Summary of a 2D Ising model simulation

Florian Rappl

Department of High Energy Physics, University of Colorado at Boulder, 80309, CO, USA and

Institut für Theoretische Physik, Universität Regensburg, D-93040 Regensburg, Germany

with help and instructions from Tom DeGrand

Department of High Energy Physics, University of Colorado at Boulder, 80309, CO, USA

(Dated: December 18, 2010)

Abstract

The Ising model has been a subject for research since it's outcome. In the era of Information Technology and with better computational abilities the simulations based on a lattice, using the Ising model, have been increased a lot. The basic task I had to fulfill was to get familiar with the Ising model and do some basic research on it using a Metropolis algorithm - basicly a Monte-Carlo-Simulation. Even though Heat-Bath algorithms are more optimized and professional, and Cluster algorithms being faster and fancier, it was a good beginner's lesson and confronted me with a lot of problems and interesting behaviors. The following text gives some brief introduction into what my program did and how results have been obtained from this, i.e. the evaluation of the the system's behavior.

I. INTRODUCTION

The Ising model has been introduced in 1925 by Ernst Ising and Wilhelm Lenz to do some research on Ferromagnetism. It is one of the most researched models of statistical physics. For Ferromagnetism we let the Hamiltonian be

$$\hat{H} = -J \sum_{\langle i,j \rangle} s_i s_j, \tag{1}$$

with the nearest neighbors $\langle i, j \rangle$ and the value of each $s_i = \pm 1$. The variable J is called the exchange integral or exchange coupling. We know that

- $J > 0$ for a ferromagnetic ground state (i.e. all spins in one direction) and
- $J < 0$ for a antiferromagnetic ground state (i.e. spinsum is zero).

This Hamiltonian is kind of familiar, since we already know from Quantum Mechanics that Coulomb interaction and Pauli princple lead to

$$\hat{H} = -J \vec{s}_1 \cdot \vec{s}_2. \tag{2}$$

This is called the Heisenberg Hamiltonian (for two electrons). Thus the Ising model is a simplified classical version. The main task for the program was to get most thermodynamic quantities and observe interesting statistical phenomenas. For the basic algorithm we take the so called *Metropolis* algorithm. Tom DeGrand (2006)[1] describes in his book that this method proposes a change $\Phi \to \Phi'$ with a probabilitiy $Q(\Phi' \leftarrow \Phi)$, such that the reverse transition has the same probability,

$$Q(\Phi' \leftarrow \Phi) = Q(\Phi \leftarrow \Phi'). \tag{3}$$

The next step is to pick a random number r distributed uniformly on the interval $[0, 1]$ and to determine wether the change should be accepted under the following conditions:

1. If $S(\Phi') < S(\Phi)$ then we always accept the change.
2. Otherwise we accept the change if $\exp(-i\Delta S) > r$.

Therefore this is often called “accept/reject” condition.

II. THERMODYNAMIC QUANTITIES

From our physical simulation we want to obtain some relevant data. As the main variation parameter we use the variable β, which is given as $1/(k_B T)$. Additionally we say that β is in units of J as well - so that we do not need to set up any J for our system. Generally the energy, the expectation value of the Hamiltonian, is the most important quantity to measure. Additionally since this is a ferromagnetic system we want to know the magnetization of the currently generated configuration as well as it's derivative, the magnetic susceptibility. For a thermodynamic system the heat capacity, which is basicly the derivative of the internal energy of the ensemble, is also very important and will give us some insight on the system's behavior.
The energy is given by

$$E = \langle \hat{H} \rangle = \frac{1}{Z} \mathrm{Tr} \left[\hat{H} \exp(\beta \hat{H}) \right] = -J \sum_{\langle i,j \rangle} s_i s_j, \tag{4}$$

where Z is the partition function of the system, i.e. the summation over all possibilities. The magnetization can be computed over the spin expectation value, since we know that a system with all spins in one direction is completly ferromagnetic (=1), wheras a configuration with half of the spins up and half of the spins down is antiferromagnetic (=0). We write this as

$$M = |\langle S \rangle| = \left| \frac{1}{N_L} \sum_{i=1}^{N_L} s_i \right|, \tag{5}$$

where N_L is the number of sites in the system, e.g. for a 16 square grid we have $N_L = 16^2 = 256$ sites. In the end the N_L factor won't matter since we will divide all of our quantities by N_L - to get a better comparison to other lattice sizes. The magnetic susceptibility is derived by

$$\chi = \beta \left(\langle M^2 \rangle - \langle M \rangle^2 \right) = \frac{\partial M}{\partial \beta} = \frac{\partial |\langle S \rangle|}{\partial \beta}. \tag{6}$$

For the heat capacity we can calculate

$$c_V = \beta^2 (\langle E^2 \rangle - \langle E \rangle^2) = \frac{\partial E}{\partial \beta} \frac{\partial \beta}{\partial T} = \frac{\partial E}{\partial T}. \tag{7}$$

We will use this overview of basic thermodynamic quantities to evaluate the output of our program.

