Classical Molecular Dynamics Simulation using Lennard-Jonnes potential

Project Report submitted to Prof Venugopal Achanta, TIFR, Mumbai.

Department of Physical Sciences

Indian Institute of Science Education and Research Berhampur

Shishira Mahunta[1] and Shreya Dwivedi[2]

[1]Roll No. 17096
[2]Roll No. 17100

April 12, 2021

Contents

List of Figures

0.1 Introduction

Molecular dynamics simulation is a numerical method for studying the dynamics of many-particle systems such as molecules and clusters. The core of most simulations is to start with the initial positions and velocities of all particles and then repeatedly apply a method to update each particle's position and velocity from time t to time t + Δt. Comparing the inter-particle distance to the de-Broglie wavelength, we find that the de-Broglie wavelength is much smaller compared to the distance between the particles therefore the motion of the atoms and ions can be taken in the classical regime. However, the interaction between the atoms and molecules arises from the electronic structure hence it requires quantum mechanics. Therefore, there are two different approaches to solve the problem- classical molecular dynamics(MD) and ab-initio molecular dynamics.
In classical molecular dynamics, interactions are approximated by classical model potentials. It involves solving the equations of motion of the system of particles using Newton's Laws of motion.It leads to simulation of purely classical many-particle problem and it works well for simple particles such as noble gases and poor for metals due to presence of free electrons. In this case simulations are fast and allow large number of particles. The impact of temperature is often included and the interplay between this thermal energy and the intermolecular energy gives

rise to the self assembly processes.

0.2 Basics of Molecular Dynamics

Using classical approximations that is, Newtonian mechanics, we describe the energy of the system as a function of the positions of the particles E(r).Newton's second law is used to describe the motion of the particles through time:

$$F = ma = d^2v/dt^2 \tag{1}$$

In 1 dimension, the force can be calculated by taking the derivative of the energy with respect to x:

$$F = -dE/dx \tag{2}$$

or in 3 dimensions, by taking the gradient:

$$F = -\nabla E(r) \tag{3}$$

0.3 The Model

The model is specified by the net force F_i on each particle of mass m_i. The force F_i can be due to all other particles and additional interactions such as effective drag forces or interactions with a wall or an external field. In the following we will consider only conservative forces which are due to all the other particles. We assume pair-wise interactions given by the Lennard-Jonnes potential:

$$E_{LJ}(r) = 4\varepsilon[(\sigma/r)^{12} - (\sigma/r)^6] \tag{4}$$

σ is the radius where the potential is zero and ε is the minimum energy.
The long range interactions are negligible therefore we put a bound such that σ/r=2.5. However, as the particles move past the cutoff distance there will be a small jump in the energy, which is not realistic, so we shift the potential so that the potential goes to zero at σ/r=2.5
The simulation has been done in python3 using the python libraries numpy and matplotlib.

In figure 1 we show the plot of Lennard-Jonnes potential (written in python) alongwith the shifted potential to compare both of them.

0.4 Reduced Units

Units are chosen in order to remove any constants and get a general behaviour for all the gases. Mass, sigma, epsilon and the Boltzmann constant are set to equal

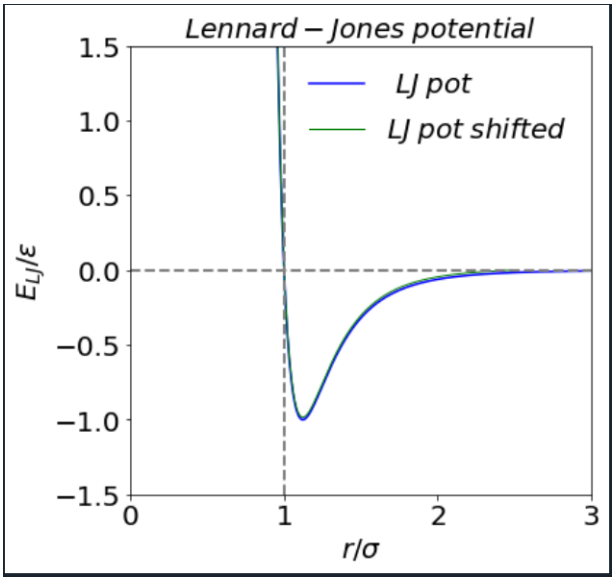

Figure 1: Lennard-Jonnes Potential Curve

one. Reduced coordinates are used for the other variables which are derived from the parameters set to one.

