Computational Physics 3500K Project: The Ising Model

Casey Hampson

Spring 2024

Introduction to Magnetic Materials

Inside of a material, individual atoms can be approximated as magnetic dipoles due to the motion of the electron orbit causing a current, roughly speaking. Additionally, free electrons act as dipoles due to their spin orientation. Because of this, these materials *magnetize* when exposed to an external magnetic field, by which its constituent dipoles align themselves with the direction of the field. Materials that magnetize parallel to the external field are called **paramagnets**, and materials that magnetize anti-parallel to the external field are called **diamagnets**. There is a special subgroup of these called **ferromagnets**, in which the magnetization effects persist inside the material when the external field is turned off, leading to more interesting properties, such as the creation of permanent magnets. In the case of ferromagnetic materials, which are the heart of the Ising Model, we consider the spin alignment of electrons within the material as the dipoles contributing to its net magnetization.

As is the case with any large system comprised of many small parts, we will need to make some approximations and rely on some statistical mechanics. In real ferromagnetic materials, there are **domains** - small regions within the material - in which the electrons are all largely aligned along one axis, and it almost never corresponds with neighboring domains' axes. This causes an ordinary hunk of metal to not be a magnet on its own; the domains all cancel each other out, leaving no net magnetization. Without too much loss of generality, we will assume there are no domains in our materials; in other words, the preferred spin axis of all domains are identical. Additionally, a particular dipole's spin would ordinarily have an effect on a number of dipoles within its vicinity (subject to some quantum mechanical properties that are beyond the scope of this paper), but we will assume that any long-range effects are neglible, and it can only affect its nearest four neighbors on a two-dimensional lattice. These assumptions, taken together, form the basis of the **Ising model**, a well studied, rough approximation to how a ferromagnetic material behaves.

The exact solution of the Ising Model in two dimensions was solved in 1944 by Lars Onsager [1] in the limit of an infinite lattice with no external magnetic field. He solved for the total free energy analytically, and from this, other quantities can be determined through appropriate derivitives in the regime of statistical mechanics. One of the most significant results that was found was that there exists a continuous phase transition at a critical temperature $T_C \approx 2.269$ (dimensionless units), where at temperatures larger than this, the overall magnetization of the material goes to zero as random thermal fluctuations destory any mutual dipole alignment. This critical temperature is called the **Curie Temperature**, and it is one of the quantities that we will attempt to confirm with our model.

The 3d Ising model has not yet been solved, and research suggests [2] that solving the partition function for a 3d model is NP-complete, meaning it cannot be solved in polynomial time. Hence, in the limit of an infinite lattice (which is what we aim to compute for these models), it becomes essentially impossible to compute. However, computational methods are still valid for approximating a critical temperature (if one exists). We will not explore them in this paper to computing limitations, as running a significantly large model to make statistically decent calculations would take far too long. However, see [3], for instance, for Monte Carlo simulation results in the case of a 3d lattice.

Statistical Mechanics

The nature of a particular configuration of an Ising model simulation is based on the temperature of the system. More accurately, it is the temperature of the surrounding "heat reservoir" that the contituent dipoles are reacting and remaining in thermal equilibrium with. Since we supply this temperature, the area of the lattice, and the number of particles, we are operating in the *canonical ensemble*, and can make use of some very useful quantities from statistical mechanics.[1]

For instance, the probability that the system is in some state s is given by

$$P(s) = \frac{1}{Z} e^{-\beta E(s)}, \tag{1}$$

where Z is the canonical partition function

[1] We don't attempt to derive any of the quantities or ideas presented in this section; we merely quote them. See, for instance, [4] or [5] for a more complete treatment on these topics.

$$Z = \sum_i e^{-\beta E_i}, \tag{2}$$

and $\beta = (k_B T)^{-1}$, where k_B is the Boltzmann constant and T is the temperature of the system.

