A Monte Carlo simulation of the Ising Model

Smit Chaudhary[1], Ignacio Fernández Graña[1], and Georgios Sotiropoulos[1]

[1]Delft University of Technology

May 16, 2021

Abstract

Ising Model is a well-known and simple model of ferromagnetism with utility in a variety of different fields. Although exact solutions are known for the 1D and the 2D models, computer simulations are still regarded as an useful tool to study the behaviour of the Model. In this work, two Monte Carlo simulation methods, the Metropolis algorithm and the Wolff algorithm, are employed to simulate a finite 2 dimensional Ising Model. A focus is made on the critical behaviour of 2D ising model is demonstrated and studied. To this end, two observables, namely the magnetisation and the energy of the resulting configurations, are studied. In addition, these values are used to further calculate the susceptibility and the specific heat of the system.

1 Introduction

The **Ising Model** is a simplified model of ferromagnetism frequently used in Statistical Mechanics. The system consists of a lattice with a spin located at each vertex. Each spin interacts with an external magnetic field and with its neighbouring spins and may only exist in one of the two possible states, spin up or spin down [1].

The Ising Model hamiltonian reads as follows:

$$H = -J \sum_{<i,j>} s_i s_j - B \sum_i s_i \tag{1}$$

where J is the nearest neighbour coupling and B the external magnetic field.

The Ising model has been exactly solved for 1 dimension by Ernst Ising himself in 1925 [2] and for 2 dimensions by Lars Onsager in 1944 [3]. It is known to undergo phase transition at some temperature T_C given by $\frac{kT_c}{J} = \frac{2}{\ln\left(1+\sqrt{2}\right)} \approx 2.269$ and exhibit critical behavior around the transition temperature when the lattice has a dimension greater than 1.

One way to study this model is to use **Monte Carlo simulations**. We shall employ two such algorithms for our paper: Metropolis and Wolff's algorithm.

Monte Carlo Methods or Monte Carlo experiments are a class of computational algorithms which rely on repeated random sampling to obtain numerical results. A system is evolved to a new state which is chosen from a randomly generated ensemble of future possible states. Then, using some predefined criteria, this new state is accepted or rejected with a certain probability [4].

One of the main types of Monte Carlo simulation is **Metropolis Monte Carlo**. In a Metropolis Monte Carlo simulation, a sequence of distributions of a system is generated in a so-called Markov Chain. The artificial dynamics forbid us from determining dynamical physical properties but we can use it to study the static properties of a system. In Metropolis algorithm a single spin of the lattice is updated in each step. There exist another type of Monte Carlo algorithms, called *cluster algorithms*, where a whole cluster of spins is updated in each step. **Wolff algorithm** is an example of the latter.

In this paper we attempt to characterize the behavior of the Ising model on a two-dimensional square lattice, by implementing both the Metropolis algorithm and the Wolff algorithm. Periodic boundary conditions are assumed and the simulations are run for various temperatures using both of these algorithms. The need for this dual approach stems from the aforementioned critical behavior of the Ising model, since the Metropolis algorithm fails to encapsulate the critical region accurately [5][4]. This inaccuracy is due to the large correlation times the Metropolis algorithm is faced with close to the critical temperature, thus needing very long simulation times in this regime. This effect is known as *critical slow down*, and can be avoided by using other algorithms such as Wolff algorithm[6].

This paper is structured as follows. In Section 2 we present the simulation methods we use and the physical quantities that we study. In section 3 our results for both algorithms are presented and discussed. We conclude our work in section 4.

2 Methods

2.1 Dimensionless Units

Throughout this work we will make use of a system of units known as *dimensionless units*. Any physical quantity can be converted to an equivalent dimensionless quantity by way of a scaling factor. Dimensionless units themselves can always be converted back to the real, physical units at any time. Thus no information is lost by using dimensionless units but there are some important advantages of working in dimensionless units. The primary advantage being we do not really need to handle unnecessary floating point numbers and suffer due to finite precision of the same. Another advantage being, we get an insight into the physics in terms of the length scales and other typical dimensions of the system.

In this dimensionless units we express the energies in units of k_bT and the spin s of a particle as the normalized value of their intrinsic magnetic moment, which has units $\frac{J}{T}$. Recall that in the Ising Model spins have only two possible values, so that s can only take values ± 1. Effectively this means that both the coupling coefficient J and the external magnetic field B will have energy dimensions in this dimensionless units. Once we define these quantities in dimensionless units, all quantities would be dimensionless with appropriate scaling factors. Hereafter, all quantities are being expressed in dimensionless units, unless otherwise specified.

2.2 Metropolis Monte Carlo

We take a further look at Metropolis algorithm for out square ising lattice.

