PHY407 Lab 11: Monte Carlo Simulation

Emma Jarvis: Q1, Q4
Lisa Nasu-Yu: Q2, Q3

December 3, 2021

1 Monte Carlo Simulations

1.a

1.a.1

We ran the program that uses Monte Carlo Simulation of an ideal gas to siulate a non-interacting quantum ideal gas in a box. We ran this code for $k_BT = 10$ and plotted the energy as a function of time. Figure 1 depict this energy as a function of time (number of steps) and Figure 2 shows the energy distribtion as a function of n.

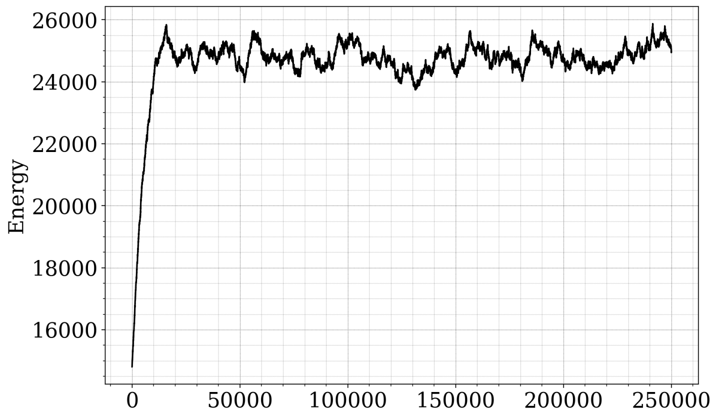


Figure 1: Energy as a function of time fot $k_B = 10$.

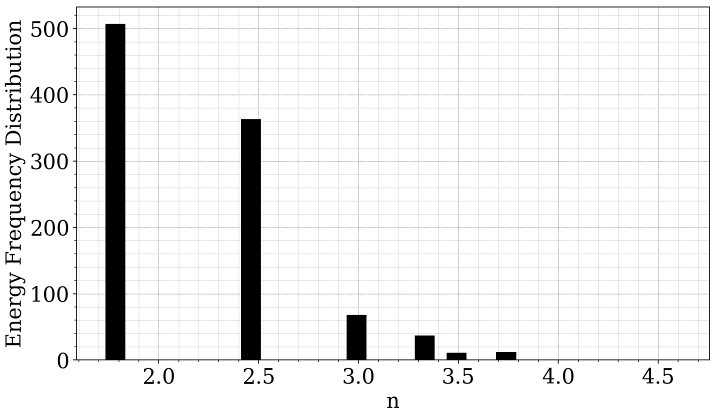


Figure 2: Energy frequency distribution for $k_B = 10$.

1.a.2

We then used this code to calculate the average value of n using 1 where $f(e_n)$ is the energy frequency distribution.

$$\overline{n} = \frac{\sum_n f(e_n) n}{\sum_n f(e_n)} \tag{1}$$

We ran this code for different values of $k_B T$ and each value required a different number of steps to come to equilibrium. To check whether the system came to equilibrium, we plotted the energy as a function of the number of steps and increased the step number until it was clear that the enrgy leveled out. Table 1 summarizes the values of $k_B T$ that were used and how many steps it took for the system to come to equilibrium.

$k_B T$	Number of Steps
10	$2.5 \cdot 10^5$
40	$2.5 \cdot 10^5$
100	$2.5 \cdot 10^5$
400	$5 \cdot 10^5$
1200	$2 \cdot 10^6$
1600	$2.5 \cdot 10^6$

Table 1: Number of steps it takes the system to come to equlibrium for different values of $k_B T$.

We then plotted the energy as a function of $k_B T$ and $\overline{n}$ as a function of $k_B T$ and figures 3 and 4 depict these figures. Both the total energy and average quantum number increase with temperature.

