Cluster Monte Carlo Simulation on 2D to 5D Ising Model

Chen Wu

December 5, 2019

Abstract

In this study, the Cluster Monte Carlo Simulation is used to simulate the thermal equilibrium states of the lattice Ising model of 2D, 3D, 4D, and 5D. Compare to the traditional Metropolis Monte Carlo Simulation, the Cluster Monte Carlo Simulation or Wolff Algorithm will significantly speed up the rate of convergence to the thermal equilibrium. This phenomenon is exceptionally obvious around the phase transition temperature. After the simulation of each dimension, data were generated and analyzed to calculate the critical exponents. These critical exponents are used to compare with the ones calculated by Mean Field Theory or other analytical (exact) methods given in the literature. Because of the limitation of the computational power, relatively smaller size is used for each dimension. There are some discrepancies between these simulated critical exponents from the ones given by the Mean Field Theory and other analytical methods because of the small grid size. However, it still shows great insight into the physics encoded inside the Ising Model.

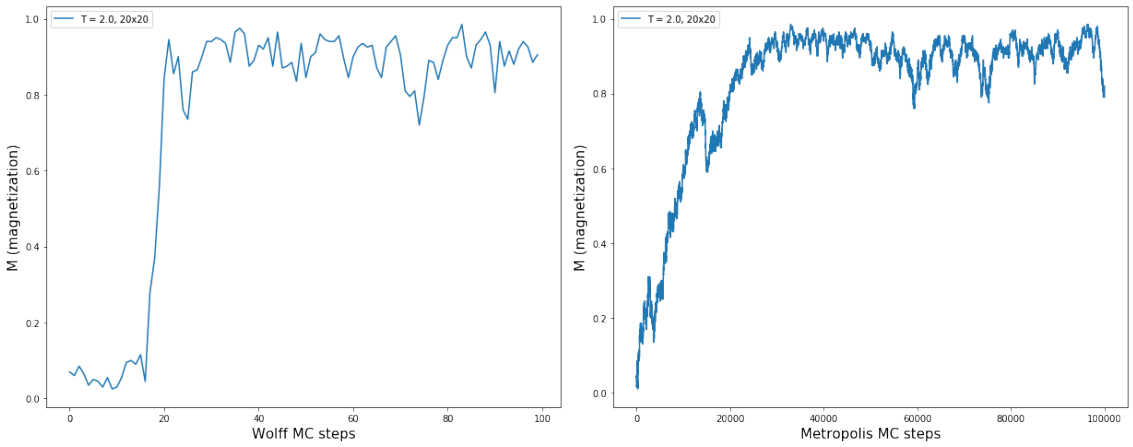
1 Introduction

Ising Model is an exemplary model in statistical mechanics. The simplicity of two-state spin flipping can encompass a great amount of physics information of different physical systems such as magnetization, energy ensemble, free energy, heat capacity, and susceptibility. However, looking for the analytical solution of the partition function can be extremely hard given the dimension is high. Therefore, two methods become very useful to approximate the partition function and then the physical quantities for the Ising model in a higher dimension. The first one is the Mean Field Theory in which the fluctuation of each spin from the average magnetization is assumed to be negligible. This assumption makes more sense when each spin has a larger number of neighbors or, equivalently, when the dimension is high. (Actually for upper critical dimension, Mean Field theory may not be guaranteed to capture all transition features). The second one is the Monte Carlo Simulation in which the thermal equilibrium is achieved by brutally flip spin or spins after

checking whether such action obeys certain criteria. Those two methods should coincide well in a high dimension. For a lower dimension, Monte Carlo methods should be comparable to some other exact methods for the Ising model such as transfer matrix methods[1].

