École polytechnique fédérale de Lausanne

Computer simulation of physical systems I

Fall 2021

Report Task II and VI

Bruno Rodriguez Carrillo
sciper 326180

Professor:
Alfredo Pasquarello

December 23rd, 2021

1 Task II: NVE molecular dynamics simulations

This exercise is about molecular dynamics (MD) simulations of a system of atoms interacting through a Lennard-Jones(LJ) pair potential in the NVE statistical ensemble. Based on the Task II sheet, we perform the following:

1. First, we generate the starting atomic positions and the starting velocities; then we change the system temperature to the desired value and take the sample to equilibrium. We equilibrate the sample by running an evolution of the NVE dynamics. Once the sample has reached equilibrium at the desired temperature, we use this sample for the all our simulations and computations
2. From the previous sample, we sample two static properties: the radial pair correlation function $g(r)$ and the structure factor $S(k)$, computed as the radial Fourier Transform (FT) of g(r) by the Fourier transform and direct sampling
3. At our final step, we compute one dynamical property; we estimate the diffusion coefficient D using two different methods, namely, Einstein's relation, $MSD(t) = C + 6Dt$, which involves the sampling of the mean square displacement (MSD) and a fitting procedure for C and D, and the time integral of the velocity autocorrelation function VACF

We should clarify that the in the second section, we compute $g(r)$ and (k) by using the so-called Monte Carlo method.

The MD codes provided for this exercise solve numerically the equations of motion for a Lennard-Jones (LJ) fluid made of N particles contained in a 3D periodically repeated rectangular box and interacting through the pair potential:

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

The codes use internally the so-called LJ units (i.e. $\sigma = 1$ and $\epsilon = 1$), so that distances are expressed in σ units, while energies are given in ϵ units. Therefore, once the parameters of the the LJ potential have been fixed, the units for all remaining physical quantities are also fixed, as listed in the Table(1).

We make use of the following conversion factors from LJ units to SI units and vice-versa:

quantity	distance	energy	velocity	temperature	time
LJ units	σ	ϵ	$\left(\frac{\epsilon}{M}\right)^{1/2}$	$\frac{\epsilon}{k_B}$	$\sigma\left(\frac{\epsilon}{M}\right)^{-1/2}$

Table 1: Unit equivalences, LJ and SI units.

We shall mention that LJ units are used in all our simulations; to transform them into SI units, we consider the following values given in Table(2). Since we are required to reproduce some of the results by Rahman, we set our parameters as follows.

For all NVE simulations, we fix the integration step size to the value in Table(2) but in LJ units, 0.0046. Similarly, we can compute the size of the simulation box as $L = 10.229$ in LJ.

We now describe and add a few figures to see how the energy and temperature fluctuate after running our equilibration simulation at the desired temperature.

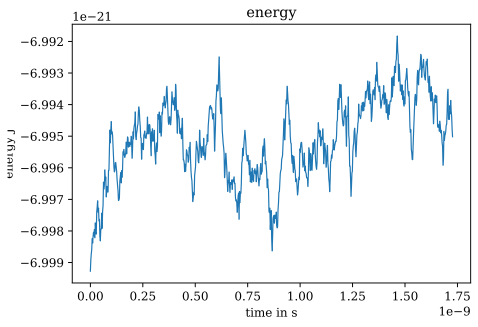
Number of atoms	864
Number of cells along each axis of the box	6
$\rho_{Ar,liquid}$	$1.374\ g \cdot cm^{-3}$
Desired temperature T	$94.4\ K$
atomic mass$_{Ar,liquid}$, M	$6.69 \cdot 10^{-23}$g
ratio $\frac{\epsilon}{k_B}$	$120\ K$
Boltzmann constant, k_B	$1.38064852 \cdot 10^{-23} m^2 kg s^{-2} K^{-1}$
Value for σ	$3.4 \cdot 10^{-10} m$
Cut-off radius, $r_{cut-off}$	2.5
Integration time step	$10^{-14}\ s$
Number steps, NVE runs	800
Number steps, $g(r)$ and $S(k)$	2000

Table 2: Parameters for simulation.

