King's College London
6CCP3131 - Third Year Project in Physics

# 2D Ising Model: Investigating Magnetisation, Susceptibility and the Phase Transition

**Written by** Moesa Mahdi

**Supervisor** Evgeny Kozik

April 2, 2020

**Abstract**

The Ising Model is a simple but powerful model which can be used to illustrate varies properties of magnetism. By investigating the effect of temperature on the net magnetisation and susceptibility of a square lattice, a phase transition can be observed. To do this the Metropolis Monte Carlo method was used - a bias sample of lattice configurations is taken in which configurations that have a bigger contribution to calculating an observable are prioritised. We observed a phase transition from para-magnetism to ferromagnetism occurring when the lattice is cooled to a critical temperature. This is in agreement with theory which states that the phase transition occurs because of spontaneous magnetisation, in which a ferromagnetic state becomes more energetically favourable. Results for the critical temperature were also inline with the established value and was found by using the concept of size-independence of binder ratios at the critical temperature. This investigation teaches us that a simple model can in fact demonstrate important physics and can serve as a gateway to study concepts that extend outside magnetism.

## 1 Introduction

**Ising Model** Imagine a 2-dimensional square lattice where each lattice point has the property spin up or spin down. The Hamiltonian is

$$H_i = -J \sum_{<ij>} \sigma_i \sigma_j \tag{1}$$

where $J$ is the coupling constant and $< ij >$ is the nearest nearest to $\sigma_i$. A negative $J$ would imply that the lattice prefers anti-parallel alignment, whereas a positive $J$ would prefer parallel alignment. For this project $J = 1$, therefore our lattice exhibits ferromagnetic properties. We will also apply the periodic boundary conditions to this system, meaning that for a point chosen at the edge of the lattice, it will interact with lattice points on the opposite side to that edge.
The partition function for such a system is given by

$$Z = \sum_r e^{-\beta H_r} \tag{2}$$

where $\beta = (k_b T)^{-1}$ and $k_b = 1$. $r$ is a state/configuration of the system or is often referred to as a vector in the phase space, which is the space of all states. By evaluating the partition function the 2D Ising model can be solved analytically. The most significant result is that a para-magnetic lattice undergoes what is called *spontaneous magnetisation* at a critical temperature $T_c$. The key idea is that the magnetic dipoles in a ferromagnet will *spontaneously* align when cooled to the critical temperature due to it being more favourable energetically to do so. This phenomena is called a phase transition and is an important study in physics. It is important to realize that the phase transition is simply the point at which a system is observed to transform

from one state or phase to another. So it can also refer to many other phenomena such as the temperature at which a metal turns into a superconductor or the formation of a Bose-Einstein condensate [1]. Many different mathematical models or phenomena will behave in the same way as they approach a particular limit. In the case of the phase transition, as the critical temperature is approached, different phenomena will have the same critical exponents; this is what it means to be in the same universality class. This implies that studying the phase transition in the Ising model can help to understand other phase transitions that appear in other phenomena belonging to the same universality class. It is useful then to investigate the Ising model and its phase transition.

The probability of finding the system in a particular state $r$ is

$$P_r = \frac{1}{Z} e^{-\beta H_r} \tag{3}$$

One more useful fact we can take from statistical mechanics is the expectation value of an observable $A$ is

$$< A >= \sum_r A_r P_r \tag{4}$$

where $A_r$ is the value of the observable at state $r$.

Consider the magnetisation $M$ of a system. Calculating the expectation value of $M$ using equation (4), for a lattice of size $L \times L$, will involve summing over every possible state; bearing in mind each spin as two states then this will result in $2^{L \times L}$ states. This will also involve all the states with very small probability $P_r$ which contribute very little to $< M >$. It is clear that this method is very inefficient and time-consuming. Taking a sample of the total number of states/configurations would help to solve this issue; it is not, however, enough to take a simple sample. This is because the sample of configurations might not represent the entire number of states well and important information might be lost. As stated before some states have small probabilities and therefore do not contribute much to the overall sum. Instead a bias sample is taken in which the configurations chosen are proportional to their contribution to the sum in equation (4). In other words in each step, the configurations with the largest contribution will be chosen first, and as the number of steps increases you will get closer to an exact answer. This also means that we can control the error bars; the error bar decreases with the square root of the number of steps.

**Metropolis Monte Carlo** Plugging eq.(3) into eq.(4) for the magnetisation gives

$$< M >= \frac{\sum_r^N M_r e^{-\beta H_r}}{\sum_r^N e^{-\beta H_r}} \tag{5}$$

To sample the appropriate configurations $r$, divide the top and bottom by the Boltzmann distribution $e^{-\beta H_r}$, therefore eq.(5) simplifies to

$$< M >= \frac{\sum_r^N M_r}{N} \tag{6}$$

So the average magnetisation is just the sum of the magnetisation of each configuration over the total number of configurations.