In view of the Ising Model, we can its Hamiltonian to describe the energy, which contains information about interactions between dipoles and with an external magnetic field:

$$H = -J \sum_{\langle ij \rangle} s_i s_j - B \sum_i s_i \tag{3}$$

$$\approx -J \sum_{\langle ij \rangle} s_i s_j. \tag{4}$$

We will use the latter expression, where we have no external magnetic field B. J is a constant characterizing the strength of the interaction between any two dipoles, and the minus sign is convention to allow J to be positive in the case of a ferromagnet. We will choose to set $J = 1$ (or, alternatively, consider the dimensionless energy E/J). The notation $\langle ij \rangle$ signifies that we are summing over all possible *pairs* of dipoles, where s_i and s_j are the spins of those dipoles. We will use the values $+1$ for "spin-up" and -1 for "spin-down".

We are going to be interested in averages, or the expectation values, of quantities, and these can be found with

$$\langle X \rangle = \frac{1}{Z} \sum_\mu X_\mu e^{-\beta E_\mu}. \tag{5}$$

where the observable X has a value of X_μ when the system is in state μ.

The main averaged quantity we will be interested in is the average energy $\langle E \rangle$, otherwise known as the internal energy U:

$$U = \langle E \rangle = \frac{1}{Z} \sum_\mu E_\mu e^{-\beta E_\mu}. \tag{6}$$

However, in our case of a finite lattice, we can compute the total energy directly from (4), and computational methods that will be described shortly can make subsequent calculations trivial so that we don't have to do exponentials or deal with the partition function.

The next important quantity we can examine is the specific heat of the system, which is the amount of heat needed to raise its temperature by one degree celcius. This can be found by

$$C = k_B \beta^2 \left(\langle E^2 \rangle - \langle E \rangle^2 \right), \tag{7}$$

which is in terms of the fluctuation of the energy, the term in parantheses. There are other relations in terms of the derivative of the partition function, but these are more computationally accessible for us. We are actually more interested in the specific heat per lattice site, denoted with a lowercase c, while also setting $k_B = 1$ for simplicity:

$$c = \frac{\beta^2}{N} \left(\langle E^2 \rangle - \langle E \rangle^2 \right) \tag{8}$$

The magnetization of a system is related to the Helmholtz free energy of the system as well as the magnetic field B:

$$M = \frac{\partial F}{\partial B}, \tag{9}$$

and is, of course, one of the most important quantities we aim to calculate with our model. It turns our, though, for the Ising model, we can formulate this quantity as simply the sum over all spins to avoid calculating this free energy (which is also in terms of the partition function).

$$M = \sum_i s_i, \tag{10}$$

from which follows the mean magnetization:

$$\langle M \rangle = \left\langle \sum_i s_i \right\rangle. \tag{11}$$

Further, just as with the specific heat, the more useful quantity is the mean magnetization per lattice site, denoted with a lowercase m:

$$\langle m \rangle = \frac{1}{N} \left\langle \sum_i s_i \right\rangle. \tag{12}$$

In a similar vein, the susceptibility χ of a particular parameter X to another parameter Y is given by

$$\chi \equiv \frac{\partial \langle X \rangle}{\partial Y}. \tag{13}$$

We will be interested in the magnetic susceptibility, which is the susceptibility of the magnetization due to changes in a magnetic field, and can similarly be given in terms of the fluctuation of the magnetic field:

$$\chi_m = \frac{\partial \langle M \rangle}{\partial B} = \beta \left(\langle M^2 \rangle - \langle M \rangle^2 \right). \tag{14}$$

Again, we consider the magnetic susceptibility per spin which we will denote with χ, dropping the subscript, which we can put in terms of the mean magnetization per spin:

$$\chi = \beta N \left(\langle m^2 \rangle - \langle m \rangle^2 \right). \tag{15}$$

These the main quantities we will attempt to determine ourselves and compare with existing data as well as the exact solution.

Computational Techniques

Motivation

The main issue with attempting to solve these quantities in terms of the partition function is that for any significantly large lattice - one that we would hope to gain something even remotely useful - the partition function would carry an unreasonable number of terms that would be impossible to calculate by brute force. To see this, we can formulate the partition function in terms of the Hamiltonian:

$$Z = \sum_{\{s_i\}} e^{-\beta H(s_i)}, \tag{16}$$

where the notation $\{s_i\}$ indicates that we are summing over every *state* of the system, rather than just the spins. In a relatively small system represented by a 10×10 lattice, this gives us 2^{100} possible states, which is on the order of $\sim 10^{30}$. If our computer could calculate a billion of these terms per second (faster than any average computer currently), it would take $\approx 10^{21}$ seconds, which is longer than the current age of the universe. Evidently, we cannot brute force the calculation of the partition function.[2]