III. THE PROGRAM

For some simplification purposes as well as some comfort in terms of visualizing data the decision for the programming language was C#. Later on bigger problems for the computation cluster will

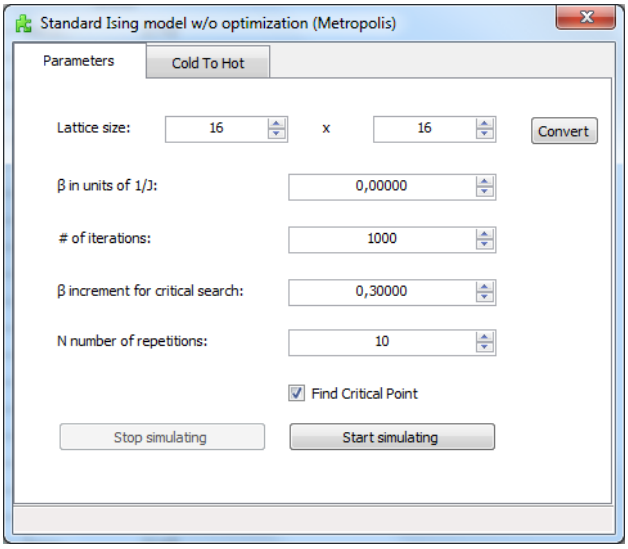

FIG. 1: The main program screen

be written in C - but as long as the program purpose is just playing around and getting used to the basic simulation algorithms this language offers a lot of advantages. Fig. 1 shows how the program looks like when the "Find Critical Point" box is checked. The program makes a difference according to the status of this checkbox. Basicly it can be summarized to:

- **Not Checked** - the β_S increment parameter as well as the number of repetitions parameter is invisible and do not play any role. The β_S parameter is used as the kind of energy the system is having - all decisions will be based on the value entered for β_S. The number of iterations are used to change the system. If there are enough iterations the system will end fluctuating around it's equilibrium for that specific $\beta = \beta_S$ value. Note that the required number of iterations is dependent on β.

- **Checked** - the β_S parameter is used a maximum point to reach for the β value. The starting value for β is set to 0. Then the β increment parameter b will be used to give the number of β increments B, given by

$$B = \lceil \beta / b \rceil \, . \tag{8}$$

  The iteration number will be used at each temperature for heating up the system. The best number of iterations is the number, which is enough to bring the system to its equilibrium for any value of β_S in the intervall $[0, \beta]$. To confirm this the number of repitions N is used. After this one warm-up run with the full iteration number I, the program will run $N - 1$ time $1/10$ of the iteration number and make statistics about it. Thus we can see for example

the heat capacity as well as taking mean values for the energy and other values. Therefore this gives us a better precision and knowledge about our system. The total iterations I_{tot} per β value is thus given by

$$I_{\text{tot}} = \frac{I \cdot (9 + N)}{10}. \tag{9}$$

After a complete run all the data can be seen in graphs, displayed by the program and accessed over the so called tabpages. Another possibility is to save the data in a textfile, which can be imported to programs like qtiPlot.

A. The basic algorithm

The basic decision is which kind of algorithm to pick. Since this project was intended to be for beginners it was useful to pick a Monte-Carlo-Algorithm, i.e. an algorithm based on random numbers. In this case a Monte-Carlo-Algorithm generates a sequence of random field configurations $\Phi^{(k)}$ with a probability distribution tailored to the measure of

$$P(\Phi) \propto \exp\left(-S(\Phi)\right), \tag{10}$$

where S is the action of our system. A more detailled explanation can be found in DeGrand (2006)[1]. According to DeGrand (2006)[1] the two most common algorithms of the Monte-Carlo category are the so called Heat Bath method and the method of Metropolis *et al.* (1953). Even tough it seems more effective to write a well programmed Heat Bath algorithm the decision here was to write everything with the Metropolis method. The main difference between those two algorithms is that

- the Heat Bath method visits each of the lattice sites while the Metropolis method visits randomly picked lattice sites,
- the Heat Bath transition is independent of the starting configuration and
- the Heat Bath method is becoming more efficient for local, simpler actions.

The main iteration loop looks basicly like the following piece of programming code, with r being an instance of a random generator class and iterations being the number of iterations I. The quantitites L_x and L_y give the number of lattice points in x and y direction. This results in the total number of lattice points being

$$N_L = L_x \cdot L_y. \tag{11}$$

```
int [,] s = new int[Lx,Ly]; //our lattice grid - a two dimensional integer array
 initialize_ising (s); //function to generate a starting configuration
for ( int i = 0; i < iterations; i++)
{
  int rx = r.Next(0,Lx); //including 0 as lowest number and excluding Lx as highest
  int ry = r.Next(0,Ly); //including 0 as lowest number and excluding Ly as highest
  int delta = 2 * deltaU(s, rx, ry); //the energy diff if the spin will be reversed
  double p = r.NextDouble(); //gives us a random number for our decision

  if (delta <= 0 || p <= exp(-(double)delta * beta)) //the switching decision
    s[i, j] = -s[i, j]; //switches the spin 1<->-1
}
```

The question which instantly arrises is the factor of 2 in the calculation of the energy difference. The code for the deltaU function is the following:

```
int deltaU(int [,] s, int x, int y)
{
  int left = x == 0 ? s[Lx - 1, y] : s[x - 1, y]; //left neighbor or most right point
  int right = x == Lx - 1 ? s[0, y] : s[x + 1, y]; //right neighbor or most left point
  int top = y == 0 ? s[x, Ly - 1] : s[x, y - 1]; //top neighbor or most bottom point
  int bottom = y == Ly - 1 ? s[x, 0] : s[x, y + 1]; //bottom neighbor or most top point
  return s[x, y] * (top + bottom + left + right);
}
```

We can see that by not taking the factor of 2 into consideration, other lattice sites would be left out since we only look at the neighbors of one site, but would not take into consideration that this one site is also a nearest neighbor to four other sites. By multiplying with 2 we take that into consideration. This can be seen by doing some trivial mathematics,

$$\Delta U = s_{i,j} \cdot (s_{i-1,j} + s_{i+1,j} + s_{i,j+1} + s_{i,j-1}) \tag{12}$$

$$+s_{i-1,j} \cdot s_{i,j} + s_{i+1,j} \cdot s_{i,j} + s_{i,j+1} \cdot s_{i,j} + s_{i,j-1} \cdot s_{i,j} \tag{13}$$

$$= 2s_{i,j} \cdot (s_{i-1,j} + s_{i+1,j} + s_{i,j+1} + s_{i,j-1}) . \tag{14}$$

For creating some statistics the main iteration loop is looped again for each β and in this loop again looped for the number of repetitions.

