Investigation of phase transitions in magnetic systems using the Ising Model

Ove Haugvaldstad, Eirik Gallefoss

November 18, 2019

Abstract

The Ising model is one of the most studied models in statistical mechanics. In this work we look at a classic application of the Ising model, namely the study of phase transitions in magnetic systems. We implement a stochastic Monte Carlo model of the Ising model following the Metropolis algorithm. We ran our model for several temperatures and grid sizes and calculated the critical temperature for each grid. Then we use a simple linear regression model to fit a line to the previously calculated critical temperatures. From the regression line we estimate the critical temperature for an infinite grid. Our result of $T_C(L = \infty) = 2.2687$ are close to the analytical result of Lars Onsager (Onsager 1944) with a relative error of 2.00×10^{-4}.

1 Introduction

There are many phenomenons that arise as a result of physical entities interacting with each other and together producing a global behaviour. Ferromagnetism is the result of such collective behaviour. A widely used model for studying these collective phenomenons is the Ising model. The model was first suggested by Wilhelm Lenz in 1920 for modelling ferromagnetism and later solved in one dimension by his PhD student Erst Ising in 1925. Since the nineteen twenties the Ising model risen to become one of the most studied models in statistical mechanics and has even transcended the traditional boundaries between the disciplines, finding application ranging from social sciences and biology to chemistry and mathematics.

In this work we will look at a classic application of the Ising model, namely the study of phase transitions in magnetic systems. We will use a common stochastic Monte Carlo approach to model the Ising model, built around the Metropolis algorithm. We will give a through explanation of our numerical model and demonstrate the behaviour of the model at different temperatures, lattice sizes and initial states. The aim with our analysis is to determine the temperature where the magnetic system transitions from a magnetized state to a non-magnetized state. This temperature will subsequently be refereed to as the critical temperature of the system. We will find the critical temperature for several lattice sizes and then extrapolate the critical temperatures to in order to estimate the critical temperature for an infinite lattice and check our estimate against Lars Onsager's analytical solution eq. (1) (Onsager 1944).

$$T_C(L=\infty) = \frac{2}{\ln\left(1+\sqrt{2}\right)} \approx 2.269 \tag{1}$$

2 Theory

Conceptually these magnetic systems are quite simple to model, consisting of binary spins which can either have the value 1 (up) or -1 (down). The energy of this system is given by summing up nearest neighbour spins of all the spins in our system. We evolve our system by randomly selecting spins and changing their values if it is energetically favourable for the system. To not have our system being stuck in the ground state (lowest energy state) we will follow the Metropolis algorithm to randomly flip some spins anyway, a behaviour is which is present in real systems.

2.1 Ising model

The Ising model without an external magnetic field is given by eq. (2), where E_i is the energy of a given microstate, s$_k$ and s$_l$ takes values of ± 1, O is the number of spins, J is a coupling constant corresponding to the strength of the interaction between the neighbouring spins, and $< kl >$ means that we sum over nearest neighbour only.

$$E_i = -J \sum_{<kl>}^{O} s_k s_l \tag{2}$$

The parameter that we can control in our experiment is the temperature, it is therefore it is natural to use the canonical ensemble where temperature is the intensive property (Hjort-Jensen 2015). The mean energy of the system is then given as an expectation value. To calculate the expectation value of for instance the energy requires a probability distribution for a given temperature. In the canonical ensemble the probability distribution is given by the Boltzmann distribution eq. (3).

$$P_i(\beta) = \frac{e^{-\beta E_i}}{Z} \tag{3}$$

Where β is the inverse temperature times the Boltzmann constant ($\beta = 1/k_b T$), E_i is the energy of a microstate and Z is the partition function for the canonical ensemble eq. (4).

$$Z = \sum_{i=1}^{N} e^{-\beta E_i} \tag{4}$$

The properties which we are interested in, the energy $\langle E \rangle$, heat capacity C_v, magnetisation $\langle M \rangle$ and susceptibility χ are then given by the following expressions.

$$\langle E \rangle = \frac{1}{Z}\sum_{i=1}^{N} E_i e^{\beta E_i} \tag{5}$$

$$C_v = \frac{1}{k_B T^2}\left(\langle E^2 \rangle - \langle E \rangle^2\right) \tag{6}$$

$$\langle M \rangle = \frac{1}{Z}\sum_{i=1}^{N} M e^{-\beta E_i} \tag{7}$$

$$\chi = \frac{1}{k_b T}\left(\langle M^2 \rangle - \langle M \rangle^2\right) \tag{8}$$

