# 2-D Ising model in the absence of External Magnetic Field using Monte Carlo Simulations

*submitted by*

**Abhinna Sundar**

**MS17204**

**4th year BS-MS student**

Under the supervision of

**Dr. Rajeev Kapri**

Associate Professor


**Department of Physical Sciences,**
**Indian Institute of Science Education and Research,**
**Mohali**

# Acknowledgements

I would like to express my sincere gratitude to **Dr. Rajeev Kapri** for the continuous support of the research, his guidance, immense knowledge and the beneficial discussions during the class.


# Contents

# 1 Introduction

It may be extremely difficult to find analytical solutions, sometimes even impossible, for many physical systems. We can take the help of numerical simulations to study many such systems. In the method of numerical simulations we start with an initial state and let the rules (usually some mathematical model) of the system work on it to generate further states. Computers are used to carry out this process. Then we can measure the observables on these states and understand the behavior of the system. Sometimes, directly simulating systems in this way can take thousands of years for even the best computer available to complete. In such cases, we resort to algorithms made for fast and efficient numerical simulations.

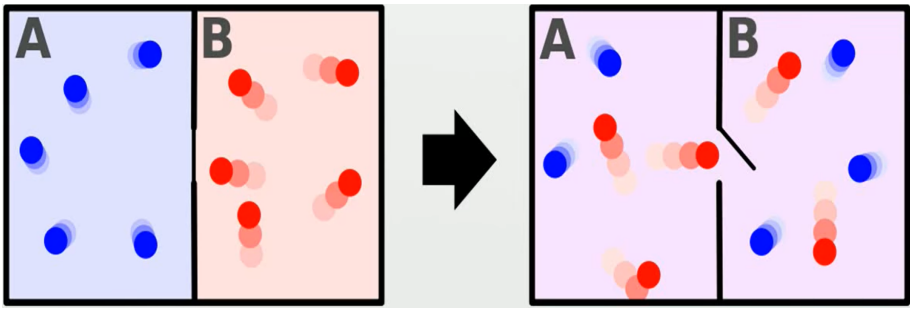



Fig: Illustration of a system becoming more disordered as rapid and slow particles mix together

In this project, I simulate the 2D Ising model using the Metropolis algorithm and measure the observables such as energy $< E >$, magnetization $< M >$, specific heat $C_V$ , and susceptibility $\chi$ for various values of the temperature $T$. I will also study the 2nd order ferromagnetic phase transition that occurs in the 2D Ising model and find the critical temperature $T_c$. All the simulation codes are written in Python (attached with this file in Moodle).

# 2 Marcov Chain Monte Carlo

In this section, I briefly discuss the method and algorithms used for the numerical simulations in this project.

## 2.1 Marcov Chains

Markov chain is a mathematical model of random process, which describes a sequence of states in which probability of attaining each state depends only on the immediate previous state. It was introduced by the Russian mathematician Andrey Markov. In this process, we

can make predictions regarding the future outcomes solely based on its present state without needing to know the full history of the process *(memorylessness property)*. The change from one state of the system to another is called the transition and the probability associated with it is called the transition probability.

## 2.2 Marcov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) is a method of random sampling.

The goal in this method is to visit a point x with probability proportional to some given distribution function, say $\pi(x)$.

It is not necessary for $\pi(x)$ to be a normalized probability distribution function and it is sufficient for it to be proportional to some probability. This method gives greater *importance* to points for which $\pi(x)$ is large. This results in points of less importance (or contribution) being less visited and makes the method much more efficient compared to uniformly visiting all possible points in the space. We first start with an initial distribution and then jump from one point to the next with the help of the transition probability function. Since it is a Markov chain, the probability of visiting a point $x_i$ depends only on the immediate preceding point $x_{i-1}$. The transition probability function is completely defined as a function of two variables $x_i$ and $x_{i-1}$ as $p(x_i|x_{i-1})$, which is the conditional probability of $x_i$ given $x_{i-1}$. This dependence results in a sequence of these points being locally correlated and because of this, later when we take measurements on a system simulated using this method, we need to take measurements after some gap to get independent measurements. Joseph (2020)

A Markov chain with stationary distribution has the property of irreducibility and aperiodicity. According to the property of aperiodicity, a state does not repeat after a given time period. According to the property of irreducibility, the probability to go from any state to every other state, in one or more steps, is greater than zero. This ensures that we explore as much of the space as possible and naturally visit closer to the points $x_i$ with greater probability $\pi(x_{i-1})$ in some finite steps. In a physical system, such points with greater probability would be close to the equilibrium states of the system.

## 2.3 Pseudo Random Number Generators

The random numbers used in Monte Carlo simulations are usually generated using a deterministic algorithm. They exhibit sufficiently random-like properties and are called pseudo-random numbers.

In the simulations of this project, the python library *numpy.random.random()* is used, which is a pseudo-random number generator. It gives us a pseudo-random number uniformly distributed in the range [0,1).

## 2.4 The Metropolis Algorithm

Metropolis-Hastings algorithm is an algorithm for Markov Chain Monte Carlo (MCMC) which is used to get random samples from a probability distribution $\Pi(x)$ given that we have a probability distribution $\pi(x)$, which is proportional to $\Pi(x)$ and the values of $\pi(x)$ can be calculated. We use this method when it is difficult to calculate the values of $\Pi(x)$ itself. Wikipedia (2021b)

Metropolis algorithm is a special case of the Metropolis-Hastings algorithm with the condition that the transitional probability function is symmetric, that is $p(x|y) = p(y|x)$ where $p(x|y)$ is probability of the next sample $x$ given the previous value $y$. Since the probability of the next sample depends only on the current sample, this algorithm forms a Markov chain. The Metropolis algorithm proceeds in the following way:

- **Initialization**: Initial point $x^{'}$ is arbitrarily chosen as a first sample.
- **Generation**: A candidate $x^{'}$ is generated for the next sample according to the transition probability $p(x^{'}|x_i)$.
- **Accept/Reject**: Metropolis ratio $r$ is calculated. The candidate $x^{'}$ is accepted for the next sample with probability $r$. Otherwise, it is rejected.

  $$r = \frac{\pi(x^{'})}{\pi(x)}$$

  Since $\pi(x)$ is proportional to $\Pi(x)$, the ratio $r$ is

  $$r = \frac{\pi(x^{'})}{\pi(x)} = \frac{\Pi(x^{'})}{\Pi(x)}$$

- Steps **2** to **3** are repeated to accept new samples and continue the chain. After sufficient number of iterations, we will move closer to the target distribution $\Pi(x)$. The initial part of the chain before getting closer to the target distribution is called the thermalization steps (or burn-in). These initial samples are not close to the target distribution and are discarded from the result.