The type of sampling that the Monte Carlo method implements is called a Markov chain. This approach stochastically (randomly) selects a state $r'$ based on the previous state $r$. The probability of going from state $r$ to $r'$ is called the transition probability $W$. The probability of going from state $r \to r'$ must be equal to its reverse $r' \to r$. This is called the detailed balance and is given by

$$P_r W(r \to r') = P_{r'} W(r' \to r) \tag{7}$$

Note that the state chosen to update can be anything, in our implementation of the Monte Carlo method we haven chosen to update one spin of the lattice per step. using eq.(3) to give

$$\frac{W(r \to r')}{W(r' \to r)} = \frac{P_{r'}}{P_r} = e^{-\beta(H_{r'} - H_r)} \tag{8}$$

where the change in the Hamiltonian using eq.(1) is

$$H_{r'} - H_r = 2J\sigma_k \sum_{<kj>} \sigma_j = \Delta E \tag{9}$$

Equation (8) is called the acceptance ratio. Plugging equation (9) into (8) we get the most useful form

$$R = e^{-\beta \Delta E} = e^{-\frac{\Delta E}{T}} \tag{10}$$

The acceptance ratio contains the necessary information to decide whether to flip a spin or not.

Let's take a step back and run through the Monte Carlo method in more general terms. For a randomly generated lattice choose a lattice point (making sure the choice is fair - all points have an equal chance of being picked) to decide whether to flip its spin or not. We know that if the energy of the flipped state $E_{r'}$ is lower than or equal to the energy of the current state $E_r$ then the spin should flip. In this case $\Delta E \leq 0$, which will give a value of $R \geq 1$. In the opposite case where $E_r > E_{r'}$ then there is a probability of flipping given by R (eq.10). In other words:

$$P(r \to r') = \begin{cases} 1, & \text{if } R \geq 1 \\ e^{-\frac{\Delta E}{T}}, & \text{if } R < 1 \end{cases} \tag{11}$$

# 2 Methodology

Many different observables can be studied by simulating the Ising model, two of which are the magnetisation and susceptibility of the system; the phase transition can be found from both of these observables. Looking at magnetisation first, as mentioned before, the magnetisation is defined as the sum of the spins in the system divided by the size of the lattice. This is very easy to calculate as the spins are represented as +1 for spin up and -1 for spin down. The lattice is initialized with a random set of spin up and down points before running the Monte Carlo algorithm, which is then run for a minimum of $N = 10^6$ Monte Carlo steps. To see the relationship between the magnetisation and the temperature, the Monte Carlo algorithm is run for temperatures between 1 and 5 with an increment of 0.1.

## 2.1 Calculating Error

**Standard Deviation** There are numerous methods available to find the error in the magnetisation, some of which include: the blocking method, the bootstrap method and the jackknifes method. All these methods involve the use of the standard deviation $\sigma$, which is given by

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N} (M_i - < M >)^2}{N}} \tag{12}$$

The bootstrap and jackknifes methods both work on the concept of taking a set of measurements and re-sampling this set a number of times, where the error is obtained by taking the standard deviation of the means of each sample. The bootstrap method re-samples by randomly choosing measurements from the original data set [2]; whereas the jackknifes method re-samples by omitting one measurement from the original data set - in other words to remove the $i$th measurement from the set for one sample, then the $(i+1)$th measurement to create another sample and so on until you reach the last element of the set [3].

**Blocking Method** Another method to calculate the error is through the blocking method. The idea behind this method is to put the measurements into blocks of size $Z$. Once $N_b$ number of blocks are filled, group adjacent blocks into one block; so blocks $A_1 + A_2$, $A_3 + A_4$ and so on, while keeping the total number of blocks $N_b$ the same. This implies that $Z$ will double, and as $N_b$ is the same then half the total amount of

blocks are now empty to be filled again. This process repeats until the algorithm stops taking measurements. The error is then given by

$$\sigma = \frac{1}{N}\sqrt{\sum_{i}^{N}(< B >_i - < M >)^2} \tag{13}$$

where average magnetisation $< M >$ is equivalent to the average of the block averages $< B >$.

Each of these methods have their pros and cons. Chernick (2008) states in his book [4] that the bootstrap (and jackknifes) methods have the advantage of not requiring any simplification to be made to the data. For a skewed population, any sample extracted from the data will also be skewed as well. However, samples need to be large enough to reflect the data well. Another benefit is that the bootstrap method does not require an analytic expression to calculate an observable, numerical approaches such as the Monte Carlo method are enough [4].
Having said this, we will use the blocking method to calculate the error. Let us take a step back to understand why. The obvious approach to obtain the error is to just calculate the standard deviation of the entire set of measurements. The problem with this is if the measurements are correlated, the calculated error will not be accurate. The main advantage of the blocking method is the ability to study the correlation between the error of each block. Recall from eq.(13) that the error is calculated by looking at the mean of each block; therefore if the block size is doubled our input to eq.(13) is different and will yield a different result if the measurements in the blocks are correlated. Remember that when the blocks are filled, the block size is doubled, which implies that a larger amount of steps $N$ will result in the block size doubling more often. Therefore the blocking method gives us a way to test at what $N$ will the measurements no longer be correlated.