Traditional Metropolis Monte Carlo Simulation will flip one spin in each trial and consider to do the flip or not by evaluating the transition energy of the whole system based on this single flip. The probability of such a single flip equals to the Boltzman Distribution of states' energy difference. For example, if the flip will cause the state of the whole system changes from E_a to E_b, where $E_a < E_b$, such flip will occur with the probability $p = \exp[-\beta(E_b - E_a)]$. If $E_a > E_b$, this flip will sure to happen. This method is also known as the local Monte Carlo Simulation. To retrieve the phase transition, we have to scan a range of temperature from an extremely low temperature to a relatively high temperature. That is, at each temperature, the Monte Carlo Simulation will be used. At a low dimension, the Metropolis Monte Carlo simulation works well except the temperature around the critical temperature T_c. The reason is that the total magnetization of the whole system becomes widely distributed. For a 2D lattice with $N \times N$ grids, the total magnetization can be ranged from $-N^2$ to N^2. Each trial of the Monte Carlo sampling can only contribute to 2 unit of magnetization change at most. It takes a large number of the trials to bring the system to convergence. This phenomena is called critical slowing down phenomena. Since the simulation will be more accurate as the grid size gets larger, it's not worthy to sacrifice the grid size. Then the only way to realize a healthy sampling rate around the critical temperature is to flip more than one spins or to flip a cluster of spins so that a larger amount of magnetization change can be brought up in each trial. This is the motivation of the cluster Monte Carlo simulation (cMC) or Wolff's Algorithm [7] which modifies the old version of the cMC called Swendsen-Wang Algorithm [5]. It is straightforward to compare Metropolis Monte Carlo method and Wolff's Algorithm in Fig.[1]. Notice that to produce a similar convergence pattern, Wolff's algorithm uses 100 steps but the Metropolis algorithm uses 100000 steps. This is at temperature $T = 2$ which is close to a 2D critical temperature, so the maximum height of magnetization is slightly less than 1. However, to boost the magnetization to the maximum, Wolff MC takes around 30 steps. But to do so, Metropolis MC takes around 30000 steps. Although through real programming Wolff's algorithm actually takes longer to prepare for one flip as we will see later, saving about 1000 times of steps is not only computational economic but also shrinking the overall computing time significantly. For a well optimized Wolff's algorithm, the backfire of a longer preparation time for flipping in each Monte Carlo trial will not be so obvious. This overall speeding up of Wolff's algorithm will be more significant for larger lattice sizes.

In this study, this kind of powerful cluster Monte Carlo (cMC) simulation will be applied to the



(a) 2D (b) 3D

Figure 1: Comparison of Wolff's and Metropolis' Monte Carlo Simulation of a 20×20 Ising Model at the temperature of T = 2. (a) Wolff's results. (b) Metropolis results.

Ising Model simulation. 2D to 5D lattice will be used to analyze the results generated by this Wolffs' algorithm. After the simulation, several physical quantities will be calculated, including average magnetization, susceptibility, heat capacity, and correlation function. At each temperature, they will be calculated through about more than 900 cMC steps during which the system is already convergent. By calculating these variables, the critical temperature and several critical exponents will be calculated:

$$M \propto |T_c - T|^{\beta} \tag{1}$$

$$\chi \propto |T_c - T|^{-\gamma} \tag{2}$$

$$C_v \propto |T_c - T|^{-\alpha} \tag{3}$$

$$\xi \propto |T_c - T|^{-\nu} \tag{4}$$

These exponents and the T_c will be calculated through fitting and compare to the published value. Since the higher the dimension, the more neighbors a spin will have. Thus it is expected that higher the dimension will give better results. However, because the whole simulation will be carried out on a 4 cores personal computer, a large 5D Ising model will be very hard to simulate. Therefore, in this simulation, although the correct pattern is shown, the results are not necessarily

better in a higher dimension. It could be further improved by parallel computing and run it on some computing hubs. But for this project, this flaw is inevitable. And we will try best to do the fit and polish the data.

2 Implementation of the cluster Monte Carlo methods

In this implementation, the neighbors of each spin are defined at the beginning. No matter how many dimensions we have, spins are indexed from 0 to N-1. In other words, it is a single chain. In the situation of different dimensions, this indexed array of spins is twisted in a serpentine shape so that it can build the grid with the required dimension. For each distinct dimension, the neighbors of each spin are uniquely defined with the assumption of periodic boundary condition. Suppose, in 2D, i^{th} spin has the neighbors a, b, c, d. Then j^{th} spin also has the neighbors a, b, c, d if and only if $i = j$. So the whole list of spin is actually a dictionary with indexed spins as keys and the set of their neighbors as the value. For d dimension, each indexed spin will have 2d neighbors been defined.

During the whole procedure, it will be assumed that there is no external field. So the Ising model used here is the most simplified one:

$$H = -J \sum_{<i,j>} S_i S_j \tag{5}$$

Therefore all the thermodynamic variables will be retrieved from this pure spin movement without information of the external field.