$$\begin{aligned} x^* &= \frac{x}{\sigma} \\ v^* &= v\frac{t^*}{\sigma} \\ t^* &= t\left(\tfrac{\varepsilon}{m\sigma^2}\right)^{1/2} \\ E^* &= \frac{E}{\varepsilon} \\ F^* &= f\frac{\sigma}{\varepsilon} \\ P^* &= P\frac{\sigma^3}{\varepsilon} \\ \rho^* &= \rho\sigma^{dimensions} \\ T^* &= T\frac{k_B}{\varepsilon} \end{aligned} \tag{5}$$

where k_B is Boltzmann's constant.

This may seem complicated but it allows all of the equations to be written very simply in the program. This also gives us physical insight - for example, the reduced temperature is the ratio of the thermal energy k_BT to the energy of the intermolecular interactions ε.

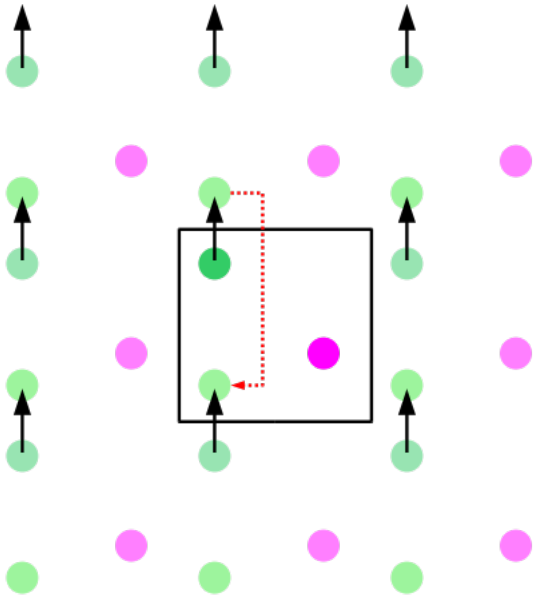

Figure 2: Simulated simple unit cell

0.5 Periodic boundary conditions

Periodic boundary conditions allow for an approximation of an infinitely sized system by simulating a simple unit cell. This is illustrated in the figure 2. The black box is the only cell we simulate; the tiled images around it are there for illustration. The green particle moves past the top boundary of the unit cell and are moved to the bottom of the box with the same velocity (illustrated by the red dashed line). This boundary condition keeps the volume and number of particles constant in the simulation.

0.6 Calculating the force and acceleration

In Newton's second law of motion, the force is a vector quantity, whereas the first negative derivative of the energy is a scalar. Therefore, it is important that we determine the force in each direction for our simulation. This is achieved by multiplication by the unit vector in that direction,

$$\mathbf{f}_x = f\hat{\mathbf{r}}_x, \text{ where } \hat{\mathbf{r}}_x = \frac{r_x}{|\mathbf{r}|} \tag{6}$$

In the above equation, r_x is the distance between the two particles in the x - dimension and $|\mathbf{r}|$ is the magnitude of the distance vector. For simplicity, we

will initially only consider particles interacting in a one-dimensional space.