Consider we start with a state that we denote by X. For the trial future state (which, let us denote by X'), we randomly choose from an ensemble of all those states which differ by only one spin. This is equivalent to choosing which one spin among all the spins in the lattice is different. In effect, there are L^2 possible future configurations X'. Thus, in practice, we randomly choose a spin from the lattice. Therefore, each spin in a L×L lattice has $\frac{1}{L^2}$ probability of being picked. Let $\Delta E(X \to X')$ be the energy difference between the new and the old state. If the trial state X' is such that $\Delta E(X \to X')$ is negative, thus if the energy has decreased, the system goes to the trial state X'. However, if the the energy increases, i.e, $\Delta E(X \to X')$ is positive, the trial state is accepted with probability $e^{-\beta \Delta E(X \to X')}$ This process is done iteratively. This algorithm satisfies the *detailed balance* condition, which ensures that the probability distribution the algorithm outputs will tend to the target (Boltzmann) distribution as the number of iterations grows ([5] presents a rigorous proof of this statement).

As is argued above, a spin is selected with probability $\frac{1}{L^2}$, thus, the average number of steps between two updates of the same spin is equal to L^2. Therefore, the average number of steps between two trial updates of the same spin is equal to L^2. Thus, the 'time' in a MC simulation is often expressed in units of *Monte Carlo Steps per spin* (MCS), 1 MCS being L^2 trials.

Metropolis algorithm is an excellent choice to study the behaviour of the lattice in the non-critical regime, that is, in temperatures well away from the critical temperature T_c. Nevertheless, close to the critical temperature the Metropolis algorithm is not a good choice as it presents some issues. The reason for this is that, close to the critical temperature, large clusters of predominantly up or down spin are formed, which contribute significantly to both the magnetisation m and the energy E. As the clusters flip from one orientation to another, they produce large fluctuations in m and E, which are often called **critical fluctuations** [5]. Since the average size of the clusters diverges when $T \to T_c$, the magnitude of the fluctuations does too. Since we always deal with finite-size lattice in our simulations, these magnitudes never really diverge, and they can become very large. Given that the specific heat and the susceptibility are computed from the energy and magnetisation respectively, we expect divergences in these observables as well.

These critical fluctuations are a characterizing feature of critical phase transitions, and they are therefore expected. The bad performance of the Metropolis algorithm in the critical region comes in combination with another intrinsic problem of the algorithm: the correlation time τ of the simulation is also large in the region close to T_c. This effect is known as *critical slowing down*. This means that, in order to get low statistical errors in the critical regime, the simulation times need to be very long in order to get enough independent measurements.

Therefore, if we are interested in the study of the critical behaviour of the lattice, Metropolis algorithm will not correctly capture the behaviour in this regime. To this end, many other more advanced algorithms have been developed which fix the aforementioned issues. In this work we have implemented **Wolff algorithm**, which will be reviewed in the next section.

2.3 Wolff Algorithm

The Wolff algorithm [6], also called Single-Cluster algorithm, belongs to a class of Monte Carlo algorithms called *cluster algorithms.* In Metropolis algorithm, only a single spin is updated at each step. In a cluster algorithm, a whole cluster of spins is flipped in each move instead of a single one. Another example of a cluster algorithms is the famous *Swendsen–Wang* algorithm[7], upon which Wolff algorithm is based. Both algorithms solve the issue of critical slow down present in Metropolis algorithm, as the correlation time in the critical region is reduced to a much lower value and therefore independent configurations can be generated at a much faster rate, eliminating the need for long simulation times. Nevertheless, this advantage only holds for this critical region, making Wolff algorithm less efficient than Metropolis both in the low and high temperature regimes[6]. Therefore, we will use Wolff algorithm to study the critical region in which Metropolis performs poorly. Wolff algorithm is also known to fulfil the detailed balanced condition [5].

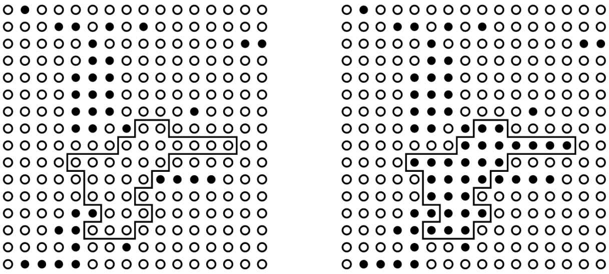


Figure 1: Flipping of a cluster in a simulation with the Wolff algorithm. Solid and open points represent up and down spins respectively, and points enclosed by the black line represent the cluster. All the spins belonging to cluster are flipped. Image taken from [6]

In Wolff algorithm a single cluster is formed at each step, and all the spins that are part of the cluster are *always flipped.* A cluster is constructed as follows:

1. A spin i is selected at random.
2. All nearest neighbours j of this spin are added to the cluster with probability $p = 1 - e^{-2\beta J}$, provided spins i and j have the same orientation and that the bond between i and j has not been considered before. Once a spin is added to the cluster, it is automatically inverted.
3. Once a spin j has been added to the cluster, all the neighbours of j are again considered and added to the cluster under the conditions stated in the previous points
4. Steps 2) and 3) are repeated iteratively until no more spins are added to the cluster.