The variance of the energy and magnetisation is defined as the following.

$$\sigma_E^2 = \left(\langle E^2 \rangle - \langle E \rangle^2\right) = \frac{1}{Z}\sum_{i=1}^{N} E_i^2 e^{\beta E_i} - \left(\frac{1}{Z}\sum_{i=1}^{N} E_i e^{\beta E_i}\right)^2 \tag{9}$$

$$\sigma_M^2 = \left(\langle M^2 \rangle - \langle M \rangle^2\right) = \frac{1}{Z}\sum_{i=1}^{N} M_i^2 e^{-\beta E_i} - \left(\frac{1}{Z}\sum_{i=1}^{N} M e^{-\beta E_i}\right)^2 \tag{10}$$

2.2 Metropolis algorithm

The issue with our current setup of the Ising Model is that our probability distribution, that we need in order to calculate the expectation values, contains the partition function Z. To calculate the partition function requires on the order 2^O,FLOPS where O is the number of spins. For large lattices this becomes impossible compute. Therefore we need to devise a scheme that lets us obtain the probability distribution without having to calculate the partition function. To achieve this end we will apply two common procedures in stochastic modeling, Monte Carlo sampling (Haugvaldstad and Gallefoss 2019a), and the Metropolis algorithm .

After setting up the initial state, the first step in devising our stochastic Ising model would be to setup our Monte Carlo sampler or random walk. The random walk will randomly select spins in our lattice. For each of the randomly selected spins we calculate the energy and then make a proposal about whether to flip the orientation of the spin.

We will accept the proposal if flipping the spin results in lowering the energy of the system. Still there is an issue with only accepting the proposals that lowers the energy. The problem is that after some steps we would start refusing every subsequent proposals, because all of the proposed moved would result in a higher energy. This would eventually lead to us getting stuck in the lowest energy state. To avoid getting stuck we would need to accept some proposals which increase the energy of the system. This is where the Metropolis algorithm comes to our rescue. The first step of the Metropolis algorithm is to draw a random number between 0 and 1 from a probability distribution, we will use the uniform distribution. Next is to check if the our random number r, $r \leq P_i/P_j$, where P_i is the probability distribution of our new state and P_j is the probability distribution of the previous state. If we recall that the probability distribution in both states are given by the Boltzmann distribution eq. (3), the probability ratio P_i/P_j then becomes eq. (11).

$$\frac{P_i}{P_j} = \frac{e^{-\beta E_i}/Z}{e^{-\beta E_j}/Z} = e^{-\beta \Delta E} \;, \quad \Delta E = E_i - E_j \tag{11}$$

It worth noticing that the probability ratio does not depend on the partition function. The Metropolis rule for which we accept or reject proposed moves then becomes $r \leq e^{-\beta \Delta E}$. It is possible to pre-calculate the exponential $e^{-\beta \Delta E}$ isfor a given temperature, because in the Ising model the energy difference can only take on five different values eq. (12).

$$\Delta E = -8J,\ -4J,\ 0,\ 4J,\ 8J \tag{12}$$

This is quite straight forward to verify by using eq. (2) to calculate the energy of the original state and the new proposed state and then finding the difference. Pre calculating the energy saves us a lot of CPU cycles as we do not need to calculate an exponential each time we propose a new state. There is a neat physical interpretation to the Metropolis algorithm as it models the contest between two fundamental principles of our universe, energy minimization and entropy maximization. This is something we will observe in our model as the entropy or acceptance ratio increases as we increase the temperature.

2.3 Analytic solution of the Ising Model for a 2x2 lattice

In reality it is impossible to calculate the partition function, because there could be an infinite number of microstates. Still for a small lattice 2x2, which only has 16 microstates it is feasible to find analytic solutions for the values of interest. Comparing with the analytical expressions will be a useful test of our numerical estimates. For details see section A.

We start by calculating the energy for a given microstate.

$$E_i = -2J\left[(s_4 + s_1)(s_2 + s_3)\right] \tag{13}$$

To calculate the expectation values we need to know the possible energies and magnetizations. The script analytic.py [1] generates all the possible states and lists the values we need as shown in table 1.

Number of spins up	Energy	Magnetization	Degeneracy
4	-8	4	1
3	0	2	4
2	0	0	4
2	8	0	2
1	0	-2	4
0	-8	-4	1

Table 1: Analytical values for a 2x2 grid.

