Molecular dynamics simulation of Argon

Authors: Carles Esteve Boncompte, Georgios Stylianou
Applied Physics, TU Delft — AP3082: Computational Physics

Abstract: A classical simulation of $N = 108$ particles has been carried out in order to study the different phases of Argon via the study of three thermodynamical macroscopic quantities. These have been computed by means of ensemble averages of the system. Agreement of the observed quantities with the theoretical predictions has been found for the different observed states of the matter.

I. INTRODUCTION

In this work we intend to study the behaviour of a N-Argon-atom ensemble via the well known Molecular Dynamics computational description. As seen in literature [1], this computational method is used to study a collection of classical particles and it consists of numerically integrating Newton's equation of motion of every particle. Different methods of integration can be used, with respective different precision, to make the system evolve in time. Two of them have been used in this work, the Euler method and the Verlet algorithm, keeping the one with best performance, Verlet algorithm, when obtaining the results. Macroscopic properties of the system, what we aim to study, are obtained via ensemble averages of the system and will be analyzed in the results section. Specifically, three of these properties will be discussed, namely, the pressure of the system, the pair correlation function and the specific heat per particle.

II. THEORETICAL BACKGROUND

Classical mechanics allows us to describe a classical system by means of $6N$ values, that is, a $6N$ dimensional vector in a generalized coordinate space. Fixing the energy of the system takes the system into the microcanonical scheme, where that $6N$ dimensional vector collapses into a surface with $6N-1$ degrees of freedom. Considering now our system as a canonical ensemble, with fixed temperature T, we can compute several macroscopic quantities which give information about global state of the system. This procedure is done taking what is usually called *ensemble averages*, a mean value over all the different microstates of the system over a period of time. For instance, in this regime, we can compute the specific heat constant (per particle) $c_V = C_V/N$ knowing the fluctuation of the energy or the pressure p making use of the Virial theorem. [2]
Since our modelling of our system treats its components as classical particles, we can write the Hamiltonian that governs their motion

$$H = \sum_{i=1}^{N} \frac{p_i^2}{2m_{Ar}} + V(R) \tag{1}$$

where $\vec{p_i}$ is the moment of each of the particles and R is the full set of positions of the N particles. We may approximate the potential that the particles feel by a pair-wise potential, which can be written as

$$V(R) = \frac{1}{2}\sum_{i\neq j}^{N} V_p(|\vec{r_i} - \vec{r_j}|) \tag{2}$$

Since we are dealing with Argon atoms, we can fairly approximate the pair wise potential we use to model their interaction by the Lennard-Jones potential, shape of which can be seen in the following

$$V_p(|\vec{r_i} - \vec{r_j}|) = V_{LJ}(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \tag{3}$$

The step of going from a general potential to a pair-wise one is vital when it comes to model a computational program that is able to perform the required calculations in a reasonable amount of time. For example, if we also included three-particle interactions, when computing the force that each of the particles feel, its contributions would scale to the order $O(N^3)$. [3] Literature gives several reasons why Lennard-Jones potential is suitable when it comes to describing the pair-wise potential that a collection of Argon atoms feel [3], here we will only expose two of these reasons:

- The first one is that this potential is in general good to model any rare gases (i.e. noble gases) [2], since it is a fairly good description for neutral particles, i.e. non Coulomb interacting. Wee may also note here that we can safely neglect quantum effects that would arise if we used another inert species different than Argon such as Helium).
- It can be applied to the solid, liquid and gas phases. [3]

As a remark, Lennard-Jones potential terms can be seen as repulsion in short distances and attraction at long distances, and it exhibits a minimum of energy $-\epsilon$ at position $r = 2^{1/6}\sigma$, with $\sigma = 3.405$ Å.
Once the system frame and the interaction is modelled, we need to let the system evolve in time and obtain macroscopic quantities by means of ensemble averages. Pressure can be computed according to the following formula

$$P = \rho k_B T - \frac{\rho}{3N}\left\langle \frac{1}{2}\sum_{i\neq j} r_{ij}\frac{\partial V_{LJ}(r_{ij})}{\partial r_{ij}} \right\rangle \tag{4}$$

which makes use of the Virial theorem and the ensemble average is denoted as $\langle \dots \rangle$. Specific heat per particle will be obtained from the fluctuations of the kinetic energy, this latter denoted as K. The exact formula can be found below