# 3 The Ising Model

The Ising model is a simple system of arrangements of spins in an n-dimensional lattice. The spins have two possible states: up or down. This model can be used to study the ferromagnetic behavior of certain systems. The neighboring spins interact with each other and contribute to the energy term of the system. Wikipedia (2021a) The Hamiltonian of the model is given by

$$\mathcal{H} = -J \sum_{<i,j>} \sigma_i \sigma_j \tag{1}$$

where $\sigma_i$ and $\sigma_j$ denote the spins. Mathematically, we assign up as $+1$ and down as $-1$. $< i, j >$ in the summation denotes nearest neighbors. $J$ is called the coupling constant and it affects the strength of the interaction among spins. The form of the Hamiltonian in Eq. 1 suggests that parallel spins lower the energy and thus are favored. This leads us to predict that the lowest energy state will have all the spins aligned in one direction. However, we cannot predict whether the direction will be up or down, and that is the symmetry inherent in the system. We can have an external magnetic field ($H$) such that the spins experience an upward magnetic field. Then, we have

$$\mathcal{H} = -J \sum_{<i,j>} \sigma_i \sigma_j - \mathcal{H} \sum_{i}^{N} \sigma_i \tag{2}$$

In this case, at the lowest energy state, even with the slightest external magnetic field $H$, will bias the spins to align upwards ($H > 0$) and we will have all spins pointing up. We will consider no external magnetic field to be present in this model for this assignment. We would like to study the behavior of this system for various temperature T. One way is the statistical mechanics approach of the canonical ensemble.

## 3.1 The Canonical Ensemble

We consider the physical system to be in contact with a large heat bath (the environment) whose temperature remains constant. When the system comes in thermal equilibrium with the environment, it can be described by the canonical ensemble Met (2017). Then the probability to find a system in microstate $\nu$ with energy $E_\nu$ at temperature $T$ is given by

$$P(\nu) = \frac{1}{Z} e^{-\beta E_\nu} \tag{3}$$

where $\beta = \frac{1}{kT}$. $Z$ in eq. 3 is called the partition function and it is defined as $Z = \sum_\nu e^{-\beta E_\nu}$ where the summation is over all microstates $\nu$ of the system, assuming all the microstates have discrete energies $E_\nu$. The Helmholtz free energy, $F$ of the system is then given by

$$P(\nu) = \frac{1}{\beta} ln(Z) \tag{4}$$

All the other thermodynamic quantities of the system can be calculated once we know $F$.

## 3.2 2D Ising Model

We consider an $(n \times n)$ arrangement of spins in a 2-dimensional square lattice. Each spin has 4 nearest neighbors. We consider periodic boundary conditions such that the spins on the $n^{th}$ row interact with the spins on the $1^{st}$ row and the spins on the $n^{th}$ column interact with the spins on the $1^{st}$ column.

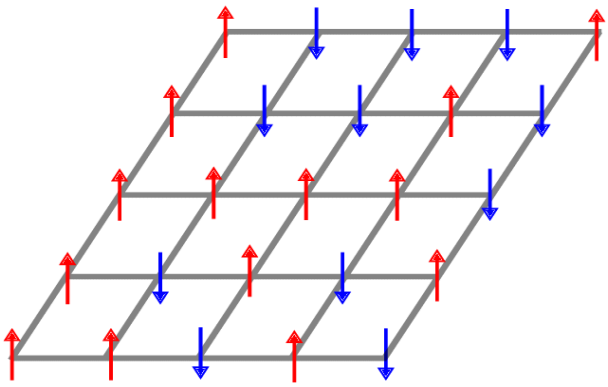


Fig: 2D Ising Model schematic

The Hamiltonian of this system is given by

$$\mathcal{H} = -J \sum_i^n \sum_j^n \sigma_{(i,j)}\sigma_{(i+1,j)} - J^* \sum_i^n \sum_j^n \sigma_{(i,j)}\sigma_{(i,j+1)} \tag{5}$$

where $\sigma_{(i,j)}$ represents the spin at $i^{th}$ row and $j^{th}$ column. $J$ is the horizontal coupling and $J^*$ is the vertical coupling. We will study the isotropic case in which $J = J^*$.

The partition function of this system is given by

$$Z = \sum_{\sigma_1} \sum_{\sigma_2} \cdots \sum_{\sigma_n} e^{-\beta \mathcal{H}} \tag{6}$$

The partition function $Z$ in this case is extremely difficult and time consuming to calculate. We will instead use the method of Metropolis algorithm to simulate the 2D Ising model.

### 3.3 The Algorithm

A system in thermal equilibrium with a temperature bath. Probability $p_\mu$ of being in state $\mu$ with energy $E_\mu$ is

$$p_\mu = \frac{1}{Z} e^{-\beta E_\mu} \tag{7}$$

where $Z$ is $\sum_\mu e^{-\beta E_\mu}$ is the **partition function**. At equilibrium, the following must be true.

$$\sum_\nu p_\mu P(\mu \to \nu) = \sum_\nu p_\nu P(\nu \to \mu) \tag{8}$$

where $P(\mu \to \nu)$ is the probability of going from state $\mu$ to $\nu$. For our numerical methods, this is difficult to enforce, but we can make it be true be setting the **detailed balance** condition.

$$p_\mu P(\mu \to \nu) = p_\nu P(\nu \to \mu) \tag{9}$$

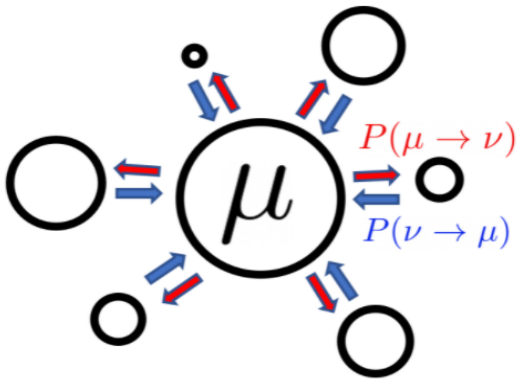



Fig: The Detailed Balance Diagram

The total energy is

$$E_\mu = \sum_{<i,j>} -J\sigma_i\sigma_j \tag{10}$$

where $\sigma_i$ is the spin of a single particle in the lattice (either $-1$ or 1) and the sum over $< i, j >$ means summing over nearest neighbours for all points in the lattice. $\mu$ corresponds to a particular configuration of the spins. Now, satisfying detailed balance

$$\frac{P(\mu \to \nu)}{P(\nu \to \mu)} = \frac{p_\nu}{p_\mu} = e^{-\beta(E_\nu - E_\mu)} \tag{11}$$

We'll start with a random lattice of spins, some pointing up and some pointing down, and make it flip around using the equation above until it fixes itself into equilibrium.

