Monte Carlo Simulation of the Ising model

Tom McIver
tomaberfoyle@gmail.com

May 13, 2020

Abstract

The properties of ferromagnetic crystals were simulated using a Monte Carlo, Markov Chain algorithm first developed by Metropolis and Hastings- the 'Metropolis-Hastings' algorithm. In particular, the behaviour of the internal energy, magnetisation, heat capacity and magnetic susceptibility with respect to temperature observed in the simulated system was in agreement with observations of real ferromagnetic lattices. In addition, hysteresis loops were successfully observed in the simulated system, and the scaling of the simulated lattice properties were found to depend on the simulated lattice size in accordance with the theoretical expectation according to Onsager's analytical results. Here, the results are presented in addition to an analysis of the implementation of the algorithm, which was written in c++.
Keywords: *Ising Model; Metropolis-Hastings; Monte Carlo Markov Chain*

1 Theory

The Hamiltonian of a regular lattice of N spins, that can point either up or down (+1 or -1), is given by:

$$H = -m_0 B \sum_{i}^{N} \sigma_i - J \sum_{i,jnn} \sigma_i \sigma_j \quad (1)$$

Where m_0, B, σ_i and J are magnetic moment of the spins, the applied magnetic field, the spin of the i^{th} point in the lattice and the interaction energy of the spins. In the model considered here, only nearest neighbour spin interactions are considered, thus the sum $\sum_{i,jnn}$ is over all the nearest neighbour pairs in the lattice. Additionally, the lattice considered is a 2.D. square lattice unless otherwise stated.

The Hamiltonian given above was first proposed by Ising, and although it simplifies the physical actuality of spins in a lattice by dropping higher order terms, it has been shown that analytical solutions exist for this model in the 1D and 2D cases which are in agreement with experimental measurements of real ferromagnetic lattices. In particular, this model predicts a phase change, from an ordered state with all spins aligned at low temperatures, to an unordered phase at higher temperatures. No analytical solution has been found for the 3D lattice however, and thus it is necessary to use numerical methods for this case.

To find solutions numerically, the evaluation of averages of macroscopic variables, which is simply the expected value of the macroscopic variable, is necessary:

$$\langle V \rangle = \sum_{i}^{N} P(state_i) V(state_i)$$

Where $V(state_i)$ is the macroscopic variable being observed, such as internal energy or magnetisation, $P(state_i)$ is the probability of that state being observed, and the sum is over all the possible states of the system. The form of $P(state_i)$ is the usual thermodynamic expression:

$$P(state_i) = \frac{e^{-\beta E_i}}{Z}$$

Where Z is the normalising constant, and E_i is the energy of $state_i$. The expected value expression quickly becomes impractical to compute using any brute-force methods however. To see why this is the case consider a lattice of the binary spins as described by the Ising Hamiltonian. For a given lattice dimension size N, the number of possible states of the system is 2^{N^2}. For $N > 17$ the number of possible states of the system is larger than the number of atoms in the universe. In these instances, assuming we could calculate the contribution of each state to the macroscopic average with one computation, it would take the fastest supercomputer approximately 10^{62} years to carry out the macroscopic average. Thus a method along these lines is clearly not practical.

Due to the nature of thermodynamic averages, sampling over the whole phase space of the system is not needed to obtain accurate results however. This is because the probability distributions associated with thermodynamic values are very often sharply peaked, which can be shown by considering the form of the probability of observing a particular partition of energy between system and reservoir ensemble:

$$P(E_i) \propto \Omega_R(U - E_i) \propto e^{ln(\Omega_R(U-E_i))}$$

$$P(E_i) \propto exp(ln(\Omega_R(U) - E_i(\frac{\delta ln(\Omega_R(U)}{\delta U}))$$

$$P(E_i) \propto ln(\Omega_R(U)e^{-E_i} \quad (2)$$

Where the following steps are followed:

1. Assuming the principle of equal a priori probability, we can write that the probability of a given energy partition is proportional to the number of states, Ω_R, that have that energy.

