FYS4150: Project 4
Studies of Phase Transitions in Magnetic Systems

Andri Karoline Lunøe Spilker
Code-Partner: Tiffany Chamandy

November 14th, 2016

Abstract

This project uses the Ising model with lattices of up and down spins represented by matrices of ±1 and the Metropolis algorithm to investigate the behaviour of energy, magnetic moment, specific heat and magnetic susceptibility as spins are flipped and the system moves towards a steady state. We find that a steady state is reached after ~100 Monte Carlo cycles containing NxN spin flips for a temperature of $1J/k_B$, while it happens faster for ordered matrices and higher temperatures. Using parallellisation with MPI we find that larger matrices give a better approximation of reality, and clearer indications of phase transitions. Ultimately the critical temperature found by averaging over results for N=40,60,100 and 140 matrices was 2.2781 J/k_B considering the specific heat, and 2.2969 J/k_B calculated with the susceptibility. This signifies that a second order phase transition is occurring at this temperature.

Contents

1 Introduction

In this project we use the Ising model in two dimensions to simulate magnetic systems. The Ising model is a mathematical model that is frequently used numerically in order to represent magnetic systems and examine phase transitions [2], [5]. Using this model, we create a two dimensional lattice of particles with spins, and examine the properties of the lattice as spins are flipped by the Metropolis algorithm. The Metropolis algorithm is a Markov chain Monte Carlo method which allows for simulating random walks [1], [3], [4]. In our case, the random walks represent spin flips and movement towards the most likely state. As spins are flipped, the energy, magnetic moment, specific heat and susceptibility of the system is computed at different temperatures in order to study phase transitions. The relevant phase transition here is one where the system makes a transition from a magnetic system to a phase with zero magnetisation (the ground state). Without an external magnetic field, the energy of a specific state i is given by:

$$E_i = -J \sum_{<kl>}^{N} S_k S_l \tag{1}$$

Where $s_k = s_l = \pm 1$ are the spins at the different locations in the lattice, and we have a two dimensional lattice of particles with spins up or down. In this project we represent a lattice by a matrix of size NxN with entries +1 and -1, where +1 represents spin up and spin down is represented by -1. J in this equation is the coupling constant expressing the interaction strength between neighbouring spins $< kl >$. We use a finite relatively small matrix to represent a large and sometimes even infinite lattice, and to be able to do this without the errors getting fatal, we use periodic boundary conditions. This means that when we are at the edges of the matrix we treat the particle at the opposite side of the matrix as the neighbour, so that all particles have four neighbours. This concept is illustrated in figure 1.

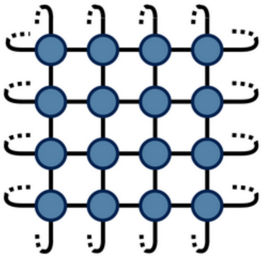


Figure 1: Illustration of periodic boundary conditions. This figure shows how the neighbours at the ends and corners are taken to be the particles on the opposite side of the lattice. The figure is from **http://itensor.org/docs.cgi?page=book/trg.**

We now define some useful quantities that will be used throughout the report. The expectation value of the energy of the system is given by the Boltzmann distribution:

$$< E(\beta) >=< E >= \frac{\sum_{allstates} E_i e^{-\beta E_i}}{Z} \tag{2}$$

Where Z is the partition function, a normalization factor, which is given by:

$$Z = \sum_{allstates} e^{-\beta E_i} \tag{3}$$

The magnetic moment of a specific state i is given by the sum of the individual spins in the system:

$$M_i = \sum S_i \tag{4}$$

The expectation value of the magnetic moment of the system is:

$$< M(\beta) >=< M >= \sum_{allstates} \frac{M_i e^{-\beta E_i}}{Z} \tag{5}$$

Heat Capacity is given by:

$$C_V = \frac{< E^2 > - < E >^2}{k_B T^2} \tag{6}$$

Magnetic susceptibility is given by:

$$\chi = \frac{< M^2 > - < M >^2}{k_B T} \tag{7}$$

1.1 Analytical expressions for a 2x2 lattice

For a 2x2 lattice we have N=2 and a total of 4 spins, so we have a total of 2^4=16 possible micro states. For particles with spins $S_i = \pm 1$ we get can calculate the energies of the different possible states with periodic boundary conditions, using the equation:

$$E = -J(S_1 S_2 + S_2 S_1 + S_1 S_3 + S_3 S_1 + S_2 S_4 + S_4 S_2 + S_3 S_4 + S_4 S_3) \tag{8}$$

The magnetic moment is:

$$M_i = \sum S_i = S_1 + S_2 + S_3 + S_4 \tag{9}$$

We can then easily find the energies and magnetic moments of our 16 different states, which are shown in table 1.

We now calculate the expectation values for the energies and the magnetic moments, as well as other useful quantities. Throughout these calculations we use the shorthand $x = J8\beta = \frac{J8}{k_B T}$, and the following properties of the cosh and sinh functions:

$$e^x + e^{-x} = 2cosh(x) \tag{10}$$

$$e^x - e^{-x} = 2sinh(x) \tag{11}$$

Number of spins up	E	M	Number of microstates
4	-8J	4	1
3	0	2	4
2	0	0	4
2	8J	0	2
1	0	-2	4
0	-8J	-4	1

Table 1: Energies and magnetic moments of the 16 possible micro states for a 2x2 lattice.