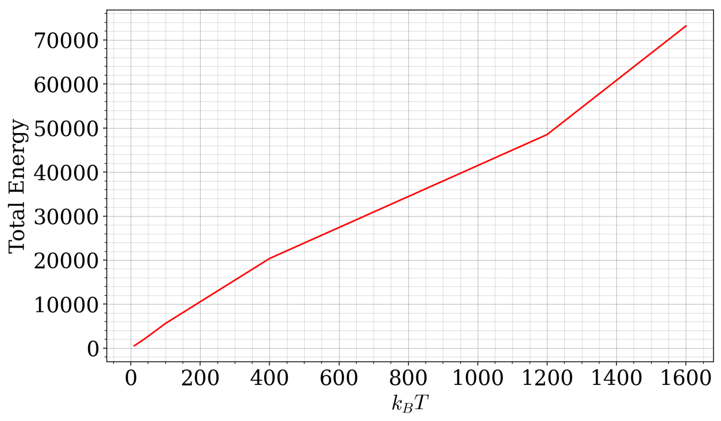


Figure 3: Total energy of the system as a function of k_BT.

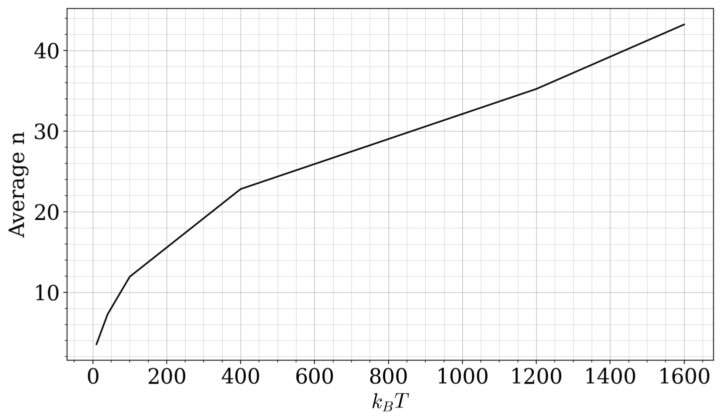


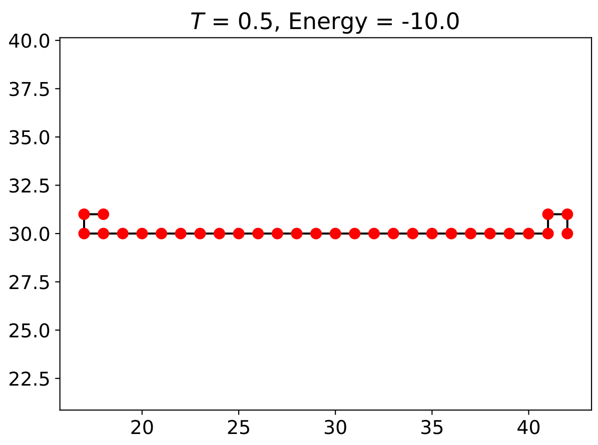
Figure 4: Average value of n (quantum number) for the system as a function of k_BT

Finally, we estimated the heat capcity of the system as $\Delta E/\Delta T$ over this range where ΔE is the change in energy between the maxiumum and minimum energy and ΔT is 1600-10 (the difference between the minimum and maximum T) and this was calculated to be 45.6761 J/K.

