Project Report: Computational Physics

Molecular Dynamics with Argon Particles

Authors:

Tim Neumann

César Hernando de la Fuente

Arturo Castaño Gallardo

MSc Quantum Information Science and Technology

Delft University of Technology

March 2025

Abstract

Most physical systems cannot be fully studied analytically due to factors including the difficulty (or impossibility) of resolving a certain equation, or the number of constituents in the system. One example of the latter is the dynamics of a system of particles like a gas, where the high number of particles makes any analytical computation unfeasible. Here we report our simulation of a system of Argon atoms using Verlet's algorithm. By measuring multiple observables as the energy, specific heat, pair correlation function or diffusion, we are able to verify the correctness of the simulation. Besides, when varying the temperature and density of the system, we are able to identify and compare the different phases of matter: solid, liquid, gas, as well as the triple point. As optimising the performance of a program is crucial in any physical simulation due to the limited numerical resources, we finalise our discussion by analysing the performance. We conclude that, while being highly efficient for a small number of particles, it would require certain adaptations for larger systems.

Contents

1 Introduction

Systems of multiple constituents interacting with one another, often referred to as N-body systems, have long been an object of research, both for their ubiquity in nature (think of the solar system or a spin lattice) and their scientific intrigue (sparked among many others by the chaotic behaviour seen in the 3-body problem). In this report, we investigate a system of Argon atoms under pairwise interaction, which is referred to as Molecular Dynamics (MD) simulation. In solving this N-body system numerically, we seek to gain insight into multiple emerging observables of this simulation, such as temperature, kinetic and potential energy, pair correlation, specific heat and diffusion. By defining a range of parameters that govern the behaviour of the MD simulation, we can extract information on phase transitions of Argon from those observables, which hints at the general idea of learning about certain behaviours of a system through the means of simulation in the field of computational physics.

In this report, we first provide the theoretical underpinnings of the MD simulation, after which we describe the methods employed to realize the project. Hereafter, the obtained results are presented and discussed, including an analysis of errors. We conclude by outlining the performance of our MD simulation and summarising the main findings.

2 Theory & Background

The choice of Argon as element for the MD simulation is inspired by its relatively simple molecular properties. Being a noble gas, the approximation of Argon molecules as classical point masses is justified in the scope of this project, which facilitates computational representation of particles. The following theory is largely a synthesis of material provided in [5, Ch.8].

2.1 Equations of Motion

In order to extract information from the system, after initialisation we let it evolve over time, storing system components such as positions and velocities of the particles at every time step. The time evolution of the ensemble is governed by classical Equations of Motion, often referred to as Newton's equations (see 5). The central equation herein is Equation 1, which evaluates Newton's Second Equation on a specific particle as a function of the gradient of the potential energy it experiences to yield

$$m\frac{\partial^2 \mathbf{x}_i}{\partial t^2} = \mathbf{F}(\mathbf{x}_i(t)) = -\nabla_i U(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N), \tag{1}$$

where each $\mathbf{x_i} = (x_i, y_i, z_i)$ encodes a particle of three Cartesian coordinates, and $i \in \{1, 2, \ldots, N\}$, for N particles. Recasting this in dimensionless units, we obtain

$$\frac{\partial^2 \tilde{\mathbf{x}}_i}{\partial \tilde{t}^2} = \tilde{\mathbf{F}}(\tilde{\mathbf{x}}_\mathbf{i}(\tilde{t})) = -\tilde{\nabla}\tilde{U}(\tilde{r}), \tag{2}$$

where from now on we describe all variables in their natural units, i.e., positions and distances in units of $\sigma = 3.405 \cdot 10^{-10}$ kg, energy in units of $\epsilon = 119.9\,\mathrm{K} \cdot k_B$, mass in units of $6.6 \cdot 10^{-26}$ kg, etc. Newton's Second Equation 1 in dimensionless units is the physical backbone of this project. It is clear that the MD is completely determined by the force acting between particles.

2.2 Forces

The forces in Equation 5 can be very complex in general, but we assume a relatively simplistic form, derived from the dipole-dipole interaction of Argon atoms. To gain traction on the gradient defined in Equation 1, we draw upon a convenient closed-form expression for the potential energy. The so-called Lennard-Jones potential describes the interaction between two atoms as

$$\tilde{U}_{LJ}(\tilde{r}) = 4\Big[\Big(\frac{1}{\tilde{r}}\Big)^{12} - \Big(\frac{1}{\tilde{r}}\Big)^{6}\Big], \tag{3}$$

where $\tilde{U}$ is parameterised by the inter-atomic distance $\tilde{r}$ and expressed in dimensionless units. The shape of the potential well can be seen in Figure 1.

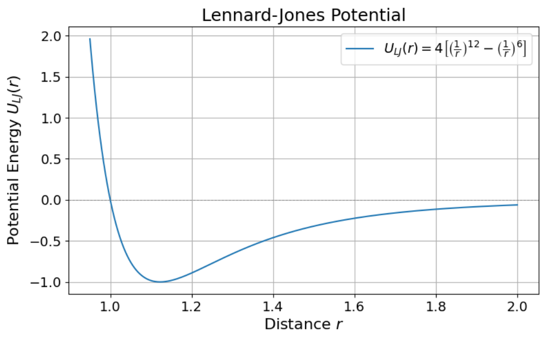


Figure 1: Lennard-Jones potential. Distance r in units of σ, potential energy U_{LJ} in units of ϵ.

Considering Equation 1, the form of the potential implies the relation

$$\mathbf{F}(x_i(t)) = -24 \sum_{\beta,\beta\neq i} \left(\frac{1}{r_{i,\beta}^8} - 2\frac{1}{r_{i,\beta}^{14}}\right)(\mathbf{x}_i - \mathbf{x}_\beta), \quad (4)$$

such that the force experienced by any individual particle depends on the distances and relative positions to all other particles (for a derivation, refer to Appendix 6.1). It shall be noted that this formulation does not take interactions between three or more atoms into account, but relies on the leading contribution of pairwise interaction.

2.3 Periodic Boundary Conditions

Placing a finite number of atoms in a system of restricted size poses the question of how to treat boundary conditions. A common choice in MD simulations is to assume periodic boundary conditions (pbc). In order to emulate this constraint, we employ the so-called minimal image convention, which effectively embeds the original system at every time step with $3^d - 1$ copies of itself, where $d = 2, 3$ denotes the dimensionality of the system, as can be seen in Figure 2. This setup may then be used to compute interatomic distances from each particle in the central box to all copies of any other given particle (distinct from the reference particle), where for distinct particle we choose the smallest particle distance among the 3^d values as true distance under pbc. The visual intuition behind this concept is that two particle positioned close to the opposite edges of the central box are not maximally separated, but of relatively close proximity under pbc, as can easily be seen through the lense of the minimal image convention.

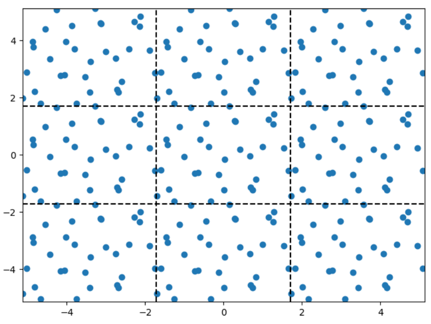


Figure 2: Embedding the initial system (centre) with $3^2 - 1 = 8$ copies of itself in order to emulate periodic boundary conditions through the minimal image convention.

Conveniently, the arithmetic expression

```
rel_pos -= ( (rel_pos + box_dim/2)
// (box_dim) ) * box_dim
```

handles the computation of the minimal image convention in one line, using *numpy* vectorization.

2.4 Ergodicity

The computation of observables implicitly relies on the assumption of ergodicity commonly employed in thermodynamics, which states that ensemble averages may be replaced by time averages. The intuitive justification of this claim is that in an ergodic system, the path of any typical atoms over time abides by the same statistical rules as the average of all particles in the ensemble at a certain time step.

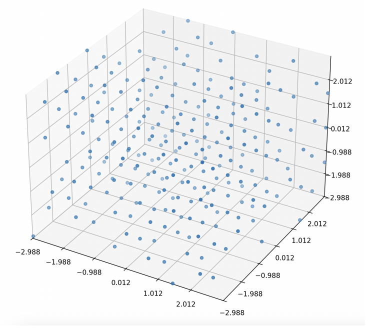


Figure 3: Ensemble of Argon atoms, where averages of a comparably small number of particles are taken over time according to the ergodic theorem.

3 Methods

3.1 Initialization of positions

A key step in our simulation of the dynamics of different phases of Argon is the initialization of the positions of the Argon atoms. First of all, it is crucial that the atoms are distant enough so that the term proportional to r^{-12} in the potential energy (see Equation 3) does not diverge. Furthermore, since we are also interested in simulating solid Argon, it is sensible to initialize the atoms in a Face-Centered Cubic (FCC) crystal lattice, which is the structure in which Argon crystallizes in nature.