Turning to something like a Monte Carlo algorithm is certainly appealing, and it would let us pretty simply calculate the quantities in the form we have listed in the previous section. However, implementing a simple Monte Carlo simulation that samples random states, computing energy differences, and calculates those quantities still doesn't work due to the sheer number of states in any modest system. Often, as the long-term behavior of the system tends toward equilibrium, it will be extraordinarily unlikely that our

[2]We did simplify many of the observables into quantities independent of the partition function, but its significance still remains: we need a stupendously large number of terms to be able to have physically valid results.

random sampling correctly identities any of these "correct" states, if at all, drastically reducing the accuracy of our calculations.

To remedy this, we can turn to **importance sampling**, which is the backbone of the Metropolis algorithm and will rely on some of the above machinery.

The Metropolis Algorithm (Monte Carlo with Importance Sampling)

As a modification of the the naive Monte Carlo method, we can use the Boltzmann factors themselves to help with our importance sampling to try and generate states that are based, statistically, on the Boltzmann distribution. One such algorithm that can achieve this is the **Metropolis algorithm** [6]. We give a quick overview of this method and its advantages in using it for the Ising model.

In general, an integral of some function can be computed with Monte Carlo methods by sampling random numbers in the domain of the integral. However, for many functions, it may be far more efficient and accurate to sample these random variables according to some probability density function. There are a number of simpler methods that can be used to achieve this (such as the von Neumann rejection method, or inversion of the weight function; see Chapter 8 of [7] for discussion on these methods), but they fail for more complex distributions.

The Metropolis algorithm is actually quite simple: it consists of generating some sample in the domain of the problem, then generating another (a "test" sample, more or less) based on the previous one. This new sample is then added to the total sample distribution based on the ratio of the weight functions of each sample. If this ratio is greater than or equal to 1, then we immediately add this test sample, and if it is less than one, then the ratio becomes the probability of adding it to the distribution anyway. After letting such a simulation run for a long time, we successfully retrieve a distribution based on our probability density function of choice.

How this will work for the ising model, is that we start with a distribution of dipoles in our lattice, then generate a new state differing only by the flipping of one dipole. We want to choose the ratio of weight functions to be that which grants us a distribution of states that is statistically compatible with Boltzmann statistics as we have motivated in the previous section. As such, we can choose our weight functions to be simply the probability of that state occuring, so this ratio is:

$$r = \frac{(1/N)e^{-\beta E_{n+1}}}{(1/N)e^{-\beta E_n}} = e^{-\beta \Delta E}, \tag{17}$$

where E_{n+1} is the energy of the system after a proposed flip of the dipole, and E_n is the current energy of the system, determined by the Hamiltonian in (4) (as we will describe shortly, this change in energy can be computed quite easily, since we are only flipping a single dipole). If this number is greater than 1 (if $\Delta E < 0$), we accept the flip, otherwise, we maybe accept it using r as the probability. This can be achieved computationally by simply generating a random number in the range $[0, 1]$, and if it is less than r we accept the flip, then pick another random point on our lattice and continue until each dipole has had a chance to flip hundreds or thousands of times. This is the entire algorithm!

To show that this method is valid, i.e. it produces states that are with compatible with Boltzmann statistics, let's consider two states that differ only in the flipping of one single dipole that have energies E_1 and E_2, such that $E_1 < E_2$. Now if the system is initially in state 1, it has a probability of transitioning to state 2 by:

$$\mathcal{P}(1 \to 2) = \frac{1}{N} e^{-\beta \Delta E}, \tag{18}$$

where the $1/N$ is to "pick the right dipole" out of the lattice, and the exponential term is given from our Metropolis algorithm definition. The probability for the inverse is simply $1/N$; since that is a lower energy state, we simply just need to pick the right dipole:

$$\mathcal{P}(2 \to 1) = \frac{1}{N}. \tag{19}$$

The ratio of these factors is therefore:

