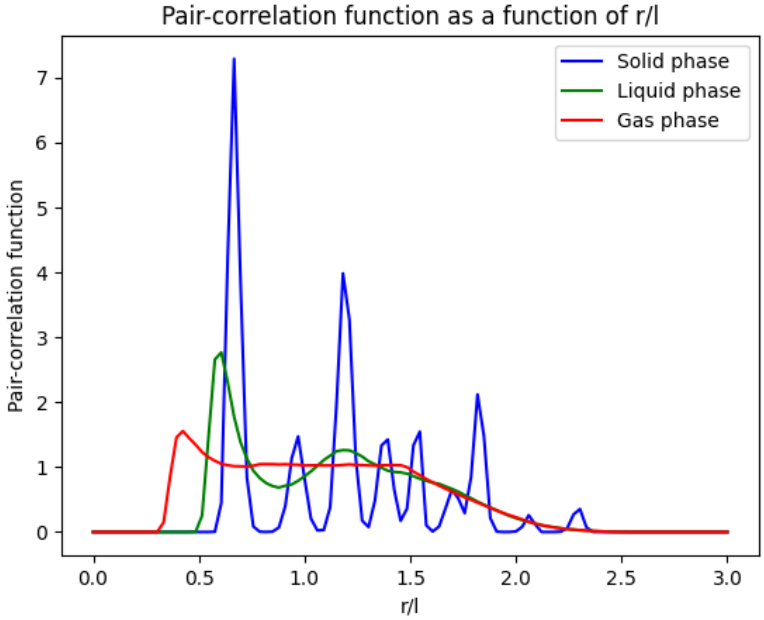

Figure 3: Pair correlation function as a function of $\frac{r}{l}$ (in dimensionless units) for solid, liquid and gas phase. r is the radial distance and l is the lattice constant. The simulations were done with 108 atoms and a 10^3 timestep. For solid phase we took $\rho = 1.2, T = 0.5$; for liquid phase $\rho = 0.8, T = 1$ and for gas phase $\rho = 0.3, T = 3$ [5]

3.3 Pressure

In order to obtain the pressure behaviour of the Lennard-Jones fluid[4] we have to keep one of the thermodynamic quantities that we control constant while scanning a range with the other one. As we mentioned in our methods, we can precisely control density but not temperature. Thus, we shall fix density and vary the system temperature. This way, we are able to set a temperature but deduce the actual temperature with greater certainty, by averaging over the time period of established equilibrium.

We will investigate how the fluid behaves while being in the gaseous phase. As we found out, for a density of $\rho = 0.3$ and a temperature of $T = 3$ it is indeed a gas.

Therefore, we will take values in the interval $T \in [3, 5]$ while keeping the density at $\rho = 0.3$. In figure 4 our results are presented.

We can clearly see that our data follows a linear relation and extract the necessary coefficients:

$$a = 0.518 \pm 0.007 \tag{9}$$

$$b = -0.6 \pm 0.03 \tag{10}$$

In the low density limit with high temperature a Lennard-Jones fluid behaves in dimensionless units as[4] :

$$P = T[\rho + (\beta - \frac{\alpha}{T})\rho^2] \tag{11}$$

$$P = T(\rho + \beta\rho^2) - \alpha\rho^2 \tag{12}$$

The first virial coefficient is then[4]:

$$a_2 = \beta - \frac{\alpha}{T} \tag{13}$$

$$\tag{14}$$

Knowing exactly the value of density that we used for the simulation ($\rho = 0.3$) we can deduce from the linear fitting that for the slope a:

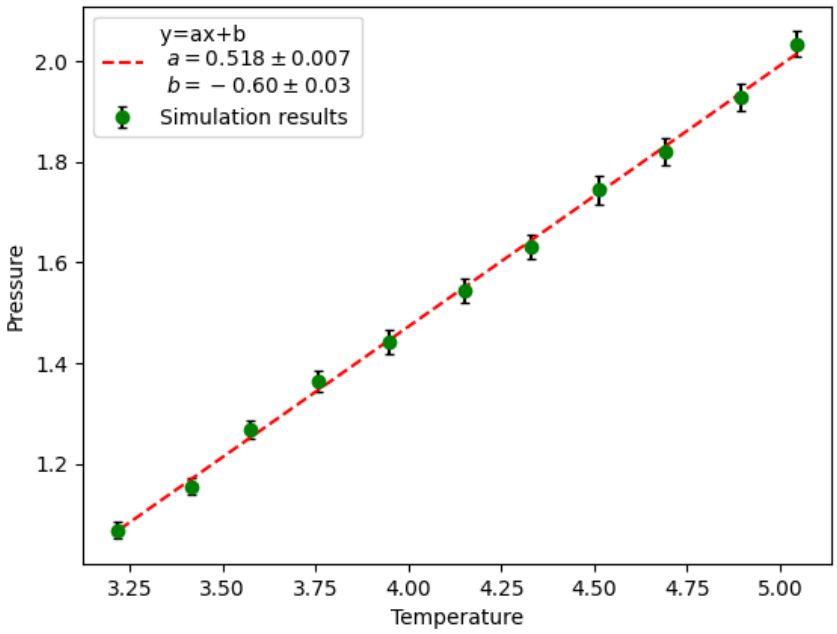

Figure 4: Pressure against temperature in natural units for $\rho = 0.3$ (gaseous phase) with 108 atoms.

$$a = \rho + \beta\rho^2$$
$$\beta = \frac{a + \rho}{\rho^2}$$
$$\beta = 9.09 \pm 0.08 \tag{15}$$

For the intercept b:

$$b = \alpha\rho^2$$
$$\alpha = \frac{b}{\rho^2}$$
$$\alpha = 6.6 \pm 0.3 \tag{16}$$

Hence, we obtained through the simulations an expression for the first virial coefficient.

3.4 Performance of the code

Real molecular systems have a large number of particles. This means that a very large number of interactions have to be computed at each step, and therefore the efficiency of the code plays a vital role when simulating relatively big systems[1]. For this reason, we briefly comment in this section the efficiency of our implementation.

In this kind of molecular simulations, one frequently has to loop over all the particle of the system. Doing a `for loop` is highly inefficient in this case. A more convenient way to carry out these operations is to vectorize them and use built-in libraries such as `NumPy`[3]. Numpy's main object, `numpy.arrays`, allows for a very efficient implementation of operations between arrays. Thus, we used `numpy.arrays` wherever applicable. The simulation time for a system of 108 atoms with a time step of 10^{-3} and for 10^5 steps was under a minute on our laptops, which made possible for us to run all the simulations we needed.

For even larger number of atoms, namely, for 4000 atoms and a time step of 10^{-3} and for 10^5 steps, the simulation finished in about 18 minutes. Note that, with a large number of particles, since the Lennard-Jones potential decays rapidly, the particles much farther away than the several times the lattice constant do not play a significant role. This fact, combined with the periodic boundary conditions imply that our simulations with significantly less atoms than an actual physical system ($\sim 10^{24}$ particles) is still reasonably accurate.

4 Conclusions

In this work we have studied a system of Argon atoms through a molecular simulation implemented with Verlet's algorithm. The correctness of our implementation was checked through literature cross-reference [5, 6]. Then, two main observables were studied in depth, the pair-correlation function and the pressure. Through the study of the pair-correlation function we could check that in the solid phase the particles were indeed structured in a FCC lattice. For gas and liquid phases the particles were more loosely distributed around the simulation box as expected. By studying the pressure for the gaseous phase (low density limit), we were able to calculate an expression for the first virial coefficient. Finally, we reflected about the efficiency of our code. Overall, we have implemented an efficient molecular simulation that can be used for any system of atoms with interactions that can be modeled through the Lennard-Jones potential, and that allows for the study of several thermodynamic quantities.

As a future outlook, our code could be improved in several ways. First of all, even though python is a very friendly-user programming language, it is not ideal for resource-intensive computations like this one. Lower level languages like C++, C or even Julia offer the possibility of implementing very efficient simulations. Furthermore, more sophisticated methods for integrating the equations of motions that take in account higher order terms could be used to improve the accuracy.

References

[1] Mark James Abraham et al. "GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers". In: *SoftwareX* 1-2 (2015), pp. 19–25. ISSN: 2352-7110. DOI: https://doi.org/10.1016/j.softx.2015.06.001. URL: https://www.sciencedirect.com/science/article/pii/S2352711015000059.

[2] E. Generalic. URL: https://glossary.periodni.com/glossary.php?en=face-centered+cubic+lattice.

[3] Charles R. Harris et al. "Array programming with NumPy". In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[4] Jos Thijssen. *Computational Physics*. 2nd ed. Cambridge University Press, 2007. DOI: 10.1017/CBO9781139171397.

[5] Jos Thijssen. *Lecture Notes Statistical Physics*. 2018.

[6] Loup Verlet. "Computer "Experiments" on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules". In: *Phys. Rev.* 159 (1 July 1967), pp. 98–103. DOI: 10.1103/PhysRev.159.98. URL: https://link.aps.org/doi/10.1103/PhysRev.159.98.

[7] Rand R. Wilcox. *Introduction to robust estimation and hypothesis testing*. 2nd ed. Academic Press, 2005.