The probability of a given microstate is given by eq. (14), with the z being a normalization factor to ensure the sum of probabilities are one, known as the partition function, given in eq. (15).

$$P_i = \frac{\exp(-\beta E_i)}{z} \tag{14}$$

$$z = \sum_i^N \exp(-\beta E_i) = \exp(8\beta) + 12 + 2\exp(-8\beta) \tag{15}$$

3 Experimental Setup

In our numerical model we will use use quadratic grids $L \times L$ and periodic boundary conditions. Periodic boundary conditions mean that our grid is repeated at the around the edges, this is necessary since we can not have infinite grids in our numerical model.

3.1 Equilibrium

In order to get good statistics in the production runs we want to start calculating expectation values when the system has reached equilibrium, or the most likely

[1] https://github.com/Ovewh/Computilus/tree/master/Project4

state. A crude method for estimating this delay value (# of MC cycles) is to plot energy and magnetization against MC cycles. This is shown in Figure 1 and Figure 2 for $T \in [1.0, 2.4]$ for both ordered (all spins pointing up) and unordered initial states. The system takes longer to reach the most likely state when the initial state is unordered. The magnetisation take longer to stabilize compared to the energy and at around 7500 cycles all values seems to be stable. Still to be on the safe side, we decide to discard the first 50000 samples, which for a run with 1 million cycles only amount to 5 % of the total samples size.

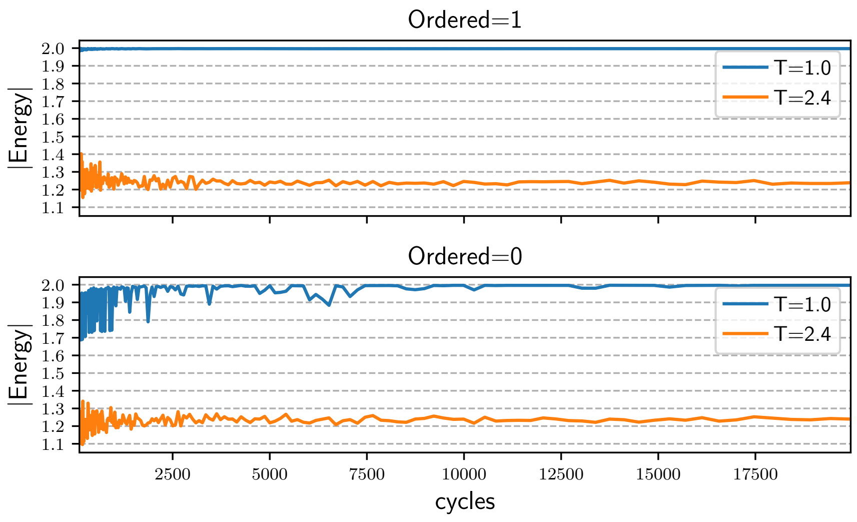


Figure 1: Absolute values of energy for different MC cycles. ord refers to the initial microstate of spins. 1=up, 0=random.

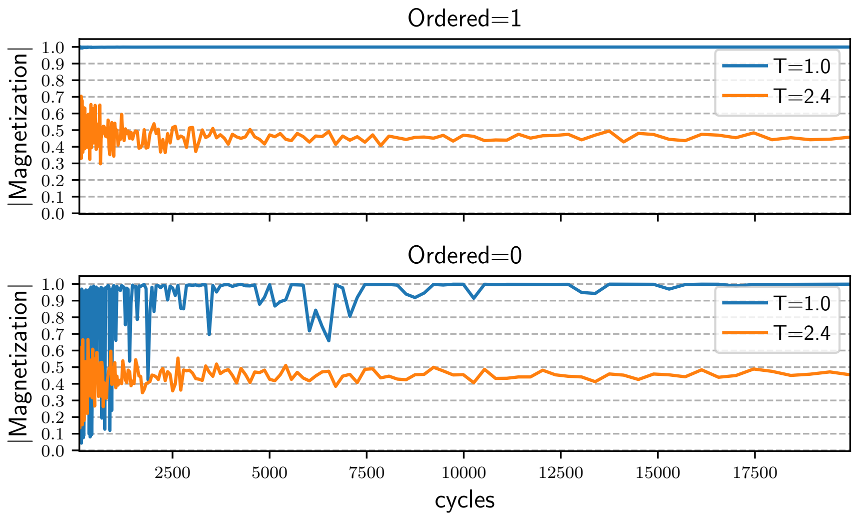


Figure 2: Absolute values of magnetization for different MC cycles. ord refers to the initial microstate of spins. 1=up, 0=random.