$$\frac{\mathcal{P}(1 \to 2)}{\mathcal{P}(2 \to 1)} = \frac{(1/N)e^{-\beta \Delta E}}{1/N} = \frac{e^{-\beta E_2}}{e^{-\beta E_1}}, \tag{20}$$

which is just the ratio of the corresponding Boltzmann factors of the two states. This is exactly what we want. Additionally, if our system goes through some intermediary state 3, we find that this still holds:

$$\frac{\mathcal{P}(1 \to 3 \to 2)}{\mathcal{P}(2 \to 3 \to 1)} = \frac{e^{-\beta E_3} e^{-\beta E_4} e^{-\beta E_2}}{e^{-\beta E_1} e^{-\beta E_3} e^{-\beta E_4}} = \frac{e^{-\beta E_2}}{e^{-\beta E_1}}, \tag{21}$$

which again is what we expect. Any other lengths of intermediate states will always result in the algorithm choosing states that are statistically based on the Boltzmann distribution, meaning this algorithm will work very well for the Ising model.

Theoretically, though, there are a few more things we should take into account. An ordinary Metropolis algorithm would use a random walker, for instance, to walk through a lattice or whatever distribution one is trying to solve. Often, "adjacent" inputs are correlated, in a way, causing the resulting distribution of values to be skewed as subsequent inputs in the random walk are not statistically independent from one another. The way we circumvent this issue is to randomly select the points in our lattice each iteration. Since our Hamiltonian is only concerened with the four nearest neighbors of a particular dipole, in a large enough lattice, we will almost never choose an adjacent point such that the two are statistically dependent.

Now, we can utilize a number of optimizations in our computer program in order to run these simulations faster. As we mentioned earlier, the energy calculation can be dramatically simplified by considering that the two states differ only be the exchange of one dipole. In fact, we only need to compute the energy using the Hamiltonian once at the beginning of a simulation. From there, we can compute the change in energy directly:

$$E_{n+1} - E_n = \Delta E = -(s_i^{n+1} - s_i^n) \sum_{k \text{ n.n. } i} s_k, \tag{22}$$

where the sum over k is for the four nearest neighbors to the ith dipole that we are considering switching, and $s_k^{n+1} = s_k^n = s_k$. Further, since the spins are always ± 1, the expression in parentheses is ± 2, based on the original spin s_i^n. So, $s_i^n - s_i^{n+1} = 2s_i^n$, so we have that

$$\Delta E = 2s_i^n \sum_{k \text{ n.n. } i} s_k, \tag{23}$$

which is very easy and quick for our computer to calculate. So, after having computed the total energy once at the beginning, we can simply compute the above quantity and add it to the total energy, saving precious time.

The magnetization can be done the same way, and it is even simpler. The magnetization cares only about the spins themselves, so the change in magnetization from one state to another is simply

$$\Delta M = s_k^\nu - s_k^\mu = 2s_k^\nu, \tag{24}$$

where s_k^μ is the k-th dipole in state μ, and s_k^ν is the same dipole in the proposed state ν. So, just like with the energy, we can compute the total magnetization for the system at the beginning of the calculation and simply change it by the above amount every iteration. Although, the main magnatization calculation is actually quite simple, so if we wait for a long enough time between calculations (for instance, we may want to calculate it once every several full sweeps of a lattice), then it may not be much better to do the above calculation. Either way, computers are more than fast enough for this difference to not matter much.

There is one more major optimization that will dramatically reduce computation time, which is the exponential in our acceptance ratio. For the computer, this will require a number of float multiplications and additions, which are far more computationally expensive than any other calculation we make. Fortunately, for our 2d lattice, ΔU can only take on a small subset of values. Additionally, we don't care about half of them (plus 1) if they are negative (or zero), and so it turns out that there are only 2 unique values that this can take on: when the main dipole and three of its neighbors are pointing parallel to each other, or if the main dipole and all of its neighbors are pointing parallel. We can calculate these at the beginning of the simulation and store them in an array so that we don't need to calculate them during the simulation.