When referring to the Wolff's algorithm, 1 Monte Carlo step corresponds to creating a single cluster and flipping the spins therein.

2.4 Observables

In order to study the behaviour of the system, we will measure two different observables: the magnetisation per spin m and the energy E. Through these observables we can then derive the specific heat c and the susceptibility χ. Observables are measured by taking the time average of some quantities over a significant number of time steps while the system is in equilibrium. For a quantity A, this time average is computed as:

$$\langle A \rangle = \frac{1}{n - n_0} \sum_{\nu > n_0}^{n} A_\nu \tag{2}$$

where n_0 is the number of time steps needed to get to equilibrium and n the total number of simulation step. From now on, $\langle \rangle$ will always refer to temporal averages.

Suppose we have a $L \times L$ lattice with $L^2 = N$ spins. The energy E of a specific configuration of the lattice is then easily computed with equation (1) as:

$$E = -J \sum_{<i,j>} s_i s_j - B \sum_{i=1}^{N} s_i \tag{3}$$

where the first sum is taken over all nearest neighbours of the spins in the lattice.
The magnetisation m of a configuration is defined as the average spin over the whole lattice:

$$m = \frac{1}{N} \sum_{i=1}^{N} s_i \tag{4}$$

Sometimes it is useful to also compute the average of the absolute of the magnetisation per spin $\langle |m| \rangle = \frac{1}{N} \langle |\sum_i s_i| \rangle$.

These two observables, energy and magnetisation, allow us to compute two other quantities: the specific heat and the magnetic susceptibility. Firstly, the specific heat c can be calculated as

$$c = \frac{k_B \beta^2}{N} \left(\langle E^2 \rangle - \langle E \rangle^2 \right) \tag{5}$$

Where $\beta = (k_b T)^{-1}$ or, in dimensionless units, $\beta = (T)^{-1}$. On the other hand, the magnetic susceptibility, or susceptibility, is defined as

$$\chi = \beta N \left(\langle m^2 \rangle - \langle m \rangle^2 \right) \tag{6}$$

In order to measure the error of the energy and the magnetisation, we make use of the Pearson correlation coefficient [8]. For a general observable A, measured in a finite time, this reads:

$$\chi_A(t) = \frac{(N-t)\sum_n A_n A_{n+t} - \sum_n A_n \times \sum_n A_{n+t}}{\sqrt{(N-t)\sum_n A_n^2 - \left(\sum_n A_n\right)^2} \sqrt{(N-t)\sum_n A_{n+t}^2 - \left(\sum_n A_{n+t}\right)^2}} \tag{7}$$

for $1 \leq n \leq N - t$, with N being the number of simulation steps and t the current simulation step. If we fit $\chi_A(t)$ to an exponential $e^{-\frac{t}{\tau}}$, being τ the correlation time, then the error will be given by:

$$\sigma_A = \sqrt{\frac{2\tau}{N} \left(\langle A^2 \rangle - \langle A \rangle^2 \right)} \tag{8}$$

The above expression allows us to compute the error of the energy $\sigma_{\langle E \rangle}$ and the magnetisation $\sigma_{\langle m \rangle}$. We can then calculate the errors of the specific heat and susceptibility via the Propagation of Error law, which states that the error of a magnitude $A(\vec{x})$ that depends on $\vec{x}$ parameters is:

$$\sigma_A^2 = \sum_i \left(\frac{\partial A}{\partial x_i} \right)^2 (\sigma_{xi})^2 \tag{9}$$

where σ_{xi} is the error associated with the x_i parameter of A. Therefore, the errors of the specific heat and magnetisation are given by:

$$\sigma_c = \frac{k_B \beta^2}{N} \sqrt{\left(\sigma_{\langle E^2 \rangle}\right)^2 + \left(2\langle E \rangle \sigma_{\langle E \rangle}\right)^2} \tag{10}$$

$$\sigma\chi = \beta N \sqrt{\left(\sigma_{\langle m^2 \rangle}\right)^2 + \left(2\langle m \rangle \sigma_{\langle m \rangle}\right)^2} \tag{11}$$

2.5 System Parameters and finite size

Equipped with the method to simulate the evolution of the system, we want to study the system under different conditions. Our control parameters are going to be the temperature of the system T, the strength of the external magnetic field B and the strength of the spin-spin coupling in the system J. The size of the lattice and the number of MC steps can be considered hyperparameters of the simulation. Unless otherwise specified, we consider the lattice to be a 20×20 square lattice.