The forces between particles can be calculated from the derivative/ gradient of their potential energy: $\mathbf{F} = -\frac{1}{r} \bigtriangledown E(\mathbf{r})$ (in spherical coordinates).

$$\mathbf{F} = -\frac{1}{r} \bigtriangledown E_{LJ}(\mathbf{r}) = -\frac{1}{r}\frac{dE_{LJ}}{d\mathbf{r}} = -24\left[2\left(\frac{\sigma}{\mathbf{r}}\right)^{14} - \left(\frac{\sigma}{\mathbf{r}}\right)^{8}\right] \tag{7}$$

Periodic boundary conditions have to be considered when we compute the forces between particles because a particle near the boundary of the unit cell has to be able to feel the force from a particle on the other side of the unit cell. For example, the pink particle above will feel the force from the green particle, even though they are far from each other because they are near opposite boundaries.

In order to easily implement periodic boundary conditions, scaled box units are used so that the particle positions are always between -0.5 and 0.5. If the distance between the particles is greater than half the scaled box units, the interaction with the particle in the next box are considered.

To substantiate the above point, a python code is written to determine the acceleration on each atom of argon due to other argon atoms. It is possible to increase the efficiency of this algorithm by applying Newton's third law, e.g. the force on atom i will be equal and opposite to the force on atom j.

The following acceleration for atoms of argon are obtained by the code ($calculate_force.py$) : $Accelerationonparticle0 = 1.453e-04eV/Åamu; Accelerationonparticle1 = -4.519e-05eV/Åamu' Accelerationonparticle2 = -1.002e-04eV/Åamu$.

0.7 Temperature

Temperature is one of the fundamental concepts in physics. It is used to measure the hotness or coldness of macroscopic objects.it represents the intensity of the thermal motion of molecules in microscopic theory.The average kinetic energy is widely used to characterize temperature in molecular dynamics simulation.Here we observe three type of average kinetic energy as measures of temperature i.e., the total kinetic energy, kinetic energy without the centroid translation part, and thermal disturbance kinetic energy.the thermal disturbance kinetic energy has wider applicability to temperature computation in non-equilibrium molecular dynamics simulation.

The definition of temperature strongly influences molecular simulation, especially molecular dynamics , in which the velocities of atoms are continuously adjusted according to various temperature-controlled algorithms. As molecular dynamics need the definition of the temperature out of the equilibrium so we need

to investigate the concept of temperature more carefully.In equilibrium statistical mechanics, the absolute temperature is proportional to the average kinetic energy. it has not yet been ascertained whether all of the components of kinetic energy contribute to temperature. Many researchers omit the rigid transnational kinetic energy from temperature calculation, or simply leave out the translation of the mass center during simulation. One question immediately arises that should we also reject the rigid rotational kinetic energy from temperature calculation ?
We believe that a temperature definition should satisfy the following four conditions in thermodynamic equilibrium :

1. The definition should yield the same value at different sample points.
2. The temperature should be almost independent of sample size when this size is sufficiently large.
3. The temperature should be independent of the choice of reference frame.
4. The temperature definition should be reducible to the classical definition in the simple equilibrium situation.

Based on the above conditions, the applicability of various types of average kinetic energy as measurements of temperature is investigated as follows.

$$T = \frac{2}{3K_B}\langle \frac{1}{N}\sum_{i=1}^{N}\frac{|p_j^2|}{2m_i}\rangle = \frac{2}{3k_B}\langle H\rangle$$

where $\langle ...\rangle$denotes the temporal average, k_B is the Boltzmann constant, m_i is the mass of atom i , and H is the average kinetic energy for each atom. In classical mechanics of discrete systems, the atom velocity can be decomposed into three parts as

$$v_i = v_c + (\omega \times \hat{r^i}) + v_i^{dis}$$

where $\hat{r^i}$ is the atom position vector relative to the center of mass, $v_c$is the centroid transnational velocity, ω is the rotational velocity around the centroid, and ${v_i}^{dis}$ is the disturbance velocity, Correspondingly, there are three possible types of average kinetic energy, i.e.,

$$H^{total} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}m_i.v_i = \frac{1}{2N}[\sum_{i=1}^{N}m_i v^{dis} + \omega J_c.\omega + Mv_c.v_c]$$

