2D Ising Model project documentation

Axel Katona / TAUHO5
BME

16 January 2023

Abstract

This paper is the documentation of my semester project, which is the Monte Carlo Simulation of the 2D Ising model. First, I present the basic concept of the used techniques, then the implementation of the code. Roughly the same documentation is written in the Jupyter Notebook of the project. I personally recommend to follow that as a documentation of the project (I have written in this paper the same as I did in the ipynb file), since the structure is better (i.e. talking about the code, and in the next cell seeing the actualimplementation).

1 Introduction

The Hamiltonian for the Ising-model is:

$$H(\vec{\sigma}) = -J \sum_{<ij>} \sigma_i \sigma_j - h \sum_i \sigma_i$$

J is the interaction strength and h is the external magnetic field. Furthermore,

$$P(\vec{\sigma}) = \frac{e^{-\beta H(\vec{\sigma})}}{Z}$$

where the partition function, Z is:

$$Z = \sum_{\{\vec{\sigma}\}} e^{-\beta H(\vec{\sigma})}.$$

The transition probability is denoted by $W(i \to j)$. For the used Metropolis-Hastings algorithm the following form was considered:

$$W(\sigma \to \sigma') = g(i \to j) \cdot A(i \to j)$$

In the case of the Monte Carlo method, the case the selection probability is uniform:

$$g(i \to j) = 1/N^2$$

. Te second term respects the Metropolis-Hasting rule:

$$A(i \to j) = \begin{cases} e^{-\beta(E_j - E_i)} & if\ E_j - E_i > 0 \\ 1 & otherwise \end{cases}$$

So, imposing the detailed balanced condition:

$$\frac{W(i \to j)}{W(j \to j)} = \frac{A(i \to j)}{A(j \to i)} = e^{-\beta(E_j - E_i)}$$

In this algorithm, we gathered the spins in a grid, lets denote it with G. Every gridpoint with coordinate (i,j) is a spin pointing up ($\sigma_+ = 1$) or down ($\sigma_- = -1$). We also used the interaction for the nearest-neighbours, and the sum of them is denoted by Sn. For a single flip, the energy change is the following:

$$\Delta E = 2G(i,j) \cdot (h + J \cdot S_{i,j})$$

.

2 Implementation

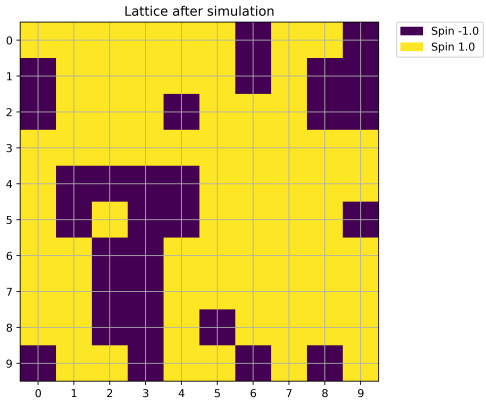
For the detailed implementation, see the ipynb. I commented everything with detail. The structure of the project is the following: first, the **initgrid** function initializes the grid, with $L \times L = 10 \times 10$ grid size. The initialization is either completely random, or all spins are up spins. I used a function, **plotspins** to plot the spinmatrix. Then, I defined the Ising model itself in **isingmodel** function. The parameters are the maximum number of steps (n), the spin matrix (M), the parameters for the energy (interaction term -K-, external field -h-, and the β- kbT-). It computes the energy, and does the actual simulation. The calculation of the magnetisation (sum of all spins) and the energy difference is defined in speparate functions.

These are the basic blocks of the project.

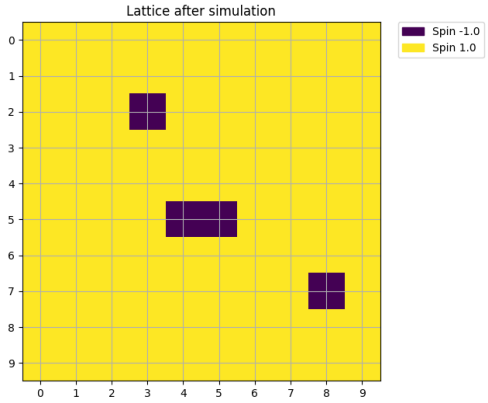
3 Results

3.1 Example run 1

The spins were initialized on a 10×10 grid. This is a demonstration of the simulation.