Onsager's solution [3] was found for $L \to \infty$, but we simulate a finite sized lattice. This could potentially mean that the results do not match the theoretical known values. It is known that the *experimental* critical point of the numerical simulations for a finite lattice is at a temperature higher compared to the theoretical value of T_C as found by Onsager [4][9][3]. With increasing lattice size, various results approach the known theoretical value. For this reason, the results that we produce here differ from the theoretical results in both the location of the peak and the size of the peak for various observables (magnetic susceptibility, specific heat, etc). Fortunately, these deviations, at least in the location of the peak are rather small (0.3% for a 25×25 lattice)[10], and thus, we take our results to be a good approximation of the actual values. In terms of the size of the peak, at critical point, the quantities diverge for an infinite lattice, here for a finite sized lattice, we see a finite peak which we take as indicative of divergence for infinite lattice. Note that the critical point is determined by $\frac{T}{J}$ and not the (dimensionless) temperature alone but we take $J = 1$ in dimensionless units and express the critical point in terms of just the temperature for brevity.

3 Results

3.1 Correctness checks

Before proceeding to obtain results from our code, we make sure that our code passes some necessary sanity checks. In order to check the validity of our simulations, we check the results both in a a qualitative sense against our physical intuition and in quantitative sense against literature.

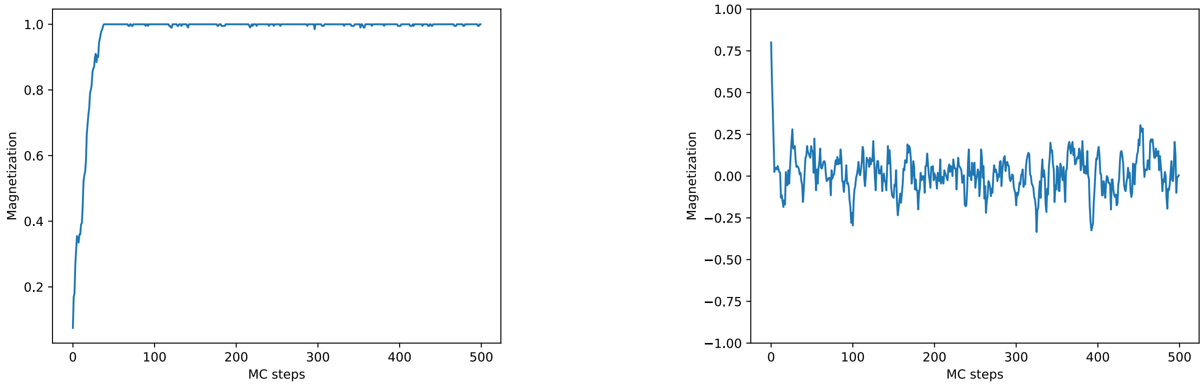
First of all, to assess the ability of the system to reach equilibrium we shall test the evolution in two different cases, which we will name case A and case B. The initial configuration and the temperature of the two cases are as follows:

- Case A will be initialized as a random lattice with a sub-critical temperature ($T < T_C$)
- Case B will be initialized as a all-spin-up lattice at a super-critical temperature ($T > T_C$)

In case A the temperature is considerably below the critical temperature, and therefore the spins rarely flip since the temperature is low. Additionally, we have no external magnetic field ($B = 0$) and $J = 1$, thus, the spins would tend to align themselves in one of the two orientations[1]. Thus, we expect the magnetisation m to be ± 1. It can be either of the two cases with equal probability and since $B = 0$, we can not choose which orientation it favours.

Case B is set up in a similar manner, with same values of J and B, but this time with the initial lattice being all spins up. Since we are beyond the critical temperature, we expect the spins to be random after sufficient time has passed. In that case, we expect the magnetisation m to be very close to zero[1].

Also, the different initial configurations are chosen such that the system is not readily in the final state, thus, it has to evolve and we can say with high degree of confidence that the results are correct.



(a) Case A. Magnetisation over time for $T = 1$, $J = 1$, and $B = 0$, all in dimensionless units

(b) Case B. Magnetisation over time for $T = 4$, $J = 1$, and $B = 0$, all in dimensionless units

Figure 2: Correctness check to verify expected time evolution of Metropolis algorithm

Figure 2a shows the evolution of the magnetisation for case A. As argued above, in this regime, the magnetisation would go to ± 1. In this run of simulation, we can see that the magnetisation indeed goes to $+1$ within some MC steps.

Similarly, the accompanying figure 2b shows the evolution of the magnetisation for case B. This result is in accordance to the theory as well. Even if we start with all spins aligned up, since the temperature is greater than the critical temperature, the spins align themselves randomly and the total magnetisation goes down. The spins keep fluctuating and this results in fluctuating magnetisation. Because the spins tend to be aligned with one another, we see small positive and negative values of magnetisation but the random fluctuations caused by the temperature prevents all spins aligning in either direction. For higher temperature, we would see smaller peaks for the fluctuations in magnetisation.