$$\frac{\langle \delta K^2 \rangle}{\langle K \rangle^2} = \frac{3}{2N}\left(1 - \frac{3}{2c_V}\right) \tag{5}$$

where $\langle \delta K^2 \rangle = \langle K^2 \rangle - \langle K \rangle^2$. Finally, the pair correlation function is computed according to

$$g(r) = \frac{2V}{N(N-1)}\frac{\langle n(r) \rangle}{4\pi r^2 \Delta r} \tag{6}$$

where $n(r)$ is the histogram with all the particle pairs at distance $[r, r + \Delta r]$ and Δr is the bin size.

III. DESCRIPTION OF THE COMPUTER EXPERIMENTS

In this section a description about how we carried out the simulation will be done. Namely, we will split this discussion into three parts, the first one giving details about how our computer experiment setup was built and the other ones discussing the correctness of the code and the performance of our simulations. The project principally consists of two files of code, a `functions.py` file and a `main.py` file (although sometimes also called `Week#.py` when looking at the Commits of the project and used to execute the computations as a `.ipynb` file) which are self explanatory regarding their content. A weekly tracing journal was also filled up while constructing the project, available in the **Gitlab repository of the project** along with the rest of the documents.

III.A. Construction of the code

In this subsection we will give some important details about the workflow in our project, in accordance with the five weeks journal we wrote when carrying it out. If any of the functions or piece of code we comment in this subsection does no longer exist in the final version of the program, please see the Commits of the code to find it if wished.

1. Week 1

In *Week 1* we roughly settled up the experiment framework. We will give some details about how we model the space where the N particles evolve in time. As we mentioned, we need N particles, which practically will be their position and velocities at all times, and a very big volume, in thermodynamic terms, infinite. Since we cannot have in practice an infinite volume, we will model our volume as a cube with side length L and we will implement periodic boundary conditions in its faces. This behaviour is modelled in the function `bring_back`, which was also improved along the following weeks of the project.
When it comes to storing all the N particle's information, for commodity reasons in our simulation, this $6N$ dimensional object we commented about above will not be a single vector but two $3 \times N$ arrays containing each coordinate of all particle's position and velocity.
When it comes to functions, the most important developments to follow up to the final stage of the project are two: the `vel_verlet` function (for the first weeks when we worked with the Euler algorithm this is a different analogous function called `partmove`) and the `k_p_energy` function. The `vel_verlet` function is a function made up to evolve the particles positions and velocities a given number of time steps l, often referred to it as `loopnum`. The way it works, in three dimensions, is the following. The input of the function is a collection of (initial) positions and velocities of a set of N particles and the output it gives is what we call a position and velocity tracker, an array for the positions and velocities of the particles in every time step, including the given ones. It is easy to see that this corresponds to two $3 \times N \times (l + 1)$ arrays. However, immediately after wanting to track the positions and velocities of the particles we are interested in tracking other magnitudes, such as the energies. The `k_p_energy` function calls the `vel_verlet` function and extends its computations to also track the "instantaneous" potential and kinetic energy of the particles, i.e. the kinetic and potential energy at every time step the system goes through. We will use the "instantaneous" notation repeatedly throughout the development of this text when referring to a microstate-value computation.

2. Week 2

In *Week 2*, as we discuss in the journal of the project, we developed the code for dimensionless units, but no specific function built in that period is relevant to understand the runs of the code that give us results. The units used in the code have been written in terms of σ, ϵ or m_{Ar} or a combination of these. Below we show the values of these magnitudes.

$$\sigma = 3.405\,\text{Å} = 3.405 \times 10^{-10}\,\text{m} \tag{7}$$

$$\epsilon = 119.8\,\text{K} \times k_B \tag{8}$$

$$m_{Ar} = 6.6 \times 10^{-26}\,\text{kg} \tag{9}$$

We will call the dimensionless parameters with a tilde (the ones we use in our computer program), so we can after express them in SI units. We will do this for the distance, the time and the energy k_BT, but any other magnitude is analogous ($K(T)$ denotes the value of the temperature in Kelvin but without the units).

$$r = \tilde{r} \times \sigma \tag{10}$$

$$t = \tilde{t} \times \sqrt{\frac{m_{Ar}\sigma^2}{\epsilon}} \tag{11}$$

$$k_B T = \tilde{k_B T} \times \epsilon = \frac{T(K)}{119.8} \times \epsilon \tag{12}$$

3. Week 3

Week 3 was also focused on improving the code but not on specifically building any function relevant for the obtainment of the results. In fact, this is the time when we changed the integration of the Newton's equation from the Euler method to the Verlet method. We may discuss the impact that this had in the next Code check subsection.