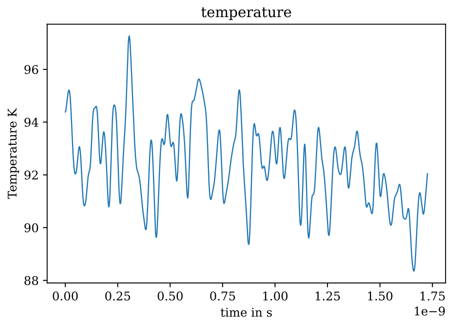
1. As seen, during the NVE run the temperature shows fluctuations and sometimes it is not close to the target temperature $T = 94.4K$. As expected, the temperature of the system can not be fixed to the desired one (94.4 K) since this simulation corresponds to a NVE simulation; furthermore, from the same figure, we can visualize that in fact the value of T oscilliates a little faraway from 94.4 K.

   From this run, we obtained a data file. This is an equilibrated atomic sample which will be used during the sample static and sample dynamical properties sections hereafter. This is really important to bear in mind: all of our following computations make use of such a data file.

   As a result and observing Fig.(1), we may say that energy is conserved.



(a) Energy.



(b) Temperature.

Figure 1: Total energy and temperature for our equlibrated sample.

We now compute our two static properties of interest: the pair correlation function $g(r)$ and the structure factor $S(k)$.

1.1 $g(r)$ and $S(k)$ computation

The radial pair correlation function $g(r)$ represents the density as a function of the distance r from a given atom I to its neighbors. Note that the density function is defined in the lectures as follows:

$$g(r) = \frac{\rho(r)}{\rho_0} \tag{2}$$

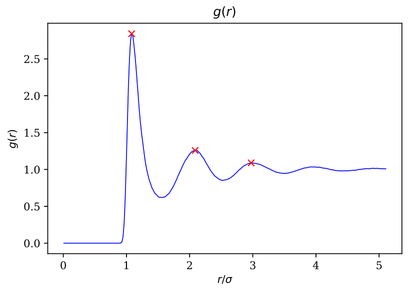
where ρ_0 corresponds to the average density. From Eq.(2), we recall two properties of the $g(r)$ function:

$$\lim_{r\to+\infty} g(r) = 1 \text{ and } \lim_{r\to 0} g(r) = 0$$

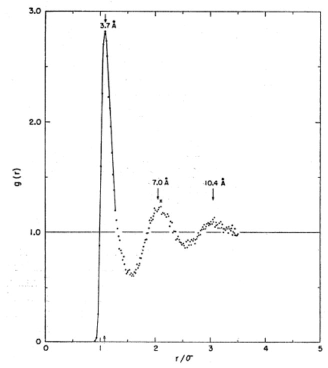
such properties will be corroborated during our simulations.

From the sample that is already at the desired temperature $T = 94.4\ K$, Fig.(1), we focus on $g(r)$ and $S(k)$. The latter can be obtained in two modes, either directly by sampling the Fourier transform (FT) of the number density, or, in the case of an isotropic system, as the FT of the pair correlation function, $g(r)$. We will proceed in both forms.

After running our simulation with the parameters given above, we have the following plot (left hand side) and a comparison to Rahman's:



(a) $g(r)$ i our simulation.



(b) $g(r)$ in Rahman's paper.

Figure 2: $g(r)$ function in our simulation and in Rahman's.

We observe that the units on the x-axis in Fig.(2) are r/σ; the peak values correspond to the red ones. Since we know that $\sigma = 3.4$Å, we can compute the values of the three observed peaks by simply using $r = \sigma \cdot u_{g(r),max}$, where $u_{g(r),max}$ represents the value on the x-axis in Fig.(2) where $g(r)$ reaches one of its first three maximum values. Then we have the following results: peak 1 $\sigma = 3.681$Å, peak 2 $\sigma = 7.10$Å and peak 3 $\sigma = 10.115$Å.

The first peak corresponds to the atomic distance between particles and the waves represent the gap between particles that are in a face-centered cubic disposition. Moreover, we can observe the two properties o of the properties of the $g(r)$ function aforementioned.