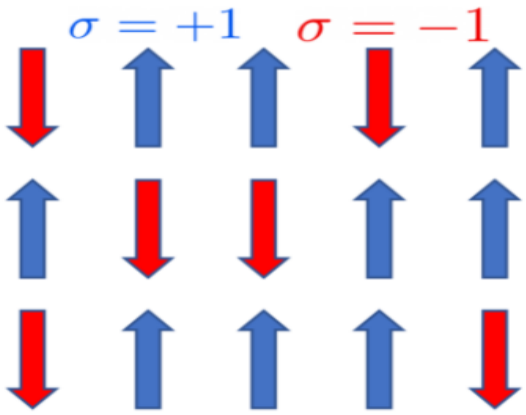



Fig: 2D Ising Model diagram

- Call the current state $\mu$.
- Pick a random particle on the lattice and flip the spin sign. Call the state $\nu$. We want to find the probability that $P(\mu \to \nu)$ that we'll accept this new state.
- (a) If $E_\nu > E_\mu$ then set $P(\nu \to \mu) = 1$ and thus by the detailed balance equation $P(\mu \to \nu) = e^{-\beta(E_\nu - E_\mu)}$.

  (b) If $E_\nu < E_\mu$ then $P(\mu \to \nu) = 1$, still satisfies detailed balance equation.
- Change to state $\nu$ (i.e. flip the spin of the particle) with the probabilities outlined above.
- Go back to step 1. Repeat the whole thing many many times and eventually you'll force out an equilibrium state.

Thus the only thing that needs to be evaluated is $\beta(E_\nu - E_\mu) = \beta J \sum_{k=1}^{4} \sigma_i \sigma_k$ where $i$ is the spin being flipped and $\sigma_k$ are the four nearest neighbours to that spin (two dimensions). On the boundaries, sometimes things might have less than 4 neighbours.

# 4 Python Code

## 4.1 Generating initial lattices

First create a random lattice grid representing spins up as $+1$ and spin down as $-1$. Here, I take 2 lattices. *lattice_n* represents lattice having 75% spin down (negative) and *lattice_p* represents lattice having 75% spin up (positive) as starting lattices.

```
# Import required packages.

import numpy as np
import matplotlib.pyplot as plt
import numba
from numba import njit
from scipy.ndimage import convolve, generate_binary_structure

# Generate a lattice grid of size 50x50

N=50
init_random = np.random.random((N,N))
lattice_n = np.zeros((N, N))
lattice_n[init_random>=0.75] = 1
lattice_n[init_random<0.75] = -1

init_random = np.random.random((N,N))
lattice_p = np.zeros((N, N))
lattice_p[init_random>=0.25] = 1
lattice_p[init_random<0.25] = -1
```

Now, define the function $E/J = -\sum_{<i,j>} \sigma_i \sigma_j$ to get the energy.

```
def get_energy(lattice):

    # applies the nearest neighbours summation
    kern = generate_binary_structure(2, 1)
    kern[1][1] = False
    arr = -lattice * convolve(lattice, kern, mode='constant', cval=0)
    return arr.sum()
```

## 4.2 Implementing Metropolis Algorithm

Now, implement the Metropolis Algorithm to our system.

- Takes in initial 2d grid of spins, number of time steps to run algorithm for, and temperature **BJ**
- Returns the total spin of all the atoms

Use the numba to fasten our program. It took nearly 3 hrs to generate the values initially, but with the use of numba, it takes just 4-5 mins.

```
numba.njit("UniTuple(f8[:],2)(f8[:,:],i8,f8,f8)",nopython=True, nogil=True
    )
def metropolis(spin_arr, times, BJ, energy):
    spin_arr = spin_arr.copy()
    net_spins = np.zeros(times-1)
    net_energy = np.zeros(times-1)
    for t in range(0,times-1):

        # 2. pick random point on array and flip spin

        x = np.random.randint(0,N)
        y = np.random.randint(0,N)
        spin_i = spin_arr[x,y] #initial spin
        spin_f = spin_i*-1 #proposed spin flip

        # compute change in energy

        E_i = 0
        E_f = 0
        if x>0:
            E_i += -spin_i*spin_arr[x-1,y]
            E_f += -spin_f*spin_arr[x-1,y]
        if x<N-1:
            E_i += -spin_i*spin_arr[x+1,y]
            E_f += -spin_f*spin_arr[x+1,y]
        if y>0:
            E_i += -spin_i*spin_arr[x,y-1]
            E_f += -spin_f*spin_arr[x,y-1]
        if y<N-1:
            E_i += -spin_i*spin_arr[x,y+1]
            E_f += -spin_f*spin_arr[x,y+1]

        # 3 / 4. change state with designated probabilities
```

```
33
34          dE = E_f-E_i
35          if (dE>0)*(np.random.random() < np.exp(-BJ*dE)):
36              spin_arr[x,y]=spin_f
37              energy += dE
38          elif dE<=0:
39              spin_arr[x,y]=spin_f
40              energy += dE
41
42          net_spins[t] = spin_arr.sum()
43          net_energy[t] = energy
44
45      return net_spins, net_energy
```