2.1 Wolff's algorithm Setup

After the neighbors are clearly defined, the cluster Monte Carlo algorithm will be implemented. This algorithm will be learned and cited from the book [2] and corresponding online courses (https://www.coursera.org/learn/statistical-mechanics/home/week/8). Initially, all N spins are initiated randomly. 1 and -1 are chaotically scattered in the whole spin list. Two lists are being created: "cluster" and "pocket"[2]. The cluster represents the final list containing all spins that are going to be flipped at the current trial. The pocket list is a "buffer" list that contains the neighbors, which have not been in the cluster yet, of the spins and have the same sign of the spin inside the cluster. Suppose initially we locate a spin and put it into both of the cluster and the pocket. Then we choose the neighbors with the same sign of spin as that cluster spin and put it into the pocket with a certain probability p_0 and later push the pocket spins into the cluster. That

is, p_0 is the criterion selection of the pocket. Any spin passed such criterion will guarantee to be inside the cluster list. When all of the spins inside the pocket have been pushed into the cluster, the pocket will be empty. Then all spins inside the cluster will be flipped. This is the time we say that the current trial is finished and the system should be ended up in some different state. Throughout this procedure, the pocket list is a dynamically changed list that helps the cluster list to grow.

Before the simulation, it is crucial to define the quantity p_0. To illustrate clearly, we define the state before the flip to be state a and the state after the flip to be state b. Physically, the goal is to bring the state to the thermal equilibrium state. Mathematically, the probability of the flows from state a to state b are equal. That is, the rate of change of the probability is 0. It is called detailed-balance condition and is defined as[2]:

$$\pi(a)A(a \to b)p(a \to b) = \pi(b)A(b \to a)p(b \to a) \tag{6}$$

Here $A(a \to b)$, called stopping probability, is the probability that the growth of the cluster is stopped right at state b. $p(a \to b)$, called acceptance probability, is the probability of the action accepted from state a to state b. The acceptance probability includes the action accepted by entering state b but not limited to state b. Stopping probability exerts such a limit so that this composite probability (times the $\pi(a)$, which is the Boltzman weight of the state a describes exactly the probability of the transition starting from state a and ending at state b. The backward transition can also be understood in a similar way. And the transition between these two states is a dynamical equilibrium. Notice that the acceptance probability here is not the probability, p_0, of taking one of the neighbors into the pocket (or cluster). p is the overall acceptance probability from state a to state b after the cluster has been flipped. In the algorithm, it is the probability that the state b accepts this movement after one round of Monte Carlo trial. Within each trial, $p0$ is a criterion that needs to be passed (or rejected) for a number of times. By evaluating **??**, we will get[2]:

$$e^{-\beta(n_1-n_2)}(1-p_0)^{n_2}p(a \to b) = e^{-\beta(n_2-n_1)}(1-p_0)^{n_1}p(b \to a), \tag{7}$$

from which we can get:

$$p(a \to b) = \min[1, \frac{e^{-\beta(n_2-n_1)}(1-p_0)^{n_1}}{e^{-\beta(n_1-n_2)}(1-p_0)^{n_2}}] \tag{8}$$

This is analogous to how the acceptance probability is retrieved from the Metropolis Algorithm. The difference is the definition of n_1 and $n2$. $n1$ is defined to be the number of spins on the inner boundary of the cluster which have different signs from their outside neighbors. $n2$ is the number

of spins in the cluster which are the same as their outside neighbors. So $n2$ is the number of outside spins that can be, but may not be, taken into the cluster. This is why A is defined to be $(1-p_0)^{n_2}$. It means all the same signed spins are rejected to be taken into the cluster, which means the cluster stops growing. For Wolff's Algorithm, we simply want to build the cluster and flip all spins inside the cluster, and repeat this process again and again, which means $p(a \to b) = 1$. Then, by some algebra, $p_0 = 1 - e^{-2\beta}$. This is the detailed probability that determines whether we add the spin into the cluster or not. With this probability, our algorithm can be constructed. In the real code, β is assumed to be just $1/T$. T is set to be different for each dimension. We "cheat" a little bit and check the published critical temperature value first. Then a denser sweep of temperature is defined around the "known" critical temperature. Lower and higher regions of the temperature will be loosely arranged because they are likely to be 1 and 0 respectively so we don't need to pay special attention there.

After several trials of different lattice sizes, it is necessary to decrease the lattice size as the dimension gets higher. In this simulation, 100×100, $20\times20\times20$, $10\times10\times10\times10$, and $6\times6\times6\times6\times6$ are used for 2D, 3D, 4D, and 5D respectively. If higher dimension lattices show the same, if not better, qualities of exponents estimation, this simulation can automatically reveal that higher dimension lattice can utilize the advantage of Mean Field Theory because of the larger amount of neighbors.

2.2 Thermodynamic variables calculation

Throughout the simulation, necessary thermodynamic variables are calculated along the way for 1000 trials of cluster Monte Carlo simulation at each temperature.