$$H^{dis+R} = \frac{1}{2N}[\sum_{i=1}^{N}m_i v^{dis}v^{dis} + \omega J_c\omega]$$

$$H^{dis} = \frac{1}{2N}\sum_{i=1}^{N} m_i {v_i}^{dis} v_i^{dis}$$

where M is the total mass of the system,and $J_c$is the rotational inertia.H^{total}, H^{dis+R},H^{dis} are the averages of the total kinetic energy, kinetic energy without the centroid translation part, and thermal disturbance kinetic energy, respectively. It should be pointed out that H^{dis} is independent of the reference frame, and is the minimum of H^{total} for all possible reference frames; therefore, it may serve as an objective quantity to characterize temperature.
For Kinetic energy we have 3 main part, .H^{total}, H^{dis+R},H^{dis}. Now we investigate which should be considered for the temperature calculation in MD simulation. It is found that (By a simulation,ref[2]) .It is found that the two average kinetic energies excluding the transnational term, H^{dis+R},H^{dis} are uniform over all sample points. However, the average total kinetic energy,H^{total} has significantly different values for different sample points, which indicates that the average total kinetic energy should not be used as the measurement of temperature in some cases. So from condition 1 it is clear that the H^{total} can not be considered in temperature calculation.
To investigate whether Condition 2 is satisfied, we study the effect of sample size on the values of the average kinetic energies H^{dis} and H^{dis+R} using this rotational ball example.

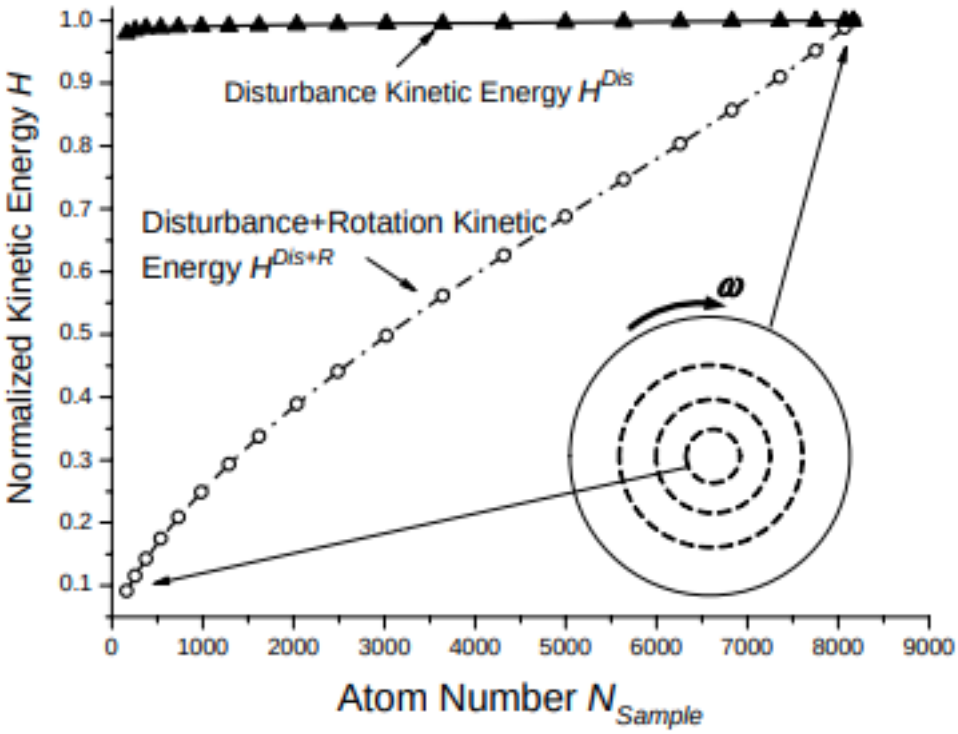

Figure 3: The normalized average kinetic energy at the center as a function of the sample size.