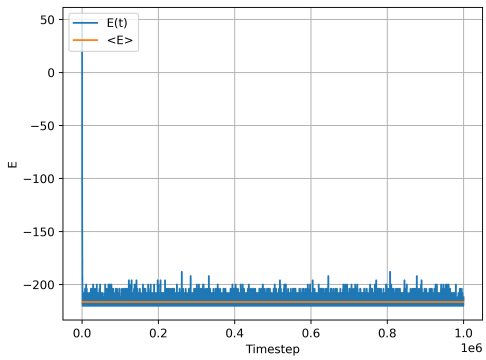


(a) $k_bT = 5$

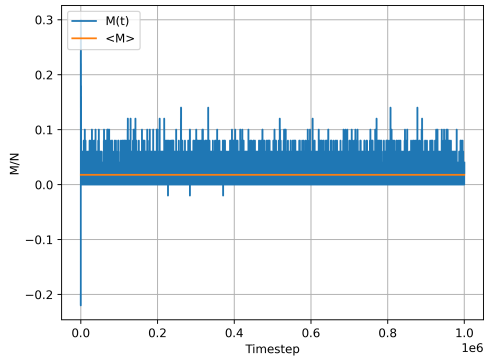


(b) $k_BT = 3$

Figure 1: Plotting the lattice after n=1,000 steps of simulations, with h=1 and K=1. High k_BT means larger excitation.

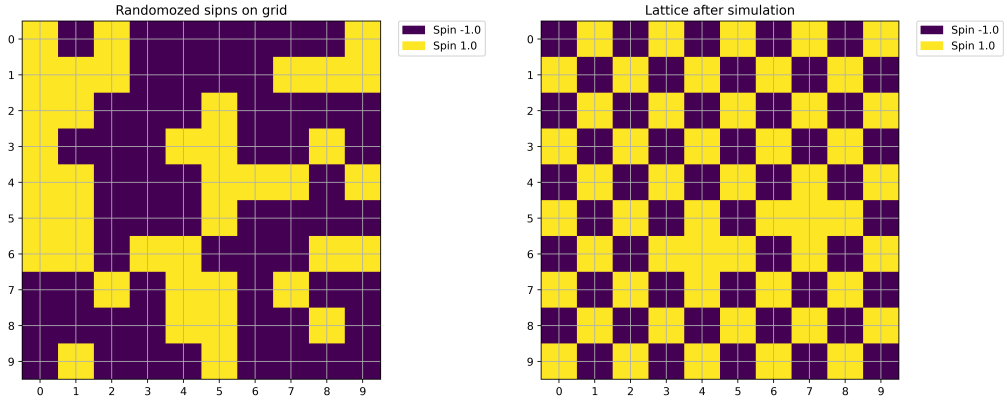


(a) *Energyrelaxation*



(b) *Magnetisation*

Figure 2: The energy and normalised magnetization. The initial state was random, the final state is organized, kBT=1 in this case, but I changed the interaction term to -1.



(a) Initial state (b) Final state

Figure 3: Changing the interaction term to -1 yields an interesting structure (right)

3.2 Relaxation time

To measure relaxation time of the (average) magnetization, I used $h = 1$, $K = 1$, $k_bT = 5$. Then I fitted an exponential to the first few hundred move (to see the actual relaxation time, not the 'noise').

From the fitting, $\tau \approx 900$ timesteps was yielded.

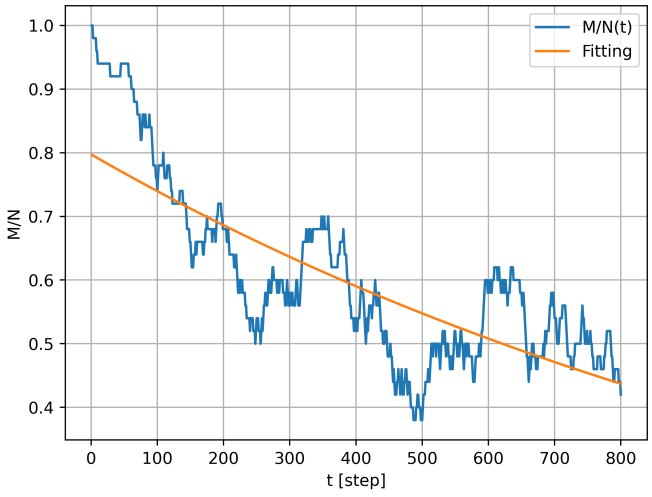


Figure 4: Fitting exponential

3.3 Varying h

I modified to code to run for long timesteps (50k), and initilaized an 'all spins up' state. was also curious how the $M(h)$ depends on k_bT, so for every h, I checked for various k_bT values. I generated h values at random, with $h \in [-10,10]$ condition.