4. Week 4

In *Week 4* was mainly aimed to create the particle's initialization function, `initialization_fcc`, and the energy rescaling function `energy_rescale`. We will give some details about these functions in the next paragraph.
The initialization of the particles sets up the initial positions and velocities of the N particles in our simulation. The positions are initialized according to an fcc lattice since the solid state of Argon is experimentally observed to order in this way. Velocities are randomly Gaussian-distributed according to the Maxwell-Boltzmann distribution for temperature T. That is equivalent to a Normal probability distribution with mean value $\mu_N = 0$ and $\sigma_N = \sqrt{\frac{k_B T}{m_{Ar}}} = \sqrt{\frac{T(K)}{119.8}}$.
The rescaling of the energy also makes use of the `vel_verlet` function in order to evolve in time. It is also convenient for us to store the energies in every time step to graphically see how the rescaling is done. We will be able to observe the time evolution of the rescaling process in the results section. Rescaling the energies basically consists on forcing the temperature of the system to be close to the input one multiplying the velocities of the particles by a factor that scales their kinetic energy K so it fits with the temperature of the system according to the following equation

$$K = (N-1)\frac{3}{2}k_B T \tag{13}$$

This is carried out in the `energy_rescale` function by means of looping a rescaling process until the kinetic energy of the system and the temperature satisfy equation (13) with a certain `precision`, which is input of the function.

5. Week 5

Week 5 was aimed to obtain the observables (and its corresponding errors) we want to study about the N particle ensemble. As we saw in the Theoretical Background section, we managed to get the thermodynamic values by taking the mean value of a collection of "instantaneous" values which we computed for every time step the system was in, and this value was what can be found inside the brackets of formulas (4), (5) and (6). The general form of measuring an observable is the following

$$\langle A \rangle = \frac{1}{N}\sum_{n=1}^{N'} A_n \tag{14}$$

where N' is the number of time steps that the system has been through after the rescaling, i.e. the number of microstates after rescaling. The error for the pressure was computed this way, as well as the error of the kinetic energy and the kinetic energy squared. The error associated with the specific heat per particle was found by propagation of errors taking into account the two latter mentioned errors related to the kinetic energy and the kinetic energy squared. The function that computes all the list of instantaneous values inside the brackets and gives a vector with all of them is named "`merge`" function in the final version of the code, and it does this procedure for the three observables we study within the same Verlet time evolution loop.
Every statistical quantity has an error associated, which we also have to compute. Our approach to do it was using the auto-correlation function. Once the set of "instantaneous" measurements of a specific observable A has been obtained, the `auto-correlation` function plots the auto-correlation function associated with this set of measurements along with the mean (thermodynamic) value $\langle A \rangle$ and its error σ_A. The error σ_A is computed according to the following formula

$$\sigma_A = \sqrt{\frac{2\tau^*}{N}\left(\langle A^2 \rangle - \langle A \rangle^2\right)} \tag{15}$$

where τ^* satisfies $\tau^* = \tau/h$ (remember we use $h = 0.01$ as a time step). The reader may observe that when plotting some auto-correlation functions in the results section τ may be less than 1 but this will not happen for τ^*.

III.B. Code check

A few code checks have been done in order to make sure the simulation worked properly. The first test we did in a very early stage of the project was the conservation of the energy of the system. Indeed, for the Verlet algorithm, when plotting the energies as a function of time if could be observed that the total energy was nearly a 0 slope constant value. This was not the case for Euler integration, although the energy

also remained steady as a whole if the evolution time was short.
Another of the tests we carried out has been relative to thermodynamical limits of the system. For very high temperatures and very low densities, Argon should be in the gas phase and be approaching a nearly perfect ideal gas behaviour. We will see in the results section how we recover the pressure value we actually obtain through the simulation using the ideal gas formula.
The other test we used to check if our code gave correct results was based on literature we consulted. Specifically, we tried to recover similar values to *Table 8.1* of [3][1] using the respective values for the number density and temperature $\rho = 0.75$ and $T = 119.0(8)\,\mathrm{K}$. Again, an agreement was found and we show it in the results section.