B. Development of a hot vs a cold system

1. *Very high temperature*

The first test for our system was to show the development of a cold system (all spins in one direction - ordered) vs a hot system (random configuration). We expect that the hot system has a spin expectation value of $\langle S \rangle = 0$ due to our random generator producing random numbers which are distributed uniformly. We first look at the extreme case of $T \to \infty$.

Fig. 2^{10} shows the development of our two test systems with a β value of 0. This means we have

FIG. 2: Development of a cold system (1st row) vs a hot system (2nd row) at $\beta_S = 0^9$

very high temperature in this system.

A guess would be that the cold systems transforms pretty quickly to a random system, having a $\langle S \rangle$ of 0 as well, while the hot system stays in its random state - just fluctuating around a spin expectation value of 0. This development could also be shown in a $\langle S \rangle(i)$, where i is the number of sweeps, diagram.

2. *Zero temperature*

Another extreme case would be $T \to 0$. Here we would expect that a cold system stays cold and a hot system would either converge in one direction

$$\langle S \rangle = \pm 1, \tag{15}$$

or "freeze" in its state, depending on the random configuration.

- A stable, freezing out configuration would be anti-ferromagnetic ($\uparrow\downarrow\uparrow\downarrow$...),
- while unstable converging states would have larger domains of spin-ups and spin-downs.

Even though I have observed all three cases,

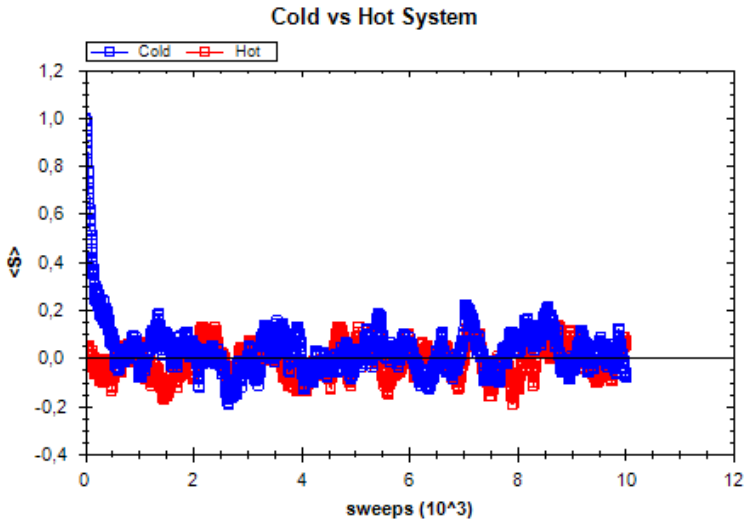

FIG. 3: The spin expectation value in dependence of the number of sweeps for $\beta_S = 0^9$

FIG. 4: Development of a cold system (1st row) vs a hot system (2nd row) at $\beta_S = 10000^9$

1. hot system goes to $\langle S \rangle = -1$,
2. hot system goes to $\langle S \rangle = +1$ and
3. hot system goes to $\langle S \rangle \approx 0$, i.e. "freezing" out,

I decided to show the case where the hot system is converging to become a duplicate of our cold system in fig. 4^{11}.

The higher the β value the less iterations were required in order to have two cold systems instead of one. The minimum required iterations is most likely to become the same number as in the $\beta = 0$ case for $\beta = \infty$, which could not be done precise enough numerically but analytically by just replacing the function

$$\lim_{\beta \to \infty} \exp(-\Delta\beta) \approx \Theta(-\Delta) = \begin{cases} 1, & \Delta \leq 0 \\ 0, & \Delta > 0. \end{cases} \tag{16}$$

One could argue that for a value of $\Delta < 0$ we have $\exp(\infty)$ which is certainly bigger than one. This is true for sure but not interesting in our case, since we are only interested in a probability.

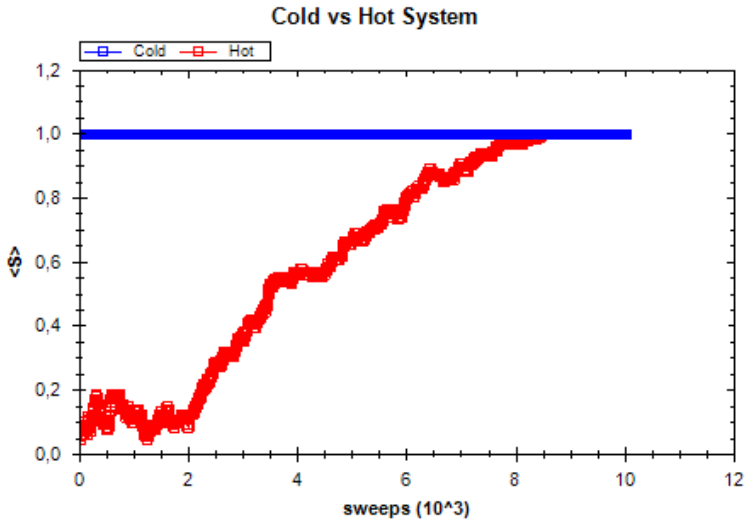

FIG. 5: The spin expectation value in dependence of the number of sweeps for $\beta_S = 10000$[9]

If the outcome is bigger than one it has the same meaning for use as if it would be exactly one. For the outcome of 0 we do not change the spin, but for a value $P \geq 1$ we change it.