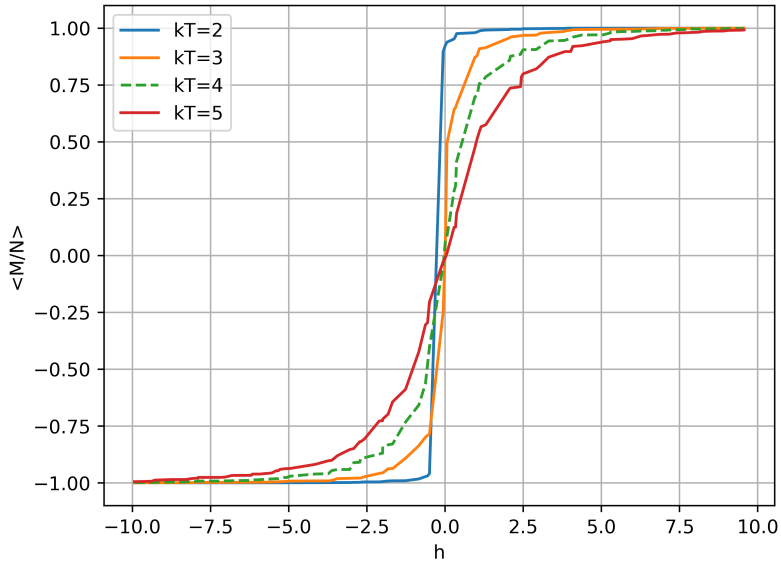


Figure 5: M(h) (normalised) for different k_bT-s with varying fields.

3.3.1 Discussion

Large h means large external field, which makes the spins orient in the same direction (up), i.e. $\lim(M)_{h->\infty} = N$. With opposite h, the opposite orientation occurs - every spin down- (M is an odd function), and $\lim(M)_{h->-\infty} = -N$. At h=0, the interaction term dominates, resulting in $M = 0$ (as many up spins as down). Higher k_bT means, that the magnetisation reach the limits slower, the phase transition is slower.

3.4 Susceptibility

In this task, I had to check if

$$\chi = \left.\frac{\partial M}{\partial h}\right|_{h=0} = \left(\left\langle M^2 \right\rangle - \langle M \rangle^2\right) \cdot \beta$$

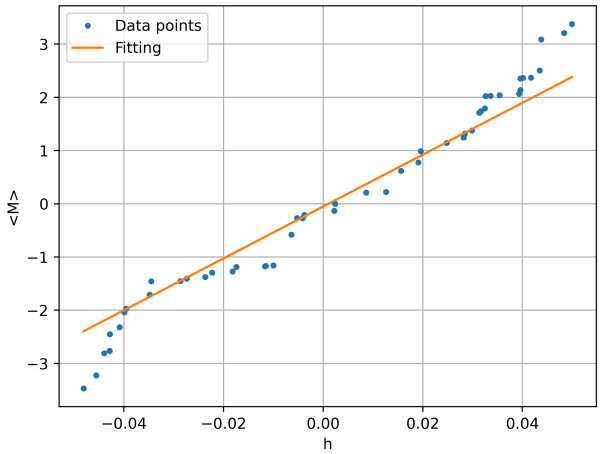


Figure 6: Fitting a line for small fields gave me the derivative at h=0. I have tried to get the derivative with other methods (e.g. central difference method), but this gave the best and most consistent results.

holds. Fitting a first-order polynomial for small h values yields the derivative at h=0 (the slope; $p1 = \chi = (58 \pm 1)$), and after that, from $Var(M)/k_bT$ I got the fluctuation, which was $p2/kbT = \chi' = (57 \pm 1)$.