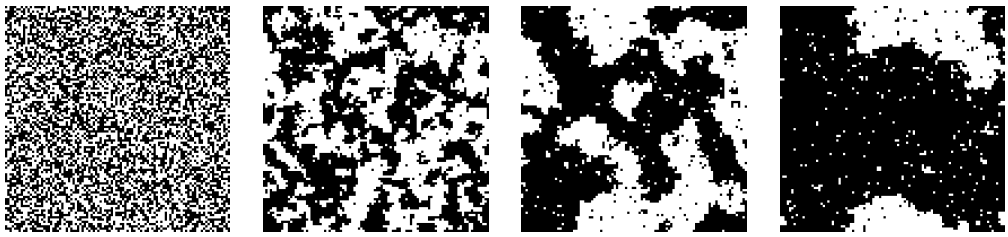

Figure 1: An Ising model simulation with $J = 1$, $T = 2.0$ (with $k_B = 1$) on a 100×100 lattice, started at a temperature of $T = \infty$, meaning all the spins were randomized. The images shown are snapshots from the visual simulation after $N = 0, 10, 100, 1000$ full sweeps of the lattice.

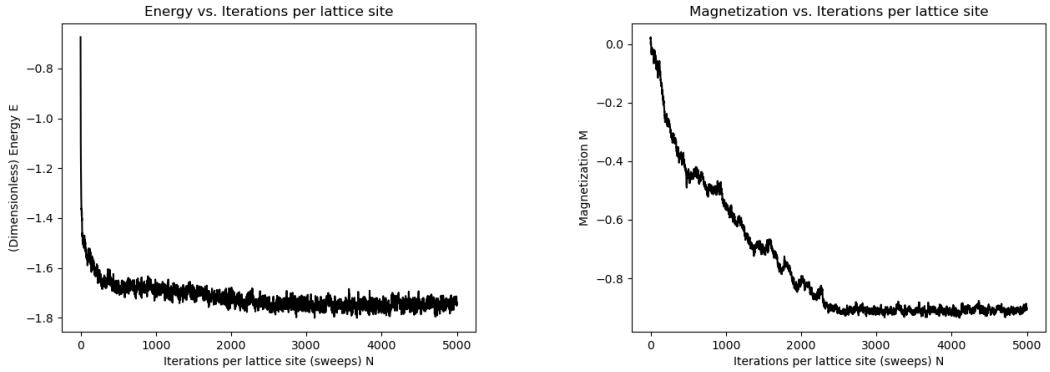

Figure 2: The energy and magnetization over the course of 5000 steps of the simulation ran with the same settings as in Fig. 1.

Minimal Working Program

We chose to program this in C++ for calculational speed, and we used the Raylib API for rendering the simulations, along with the open-source Matplotlibcpp to pipe plotting information directly to a Python interpreter equipped with Matplotlib. The code can be found in the GitHub repository: https://github.com/champso1/Ising_Model. To start, we programmed the visual component of the simulation, as well as some basic measurements of energies and magnetizations.

In Fig. 1, some results of the visual simulation are shown at different points in time. The white squares are spin up dipoles, and the black squares are spin down dipoles. It was ran given the initial condition of $T = \infty$, meaning all the spins are randomly orientied in the lattice. The equilibrium (dimensionless) temperature that the system was attempting to reach was $T = 2.0$. We notice that very quickly, domains of like-spin dipoles start forming, and by the 1000^{th} iteration, a large spin-down domain dominates the center of the lattice. However, we still notice the effects of the random thermal motions. Inside the big spin-down domain, we have a number of lone spin-up domains that resulted in a higher energy, but due to the relatively higher temperature, the ratio computed in the algorithm was actually not particularly low, so we had a decent chance of flipping it anyway.

Additionally, in Fig. 2, we have plotted some more long-term behavior of the simulation with the same settings. We notice that the energy drops very rapidly in the beginning of the simulation, then fluctuates heavily and slowly decreases until roughly $N = 3000$, where it continues fluctuating around a constant value. The magnitzation in general fluctuates a lot less, and decreases roughly constantly until the same $N = 3000$ time, where it remains constant afterwards.

Equilibrium Calculations

We need to know a few things before we can start calculating these quantities. The typical two temperatures we start any given simulation at are $T = 0$, where every dipole is oriented in one direction, or $T = \infty$, where

every dipole is oriented randomly. We could also start at a closer temperature, or even the exact temperature itself, however even in that case we still want to give every dipole a chance to flip numerous times. In any case, we need to give the system time to equilibrate, and intuitively, we expect for this to be different based on the lattice size.