## 4.3 Plotting Time evolution of Net Magnetization

```
1 #Get arrays of spins and energies throughout the flipping (time evolution)
    .
2 spins_n, energies_n = metropolis(lattice_n, 1000000, 0.7, get_energy(
    lattice_n))
3 spins_p, energies_p = metropolis(lattice_p, 1000000, 0.7, get_energy(
    lattice_p))
4
5 #Plot the net Magnetization Evolution with time.
6
7 plt.figure(figsize=[12,5])
8 plt.plot(spins_n/N**2, label='75% of spins started negative', color='r')
9 plt.plot(spins_p/N**2, label='75% of spins started positive', color='k')
10 plt.xlabel('Algorithm Time Steps', fontsize=15)
11 plt.ylabel(r'Net Magnetization $(\bar{m})$', fontsize=15)
12 plt.suptitle(r'Time Evolution of Net Magnetization for $\beta J = 0.7$',
    fontsize=18)
13 plt.legend(loc='best', fontsize=15)
14 plt.grid()
15 plt.show()
```

## 4.4 Plotting Time evolution of Net Energy

```
1 #Plot the net Energy Evolution with time.
2
3 plt.figure(figsize=[12,5])
4 plt.plot(energies_n, label='75% of spins started negative', color='r')
5 plt.plot(energies_p, label='75% of spins started positive', color='k')
```

```
plt.xlabel('Algorithm Time Steps', fontsize=15)
plt.ylabel(r'Energy $E/J$', fontsize=15)
plt.suptitle(r'Time Evolution of net Energy for $\beta J = 0.7$', fontsize
    =18)
plt.grid()
plt.legend(loc='best', fontsize=15)
plt.show()
```

## 4.5 Calculating Statistical Values for different quantities

I can get $\bar{m}$ and $E/J$ for many different values of $\beta J$. The values will be the average of the last 100000 points. The $BJ$ values range from 0.1 to 2 in steps of 0.05.

```
def get_spin_energy(lattice, BJs):
    ms = np.zeros(len(BJs))
    ms_stds = np.zeros(len(BJs))
    E_means = np.zeros(len(BJs))
    E_stds = np.zeros(len(BJs))
    for i, bj in enumerate(BJs):
        spins, energies = metropolis(lattice, 1000000, bj, get_energy(
    lattice))
        ms[i] = spins[-100000:].mean()/N**2
        ms_stds[i] = spins[-100000:].std()
        E_means[i] = energies[-100000:].mean()
        E_stds[i] = energies[-100000:].std()
    return ms, ms_stds, E_means, E_stds

BJs = np.arange(0.1, 2, 0.05)
ms_n, ms_stds_n, E_means_n, E_stds_n = get_spin_energy(lattice_n, BJs)
ms_p, ms_stds_p, E_means_p, E_stds_p = get_spin_energy(lattice_p, BJs)
```

## 4.6 Variation of net Magnetization (avg spins) with Temperature

I plot now net magnetization $\bar{m}$ as a function of temperature $T = \dfrac{1}{\beta k} = \dfrac{J}{(\beta J)k}$

```
plt.figure(figsize=[14,6])
plt.plot(1/BJs, ms_n, 'o--', label='75% of spins started negative', color=
    'r')
plt.plot(1/BJs, ms_p, 'o--', label='75% of spins started positive', color=
    'k')
plt.xlabel(r'$\left(\frac{k}{J}\right)T$', fontsize=15)
plt.ylabel(r'Net Magnetization $(\bar{m})$', fontsize=15)
```

```
plt.legend(loc='best', fontsize=15)
plt.suptitle(r'Variation of Net Magnetization with Temperature for $\beta J=$0.7', fontsize=18)
plt.grid()
plt.show()
```

## 4.7 Variation of net Energy with Temperature

Now, plot net energy $\bar{E}$ as a function of temperature $T = \frac{1}{\beta k} = \frac{J}{(\beta J)k}$

```
plt.figure(figsize=(14,6))
plt.plot(1/BJs, E_means_n, 'o--', label='75% of spins started negative', color='r')
plt.plot(1/BJs, E_means_p, 'o--', label='75% of spins started positive', color='k')
plt.xlabel(r'$\left(\frac{k}{J}\right)T$', fontsize=15)
plt.ylabel(r'Net Energy $(\bar{E})$', fontsize=15)
plt.legend(loc='best', fontsize=15)
plt.suptitle(r'Variation of Net Energy with Temperature for $\beta J=$0.2', fontsize=18)
plt.grid()
plt.show()
```

## 4.8 Heat Capacity dependence on Temperature

Then, plotting Heat Capacity $C_V$ as a function of temperature, using the fact that

$$C_V = \frac{\sigma_E^2}{T^2}$$

$$= (< E^2 > - < E >^2) \cdot \beta^2 k^2$$

$$= (< (E/J)^2 > - < E/J >^2) \cdot (\beta J)^2 k^2$$

$$= \sigma_{E/J}^2 \cdot (\beta J)^2 k^2$$

```
plt.figure(figsize=(14,6))
plt.plot(1/BJs, E_stds_n*BJs, label='75% of spins started negative', color='r')
plt.plot(1/BJs, E_stds_p*BJs, label='75% of spins started positive', color='k')
plt.xlabel(r'$\left(\frac{k}{J}\right)T$', fontsize=15)
plt.ylabel(r'$C_V / k^2$', fontsize=15)
```

```
plt.legend(loc='best', fontsize=15)
plt.suptitle(r'Variation of Heat Capacity with Temperature for $\beta J=$0
    .7', fontsize=18)
plt.grid()
plt.show()
```

## 4.9 Magnetic Susceptibility dependence on Temperature

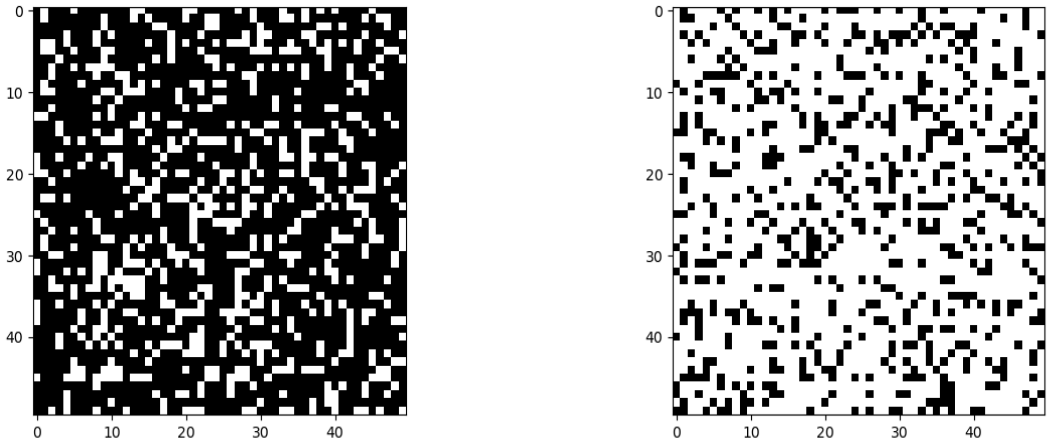
Now, I plot Magnetic Susceptibility $\chi$ as a function of temperature, using the fact that