3. The critical temperature

The third extreme case is much harder to find than the first ones - and is basicly what our search and the simulation is about. We call this third one the *critical point* and define T_C or β_C at it. Fig. 6[12] shows the development at the critical point known from literature, e.g. L. Witthauer and M. Dieterle (2007)[7].

Although it can be found through statistics - and for the 2D Ising model also analytically by Lars

FIG. 6: Development of a cold system (1st row) vs a hot system (2nd row) at $\beta_S = 0.4407$[9]

Onsager (1944)[8] - it has some properties which give us a hard time. First of all the correlation length becomes infinite at the critical point. At this point it is in a way true to say that everything is possible. I took some snapshots of the cold and hot systems converging to one system in the middle of $\langle S \rangle_C - \langle S \rangle_H$.

The graph in fig. 7 could provide the wrong impression that both systems easily converge. Since

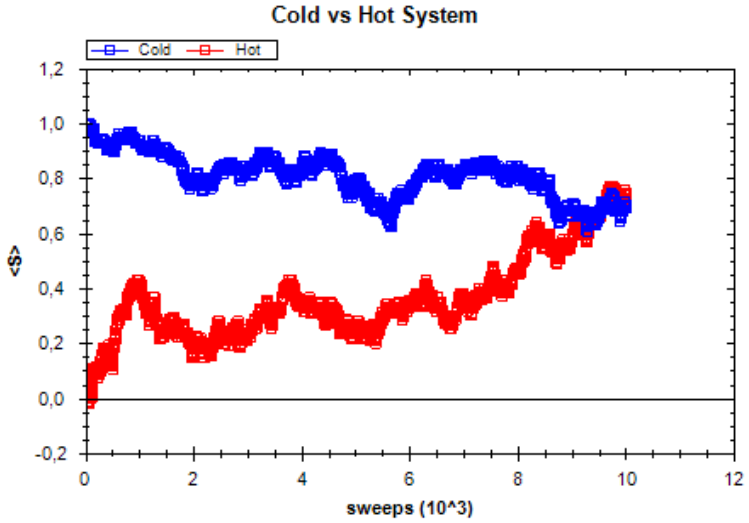

FIG. 7: The spin expectation value in dependence of the number of sweeps for $\beta_S = 0.4407$[9]

the correlation length at the critical point goes to infinity the process shown is highly unstable and can only be observed for a certain iteration number having done several simulations.
The more interesting process is a certain system (let's say we use our formerly known cold system) developing with increasing β. This can be used to get as much data as possible and create all the statistics we already talked about. We will use this to find the critical point on our own.

C. Data created by the program

For finding the critical point the program was designed to give us directly the output in form of useful diagrams - namely the energy expectation value U, the magnetization expectation value M and the specific heat C. We already know that the specfic heat is the variance of the energy, i.e. the Hamiltonian, from equation 7. To calculate this we use the technique to get as many energy and energy^2 datapoints as possible. After we are finished with a certain value for β and before we move on to the next value of β, we make our statistics based on these numbers.

Fig. 8 shows a comparison of $c_V(\beta)$ on some different lattice sizes. We can see that from the 16×16 lattice to the 32×32 lattice the peak increases, while the FWHM gets smaller. If the iteration number would have been high enough for the other lattice sizes, i.e. 48×48, 64×64 etc., we would have observed even higher peaks. Therefore we can only note that the number of iterations for the bigger lattices was not high enough. The used number of iterations for all measurements was 50000, i.e. 50000 for the warm-up-run at each value for β, then 5000 for taking statistics. In fig. 9

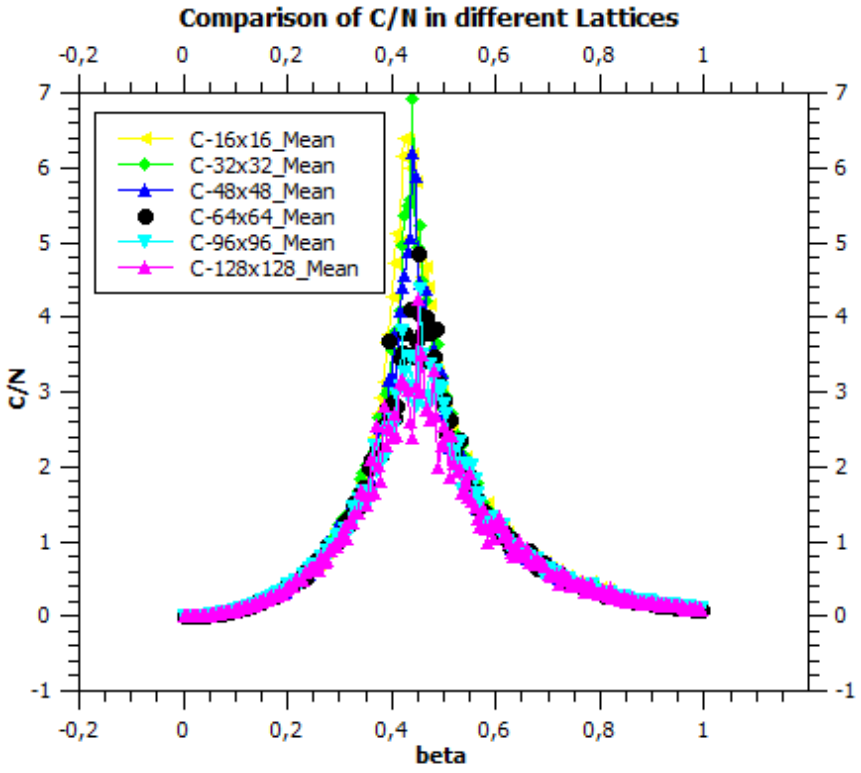

FIG. 8: Comparison of the $c_V(\beta)$ curve observed under different lattice sizes

we see that the critical temperature can be found by extrapolating the measurements on different lattice sizes. In the thermodynamic limit of $N \to \infty$, which would be in our case $L^2 \to \infty$ or $1/L^2 \to 0$, we would obtain an exact value for T_C. The graph shown can not be used for a precise number of T_C since

- we have seen in fig. 8, that our number of iterations was too low for measuring lattice sizes of 48×48 and higher and
- we increased β only by 0.005 per run. For a high precicion number we would need at least 0.001 per run.