The positions of the atoms in a FCC lattice are generated by integer linear combinations of the basis primitive vectors [1]. However, the size of the box imposes constraints on the allowed integers. In our implementation we use three nested for loops to store the initial positions of the particles. Furthermore, the atoms cannot be in opposite faces of the box, as this implies that their relative distance is 0. Thus, the atoms in three of the faces are removed. Counting the number of atoms N allocated for different values of the ratio r of the length of the box L and the lattice constant a, we deduced the following relation: $N = 4(\frac{L}{a})^3 = 4r^3$. This relation agrees with [5].

In order to simulate the dynamics for different sates of matter, we vary the dimensionless density $\tilde{\rho} = \rho\sigma^3 = N\sigma^3/L^3$ in the range from 0.3 to 1.7. We set the number of particles to $N = 256$, and from there calculate the length of the box $\tilde{L} = L/\sigma = \sqrt[3]{N/\tilde{\rho}}$ and the lattice constant $\tilde{a} = a/\sigma = \tilde{L}/r$, where $r = \sqrt[3]{N/4}$

Once the positions of the particles are initialised following the FCC configuration, we give each particle an initial velocity following the Maxwell-Boltzmann distribution function with normalised variables (see 6.2 and [5]). This velocity initialisation (done component-wise) is plotted alongside the Gaussian distribution with variance $\tilde{T}_{obj}$[2] (the normalised temperature), verifying the correctness of the initialisation. This can be visualised in Figure 4.

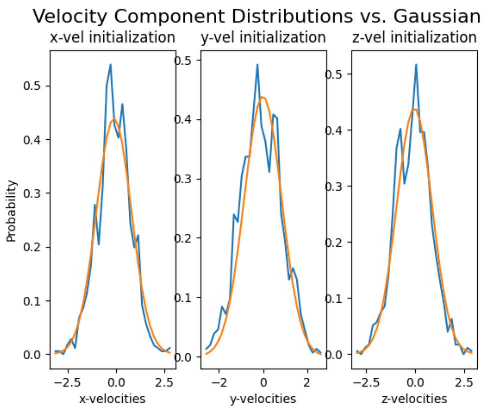


Figure 4: Velocity initialisation of 864 particles at normalised temperature $\tilde{T}_{obj} = 0.827$ following the Maxwell-Boltzmann distribution. The blue curve represents the distribution of velocities of the particles in the simulation. The orange curve is a Gaussian distribution centred at 0 with variance $\tilde{T}_{obj}$.

3.2 Numerical simulation of the dynamics

As mentioned in section 2.1, the probably most important element of this project is to compute the dynamics of our system (after all, everything else relies on having a correct representation of the MD). Though beautiful, equation (1) is extremely difficult to compute in the presented form, making it unusable for our practical purpose. Thus, we need to get some simpler approximations that are possible to actually simulate on a computer. We then present here two approximations giving rise to Euler's and Verlet's methods.

3.2.1 Euler's method

The first approximation we can make is by only keeping the linear terms in the dynamics. From equation (1), we can approximate the dynamics of the particles in the linear regime with the system of equations:

$$\begin{aligned} \mathbf{x}_i(t_{n+1}) &= \mathbf{x}_i(t_n) + \mathbf{v}_i(t_n)h, \\ \mathbf{v}_i(t_{n+1}) &= \mathbf{v}_i(t_n) + F(\mathbf{x}_i(t_n))h, \end{aligned} \tag{5}$$

[1]See Appendix 6.4 for more details on the initialisation of the atoms in a FCC lattice

[2]"Normal_Temp", l. 11, parameters.py

where h is the distance between time steps, t_n represents the n^{th} time step, and $\mathbf{v}_i = (v_{x_i}, v_{y_i}, v_{z_i})$ denotes the component-wise velocity of particle i. Evaluating these equations simultaneously for all particles is the core of Euler's algorithm at hand.

This algorithm presents multiple problems, not conserving the total energy of the system (an absolute necessity for any MD simulation) being the most relevant one. Consequently, this method is not actually used in the simulation.

3.2.2 Verlet's method

Instead of staying in the linear regime, we can go to second-order terms in (1), obtaining the system:

$$\mathbf{x}_i(t_{n+1}) = \mathbf{x}_i(t_n) + h\mathbf{v}_i(t_n) + \frac{h^2}{2}F(\mathbf{x}_i(t_n)),$$
$$\mathbf{v}_i(t_{n+1}) = \mathbf{v}_i(t_n) + \frac{h}{2}\left[F(\mathbf{x}_i(t_{n+1})) + F(\mathbf{x}_i(t_n))\right]. \tag{6}$$

As before, h is the distance between time steps, t_n represents the n^{th} time step, and $\mathbf{v}_i = (v_{x_i}, v_{y_i}, v_{z_i})$ denotes the component-wise velocity of particle i.

As it will be discussed in subsection 4.1.1, with this method we get a time-evolution of the system that respects the conservation of energy.

3.3 Error analysis

3.3.1 Autocorrelation

One approach to compute the error of an observable A that contains time-correlated data is by computing the correlation time τ and using the following formula deduced in [5]:

$$\sigma_A = \sqrt{\frac{2\tau}{n_t}(\langle A^2\rangle - \langle A\rangle^2)} \tag{7}$$

In order to address how to calculate the correlation time, we need to define first the autocorrelation function, which quantifies how much memory a data sequence has of previous configurations. For finite length simulation data, it can be computed as follows [5]:

$$\chi_A(t) = \frac{(n_t - t)\sum_n A_n A_{n+t} - \sum_n A_n \sum_n A_{+t}}{\sqrt{(n_t - t)\sum_n A_n^2 - \left(\sum_n A_n\right)^2}\sqrt{(n_t - t)\sum_n A_{n+t}^2 - \left(\sum_n A_{n+t}\right)^2}} \tag{8}$$

where $1 \leq n \leq n_t - t$ and n_t is the number of simulation steps.

Typically, the autocorrelation function has an exponential decay $e^{-t/\tau}$ (at least for the beginning of the simulation), where τ denotes the correlation time. Thus, we can obtain τ by fitting the data to an exponential.

3.3.2 Data Blocking

A second way to estimate the error of the observable A is by splitting the data into blocks of size b, computing the standard deviation $\sigma_A(b)$ of the means of all the blocks, and studying the convergence of $\sigma_A(b)$ when increasing b. More specifically, let us say that the observable A has the values (A_1, A_2, ..., A_{n_t}), where the subindex indicates the time-step. For a certain block size b we take the values

$$a_i = \frac{1}{b}\sum_{m=i\cdot b}^{(i+1)b-1} A_m,$$

for $i \in \{0, N_b - 1\}$, $N_b = n_t/b$. With this *new* set of data $(a_0, ..., a_{N_b-1})$, we can compute its standard deviation with the simple expression:

$$\sigma_A(b) = \sqrt{\frac{1}{N_b - 1}(\langle a^2\rangle - \langle a\rangle^2)}. \tag{9}$$

The value of b is increased until $\sigma_A(b)$ converges to a certain value, taken as our error estimation. The idea behind this method is that, once the block size b becomes larger than the correlation time τ of the observable A, the standard deviation should not present large variations (only fluctuations), thus getting the error for our observable.

In order to see the convergence of $\sigma_A(b)$ we fit it with the function $f(x) = \alpha - \beta e^{-x/\tau}$. The estimated convergence value of $\sigma_A(b)$ is then simply given by α. In Figures 5 and 6 it is possible to visualise the data blocking method for the specific heat and diffusion respectively (with different values of $\tilde{T}$ and ρ each).

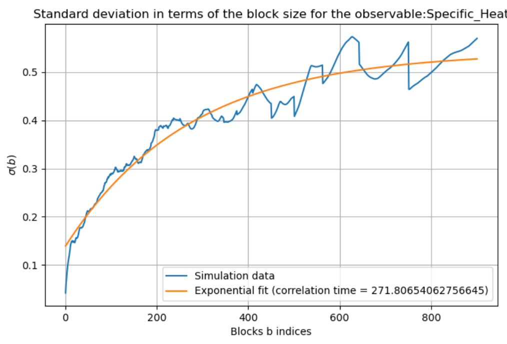


Figure 5: Data blocking estimation for the normalised total specific heat $\tilde{C}_V$, with $\tilde{T}_{obj} = 3$ and $\tilde{\rho} = 0.3$ (gas state). The blue curve represents $\sigma_{\tilde{C}_V}(b)$, which is fitted by $f(x) = \alpha - \beta e^{-x/\tau}$ in orange. The convergence value is $\alpha = 0.542$.

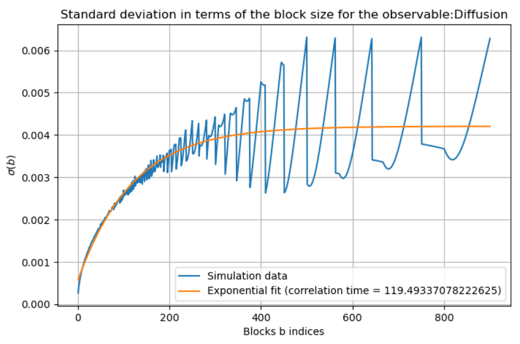


Figure 6: Data blocking estimation for the diffusion D, with $\tilde{T}_{obj} = 1$ and $\tilde{\rho} = 0.8$ (liquid state). The blue curve represents $\sigma_D(b)$, which is fitted by $f(x) = \alpha - \beta e^{-x/\tau}$ in orange. The convergence value is $\alpha = 0.0042$.

