Homework3

SHI Yuting

21071210

yshiby@connect.ust.hk

January 5, 2025

1 Monte Carlo (MC) simulations of Lennard-Jones argon

Use the Metropolis Monte Carlo method and the Hamiltonian Monte Carlo (also known as hybrid Monte Carlo) method to calculate the radial distribution function (RDF) in the homework 2.

A summary of the Metropolis Monte Carlo method is listed:

1. Random starting states.
2. Random move: $x_0 \to x_0 + \Delta(Rand(0,1) - 0.5)$
3. $acc(0 \to n) = min(1, e^{-\beta(E_n - E_0)})$
4. Measure.
5. Iteration.

The E_n in this method is $E_n = V$.

And there is the summary of the Hamiltonian Monte Carlo method:

1. Random starting states.
2. Random $\vec{v_i}$ $N(0, k_B T)$, then remove the velocity of the mass centrual.
3. Velocity Verlet: $t = 0 \to t = h$, a MD progress.
4. $acc(0 \to n) = min(1, e^{-\beta(E_n - E_0)})$
5. Iteration.

The E_n in this method is $E_n = T + V$.

1.1

Based on this list of method, we can write the algorithm based on the MD code in the homework2. And the code is attached. Noted that I only write the algorithm about the Monte Carlo and the Calculation part, there are still some functions like initializing the position and velocity or read the data from files. All of theese parts are showed in the code. I only write the important thing.

Algorithm 1 Monte Carlo Methods and RDF Calculation

FUNCTION simulate_MetroMC(file)
INITIALIZE positions and velocities
for each iteration **do**
  GENERATE random perturbations
  CALCULATE acceptance probability:

$$\text{acc\_prob} = \min\left(1, e^{-\beta(\text{prop\_pot}-\text{pot})}\right)$$

  if random_number < acc_prob **then**
    ACCEPT proposed positions
  else
    REJECT proposed positions
  end if
  RECORD energies and write to file
end for
END FUNCTION
FUNCTION simulate_Hamilton(file)
INITIALIZE positions and velocities
for each iteration **do**
  UPDATE positions and velocities using Hamiltonian dynamics
  CALCULATE acceptance probability:

$$\text{acc\_prob} = \min\left(1, e^{-\beta(H(\text{new})-H(\text{old}))}\right)$$

  if acc_prob==1 **then**
    ACCEPT new positions and velocities
  else
    REJECT new positions and velocities
  end if
  RECORD energies and write to file
end for
END FUNCTION
FUNCTION calculate_rdf(positions, n_bins, L_box)
INITIALIZE histogram for RDF
DEFINE bins for RDF
for each frame of positions **do**
  for each pair of particles **do**
    CALCULATE distance
    UPDATE histogram based on distance
  end for
end for
NORMALIZE histogram to get RDF
return distances and RDF values
END FUNCTION
FUNCTION main()
DEFINE output files for MetroMC and Hamiltonian simulations
CALL simulate_MetroMC(file_metro)
CALL simulate_Hamilton(file_hami)
READ results from output files
CALL calculate_rdf(positions)
PLOT results
END FUNCTION

1.2

In the Metropolis Monte Carlo simulation. The trial displacement is very important. This displacement cannot be too large to cause the acceptance rate to be too low. This results in excessively high self-correlation in the simulated samples, making it difficult to search for some details. This displacement cannot be too small, causing the energy to be trapped in a local extreme value and unable to traverse the entire energy landscape.

In this homework, we choose the reduced unit, I focus on the displacement based on the time step:

$$\delta = \sqrt{k_B T_0} dt \tag{1}$$

in which the dt is the time step in the Hamiltonian Monte Carlo simulation. So I will discuss the time step of Hamiltonian Monte Carlo first.

The time step should not be too large because we need to do the Velocity Verlet method to update the velocity and position. We need to avoid the energy drift. But the time step also should not be too small, if the time step is too small, it will lead to many useless attempts and increase the computational cost.