**Calculating susceptibility & its error** The susceptibility $\chi$ is defined as

$$\chi = \frac{dm}{dh} \tag{14}$$

which is the change in magnetisation for a change in an applied external field. We can use (from [5])

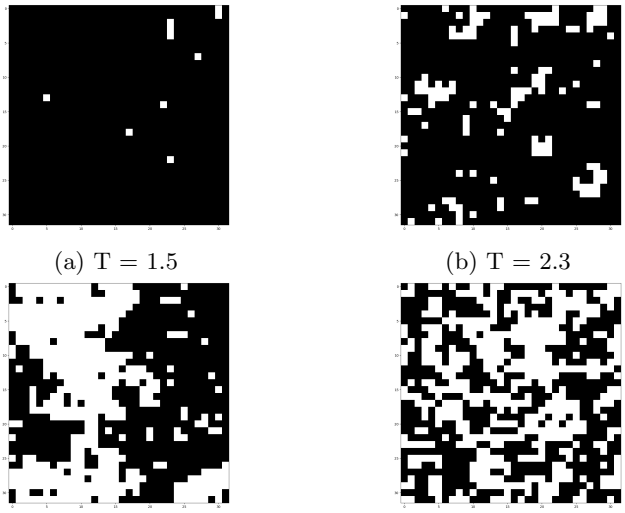
$$\chi = \frac{L^2}{T}(< M^2 > - < |M| >^2) \tag{15}$$

The error in the susceptibility can be calculated through error propagation. The error in $< M^2 >$ is calculated through eq.(13) by replacing $M$ with $M^2$. The error in $< |M| >^2$ is just the square of eq.(13). Hence the error in susceptibility $\sigma_\chi$ is

$$\sigma_\chi = \frac{L^2}{T}\sqrt{\sigma^2_{<M^2>} + \sigma^2_{<|M|>^2}} \tag{16}$$

# 3 Results

**Spin lattice configuration** Running the Monte Carlo algorithm for $N = 2 \times 10^6$ (for all following plots) we obtain the following visual of lattice configurations.



(a) T = 1.5 (b) T = 2.3

(c) T = 2.5 (d) T = 4.5

Figure 1: Lattice configurations (end of Monte Carlo algorithm) for system size $L = 32$ for various temperatures

Figure (1) illustrates that at high temperatures the lattice is para-magnetic; as the system approaches the critical temperature, a phase transition is observed from a para-magnetic state to a ferromagnetic state. At $T > T_c$, the lattice never keeps it's magnetism because all the spins will always have enough energy to flip. Whereas at $T < T_c$ it is energetically favourable for the spins to align and create a net magnetic lattice. A phase transition can be seen between lattice (b) and (c), between temperatures 2.3 and 2.5. This is not however a effective way to find the critical temperature.

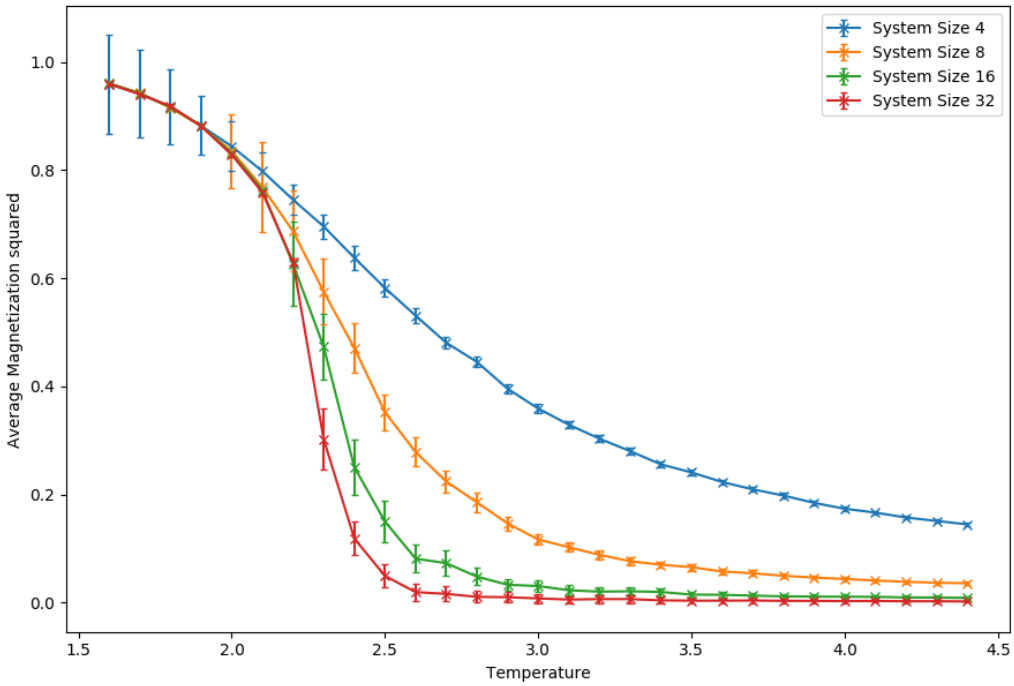



Figure 2: Plot of average magnetisation squared $< |M| >^2$ against the temperature for various system sizes