Additionally, one more sanity check one can seek is that, in presence of an external magnetic field, the spins have a preferred direction (parallel to the external field) and thus, the spins would align themselves with the external field. To this end we consider a case C where we simulate a system with external magnetic field $B = 1$, $J = 1$, and $T = 1.5$. We initialise the lattice with all spins down. We expect the system to evolve and all the spins to flip and be up instead (aligned with the external field).

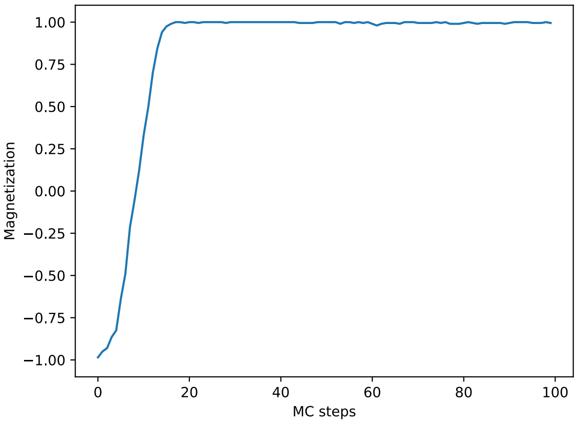


Figure 3: Case C. Magnetisation over time using Metropolis algorithm for $T = 1.5$, $J = 1$, and $B = 1$ on a spin-down initialized lattice, all in dimensionless units

Figure 3 shows the magnetisation over time for case C. As expected, we see that the spins flip and the magnetisation rises up to 1.

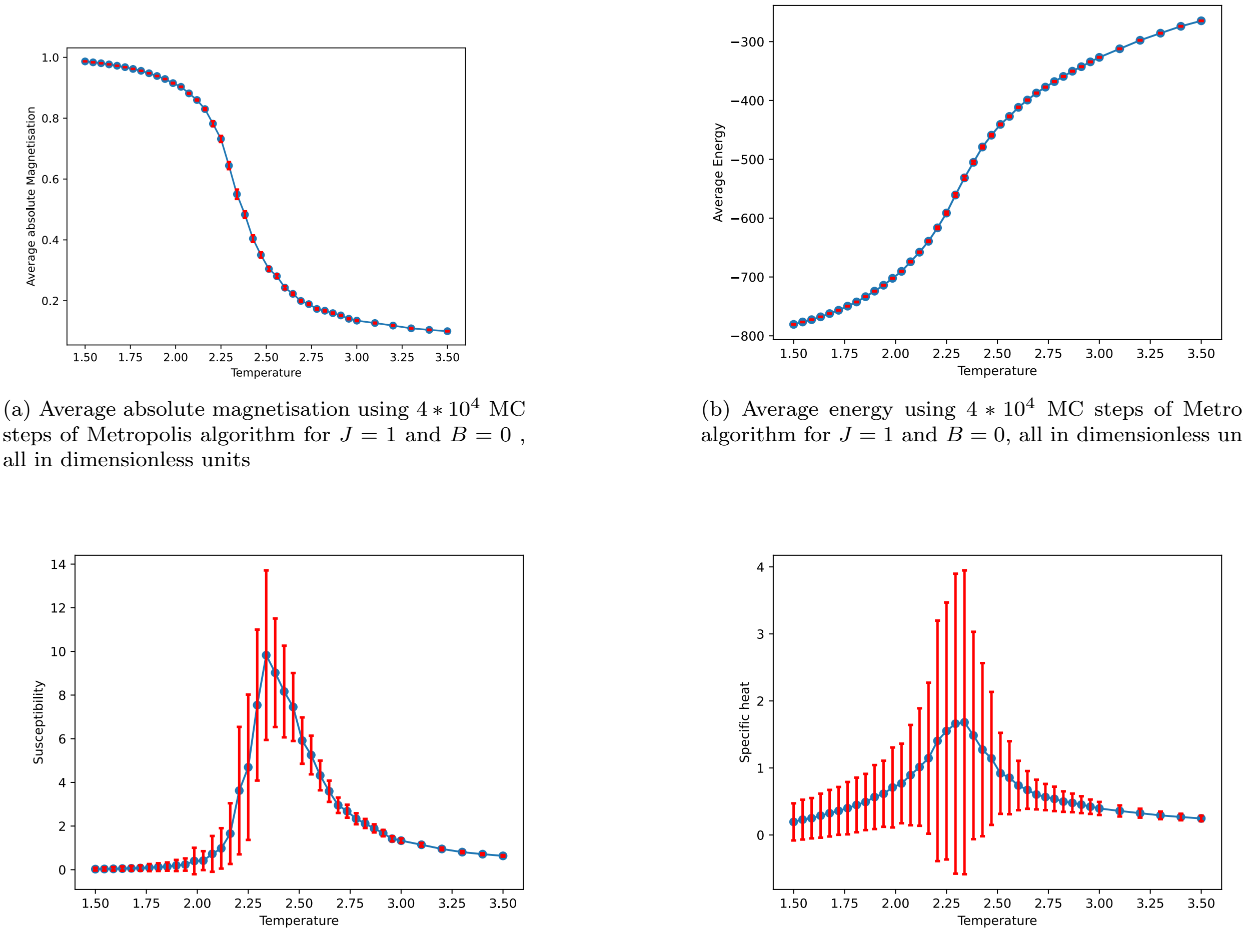
Although we do not present sanity checks for the Wolff algorithm in this section, we will later see that it produces results really close to what the Metropolis algorithm gives near the critical region but with much lower error rates. This ensures us that Wolff's algorithm also works as intended.