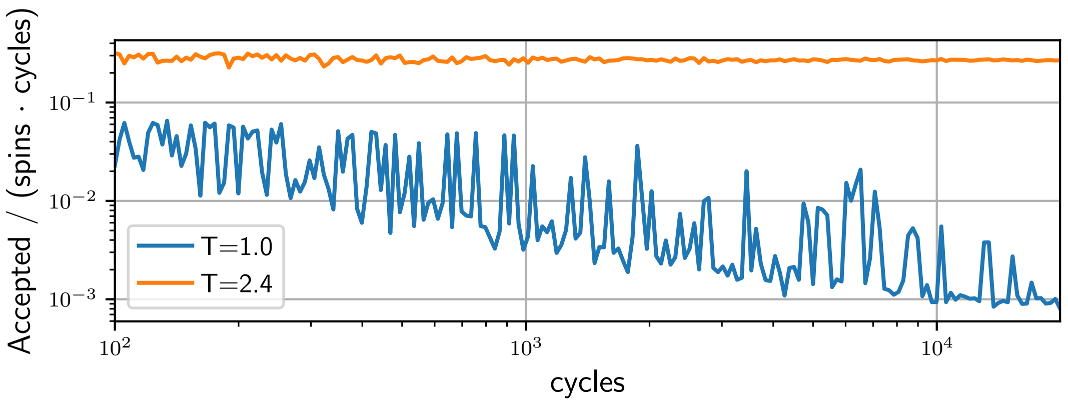
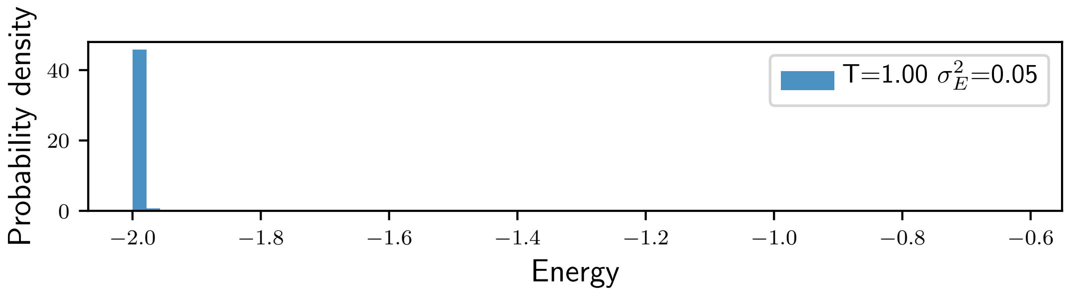


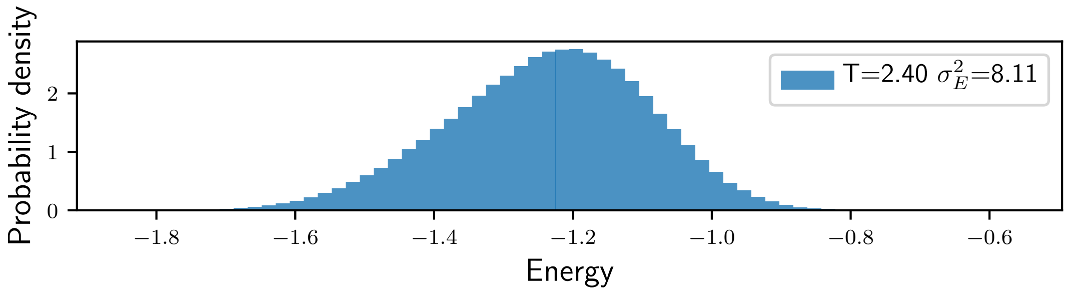
Figure 3: Accepted microstates scaled with number of spins and MC cycles.

3.2 Probability distribution

The probability distribution of energy (section 3.2) shows that at low temperature almost all of the states are in a low energy state. As the temperature and the variance increases the distribution becomes more spread out.



(a) When the temperature is low σ_E^2 is also low and almost all of the energies are concentrated narrowly around the most likely state.



(b) Higher temperature gives a higher σ_E^2 and fluctuations around the most likely state.

Figure 4: Probability distribution of energy, scaled with number of spins. L=20 MC cycles = 1×10^6.