Looking at figure (2), the trend (for an increase in temperature) for all system sizes is for the lattice to start at a net magnetized state (net +1 spin or ferromagnetic) and then upon being in close proximity to the critical temperature, a phase transition occurs and the lattice tends to a net zero magnetisation (paramagnetic). Notice also that larger system sizes have a steeper decrease in magnetisation, reaching near zero much faster. To understand this let's look at the finite-size scaling hypothesis.

**Finite-size scaling hypothesis** The correlation length is the size of a cluster of spin up or down. This quantity can be expressed as $\xi = t^{-\nu}$, where $t = T - T_c$. The hypothesis takes the guess that $\frac{\xi}{L}$ controls the behaviour of an observable [6]. For magnetisation this will be of the form

$$M = L^{\sigma} f(\frac{\xi}{L}) = L^{\sigma} f(\frac{t^{-\nu}}{L}) = L^{\sigma} f((tL^{\frac{1}{\nu}})^{-\nu}) = L^{\sigma} g(tL^{\frac{1}{\nu}}) \tag{17}$$

To determine $\sigma$, we know that for an infinitely sized lattice and small $t$, the magnetisation is [5]

$$M \sim (T_c - T)^{\beta} \tag{18}$$

When $tL^{\frac{1}{\nu}} \to \infty$ for our guess, then $g(tL^{\frac{1}{\nu}}) = (tL^{\frac{1}{\nu}})^a$. Combining this with equation (17) we get

$$M = L^{\sigma}(tL^{\frac{1}{\nu}})^a = t^a L^{\sigma + \frac{a}{\nu}} \tag{19}$$

In order for equation (18) to be true, the exponent $a = \beta$ and $\sigma = -\frac{\beta}{\nu}$. So then equation (17) becomes

$$M = L^{-\frac{\beta}{\nu}} g(tL^{\frac{1}{\nu}}) \tag{20}$$

When we approach the critical temperature $t \to 0$, at this point $g(tL^{\frac{1}{\nu}}) = g(0) =$ constant. Therefore we can see from eq.(20) that the magnetisation will be dependent on $L^{-\frac{\beta}{\nu}}$. This explains the large differences in magnetisation at the critical temperature seen in figure (2). Whereas for $T >> T_c$ and $T << T_c$, in other words when $t \to \infty$, we stated that our guess $g(tL^{\frac{1}{\nu}}) = (tL^{\frac{1}{\nu}})^{\beta}$. It is clear that if you plug this into eq.(20), $L$ will cancel and therefore the magnetisation will not depend on system size.

**Susceptibility** We know that for an infinitely sized lattice and small t we have [5]

$$\chi \sim t^{-\gamma} = \frac{1}{(T - T_c)^{\gamma}} \tag{21}$$

where $\gamma$ is the critical exponent. Clearly as $T_c$ approaches $T$, the susceptibility approaches infinity. So then theoretically $\chi$ should peak at the phase transition which can correctly be seen in figure (3).