3.2 Observables

Having made sure that our code produces results that agree with the physical intuition behind the model, we proceed to study with it the system by running simulations for various temperatures. All the simulations are done with a 20×20 lattice for $J = 1$, $B = 0$ and temperatures ranging from 1.5 to 3.5. The total Monte Carlo steps are selected to be $4 * 10^4$. The results for magnetisation, energy, specific heat and susceptibility are presented for Metropolis algorithm in Figure 4 and for Wolff algorithm in Figure 5.

First we focus on the average absolute magnetisation. As argued above in section 3.1 for low temperatures, the spins tend to align themselves. Therefore the average absolute magnetisation will be 1. In Metropolis algorithm, this is the result of the low probability of flipping a single spin at low temperatures. As barely no spins are flipped, the majority of the spins in the lattice stay aligned. In Wolff algorithm, the size of the clusters for low temperatures are very large. In fact, for low enough temperatures the clusters encapsulate the whole lattice. This means that at each step every spin is inverted. Since we are taking the average of the *absolute* value of the magnetisation, this will be 1 for low temperatures. For high temperatures, the spins tend to be disordered, with random orientations that make the average absolute magnetisation 0. Indeed, the behaviour in both algorithms is as expected, as it can be seen in the figures 4a and 5a. One may note that, with the Metropolis algorithm, the points close to the critical region have bigger errors than with the Wolff algorithm. This showcases the fact that Wolff algorithm, for the same number of MC steps, achieves lower statistical errors in the critical region.

In line with the reasons presented above, one expects energy to rise as spins become more and more random. Thus, with higher temperatures, as absolute magnetisation goes down, the average energy goes up. The energy is plotted in figures 4b and 5b for Metropolis and Wolff algorithm respectively. As we are simulating a 20×20



(a) Average absolute magnetisation using $4 * 10^4$ MC steps of Metropolis algorithm for $J = 1$ and $B = 0$, all in dimensionless units

(b) Average energy using $4 * 10^4$ MC steps of Metropolis algorithm for $J = 1$ and $B = 0$, all in dimensionless units

(c) Susceptibility using $4 * 10^4$ MC steps of Metropolis algorithm for $J = 1$ and $B = 0$, all in dimensionless units

(d) Specific heat using $4 * 10^4$ MC steps of Metropolis algorithm for $J = 1$ and $B = 0$, all in dimensionless units