2. Expand the exponential in a Taylor series to arrive at (2).

Expression (2) is sharply peaked, and zero for most values of E_i, meaning that sampling the whole probability space is not necessary. Therefore, to compute the macroscopic averages in an efficient way, it is necessary to bias the sampling in some way so that most samples are taken from regions of the phase space that are more likely to occur. This is the motivation behind the Metropolis-Hastings algorithm.

1.1 Metropolis-Hastings Algorithm

The Metropolis-Hastings (MH) algorithm is referred to as a Markov Chain, Monte Carlo(MC) process. This is because the probability space is sampled in a sequence in which the ith

sample of the space is chosen based on the properties of the previously sampled state (Markov Chain), and the dependency of the i^{th} is itself proportional to a uniform random distribution (Monte Carlo). In particular, the algorithm has the following steps:

1. Initialise a random lattice state (all spins chosen as up or down with 50% probability).
2. Choose a random point in the lattice.
3. Calculate the change in energy ΔE which would result from flipping the spin at this point.
4. If the energy change is less than zero, flip the spin.
5. If the energy is grater than zero, compare $e^{\beta \Delta E}$ to a random variable x generated from a uniform random distribution between zero and one.
6. If $e^{\beta \Delta E}$ is greater than x flip the spin.
7. If $e^{\beta \Delta E}$ is less than x chose a different point in lattice and compare the energy again
8. Repeat steps $2 - 7$, measuring the lattice properties with a fixed period to generate results.

In this process, the flipping of a random point corresponds to taking a new sample and iterating the Markov Chain process. The comparison of $e^{\Delta E}$ to x is a MC process. The intuition behind this is that if the change in energy associated with flipping a randomly selected spin is large and positive, then it is less likely to be flipped. Thus, if ΔE is large, $e^{\Delta E}$ is less likely to be large than x, and the algorithm is less likely to flip this spin. Additionally, the factor of β in the exponential allows for the thermal energy of the system to be account for. Specifically, as temperature increases, the exponential tends to 1, thus corresponding to the observed physical behaviour that spins can freely flip at high temperatures as they are not frozen into fixed lattice states. Crucially, this algorithm samples the probability space randomly however, as there always a finite probability of any lattice point being flipped due to the MoC comparison step.

2 Implementation

2.1 Optimisation

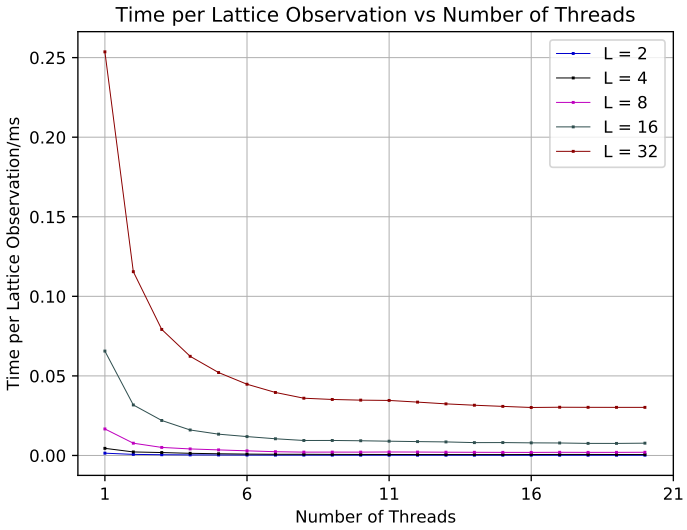


Figure 1: Number of threads used in simulation vs the time taken to simulate one lattice observation. The plotted variable, time per lattice observation is the total time taken for the simulation to run divided by the number of threads multiplied by the number of Monte Carlo cycles (labelled MCC in the c++ script). This simply gives the number of times the lattice was observed per time for each simulation. MCC was set to 100,000 for the data in this plot. N.b. because magnitude of the standard deviations were very small (of order 10^{-6}), and also machine dependent, they are not included in the above figure.