3.3 Model testing and error analysis

Implementation of tests [2] showed that the analytical solutions was correct down to an absolute error of 1×10^{-12} when comparing against a brute force method where we created all the possible microstates.

To test the implementation of the ising model we calculated the relative error in energy, magnetization, heat capacity and susceptibility (fig. 5) for L=2 (where we have calculated analytical solutions). The relative error was proportional to $1/\sqrt{\text{MC cycles}}$. While there were some differences in the relative error between the ordered and unordered inital configuration for few Monte Carlo cycles, the difference decreased as we run more cycles.

[2]test_analytic.py, test_ising.py

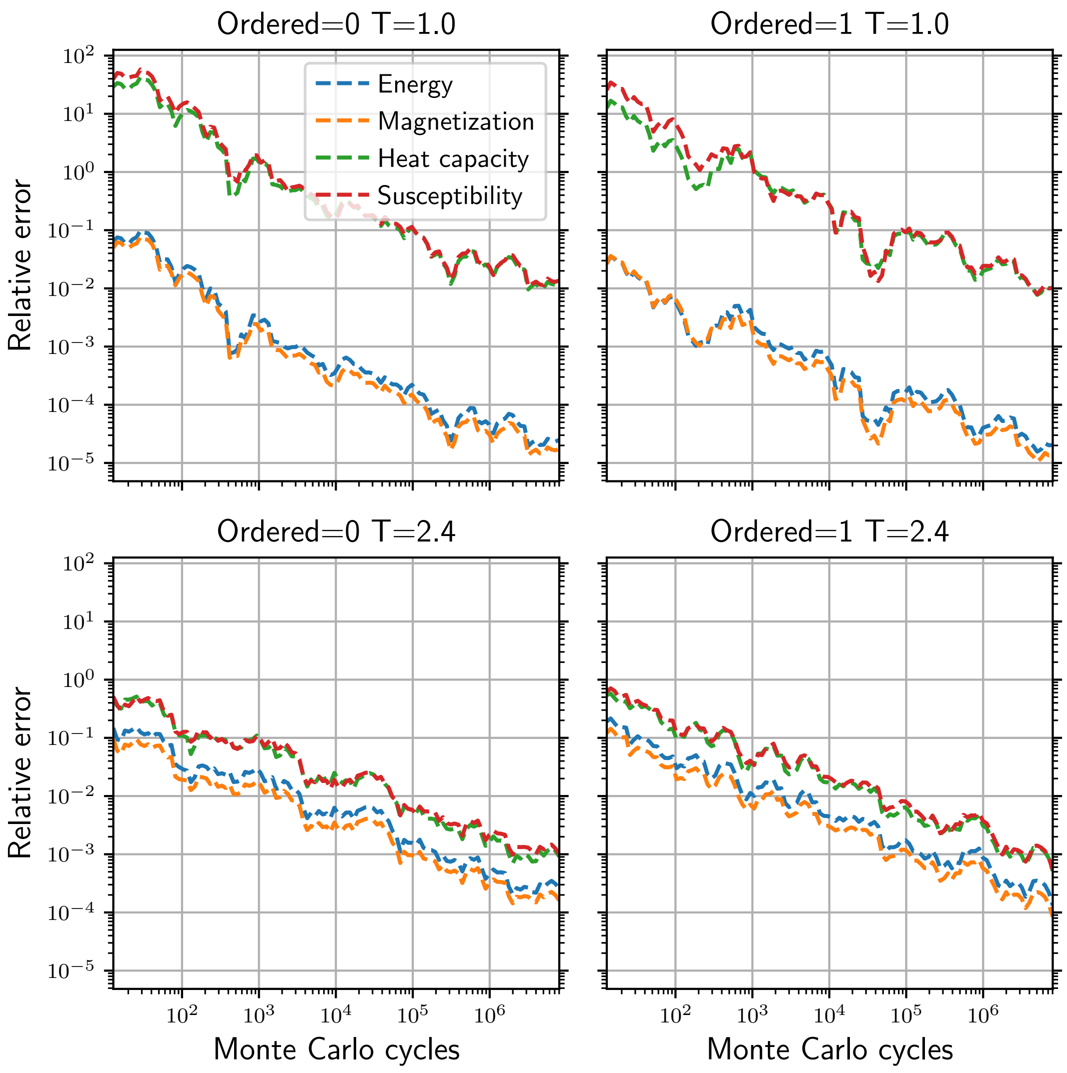


Figure 5: Relative error vs MC cycles for L=2 for two different temperatures. Ordered refers to the inital spin configuration, with 1 meaning all spins pointing up and 0 meaning a random configuration. Plots were smoothed by taking a rolling mean with window size 5. The relative error is proportional to $\frac{1}{\sqrt{MC\ cycles}}$.