We intend to run our simulations for only a handful of lattice sizes, and since a single run on a large lattice takes at most a few seconds, we can run a couple simulations and examine the total energy/magnetization and gauge, by eye, how many iterations it takes before they level off. This isn't the most rigorous way of doing this, but it works for the purposes of this paper. Running several iterations for a particular lattice size helps ensure that we don't end up at a *local* energy minimum as opposed to a *global* energy minimum. Based on Fig. 2, we can reasonably conclude that the equilibration time for a 100×100 lattice is roughly $N = 3000$ iterations per lattice cite.

Additionally, in any given simulation which has reached equilibrium, we want to average over a sufficiently long period of time in order to more accurately measure our observables. To determine this time period, we need a measure of how long it takes for one state to transition into another that is statistically independent of the first state. Of course, if we measure the magnetization after only one dipole flipping, it will be almost completely identical to the previous value Similar to before, such a state could perhaps be at the peak or trough of some fluctuation, meaning that even if we take the average over several steps, we aren't getting good values for the observable. To remedy this, we find the **correlation time** for the system, which is the amount of time it takes to transition from one state to another that is significantly different from the previous one.

We are specifically going to be looking at the *time-displaced autocorrelation function*, denoted by $\chi(t)$. Again considering the magnetization, this function is

$$\chi(t) = \int \left[m(t')m(t'+t) - \langle m \rangle^2 \right] \, \mathrm{d}t'. \tag{25}$$

After a time $t_{\max}$ in our simulation, we will have samples of $m(t)$ measured at evenly-spaced times, so we can use the discrete version of the above formula:

$$\chi(t) = \frac{1}{t_{\max} - t} \sum_{t'=0}^{t_{\max}-t} m(t')m(t'+t) - \frac{1}{t_{\max} - t} \sum_{t'=0}^{t_{\max}-t} m(t') \times \frac{1}{t_{\max} - t} \sum_{t'=0}^{t_{\max}-t} m(t'+t), \tag{26}$$

where the second term is a recalculation of the mean, rather than using what we have calculated before, in order to make this function a little cleaner.

For instance, for a simulation with the same settings as before, Fig. 3 shows the magnetization autocorrelation function for up to 600 iterations per lattice site. The correlation time is rate of exponential decay of the autocorrelation function, and it is denoted by τ in the literature. Because of this, we have

$$\frac{\chi(t)}{\chi(0)} = e^{-t/\tau}, \tag{27}$$

where we have normalized the autocorrelation function to have $\chi(0) = 1$. If we focus on the first few hundred values, then we will isolate the actual decay of the autocorrelation function before it tends to zero and error fluctuations take over. If we take the logarithm of the data set, we can fit a line to it and determine the slope to find the value of τ for the magnetization, which turns out to be roughly 100. This means that every 100 iterations, we expect to find a new state of our system that is statistically independent of the previous one. To play it safe, we will double this value, so that over an time interval of Δt, we can sample

$$n = \frac{\Delta t}{2\tau} \tag{28}$$

values to get an accurate estimate of our quantities.

Full Calculations

Our next step is to calculate these quantities over a range of temperatures in order to gauge what happens near the critical temperature. For instance, we will start the first simulation at a temperature of $T = 0.2$,

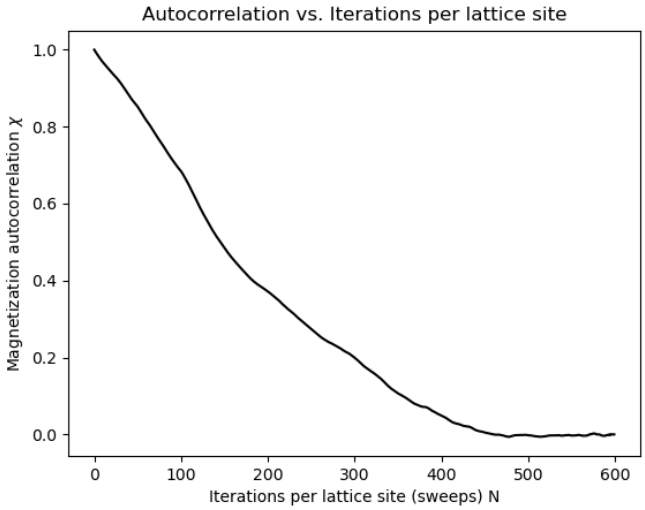

Figure 3: The magnetization autocorrelation function as the simulation progresses, using the same settings again as Fig 1.