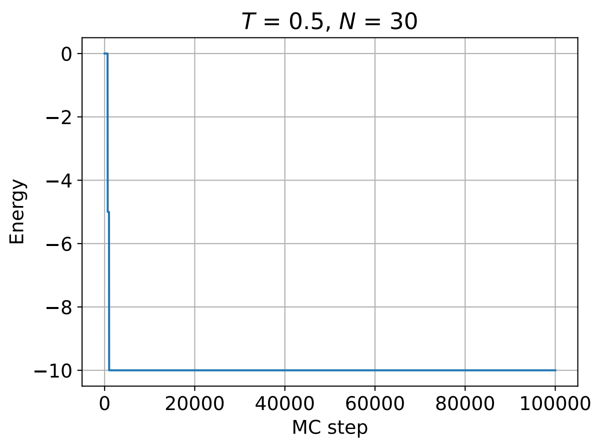
2 Protein Folding

2.a

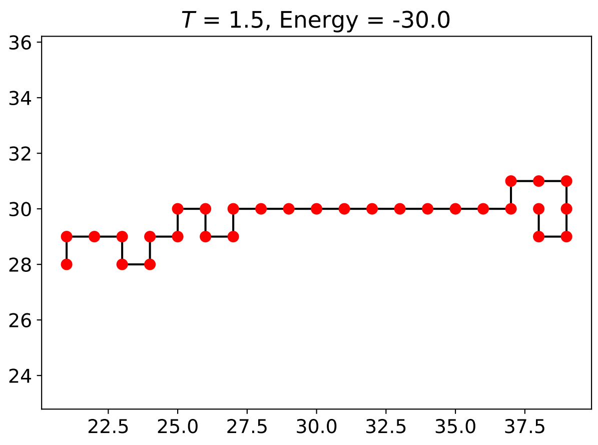
We ran the script L11-protein-start.py for N=30, $n = 10^5$, $\epsilon = -5$, and various temperatures T. Only parameters were changed in the script for both parts a and b.