$$\chi = \frac{1}{T} \cdot (< M^2 > - < M >^2)$$
$$= \beta J \frac{k}{J} \cdot (< M^2 > - < M >^2)$$
$$= \sigma^2_{M/\sqrt{J}} \cdot (\sqrt{\beta J})^2 k$$

```
plt.figure(figsize=(14,6))
plt.plot(1/BJs, ms_stds_n*np.sqrt(BJs), label='75% of spins started
    negative', color='r')
plt.plot(1/BJs, ms_stds_p*np.sqrt(BJs), label='75% of spins started
    positive', color='k')
plt.xlabel(r'$\left(\frac{k}{J}\right)T$', fontsize=15)
plt.ylabel(r' Magnetic Susecptibility $(\chi)$', fontsize=15)
plt.legend(loc='best', fontsize=15)
plt.suptitle(r'Variation of Magnetic Suceptibility $\chi$ with Temperature
    for $\beta J=$0.2', fontsize=18)
plt.grid()
plt.show()
```

# 5 Results and Analysis

I started by generating 2 initial lattices: (i) where I have 75% spin up states and (ii) where I have 75% spin down states using pseudo random number generator in python. The output of it is shown below in the figure 6.




(a) Lattice with 75% spin up states

(b) Lattice with 75% spin down states

Fig 6: Initial lattices generated by pseudo random number generator

Using these lattices, I calculated different thermodynamic observables required to study the system. I evaluated the time evolution of average Magnetization $< M >$ and average Energies $< E >$ of the system. I have taken different values of temperatures for the time evolution plots: $\beta J = 0.7$ and $\beta J = 0.2$. We can observe the changes which holds at different temperatures (shown as inverse temperatures $\beta J$).

Moreover, I plotted the Specific Heat Capacity $C_V$ and the Magnetic Susceptibility $\chi$ of the system. For these, I evaluated the standard deviations of the energies and the net magnetization of the last 100000 points (acceptable approximation).I have taken the range of the $\beta$ values ranging from 0.1 to 2 in steps of 0.05 to see the variations in the above thermodynamic quantities. The following subsections describe the results obtained and the detailed analysis of the outcomes.

I have performed the **Simulations of the Time Evolution of these lattices** and how they align or misalign is inferred from the **gif video** which I generated. It has the 3 different initial $50 \times 50$ lattices- (a) starting with 25% spin up (b) 50% spin up (c) 75% spin up.

The .gif video can be accessed by clicking here: 2D Ising Model Simulations

## 5.1 Time Evolution of Net Magnetization

So, initially the 75% of the spins were pointing up (black), so the net spins started from 0.5 and as the system evolves in time, the system fluctuates and after a period, all the spins are aligned up. The spins are still oscillating but the overall system comes to an equilibrium state or the detailed balanced state by the eq. 11, so the net magnetization remains +1. We can see the similar but vice versa results for the system which started initially with the 75% of the spin pointing downwards (see fig 7).

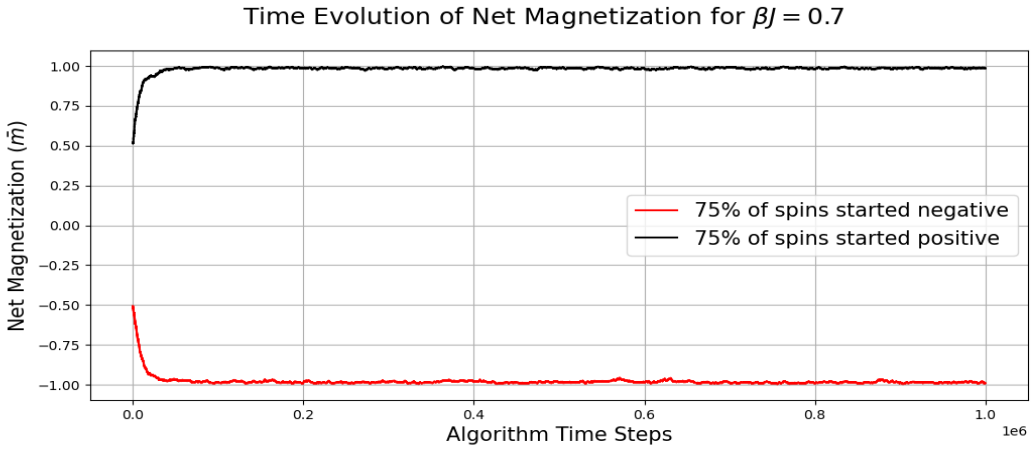



Fig 7: Time evolution of Net Magnetization with $\beta J = 0.7$

However, a remarkable change is seen when I increase the temperature bath, i.e. the value of $\beta J$ is decreased ($\beta J \propto T^{-1}$). The system tend towards misalignment. So, the system gets demagnetized, which means if we increase the temperature over a certain value (known as Curie Temperature $T_c$), the magnetization of the material is lost (see fig 8).

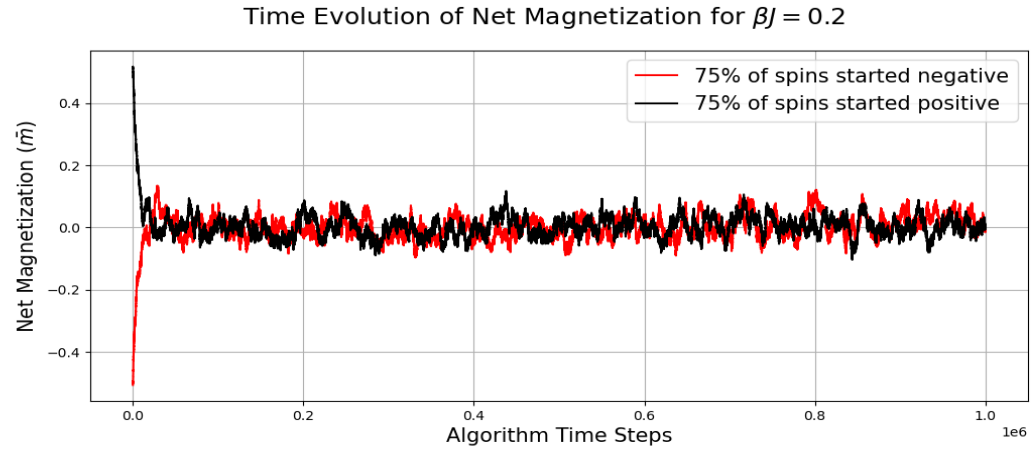



Fig 8: Time evolution of Net Magnetization with $\beta J = 0.2$

## 5.2 Time Evolution of Net Energy

If I look at the time evolution of another observable i.e. Average Energy $< E >$, then what we see is that if we have a low temperatures ($\beta J = 0.7$), then the system goes to the least energy state i.e the most favourable state (see fig 9).