3.3.3 Bootstrapping

In order to estimate the error on observables that are derived from other observables, we may also employ a technique referred to as bootstrapping. In essence, this is a resampling method, where underlying data is repeatedly sampled, while calculating the observable for each such sample. To make this more precise, let $\mathcal{O}_{\mathcal{D}}$ be some observable depending on some data $\mathcal{D}$. In our MD simulation, this could resemble the kinetic energy depending on the velocities, to provide a simple example. Then we create a sample of new data $\mathcal{D}'$ by choosing $|\mathcal{D}|$ points from $\mathcal{D}$ *with* replacement. For each such draw, the observable $\mathcal{O}_{\mathcal{D}'}$ is determined and stored. We iterate this procedure 10,000 times in our simulation, each time storing the computed value of the observable, and derive the error estimate on our observable $\mathcal{O}$ as

$$\sigma_{\mathcal{O}} = \sqrt{\langle \mathcal{O}^2 \rangle_{\mathcal{D}'} - \langle \mathcal{O} \rangle_{\mathcal{D}'}^2}. \tag{10}$$

While more computationally expensive, this method allows us to approximate errors without performing error propagation analysis.

4 Results

4.1 Observables

4.1.1 Energies and Temperature

Let us first recall that the kinetic energy for a system of particles is given by the expression:

$$K_{total}(t) = \frac{1}{2}\sum_i m_i v_i^2(t), \tag{11}$$

where the sum is taken over all the particles of the system, t is the time step, m_i and v_i the mass and speed of the i-th particle. Now, in our program, this expression is simplified. Firstly, we consider all the particles to be Argon atoms, with the same mass, i.e. $m_i = m, \forall i$. Secondly, we are taking normalised units, where we have $v = \sqrt{\frac{\varepsilon}{m}}\tilde{v}$, with $\tilde{v}$ the normalised velocity. With these two elements, defining the normalised kinetic energy $\tilde{K}$ by the expression $K = \varepsilon\tilde{K}$, we then get:

$$\tilde{K} = \frac{1}{2}\sum_i \tilde{v}_i^2. \tag{12}$$

Before analysing the potential energy, it is convenient to first make some comments about the temperature. As discussed in Appendix 6.3, with the kinetic energy we can get the normalised temperature $\tilde{T} = \frac{k_B T}{\varepsilon}$ using Expression (26). Now, in our simulation, we want our system to be at a certain temperature $\tilde{T}_{obj}$, a fixed parameter. To "force" our system to settle around this desired temperature, i.e. $\tilde{T} \approx \tilde{T}_{obj}$, we apply at the beginning of the simulation a procedure called *rescaling*. More concretely, we transform the velocities of the particles with the parameter λ given in Appendix 6.3, i.e. $\mathbf{v}_i \to \lambda\mathbf{v}_i$. This rescaling is applied during the initial 10% of the time and only whenever $\tilde{T} \notin [\tilde{T}_{obj} - \tilde{T}_{obj}\cdot 0.05, \tilde{T}_{obj} + \tilde{T}_{obj}\cdot 0.05]$ (i.e. outside a 5% range).

For $\tilde{T}_{obj} = 0.827$ and $N = 256$ atoms, we obtain Figure 7.

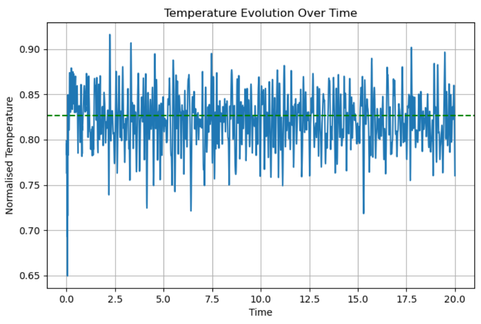


Figure 7: Time evolution of the normalised temperature of the system, with $\tilde{T}_{obj} = 0.827$ and $N = 256$ atoms. In green, the constant value of $\tilde{T}_{obj}$ is plotted.

Continuing the discussion of energies, the total potential energy is given by the expression

$$U_{LJ}(x_1, x_2, \ldots, x_N) = \frac{1}{2}\sum_{\alpha\neq\beta} U_{LJ}(|x_\alpha - x_\beta|), \tag{13}$$

where $U_{LJ}(|x_\alpha - x_\beta|) = U_{LJ}(r_{\alpha,\beta})$ is given by Equation (3). As for the kinetic energy, we consider normalised units with the expressions $\tilde{r} = \frac{r}{\sigma}$ and $U = \tilde{U}\varepsilon$, so that we get

$$\begin{aligned}\tilde{U}_{LJ}(\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_N) &= \frac{1}{2}\sum_{\alpha\neq\beta} \tilde{U}_{LJ}(\tilde{r}_{\alpha,\beta}) \\ &= 2\sum_{\alpha\neq\beta}\left(\tilde{r}^{-12} - \tilde{r}^{-6}\right).\end{aligned} \tag{14}$$

This (normalised) potential energy is computed at each time step of the simulation.

Having described the relations (12) and (14) used in our program to compute the kinetic and potential energy, we can add them to get the total energy of the system:

$$\tilde{E}(t) = \tilde{K}(t) + \tilde{U}_{LJ}(t). \tag{15}$$

For $\tilde{T}_{obj} = 0.827$, $\tilde{\rho} = 1.06$ and $N = 256$ atoms, the simulation yields Figure (8). Note that at the beginning, the total energy exhibits sudden jumps as a consequence of the rescaling.

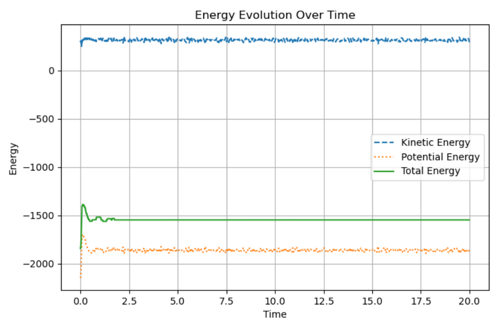


Figure 8: Time evolution of the Kinetic (blue, dashed), Potential (orange, dotted) and Total (green, straight) energies for $\tilde{T}_{obj} = 0.827$, $\tilde{\rho} = 1.06$ and $N = 256$ atoms. The first 10% evolution is done by applying the rescaling.

The errors for the kinetic and potential energy are computed using the autocorrelation method, as can be seen in Figures 9 and 10 for the kinetic and potential energy respectively for $\tilde{T}_{obj} = 3$ and $\tilde{\rho} = 0.3$.

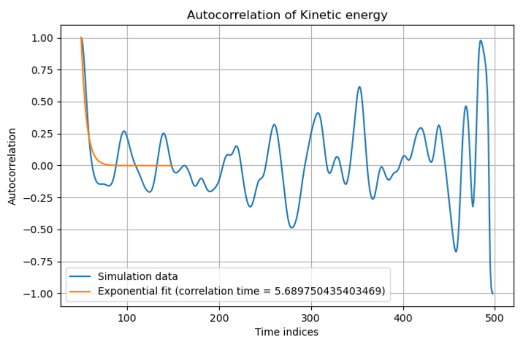


Figure 9: Autocorrelation of the Kinetic energy for $\tilde{T}_{obj} = 3$, $\tilde{\rho} = 0.3$ and $N = 256$ atoms.

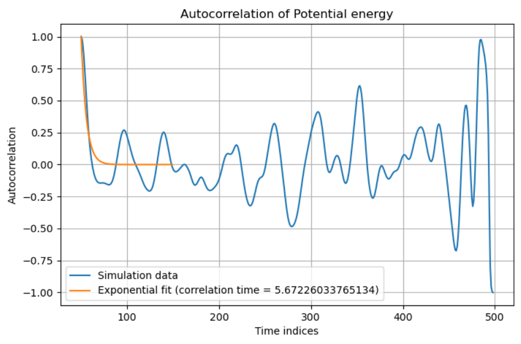


Figure 10: Autocorrelation of the Potential energy for $\tilde{T}_{obj} = 3$, $\tilde{\rho} = 0.3$ and $N = 256$ atoms.

For these values of $\tilde{T}_{obj}$ and $\tilde{\rho}$, corresponding to the gas state, we get as total energy in normalised units:

$$\tilde{E} = 683 \pm 2 \implies \frac{\tilde{E}}{N} = 2.67 \pm 0.01. \tag{16}$$

Let us compare this value with the theoretical value of an ideal gas. Let us recall that, according to the equipartition theorem, the total energy of an ideal gas is given by the expression $E_{ideal} = \frac{3}{2}Nk_BT$ [3]. In dimensionless units, with $\tilde{E}_{ideal} = \varepsilon\tilde{E}$, we get the formula for the total energy per particle:

$$\frac{\tilde{E}_{ideal}}{N} = \frac{3}{2}\tilde{T} \overset{\tilde{T}=3}{=} \frac{9}{2} = 4.5. \tag{17}$$

This theoretical value does not match the computed value in the simulation. This is a consequence of the fact that we are not working with an ideal gas, as the Argon atoms interact with each other with a certain potential. Nevertheless, it is worth noting that the (normalised) kinetic energy for $\tilde{T} = 3$ and $\tilde{\rho} = 0.3$ is $\tilde{K} = 1128 \pm 1 \implies \frac{\tilde{K}}{N} = 4.41 \pm 0.01$, which almost matches the value predicted for an ideal gas. This indicates that Argon could behave as an ideal gas if the interaction between atoms was negligible.