The partition function for a 2x2 lattice is:

$$\begin{aligned} Z = \sum_{i=1}^{16} e^{-\beta E_i} &= 2e^{+J8\beta} + 2e^{-J8\beta} + 12 \\ &= 2(e^{+J8\beta} + e^{-J8\beta} + 6) = 2(2cosh(x) + 6) \\ &= 4(cosh(x) + 3) \end{aligned} \tag{12}$$

The expectation value of the energy is:

$$\begin{aligned} < E > = \frac{\sum_{i=1}^{16} E_i e^{-\beta E_i}}{Z} &= \frac{-16Je^{+J8\beta} + 16Je^{-J8\beta}}{Z} = \frac{-16}{Z}(e^x - e^{-x}) \\ &= \frac{-16 \cdot 2sinh(x)}{4(cosh(x) + 3)} = -\frac{8sinh(x)}{cosh(x) + 3} \end{aligned} \tag{13}$$

We will also need the expectation value of the energy squared:

$$\begin{aligned} < E^2 > = \frac{1}{Z}\sum_i E_i^2 e^{-\beta E_i} &= \frac{1}{Z}(8^2J^2e^x + 2 \cdot 8^2J^2e^{-x} + 8^2J^2e^x) \\ &= \frac{8^2 * 2 * J^2}{Z}(e^x - e^{-x}) = \frac{64J^2cosh(x)}{cosh(x) + 3} \end{aligned} \tag{14}$$

The expectation value of the magnetic moment is:

$$< M >= \sum_{i=1}^{16} \frac{M_i e^{-\beta E_i}}{Z} = 0 \tag{15}$$

But our program will calculate $<| M |>$, so we also calculate the value of this quantity in order to compare with our computed value:

$$\begin{aligned} <| M |> = \sum_{i=1}^{16} \frac{| M_i | e^{-\beta E_i}}{Z} &= \frac{1}{Z}(4e^{+8J\beta} + 4 \cdot 2e^0 + 4 \cdot 2e^0 + 4e^{+8J\beta}) \\ &= \frac{1}{Z}(8e^x + 16) = \frac{8(e^x + 2)}{4(cosh(x) + 3)} = \frac{2(e^x + 2)}{cosh(x) + 3} \end{aligned} \tag{16}$$

Again we also need $< M^2 >$:

$$\begin{aligned} < M^2 > &= \frac{1}{Z}\sum_i M_i^2 e^{-\beta E_i} = \frac{1}{Z}(16e^x + 4 \cdot 4e^0 + 4 \cdot 4e^0 + 16e^x) \\ &= \frac{1}{Z}(32e^x + 32) = \frac{32}{Z}(e^x + 1) = \frac{32(e^x + 1)}{4(cosh(x) + 3)} = \frac{8(e^x + 1)}{cosh(x) + 3} \end{aligned} \quad (17)$$

The specific heat of the system is:

$$C_V = \frac{< E^2 > - < E >^2}{k_B T^2} = \frac{1}{k_B T^2}(\frac{64J^2 cosh(x)}{cosh(x) + 3} - (\frac{8sinh(x)}{cosh(x) + 3})^2) \quad (18)$$

And finally, the magnetic susceptibility is:

$$\chi = \frac{< M^2 > - <| M |>^2}{k_B T} = \frac{1}{k_B T}(\frac{8(e^x + 1)}{cosh(x) + 3} - (\frac{2(e^x + 2)}{cosh(x) + 3})^2) \quad (19)$$

We calculated the values of the above quantities with a temperature of 1 in units of J/k_B, so that $x = J8\beta = \frac{J8}{k_B T} = 8$. The resulting values are shown in table 2.

$< E >$	$< E^2 >$	C_V	$< M >$	$< M^2 >$	χ	$<\| M \|>$	$\| \chi \|$
-7.983928	63.871441	0.128329	0	15.973215	15.973215	3.994643	0.01604296

Table 2: Analytic results for expectation values of our numeric experiment for a 2x2 lattice with a temperature of 1 J/k_B.

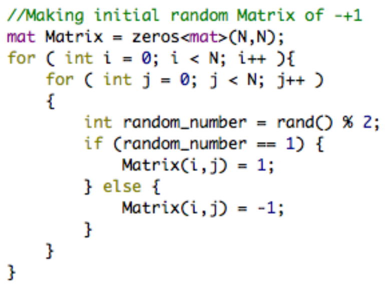
2 Method and Results

All code for this report was written in cooperation with Tiffany Chamandy. The program was written in C++, while plots and figures were made in Python. All programs, figures, text files and data files can be found at: **https://github.com/akspilke/Project4**.

2.1 Generating a Code for the Ising Model

In order to set up the Ising model numerically, it was necessary to create an initial matrix of ±1 to represent the lattice of spins up and down. This was done by initialising an NxN matrix of zeros, before we looped through every row and column and generated a random number of 0 or 1 at each position. The zeros were then changed to -1, and the initial matrix was ready. The code for this can be seen in figure 2.