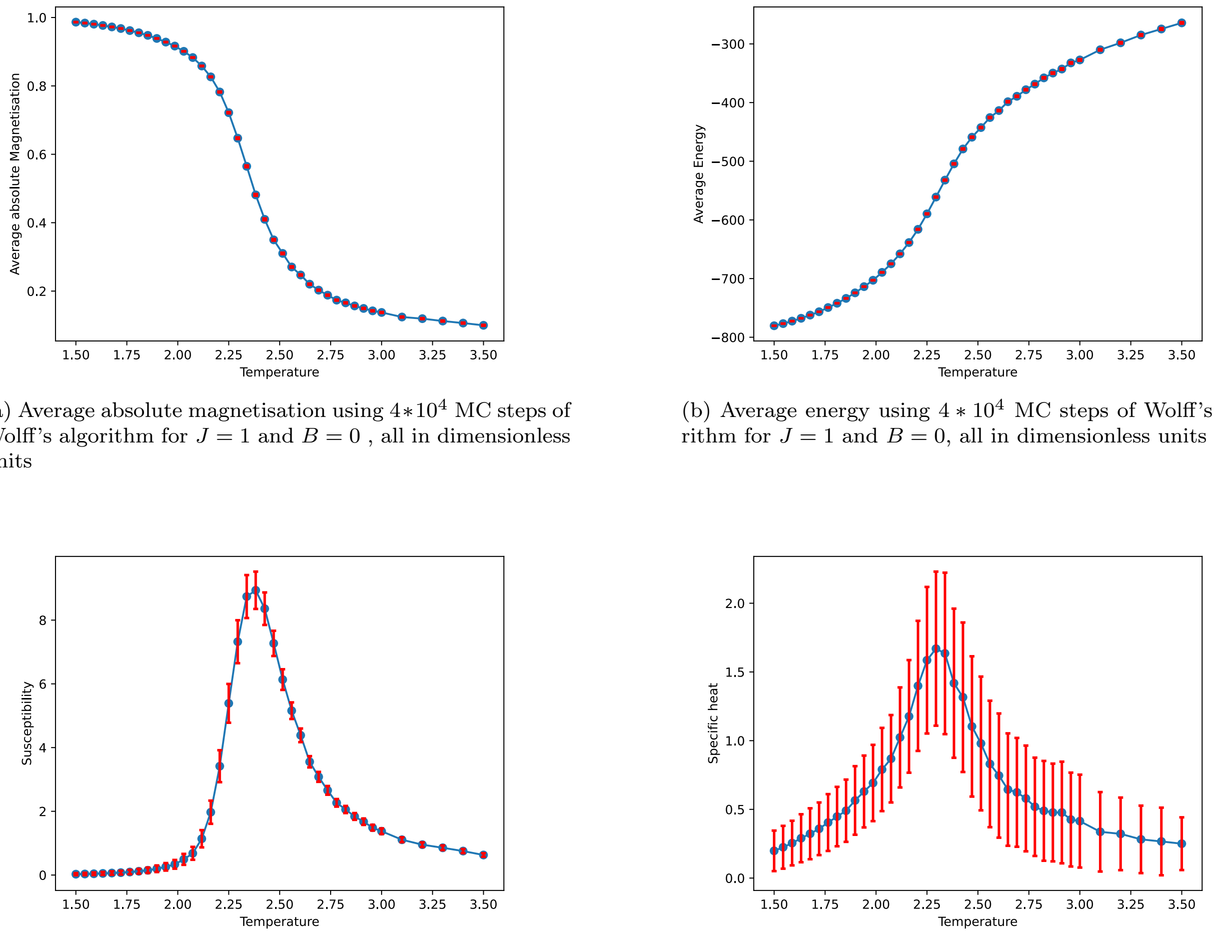
Figure 4: Results using the Metropolis algorithm

lattice for $J = 1$ and $B = 0$, the minimum energy the system can achieve is -800, according to equation 3. Indeed, one can see that for low temperatures the energy tends to that value.

Switching our focus to the susceptibility plots (Fig. 4c & 5c), we notice that the results for both algorithms yield results close to what is expected according to literature[9, 1]. The susceptibility peaks for a temperature of around $T \approx 2.35$ in dimensionless units with a maximum value of around $\chi \approx 10$, again expressed in dimensionless units. In work by Gebarowski[9], the susceptibility peaks that are obtained for lattices of $L = 16$ and $L = 32$ lie on both sides of our plot for $L = 20$. The susceptibility decreases exponentially in both cases when moving away from the critical temperature. However, Wolff's algorithm completely outperforms the Metropolis algorithm in terms of statistical error for the same number of MC steps in the critical region as it does not suffer from critical slow down.

The specific heat plots tell a consistent story as well, even though they exhibit higher error rates. For both algorithms, the peaks are found around the previously identified critical temperature $T \approx 2.35$ and they have a value of around $c \approx 1.7$ in dimensionless units. The specific heat also decreases exponentially when moving away from the critical temperature. This aligns with the results presented in [4] both in terms of critical temperature and values for the specific heat. Comparing the errors we understand the aforementioned need for the dual algorithmic approach of this paper. The error rates outside the critical region of the Metropolis algorithm are noticeably smaller than the ones that the Wolff algorithm produces, especially for higher temperatures. The latter is intuitively expected since the cluster size becomes quite small for high temperatures and thus our raw data is highly correlated, producing high error rates. In the critical region, Wolff outperforms the Metropolis algorithm, as it was extensively argued in section 2.2 and 2.3.

As argued above in section 2.5 and as shown in literature [9], one expect the finite-sized system to exhibit



(a) Average absolute magnetisation using $4 * 10^4$ MC steps of Wolff's algorithm for $J = 1$ and $B = 0$, all in dimensionless units

(b) Average energy using $4 * 10^4$ MC steps of Wolff's algorithm for $J = 1$ and $B = 0$, all in dimensionless units

(c) Susceptibility using $4 * 10^4$ MC steps of Wolff's algorithm for $J = 1$ and $B = 0$, all in dimensionless units

(d) Specific heat using $4 * 10^4$ MC steps of Wolff's algorithm for $J = 1$ and $B = 0$, all in dimensionless units

Figure 5: Results using the Wolff's algorithm

critical behaviour at higher temperature than the theoretical value. For the system parameters, $J = 1, B = 0$, the theoretical value would be $T_C^* \approx 2.269$. Our system of size 20×20 exhibits critical behaviour around $T_C \approx 2.35$, thus in accordance with the literature.