4.1.2 Pair Correlation Function

Broadly speaking, the pair correlation function $g(r)$, also referred to as radial distribution function, measures the probability of finding a particle at a distance r from another particle, relative to an ideal gas. We compute this observable by retrieving the array of all relative distances after equilibration, creating a histogram $n(r)$ of this data, and evaluating the expression

$$g(r) = \frac{2V}{N(N-1)}\frac{\langle n(r)\rangle}{4\pi r^2 \Delta r}, \tag{18}$$

where Δr is the bin size of the histogram, and the prefactor contains the term $\binom{N}{2}^{-1}$, which accounts for the total number of pairs in a given volume V. Most prominently, g assumes a characteristic shape depending on the phase of the system. Here, we investigate the pair correlation function for four sets of parameters, which correspond to the solid, liquid and gas phase, as well as at the triple point of Argon. For these simulations, we use 256 particles and 5,000 timesteps.

For dimensionless parameters $\tilde{T}_{obj} = 0.5, \tilde{\rho} = 1.2$, we obtain the pair correlation function depicted in Figure 11, which exhibits distinct peaks far surpassing the ideal gas reference (as indicated by the green dashed line). This is the characteristic signature of a solid, where (neglecting periodic boundary conditions) the distances $r = \frac{1}{\sqrt{2}}a, a, \sqrt{\frac{3}{2}}a, 2a, \ldots, \sqrt{\frac{k}{2}}a, k \in \mathbb{N}$, for lattice constant a are likely to be occupied due to the rigid structure of the material. For the specific parameters chosen here, we find $a = \frac{L}{r} = 1.494$, from which we predict a first peak of $g(r)$ at $r = 1.056$ and the following peaks according to the sequence listed above, as can be verified in the figure. Note that the choice of periodic boundary conditions prevents particle distances $r > \frac{\sqrt{3}}{2}L = 5.17$.

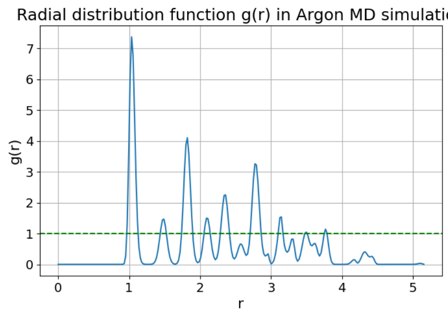


Figure 11: Pair correlation function of a solid for system parameters $\tilde{T}_{obj} = 0.5$ (units of $\frac{\epsilon}{k_B}$), $\tilde{\rho} = 1.2$ (units of σ^{-3}), 256 particles and 5,000 timesteps. Distance r and box size $L = 5.98$ in units of σ.

For dimensionless parameters $\tilde{T}_{obj} = 1, \tilde{\rho} = 0.8$, we obtain the pair correlation function depicted in Figure 12, which exhibits a shape characteristic of a liquid. Note that the repulsive term $\propto r^{-12}$ in the Lennard-Jones potential 3 makes particle distances $r \ll 1$ very unlikely, as can be seen by the potential shape in 1. Now, heuristically, particles spend more time in the range of the unit distance due to the inert effect of repulsion, which results in the given peak at this threshold. For larger distances, the probability density oscillates. Finally, for $r > L/2 = 3.42$, the density decays, since the periodic boundary conditions make large distances decreasingly likely, up to a threshold of $\frac{\sqrt{3}}{2}L = 5.92$, which is the maximal distance at which two particles in a system of size L can be located.

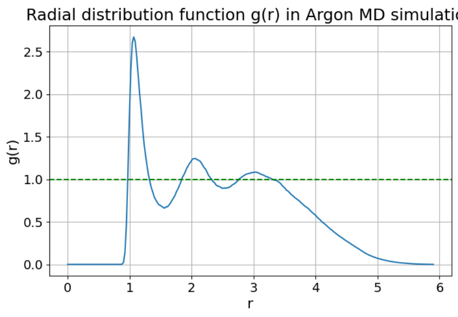


Figure 12: Pair correlation function of a liquid for system parameters $\tilde{T}_{obj} = 1$ (units of $\frac{\epsilon}{k_B}$), $\tilde{\rho} = 0.8$ (units of σ^{-3}), 256 particles and 5,000 timesteps. Distance r and box size $L = 6.84$ in units of σ.

For dimensionless parameters $\tilde{T}_{obj} = 3, \tilde{\rho} = 0.3$, we obtain the pair correlation function depicted in Figure 13, which exhibits similar peak and decay as before, but a characteristic plateau at $g = 1$ in the intermediate regime up to $L/2 = 4.74$. Since the probability density is given with respect to an ideal gas, we can interpret this as the pair correlation function of Argon in the gas phase.

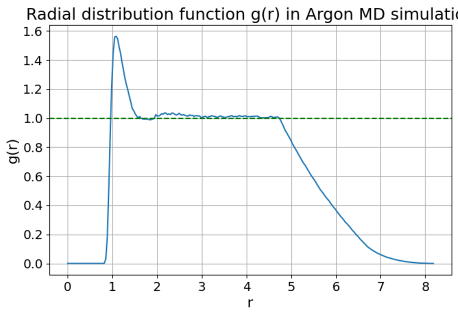


Figure 13: Pair correlation function of a gas for system parameters $\tilde{T}_{obj} = 3$ (units of $\frac{\epsilon}{k_B}$), $\tilde{\rho} = 0.3$ (units of σ^{-3}), 256 particles and 5,000 timesteps. Distance r and box size $L = 9.49$ in units of σ.

We find that the inter-atomic distances range on the order of the natural scale of length, as expected. Also, the the given interpretations of the radial distribution plots match the classification of phases, as given in [5]. For another verification of the pair correlation function with literature results, we refer the reader to Appendix 6.5.

Lastly, we investigate the triple point, which is found for parameters $\tilde{T}_{obj} = 0.827, \tilde{\rho} = 1.06$ [5]. The resulting pair correlation is depicted in Figure 14.

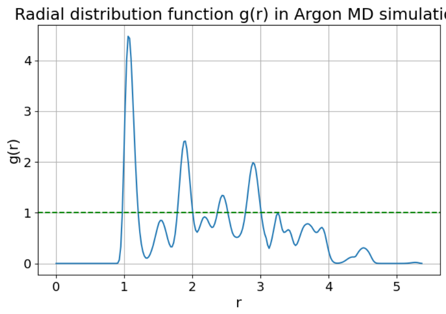


Figure 14: Pair correlation function of a solid for system parameters $\tilde{T}_{obj} = 0.827$ (units of $\frac{\epsilon}{k_B}$), $\tilde{\rho} = 1.06$ (units of σ^{-3}), 256 particles and 5,000 timesteps. Distance r and box size $L = 6.23$ in units of σ.

At the triple point, we find none of the aforementioned characteristic shapes of g distinctly, indicating that the system is close to a phase transition here. Summarising, we find that the pair correlation gives us insight into the state of the system, which can be particularly useful for MD simulations of materials for which given phase transition parameters are yet unknown.

4.1.3 Diffusion

Another way to distinguish different phases of Argon is by determining how much the atoms diffuse in average. The mean square displacement (MSD) quantifies this and is defined as:

$$\Delta^2 \mathbf{x}(t) = \langle [\mathbf{x}(t) - \mathbf{x}(0)]^2 \rangle, \tag{19}$$

where in our case we choose our initial reference time when the scaling period ends. It is important to note that the displacement of particles in a specific period of time is measured taking into account the boundary conditions, i.e., we determine the relative displacement. Depending on the phase of matter, the MSD shows a different behaviour in its evolution. However, for all phases, the MSD grows quadratically during a short period of time in the beginning, which can be deduced easily by making a Taylor expansion of $\mathbf{x}(t)$. However, the long term behaviour differs for the different states of matter.

For solids, the atoms will oscillate around their equilibrium positions, so we would expect the MSD to end up being low and constant in time, as the atoms barely diffuse. Figure 15 shows the evolution of the MSD when we set the temperature to $\tilde{T}_{obj} = 0.5$ and the density to $\tilde{\rho} = 1.2$. As we discussed in the pair correlation section, these parameters correspond to a solid, which agrees with the behaviour of the MSD.

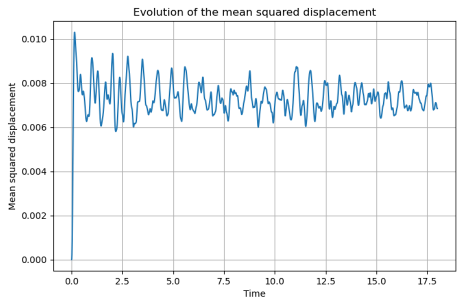


Figure 15: Mean squared displacement of Argon solid with system parameters $\tilde{T}_{obj} = 0.5$, $\tilde{\rho} = 1.2$, 256 atoms and 5000 timesteps. Time in units of $(\frac{m\sigma^2}{\epsilon})^{1/2}$ and MSD in units of σ^2.