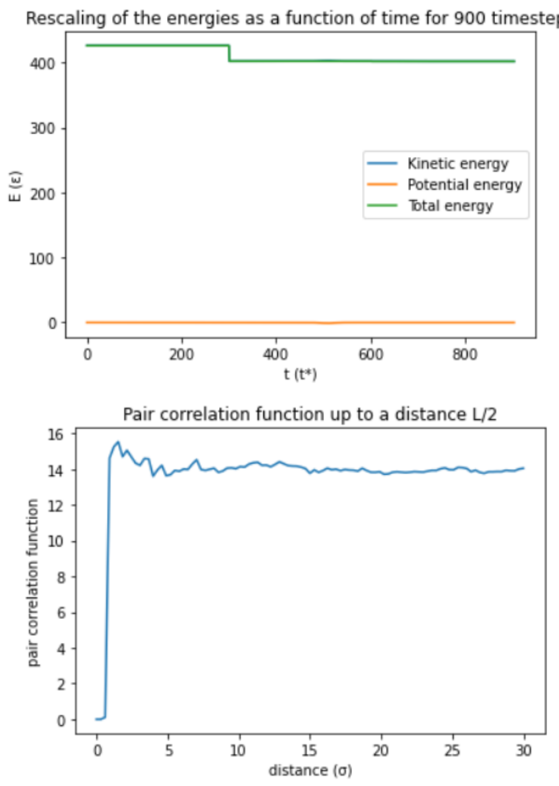


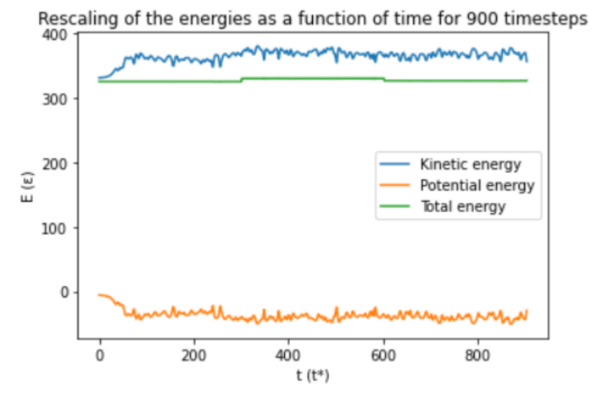
Figure 1: Rescaling of the energies and pair correlation function for simulation $\mathcal{B}$.