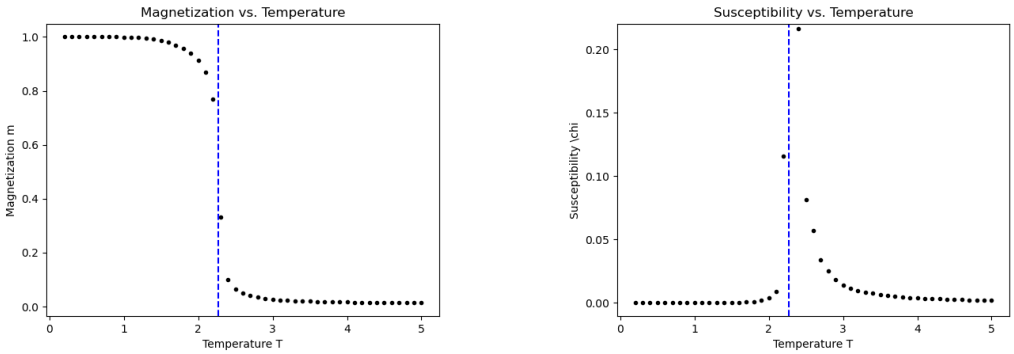

Figure 4: The magnetization and susceptibility of a 100×100 lattice computed over the temperature range $T = [0.2, 5.0]$ with a step size of $\Delta T = 0.1$. There is a vertical line drawn at the analytical critical point of $T \approx 2.269$.

then let it run to equilibrium, which for a 100×100 lattice we have already determined to be roughly 3000 iterations per lattice site. From there, we calculate, over an extended period of time, the average values of quantities, evenly spacing their samplings based on the correlation time to ensure that the calculations are mostly statistically independent. From the averages of the energies and magnetizations, we can then determine the specific heat as well as the susceptibility.

After doing this, we could reset the lattice back to a $T = \infty$ state, but this would be inefficient; we can instead leave the lattice alone. Since a lattice at a nearly identical temperature has just reached equilibrium, we have already done most of the work for the next lattice, it only needs to equilibrate a little before being ready to calculate values. So, we can start off with a relatively large number of iterations to reach equilibrium, then decrease the number of iterations in subsequent simulations to save some computational time.

In our simulation, we chose to run simulations from a very small temperature of $T = 0.2$ to $T = 5.0$, in steps of $\Delta T = 0.1$. We knew from before that the equilibration time was roughly 2000 iterations per lattice site, so we let the model equilibrate over that duration, then let it run for another 15000 iterations in order to generate a sufficient number of data points that were spaced 2τ apart, where we also determined τ earlier.

In Fig. 4, we have plotted the magnetization per temperature and susceptibility per temperature, as well

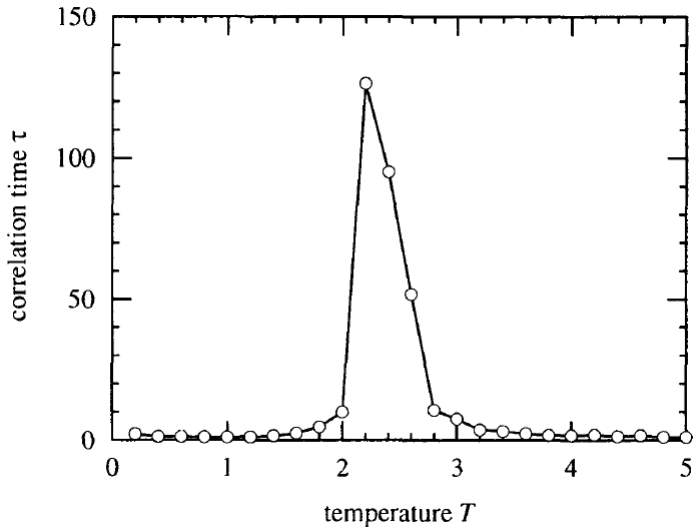

Figure 5: Graph of the correlation time found from the magnetization autocorrelation function. This was retrieved from [8].

as a vertical dashed line where the analyical value for the critical temperature lies. We can immediately see that there is a type of singularity at roughly $T \approx 2.2$ or $T \approx 2.3$, and it strongly corresponds with the exact solution for the critical temperature.