In the other extreme we have gases, which are characterised by a higher temperature and a lower density. In gases, atoms interact less with each other (ideal gases do not interact at all) and therefore, they diffuse considerably. Nevertheless, due to the limited size of the box, the relative distance between atoms at different times is upper bounded by $\frac{1}{2}\sqrt{3}L = 8.2$. To sum up, we expect the MSD to grow quickly at initial times and then converge to a maximum value with some fluctuations.

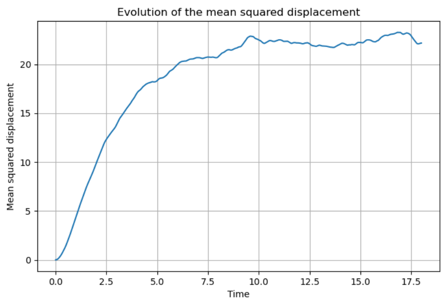


Figure 16: Mean squared displacement of Argon gas with system parameters $\tilde{T}_{obj} = 3$, $\tilde{\rho} = 0.3$, 256 atoms and 5000 timesteps. Time in units of $(\frac{m\sigma^2}{\epsilon})^{1/2}$ and MSD in units of σ^2.

Figure 16 represents the evolution of the MSD obtained when setting the temperature to $\tilde{T}_{obj} = 3$ and the density to $\tilde{\rho} = 0.3$, which corresponds to a gas according to the pair correlation function. We observe that its behaviour corresponds to the diffusion of a gas. Furthermore, for long times it reaches an asymptote at around 23 (a.u.), which corresponds to an average displacement of $\sqrt{23} = 4.8$. Thus, the average displacement of the atoms with respect to their initial positions is around 58% of the maximum possible distance, which is reasonable.

Finally, in between the solid and gas phases there is the liquid phase. In this case, the MSD is expected to grow linearly [4].

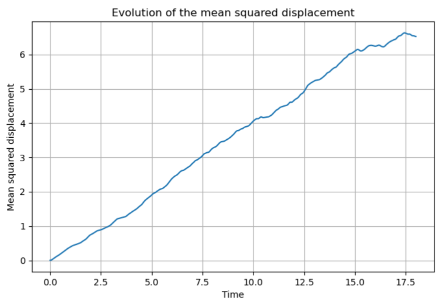


Figure 17: Mean squared displacement of Argon gas with system parameters $\tilde{T}_{obj} = 1$, $\tilde{\rho} = 0.8$, 256 atoms and 5000 timesteps. Time in units of $(\frac{m\sigma^2}{\epsilon})^{1/2}$ and MSD in units of σ^2.

Figure 17 shows the MSD for a liquid with temperature $\tilde{T}_{obj} = 1$ and density $\tilde{\rho} = 0.8$, which grows linearly as expected. In order to quantify the rate at which atoms diffuse, researchers use the following definition of diffusion: $D = \frac{\Delta^2 \mathbf{x}(t)}{6t}$, which is proportional to the slope of the MSD for liquids.

Using this definition, we can calculate the diffusion for the three states of matter, as well as its error using the data blocking method. Multiplying the values obtained by the units $\frac{\sigma^2}{\sqrt{\frac{m\sigma^2}{\epsilon}}} = 4.65\ m^2 s^{-1} = 4.65 \times 10^4\ cm^2 s^{-1}$, we obtain:

- Solid ($\tilde{T}_{obj} = 0.5, \tilde{\rho} = 1.2$): $(D = 8 \pm 2) \times 10^{-8}\ cm^2 s^{-1}$
- Gas ($\tilde{T}_{obj} = 3, \tilde{\rho} = 0.3$): $D = (1.67 \pm 0.3) \times 10^{-4}\ cm^2 s^{-1}$
- Liquid ($\tilde{T}_{obj} = 1, \tilde{\rho} = 0.8$): $D = (2.6 \pm 0.2) \times 10^{-5}\ cm^2 s^{-1}$

The experimental value of the diffusion of liquid Argon at $T \approx 85K$ obtained in [1] is $(1.53 \pm 0.03) \times 10^{-5}\ cm^2 s^{-1}$, which has the same order of magnitude of the value of our simulation. However, it is important to note that they are obtained at different temperatures, as we used $\tilde{T} = 1 \Leftrightarrow T = 119.8\ K$. Thus, it makes sense that we get a higher value of D since we are using a higher temperature.

4.1.4 Specific heat

When it comes to the specific heat, we can easily get an expression for it:

$$\frac{\langle \delta \tilde{K}^2 \rangle}{\langle \tilde{K} \rangle^2} = \frac{\langle \tilde{K}^2 \rangle}{\langle \tilde{K} \rangle^2} - 1 = \frac{2}{3N}\left(1 - \frac{3N}{2\tilde{C}_V}\right) = \frac{2}{3N} - \frac{1}{\tilde{C}_V}$$

$$\Longrightarrow \tilde{C}_V = \left[\frac{2}{3N} + 1 - \frac{\langle \tilde{K}^2 \rangle}{\langle \tilde{K} \rangle^2}\right]^{-1}. \qquad (20)$$

This expression (20) refers to the total specific heat in normalised units. The specific heat per particle is simply obtained by dividing by N: $\tilde{c}_V = \frac{\tilde{C}_V}{N}$. Its value in SI units is computed by multiplying by $\frac{k_B}{m_{Ag}}$[3], where $m_{Ag} \approx 6.6 \cdot 10^{-26}$kg denotes the mass of Argon. More specifically: $C_V = \tilde{C}_V \frac{k_B}{m_{Ag}}$.

We can compute the time evolution of the specific heat by simply taking the averages of $\tilde{K}^n$ up to a certain time-step t, i.e. $\langle \tilde{K}^n \rangle(t) = \frac{1}{t}\sum_{i=1}^{t} \tilde{K}(i)$. Nevertheless, it is crucial to keep in mind that the important value of the specific heat (the one that we actually care about) is the one computed at the end of the simulation.

As analysed in subsection 4.1.2, for $\tilde{T}_{obj} = 0.827$ and $\tilde{\rho} = 1.06$ we have the triple point. In Figure 18 it is possible to see the computed value for the normalised specific heat for the total system (left figure) and per particle (right figure) through time. Their final values are $\tilde{C}_V = 761 \pm 16$ and $\tilde{c}_V = 2.97 \pm 0.07$, where the

[3]Note that this factor has indeed units of $\frac{J}{kgK}$

error value is obtained by taking the maximum error of data blocking and bootstrap method. The specific heat per particle in SI is then $c_V = 622 \pm 14$ JK^{-1}kg^{-1}.

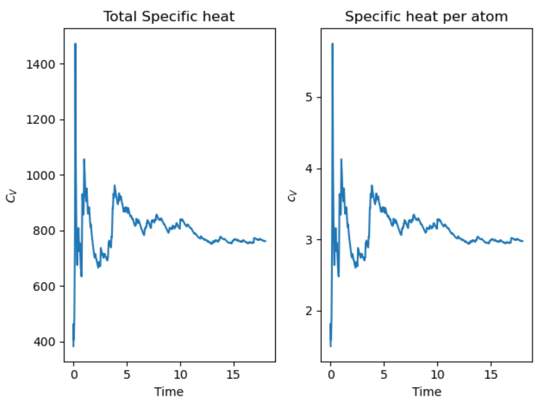


Figure 18: Time evolution of the normalised total specific heat (left) and normalised particle specific heat (right) for $\tilde{T}_{obj} = 0.827$, $\tilde{\rho} = 1.06$ and $N = 256$ atoms. At the end of the simulation, the value of the specific heat per particle in SI units is $c_V = 622 \pm 14$ JK^{-1}kg^{-1}.

When it comes to the liquid phase, given by $\tilde{T}_{obj} = 1$ and $\tilde{\rho} = 0.8$ we obtain Figure 19, and final values $\tilde{C}_V = 650 \pm 28$, $\tilde{c}_V = 2,54 \pm 0,1$. In SI units, the specific heat per particle is $c_V = 531 \pm 21$JK^{-1}kg^{-1}.

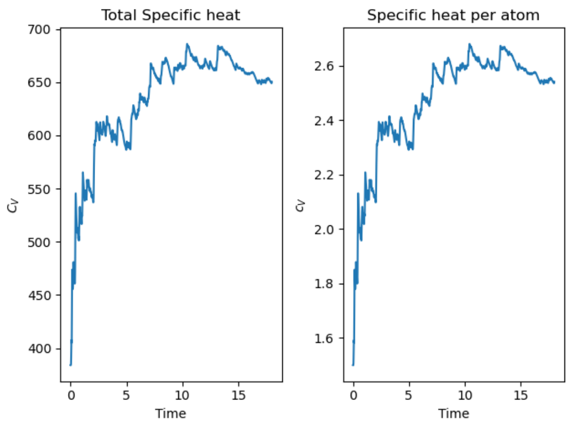


Figure 19: Time evolution of the normalised total specific heat (left) and normalised particle specific heat (right) for $\tilde{T}_{obj} = 1$, $\tilde{\rho} = 0.8$ and $N = 256$ atoms. At the end of the simulation, the value of the specific heat per particle in SI units is $c_V = 622 \pm 14$ JK^{-1}kg^{-1}.