Furthermore, we have the following chart to compare our results, ”simulation” column, to Rahman’s:

r value in argmstrong	Rahman’s results	Simulation
r_{peak_1}	3.7	3.681
r_{peak_2}	7.0	7.10
r_{peak_3}	10.4	10.115

Table 3: Values of peaks for $g(r)$ function

Now we compare our computation of the structure factor $S(k)$ using the Fourier transform and direct sampling. For comparison purposes, we plot both graphs in a single plot:

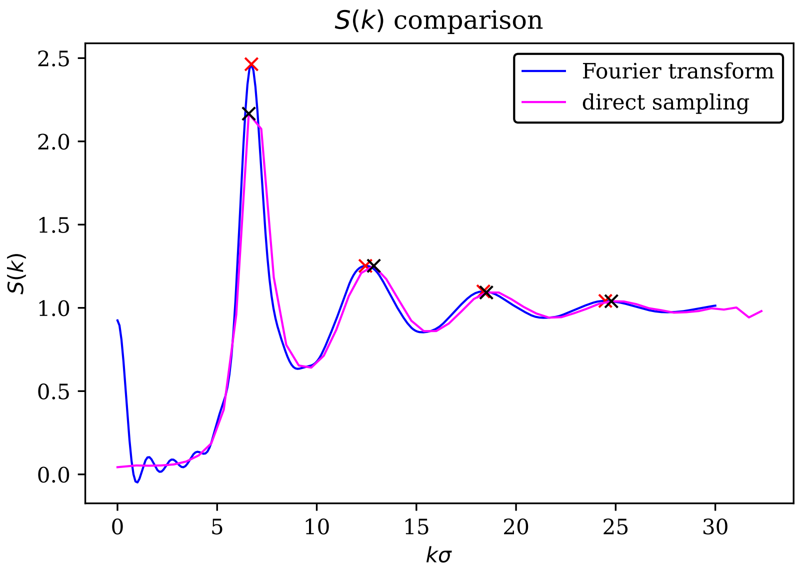


Figure 3: $S(k)$ function.

and we have the following values to compare to Rahman’s:

peaks in $S(k)$	Rahman’s results	Fourier transform	Direct sampling
$k_1\sigma$	6.8	6.722	6.588
$k_2\sigma$	12.5	12.441	12.863
$k_3\sigma$	18.5	18.361	18.510
$k_4\sigma$	24.8	24.482	24.784

Table 4: Values of peaks for $S(k)$ function by Fourier transform and direct sampling.

From Fig.(3), we clearly observe that the Fourier transform captures more oscillations before $S(k)$ reaches its first peak and has a smoother shape or transition; on the other hand, direct sampling seems to not reproduce fully the behaviour of $S(k)$ as $k\sigma$ goes to infinity. This is evident near the first peak in Fig.(3), where there is a clear difference between the value of $S(k)$ computed by both methods. The direct sampling computation is significantly more expensive than computing the Fourier transform.

1.2 Diffusion coefficient D

The study of the dynamical diffusion coefficient is accessible through MD simulations. This quantity can be computed from the mean square displacement (MSD) of the atomic positions through Einstein's relation, or from the integral of the velocity autocorrelation function (VACF).

From the lecture notes, we have the Einstein relation given as:

$$\frac{\partial}{\partial t}\langle r^2 \rangle = 6D \tag{3}$$

where:

$$D = \lim_{t \to +\infty} \frac{1}{6t} \langle \| \Delta \overrightarrow{r}(t) \|^2 \rangle \tag{4}$$

and $\langle \| \Delta \overrightarrow{r}(t) \|^2 \rangle$ is the mean square displacement and it is given by the following expression:

$$\langle \| \Delta \overrightarrow{r}(t) \|^2 \rangle = \frac{1}{N} \sum_{J=1}^{N} \| \overrightarrow{r_J}(t) - \overrightarrow{r_J}(0) \|^2 \tag{5}$$

As we did to compute $g(r)$ and $S(k)$, we run our simulations from a sample that is already at the desired temperature, $T = 94.4\ K$. After running the provided code with the parameters in Table(5), we obtain the following results in SI and LJ units.

Number of steps	integration step dt	Number of simulations	cut
900	$dt = 5 \cdot 10^{-4}$	20	400

Table 5: Parameters to compute MSD and VACF.

The number of steps corresponds to how many points, separated by a integration step dt, we compute for each of these 20 different runs; number of simulations indicates how many independent simulations we run and cut corresponds to the time step from which we compute the linear fit of the mean square displacement MSD . That is, since our cut value is 400, we use the values of MSD -computed in each run- from the 400 time step and on.

We mention that we modified the parameters given in the original code since we are computing dynamical quantities and thus the description of the particle trajectories are important and should compute sufficiently accurate.