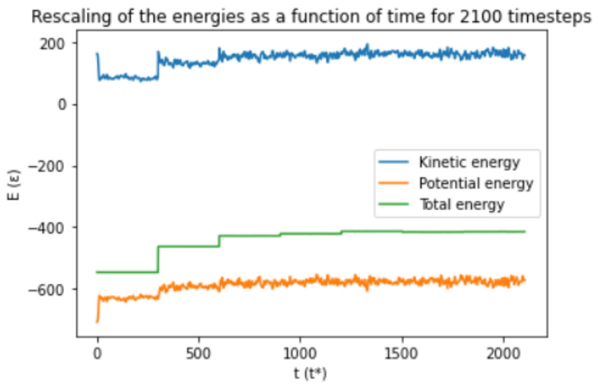
III.C. Code performance

Some modifications to the code have been done after *Week 5* since we still wanted to build a solid script that ran the code efficiently according to the observables we wanted to obtain. The main improvement we have done is merging all the computations that needed a time evolution storage into the same Verlet loop (see `merge` function). This fact, a part from giving us faster performance of the code (in the sense that we did not have to repeat the time evolution when we wished to compute every single observable), made clearer the flow of the final `main.py` script with which we ran the code and obtained the final results presented in this work.
Throughout our computations, we have used a uniform value of the time step $h = 0.01$ which would be equivalent to $\Delta t = 2.15 \times 10^{-14}\,\mathrm{s}$ which coincides in order of magnitude with what other authors used in their work [4]. The rescaling number of steps (`loopnum`) was also chosen in accordance with this previous cited reference (300) although other sources in literature even suggest an even smaller number of time steps [1]. The precision for the convergence of the temperature when the rescaling was performed was also taken the same value in all the computations we carried out, namely $|\lambda - 1| < 0.005$ and the number of x-axis data points used to obtain the pair correlation function figures was also always chosen to be 100. The reason of this choice is that it is the same order of magnitude as we found in other analogous computations (for example, in [1] they choose 200) and we perceived that the obtained pair correlation plots were clear enough in order to discuss the ensemble's behaviour.
The time required to preform the rescaling of the velocities of the system obviously depends on how many rescales we need in each run, but a rough time average of $50\,\mathrm{s}$ per rescale was observed. The number of steps that we used to carry on with the time evolution simulation (i.e. after the rescaling of the velocities was done) was of 2000 and 20000 steps. These choices are also backed up by literature, accordingly, some works have been carried out successfully with less steps in time (1200) [4]. We could reduce the computation of our code reducing the time steps used both for the rescaling and the time evolution of the system but, from the modification of this latter, the results may start to get distorted due to lack of statistical sample. Further code optimization could also be performed to our code although we have intended to minimize computation time throughout its development. A rough final average computation time for the whole program is of about $400\,\mathrm{s}$ for $l = 2000$ Verlet time steps and $3200\,\mathrm{s}$ for $l = 20000$

IV. RESULTS

In all of our computations we have restricted ourselves in using $N = 108$ particles, which is equivalent to choose the `cell_parameter_integer` $\frac{L}{a} = 3$ in our code since the number of particles is restricted to be $N = 4(\frac{L}{a})^3$ because of the 4 particles per cell that the fcc lattice imposes. The way we tune the particle density of the system was indirect and it is done via the length of one side of our cubic system L. All of this can be easily seen in the `main.py` file. The code accepts either integer or





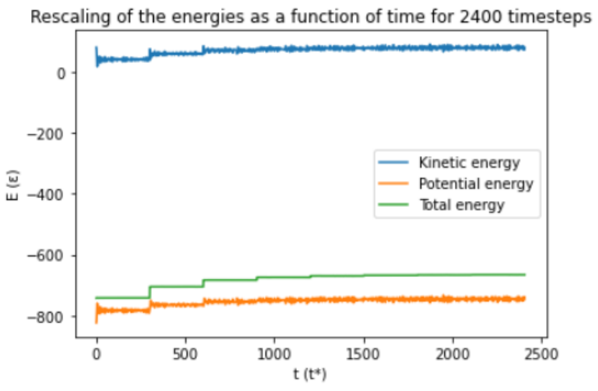
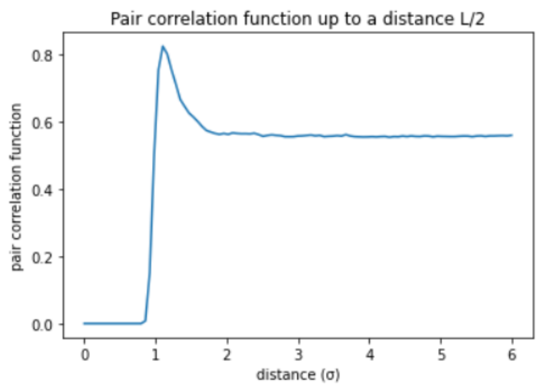
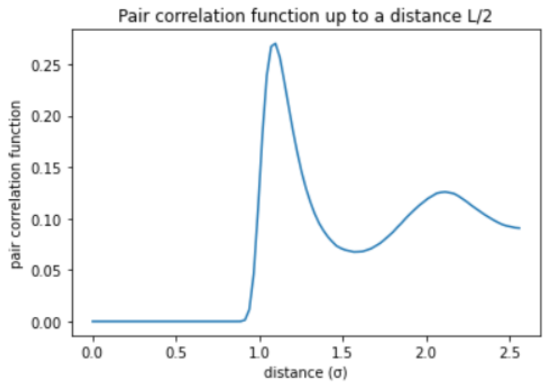


Figure 2: Energy rescales for the simulations $\mathcal{D}$, $\mathcal{E}$ and $\mathcal{F}$ from top to bottom respectively.