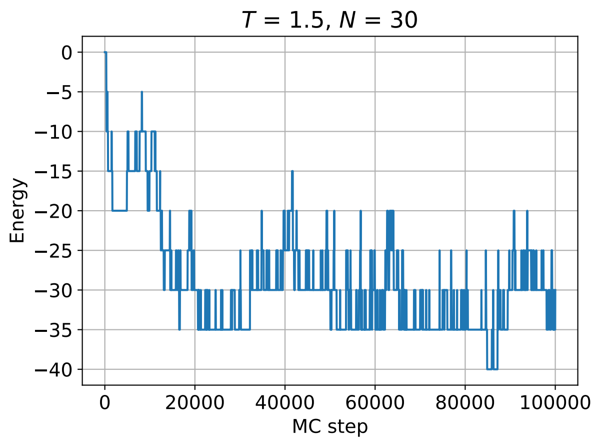
(a) Final Structure, T=0.5



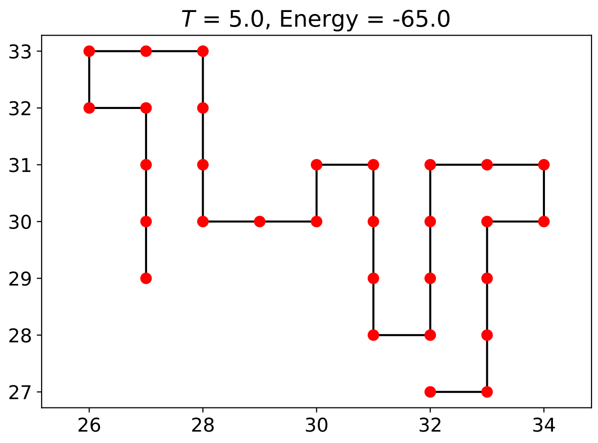
(b) Energy vs MC Step, T=0.5



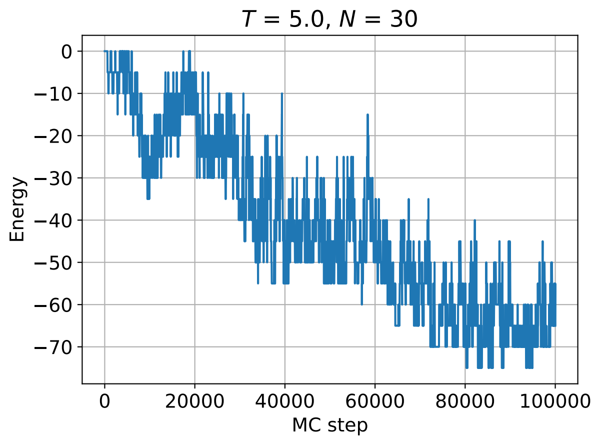
(c) Final Structure, T=1.5



(d) Energy vs MC Step, T=1.5



(e) Final Structure, T=5



(f) Energy vs MC Step, T=5

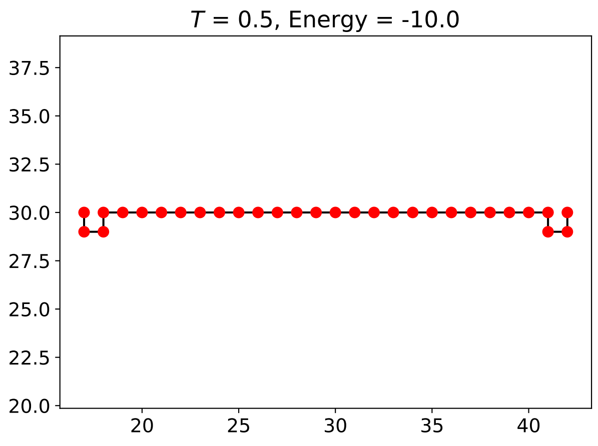
Figure 5: Final protein structure and energy vs step for N=100000 steps and varying temperatures T.

For T=1.5 (Fig 5c, 5d), the final structure of the protein has a moderate amount of folding at the ends of the chain. Overall the energy decreases with steps taken and stays at approximately -30 after about 20000 steps.

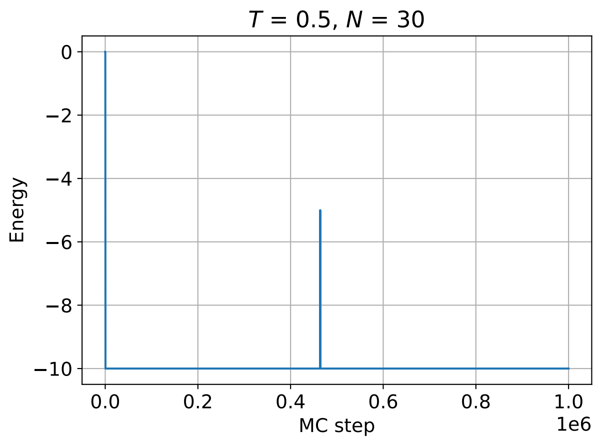
At the lower temperature of T=0.5 (Fig 5a, 5b), the final structure has less folds and the energy drops to its typical value of approximately -10 much faster.

At a higher temperature of T=5.0, the protein has large folds throughout the entire chain, and the energy takes much longer (around 80000 steps) to reach its typical state of approximately -65.