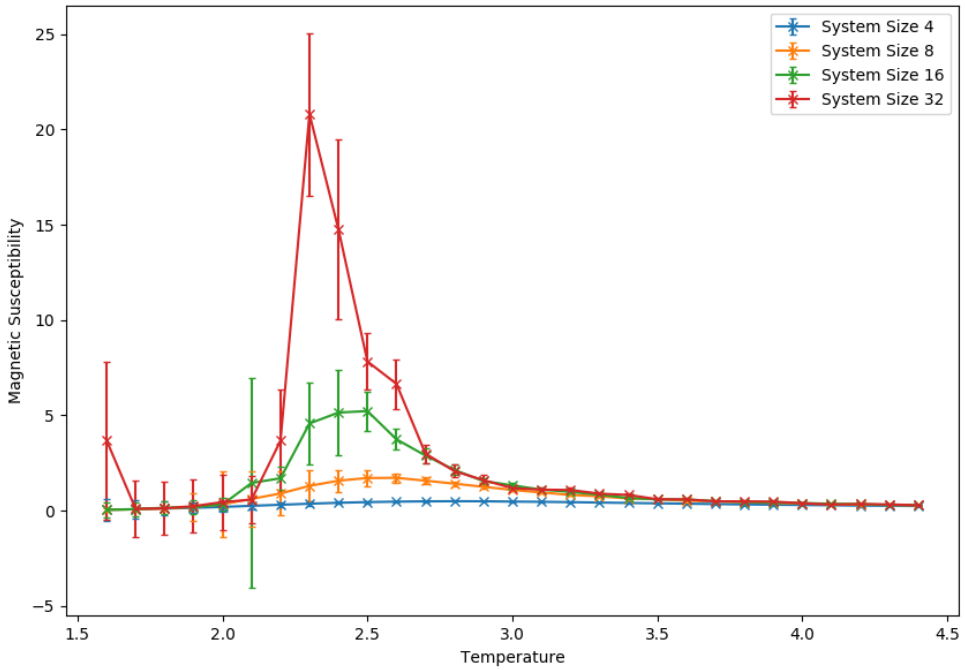



Figure 3: Plot of the susceptibility against temperature

Just like the magnetisation as seen in eq.(17), the susceptibility can also be said to be

$$\chi = L^{\rho} f(\frac{\xi}{L}) = L^{\rho} g(tL^{\frac{1}{\nu}}) \tag{22}$$

Again we can determine $\rho$ by using the same argument as before. When $L^{\frac{1}{\nu}} \to \infty$, we get

$$\chi = L^{\rho}(tL^{\frac{1}{\nu}})^{b} = t^{a} L^{\rho + \frac{b}{\nu}} \tag{23}$$

From eq.(21), we must have $b = -\gamma$ and $\rho = \frac{\gamma}{\nu}$ and we get

$$\chi = L^{\frac{\gamma}{\nu}} g(tL^{\frac{1}{\nu}}) \tag{24}$$

When we approach $T_c$, $t \to 0$ and so we will get $g(0)$ just as with magnetisation, hence $\chi$ will be dependent on the system size. This explains the large differences in $\chi$ between system sizes.

## 3.1 Obtaining Critical Temperature

The finite-size scaling hypothesis gives us a way to get an accurate value for the phase transition. From Sandvik (2018) [5], we know that for an infinitely sized lattice at the thermodynamic limit,

$$< M^2 > \sim t^{-\gamma} \tag{25}$$

$$< |M| > \sim t^{-\frac{\gamma}{2}} \tag{26}$$

Therefore using the same arguments as done for magnetisation and susceptibility, the finite-size scaling for these two quantities will be

$$< M^2 >= L^{-\frac{\gamma}{\nu}} g(tL^{\frac{1}{\nu}}) \tag{27}$$

$$< |M| > L^{-\frac{\gamma}{2\nu}} g(tL^{\frac{1}{\nu}}) \tag{28}$$

Clearly then if we square eq.(28) and take the ratio of the two, $L$ will cancel. This ratio is called the Binder ratios [8] and is given by

$$B = \frac{< M^2 >}{< |M| >^2} \tag{29}$$

To reiterate the binder ratio will be independent of the system size at the critical temperature. This is very useful because it implies that plots of this quantity against temperature for various $L$ will intersect at $T_c$. A close up of this is seen in figure (4). The error for the binder ratio values (not $T_c$) is calculated through error propagation given by

$$\sigma_B = B\sqrt{\left(\frac{\sigma_{<M^2>}}{< M^2 >}\right)^2 + \left(\frac{\sigma_{<|M|>^2}}{< |M| >^2}\right)^2} \tag{30}$$