The simulation was implemented in c++, with a class being used to represent the crystal lattice. In order to minimise simulation run times a number of adjustments were made to the program. The most significant improvements came from running the program in multiple threads simultaneously, which consisted in practice of simulating multiple lattices at different temperatures simultaneously. This was only valid when the simulated lattices were independent of one another however. This meant that run-time speed-ups from parallelisation were not possible when studying some behaviours, such as hysteresis. Figure 1 above shows how the extent of parallelisation impacted the time required for one observation of the lattice.

When more than 10 threads were added to the simulation, there was no further improvement in the time per lattice simulation, and it was found that the time per observation remained constant beyond this number of threads. This was still a significant improvement however, as it meant that the proportionality constant of the linear relationship between run-time and number of lattice simulations was much smaller, and un-threaded simulations run times were several hundred percent longer than the threaded equivalent, as shown in figure 2. In particular, speed-ups of approximately 90% in time per lattice observation were observed for all lattice sizes when comparing a single thread to 10 threads.

Figure 2 below, which shows the total simulation time for a threaded and un-threaded simulation, with all other conditions equal, illustrates this considerable speed up from parallelisation.

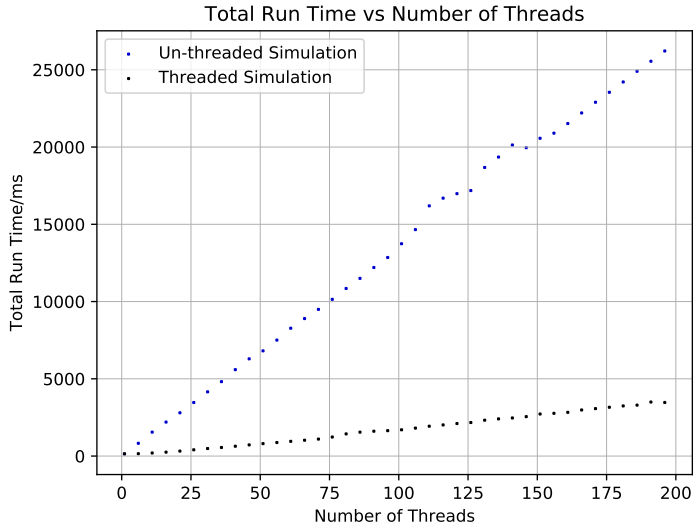
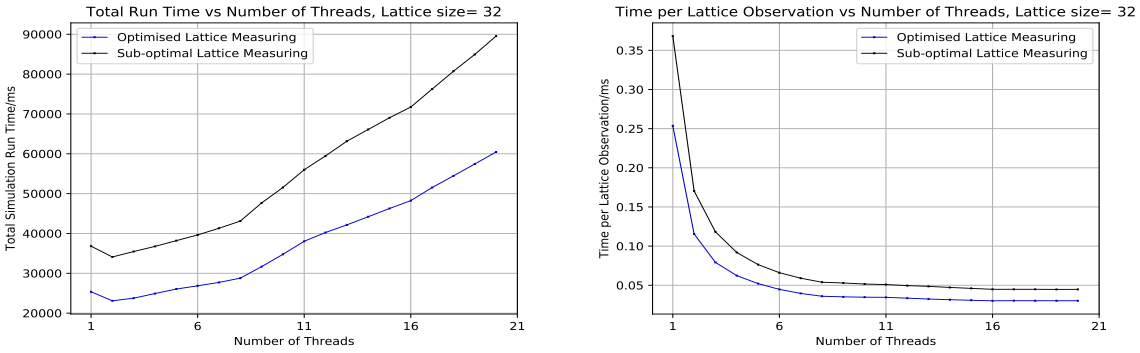


Figure 2: Number of simulated lattices vs total run time.

