Monte Carlo: Simulating liquid Argon

Course: FK8028

Author: Andreas Evensen

Submission date: March 8, 2024

Stockholm University
Computational Physics
Sweden

Contents

Introduction

In this report, the behavior of an NVT ensemble when modeled with Metropolis Monte Carlo, composed of Argon atoms, was investigated. Quantities such as: Potential energy, and position was studied. The position of the atoms manifested itself in the form of a radial distribution function $g(r)$, which represented the system's probability of finding an atom at a distance r from another atom. The potential energy of the system was also studied, and the energy of the system was plotted as a function of the number of accepted moves.

Quantities such as the specific heat capacity c_v was computed, and the result was compared to previously obtained results. The simulation implemented periodic boundary conditions, and thus effectively simulated an infinite system, in this case liquid Argon.

Theory & Method

Monte Carlo simulations are a class of algorithms that can be used to sample a molecular system. In contrast to molecular dynamics, Monte Carlo simulations do not require the equations of motion to be solved, and thus are computationally less expensive. This however implies that the Monte-Carlo simulation can not be used to simulate the time evolution of the system but rather the equilibrium state of the system. The Metropolis Monte Carlo algorithm is specific algorithm that implements detailed balance, which implies balance. This manifests in the following equation:

$$N(\mathbf{x}) \cdot \alpha(\mathbf{x} \to \mathbf{x}') \cdot \beta(\mathbf{x} \to \mathbf{x}') = N(\mathbf{x}') \cdot \alpha(\mathbf{x}' \to \mathbf{x}) \cdot \beta(\mathbf{x}' \to \mathbf{x}). \tag{1}$$

In the above equation, $\mathbf{x}$ is the state of the system and $N(\mathbf{x})$ is the population of state $\mathbf{x}$; the probability function α is probability of receiving a change in state from $\mathbf{x} \to \mathbf{x}'$, whilst the probability function β is the probability of accepting such a state. In Metropolis Monte Carlo, $\alpha(\mathbf{x} \to \mathbf{x}') = \alpha(\mathbf{x}' \to \mathbf{x})$, indicating that the probability of picking an atom is the same, as well as moving it in the same direction. This in itself is achieved by uniform randomness.

The Metropolis Monte Carlo algorithm is structured in the following manner:

1. Pick a random atom
2. Propose a move
3. Calculate the change in energy if this move was supposed to occur
4. Accept or reject the move depending on $\min\left(1, \exp\left[-\beta \Delta E\right]\right)$
5. Update the system (if move was accepted, otherwise keep the system as it was)

This process is repeated for many iterations, such that the system reaches equilibrium. In contrast to molecular dynamics simulations, the Metropolis Monte Carlo algorithm is faster, it runs in $\mathcal{O}(N)$ time complexity, where N is the number of atoms in the system, per iteration[1]. Moreover, the Monte Carlo simulation is a stochastic process, and thus the results fluctuate around the true value, even though the system has reached equilibrium.

[1] In the previously implemented Molecular dynamics simulation, the runtime was $\mathcal{O}(N^2)$ per iteration.

The radial function distribution, (RDF), $g(r)$ is measurement of the probability of finding an atom at a distance r from another atom, and is defined as

$$g(r_i) = \frac{V}{N} \frac{h(r_i)}{4\pi r_i^2 \Delta r}. \tag{2}$$

The specific heat capacity c_v is defined as:

$$c_v = \frac{3k_b}{2} - \frac{N\sigma^2(v)}{k_b T^2}, \tag{3}$$

describes how the energy of the system changes with temperature, and is a measure of the system's ability to store energy. Here v is defined as $\frac{\mathcal{V}}{N}$, where $\mathcal{V}$ is the potential energy and N is the number of atoms.

Result & Discussion

The Monte Carlo simulation was written in V, a static typed compiler language. The simulation ran for $0.5 \cdot 10^6$ iterations, where the potential energy and the radial function distribution was computed every 1000 iterations. This was done to ensure that the system had reached equilibrium, and that the system was ergodic. The temperature was set to 94.4 K, as this was the temperature at which the system was previously simulated with a Molecular dynamics simulation. The system was constructed in such a way that the acceptance rate was 28.54%. This is close to the target acceptance rate 30%.

The simulation is written in V, and hence, the following random number generator seed was used:

```
    rng: &rand.PRNG(pcg32.PCG32RNG{
    PRNGBuffer: buffer.PRNGBuffer{
        bytes_left: 0
        buffer: 0
    }
    state: 2811831097841394232
    inc: 5511981685926715381
})
```

Listing 1: Random number generator seed

The radial function distribution $g(r)$ was computed for each for a subset of iterations, and the average of the radial function was plotted in the figure below, fig 1.

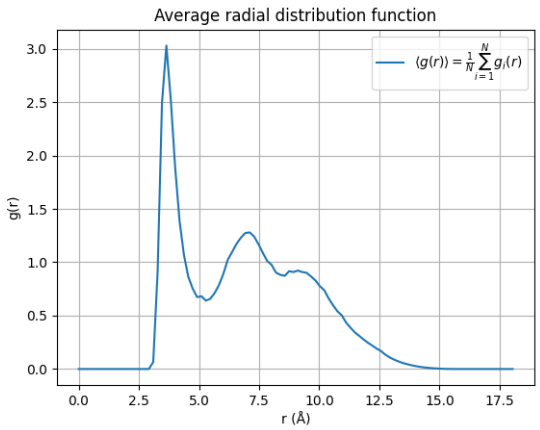

Figure 1: Average radial function $g(r)$

The peaks are located around $r = 3.7$ Å, $r = 7.3$ Å, and $r = 8.7$ Å, which agrees with preciously found results; the previous results were obtained with a Molecular dynamics simulation. Since Monte Carlo simulations are stochastic, the RDF fluctuates around the average value, which is given by the ensemble average, since we enforce ergodicity. Thus, even though this simulation is computed in the NVT ensemble, the RDF still predicts the same equilibrium state as the molecular dynamics simulation, which predicted an NVE ensemble.