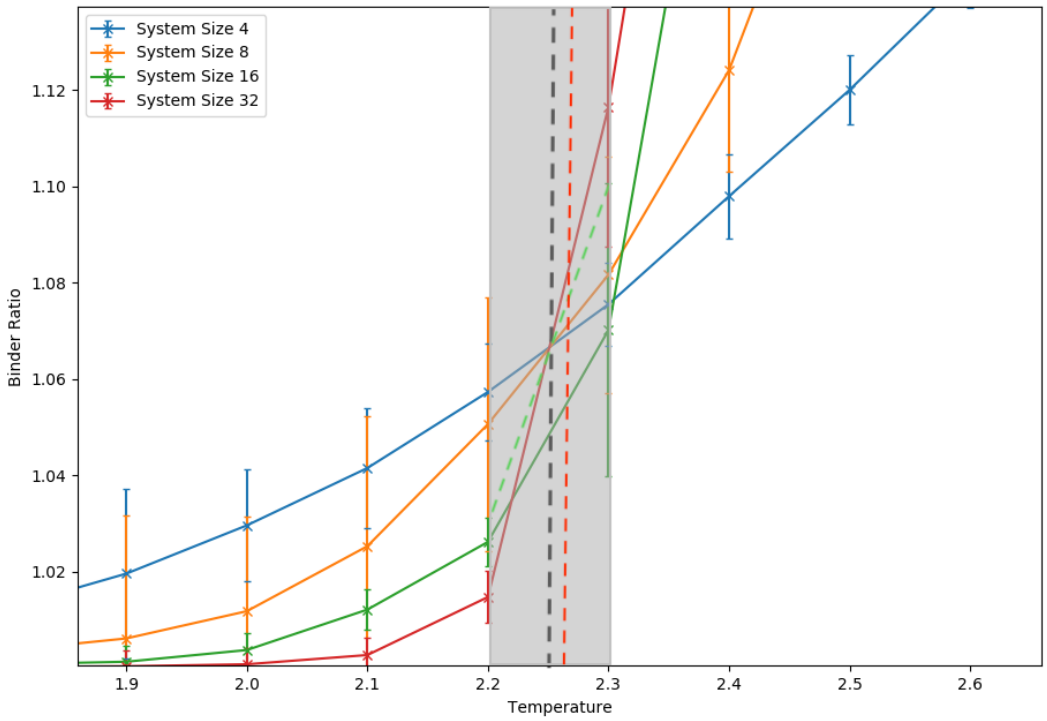



Figure 4: Plot of binder ratio vs temperature, zoomed in. Dotted black line shows the obtained critical temperature; dotted red line shows the exact critical temperature. Dotted green line shows another possible line (for L=16) within its error bars which crosses the critical temperature. The shaded grey region includes all possible values of $T_c$.

Figure (4) shows the system sizes 4, 8 and 32 crossing at the same point at 2.25. System size 16 does not cross at this point, but there is a line within its error bars which does cross at the same point (shown by the dotted green line. By examining the error bars of each system size around $T_c$ the error in $T_c$ can be

determined visually. The range of possible critical temperatures are within 2.2 and 2.3 as seen by the grey box in figure (4). Therefore $T_c = 2.25 \pm 0.05$J (in units of joules as $k_b = 1$), which agrees with the exact solution of 2.269 solved by Onsager [9].

# 4 Discussion & Improvements

In this simple approach to investigate the relationship between magnetisation and susceptibility with temperature, and to find the critical temperature, the results found were in agreement with established results. This investigating has been a gateway into ideas which reach far beyond just describing ferromagnetism. We have learnt that in fact the Ising model can be used to understand many different statistical models. A close look at the methods we used also introduces us the more efficient methods to simulate physical systems.

It was found that a high number of Monte Carlo steps was required to obtain reasonable results, which on a basic computers is very time consuming. The algorithm especially slows down near the critical temperature; this is because our Monte Carlo implementation updates only one spin at a time. Running the code on $N < 5 \times 10^5$ steps results in very large fluctuations and unreadable results. This limited other interesting possible experimentation to be done for this project; for example system sizes above $L = 32$ were too time consuming to investigate. A possible solution to this would be to use a more fundamental programming language such as C++ as opposed to python. Python works by interpreting code at runtime rather than being compiled to native code at compile time. Another possible solution is to run separate scripts simultaneously, where each explores a different observable or parameter. We could also only study temperatures close to the critical temperature which will allow larger system sizes and larger number of steps to be studied.

Possible further research that could be done is investigating the critical phenomena using the concepts of finite-size scaling. Obtaining the critical exponents associated with the Ising model and exploring other observables or functions such as the heat capacity and the correlation function are also compelling routes to go when looking at the Ising model.

# 5 Conclusion

The Ising model is a fascinating topic to explore due to being able to demonstrate complex concepts with a relatively small amount of effort and technical ability. However obtaining the exact expectation value of an observable would involve a summation over all possible states of the system; this kind of calculation is unreasonable. The metropolis Monte Carlo method provides a solution to this by taking a bias sample of states in which the states/configurations chosen is proportional to their contribution to the sum.

A variety of phenomena can be observed through this model. This project explored how the temperature affects the net magnetisation and susceptibility of a system. By looking at the lattice configuration at various temperatures and plotting the net magnetisation against the temperature it was evident that a phase transition occurs from para-magnetism to ferromagnetism at the critical temperature $T_c$. We also learnt that close to the critical temperature the correlation length approaches infinity through the observable in question will depend on the size of the system. Whereas at $T << T_c$ or $T >> T_c$ the correlation length approaches zero and the observable not longer depends system size.
Looking at the susceptibility showed similar results. Peaks were seen close to the critical temperature, with peaks from larger system sizes appearing closer to $T_c$ than smaller system sizes. A plot of the binder ratios served to be an effective method to finding the critical temperature due to the size independence of the binder ratios at the critical temperature.

On the other hand, it was also made clear that the simple implementation of the Ising model using the Monte Carlo method done in this project and the operating systems which were used to run the scripts limited further extensive research to be done. Only a few observables can be investigated at once and only

so much accuracy can be achieved because of the inefficient organisation of the script making the use of large Monte Carlo steps and large system sizes too time consuming to run.

# References


[1] Examples of Phase Transitions - https://web.mit.edu/8.334/www/grades/projects/projects10/AlexanderPapageorge/Page4.html

[2] Stine, R. (1989). An Introduction to Bootstrap Methods: Examples and Ideas. Sociological Methods & Research, 18(2–3), 243–291. https://doi.org/10.1177/0049124189018002003.