To compute the diffusion coefficient from the linear fit, we proceed as follows. From the values computed over all 20 runs, we fit our data to the model $C + 6Dt$; this model is in LJ units in the provided codes and to transform it into SI units, we do the following for the slope.

$$\text{slope SI} = 6D = \text{slope LJ} \cdot \frac{\sigma^2}{\Delta u} \left(\frac{3.4 \cdot 10^{-10} m}{1\sigma} \right)^2 \left(\frac{1\Delta u}{a\ s} \right) \left(\frac{100cm}{1m} \right)^2$$

where Δu is time in LJ units and $a = 3.4 \cdot 10^{-10} m \left(\frac{120K \cdot 1.38 \cdot 10^{-23} m^2 kg/s^2 K}{6.69 \cdot 10^{-26} kg} \right)^{-1/2}$. Then we have a value slope SI $= 1.6098 \cdot 10^{-4} cm^2/s$. Finally, we have

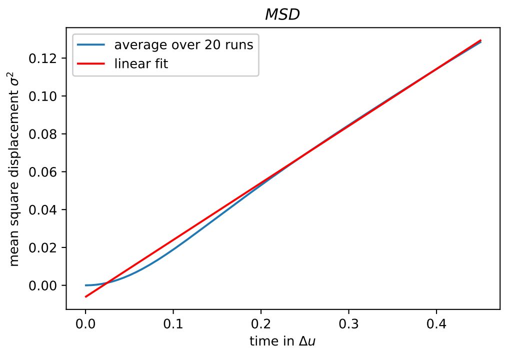
$$\text{slope SI} = 6D \to D = \text{ slope SI}/6 = 2.6831 \cdot 10^{-5} cm^2/s$$

Using a similar procedure, we transform the intercept value C into SI units, which gives $C = -7.0624 \cdot 10^{-18}\ cm^2$ and as a consequence we have the following plots of the linear fit in LJ and SI units, Fig.(4). Furthermore, the standard error of the slope computed from linear fit is $error_{slope} = 2.6484 \cdot 10^{-4}$.

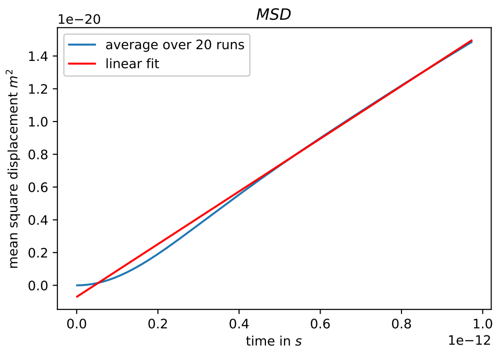
On the other hand, we can also compute D by simply averaging the value computed in each of the runs; this gives the following value in SI units:

$$D = 2.6838 \cdot 10^{-5} cm^2/s$$

and the standard deviation of D from averaging the D values of the 20 runs is $std = 1.1239 \cdot 10^{-6}$.



(a) MSD in LJ units.



(b) MSD in SI units.

Figure 4: MSD computation plots.

In Fig.(4), we can clearly observe at initial times the nonlinear regime where the particles moves freely until they start interacting with each other. Additionally, both computation methods achieve similar results and accuracy, which in turn are close to the values reported by Rahman. We keep the value of D computed by averaging in Table(6).

Next section, we turn to computing the diffusion coefficient by integrating the velocity autocorrelation function $VACF$.

1.3 Velocity autocorrelation function ($VACF$)

This function represents the correlation between the particles velocity at time t_i compared to the velocity at time $t_0 = 0$. Thus, when the system has just started, the velocities at t_1 are very similar to the velocities at $t_0 = 0$. At infinity, the correlation is equal to 0. Note that we can see a rebound before the stabilization of the correlation function at 0. This corresponds to the time the particles need to find their correct positions in the lattice. We can obtain the diffusion coefficient D by integrating the velocity correlation function as follows:

$$D = \int_0^\infty d\tau \langle V_x(\tau) V_x(0) \rangle = \int_0^\infty d\tau \frac{1}{3} \cdot \langle \mathbf{v}(\tau), \mathbf{v}(0) \rangle \tag{6}$$

where we should indicate that we are working with a 3-dimensional cube, we assume an isotropic system, which gives the 1/3 factor in Eq.(6) and $\langle \cdot, \cdot \rangle$ is the usual inner product between two vectors.