Some research claimed that a reasonable acceptance rate is between [0.15, 0.50]. So I change the dt to make the acceptance rate of the Hamiltonian Monte Carlo is in this interval. Finally I set the number of iterations is 16000 and $dt = 120/16000$(We need a large number of iterations to traverse all states). In this situation the acceptance rate=0.49 and the energy is monitored that there is no significant energy drift.

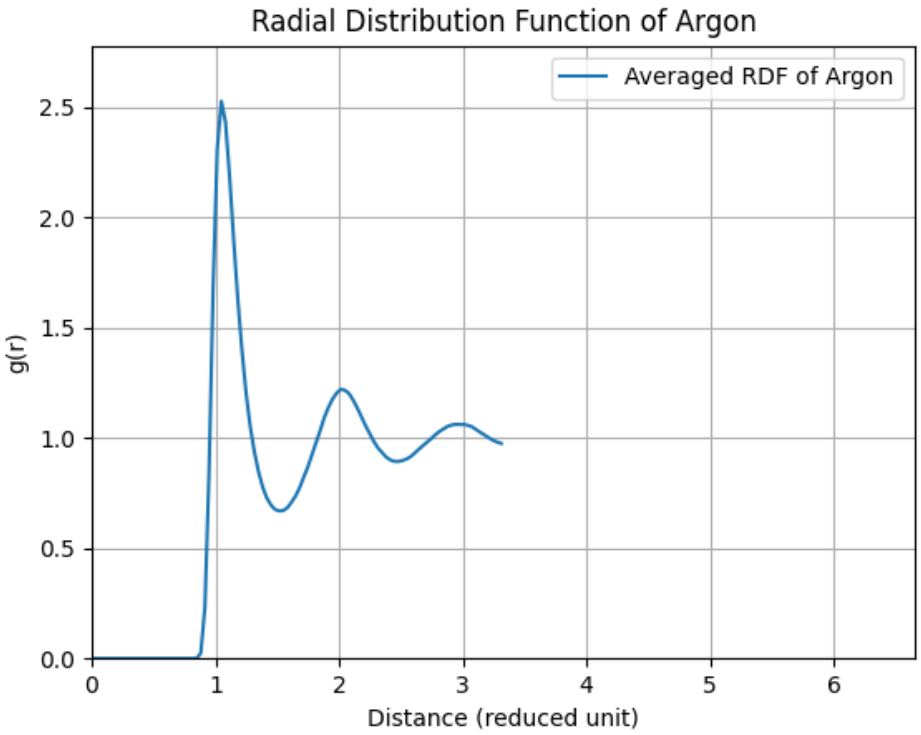

Figure 1: RDF from Hamiltonian Monte Carlo Simulation

After that I use it and 1 as a basis to change the value of Δ. I calculated the acceptance rate, and I changed the Δ to control the rate is in [0.15,0.50]. Finally I set $\Delta = 1.5\sqrt{k_B T_0} dt$ and the acceptance rate will be about 0.39.