3.3 Performance of the code

Running Monte Carlo simulations can be quite computationally expensive. Consequently, optimization and wise allocation of resources becomes vital, especially when attempting to simulate large lattices over a large amount of MC steps.

In this kind of simulations, very large arrays of data may be obtained, rendering the calculation of the errors almost impossible at times. Doing a `for loop` is highly inefficient in this case. A more convenient way to carry out these operations is to vectorize them and use built-in libraries such as `NumPy`[11]. Numpy's main object, `numpy.arrays`, allows for a very efficient implementation of operations between arrays. Thus, we used `numpy.arrays` wherever applicable.

As a reference point, our laptops equipped with 16 GB of RAM and an AMD Ryzen 7 4800H CPU(8 Cores, 2.9GHz) were able to obtain results with errors for 40 different temperatures in around ≈ 2 hours for a 20x20 lattice and $4 * 10^4$ MC steps using either of the two algorithms. Given that this amount of MC steps is considered enough[4], we deem that our code performs sufficiently well.

4 Conclusions

In this work we have studied and implemented two Monte Carlo simulation methods for finite size Ising Model lattices. The two implemented algorithms are the Metropolis algorithm, which flips a single spin at each step, and the more advanced Wolff algorithm, which simulates the evolution of the system by forming a single cluster of spins in each step and inverting all the spins belonging to the cluster. The correctness of our implementation was checked by simulating the system in conditions where the underlining physical intuition was easy to grasp. Then, four observables were studied for both algorithms: the average absolute magnetisation, the energy, the specific heat and the magnetic susceptibility. Through these observables, the behaviour of the system was studied in depth, with a focus on the criticality effects. Furthermore, the study of these quantities allowed us to compare the efficiency of both algorithms, and to showcase the regions where they perform the better. In fact, the main downside of the Metropolis algorithm, critical slow down, is indeed solved by the Wolff algorithm, making Wolff a better candidate to simulate the system close to criticality. The finite-size effects of the simulation were commented and compared to the literature. Finally, we reflected about the efficiency of our code.

As a future outlook, our current implementation could be improved in several ways. First of all, increasing the number of MC steps on better hardware would be able to yield lower statistical errors, thus providing more accurate results for the observables in the same timeframe. Secondly, even though Python is a very friendly-user programming language, it is not ideal for resource-intensive computations like this one. Lower level languages like C++, C or even Julia offer the possibility of implementing very efficient simulations. Lastly, more sophisticated methods for Monte Carlo simulations could be used to improve the performance.

References

[1] Jos Thijssen. *Lecture Notes Statistical Physics.* 2018.

[2] Ernst Ising. "*Beitrag zur Theorie des Ferromagnetismus*". In: *Zeitschrift für Physik* 31.1 (Feb. 1925), pp. 253–258. ISSN: 0044-3328. DOI: 10.1007/BF02980577. URL: https://doi.org/10.1007/BF02980577.

[3] Lars Onsager. "*Crystal Statistics. I. A Two-Dimensional Model with an Order-Disorder Transition*". In: *Phys. Rev.* 65 (3-4 Feb. 1944), pp. 117–149. DOI: 10.1103/PhysRev.65.117. URL: https://link.aps.org/doi/10.1103/PhysRev.65.117.

[4] Jos Thijssen. *Computational Physics.* 2nd ed. Cambridge University Press, 2007. DOI: 10.1017/CBO9781139171397.

[5] M. E. J. Newman and G. T. Barkeman. *Monte Carlo Methods in Statistical Physics.* 1st ed. Oxford University Press, 1999.

[6] Ulli Wolff. "*Collective Monte Carlo Updating for Spin Systems*". In: *Phys. Rev. Lett.* 62 (4 Jan. 1989), pp. 361–364. DOI: 10.1103/PhysRevLett.62.361. URL: https://link.aps.org/doi/10.1103/PhysRevLett.62.361.

[7] Robert H. Swendsen and Jian-Sheng Wang. "*Nonuniversal critical dynamics in Monte Carlo simulations*". In: 58.2 (Jan. 1987), pp. 86–88. DOI: 10.1103/PhysRevLett.58.86.

[8] Rand R. Wilcox. *Introduction to robust estimation and hypothesis testing.* 2nd ed. Academic Press, 2005.

[9] Robert Gebarowski. "*Monte Carlo simulations of the Ising model on a square lattice with random Gaussian interactions*". In: 2015.

[10] M. Yu. Malsagov, I. M. Karandashev, and B. V. Kryzhanovsky. *The Analytical Expressions for a Finite-Size 2D Ising Model.* 2017. arXiv: 1706.02541 [cond-mat.dis-nn].

[11] Charles R. Harris et al. "*Array programming with NumPy*". In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.