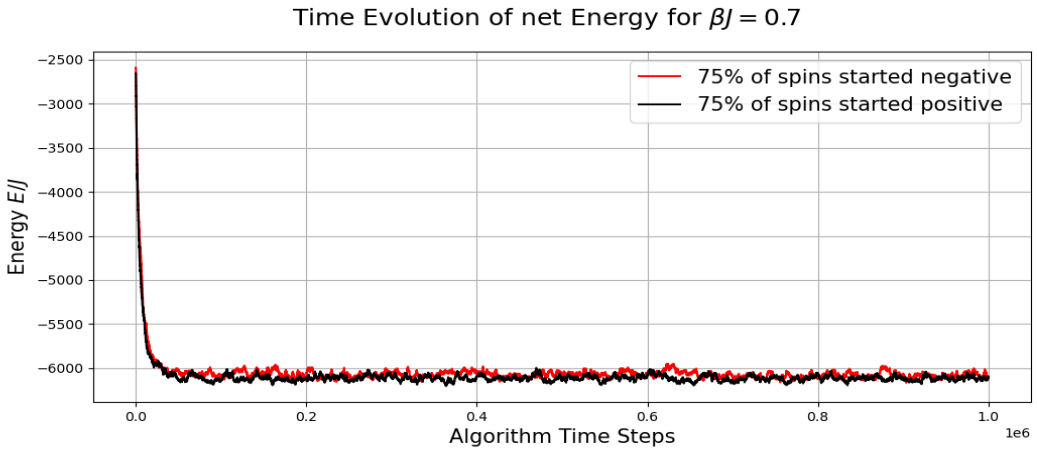



Fig 9: Time evolution of Net Energy with $\beta J = 0.7$

So, here we can see that the energy of the system has increased. It is because when we keep our system in a higher temperature thermal bath, the bath itself provides aided energies to the system which increases its energy and then oscillates at the same state (see fig. 10).

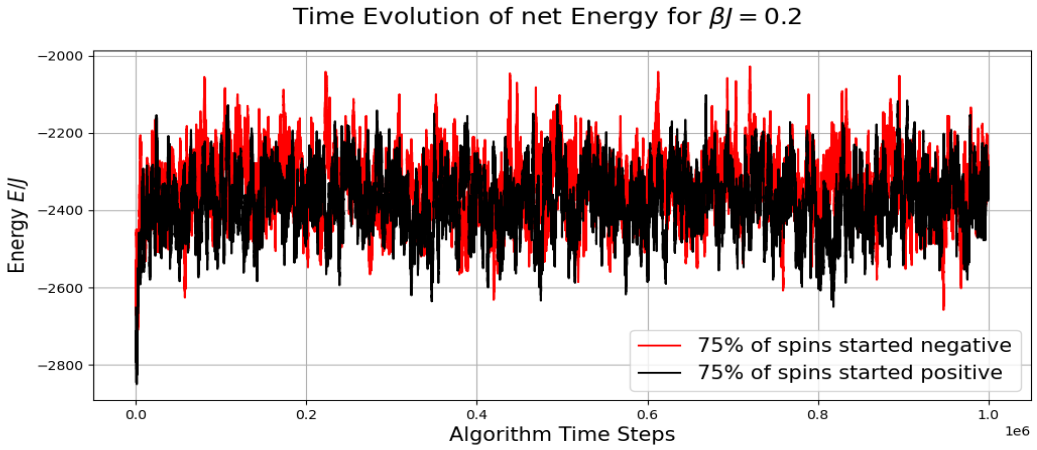



Fig 10: Time evolution of Net Energy with $\beta J = 0.2$

## 5.3 Variation of Net Magnetization with Temperature

Now, to know how thee average spins or the magnetization in the detailed balance depends on temperature since, we saw that at one temperature the spins co-align all together, whereas at some higher temperature, all the spins misalign themselves and the material loses its magnetization. So, whatever weird going on between these two temperatures, one must find out. I took the last 100000 points and took its average (good approximation) when the system is in equilibrium as it reaches very quickly, so the last 100000 is pretty safe.

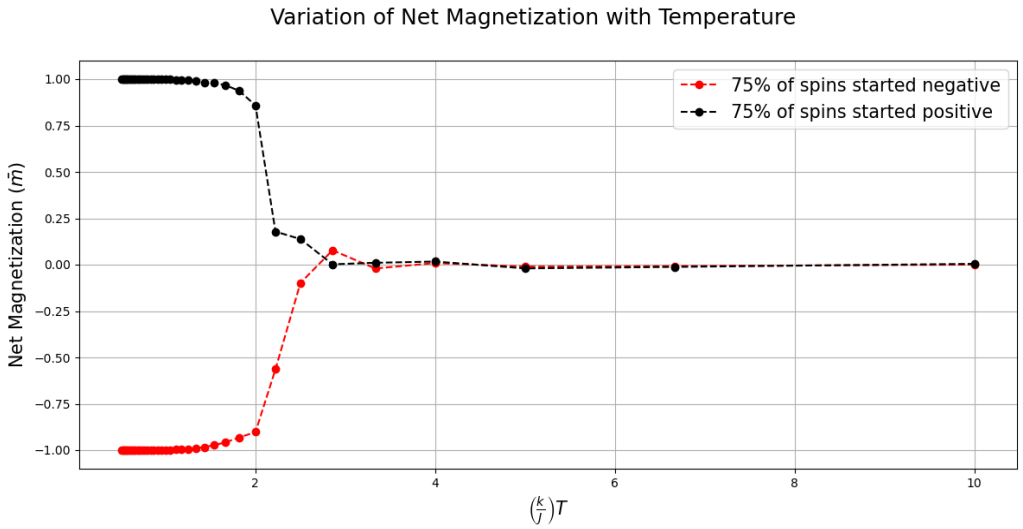



Fig 11: Variation of Net Magnetization $< M >$ with Temperature T

The figure above has the x-axis as the temperature variable i.e. $A \cdot T$, where $A$ is a constant. So, the values to the right imply the increase in temperature values. So, at the left extremities, when the temperature of the surroundings of our system (thermal bath is cold), so the spins favour to align themselves to align in the same direction (energetically favourable). So, depending on the initial conditions (whether most of them are pointing up initially or pointing down), it starts off by alignment of the spins altogether at low temperatures ($\bar{m} = \pm 1$).