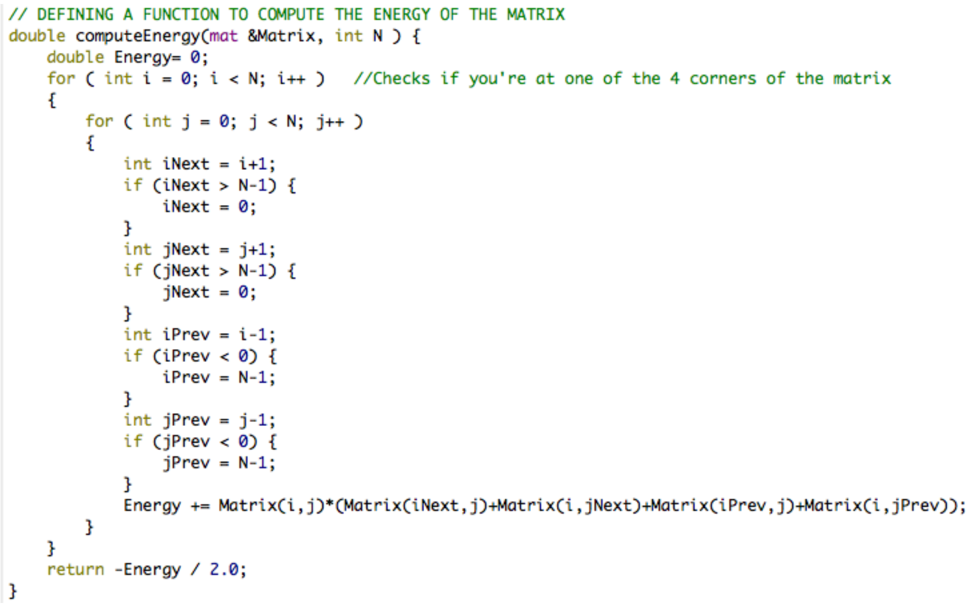
Our next task was to find the energy of the computed matrix, and this was done by making a function. In order to take care of the periodic boundary conditions, we treat the opposite positions in the matrix as neighbours to the ”particles” at the edges. The energy and magnetic moment of the system could then be found as in equation (8) and (9), shown in code in figure 3 and 4.



```
//Making initial random Matrix of -+1
mat Matrix = zeros<mat>(N,N);
for ( int i = 0; i < N; i++ ){
    for ( int j = 0; j < N; j++ )
    {
        int random_number = rand() % 2;
        if (random_number == 1) {
            Matrix(i,j) = 1;
        } else {
            Matrix(i,j) = -1;
        }
    }
}
```

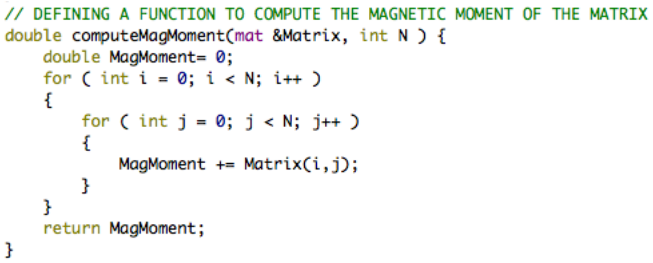
Figure 2: Code for generating initial NxN matrix with components of ±1.

The procedure of the Metropolis algorithm was then followed, and spins were flipped in order to get to the most likely state where the expectation energy, magnetic moment, specific heat and susceptibility of the system can be computed. After one spin is flipped, the new state is accepted if the energy of the state is lower than the previous, or if the quantity $w = e^{(-\Delta E/T)}$ is larger than a random number between 0 and 1. The second condition is included in order to reproduce the random quantum mechanical fluctuations which are known to happen in such a system [7]. The code for this is shown in figure 5. With these methods we computed the same parameters as the ones that were computed analytically, and table 3 gives a comparison between the analytical and computed values.