The parameters $\tilde{T}_{obj} = 0.5$ and $\tilde{\rho} = 1.2$ define a solid. In Figure 20, we display the normalised specific heat both for the total system (left) and per particle (right). Their final values are $\tilde{C}_V = 782 \pm 31$ and $\tilde{c}_V = 3.05 \pm 0.13$. The specific heat per particle in SI is then $c_V = 638 \pm 26$ JK^{-1}kg^{-1}. This value can be compared with the Dulong-Petit prediction for a harmonic solid, given by the equation $c_V = 3k_B\frac{1}{m}$ in JK^{-1}kg^{-1} [3, p 204]. We obtain then, for $m_{Ag} \approx 6,6 \cdot 10^{-26}$kg, the theoretical result $c_V \approx 627$JK^{-1}kg^{-1}, which is inside the range of the result obtained by the simulation.

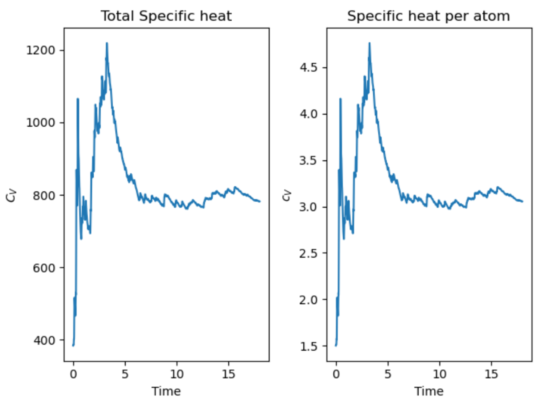


Figure 20: Time evolution of the normalised total specific heat (left) and normalised particle specific heat (right) for $\tilde{T}_{obj} = 0.5$, $\tilde{\rho} = 1.2$ and $N = 256$ atoms. At the end of the simulation, the value of the Specific heat per particle in SI units is $c_V = 638 \pm 26$ JK^{-1}kg^{-1}.

Now, the most interesting case to examine when it comes to the case when the system is in the gas phase, as there is plenty of documentation about the experimental value of the specific heat per particle in SI units for an Argon gas [5]. The specific heat per particle for an ideal monoatomic gas is given by the expression [2, p 140]

$$c_V = \frac{3}{2}\frac{k_B}{m}, \tag{21}$$

in units of JK^{-1}kg^{-1}. In the case of Argon, where $m_{Ag} \approx 6,6 \cdot 10^{-26}$kg, we get the theoretical value $c_V \approx 313,6$JK^{-1}kg^{-1}. Let us now compare this value with that obtained in the simulation considering $\tilde{T}_{obj} = 3$ and $\tilde{\rho} = 0.3$ (gas phase). The time evolution of the normalised specific heat can be seen in Figure 21, and the obtained value of the specific heat per particle is $c_V = 342 \pm 1$ JK^{-1}kg^{-1}. Note that, even though the theoretical and simulated values do not match, they are still really close, indicating the correctness of the simulation. Indeed, note the limitations of both the theoretical and simulated result: the former is a theoretical approximation, while the latter is a simulation with a finite (and relatively small) number of particles. With this in mind, we can then conclude that the result of the simulation *matches* the theoretical result.

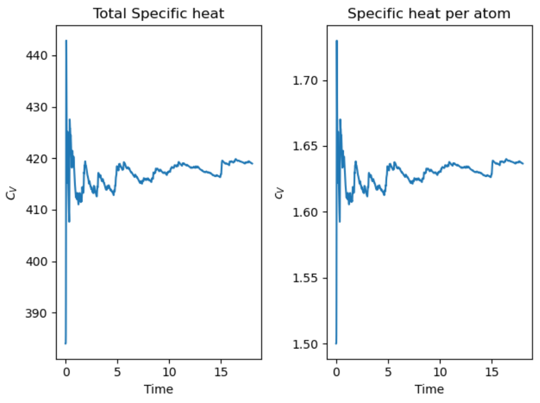


Figure 21: Time evolution of the normalised total specific heat (left) and normalised particle specific heat (right) for $\tilde{T}_{obj} = 3$, $\tilde{\rho} = 0.3$ and $N = 256$ atoms. At the end of the simulation, the value of the Specific heat per particle in SI units is $c_V = 342 \pm 1$ JK^{-1}kg^{-1}.

4.2 Performance Analysis

In order to gauge how well our MD simulation performs in terms of efficiency, we compare different particle counts and simulation times and measure the runtime of both only the simulation, and the simulation including the computation of all observables and error estimates. Since the most compute-intensive operations in our simulation, such as the pairwise distance function, involve tensors of size N along two dimensions, we expect a quadratic scaling of the runtime in the number of atoms. Indeed, as Figure 22 verifies, the simulation time scales favourably from 32 to 256 atoms.

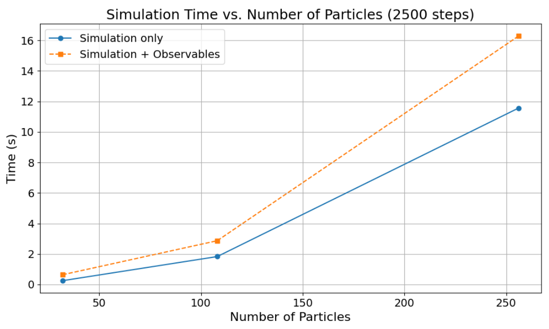


Figure 22: Simulation time per number of particles in MD simulation for 2500 timesteps, $\tilde{T}_{obj} = 1$, $\tilde{\rho} = 0.8$ (natural units), which exhibits quadratic scaling.

Increasing the atom count to 500, as shown in Figure 23, however drastically increases the runtime. This is not surprising, since in our optimization with *numpy* vectorization, we made extensive use of broadcasting operations for brevity. Now these broadcasting operations are efficient and support the vectorization benefits of our MD simulation well for the desired regime of particle numbers (to extract meaningful information about the observables, 256 particles are sufficient). However, once the tensors become too large along any given axis, this very broadcasting most likely leads to cache overflows and costly data-accessing operations. Extracting the runtime components of our simulation with 500 particles through a profiler (see Appendix 6.6), we find that a significant amount of time is spent on the *.ravel()* and *.copy()* operations. These are not used explicitly in the simulation, but utilized internally when broadcasting tensors. In order to increase efficiency for larger particle numbers, we could thus either revert back to a nested for-loop architecture, or utilize pre-made function from *scipy*, which we did not employ here for educational purposes.

Conclusively, the scope of this project did not necessitate particle numbers larger than 256 and our code is optimized for precisely this regime, where the simulation time scales quadratically. In particular, a full simulation of 256 Argon atoms over 2500 time steps including computation of observables takes less than 20 seconds on a Apple M3 Macbook Air, which allowed for both quick testing and sufficient computational accuracy.

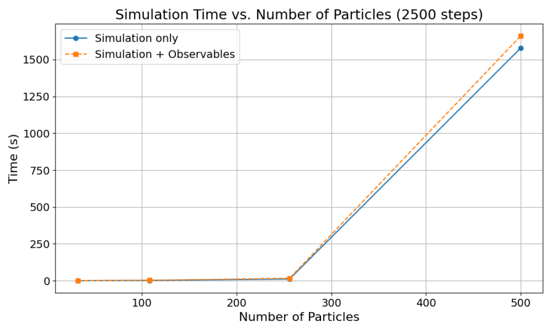


Figure 23: Simulation time per number of particles in MD simulation for 2500 timesteps, $\tilde{T}_{obj} = 1$, $\tilde{\rho} = 0.8$ (natural units). Strong scaling most likely caused by cache limitations in the usage of broadcasting operations.

Secondly, we considered how our simulation scales with the number of simulated time steps. Here, any reasonable amount of time steps [cf. [5, p.224]] yields a runtime of $< 25s$, and we do not encounter any cache overflows. Since the size of the main tensors depends linearly on the number of time steps, the linear dependence visualized in Figure 24 is expected.

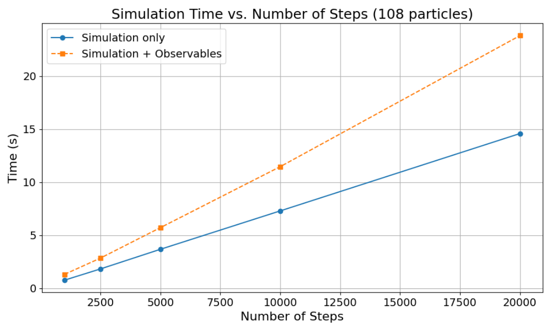


Figure 24: Simulation time per number of timesteps in MD simulation for 108 particles, $\tilde{T}_{obj} = 1$, $\tilde{\rho} = 0.8$ (natural units). Strong scaling most likely caused by cache limitations in the usage of broadcasting operations.