In this case the value of T_C would have been (by extrapolating)

$$T_C = (2.26 \pm 0.04)\ 1/\mathrm{J}. \tag{17}$$

This is quite a good value thinking of the circumstances (e.g. not enough iterations). The literature value is 2.2693 1/J according to L. Witthauer and M. Dieterle (2007)[7]. To show the capabilities of the program we let it run with 100000 iterations in a range from $\beta = 0$ to $\beta = 0.9$. To choose a

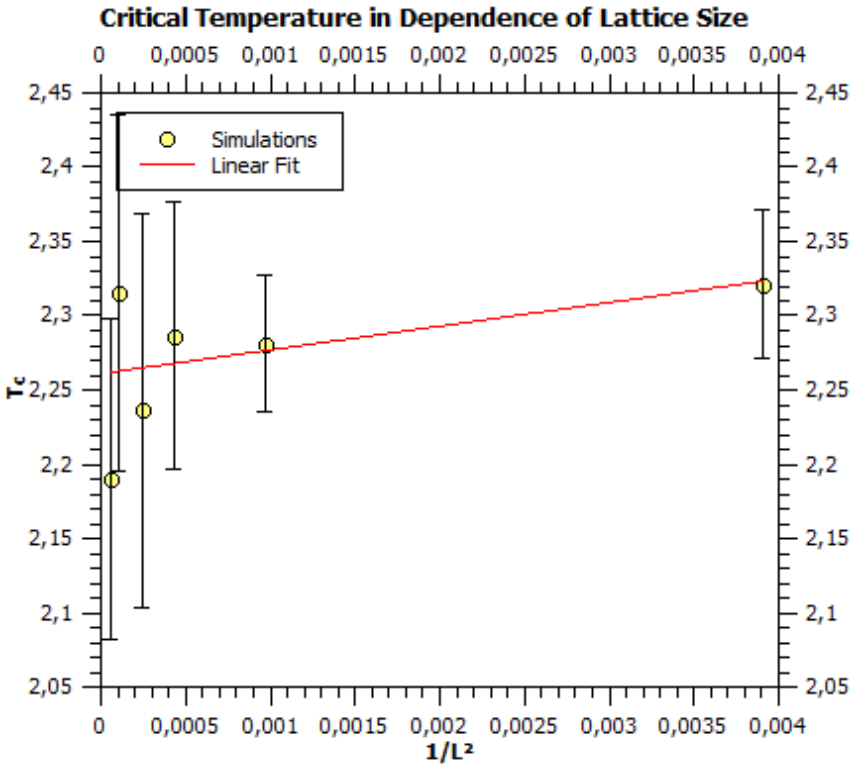

FIG. 9: Different values for $T_C = 1/\beta_C$ depending on the lattice size L

different lattice size we pick 32×32. We already know that the iteration number is high enough for this lattice size - since less iterations already did a good job. We set the number of repetitions to 50 - to have a quite high accuracy. The β increase per run is again 0.005.

In fig. 10 we see that at the critical point we have a quite remarkable behavior. If $\beta < \beta_C$ we see that that the systems tend to be antiferromagnetic - it even converges to this. If $\beta > \beta_C$ we have the opposite: the system converges to be ferromagnetic. We can directly see some really big jumps around β_C. This prooves again two things:

- Around β_C we have to do a slower β increase, i.e. not increasing β by 0.005 but for instance by 0.0005 or even less.

- Since $\chi = dM/d\beta$ we see the famous jump in χ at this point. This can be used to calculate the critical exponent. We will discuss this later on.

The next statistics is about the energy of the system. The energy is very useful quantity which gives us information about many properties, since it is the only preserved quantity, constraining the phase space (in addition to the volume V, number of particles N). This is because large systems

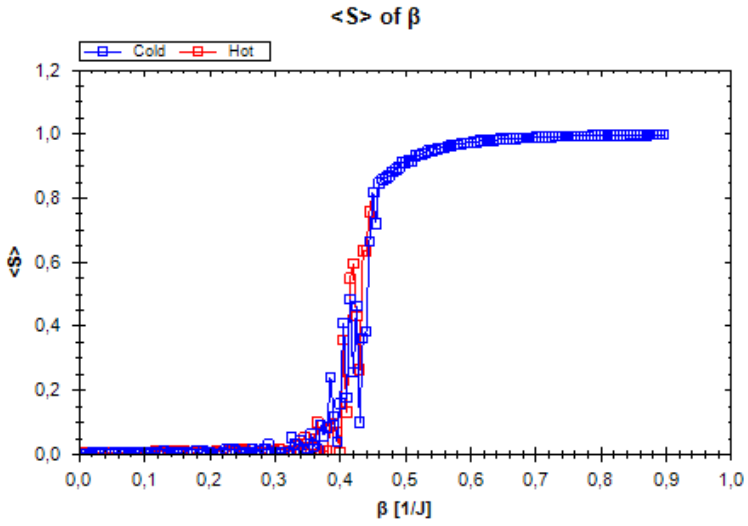

FIG. 10: The spin expectation value in dependence of β for a 32×32 lattice[9]

"forget" their initial conditions, i.e. all other integrals of motions are "forgotten".

The energy graph shown in fig. 11 gives us again an important hint for β_C.