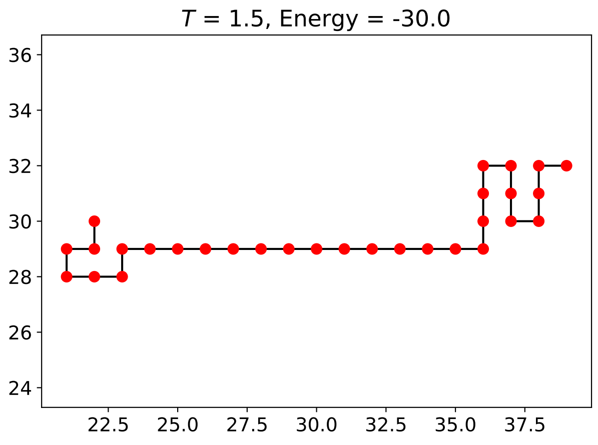
2.b



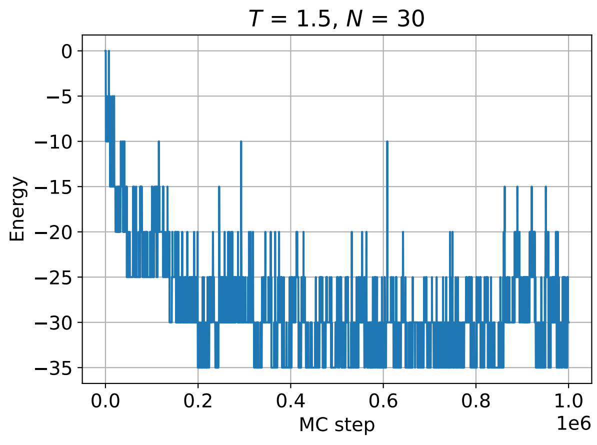
(a) Final Structure, T=0.5



(b) Energy vs MC Step, T=0.5



(c) Final Structure, T=1.5



(d) Energy vs MC Step, T=1.5

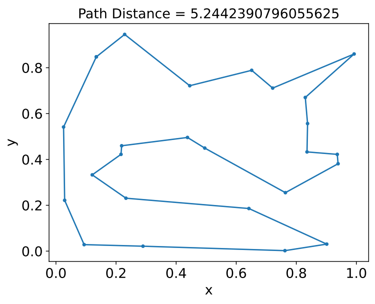
Figure 6: Final protein structure and energy vs step for N=1000000 steps and varying temperatures T

By visual inspection of Fig 6b and 6d, the typical energy over the second half of the simulation is lower for T=1.5 (E≈-30) than for T=0.5 (E≈-10). Yes, this is as expected because a higher initial kinetic energy allows more movement through protein folding, thus raising the chances of the protein finding itself in a lower energy minimum. For T=0.5, the protein has less kinetic energy and cannot fold as much as the higher temperature case. It only folded enough to find itself in a local minimum with an energy of -10.

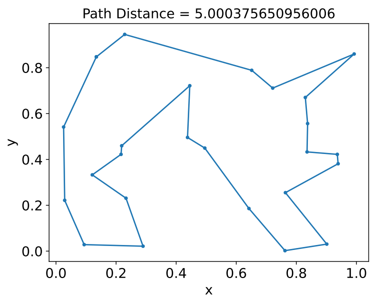
3 Examples of Simulated Annealing Optimization

3.a

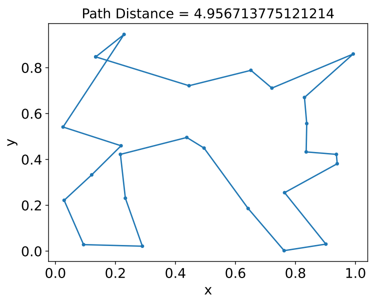
We ran the file salesman.py for various seeds and τ to determine an optimal path between N points.



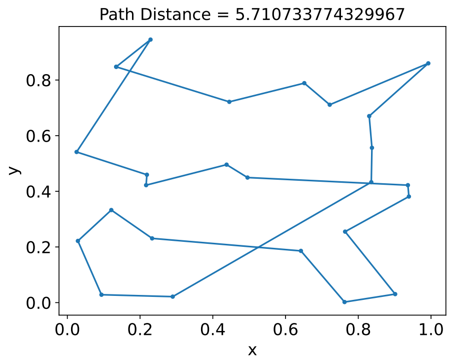
(a) Seed=1, $\tau = 10^4$



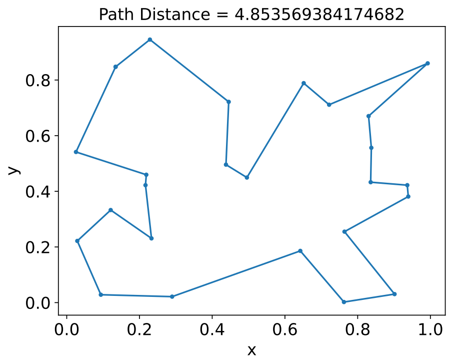
(b) Seed=2, $\tau = 10^4$



(c) Seed=3, $\tau = 10^4$



(d) Seed=4, $\tau = 10^3$



(e) Seed=5, $\tau = 10^5$

Figure 7: Optimal paths between 25 points computed with various seeds and τ.

We first ran the script with the same value for τ but different values for the seed, shown in Figs 7a, 7b, 7c. The optimal path distance varied within a range of approximately 0.29. Of the 3 seeds we tested, the smallest distance was with a seed of 3, at approximately 4.957, and the highest distance was with a seed of 1, at approximately 5.244.