Mean magnetization, M, is calculated firstly by summing up all of the spins and divide the sum by the spin number. And then this number begins to be summed up at the end of each trial after, for example, the 5th or 6th trial and divided by the number of the trials which have been taken into such data acquisition process. Wait for several trials is to wait for reaching the thermal equilibrium. This idea will also be used when other variables are calculated. For different dimensions, this waiting time will be different. It's likely that the lower dimension one will need a longer relaxation period because it's larger sites over a smaller number of neighbors. With the same p_0, the pocket list cannot touch too many neighbors within each trial for the low dimensional lattice.

The susceptibility, χ, should be calculated from the derivative of magnetization with respect to the external field and then switch field to 0. After some derivation from the partition function,

χ can be calculated from the equation:

$$\chi = \beta(\langle m^2 \rangle - \langle m \rangle^2), \tag{9}$$

where m is the magnetization per site.

To calculate the heat capacity, C_v, energy, E, of the system per site needs to be calculated first. During the simulation, the whole configuration of the N spins is known and their neighbors are also known. Therefore, after each Monte Carlo trial, the energy per site is calculated by subtracting the product of each spin with each of its neighbors from the previous energy level and divide it by N. Finally, this sum (actually sum by subtracting) is divided by the overall number of trials in the equilibrium states. Just like calculating the susceptibility, C_v can be calculated from:

$$C_v = \beta^2(\langle E^2 \rangle - \langle E \rangle^2), \tag{10}$$

where m is the magnetization per site.

All of the above three variables will be used to fit the function:

$$F(T, T_c, x, A, b) = A|T_c - T|^x - b. \tag{11}$$

The calculation of the correlation length is a bit sloppy here because of the limitation of lattice size and the pursuit of simplicity. The correlation, $\langle s_0 s_r \rangle$, is calculated by only considering a single axis for all dimensions. On this single axis, only around half to three-quarters of the length will be taken into account because the periodic boundary condition will increase the correlation near the other end of the boundary. This simplification will significantly affect the correlation estimation when the dimension is high. For a higher dimension, smaller lattice sizes will be used to avoid the crash of the program. Here we use $6 \times 6 \times 6 \times 6 \times 6$ for 5 dimensions. But at most we will only have about 4 or 5 data points that can be used to fit the correlation function. No matter what the critical exponents are, the fit might not make too much sense. The correlation function is defined as:

$$G(r) = \langle s_0 s_r \rangle. \tag{12}$$

The constants $\langle_r\rangle$ and $\langle_0\rangle$ are omitted since the constant will not affect significantly. This set of values will be fitted by using

$$G(r) = r^{-d+2+\eta} \tag{13}$$

where d is the dimension of the Ising model and η is a critical exponent. This function is only

valid at the transition point[6]. Therefore, a relatively accurate transition temperature needs to be found before this step. To calculate ν, the equation

$$\nu = \frac{\gamma}{2 - \eta} \tag{14}$$

will be used. Recall that γ is the critical exponent defining for the $\chi(T)$ function.

Of course, there are other critical exponents. But they can be calculated by manipulating these critical exponents.