As shown in 3, different-sized concentric sphere samples are tested. The MD simulation results indicate that with an increase in the sample size, the average disturbance kinetic energy,H^{dis}, quickly converges to a constant value, whereas,H^{dis+R} does not converge and continuously increases, hence violating Condition 2. Therefore H^{dis} is an appropriate candidate to characterize temperature.
Here, For our simulation we have used the following expression to calculate temperature ;

$$k_B T = \frac{2}{3} \sum_N E_K$$

In the above formula, E_k is same as H^{dis}. And from equipartition theorem we get the above constant parameter.

0.8 Molecular dynamics program

This molecular dynamics program contains all the instructions that a computer needs to move the particles through time. We will use Python to get the understanding of how molecular dynamics is implemented.

This simulation of molecular dynamics can be done in following steps;

1. Initializing the position of the particles.
2. Then calculating the pairwise forces on the particles, which can be found by taking the gradient of the potential energy: $\mathbf{F} = - \bigtriangledown E(\mathbf{r}) = -\frac{1}{r}\frac{\partial E(\mathbf{r})}{\partial r}$.
3. Computing the new positions by integrating the equation of motion.
4. Scaling of velocity for controlling the temperature.
5. Then recomputing the forces by going back to step 2 and continuing until the maximum number of steps.

0.8.1 Initializing the particles

We use python programming language to fix the initial positions of the particles.The code for this initialisation can be found in the attached file.

0.8.2 Integrating the equations of motion

We will integrate Newton's equations of motion in 1D using the method of velocity verlet integrator.Newton's equation of motion in 1D is given by,

$$\frac{dx}{dt} = v \quad and \quad \frac{dv}{dt} = \frac{dF(x)}{m} \tag{8}$$

The velocity verlet algorithm usually splits the velocity update into two steps.

- The algorithm initially does a half step.
- Then it modifies the acceleration and does the second velocity update.

The steps involved are

1. Calculating $x(t+\Delta t)$ = x(t) + v$\left(t+\frac{1}{2}\right)\Delta t$
2. Then calculating v$\left(t+\frac{1}{2}\right.\Delta$t$= v(t)+\frac{1}{2}a(t)\Delta$t
3. Then deriving a(t+Δt) from the interacting potential using x(t+Δt)
4. Finally calculating v(t+Δt) = v$\left(t+\frac{1}{2}\right)+\frac{1}{2}a(t+\Delta$t)$\Delta$t

The velocities have been rescaled between step 1 and 2 to maintain the temperature at the required value.

The output of all the codes are saved in the same folder as the ipython notebook as traj.xyz and can be accessed by Avogadro or VMD.

0.9 Results

We have taken data about *100* particles from the referred website. When the codes were run for those data we got following results.

- Volume = 100.0
- Density = 0.32
- Temperature = 4.319698725757904e+37 +- 6.047578216060938e+38
- Pressure = 1.3823035922425293e+37 +- 1.9352250291395e+38

The below mentioned plots were obtained by running the code which is in the file bearing the name '*Molecular Dynamics Program.py*'.

Refer figure 4 to view the plot for the positions of the molecules. Refer figure 5 to view the variations of Avg. kinetic energy, Avg.potential energy, temperature and pressure with the number of molecules.