[3] Efron, B. (1979). Bootstrap Methods: Another Look at the Jackknife. Ann. Statist. 7, no. 1, 1–26. doi:10.1214/aos/1176344552. https://projecteuclid.org/euclid.aos/1176344552.

[4] Chernick, M. (2008). Bootstrap Methods : A Guide for Practitioners and Researchers. 2nd ed. Newtown, PA.

[5] Sandvik, A. (2018). Monte Carlo simulations in classical statistical physics. Boston University. p.13.

[6] Garcia-Ojalvo, J., Sancho, JM. (1999). Noise in Spatially Extended Systems. Berlin, Germany: Springer.

[7] Miyashita, S. (2010). Phase transition in spin systems with various types of fluctuations. Proceedings of the Japan Academy. Series B, Physical and biological sciences, 86(7), 643–666. https://doi.org/10.2183/pjab.86.643

[8] Binder, K. (1981). Critical Properties from Monte Carlo Coarse Graining and Renormalization. Phys. Rev. Lett. doi:10.1103/PhysRevLett.47.693.

[9] L, Onsager. (1944). Crystal statistics. I. A two dimensional model with an order-disorder transition. Phys. Rev. 65, 117-149.


# Appendix Code

```
import math
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.cm as cm

def spin_flip(d, J, M, T, L, nlattice):
        # Choose a lattice point at random
        dim = [0]*d
        for i in range(d):
                dim[i] = np.random.randint(0,L)
        x = dim[0]
        y = dim[1]

        # Calculate dE
        dE = ( 2*J*nlattice[y][x]*(nlattice[(y+1)%L][x%L]
                                +nlattice[(y-1)%L][x%L]
                                +nlattice[y%L][(x+1)%L]
                                +nlattice[y%L][(x-1)%L]) )
        # Calculate acceptance and attempt flip
        R = math.exp(-dE/T)
        if R >= 1:
                lattice[y][x] *= -1
```

```
                M += 2.*nlattice[y][x]/(L*L)

        elif R > np.random.random():
                lattice[y][x] *= -1
                M += 2.*nlattice[y][x]/(L*L)

        return M, nlattice

## Creates a block and its corresponding block sizes and no. and counters
def create_blocks():
        Z = 10 # Size of blocks
        Nb = 100 # No. of blocks
        block = 0 # block term counter
        zcounter = -1 # term inside the block counter
        blocks = [[0 for val in range(Z)] for bloc in range(Nb)]
        return blocks, Z, Nb, block, zcounter


## Updates block array ##
def update_blocks(M, Mblocks, Z, Nb, zcounter, block):
        zcounter += 1

        Mblocks[block][zcounter] = M

        if zcounter == Z-1:
                block += 1
                zcounter = -1

        if block == Nb: # tests if array of blocks are full
                Mblocks, block, Z = move_values(Mblocks, block, Z, Nb)

        return Mblocks, zcounter, block, Z

## Concatenates pairs of block to create space for new measurements ##
def move_values(Mblocks, block, Z, Nb):
        Z *= 2
        new_Mblocks = [[0 for i in range(Z)] for j in range(Nb)]
        for n in range(Nb/2):
                new_Mblocks[n] = Mblocks[2*n] + Mblocks[2*n+1]
        block = 50
        return new_Mblocks, block, Z

## Calculated the error for a given block array
def stdev(Mblocks, Nb, Z):
        blockarray = []
        for i in range(len(Mblocks)):
                blockarray.append(sum(Mblocks[i])/Z)
        blockarray = [x for x in blockarray if x != 0]
        avgblock = sum(blockarray)/len(blockarray)
        block_minus_average = []
        for i in range(len(blockarray)):
                a = blockarray[i] - avgblock
                block_minus_average.append(a*a)
        std = (1./Nb) * math.sqrt(sum(block_minus_average))
```

```
        return std

"""Start of code, initialize variables"""

d = 2 # Lattice Dimensions
Llist = [4,8,16,32] # List of lattice side lengths
J = 1 # Energy/Temperature ratio
N = 2000000 # Number of iterations
M = 0.
# The following are lists which will lists of measurements of observables
# and their errors of varies system sizes
M_plots = []
stdM_plots = []
binder_plots = []
stdbinder_plots = []
chi_plots = []
stdchi_plots = []
for L in Llist:
        T_list = [] # List containing the temperatures
        avgM_list = [] # List containing the average magnetisations
        avgstd_list = [] # List containing the average standard deviations
        binder_list = [] # binder ratios
        stdbinder_list = [] # binder ratios error
        chi_list = [] # Magnetic susceptibility
        stdchi_list = [] # susceptibility error

        # for subsequent plotting of the spin glass
        fig, ax = plt.subplots(nrows=1, ncols=1, figsize=(20, 12))