Next, we continued to vary the seed, but also changed the value of τ, shown in Figs 7d, 7e. At a lower τ of 10^3 (previously $\tau = 10^4$, the path was significantly higher, at approximately 0.47 higher than the previously tested highest value (seed 1). For a higher τ of 10^5, the path was approximately 0.1 lower than the previously tested lowest value (seed 3). We conclude that slower cooling decreases the path distance and faster cooling increases the distance.

3.b

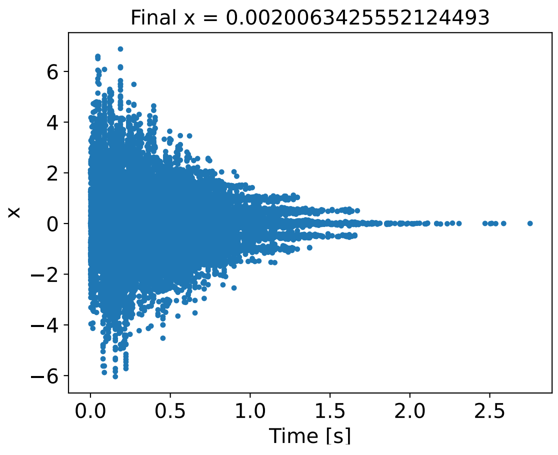
We ran simulated annealing optimization for 2 example functions for parts i) and ii), respectively:

$$f(x, y) = x^2 - cos(4\pi x) + (y - 1)^2 \quad (2)$$

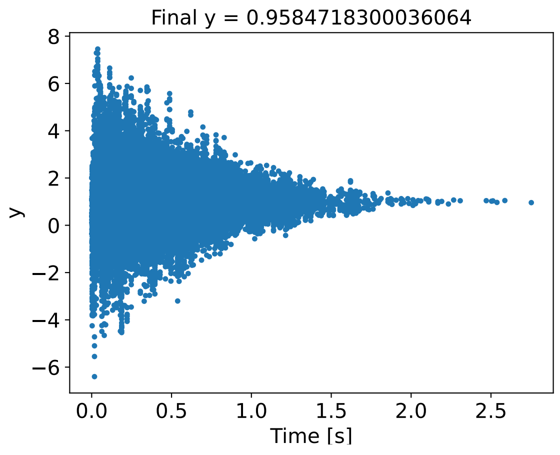
$$f(x,y) = cosx + cos(\sqrt{2}x) + cos(\sqrt{3}x) + (y-1)^2 \tag{3}$$

We used the values $\tau = 10^4$, $T_{min} = 10^{-3}$, and $T_{max} = 10.0$ for both parts.