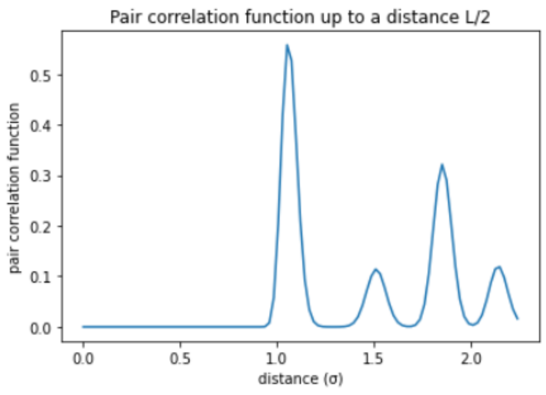


Figure 3: Pair correlation functions for the simulations $\mathcal{D}$, $\mathcal{E}$ and $\mathcal{F}$ from top to bottom respectively.

floating point number as values of L and a Varying the temperature was immediate.

We may first proceed with the two runs of the simulation that we carried out with low number density and high temperature input values. We may call these simulations

$$\mathcal{A} : l = 20000, L = 60, T = 300.2(4)\,\mathrm{K} \rightarrow \rho = 0.0005$$

and

$$\mathcal{B} : l = 2000, L = 90, T = 500.1(2)\,\mathrm{K} \rightarrow \rho = 0.0002$$

Note that the input values of the temperature for both simulations were different, we obtain the real ones from the actual system taking ensemble averages of the kinetic energy and l is the number of time iterations in the Verlet time evolution (`loopnum`). Given that the results obtained happened to be very similar, we will only show some of the relevant results for simulation $\mathcal{B}$ and comment if simulation $\mathcal{A}$ was different in a sense. The energy resacling and the pair correlation plots can be seen in figure 1. We can observe that indeed, when energy is not being rescaled the total energy is conserved along the time evolution of the system. We can also see that, as we should expect, in this ideal gas limit the potential energy goes to 0 and therefore all the energy is kinetic and it is also constant in time. A final hint that tells us that the ideal gas behaviour is satisfied is the pair correlation

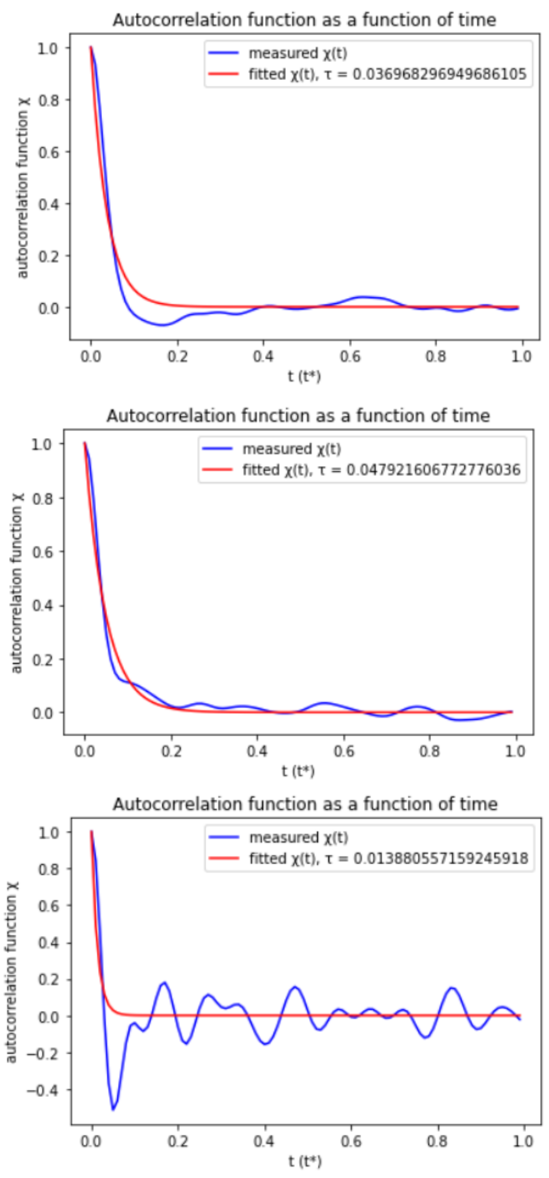


Figure 4: Pressure auto-correlation functions for the simulations $\mathcal{D}$, $\mathcal{E}$ and $\mathcal{F}$ from top to bottom respectively.