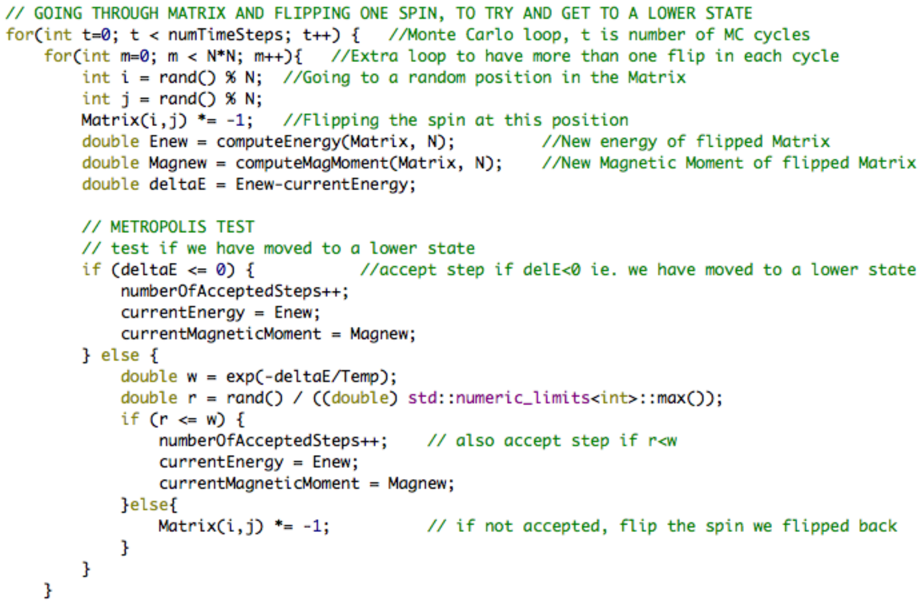
```
// DEFINING A FUNCTION TO COMPUTE THE ENERGY OF THE MATRIX
double computeEnergy(mat &Matrix, int N ) {
    double Energy= 0;
    for ( int i = 0; i < N; i++ )   //Checks if you're at one of the 4 corners of the matrix
    {
        for ( int j = 0; j < N; j++ )
        {
            int iNext = i+1;
            if (iNext > N-1) {
                iNext = 0;
            }
            int jNext = j+1;
            if (jNext > N-1) {
                jNext = 0;
            }
            int iPrev = i-1;
            if (iPrev < 0) {
                iPrev = N-1;
            }
            int jPrev = j-1;
            if (jPrev < 0) {
                jPrev = N-1;
            }
            Energy += Matrix(i,j)*(Matrix(iNext,j)+Matrix(i,jNext)+Matrix(iPrev,j)+Matrix(i,jPrev));
        }
    }
    return -Energy / 2.0;
}
```

Figure 3: Code for computing the energy of the matrix.



```
// DEFINING A FUNCTION TO COMPUTE THE MAGNETIC MOMENT OF THE MATRIX
double computeMagMoment(mat &Matrix, int N ) {
    double MagMoment= 0;
    for ( int i = 0; i < N; i++ )
    {
        for ( int j = 0; j < N; j++ )
        {
            MagMoment += Matrix(i,j);
        }
    }
    return MagMoment;
}
```

Figure 4: Code for computing the magnetic moment of the matrix.



```
// GOING THROUGH MATRIX AND FLIPPING ONE SPIN, TO TRY AND GET TO A LOWER STATE
for(int t=0; t < numTimeSteps; t++) {   //Monte Carlo loop, t is number of MC cycles
    for(int m=0; m < N*N; m++){   //Extra loop to have more than one flip in each cycle
        int i = rand() % N;  //Going to a random position in the Matrix
        int j = rand() % N;
        Matrix(i,j) *= -1;   //Flipping the spin at this position
        double Enew = computeEnergy(Matrix, N);         //New energy of flipped Matrix
        double Magnew = computeMagMoment(Matrix, N);    //New Magnetic Moment of flipped Matrix
        double deltaE = Enew-currentEnergy;

        // METROPOLIS TEST
        // test if we have moved to a lower state
        if (deltaE <= 0) {           //accept step if delE<0 ie. we have moved to a lower state
            numberOfAcceptedSteps++;
            currentEnergy = Enew;
            currentMagneticMoment = Magnew;
        } else {
            double w = exp(-deltaE/Temp);
            double r = rand() / ((double) std::numeric_limits<int>::max());
            if (r <= w) {
                numberOfAcceptedSteps++;    // also accept step if r<w
                currentEnergy = Enew;
                currentMagneticMoment = Magnew;
            }else{
                Matrix(i,j) *= -1;          // if not accepted, flip the spin we flipped back
            }
        }
    }
```

Figure 5: Code for flipping spins in the matrix and accepting movement to lower states.

Quantity	Analytic	Computed (10^4)
Z	5973.9166	-
$< E >$	-7.983928	-7.9896
$< E^2 >$	63.871441	63.9168
C_V	0.128329	0.0830835
$< M >$	0	-
$< M^2 >$	15.969198	15.9828
χ	15.969198	-
$<\vert M \vert>$	3.992634	3.9928
$\vert \chi \vert$	0.0320873	0.0207461

Table 3: A comparison between analytic and computed values for a 2x2 lattice with T=1J/k_B and 10 000 timesteps (Monte Carlo cycles).

2.2 When is the most likely state reached?

In this section, we investigate how long a 20x20 lattice of spins takes to reach the most likely state, the equilibrium state. We study the behaviour of lattices starting with random configurations of spins, versus ordered lattices where all spins start out pointing upwards, at temperatures of 1 and 2.4 J/k_B. Time is here measured in number of Monte Carlo cycles, where NxN spins are flipped in each cycle in order to have less correlation between the measured states. In the end of this section we look at the relation between number of Monte Carlo cycles and number of accepted configurations.