Let us now recall the definition of the normalized velocity auto-correlation function:

$$VACF = \frac{\langle \mathbf{v}(\tau), \mathbf{v}(0) \rangle}{\langle \mathbf{v}(0), \mathbf{v}(0) \rangle} \tag{7}$$

Eq.(6) corresponds to the so-called Green-Kubo formula. To compute Eq.(6) numerically, we perform the following sum for each run j-th, which is the procedure implemented in the provided codes:

$$D_{run_j} = \frac{1}{N_{runs}} \cdot \sum_{i=1}^{N_{steps}} C_{VACF} \cdot d\tau \cdot \frac{1}{3} \cdot \langle \mathbf{v}(\tau_i), \mathbf{v}(0) \rangle$$

where C_{VACF} is a constant due to unit transformation, we assume an isotropic system and j goes from 1 to Number of simulations (20). Specifically, in our case, $d\tau$ corresponds to the time step dt and $\mathbf{v}(\tau_i)$ represent the velocity at time step τ_i.

As a result, we compute D as an average of the diffusion coefficient from the previous 20 runs, that is:

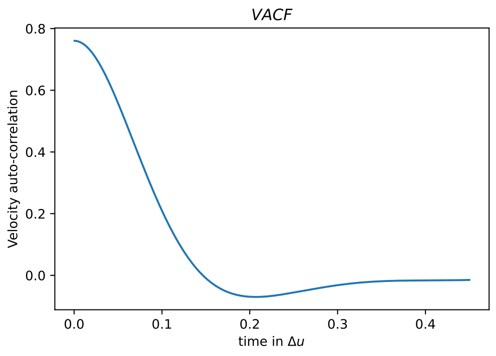
$$D = \frac{1}{N_{runs}} \cdot \sum_{j=1}^{N_{runs}} D_{run_j} \tag{8}$$

Using Eq.(8) and after running the simulations, we have the following value for D in SI units:

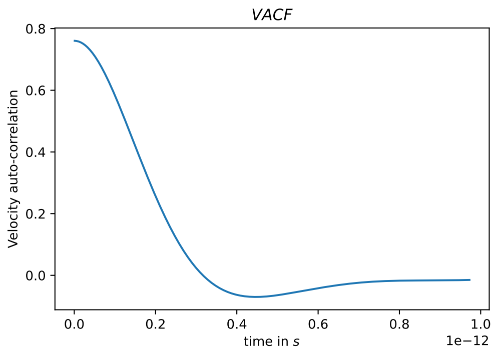
$$D = 2.499 \cdot 10^{-5} \frac{cm^2}{s} \tag{9}$$

Moreover, we compute the standard deviation from such a value of D, $std = 1.525 \cdot 10^{-6}$.

From the simulations, we have the following plots in LJ and SI units:



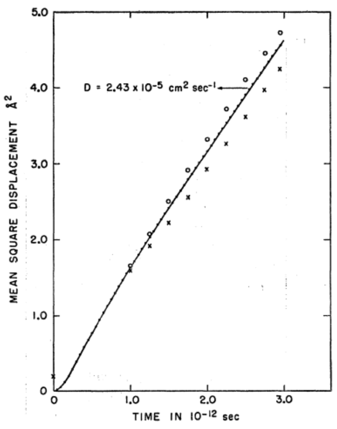
(a) VACF in LJ units.



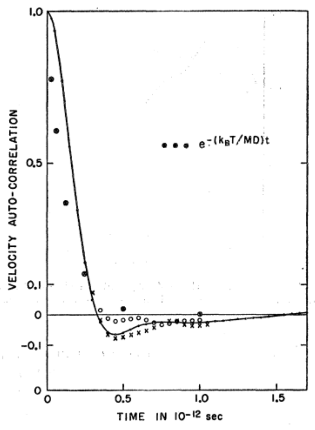
(b) VACF in SI units.

Figure 5: $VACF$ computation.

Rahman's results, on the other hand and for comparison purposes, are as follows:



(a) MSD.



(b) VACF.

Figure 6: Rahman's results for MSD and VACF.

The following table summarizes our results of the D computation and compares to Rahman's:

Rhaman's	MSD	std	VACF	std
$D = 2.43 \cdot 10^{-5}$	$D = 2.6838 \cdot 10^{-5}$	$1.123 \cdot 10^{-6}$	$D = 2.499 \cdot 10^{-5}$	$1.525 \cdot 10^{-6}$

Table 6: Values of D via MSD and VACF.