3.4 Production runs

The production runs (phase_transitions.py) was parallelized using $@njit(parallel = True)$ functionality within numba. It was easiest to run calculations for each temperature in parallel, instead of parallelizing the Ising model itself. A limitation of the using numba for parallelization was that we were unable to write to file after each run of the Ising model. This made our code vulnerable to crashes, since if the run did not finish we would lose all the results. Timings with and without parallelization for small grid sizes showed a speedup of roughly 3.4 on a computer with 4 physical cores.

To get a measure on how long the production runs would take, we created a script [3] that estimates the time to run phase_transitions.py with the given parameters. We used this script to avoid starting too time consuming runs.

We did two production runs, one over a larger temperature range from $\beta \in [2.0, 2.8]$ where aim we capture the larger picture, both before and after the phase transition. Then the second run centred around the critical temperature $\beta \in [2.2, 2.35]$. Both runs was ran for four different grid sizes $L \in [40, 60, 80, 100]$, and with 1 million MC cycles, for which we discarded the first 50000 samples. The temperature step size dT for the two runs was 0.001 and 0.0005 respectively. All production runs were started with an unordered state.

4 Results

4.1 Phase transitions

The results (fig. 6) indicated a phase transitions for T $\in [2.2, 2.35]$. The simulations with higher grid size exhibits sharper changes, meaning that they will capture the phase transition better, because of a smaller contribution from the boundaries.

The run centred around the critical temperature with $dT = 5 \times 10^{-4}$ was better at capturing the the peaks of C_v and χ. We fitted sixth order polynomials to C_v and found the critical temperature Tc$_L$ for each grid size (7a).

$$T_C(L = \infty) = T_C(L) - aL^{-1/\nu} \qquad (16)$$

Using the finite size scaling relation in eq. (16) with $\nu = 1$, we estimated $T_c(L = \infty)$ by doing a linear regression with the four different T$_C$(L), using $1/L$ as the