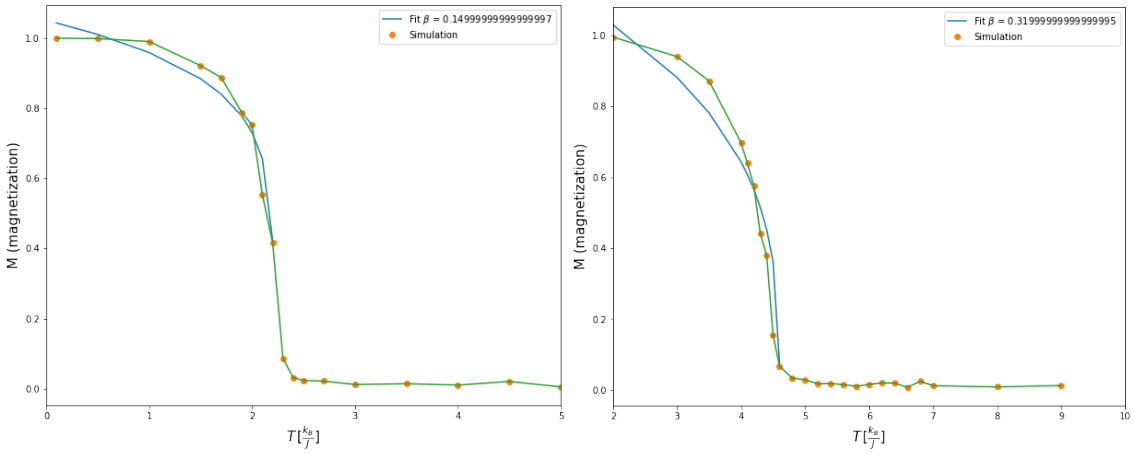
3 Result

By using the simulation and the function of variable retrieval, we will show the result of the Ising model from 2D to 5D. For each dimension, the critical temperature, T_c, is the only parameter that will appear in all three fittings: M, χ, and C_v. Therefore, it is reasonable to average the estimated critical temperature for all these three cases for each dimension. However, for an easy figure-presentation on this paper, we will put the same physical variable of 4 different dimensions together. During the fit, python sometimes does not give a reasonable value. Two ways were used to achieve reasonable values. Firstly, an initial guess is used by either exponentiating the result from the logarithmic fit or by eye-balling the correct pattern. Secondly, in the case of python giving a set of fitting parameters which is way off the correct range, bound values of each parameter are set. The goal is to put as less human-intervention as possible during the fitting, so most of the bounds are either kept as default or set at one end but leave the other end to be infinity or zero.

3.1 Magnetization

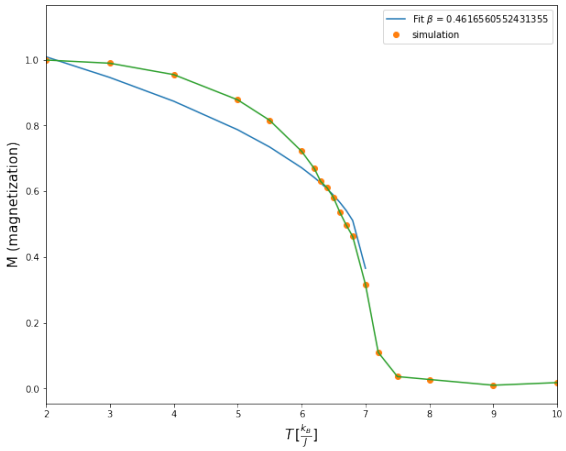
For the data analysis, the Python Scipy package is used. Although the curve-fitting method in this package works very well for the linear fit, it does not always give good results of the nonlinear fitting. Through several different trials and try, the curve fitting works better on variables other than magnetization. Therefore, I took the logarithm of magnetization to linearize the exponential equation. And then I exponentiated the fitted parameters to set them as the initial guess points for the linear fitting.

As shown in Fig.[2], the fit is not as good as expected. It seems no matter for which dimension and no matter use which parameters, the magnetization stay at 1 for too long. However, the good aspect is that it shows the phase transition. Usually, when the lattice size is two small, it is likely that the phase transition does not even occur and the magnetization will just smoothly go