        ### INITIALIZING LATTICE AND ARRAY OF BLOCKS ###
        # Make a LxL array all zeros
        inlattice = [[0 for x in range(L)] for y in range(L)]
        for y in range(L): # Initialize the spin configuration lattice
                for x in range(L):
                        inlattice[y][x] = int(np.random.randint(0,2) * 2 - 1)

        for T in np.arange(1.5, 4.5, 0.1):
                dE = 0. # change in energy
                R = 0. # acceptance ratio
                M = 0. # Total magnetisation
                Mlist = [] # List of magnetisation values
                M2list = [] # List of magnetisation squared values

                lattice = inlattice

                for y in range(L): # Initialize the spin configuration lattice
                        for x in range(L):
                                M += float(lattice[y][x])/(L*L)

                # Mblocks contains all the magnetisation measurements stored in a 2d
                # array, where each row contains a block, and the
                # values in the row are the measurements
                Mblocks, Z, Nb, block, zcounter = create_blocks()
```

```
        # M2blocks will contain the square of the magnetisations,
        # it has its own set of counters and block sizes
        M2blocks, Z2, Nb2, block2, zcounter2 = create_blocks()

        ### RUN ALGORITHM FOR N ITERATIONS ###
        for n in range(N):
                #print M
                M , lattice = spin_flip(d, J, M, T, L, lattice)
                Mlist.append(abs(M))
                M2 = M*M
                M2list.append(M2)

                Mblocks, zcounter, block, Z = update_blocks(M, Mblocks,
                Z, Nb, zcounter, block)
                M2blocks, zcounter2, block2, Z2 = update_blocks(M2, M2blocks,
                Z2, Nb2, zcounter2, block2)

        # Plots lattice configuration
        if L == 32:
                fig.clf()
                ax = fig.add_subplot(1, 1, 1)
                im_content = ax.imshow(lattice, cmap=cm.afmhot,
                interpolation='nearest')
                fig_name = 'B' + str(round(T,2)) + '.png'
                plt.savefig(fig_name, bbox_inches='tight')

        # magnetisations
        avgM = sum(Mlist)/len(Mlist)
        avgM2 = sum(M2list)/len(M2list)
        avgM3 = avgM*avgM
        avgstd = stdev(Mblocks, Nb, Z)
        stdM2 = stdev(M2blocks, Nb2, Z2)
        stdM3 = avgM3*2*(avgstd/avgM)

        # Binder ratios
        binder =  (avgM2) / (avgM3)
        stdbinder = binder*math.sqrt( (stdM2/avgM2)**2 + (stdM3/avgM3)**2 )

        # Susceptibility
        chi = (L*L) * (1./T) * (avgM2 - avgM3)
        stdchi = ((L*L)) * (1./T) * math.sqrt( stdM2**2 + stdM3**2 )

        # Create list of measurements
        T_list.append(T)
        avgM_list.append(avgM)
        avgstd_list.append(avgstd)
        binder_list.append(binder)
        stdbinder_list.append(stdbinder)
        chi_list.append(chi)
        stdchi_list.append(stdchi)

# Creates list of lists of measurements
M_plots.append(avgM_list)
stdM_plots.append(avgstd_list)
```

```
        binder_plots.append(binder_list)
        stdbinder_plots.append(stdbinder_list)
        chi_plots.append(chi_list)
        stdchi_plots.append(stdchi_list)

# Removes the first element in each list of observables to fix the issue
# of thermalization at the start
for w in range(len(Llist)):
        M_plots[w].pop(0)
        stdM_plots[w].pop(0)
        binder_plots[w].pop(0)
        stdbinder_plots[w].pop(0)
        chi_plots[w].pop(0)
        stdchi_plots[w].pop(0)
T_list.pop(0)

### Plots ###
name = 'magnetisation'
name2 = 'Binder'
name3 = 'Susceptability'
plt.figure(name)
for s in range(len(Llist)):
        plt.errorbar(T_list, M_plots[s], stdM_plots[s], marker = 'x',
        capsize = 2, label = "System Size %s" %Llist[s])
        plt.legend(loc="best")
plt.xlabel('Temperature')
plt.ylabel('Average magnetisation')

plt.figure(name2)
for s in range(len(Llist)):
        plt.errorbar(T_list, binder_plots[s], stdbinder_plots[s], marker = 'x',
        capsize = 2, label = "System Size %s" %Llist[s])
        plt.legend(loc="best")
plt.xlabel('Temperature')
plt.ylabel('Binder Ratio')

plt.figure(name3)
for s in range(len(Llist)):
        plt.errorbar(T_list, chi_plots[s], stdchi_plots[s], marker = 'x',
        capsize = 2, label = "System Size %s" %Llist[s])
        plt.legend(loc="best")
plt.xlabel('Temperature')
plt.ylabel('Magnetic Susceptibility')
plt.show()
```