For all values of D in Table(6), the units are cm^2/s and std represents the standard deviation. We also note that although the values of D computed by either of the methods are close to the value reported by Rahman, the value computed by MSD seems to be different from the rest. There can be two main reason why this happens. First, the integration step may not have been sufficiently small. Second reason, the total number of samples or cut value could not have been the appropriate ones.

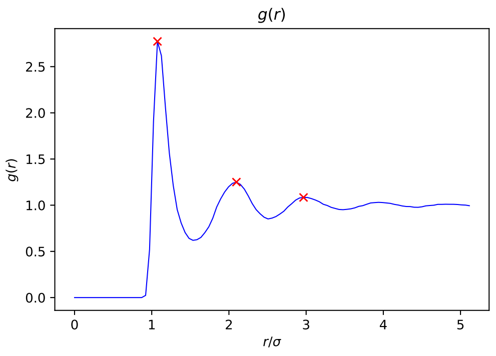
2 Task VI: Monte-Carlo simulation of Lennard-Jones liquid

2.1 $g(r)$ and $S(k)$ computation

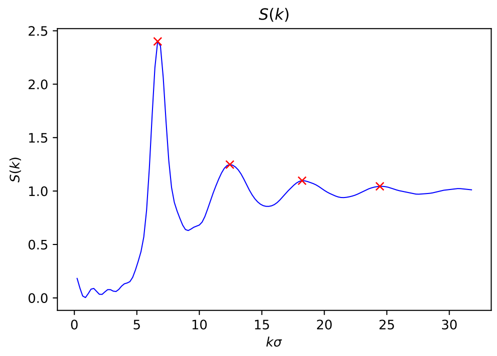
We now turn to the computation of the pair correlation function $g(r)$ and structure factor $S(k)$ by using random sampling, i.e, Monte Carlo simulation. For this purpose, we use exactly the same parameters as the one used in Task II. There exist a few additional parameters that we need to work on and are intrinsic to the Monte Carlo integration approach. The Monte Carlo simulation is performed using a sampling size similar to the correlation length obtained in the block average step (see next section), the number of production cycles is set to 2000 (this is the number of samples that we actually compute) and the number of equilibration cycles is fixed to 10000. It is important to note that after the equilibration cycles, we had an acceptance rate of 50%, which corresponds to the optimal value.

We point out that to compute these quantities, we start our simulation from a first simulation that is already at equilibrium with the Rahman's parameters. That is, we first run our code with the number of equilibration cycles. From the generated data, we compute $g(r)$ and $S(k)$. By following a similar procedure to the one used in Task II, we can compute the values of r such that $g(r)$ reaches its three maximum values.

After running the provided codes, we have the following plots:



(a) $g(r)$ function.



(b) $S(k)$ function.

Figure 7: $g(r)$ and $S(k)$ functions via Monte Carlo.

and the following charts summarizing our results and Rahman's:

r value in argmstrong	Rahman's results	Simulation
r_{peak_1}	3.7	3.652
r_{peak_2}	7.0	7.13
r_{peak_3}	10.4	10.086

Table 7: Values of peaks for $g(r)$ function via Monte Carlo

For the function $S(k)$:

peaks in $S(k)$	Rahman's results	Fourier transform
$k_1\sigma$	6.8	6.667
$k_2\sigma$	12.5	12.444
$k_3\sigma$	18.5	18.222
$k_4\sigma$	24.8	24.444

Table 8: Values of peaks for $S(k)$ function by Fourier transform via Monte Carlo.

Finally, as we have observed, by performing our simulations by NVE(Task II) and Monte Carlo (Task VI), we obtained results that are close to those reported by Rahman as long as the simulation parameters are equal to the ones Rahman used for his paper. In addition, we corroborated that by using random sampling we could compute two static properties correctly, which was expected since the Monte Carlo approach works when it comes to static properties since these do not depend on the trajectories of the particles; in fact, the trajectories computed by such a method are fictitious, that is, they have no physical meaning. This also means that we cannot use Monte Carlo to compute dynamical properties such as the diffusion coefficient, which at the same time shows one advantage of MD simulation over random sampling.