[3]timings.py

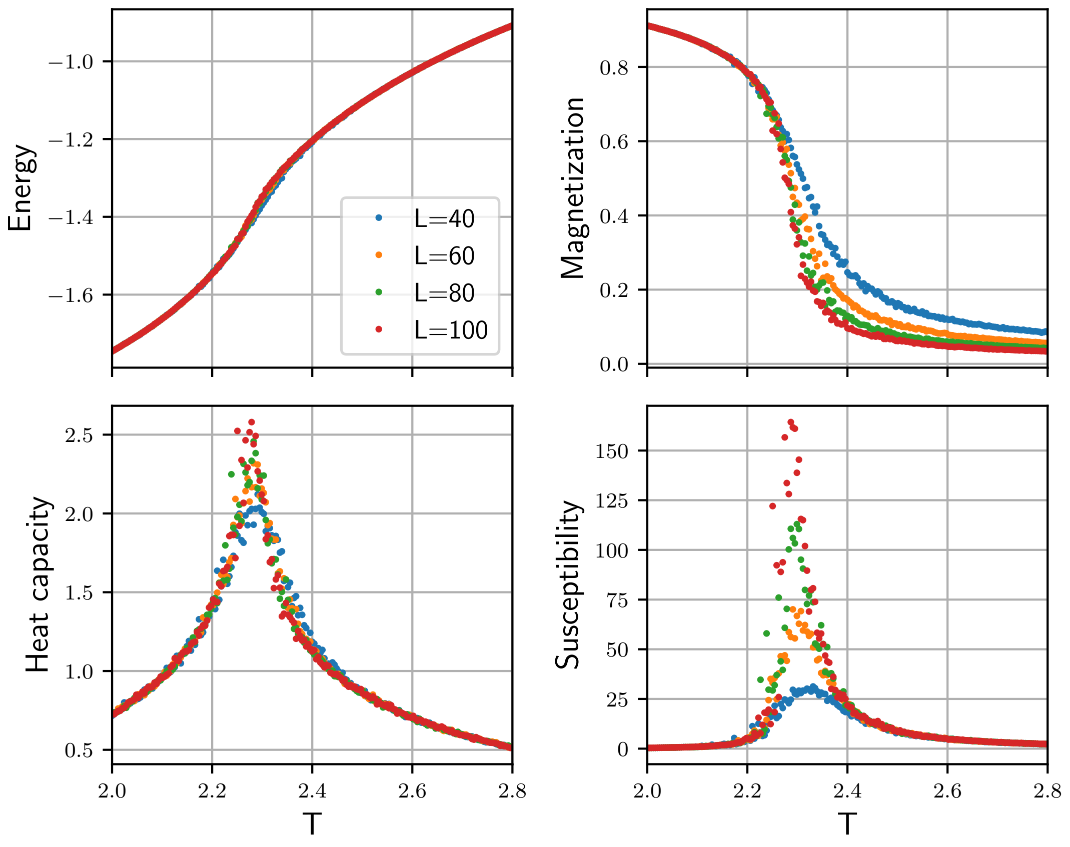


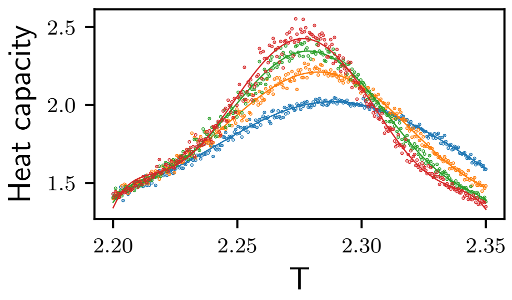
Figure 6: Simulating for various temperatures looking for phase transitions.
$T \in [2.0, 2.8]$, $dT = 4 \times 10^{-3}$, 1×10^{6} MC cycles, delay=50000

independent variable, and extracting the intercept (fig. 7b) [4]. This gave us the result of $T_C(L = \infty) = 2.2687$, with the 99% confidence interval $[2.2617, 2.2757]$.

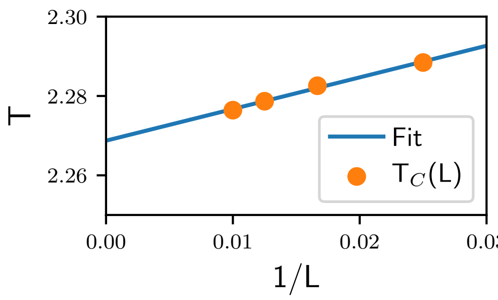
	Value	CI low	CI high
T_C	2.2687	2.2617	2.2757

Table 2: Results for $T_C(L = \infty)$. The confidence intervals only take into account the extrapolation of $T_C(L = \infty)$.

[4]Implemented in critical_temperature.py



(a) Fit of sixth order polynomials (lines) to the measurements of heat capacity (circles).



(b) Linear regression on the critical temperatures for $L \in [[40, 60, 80, 100]]$ to extrapolate $T_C(L = \infty) = 2.2687$ with the 99% confidence interval $[2.2617, 2.2757]$.

5 Discussion and Conclusion

Our model seems to have captured the phase transition well. As we increased the grid size the phase change in magnetization started to behave more and more like a discontinuity. Compared to Onsager's results (eq. (1)) we got an absolute error of 4.54×10^{-4} and a relative error of 2.00×10^{-4} which we are very pleased with.

The values for the confidence intervals around the T_c(L=∞) assumes that the values for T_c(L) are certain. This is not correct, since we have uncertainties inherent in the Monte Carlo simulations and in the method we use to estimate T_c(L). As shown in fig. 5 our model exhibited relative errors on the order of 1×10^{-3} compared to the analytic solution using a 2x2 grid. If the errors on a larger grid are comparable this should not be a major source of error.

We tested two other methods for estimating $T_C(L)$. The first method was to take the mean of the n points with largest heat capacity. The second was to take a rolling mean with a window of n. We did this for various n, and as long as we kept n reasonably small compared to the 300 different temperatures of our dataset this gave similar results for $T_C(L)$ compared to fitting a sixth order polynomial. Therefore the method we have used for selecting $T_C(L)$ does not seem to be a major source of error either.

We encountered a few problems using numba to speed up and parallelize the code. The first was that it is generally difficult to split up code into smaller functions that can be unit tested. The other was that we did not manage to both parallelize and write to file after the run for each temperature and grid size. This meant we had to do the production runs, that took roughly 15 hours, without the possibility of looking at the current progress. In one run this led to no results being written to file, since there was an error in the file writing function.

References

[1] O. Haugvaldstad and E. Gallefoss. "Solving multidimensional Integrals using Monte Carlo Integration". 2019. URL: https://github.com/Ovewh/Computilus/blob/master/Project3/project3.pdf.