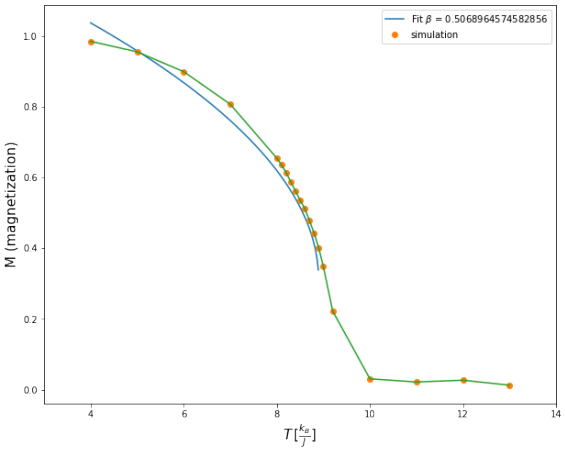


(a) 2D

(b) 3D



(c) 4D



(d) 5D

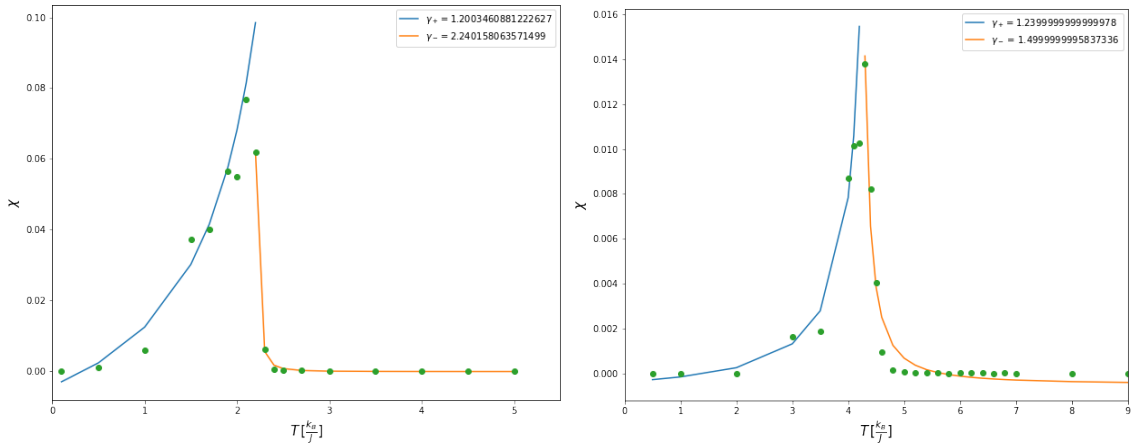
Figure 2: The magnetization fit of (a) 2D (b) 3D (c) 4D (d) 5D. The critical temperature for each dimension in this round of M-fit is $2D$: 0.2990850; $3D$: 4.59925076; $4D$: 7.00000003103, $5D$: 8.896484828878513

down to zero. Although an absolute non-analytical transition can hardly be made, we can see that the curve has some abrupt change around the transition temperature T_c. If time were allowed, a denser scan around the critical temperature seems necessary. We can see that the β is still in a relatively reasonable range. However, with such a sloppy fit, this critical exponent might not provide an excellent source to compare. (Notice that the temperature fitted for this case is not the final critical temperature used to compare to the published value.)

3.2 Susceptibility

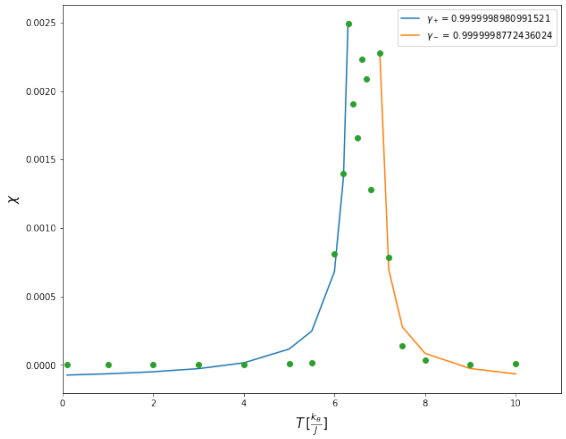
The susceptibility is fitted separately. Because the value on the critical temperature is assumed to be very large if not completely infinity. Therefore, it was not assumed in the beginning that γ_+ and γ_- are equal. The fit will be even worse if one single γ is blindly used. The initial guess of the

critical temperature tried for several times and decided by whether the fitting makes sense. And then the overall γ is determined by taking the average of γ_+ and γ_-. From Fig.[3], the simulation

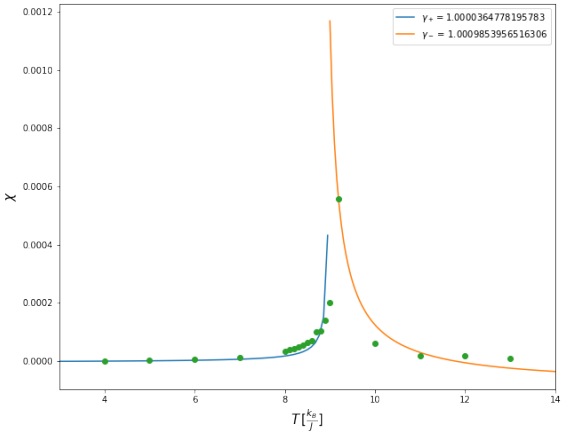


(a) 2D

(b) 3D



(c) 4D



(d) 5D

Figure 3: The susceptibility fit of (a) 2D (b) 3D (c) 4D (d) 5D. The critical temperature for each dimension in this round of χ-fit is $2D$: 2.265785994598976; $3D$: 4.3226819498406766; $4D$: 6.663988364958619, $5D$: 8.899999864337355

of 4D and 5D gives a more unpredictable scattering of points. However, during the fitting, γ_+ and γ_- for 4D and 5D do not change drastically by using different initial guesses. And it is clear that γ_+ and γ_- gets closer at a higher dimension than at a lower dimension. Considering the lattice size is super small for either 4D or 5D, the fitting of this critical exponents is actually pretty good.

3.3 Heat Capacity

As mentioned in the last section, heat capacity is calculated based on the average energy per site. Generally, C_v cannot be infinite anywhere. It has a non-analytical but continuous behavior around the critical temperature. Therefore, one single α is strictly assumed.