2.2.1 Random matrix, T=1

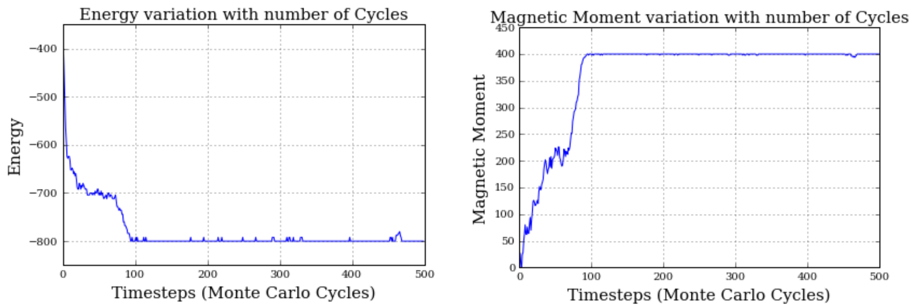


Figure 6: Energy and magnetic moment versus number of timesteps/ number of Monte Carlo cycles for a random matrix with a temperature of 1 J/k_B. It is clear that the system does not start in equilibrium, but that equilibrium is reached just before 100 steps are made.

2.2.2 Ordered matrix, T=1

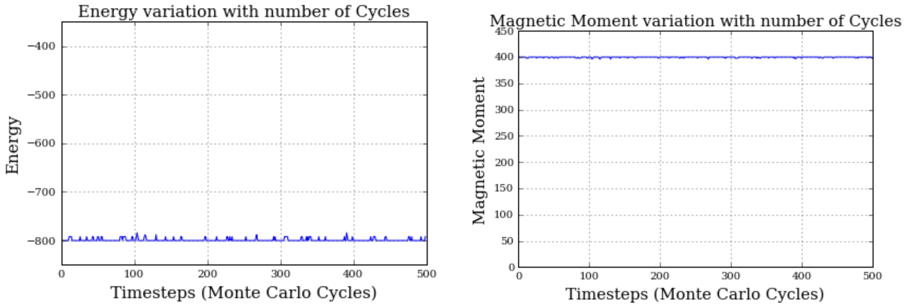


Figure 7: The same figure as above, but now starting with an ordered matrix where all spins are initially pointing up. It seems like equilibrium is reached nearly instantly, and the only changes that are seen in the energy and the magnetic moment are the small quantum mechanical fluctuations [7]. In our numerical experiment, these fluctuations are due to the random number factor in the Metropolis test.

2.2.3 Ordered matrix, T=2.4

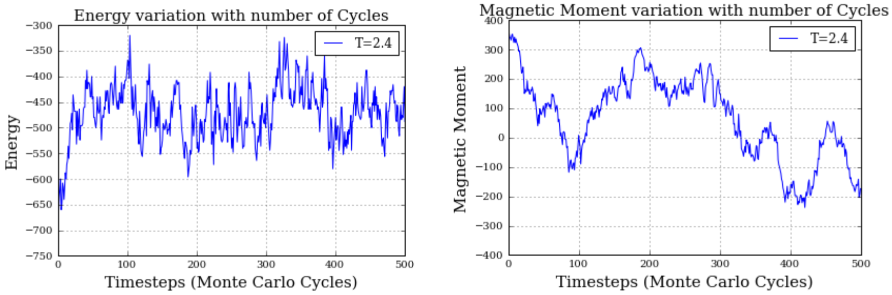


Figure 8: Energy and magnetic moment versus number of timesteps/ number of Monte Carlo cycles for an ordered matrix with a temperature of 2.4 J/k_B. Significant fluctuations are seen, and it is clear that the variations are much larger for this higher temperature, and that the changes in magnetic moment are larger than the ones in energy. It can also be argued that a slightly different trend is present in the first ~20-100 steps.

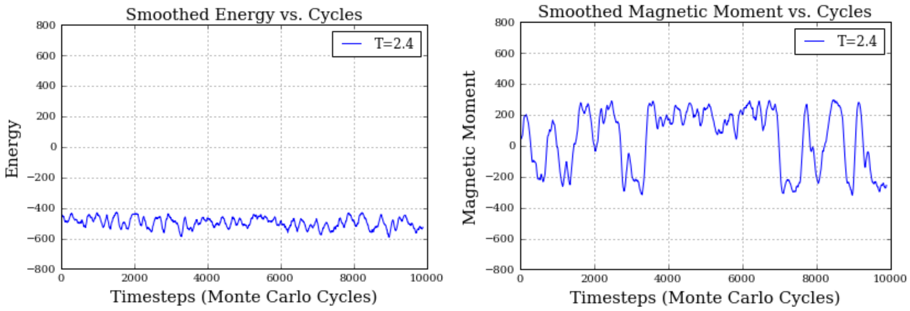


Figure 9: Similar plots as in the above figure, but now over 10000 timesteps and smoothed over 100 timesteps. From this, it is rather hard to determine where equilibrium is reached, as the fluctuations look similar at all steps, and this was the reason we limited to 500 steps in the above figure. It should be noted that the trend seen in the first steps above might just be a random fluctuation due to the thermal energy of the ”particles”.