A second optimisation of the algorithm consisted of implementing a more efficient method of keeping track of the macroscopic variables of the lattice- in particular, the magnetisation and energy. These macroscopic variables depend on the configuration of the whole lattice, and therefore to compute them it is necessary to observe the microscopic contributions arising from each individual spin. To observe the magnetisation and energy it is therefore necessary to loop through the entire lattice. This is a very costly computation however, which scales as $O(L^2)$ if L is the lattice dimension length. Thus it is beneficial to avoid this routine where possible.

One place where such a complete lattice measurement is avoidable is in the MC loop. The MC loop consists of a smaller loop which iterates the lattice N times, applying the MH algorithm in each iteration, and flipping the spins as appropriate. After the application of the MH routine, the macroscopic variables of the lattice are then recorded, and the loop terminates. In a naive implementation, a brute-force measuring of the lattice by looping through the L^2 spins might be used at the end of the MC loop. However, by taking advantage of the fact that the change in energy and magnetisation of each spin is already being computed in each call of the Metropolis-Hastings algorithm, we can keep track of macroscopic variables by simply adding the change in magnetisation and energy to a 'running' magnetisation and energy variable when the spin is flipped. When the lattice is measured this running variable can then be saved in place of the brute-force algorithm that runs through all the spins. Since the number of times the MC loop is called is typically very large (MCC was set to 10^5 to produce most results presented in section 3), the time saved by not looping through the lattice at this point is significant.

Figure 3 below shows a comparison of run times for the optimised and un-optimised lattice measuring for a lattice of size 32. The speed up of $30 - 35\%$ for a single lattice observation as shown in this figure was found to be independent of lattice size.



(a) (b)

Figure 3: Plots comparing the optimised and un-optimised lattice observation mechanism. The data for the above plots was generated using 100,000 Monte Carlo lattice observations ($MCC = 10^5$).

2.2 Implementation of Lattice Class

The full implementation of the program used can be found in the accompanying github repository. The program was primarily written using a 'Crystal' class. This class contained a representation of the spin lattice, which was implemented using a two dimensional square vector, from the std vector class. In addition, vectors were defined in which to hold the macroscopic variables. Various member functions were also written in the class, and of particular importance was the 'UpdatePosition' function, which selects a random point in the lattice, runs the MH algorithm, and then iterates the lattice and variable vectors if the conditions are satisfied. Public member functions are also defined in the Crystal class to iterate the lattice, observe the lattice and return the simulated lattice properties such as average magnetisation and error in the average magnetisation.

In order to maintain some flexibility of usage of this class, not all the program routines were contained in the class however, which allowed for various regimes of physics to be investigated. For example, the parallelisation of the program was carried out in the main body, so that it could used only when appropriate. In particular, a different parallelisation method was needed when studying the lattice in regimes of equilibrium when the applied field was zero and regimes when the applied field was nonzero, the key difference being that at zero field the state of the lattice is independent of the previous state which is not true for the nonzero case. For this reason, there are a number of different main program included in the accompanying repository which were required during the investigation, and comments within the script explain in more detail the exact working of each routine included.

2.3 Choosing the Value of N

N is the number of iterations of the lattice that are carried out per Monte Carlo cycle in the program. This corresponds to the period with which observations of the lattice are to be taken, or in other words the period of time over which averages of the lattice are to be taken over. This is of primary importance in obtaining results, because if N is not large enough then the averages taken will not reflect the equilibrium values of the macroscopic properties. In particular, from thermodynamic theory it is expected that small fluctuations from equilibrium values will occur with some non-negligible probability. Thus, if we do not average over a long enough time period there is a chance that the values observed will reflect one of these fluctuations from equilibrium rather than the actual equilibrium value.