Our code was unable to reproduce the correlation time due to some unforseen bugs and an inadequate amount of time to devote to bug fixing, however, from previous results, we know that we had the correct formulas and gneeral data, so we retrieved a metric from [8] to show what the correlation time from the magnetization autocorrelation function against temperature looks like. We notice that is also has some sort of singularity present at the critical temperature.

Why all of these values should have this behavior is due to the fact that at this temperature, there is essentially an equal competition between thermal motions and the tendency of the system to want to magnetize. Because of this, the correlation length, which essentially characterizes how large the domains that have formed are, will actually diverge, as we are just as likely to find regions in which there are extremely small clusters as we are to find regions where there are extremely large clusters [8]. At this critical temperature, this is determined entirely by the random thermal motions, as sometimes they will push the system over into a state where overall wants to magnetize, and sometimes it will push the system over into a state where it overall wants to be random. This is inherently uncertain as it is entirely based on the thermal fluctuations of the system.

Additionally, we notice that correlation time starts to diverge, and we expect that for larger and larger lattice sizes, this will diverge further. This corresponds with the well-known effect called **critical slowing down**, in which near a phase transition, it takes increasingly long for the system to reach a new state that is statistically independent of the previous states, so our sampling of states in this region leads to more inaccurate results when we are closer to the critical temperature. Performing simulations for increasingly longer or on increasinly larger lattices can circumvent this, and there are also other, more complicated algorithms that can account for this such as Wolff's algorithm, but for the purposes of this paper we find it sufficient to leave it here.

Conclusion and Discussion

The Metropolis algorithm is a moderately simple algorithm that is surprisingly powerful for approximating solutions to the Ising model. The main reason why it is so powerful is that by choose our weight function to be the Boltzmann factor, we have ensured that subsequent states that the algorithm generates are in

accordance with statistical mechanics, and as such, it generates states that are perfectly likely.

In this project, we were able to confirm that the 2-dimensional Ising model has a critical point in the range $T \approx [2.2, 2.3]$, which matches the analytical value found by Onsager to be $T = 2.269$. We have also been able to confirm a number of important results near, such as the presence of critical slowing down.

A 3 dimensional simulation would have been an interesting thing to compute, however, as mentioned in the Introduction, there is no analytical solution, so it would not have been particularly insightful as we would have had no exact solution to compare to. Additionally, for any significant calculation that wasn't dominated by statistical errors, the computational time would have drastically risen, and for the scope of this project, it would not have been feasible to continue running tests while debugging.

References

[1] L. Onsager, "Crystal statistics. i. a two-dimensional model with an order-disorder transition," *Phys. Rev.*, vol. 65, pp. 117–149, Feb 1944.

[2] S. Istrail, "Statistical mechanics, three-dimensionality and np-completeness: I. universality of intractability of the partition functions of the ising model across non-planar lattices," in *32nd ACM Symposium on the Theory of Computing (STOC00)*, (Portland, Oregon), pp. 87–96, ACM Press, 2000.

[3] A. L. Talapov and H. W. J. Blöte, "The magnetization of the 3d ising model," *Journal of Physics A: Mathematical and General*, vol. 29, p. 5727, sep 1996.

[4] D. V. Schroeder, *An Introduction to Thermal Physics*. Oxford University Press, 1 ed., 2021.

[5] L. E. Reichl, *A Modern Course in Statistical Physics*. Wiley-VCH, 4 ed., 2016.

[6] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of State Calculations by Fast Computing Machines," *The Journal of Chemical Physics*, vol. 21, pp. 1087–1092, 06 1953.

[7] S. E. Koonin and D. C. Meredith, *Computational Physics - Fortran Version*. CRC Press, 2018.

[8] M. E. J. Newman and G. T. Barkema, *Monte Carlo Methods in Statistical Physics*. Clarendon Press, 1999.