2.2.4 Random matrix, T=2.4

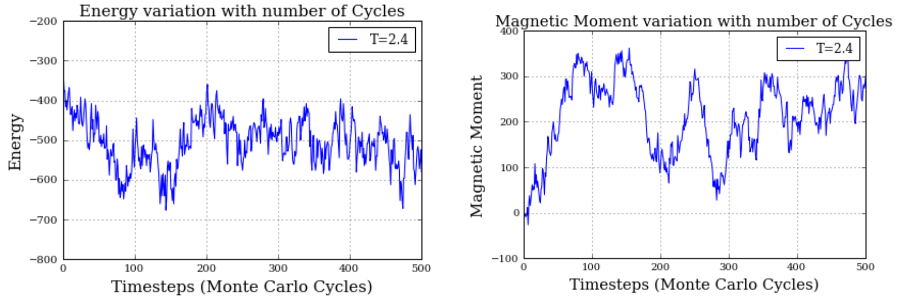


Figure 10: Energy and magnetic moment versus number of timesteps/ number of Monte Carlo cycles for a random matrix at a temperature of 2.4 J/k_B. Again, it is difficult to draw a conclusion as the fluctuations are large, but a trend may be present in the first ~80-100 steps.

2.2.5 When is the most likely state reached?

From the above figures, we conclude that much larger fluctuations are present when the temperature is increased, as is expected from thermodynamics. This makes it more difficult to determine when the system has reached equilibrium, but it can also mean that we can assume equilibrium after only a few steps. It is clear that a lattice of ordered spins reaches equilibrium faster than a random lattice. It may therefore be more efficient to use an ordered matrix, but lattices with ordered spins are less physical as they rarely occur in nature, and we therefore continue to use random matrices.

2.2.6 Accepted configurations versus timesteps

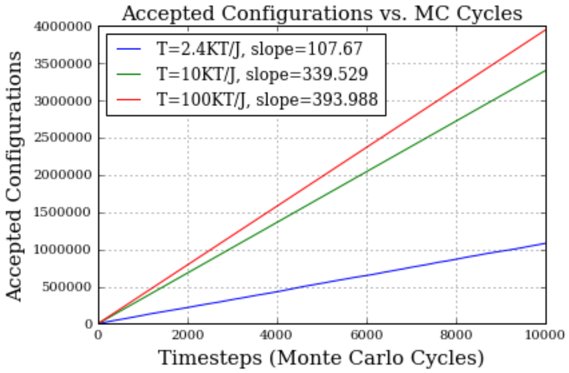


Figure 11: Number of accepted configurations versus number of Monte Carlo cycles for three different temperatures.

Figure 11 shows a perfect linear relationship between accepted spin configurations and number of time steps. We also see that the slope of the line gets steeper with temperature. This is to be expected, as the first condition for accepting a step in the Metropolis algorithm is that it goes to a lower or similar state, and the second is that a random number between 0 and 1 is smaller than or equal to $w = e^{\frac{\Delta E}{T}}$. The first condition is random, and only depends on the random flip of the spins, so should be around 50%. The second condition depends on the temperature, with increasing temperature the value of w increases and the condition that r is smaller than w becomes more likely, explaining the steeper slope for the higher temperature.

2.3 Analysing the probability distribution

In this section the probability distributions of energy for temperatures of 1 and 2.4 J/k_B are compared, see figure 12. In order to produce this figure, the first 1000 time steps were removed to make sure we have reached equilibrium before we start taking data. The figure illustrates that the probability distribution for a temperature of 1 J/k_B is at a lower energy than the 2.4 J/k_B, and the distribution is more localised. This can also be seen from the figures in section 2.2; the lower temperature fluctuates a lot less, around a lower energy. The higher energy for a higher temperature is also expected from thermodynamics, and the relation E $\propto$ k_BT.

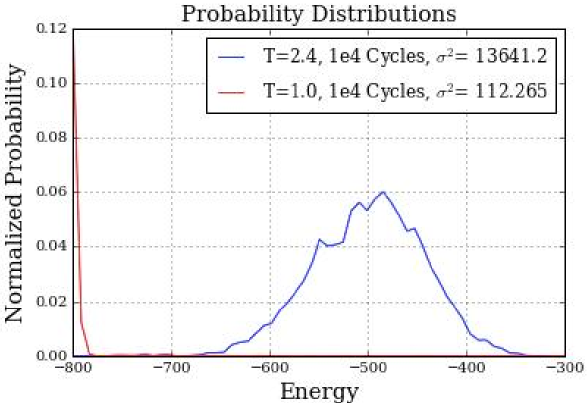


Figure 12: Probability distributions of different energies for temperatures of 1 and 2.4 J/k_B.

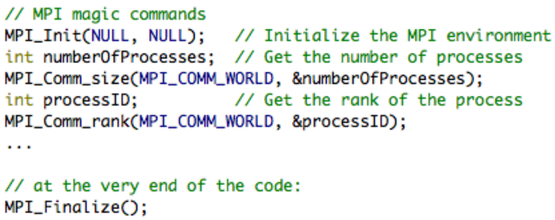
The standard deviation of the probability distributions in figure 12 are displayed in the top right corner of the plot. These were computed in python using the numpy.var function. In our program we calculated the variance using the formula:

$$variance = \sigma^2 =< E^2 > - < E >^2 \tag{20}$$

The values obtained were 146.766 for a temperature of 1, and 3129.43 for a temperature of 2.4. There is some deviation here, but our methods have some errors, so some differences are to be expected.