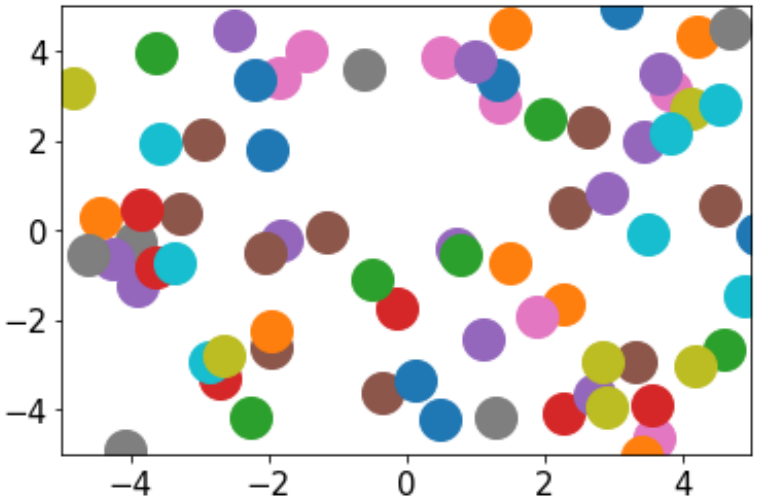

Figure 4: Positions of the molecules

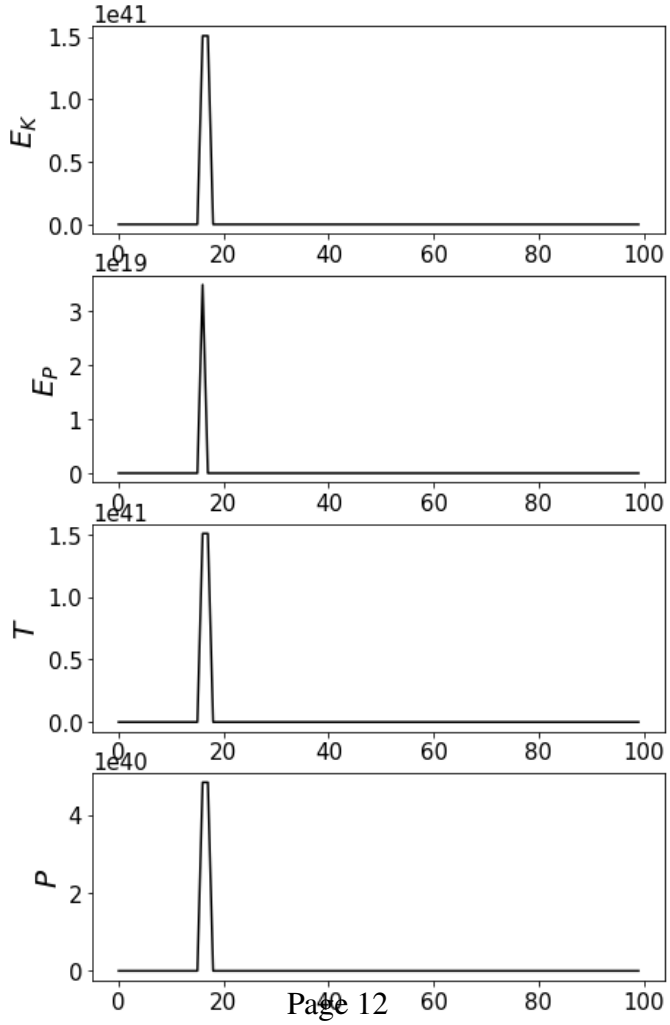

Figure 5: Variations with No. of molecules

0.10 Discussions and Conclusions

Overall we interpreted the molecular dynamics (taking the data for Argon) on Python under the aspects of Classical mechanics.Although the quantum mechanical simulation will give a better insight of the dynamics,we did get good results with Lennard-jonnes potential.We have also included the impact of temperature which brings thermal energy into role play.Thanks to the periodic boundary conditions which made our life easier in simulating the molecules.

For simplicity we considered 1D situation. We did the simulations using multiple libraries such as Numpy, matplotlib etc. We used the velocity verlet algorithm to get the integration for 1D equations of motion.Those codes which we used can be used to get the simulations for some other molecules, and accordingly the plots will also vary.

0.11 References

1. https://nznano.blogspot.com/2017/11/molecular-dynamics-in-python.html
2. https://arxiv.org/pdf/0912.4393
3. https://arxiv.org/pdf/2001.07089.pdf
4. https://nptel.ac.in
5. https://www.cp2k.org/exercises