However, we must mention that Monte Carlo has several advantages over deterministic numerical integration. Two of which are its simplicity to implement (we require independent and identically distributed samples according to a given/target distribution) and its non-dependence on a system's dimensionality.

2.2 Blocking analysis

For this section, we initially choose a temperature of 2.0 in LJ units, corresponding to 240 K. The density is set to 0.5, which in SI corresponds to 0.851 g/cm^3 and the number of particles is 200. As an example, we first run our program to observe when the energy and pressure stay close to a specific value. For this run, the number of samples for the Monte Carlo simulation (*lmax* in the code) is set to 10000 and the number of equilibration cycles (*nequil* in the code) to 10000. To illustrate the energy evolution of our system, we have the following plot:

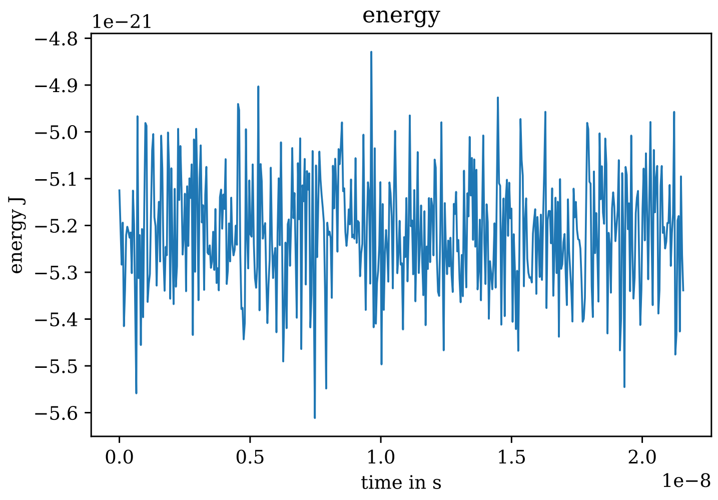


Figure 8: Energy evolution, using $nsamp = 10$.

The blocking analysis we perform afterwards will be for energy only. Furthermore, during the first run in Monte Carlo, the step length is adjusted in order to attain a more efficient acceptance rate, which corresponds to 50% approximately.

On the other hand, for blocking analysis, we keep the same values for temperature, number of particles and density as before, but now we use 100000 samples, number of equilibration cycles 10000, and we vary the number of steps between samplings ($nsamp$ in Fig.(9)), which gives us:

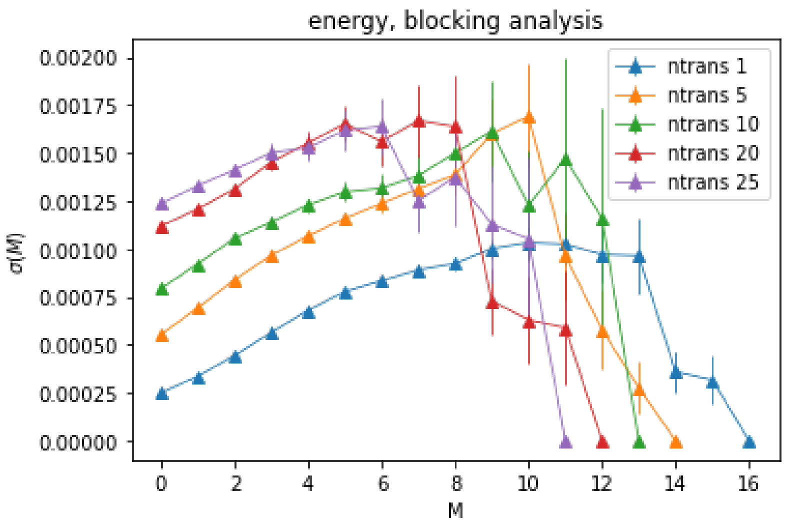


Figure 9: Blocking analysis as a function

From picture (9), we observe that there is a plateau zone when M is between 6 and 14 when using $nsamp = 1$. Besides, we note that the error increases as M does, which was expected since as M gets larger fewer blocks we have then the standard deviation increases. In both previous pictures std stands for standard deviation. We select $nsamp = 10$ to compute $S(k)$ and $g(r)$ in the previous section.

It is quite important to say that the blocking analysis is required to know where or how far we can find uncorrelated samples from each other; and this is is one of the requirements of the samples for Monte Carlo to be applied: we average over the configurations that are uncorrelated.