function, which is roughly constant once it rises after the repulsive forbidden region that Lennard-Jonnes creates on each particle.
The pressure values obtained for each of the simulations have been $p = 52\,410(10)\,\mathrm{Pa}$ for $\mathcal{A}$ and $p = 25\,584(6)\,\mathrm{Pa}$ for $\mathcal{B}$. These values fairly agree with the theoretical prediction for the ideal gas equation

$$p = \rho R T \tag{16}$$

with $R \approx 8.314\,\mathrm{JK^{-1}mol^{-1}}$, which are respectively $p = 52\,457\,\mathrm{Pa}$ and $p = 25\,879\,\mathrm{Pa}$ for $\mathcal{A}$ and B. We note that in the "ultrathermodynamic" case $\mathcal{B}$ the agreement is within the error.
The values of the specific heat per particle are found to be $c_V = 1.50(6)$ and $c_V = 1.50(6)$ again for $\mathcal{A}$ and B respectively. We are also satisfied with these results because both of them agree (within the error) with the c_V for ideal gases, which dictates that $c_V = \frac{3}{2}$. The second check we wanted to look is obtaining similar results as authors in [3][1]. We carried out the simulation

$$\mathcal{C} : l = 2000, L = 5.2, T = 118(6)\,\mathrm{K} \rightarrow \rho = 0.77$$

The result obtained was $\frac{p}{\rho k_B T} = 1.17(4)$ which is clearly between the two last entries of Table 8.1 in ¡[3][1].
Finally we proceed to obtain our own results for the simulation and analyze the different phases that Argon undergoes depending on the number density ρ and the temperature T of the system. We have carried out three simulations, with different input values for which we know the phase in which Argon should be, and we have run two same simulations for these same input values, only changing the number of time steps of the Verlet time evolution, namely, $l = 2000$ and $l = 20000$. Again, we will show results only for the $l = 20000$ case and comment if we observed any particular discrepancy between the high sampled and the low sampled simulation. We will call the different input values simulations

$$\mathcal{D} : L = 12, T = 275(2)\,\mathrm{K} \rightarrow \rho = 0.065$$

$$\mathcal{E} : L = 5.12, T = 119(6)\,\mathrm{K} \rightarrow \rho = 0.805$$

and

$$\mathcal{F} : L = 4.48, T = 60(4)\,\mathrm{K} \rightarrow \rho = 1.201$$

As the reference says, these $\mathcal{D}$, $\mathcal{E}$ and $\mathcal{F}$ simulations correspond to gas, liquid and solid states of Argon respectively (see [3][1] pg. 224).
If we first look at Figure 2 we can see the rescaling of the energies (velocities) in each one of the simulations. Again, we can see that during a time evolution with no rescaling in the middle, the total energy is perfectly conserved, as it should be, and the gas phase has a larger total energy and it decreases when we first go to the liquid phase and then to the solid phase.
Figure 3 is key to determine the phase in which Argon can be found. The results perfectly agree with what we should obtain; we clearly see that in all phases the pair correlation exhibits a maximum near the position of the minimum of the Lennard-Jones potential and for the gas state it slightly goes down and then it remains constant plateau-like, as it should. [2] The liquid phase is also very alike to what a typical liquid should give (see [2] pg. 155), as well as the solid phase, showing peaks approximately where the typical nearest-neighbours distances are located in the fcc lattice.
Values for the pressure for the If we go to Figure 4, we can see the auto-correlation function for the pressure computations simulations $\mathcal{D}$, $\mathcal{E}$ and $\mathcal{F}$ respectively gave $\frac{p}{10^6} = 5.71(1)\,\mathrm{Pa}$, $\frac{p}{10^6} = 59.9(4)\,\mathrm{Pa}$ and