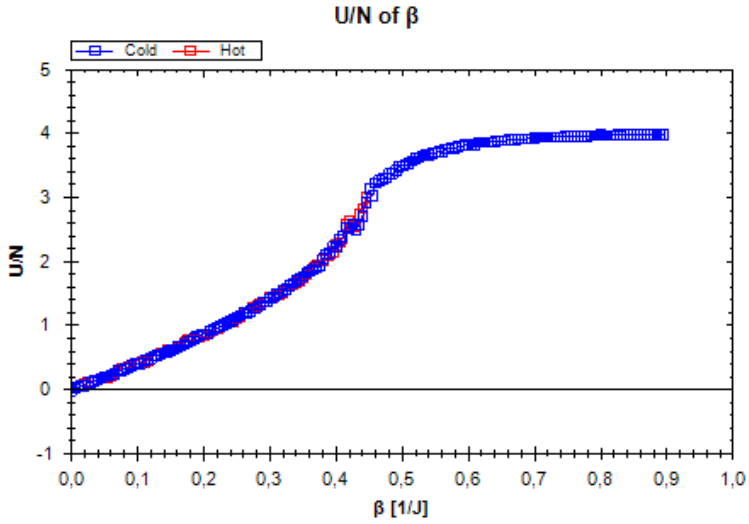

FIG. 11: The energy value in dependence of β for a 32×32 lattice[9]

At β_C we have an inflection point meaning that the second derivative is zero. That means that the inverse of the second derivative is infinite - which results in a phase transition of the second order. A first order phase transition would have an extrem point in the energy at β_C, i.e. the first derivative would be zero. We see again that around β_C a slower heating up is required in order to have a high precision, since there are again larger jumps visible.

The last quantity which can be directly seen with the program is the specific heat (e.g. for χ

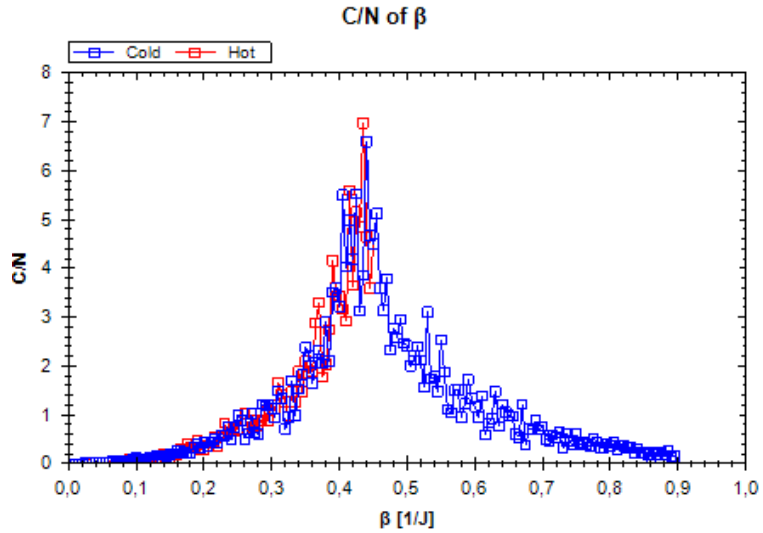

FIG. 12: The specific heat in dependence of β for a 32×32 lattice[9]

there is no inbuilt included - but we can take the derivative of the spin expectation value). The specific heat also shows some remarkable behavior at the critical point - and is very important for calculating the critical exponents. We can clearly see in fig. 13 that for enough iterations we get

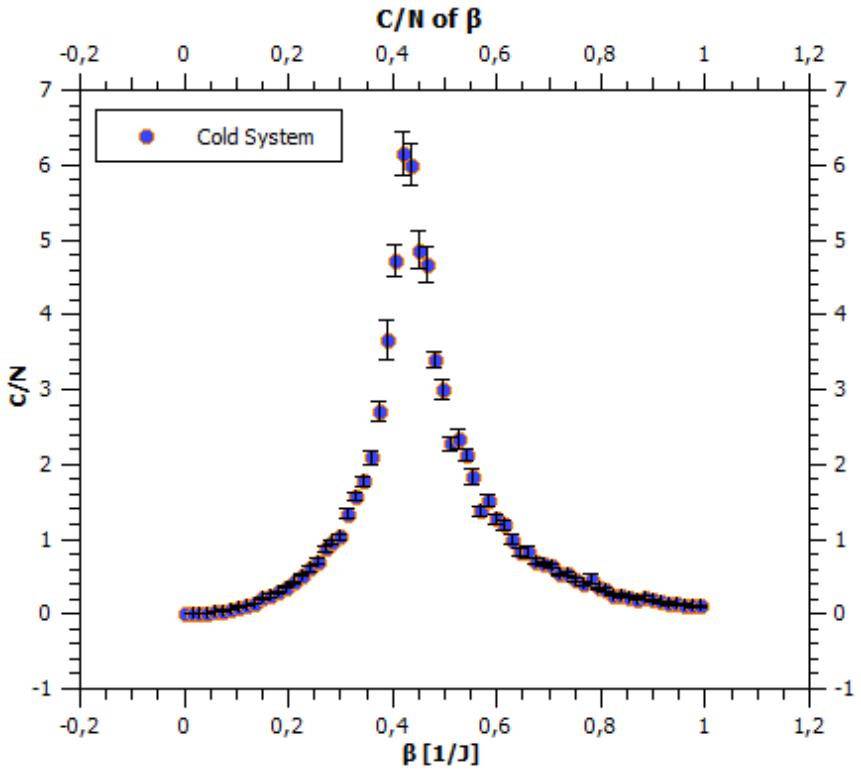

FIG. 13: The specific heat in dependence of β for a 16×16 lattice

large errors around β_C and small errors in the limits of $\beta \to 0$ or $\beta \to \infty$. This means that slower, i.e. $\Delta\beta \to 0$ or additional runs around β_C are required in order to get precise values in that region.