But, as the temperature of the bath increases, there is a phase transition which occurs (as can be seen from the fig. 11). This phase transition is the one, on sides of which, the spins align or misalign themselves. At higher temperatures, the all the spins tend to misalign themselves (as the thermal bath allows it so) and the system gets demagnetized.

## 5.4 Variation of Net Energy with Temperature

Now, if I look at the variation of the mean energy $< E >$ with the temperatures $T$, we see that at low temperatures, the system tries to be in its low energy state (the thermal bath is also not providing enough energy to the system to go to a high energy state). All the spins co-align and the domains of the material gets magnetized completely. So, at low temperatures, the interactions between the spins cause them to align in the same direction (see left side of the fig. 12).

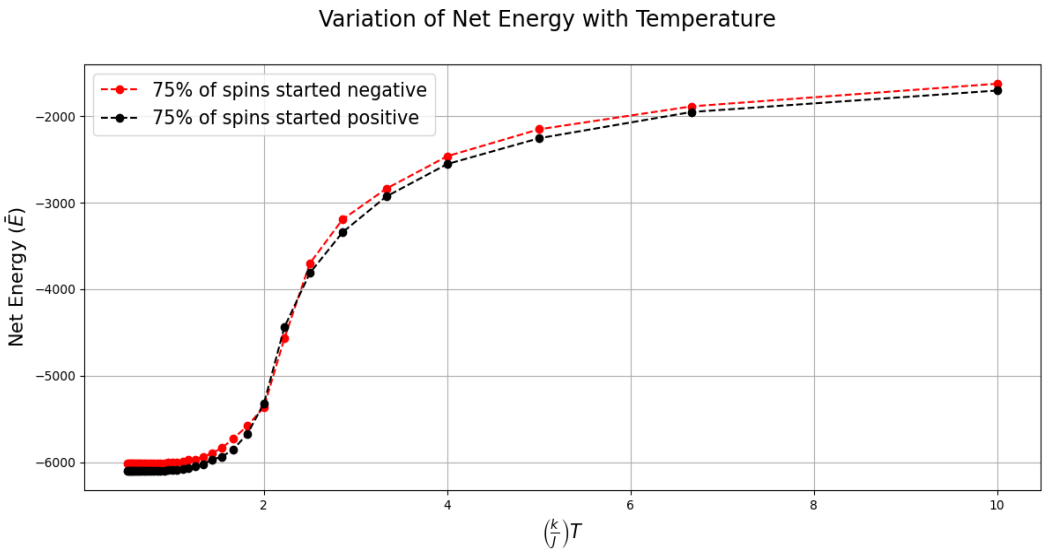



Fig 12: Variation of Net Energy $< E >$ with Temperature T

When the bath is heated to higher temperatures, the orientation and the behaviour of the system changes drastically. There exists a temperature known as Curie Temperature $T_c$ at which the material is no longer *ferromagnetic*. Above the Curie temperature $T_c$, the material undergoes a phase transition and becomes paramagnetic, with all the magnetic moments orienting randomly due to thermal effects. This also indicates the possibility of the phase transition at this critical temperature $T_c$. The curve at the right side of the plot becomes increasingly flatter and flatter with increase in the mean energy $< E >$ as the temperature of the bath increases. High temperatures imply high rise in the mean energy of the system dominated by thermal effects.

## 5.5 Heat Capacity dependence on Temperature

Now, I look at how Specific heat capacity changes with the temperatures and information it contains. I calculated this by evaluating the standard deviations of the mean energy for the last 100000 points. It is a nice approximation as the system reaches equilibrium at very early stages (see fig. 9 and 10). Thus the Specific heat capacity is calculated from the standard deviations of the observable energy $E$. It is derived from the equation:

$$C_V = \frac{1}{T^2}(< E^2 > - < E >^2) \tag{12}$$

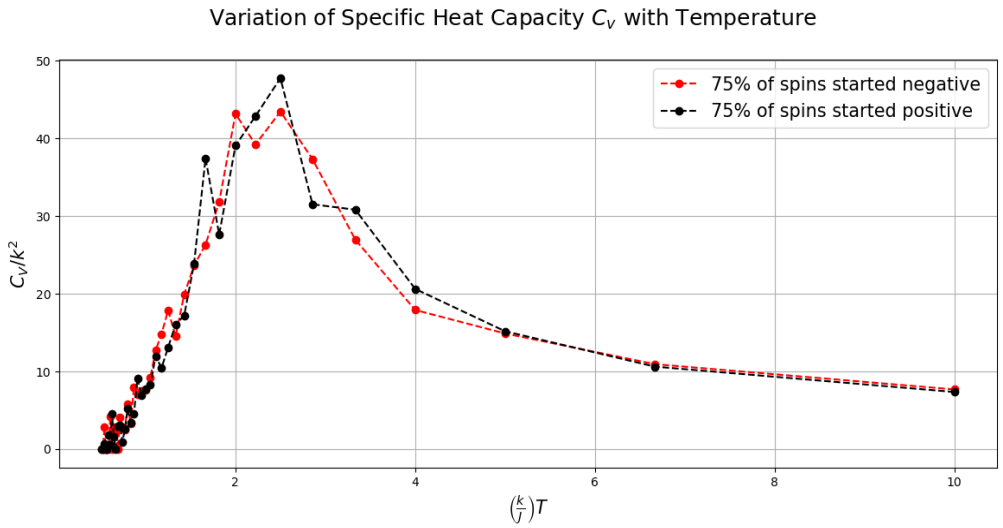



Fig 13: Variation of Heat Capacity $C_V$ with Temperature T

Looking at the graph itself, it is very clear the phase transition occurs at the $\frac{k}{J}T \approx 2.5$. If we take natural units $k = 1$ and $J = 1$, then at $T \approx 2.5$, there is a strong possibility that a second order phase transition occurs here. This continuous change in the behavior of the system from disorder ($< M >= 0$) to order, with the operation $s_{ij} \rightarrow -s_{ij}$ resulting in $< M > \rightarrow - < M >$, is characteristic of a second order phase transition.