The potential energy of the system, modeled by a Lennard-Jones potential, was used to model the energy of the system. If the energy decreased, the move was accepted, and if the energy increased, the move was accepted with a probability $\exp[-\beta\Delta E]$. Here ΔE is the difference in energy between the potential new configuration and the old configuration, and $\beta = \frac{1}{k_b T}$. The average energy of the system is computed in accordance to:

$$\langle \mathcal{V} \rangle = \frac{1}{n} \sum_{i=1}^{n} \mathcal{V}_i,$$

where n is the number of data points. This was computed to be: $\langle \mathcal{V} \rangle = -14.80$ eV, which is in agreement with the previously obtained results. The energy of the system was plotted as a function of the number of accepted moves, and the result is shown in fig 2.

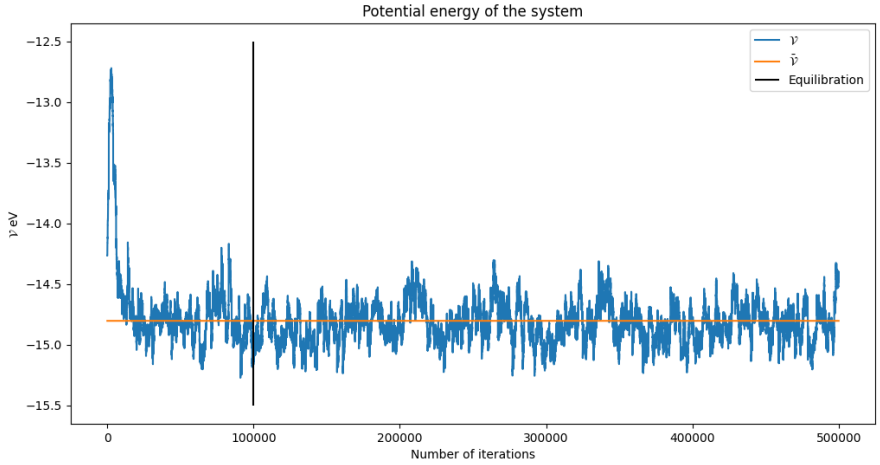

Figure 2: Energy of the system as a function of accepted moves

As the number of accepted moves increases, the potential energy quickly equilibrates, and fluctuates around the average value. To ensure that the system had equilibrated, the block-average method was implemented. This is shown in the figure below:

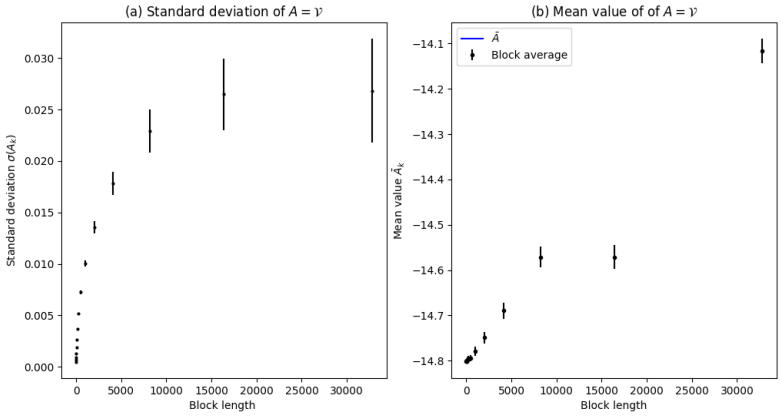

Figure 3: Block average of the potential energy

There is a plateau around $2^{15} = 32768$ block length. This indicates that the system has equilibrated, and that the potential energy indeed fluctuates around the true average of the ensemble. The standard deviation of the potential energy is thus computed to be $\sigma(\mathcal{V}) = 0.23$ eV. The standard deviation of $\sigma(\bar{\mathcal{V}}) = 0.027$ eV, which is signifcantly smaller.

The specific heat-capacity is computed in accordance to eq (3), and was found to be: 37.75 J/(K mol). This result is significantly bigger than the previously obtained result, which was 23.83 J/(K mol). This is due to the random number generator used. Varying the seed of the random number generator, the specific heat capacity fluctuates significantly.

Conclusion

The Metropolis Monte Carlo (MC) algorithm implemented agrees with most of the previous found results, even though the MC algorithm modeled a different ensemble (NVT) instead of an NVE ensemble. The radial function distribution, eq (2), manifests the same distance between atoms in our MC algorithm as previously found in the molecular dynamics simulation, and the average energy $\langle \mathcal{V} \rangle$ is also in agreement to previous results. The standard deviation $\sigma(\bar{\mathcal{V}})$ is comparable with the previously found result, but differs with order 10. This is not unexpected since MC is a stochastic process.

The specific heat capacity c_v differed significantly compared to our molecular dynamics simulation. As this is a result of the random number generator used, an improvement for future work would be to implement another generator which can provide suitable numbers.