3.b.1



(a) x vs time

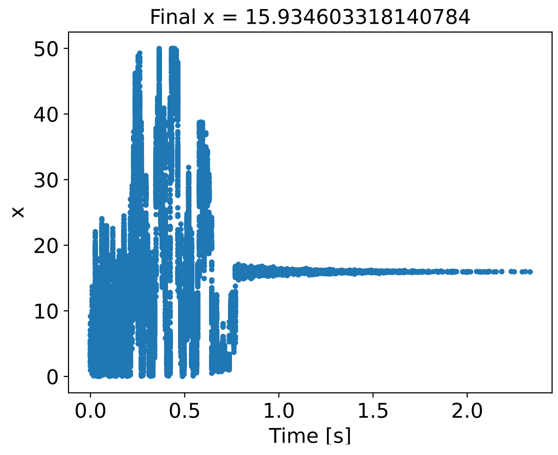


(b) y vs time

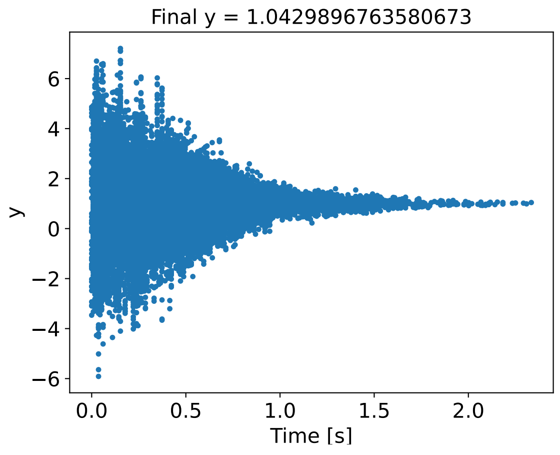
Figure 8: In approximately 3 seconds, we obtained a final position approximately equal to the expected global minimum of (0,1).

3.b.2

We search for minimums to Eq. 3.b within the range $0 < x < 50$ and $-20 < y < 20$.



(a)



(b)

Figure 9: In approximately 2.5 seconds, we obtained a final position approximately equal to the expected global minimum of (16,1).

4 Ising Model

4.a

We first wrote a program that calculates the total energy of the Ising model on a square lattice for a system of 20 by 20 spins using 4.

$$E = -J \sum_{i,j} s_i s_j \tag{4}$$

4.b

We then wrote a program that uses this energy calculation as the basis for a Metropolis-style simulation of the Ising model with $J = 1, T = 1$ and $k_B = 1$. We set all of the spins in the 20 by 20 grids randomly so that about half of them are spin up and the other half are spin down. This was done using `random.random`. We then picked one of these dipoles at random and flipped it and calculated the new energy. To decide whether to accept this flip, we used the Metropolis formula ($P_a = 1$ if $E_j \leq E_i$ and $P_a = e^{-(E_j - E_i)/kT}$ if $E_j > E_i$ where E_j is the new energy and E_i is the old energy before the random dipole was flipped) and if the flip was rejected, we flipped the dipole back to its original spin and added the old energy not the new energy.

4.c

We then made a plot of the total magnetization (which is the sum of all the dipoles in the grid) for 1 million Monte Carlo steps. Figures 10 and 11 depict the total magnetization and energy respectively. The system develops an overall magnetization of 400 which indicates that all dipoles in the 20 by 20 grid have a +1 spin.

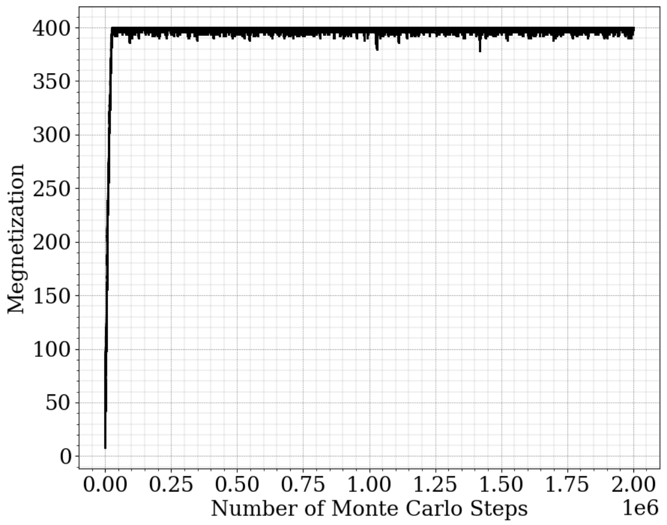


Figure 10: Total magnetization as a function of the step number. Magnetization levels off at 400 after 20 000 steps.