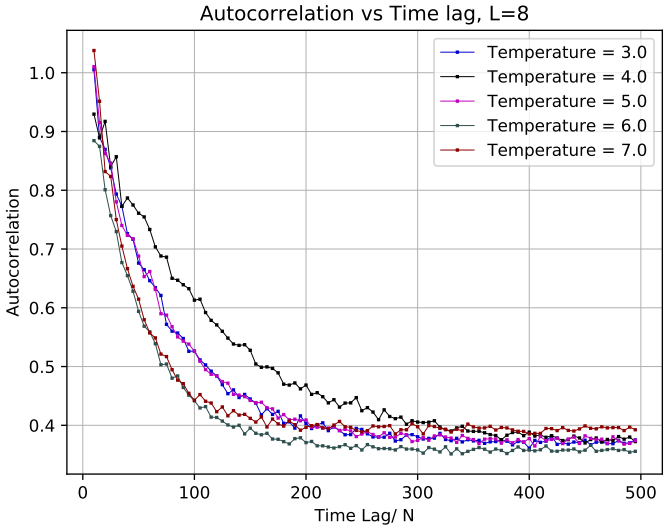


Figure 4: Auto-correlations vs number of lattice iterations between samples for various temperatures.

To decide what N to use, the auto-correlation was used to characterise these fluctuations from equilibrium. This quantity is defined as:

$$a(T) = \frac{A(t_{lag})}{A(t_{lag} = 0)}$$

$$A(t_{lag}) =< M^{'}(t)M^{'}(t + t_{lag}) >$$

$$M^{'}(t) = M(t) - M_{av}$$

Where lower case ts indicate the variable time and upper case Ts temperature- all the above quantities are dependent on T, and the computation of $a(T)$ is thus carried out from $A(t_{lag})$ values all recorded at the same temperature. Intuitively, the auto-correlation gives a measure of how correlated two adjacent measurements are to one another. To avoid measuring in the non-equilibrium regime, we want to be in the region of minimum, or at least low, auto-correlation for a given set of parameters. Figure 4 above shows various

plots of auto-correlation.

The auto-correlation was found to be highly dependent on the temperature. Indeed, below the critical temperature, there was no dependence of the auto-correlation on the time lag, while at high temperatures the auto-correlation was found to have a strong exponentially decaying relationship to the time lag. This as expected, as there is a phase change from ordered to disorder lattice when moving from the low to high temperature regime. Additionally, as temperature increased, the length of the decay also increased- with the equilibrium value of auto-correlation increasing from approximately 200 to over 500 for a temperature increase of $2K/k_b$ to $7K/k_b$.

What are the ramifications of this for the hyper-parameters of the simulation? It turns out that the combination of N and MCC is the most significant quantity, rather than just N or MCC individually. This is because if MCC is very large, then a significant enough proportion of the phase space is observed that the effect of taking non-equilibrium measurements of the lattice is 'washed-out' in the long run. In other words, many observations of the crystal when it is in small fluctuations from equilibrium are observed, meaning that the total effect is averaged over.

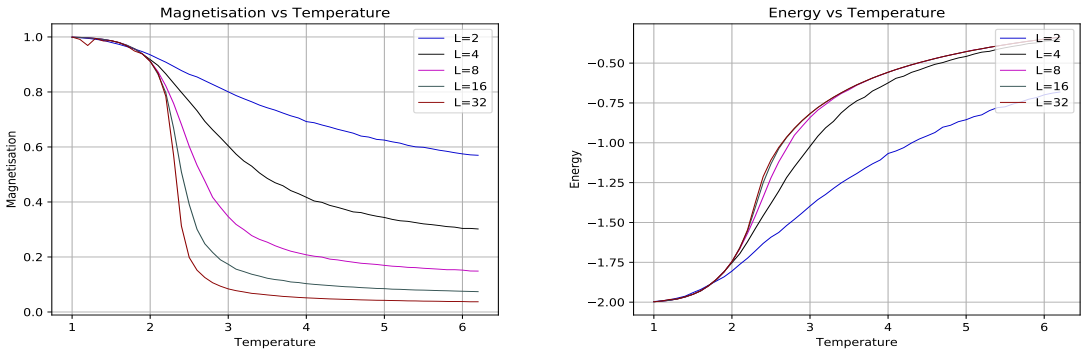
The value of N is thus dependent on MCC- we need to ensure N is large enough for the given MCC value, so that averages computed are over a large enough region of the phase space. At the same time, it is beneficial to to minimise the run time, rather than just choosing a large value for N and MCC. In practice, this was done by trial and error, choosing multiple value of N and trying to minimise MCC by comparing the error in the recorded averages. It was found that running 100,000 MC cycles, and setting N to the size of the lattice worked well for all lattice sizes and temperatures investigated, thus this was chosen to produce the results in section 3.