5 Conclusion & Outlook

In this report, we described and analysed the simulation of a system of 256 Argon atoms that interact classically via the Lennard-Jones potential. Varying the temperature and density of the system, and interpreting emergent observables such as the pair correlation function, we were able to identify different phases of matter of Argon. Observables like the mean square displacement and specific heat were determined, across different phases, in accordance with experimental data [1], theoretical frameworks such as the Dulong-Petit and ideal gas laws, and literature [5].

In terms of computational performance, we showed that our code is optimised for the desired regime of number of atoms and time steps, which allows for computation of all desired observables and error estimates in a reasonable amount of time.

Overall, this report demonstrated that one can extract information about observables, and on a more abstract level, about the state of the system through computational means. Potential further lines of work include the simulation of more complicated molecules, such as diatomic molecules such as hydrogen, or polar molecules such as water. Moreover, we expect the analysis of pressure as additional observable would be useful to deduce an equation of state relating the pressure, volume, density and temperature of the system.

References

[1] G Cini-Castagnoli and FP Ricci. "Self-diffusion in liquid argon". In: *Journal of Chemical Physics* 32.1 (1960), pp. 19–20.

[2] Kerson Huang. *Statistical Mechanics*. 2nd ed. John Wiley & sons, 1991. ISBN: 978-0-471-81518-1.

[3] D.A. McQuarrie. *Statistical Mechanics*. Mill Valley, California, 1973. ISBN: 9780060443658.

[4] Aneesur Rahman. "Correlations in the motion of atoms in liquid argon". In: *Physical review* 136.2A (1964), A405.

[5] J.M. Thijssen. *Computational Physics*. 2nd ed. Cambridge University Press, 2007. ISBN: 9781139171397. DOI: `https://doi.org/10.1017/CBO9781139171397`.

6 Appendix

6.1 Derivation of particle force form Lennard-Jones potential

In the following, we derive the formula

$$F(\mathbf{x}_i(t)) = -24\epsilon \sum_{\beta,\beta\neq i} \Big(\frac{\sigma^6}{r_{i,\beta}^8} - 2\frac{\sigma^{12}}{r_{i,\beta}^{14}}\Big)(\mathbf{x}_i - \mathbf{x}_\beta),$$

which provides updates to the equations of motion.

In order to do so, first note that we have

$$\mathbf{F}(\mathbf{x}_i(t)) = -\nabla_i U_{LJ,i}(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) = -\nabla_i \sum_{\beta,\beta\neq i} U_{LJ}(|\mathbf{x}_i - \mathbf{x}_\beta|),$$

where the Lennard-Jones potential

$$U_{LJ}(r) = 4\epsilon\Big[\Big(\frac{\sigma}{r}\Big)^{12} - \Big(\frac{\sigma}{r}\Big)^{6}\Big]$$

is only dependent on the inter-atomic distances $|\mathbf{x}_i - \mathbf{x}_\beta|$. The above form requires us to compute $\nabla_i U_{LJ}$, which we may rewrite as

$$\nabla_i U_{LJ} = (\frac{\partial}{\partial x_i}U_{LJ}, \frac{\partial}{\partial y_i}U_{LJ}, \frac{\partial}{\partial z_i}U_{LJ}) = \frac{\partial U_{LJ}}{\partial r_{i,\beta}}(\frac{\partial r_{i,\beta}}{\partial x_i}, \frac{\partial r_{i,\beta}}{\partial y_i}, \frac{\partial r_{i,\beta}}{\partial z_i}),$$

where the distance r is given by the usual L2 norm $r_{i,\beta} = \sqrt{(x_i - x_\beta)^2 + (y_i - y_\beta)^2 + (z_i - z_\beta)^2}$. Differentiating r with respect to x_i yields

$$\frac{\partial r_{i,\beta}}{\partial x_i} = \frac{x_i - x_\beta}{|x_i - x_\beta|},$$

and likewise for the y_i and z_i components of particle i. Substituting in our initial equation, we find

$$\mathbf{F}(\mathbf{x}_i(t)) = -\sum_{\beta,\beta\neq i} \nabla_i U_{LJ}(|\mathbf{x}_i - \mathbf{x}_\beta|) = -\sum_{\beta,\beta\neq i} \frac{\partial U_{LJ}}{\partial r_{i,\beta}} \frac{\mathbf{x}_i - \mathbf{x}_\beta}{|\mathbf{x}_i - \mathbf{x}_\beta|}.$$

Lastly, we find the derivative of the Lennard Jones potential

$$\frac{\partial U_{LJ}}{\partial r_{i,\beta}} = 24\epsilon\Big(\frac{\sigma^6}{r_{i,\beta}^7} - 2\frac{\sigma^{12}}{r_{i,\beta}^{13}}\Big),$$

which upon replacement in the above expression yields

$$\mathbf{F}(\mathbf{x}_i(t)) = -\sum_{\beta,\beta\neq i} 24\epsilon\Big(\frac{\sigma^6}{r_{i,\beta}^7} - 2\frac{\sigma^{12}}{r_{i,\beta}^{13}}\Big)\frac{\mathbf{x}_i - \mathbf{x}_\beta}{|\mathbf{x}_i - \mathbf{x}_\beta|} = -24\epsilon \sum_{\beta,\beta\neq i} \Big(\frac{\sigma^6}{r_{i,\beta}^8} - 2\frac{\sigma^{12}}{r_{i,\beta}^{14}}\Big)(\mathbf{x}_i - \mathbf{x}_\beta),$$

as desired.

Expressing the above formulae in natural units yields the functional forms we utilize in the MD simulation.

6.2 Maxwell-Boltzmann distribution

In statistical mechanics, the velocity distribution of each particle in a gas is given by the Probability Distribution Function:

$$f(v_i)dv_i = \frac{1}{\sqrt{2\pi}} \cdot \sqrt{\frac{m}{k_BT}} e^{-\frac{mv_x^2}{2\cdot k_BT}} dv_i, \tag{22}$$

for $v_i \in \{v_x, v_y, v_z\}$. Using the normalised velocity found in the lectures, given by $v_i = \sqrt{\frac{\epsilon}{m}}\tilde{v}_i$ we then obtain:

$$\sqrt{\frac{\epsilon}{m}} f(\tilde{v}_i)d\tilde{v}_i = \frac{1}{\sqrt{2\pi}} \cdot \sqrt{\frac{\epsilon}{k_BT}} e^{-\frac{\epsilon\tilde{v}_x^2}{2\cdot k_BT}} d\tilde{v}_i. \tag{23}$$

Defining the normalised temperature $\tilde{T}$ according to the relation $\frac{1}{\tilde{T}} = \frac{\epsilon}{k_BT}$, we obtain:

$$\sqrt{\frac{\epsilon}{m}} f(\tilde{v}_i)d\tilde{v}_i = \tilde{f}(\tilde{v}_i)d\tilde{v}_i = \frac{1}{\sqrt{2\pi}} \cdot \frac{1}{\sqrt{\tilde{T}}} e^{-\frac{\tilde{v}_x^2}{2\cdot\tilde{T}}} d\tilde{v}_i. \tag{24}$$

Note that the RHS is indeed a Probability Distribution as it integrates to 1. Actually, it is a Gaussian centred at 0 and with variance $\tilde{T}^2$.

Recall that, in the case of Argon, we have $\frac{\epsilon}{k_b} = 119.8\text{K}$. So, if we take $T \sim 119.8\text{K} \Rightarrow \tilde{T} \sim 1 = 10^0$.

6.3 Handling of the velocities and temperatures (rescaling)

The kinetic energy is given by the expression:

$$K = \frac{3}{2}(N-1)k_BT, \tag{25}$$

with N the number of particles and T the temperature. Now, taking the normalised temperature $\tilde{T}$ defined before, we get:

$$K = \frac{3}{2}(N-1)\epsilon\tilde{T} \quad \overset{K=\epsilon\tilde{K}}{\longrightarrow} \quad \tilde{K} = \frac{3}{2}(N-1)\tilde{T} \Leftrightarrow \tilde{T} = \frac{2}{3} \cdot \frac{\tilde{K}}{N-1}. \tag{26}$$

From this later formula we can obtain the normalised temperature at every time-step in the simulation as we know the kinetic energy.

Let us now get the rescaling factor

$$\lambda = \sqrt{\frac{3(N-1)k_BT_{obj}}{\sum_i mv_i^2}}$$

with the normalised parameters. First of all, the normalised velocity verifies $v_i = \sqrt{\frac{\epsilon}{m}}\tilde{v}_i \implies \sum_i mv_i^2 = \epsilon\sum_i v_i^2 =: \epsilon V$. Hence, we deduce:

$$\lambda = \sqrt{\frac{3(N-1)k_B T_{obj}}{\epsilon V}} = \sqrt{\frac{3(N-1)\tilde{T}_{obj}}{\sum_i \tilde{v}_i^2}} = \sqrt{\frac{3(N-1)\tilde{T}_{obj}}{2\tilde{K}}}, \tag{27}$$

which gives the rescaling factor λ in terms of the normalised parameters. Note that the sum i is done over all the particles in the system.