D. Critical exponents

Following the work of Cardy (1996)[6] I tried to understand the theory of finite size scaling. Basicly the parameter γ - called critical exponent - is one of the most important quantities for our system. The most important property of this parameter is that it is scaling invariant - meaning it has no L dependence. This can be seen through the following equations. According to Cardy (1996)[6] we need to have a parameter ν which is given by the FWHM of e.g. our specific heat. We therefore set $\Delta x \equiv$ the FWHM. We have

$$L^{-1/\nu} = \Delta x. \tag{18}$$

In order to get ν we only have to take the logarithm and obtain

$$\nu = -\frac{\ln L}{\ln \Delta x}. \tag{19}$$

If we now want to calculate the γ-parameter we have to calculate

$$L^{\gamma/\nu} = x_m, \tag{20}$$

where x_m is the height of the maximum of $\chi(\beta)$, i.e. $\chi(\beta_C)$. Therefore we directly see that

$$\gamma = \nu \frac{\ln x_m}{\ln L} = -\frac{\ln L}{\ln \Delta x}\frac{\ln x_m}{\ln L} = -\frac{\ln x_m}{\ln \Delta x}. \tag{21}$$

So γ is independent of L and therefore universal. By taking the FWHM Δx of c_V (difference of two points which are located at half the maximum) and taking the derivative of M by just taking the two neighbor points with the biggest difference, according to

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x} \approx \frac{f(x_{i+1}) - f(x_i)}{\Delta x}, \tag{22}$$

we obtain the value for the maximum of χ, x_m. For our sample data we obtained

$$\gamma = (1.95 \pm 0.15). \tag{23}$$

The literature value here is 1.8 - which is located in our tolerance. We can also plot the logarithmic values of Δx versus x_m in order to retrieve γ as the negative slope of the fitted linear curve following $y = \ln(x_m) = -\gamma \cdot \ln(\Delta x) \equiv A \cdot x$. The result is displayed in fig. 14. The slope in the shown graph

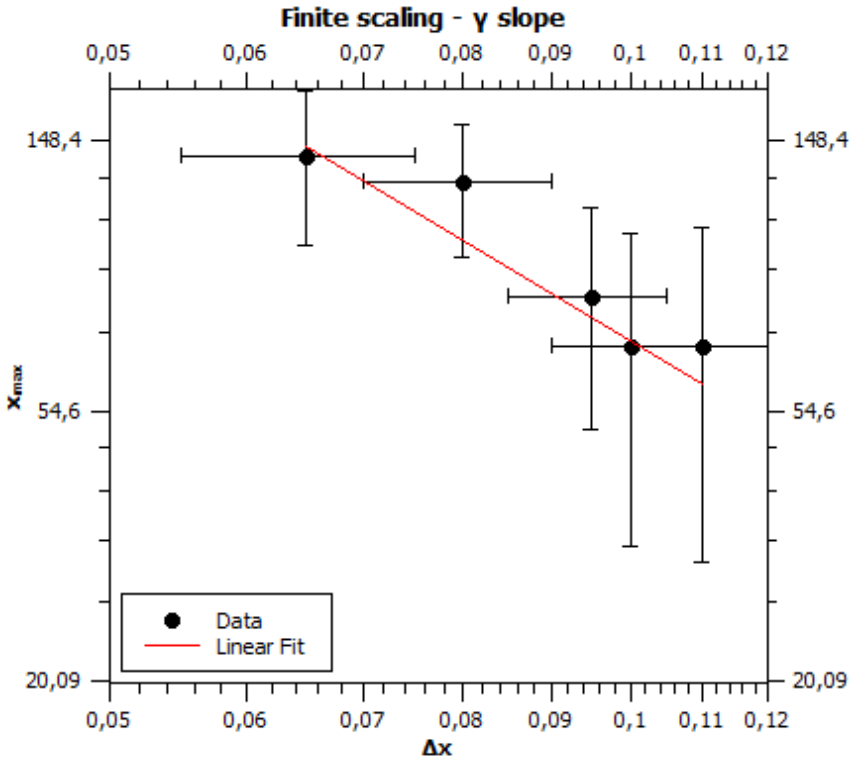

FIG. 14: γ as the slope of a log-log plot of $\ln(x_m) - \ln(\Delta x)$ with values from different 16×16 evaluations

was $\gamma = (1.65 \pm 25)$. Other important critical exponents can also be obtained like

$$\alpha = 2 - \nu d, \tag{24}$$

$$\eta = 2 - \gamma/\nu, \tag{25}$$

$$\beta = \frac{\nu}{2}(\eta + d - 2), \tag{26}$$

where d is the dimension (here we used a 2 dimensionsal lattice). The values for γ and ν have already been obtained by the steps above. In fig. 15 the resulting curve is fitted on the left and on the right side of the peak with the equations

$$lhs(x) = A \cdot \left(\frac{\beta_c - x}{\beta_c}\right)^{-\nu}, \tag{27}$$

$$rhs(x) = B \cdot \left(\frac{x - \beta_c}{\beta_c}\right)^{-\gamma}. \tag{28}$$

The amplitudes A and B have to be found out by fitting the curve to the datapoints. Overall for 10 example runs we were able to get a good result of $A = 0.01$ for $\beta_C = 0.435$ and $\gamma = 1.7$ for the right side and $B = 0.23$ and $\nu \approx 1$ for the left side.