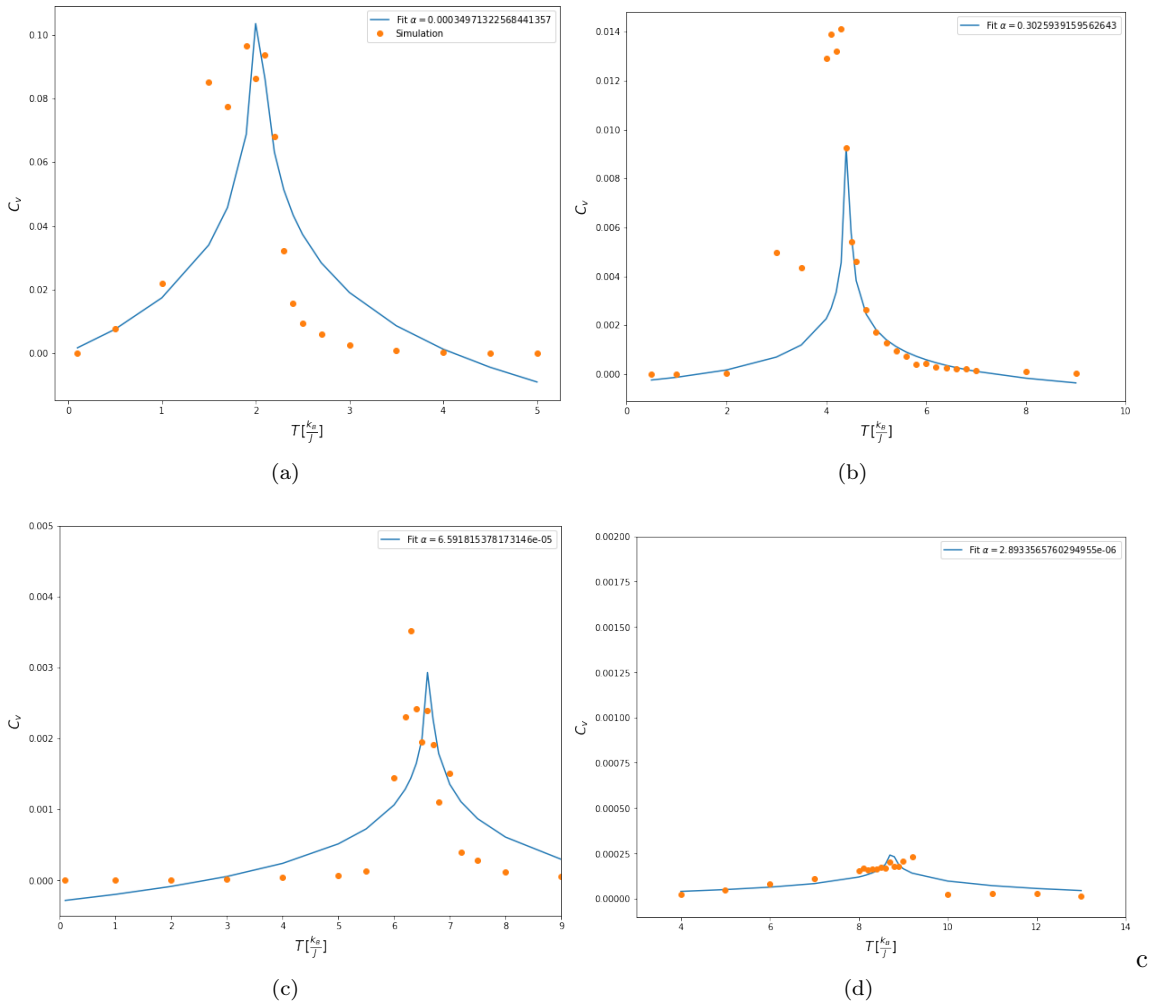


Figure 4: The heat capacity fit of (a) 2D (b) 3D (c) 4D (d) 5D. The critical temperature for each dimension in this round of C_v-fit is $2D$: 22.112762356573582; $3D$: 4.424527305101592; $4D$: 6.624041505430623, $5D$: 8.744330196047304

In Fig.[4], almost all of the fittings show reasonable results. 2D Ising shows an α which is closed to 0 because it has a quite large lattice size. 4D and 5D Ising models show α extremely close to 0 because of the larger number of neighbors. For the 5D case, at first glance, the fit is not right on the data points. It is because the real value of the data is already extremely small. The discrepancy, which is not large, could be deemed to be large. No matter how to define the initial parameter, α will go to a very small value. Because all the values for 4D and 5D are close to 0, the number of outliers of the data is smaller than that of the low dimensional data. For 3D, despite the outliers around $T = 3$, 4 highest values are actually not included inside the fitted function.