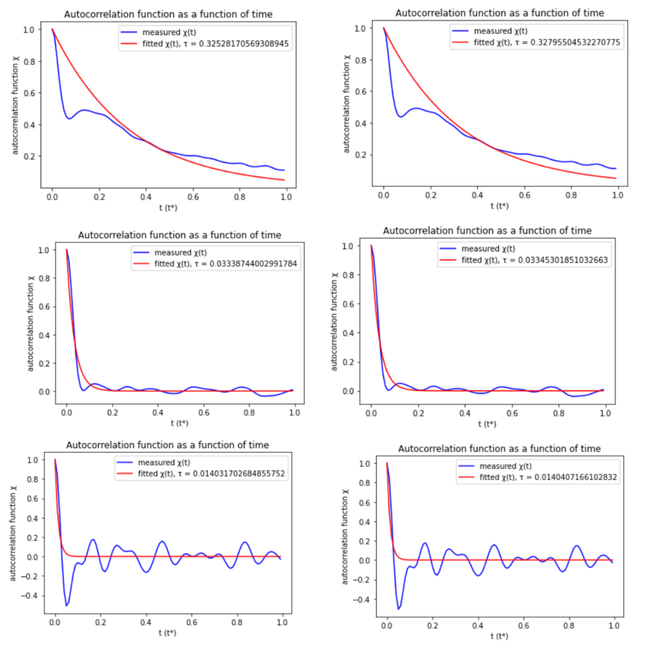


Figure 5: Kinetic energy (left) and kinetic energy squared (right) auto-correlation functions for the simulations $\mathcal{D}$, $\mathcal{E}$ and $\mathcal{F}$ from top to bottom respectively.

$\frac{p}{10^6} = 707.6(2)\,\text{Pa}$. We observe an increasing pressure when going from the gas state to the solid state, which is reasonable. Analogously, values for the specific heat per atom, in the same order, are $c_V = 1.5(9)$, $c_V = 2(2)\,\text{Pa}$ and $c_V = 3(2)\,\text{Pa}$. One thing we have noticed si that even though the computations were made with a rather large amount of loop steps ($l = 20000$) we get a very big statistical error in the specific heat, which is even bigger for the cases we studied with $l = 2000$. However, the gas specific heat is in agreement with ideal gas c_V (previously mentioned) and the solid one is in agreement with Dulong-Petit law ($c_V = 3$).
Finally, we can analyze the auto-correlation plots for the statistical observables, i.e. the pressure (Figure 4) and the kinetic energy and kinetic energy squared (Figure 5). We can see that the pressure exhibits a correlation of $\tau^* = \frac{tau}{h} = 100\tau$ time steps, which is equal to, approximately, 4, 5 and 2 time steps for the gas, liquid and solid state respectively. With a total of $l = 20000$ time steps we obtain pressure values for a statistical independent sample of about 5000, 4000 and 10000 samples, which is enough to obtain reliable statistical values as we could see. An analogous rough calculation could be made for the correlation of the kinetic energy and the kinetic energy squared samples but the study associated with the error obtained in the c_V is not so immediate due to the fact that its error has to be propagated from these original kinetic energy ones.

V. CONCLUSIONS

We have successfully recovered some important results for the analyzed observables which give support to the fact that our model for the Argon particle ensemble is correct. We have been able to study the three different phases that Argon exhibits quite clearly and the theoretical predictions are contemplated within the errors of the magnitudes we observe, such as the ideal gas pressure, the mono-atomic ideal gas c_V or the high temperature solid limit for the c_V (Dulong-Petit law). Although we have seen that we did not need to increase the number of steps in time in our simulation to get correct enough results, we may note that no simulation for more than $N = 108$ particles was carried out. Noticeably, the choice of using this amount of particles in our system was not reasonably chosen, it was rather used as the maximum amount of particles for which we could carry out the simulation in a reasonable amount of time. However, given that the results obtained were quite satisfactory with what we expected to obtain we can conclude that a model with this amount of particles in this case is enough to make a proper description.

[1] Jos Thijssen. Molecular dynamics simulations. In *Computational physics*, chapter 8, pages 210–300. Cambridge University Press, Oxford, 2007.
[2] M. Plischke and B. Bergersen. *Equilibrium Statistical Physics (3rd Edition)*. World Scientific Publishing Company, 2006.
[3] Jos Thijssen. Classical equilibrium statistical mechanics. In *Computational physics*, chapter 7, pages 180–209. Cambridge University Press, Oxford, 2007.
[4] Loup Verlet. Computer "experiments" on classical fluids. i. thermodynamical properties of lennard-jones molecules. *Physical review*, 159(1):98, 1967.