The fact that these values for the hyper-parameters produced averages with acceptable error values was important, as it ensured that no implementation dependent behaviour was recorded- i.e. any changes between results at different temperatures were due to physical changes, not changes due to carrying the period over which averages were calculated. Additionally, this choice of hyper-parameter produce run times of no more than a couple of minutes, and in most cases seconds, which was acceptable to work with from a practical perspective.

3 Results

3.1 Phase Change Behaviour

Figure 5 shows the behaviour of the simulated lattice with respect to temperature, which was found to be in agreement with the analytical results derived by Onsager (Onsager, 1944). In particular, a phase change was observed between a scaled temperature of 2 and 3, and as lattice size was increased the critical point converged towards a value of approximately 2.3. For an infinite lattice, a phase change is predicted to occur at $\frac{2}{ln(1+\sqrt{2})}$, which is agrees well with the observed simulated results.



(a) (b)

Figure 5: Plots showing the behaviour of average total lattice energy per atom and average absolute magnetisation per atom with respect to temperature for various lattice sizes. The results are displayed using scaled units (su), with k_b and the interaction energy, J set equal to 1.

Additionally, the energy varied as expected, converging to $-2su$ per atom as temperature falls to zero. We can easily verify that this behaviour is correct by comparing this value to the hamiltonian (1) of the system. The minimum value of energy which an individual atom can take is -4 four, and since each of the bonds in the lattice are shared between two atoms, overall we expect there to be two units of energy per atom when the total lattice is in the minimum energy state. By inspecting the behaviour of the heat capacity, further analysis of whether the low temperature behaviour of the simulated energy is possible. Figure 6 shows the computed heat capacity.

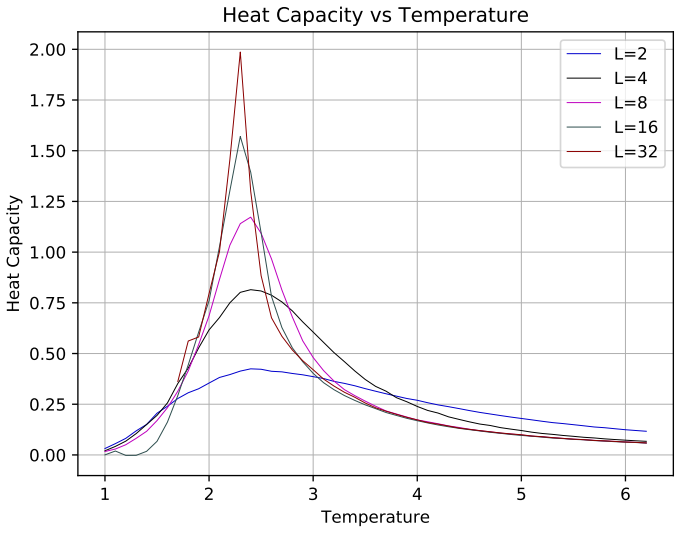


Figure 6: Auto-correlations vs number of lattice iterations between samples for various temperatures.

Using standard Thermodynamic principles, the heat capacity can be written in terms

of the average energy and average squared energy, which is how the data for figure 6 was computed. The plot shows a peak in heat capacity around the transition temperature, which becomes sharper as the lattice size is increased. This is as expected, as given the step function shape of the energy with respect to temperature in this region, we would expect a discontinuity in the derivative in the same region, and the heat capacity is the derivative of energy with respect to temperature. Thus a peak tending to infinite is what we expect the heat capacity temperature dependence to look like in this region.