2.4 Numerical studies of Phase Transitions

In order to study the behaviour of our expectation values, specific heat and susceptibility around the critical temperature, a temperature loop was inserted outside the Monte Carlo loop in figure 5. This loop contained 32 temperature steps between T=2.0 and 2.4J/k_B, so ΔT=0.0125 J/k_B. We studied the the dependency of temperature for matrices with $N = 40, 60, 100$ and 140. For the 140x140 matrix we used parallellisation with MPI in order to speed up the computation of our program. We also tried to run our code for this matrix without parallellisation as we could not get the parallellisation to work for a long time, but this took over 17 hours in CPU time and still did not finish. When we finally got the parallellisation to work however, we ran on 4 cores and it took 23.7 CPU minutes: a huge improvement. For the parallellisation, we needed to include the four magic MPI commands shown in figure 13 in our code, and compile in terminal with optimisation flag using the command **mpic++ -o main -O3 main.cpp** before running with the command **mpirun -n 4 main**, where main is our main.cpp file. The result of our calculations are displayed in figures 14 and 15.



```
// MPI magic commands
MPI_Init(NULL, NULL);   // Initialize the MPI environment
int numberOfProcesses;  // Get the number of processes
MPI_Comm_size(MPI_COMM_WORLD, &numberOfProcesses);
int processID;          // Get the rank of the process
MPI_Comm_rank(MPI_COMM_WORLD, &processID);
...

// at the very end of the code:
MPI_Finalize();
```

Figure 13: Magic code lines necessary to run in parallellisation with MPI.

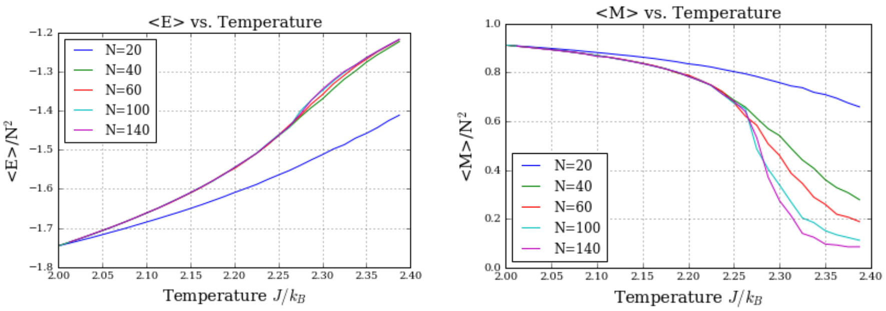


Figure 14: Expectation values over number of spins for energy (left) and magnetic moment (right) plotted against temperature.

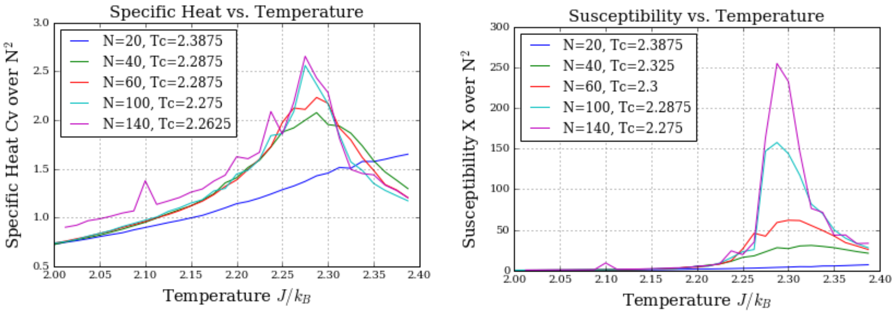


Figure 15: Specific heat (left) and magnetic susceptibility (right) over number of spins, plotted against temperature.

From the figures in this section, we see clear peaks in specific heat and susceptibility around a temperature of 2.3 J/k_B for the matrices larger than 20x20 spins, showing clear indications of a second order phase transition. For the 20x20 lattice, we see no indication of any phase transition and conclude that larger lattices are needed to see the transition. The peak also becomes sharper with larger lattices, making the advantage of large lattices and parallellisation obvious. As indicated in the top left legends in figure

15, the critical temperature, here taken to be the highest point on the graphs, moves towards lower temperatures with larger lattices.

The reason for this behaviour is that we are representing an infinite lattice by a relatively small finite system, and as we move closer to the critical temperature the relatively few spins of the system become more and more correlated, dependent of the lattice size. This correlation is quantified by the correlation length, which increases close to the critical temperature:

$$\xi(T) \sim\mid T_C - T \mid^{-\nu} \tag{21}$$

Through finite scaling relations the behaviour of finite lattices can be related to infinitely large lattices:

$$T_C(N) - T_C(N = \infty) = aN^{-1/\nu} \tag{22}$$

In these equations a is a constant and ν is defined by (21). By using the ν defined here, and the parameters α=0, β=1/8 and γ=7/4, we also get a power law dependence on the lattice size for the expectation value of the magnetic moment, specific heat and magnetic susceptibility:

$$< M(T) >\sim (T - T_c)^{\beta} \to N^{-\beta/\nu} \tag{23}$$

$$C_V(T) \sim\mid T - T_c \mid^{-\alpha} \to N^{\alpha/\nu} \tag{24}$$

$$\chi(T) \sim\mid T - T_c \mid^{-\gamma} \to N^{\gamma/\nu} \tag{25}$$

These power law dependencies are seen in the figures above. We note that the expectation energy is not listed to have a dependency on N here, and this is understood from the left panel in figure 14. In this figure, we see minimal changes of $< E > /N^2$ with the change of N. This is because the energies of the lattice are uncorrelated, they are not affected by their neighbours and remain a similar value total per spin for any lattice size. For the expectation value of the magnetic moment, we see that the curves follow each other closely until they come close to the critical temperature and begin to deviate. This can be explained by equation (23); the decline is steeper with larger lattice size. In physical terms this means that the phase transition happens faster and is more clearly defined for larger lattices. Larger lattices are also closer to real situations, where phase transitions often happen nearly instantly, such as the freezing or vapourisation of water at 0 or 100 degrees.

The specific heat and magnetic susceptibility have clear peaks around the critical temperature (see figure 15), and the peak here becomes sharper with higher number of spins included in the system. Again this can be explained by the lattice more closely approximating a real situation, requiring more heat to be added to the system in order to change the temperature and let the lattice go through the phase transition. The increment in the slopes of the lattices containing a higher number of spins can also be explained by equations (24) and (25).

2.5 Extracting the critical temperature

From figure 15, we see that the peak indicating the phase transition and the critical temperature moves towards lower temperatures for larger lattices. This is also displayed in table 4 for closer inspection.

Lattice size NxN	$T_c(C_V)$	$T_c(\chi)$
40x40	2.2875	2.325
60x60	2.2875	2.3
100x100	2.275	2.2875
140x140	2.2625	2.275
Average	2.2781	2.2969

Table 4: A comparison between the critical temperatures (x values of peaks in figure 15) found from specific heat and magnetic susceptibility for different lattice sizes.

The real value for the critical temperature of the system is $k_B T/J = 2/ln(1 + \sqrt{2}) = 2.269$ from Lars Onsager [6]. Our computed values are rather close to this, especially for the 140x140 lattice, and we conclude that larger lattices give a better result, but that our relatively small lattices don't do a bad job at estimating the critical temperature of this Ising model system.

3 Analysis and Conclusion

In this project, we have investigated energy, magnetic moment, specific heat and magnetic susceptibility of matrices of ±1 as numbers are flipped. This system represents an Ising model crystal lattice of NxN particles of spins up and down and its movement towards equilibrium/ a steady state. The dependency of lattice size and temperature were investigated for random and ordered matrices, ultimately with the goal of studying phase transitions and the critical temperature.

We found that higher temperatures give a larger scatter in energy and magnetic moment, as the particles have larger velocities and the quantum mechanical fluctuations/vibrations are larger. In our study, equilibrium is reached faster for ordered matrices than for random matrices, but since lattices of all spins pointing in one direction are not very likely in nature, we use random matrices throughout the project.

We see that matrices larger than 20x20 are necessary to see the phase transition, and that larger matrices give clearer indications and better estimates for the critical temperature of the system. The reason for this is that they resemble the physical situation. For the critical temperature, we computed an average of the values found from matrices of 40x40, 60x60, 100x100 and 140x140, and find a value of 2.2781 J/k_B using the specific heat, and 2.2969 J/k_B using the magnetic susceptibility. At this temperature clear indications of a second order phase transition are seen. The values found are rather close to the accepted value of 2.269 J/k_B. From our data, we can infer that larger matrices give results closer to real values, but that significant time, parallellisation or powerful computers are necessary to treat such matrices.

In future projects, larger matrices could be treated with more temperature steps and more time steps in order to produce increasingly accurate results. This could be done by parallellising on more than 4 cores, by using a computer cluster such as Computational Physic's Smaug. We attempted this in this project, but were unsuccessful as the deadline came close.

Through completion of this project we have learnt that quantum mechanical experiments can be accurately treated with computer simulations, and that the advantages of parallellisation and supercomputing can have important consequences for our ability to use numerical experiments to understand the world.

References

[1] Siddhartha Chib and Edward Greenberg. Understanding the metropolis-hastings algorithm. *The american statistician*, 49(4):327–335, 1995.

[2] Roy J Glauber. Time-dependent statistics of the ising model. *Journal of mathematical physics*, 4(2):294–307, 1963.

[3] Heikki Haario, Eero Saksman, and Johanna Tamminen. An adaptive metropolis algorithm. *Bernoulli*, pages 223–242, 2001.

[4] W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[5] Tsung-Dao Lee and Chen-Ning Yang. Statistical theory of equations of state and phase transitions. ii. lattice gas and ising model. *Physical Review*, 87(3):410, 1952.

[6] Lars Onsager. Statistical hydrodynamics. *Il Nuovo Cimento (1943-1954)*, 6:279–287, 1949.

[7] Eugene Wigner. On the quantum correction for thermodynamic equilibrium. *Physical review*, 40(5):749, 1932.