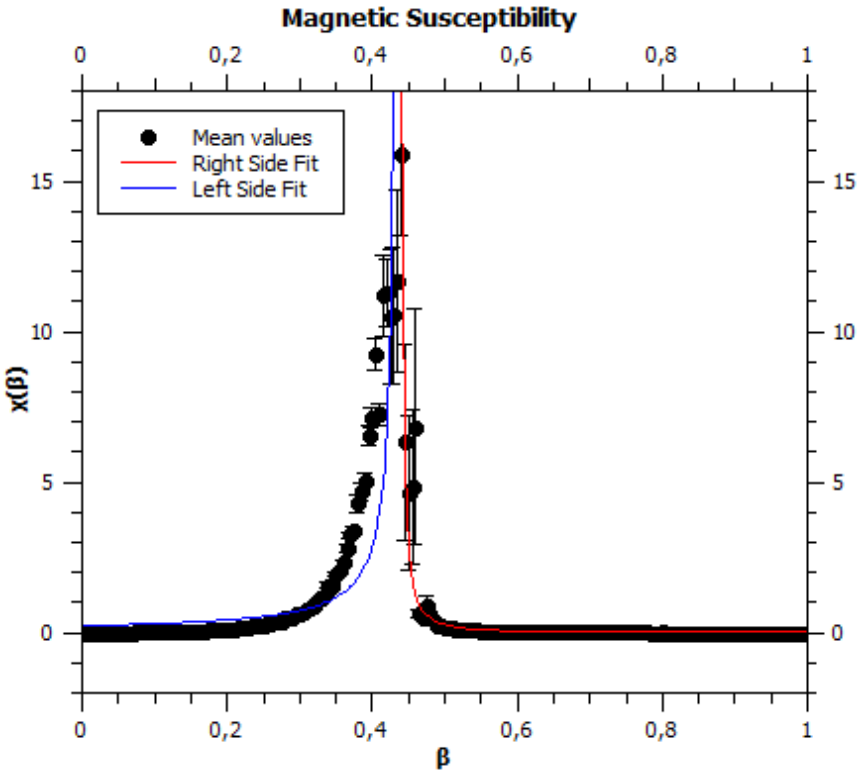

FIG. 15: The Magnetic Susceptibility $\chi(\beta)$ for a 16×16 lattice with errorbars from 10 different runs

IV. CONCLUSION

The programmed Ising 2D ising model was able to reproduce all formerly known diagrams and to show some interesting aspects of a critical point. I was able to understand the Metropolis *et al.* method and could get known to the most basic principles of simulating a physical system using a Monte-Carlo algorithm. The precision was quite good as a matter of being effective, since the program running durations were mostly lower than one minute. The precision can easily be optimized by following some of the steps mentioned before, i.e.

- more iterations,
- more repitions and
- increasing β slower - mostly around β_C.

Acknowledgments

I want to thank Tom DeGrand for giving me some insight in current research and providing me with literature and papers for further reading.

---

[1] T. DeGrand and C. DeTar, "Lattice Methods for Quantum Chromodynamics," World Scientific Publishing, Singapore, ISBN **981-256-727-5**, (2006).

[2] D. Friedan, Z. Qiu and Stephan Shenker, "Conformal Invariance, Unitarity, and Critical Exponents in Two Dimensions," Phys. Rev. Lett. **52(18)**, 1575 (1984).

[3] V. Privman and M. E. Fisher, "Universal critical amplitudes in finite-size scaling," Phys. Rev. B **30(1)**, 322 (1984).

[4] M. N. Barber, R. B. Pearson, D. Toussaint and J. L. Richardson, "Finite-size scaling in the three-dimensional Ising model," Phys. Rev. B **32(3)**, 1720 (1985).

[5] M. E. Fisher and M. N. Barber, "Scaling Theory for Finite-Size Effects in the Critical Region," Phys. Rev. B **28(23)**, 1516 (1972).

[6] J. Cardy, "Scaling and Renormalization in Statistical Physics," Cambridge University Press, Cambridge, **ISBN 052-149-959-3**, (1996).

[7] L. Witthauer and M. Dieterle, "The Phase Transition of the 2D-Ising Model," http://quantumtheory.physik.unibas.ch/bruder/Semesterprojekte2007/p1/index.html (2007).

[8] L. Onsager, "Crystal Statistics. I. A Two-Dimensional Model with a Order-Disorder Transition," Phys. Rev. **65**, 117 (1944).

[9] Output generated by the Simulation program.

[10] 10000 iterations taken at 0, 1475, 4920 and 10000 iterations

[11] 10000 iterations taken at 0, 3045, 6535 and 10000 iterations

[12] 10000 iterations taken at 0, 3095, 6490 and 10000 iterations

Example of output data

The program is able to write some output data in a textfile in a csv format, so that the data can be important. A sample file looks like the following.

```
beta;<S>/N;Error <S>/N;<U>/N;Error <U>/N;C/N;Error C/N
0;0,00046875;0,0636128435802961;0,0109375;0,172276276290367;0;0
0,005;0,0028125;0,0665744953505986;−0,006875;0,185892732387647;0,0002189475;0,000217887292741151
0,01;0,002109375;0,0561597792000184;0,021875;0,170672734932956;0,00073825;0,000734565799001012
0,015;0,006171875;0,0667850714473614;0,0703125;0,170549873385672;0,001658671875;0,00164792462758871
0,02;0,00609375;0,0742536716752788;0,0396875;0,171212566413004;0,00297171;0,00295589007632834
0,025;0,00203125;0,068402437284442;0,113125;0,171438552837423;0,0046555625;0,00461310302623763
0,03;0,01453125;0,0650238140501135;0,1315625;0,19795628590689;0,0089383275;0,00885691428297634
0,035;0,007109375;0,0592916143437396;0,1209375;0,144518900475671;0,006484261875;0,00640837252898022
...
```

In the first column we have the β value, in the next column the spin expectation value then the error in the previous value. The next four columns are for the energy and its error as well as the specific heat and its error.