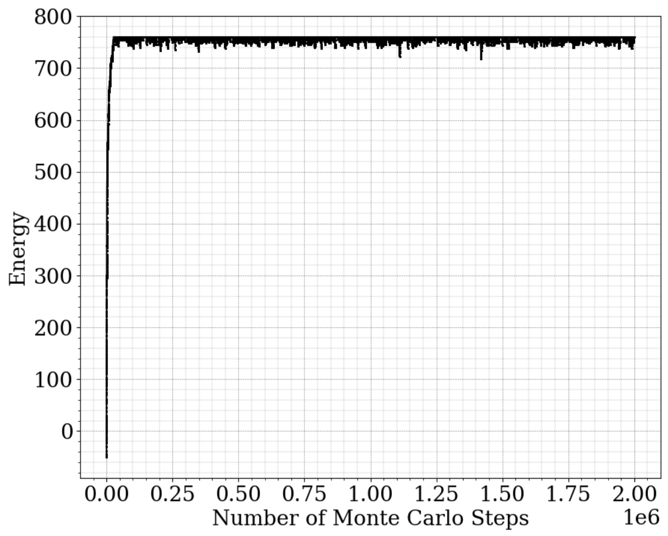


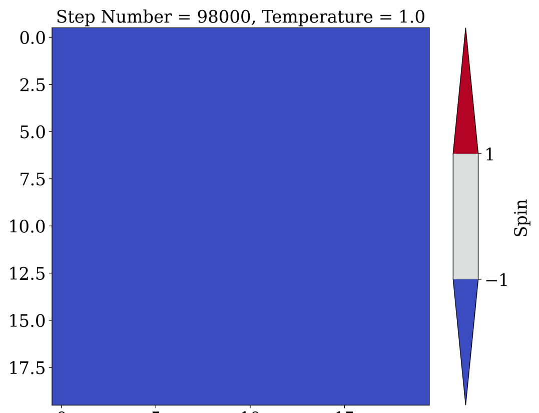
Figure 11: Total energy as a function of step number.

4.d

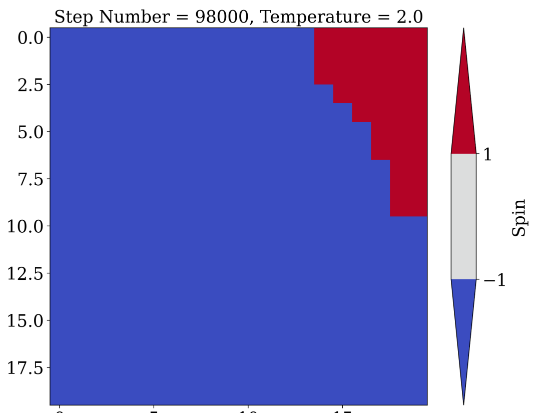
We then ran the simulation a few times. Every time the whole system develops an overall magnetization of either +400 or -400. This is because eventually over time all of the dipoles will line up to be in the same direction, depending on what the direction is of the surrounding dipoles. So if the surrounding dipoles are positive then all of the dipoles will eventually end up with spin up but if most of the dipoles end up with a spin down then all of the dipoles will end up with this spin as well.

4.e

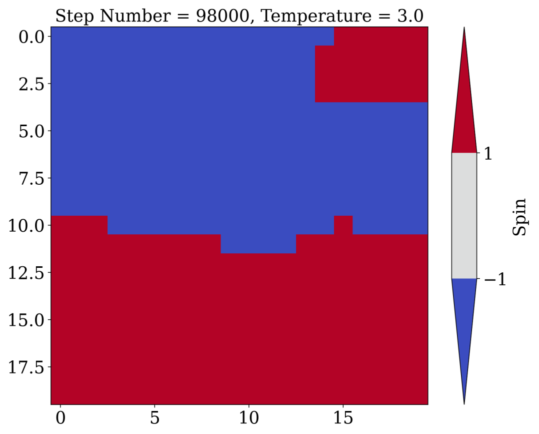
We then made an animation to show how the system of dipoles changes over time. We ran his animation for temperature of 1, 2 and 3 and the results at the end of each simulation are depicted in Figures 12a, 12b and 12c respectively. This animation shows that as the temperature increases, it takes longer for the system to obtain all the dipoles in the same direction. This is because an increase in temperature causes more misalignment of the dipoles.



(a)



(b)



(c)

Figure 12: System showing the spin of the dipoles after 98 000 steps for temperatures of 1, 2 and 3 where blue squares represent spin down (-1) and blue squares represent spin up (+1).