6.4 FCC initialization

The FCC, or face-centered cubic, is one of the most important Bravais lattices in Solid State Physics. In this lattice, the atoms arrange in the vertices of cubes but also in the centre of the faces. This geometrical arrangement can be visualized in Figure 25.

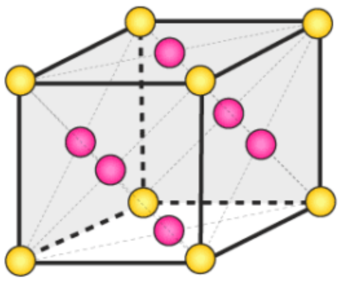


Figure 25: Geometrical arrangement of FCC Bravais lattice. Retrieved from `https://byjus.com/chemistry/bcc-fcc-primitive-cubic-unit-cell/` (accessed March 25, 2025)

The positions of the atoms in this lattice can be parametrized using the following primitive basis vectors:

$$\left\{ a_1 = \begin{pmatrix} a/2 \\ a/2 \\ 0 \end{pmatrix}, a_2 = \begin{pmatrix} 0 \\ a/2 \\ a/2 \end{pmatrix}, a_3 = \begin{pmatrix} a/2 \\ 0 \\ a/2 \end{pmatrix} \right\}, \tag{28}$$

where a is the length of the unit cube and what we refer in the code as lattice constant. Thus, any position in the lattice can be expresses as an integer linear combination of these vectors:

$$r(n_1, n_2, n_3) = n_1 a_1 + n_2 a_2 + n_3 a_3; \quad n_1, n_2, n_3 \in \mathbb{Z} \tag{29}$$

However, our Argon simulation takes place in a limited space, a box of length L. Thus, we need to figure out what is the range of n_1, n_2 and n_3. For the sake of simplicity, let us assume that the coordinate origin is in one of the vertices of the box and the centre of the box is therefore in $(a/2, a/2, a/2)$.

First, we notice that the vertices that are in the Cartesian axes can be reached setting two integers the same, and the other to the negative of that number. We find by trial and error that the range of n_1 has to be between $-L/a = r$ and $2L/a = 2r$. To find out the range of n_2 and n_3, we first expand Equation 29 and impose the box constraints:

$$0 \leq r(n_1, n_2, n_3) = \frac{a}{2} \begin{pmatrix} n_1 + n_3 \\ n_1 + n_2 \\ n_2 + n_3 \end{pmatrix} \leq L \tag{30}$$

From this we obtain the range of n_2 and n_3:

$$-n_1 \leq n_2 \leq 2r - n_1 \tag{31}$$

$$-n_1 \leq n_3 \leq 2r - n_1 \quad and \quad -n_2 \leq n_3 \leq 2r - n_2 \tag{32}$$

This last condition is equivalent to:

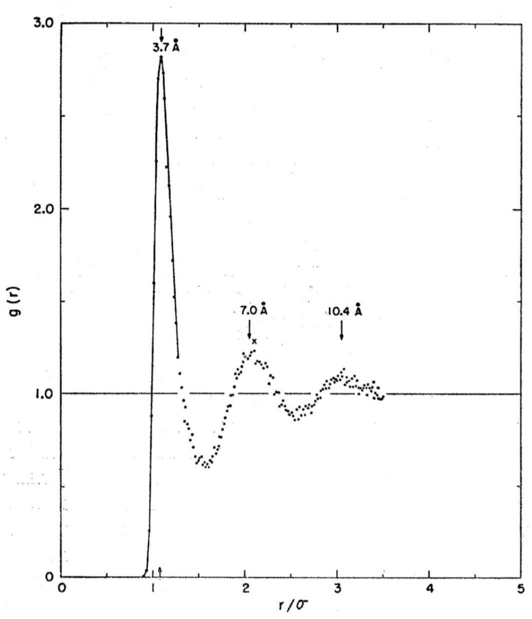
$$-min(n_1, n_2) \leq n_3 \leq 2r - max(n_1, n_2) \tag{33}$$

To sum up, we have obtained the range of the three integers that parametrize the positions of a FCC lattice constrained to a box of length L. This can be implemented by three for loops. Finally, in our code the origin is in the centre of the box, so we have to shift all the positions by the vector $-(L/2, L/2, L/2)$.

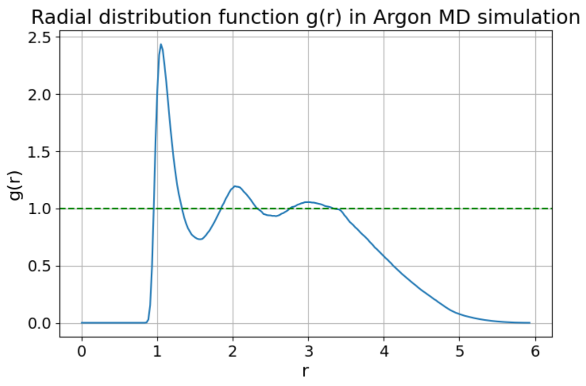
We noticed that with this initialization the energies diverged as there atoms in opposite faces of the box whose relative distances were 0 in the beginning. Thus, we remove the atoms located at the faces $x = L,\ y = L,$ and $z = L$.

6.5 Pair Correlation function $g(r)$ in literature

In order to provide another verification of our computational results of the pair correlation function $g(r)$, we compare the results obtained in [4]. In this work, the author depicts the pair correlation function for system parameters $T = 94.4\text{K}$, $\rho = 1.374\text{gcm}^{-3}$. Recalling that we implicitly assume a particle mass of $6.6 \cdot 10^{-26}$kg, as well as $\sigma = 3.405 \cdot 10^{-10}$m, we transform the given parameters to the dimensionless simulation parameters $\tilde{T} = 1.627$, $\tilde{\rho} = 0.788$. Performing the simulation and comparing the functions in Figure 26 verifies the correctness of our computations. In particular, the locations of the peaks match precisely.



(a) Pair correlation function for a system with parameters $T = 94.4°\text{K}$, $\rho = 1.374\text{gcm}^{-3}$, according to [4, Figure 2]



(b) Pair correlation function as obtained from MD simulation with equivalent system parameters $\tilde{T}_{obj} = 1.627$, $\tilde{\rho} = 0.788$. Simulation performed with 256 particles and 5,000 time steps. Distance r and box size $L = 6.87$ are in units of σ.

Figure 26: Comparison of pair correlation function from literature with MD simulation.

6.6 Profiler information for performance analysis

This is the (truncated) profiler output of our MD simulation with 500 atoms, sorted by decreasing internal execution time, as described in Section 4.2. We find that the internal *.ravel()* and *.copy()* functions dominate the runtime.

```
ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1 1895.854 1895.854 1949.525 1949.525 simulation.py:12(simulate)
      446   50.584    0.113   50.584    0.113 {method 'ravel' of 'numpy.ndarray' objects}
    11521   29.369    0.003   29.369    0.003 {method 'copy' of 'numpy.ndarray' objects}
     2500   20.503    0.008   25.895    0.010 simulation.py:299(lj_force)
     2501   14.363    0.006   19.022    0.008 simulation.py:248(atomic_distances)
     8600   13.964    0.002   13.964    0.002 {method 'sort' of 'numpy.ndarray' objects}
     2500    6.692    0.003    8.620    0.003 observables.py:31(potential_energy)
   199581    4.287    0.000    4.287    0.000 {method 'reduce' of 'numpy.ufunc' objects}
     2526    3.124    0.001    3.597    0.001 numeric.py:2337(isclose)
     2502    1.421    0.001    4.660    0.002 _linalg.py:2575(norm)
     7500    0.797    0.000    1.017    0.000 _twodim_base_impl.py:1124(<genexpr>)
     2500    0.627    0.000    0.627    0.000 {method 'repeat' of 'numpy.ndarray' objects
        1    0.437    0.437    0.439    0.439 observables.py:122(diffusion)
     2500    0.297    0.000    1.782    0.001 _twodim_base_impl.py:1041(triu_indices)
     2508    0.207    0.000    0.207    0.000 {method 'outer' of 'numpy.ufunc' objects}
   152820    0.196    0.000    1.275    0.000 fromnumeric.py:69(_wrapreduction)
    28662    0.187    0.000    0.187    0.000 {built-in method builtins.abs}
     1822    0.183    0.000    0.184    0.000 __init__.py:65(check_isinstance)
   112092    0.165    0.000    1.370    0.000 fromnumeric.py:2338(sum)
        1    0.142    0.142   94.278   94.278 observables.py:65(pair_correlation)
  107/104    0.134    0.001    0.138    0.001 {built-in method _imp.create_dynamic}
     5000    0.133    0.000    0.192    0.000 _stride_tricks_impl.py:340(_broadcast_to)
      643    0.093    0.000    0.093    0.000 {method 'read' of '_io.BufferedReader' obje
     2500    0.088    0.000    0.126    0.000 observables.py:11(kinetic_energy)
```

Note on authorship: All three authors contributed equally to the project, with tasks clearly defined and split as documented in the weekly journals. This is also reflected in consistent commits of all team members.