[2] O. Haugvaldstad and E. Gallefoss. "Source files for this project". 2019. URL: https://github.com/Ovewh/Computilus/tree/master/Project4.

[3] M. Hjort-Jensen. "Computational physics, lecture notes fall 2015". 2015. URL: https://github.com/CompPhysics/ComputationalPhysics/blob/master/doc/Lectures/lectures2015.pdf.

[4] Lars Onsager. "Crystal Statistics. I. A Two-Dimensional Model with an Order-Disorder Transition". In: *Physical Review* 66 (4 1944). URL: http://www.phys.ens.fr/~langlois/Onsager43.pdf.

Appendices

A Analytic solutions

$$
\begin{aligned}
E_i &= -J \sum_{<k,l>} s_k s_l = -\frac{1}{2} J \sum_i \sum_{nn} s_i s_{nn} \\
&= -\frac{1}{2} J \left[2s_1(s_2+s_3) + 2s_2(s_1+s_4) + 2s_3(s_4+s_1) + 2s_4(s_2+s_3) \right] \\
&= -J \left[2(s_4+s_1)(s_2+s_3) \right] \\
&= -2J \left[(s_4+s_1)(s_2+s_3) \right]
\end{aligned}
$$

$$
\begin{aligned}
\langle E \rangle &= \sum_{i=1}^{N} E_i P_i && (17) \\
&= \left[2 \cdot -8 \exp(8\beta) + 2 \cdot 8 \exp(-8\beta) \right] / z && (18) \\
&= \frac{16}{z} (\exp(-8\beta) - \exp(8\beta)) && (19) \\
&= -\frac{32}{z} \sinh(8\beta) && (20)
\end{aligned}
$$

$$
\begin{aligned}
\langle E^2 \rangle &= \sum_{i=1}^{N} E_i^2 P_i && (21) \\
&= \Big[2(-8)^2 \exp(8\beta) + 2(8)^2 \exp(-8\beta) \Big] / z && (22) \\
&= \frac{128}{z} \left[\exp(8\beta) + \exp(-8\beta) \right] && (23) \\
&= \frac{256}{z} \cosh(8\beta) && (24)
\end{aligned}
$$

$$
\begin{aligned}
\langle E \rangle^2 &= (\frac{16}{z})^2 (\exp(-16\beta) + \exp(16\beta) - 2) && (25) \\
&= \frac{32^2}{z^2} \sinh(8\beta)^2 && (26)
\end{aligned}
$$

$$\begin{aligned} \langle M \rangle &= \sum_{i=1}^{N} |M_i| P_i && (27)\\ &= \left[2 \cdot 4 \exp(8\beta) + 8 \cdot 2 \exp(0)\right] / z && (28)\\ &= \frac{8}{z} \left(\exp(8\beta) + 2\right) && (29) \end{aligned}$$

$$\begin{aligned} \langle M^2 \rangle &= \sum_{i=1}^{N} |M_i|^2 P_i && (30)\\ &= \left[2(4)^2 \exp(8\beta) + 8(2)^2\right] / z && (31)\\ &= \frac{32}{z} \left[\exp(8\beta) + 1\right] && (32) \end{aligned}$$

$$\begin{aligned} \langle M \rangle^2 &= \frac{8}{z}^2 \left(1 + 2\exp(8\beta)\right)^2 && (33)\\ &= \frac{64}{z^2} \left(\exp(16\beta) + 4\exp(8\beta) + 4\right) && (34) \end{aligned}$$

$$\begin{aligned} C_v = \frac{\sigma_E^2}{k_b T^2} &= \left(\langle E^2 \rangle - \langle E \rangle^2\right) / \left(k_b T^2\right) && (35)\\ &= \left(\frac{256}{z} \cosh(8\beta) - \frac{32^2}{z^2} \sinh(8\beta)^2\right) / \left(k_b T^2\right) && (36) \end{aligned}$$

$$\begin{aligned} \chi = \frac{\sigma_M^2}{k_b T} &= \left(\langle M^2 \rangle - \langle M \rangle^2\right) / \left(k_b T\right) && (37)\\ &= \left(\frac{32}{z} \left[\exp(8\beta) + 1\right] - \frac{64}{z^2} \left(\exp(16\beta) + 4\exp(8\beta) + 4\right)\right) / \left(k_b T\right) && (38) \end{aligned}$$