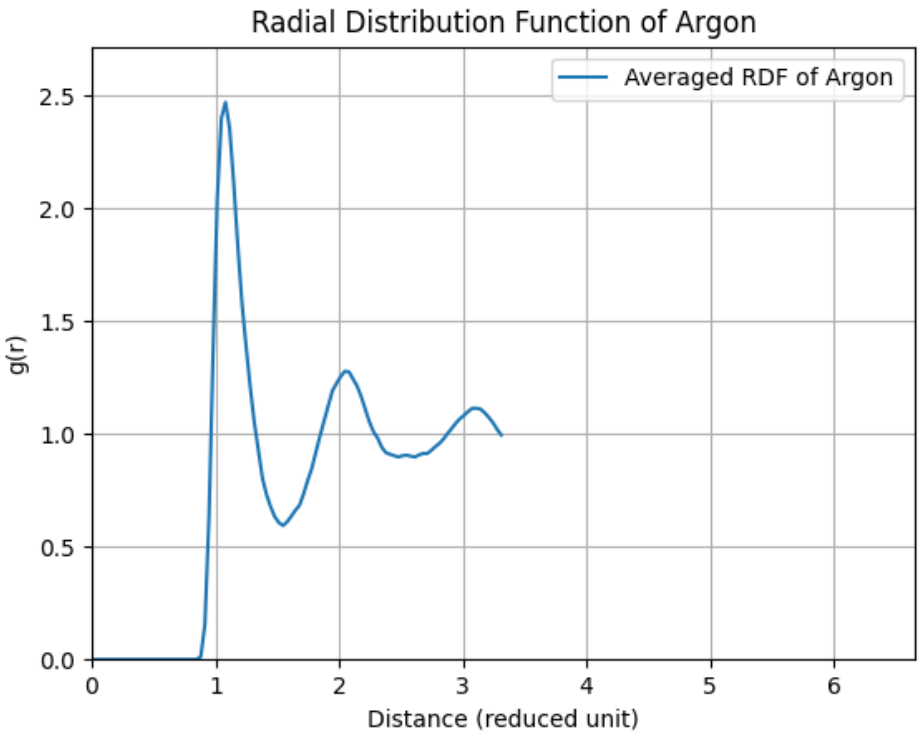

Figure 2: RDF from Metropolis Monte Carlo Simulation

1.3

The RDF which is from MD is showed below:

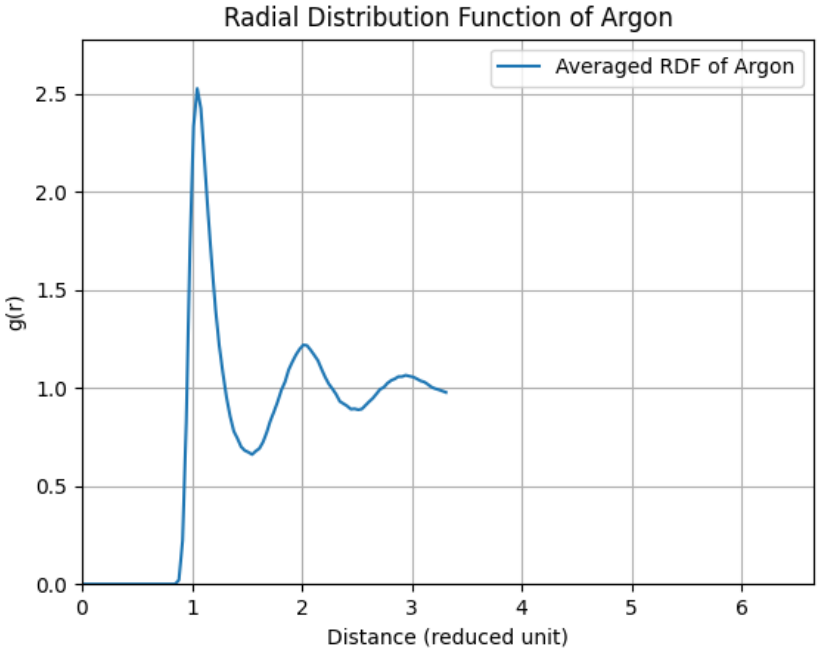

Figure 3: RDF from Metropolis MD(4000 iterations and 6 total time)

You can find that even there are some noises(Maybe we need more iterations) but the shape of the 3 results are consistent with each other.

From my experience, Metropolis Monte Carlo only needs to calculate the energy, so it may be more convenient, but it needs a lot of iterations to go through all the states, otherwise there will be some noise in the results. Convergence depends on the choice of trial shifts and the acceptance rate. Hamiltonian Monte Carlo requires fewer iterations, but the same number of iterations will take longer than Metropolis Monte Carlo because it needs the velocity Verlet method. But since there are fewer steps to achieve convergence, the comparison of the cost time required will depend on the actual problem.

MD simulation is like Hamiltonian Monte Carlo with acceptance rate = 1. So Hamiltonian must be more efficient than MD simulation.

I prefer the Hamiltonian Monte Carlo for this question, but we need to choose the right time step. And there must be some other situation like equilibrium sampling in lower-dimensional spaces, for which Metropolis MC is great.

1.4

It is used for posterior sampling of high-dimensional neural networks, such as Bayesian neural networks. It can help the network reach convergence faster.