There are two results shown above. One where the system started with more spins pointing up and the other one had more spins pointing down initially. Ideally these 2 plots should be the same. Moreover, the critical temperature where there is a phase transition, the metropolis algorithm is known to perform poorly. So, this means that there is a particular temperature where if I go beyond it, a lot of heat is added to the system, and the spins get misaligned. The more I go beyond $T_c$, less energy is getting added up to the system, so the graph dips below.

## 5.6 Magnetic Susceptibility dependence on Temperature

The magnetic susceptibility is a measure of how much a material becomes magnetized when in a magnetic field. It is represented by $\chi$. Now, I evaluate the magnetic susceptibility $\chi$ of the system and see how it varies with the changes in the temperatures of the bath. Again, I calculated this by evaluating the standard deviations in the mean magnetization for the last 100000 points which is a good approximation given that the system reaches equilibrium at very early stages of the time (see fig. 7 and 8). So, the magnetic susceptibility is calculated from the standard deviations of the observable magnetization $\bar{m}$. It is derived below:

$$\chi = \frac{1}{T}(< M^2 > - < M >^2) \tag{13}$$

Plotting the same we get the figure 14 shown below.

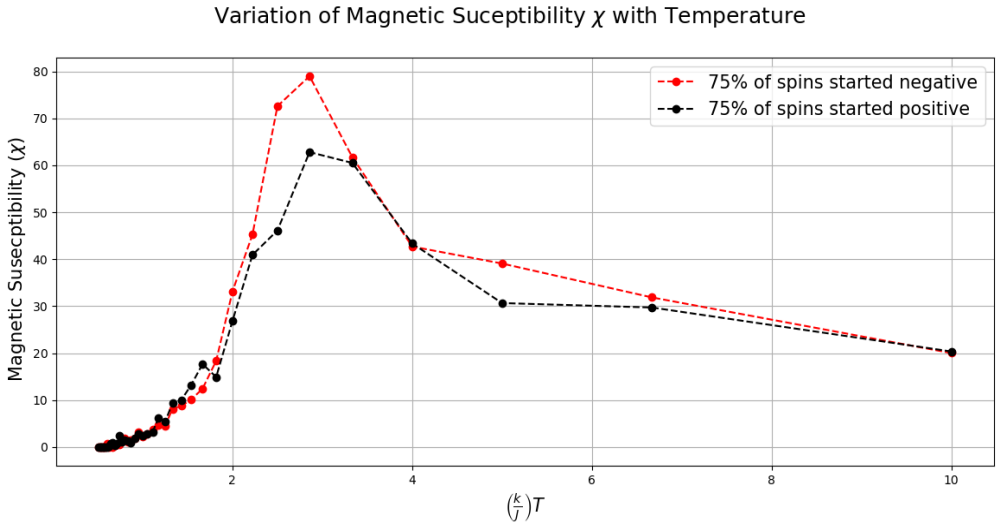



Fig 14: Variation of Magnetic Susceptibility $\chi$ with Temperature T

We can interpret from the graph above that after a particular temperature, i.e. Curie's temperature ($T_c$) which is around 2.5 in natural units, the susceptibility $\chi$ of the system starts decreasing because the increased temperature also increases random motion between the molecules, this counteracts the magnetization and thus decreases the susceptibility of the material (right extremity of the graph above).

# 6 Conclusions

In this article, I focused on four physical quantities: energy, magnetic susceptibility, magnetization, and heat capacity. These quantities are those that are interconnected and define a system well because they are fundamental physical properties although energy is the basis of all the relationships.

- From the time evolution plots in the sections 5.1 and 5.2, we see that for given certain temperature of the bath, the system either goes to a state where all the spins become aligned with the material getting magnetised, or the system possesses random spins and the material getting demagnetized.
- From a qualitative point of view, we notice that there is an inflection point for $T = T_C$ the energy profile (ref. fig. 11, 12, 13 and 14) but also that magnetization, thermal capacity and susceptibility fall to zero from this temperature (we can observe this in section 5). This means a change in the system behavior, which corresponds to the definition of the critical temperature known as Curie Temperature above which the material loses all its permanent magnetic properties. The permanent magnetic properties are caused by the alignment of the magnetic moments. Thus, above $T_C$, there are on average an equal number of domains ordered with upward and downward spin, which can be illustrated by the convergence of magnetic moment, specific heat and susceptibility towards zero.
- For the 2D Ising model, we observed in the sections 5.5 and 5.6 that there is a characteristic change in the behavior of the system below a certain temperature as we decreased the temperature. This indicates that a second order ferromagnetic phase transition took place and we suggested $T_C \approx 2.5$ to be the critical temperature based on our results. The acceptance rate is also very low for lower temperatures and no tweaking of the simulation could fix it. This might indicate that the Metropolis algorithm might be inefficient to study this model at low temperatures and some other method with higher acceptance rates could be much faster. Nevertheless, the Metropolis algorithm was able to simulate the 2D Ising model with excellent results.
- There exists a strong relationship between the accuracy of the results and the size of the system depends we are examining. The larger the size of the system, the higher the computational cost. Thus, since the cost of time is really important, we need to think about the trade-off: time and precision. We can check whether the simulation results for large systems actually converge towards the thermodynamic limit behaviour.

# References

(2017). The metropolis algorithm and the ising model. In *Computational Physics*, pages 89–104. WORLD SCIENTIFIC.

Joseph, A. (2020). Markov chain monte carlo methods in quantum field theories: A modern primer. *SpringerBriefs in Physics*.

Wikipedia (2021a). Ising model Wikipedia, the free encyclopedia. [Online; accessed 22-July-2004].

Wikipedia (2021b). Monte carlo method Wikipedia, the free encyclopedia. [Online; accessed 22-July-2004].