Table 1: Compare MC value and published value

Dim	T_c	β	γ	α	ν
2D	2.226, 2.269	0.150, 0.125	1.720, 1.750	0.000, 0.000	0.982, 1.000
3D	4.449, 4.512	0.320, 0.326	1.370, 1.237	0.303, 0.110	0.632, 0.630
4D	6.762, 6.680	0.462, 0.500	1.000, 1.000,	0.000, 0.000	0.359, 0.500
5D	8.847, 8.878	0.507, 0.500	1.000, 1.000	0.000, 0.000	0.472, 0.500

3.4 Summary of the critical exponents and critical temperature

By taking the average of the critical temperature, we can get a more reliable average critical temperature T_c for each dimension. Using this temperature, we can fit the correlation function, equation [13], around the critical temperature and get the critical exponents η and then use γ to get the critical exponent ν by using equation [14]. The overall result is presented in the table 1 and kept up to 3 digits. The every pair of the presented data is in the form "simulated value, published value".

It shows that the simulation is in some way make sense. The value is either cited from Wikipedia "Ising Critical Exponents" or different papers including [3],[4]. Overall this simulation is not bad. At a low dimension, the larger lattice size helps to maintain reasonable exponents. At a higher dimension, the larger number of neighbors helps to maintain reasonable exponents. From the observation, 3D does not have a good prediction. One the reason is that 3D lattice has the smallest lattice size which is $N = 8000$. 4D and 5D are usually predicted by Mean Field Theory and Renormalization Group in the literature. Therefore, the high density of neighbors will coincide with the idea of the Mean Field Theory well. However, I believe the extremely high consistency in the comparison of ν is just a mirage. It is not directly calculated by fitting a function. Therefore, it is hard to claim that these ν are reliable. It turns out that the value of η for each dimension is not as precise as ν except for the 2D case in which $\eta = 0.247$. In fact, $\eta = 1/4$ in 2D. It is not surprising that 2D cMC will give a better estimation of η and ν because 2D lattice in the simulation has the longest range of distance in one single axis among all the testing dimensions. Even using half on the axis, which is 50 data points, will give enough values for relatively accurate fitting. But for others, short distance, scarce of data, and the effect of periodic boundary condition will all give a less valuable prediction.

Notice that the γ from low dimension (2D and 3D) are got by taking an average of two numbers which have a relatively large difference. So they are not as reliable as γ of higher dimensions, though they look good.

4 Conclusion

In this section, several difficulties throughout the whole project will be described and some polish of the simulation in the future will be mentioned.

The greatest difficulty is the limit of the lattice size. It is expected that the higher dimension will give a better estimation of all the critical exponents. But the lattice size washed away this advantage. And it makes it harder to get a reliable correlation function. In addition, a parallel computation to distribute the computing job by different temperatures will largely increase the whole process of this algorithm.

Another unsolved problem is the random fluctuation of the susceptibility and heat capacity near the critical temperature. Currently, I believe it is because of the small lattice size because such fluctuation is not significant in the lower dimension where the lattice size is large. It is worthy to re-scan the temperature will higher resolution to see what it will look like.

In this paper, only four critical exponents have been calculated. However, it will be interesting to use these four exponents to calculate more exponents and to check whether they are still in reasonable ranges.

The code is presented in Github Link.

References

[1] Mehran Kardar. *Statistical physics of fields*. Cambridge University Press, 2007.

[2] Werner Krauth. *Statistical mechanics: algorithms and computations*, volume 13. OUP Oxford, 2006.

[3] Per-Håkan Lundow and Klas Markström. The scaling window of the 5d ising model with free boundary conditions. *Nuclear Physics B*, 911:163–172, 2016.

[4] Per Håkan Lundow and Anders Rosengren. The p, q-binomial distribution applied to the 5d ising model. *Philosophical Magazine*, 93(14):1755–1770, 2013.

[5] Robert H Swendsen and Jian-Sheng Wang. Nonuniversal critical dynamics in monte carlo simulations. *Physical review letters*, 58(2):86, 1987.

[6] Ettore Vicari. Critical phenomena and renormalization-group flow of multi-parameter\phiˆ 4 field theories. *arXiv preprint arXiv:0709.1014*, 2007.

[7] Ulli Wolff. Comparison between cluster monte carlo algorithms in the ising model. *Physics Letters B*, 228(3):379–382, 1989.