Finally, the results presented above are also in agreement with experimental observations of real lattices, and agree with well established theory aside from onsager - see chapter 12 in (Kittel, 2005). Although for clarity the plots presented in this report are do not include error bars, it was found that all the presented results had converging associated errors, that were well within acceptable limits. Plots of individual lattice simulations, rather than comparison of lattice simulations for varying L as presented here, are included in the accompanying github repository which are plotted with error bars.

3.2 Hysteresis

A second method to verify the simulated crystal reproduced physical results was to observe the hysteresis behaviour of the simulation. In particular, hysteresis loops of magnetisation are a well established phenomena observed in real ferromagnetic lattices (Kittel, 2005), which have significant consequences in practical situations. Thus, if the model simulated here is to be of any worth it must reproduce hysteresis effects. Figure 7 shows that indeed hystersis effects were shown in the model.

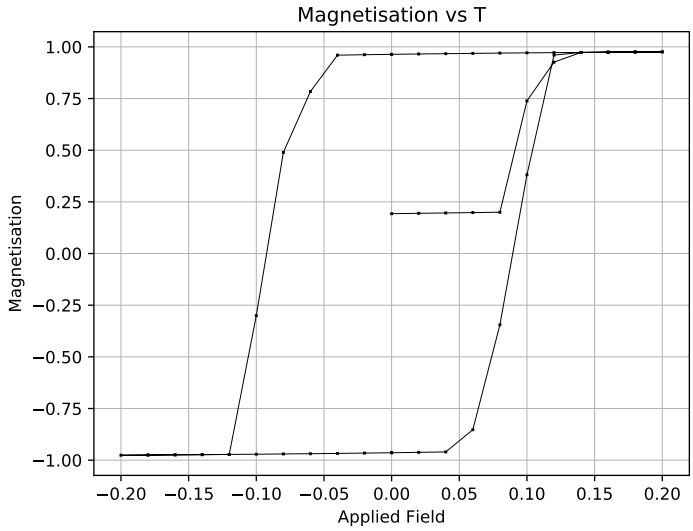


Figure 7: Hysteresis loop for a lattice of size $L = 16$, cycled at $T = 1.75su$. The Magnetisation is in units of average magnetisation per spin and the applied field in normalised units of $\frac{Tesla}{k_b}$.

The area of the hysteresis loop was found to depend strongly on the hyper-parameters

of the simulated model. In particular, the lattice dimension length had a marked impact on the size of the observed hysteresis loop, with larger L producing more prominent loops, while loops at $L < 16$ did not display any significant hysteresis effects. This was likely due to the use of cyclical boundary conditions in the simulation, which meant that in lattices of small L there was no space for large spin aligned domains to form which would be stable outside of their equilibrium conditions (see (Kittel, 2005) for theoretical background of Hysteresis). At larger L, such large spin aligned domains could form however, and in fact as L was increased, the temperature region in which hysteresis could be observed increased. This corresponded to the simulation converging to the real physical situation as the lattice size was increased, and at large Ts the expected behaviour of hysteresis being observed at temperatures slightly above the Curie temperature was observed. The appendix folder in the accompanying github repository includes various Hysteresis loops observed under different conditions.

4 Discussion and Conclusions

The successful reproduction of the primary behaviour of a 2.d lattice here suggests that there is scope for the model to be extended to three dimensions and to more extreme conditions. Additionally, other lattice structures or or non-crystalline materials could be investigated. Scaling of the algorithm presented here may cause some issues however, especially in regimes where parallelisation is not easily implemented. An area which may be of particular use would be to investigate whether simulations with no natural parallelisation method my be internally divided so to make certain sub sections of code sympathetic to parallelisation, thus making more complex systems applicable for the MCMC analysis presented here.

References

Kittel, C. (2005). *Introduction to solid state physics 8th ed*.

Onsager, L. (1944, Feb). Crystal statistics. i. a two-dimensional model with an order-disorder transition. *Phys. Rev.*, *65*, 117–149. Retrieved from `https://link.aps.org/doi/10.1103/PhysRev.65.117` doi: 10.1103/PhysRev.65.117