Wolff Cluster Algorithm*

Dániel Juhász and Maximilian Mucha
(Dated: March 7, 2021)

The Ising model is a well known model of ferromagnetic materials and phase transitions. Although exact results are known for specific cases, simulating these systems is a highly researched topic. When the system is near a phase transition, successive measurements tend to be highly correlated, therefore extracting statistically significant data becomes difficult. In this final report we demonstrate one of the most efficient algorithms that can deal with this problem: the Wolff cluster algorithm. After testing the correctness of our implementation, we calculate the dynamical exponent, and compare the result with the Metropolis-Hastings method. We also calculate the critical temperature for the 2D and 3D Ising models.

I. INTRODUCTION

The Ising model is a well known model of materials that exhibit phase transitions. Based on this model, a material is a periodic grid of spins. Each spin can point in two directions, we will denote them by $+1$ and -1. There are two constants that describe the material: J, which describes the coupling strength between neighboring spins, and h, which describes external magnetic field. The total Hamiltonian is:

$$H = -h \sum_i s_i - J \sum_{\langle i,j \rangle} s_i s_j$$

where the summation with the angle brackets means summing over all neighboring spin pairs. s_i and s_j are the spin states, so they are ± 1. In our project we will only consider square (2D) and cubic (3D) lattices in zero external field ($h = 0$).

The 2 dimensional infinite Ising model has a critical temperature: above this temperature the magnetization is zero, below that the absolute value of the magnetization is nonzero. Onsager derived the exact value for this case. But in three dimensions we need to rely on computer simulations.

The number of possible states grows exponentially with the system size. Therefore direct calculation of the partition function is not feasible. One way to circumvent this limitation is to use Monte-Carlo simulations. However, at low temperatures only very few relatively ordered states will give significant contributions to the partition sum, therefore the simulation is inefficient. With importance sampling each configuration's probability will be proportional to its importance in the partition sum. One such method is the Markov-chain Monte Carlo method. From each state we can reach other states with specific probabilities. When some conditions hold (for example ergodicity and detailed balance), the states converge to the required distribution. We will consider two algorithms: the Metropolis-Hastings and Wolff algorithms.

There are two issues with such methods: on one hand, they need time to reach their final distribution. This is called warm-up time. On the other hand, since configurations are generated from each other, successive configurations are highly correlated. This makes it difficult to get statistically independent data from measurements.

The Wolff algorithm was invented specifically to overcome this critical slowing down. The dynamical exponent (which describes roughly how much time an algorithm needs to generate independent data) is much smaller than the Metropolis algorithm.

In this project we use the Wolff algorithm to calculate the critical temperature of the Ising model, and show the difference between the dynamical exponents of the two algorithms.

II. THEORETICAL BASIS

Since there is no external field in our simulations, we will use the simplified Hamiltonian

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

There are some important quantities:

- The grid length L
- The number of spins $N = L^2$ or L^3
- The magnetic moment is the sum of spins: $M = \sum_i s_i$
- The magnetization: $m = M/N$
- The average energy per site: $\epsilon = H/N$
- The temperature T and the critical temperature T_c
- $\tilde{\beta} = 1/(k_B T)$ where k_B is Boltzmann's constant

A spin configuration's probability can be calculated from its energy:

$$P(\boldsymbol{s}) \sim \exp\left(-\tilde{\beta} H(\boldsymbol{s})\right)$$

From this we also see that the probability only depends on the product of J and $\tilde{\beta}$. Therefore we will set $J = 1$ in

* A final project for the Computational Physics lecture

all our simulations, as the behavior is similar at different J values.

The system can exhibit phase transitions: at a certain temperature (called the critical temperature), quantities like the susceptibility diverge or have an abrupt jump. At low temperature, almost all spins point to the same direction, while at high temperature, they point randomly. Around the critical temperature large clusters appear in which spins point to the same direction. The critical exponents $\alpha, \beta, \gamma, \nu$ are defined as

$$T_r = \frac{T - T_c}{T_c} = \frac{\tilde{\beta}_c - \tilde{\beta}}{\tilde{\beta}} \tag{2}$$

$$c \sim T_r^{-\alpha} \qquad m \sim (-T_r)^{\beta} \qquad \chi \sim T_r^{-\gamma} \qquad \xi \sim T_r^{-\nu} \tag{3}$$

where T_c, c, m, χ, ξ are the critical temperature, specific heat, magnetization, susceptibility and correlation length, respectively.

Finally, the dynamical exponent z is defined as

$$\tau \sim |T_r|^{z\nu}$$

A usual method to measure z is to assume that in a finite system the correlation length can not diverge, $\xi \approx L$. Then,

$$\tau \sim L^z \tag{4}$$

III. METHODS

The project is available at https://github.com/juhdanad/computational-physics-project-bonn-2021.

A. Algorithms

Metropolis-Hastings algorithm The algorithm examines a single spin's flip, and accepts or rejects it:

1. Start from an initial configuration (in our report, random spins)
2. Choose a random site
3. Calculate the energy change ΔH when we flip this spin
4. Accept the spin flip with probability $\min\left(1, \exp\left(-\tilde{\beta} \cdot \Delta H\right)\right)$, if rejected, stay on the previous configuration
5. Repeat from step 2

To reduce the correlation of measurements, we only do a measurement after we have done as many accept/reject steps as the number of spins.

Wolff algorithm Here we flip a whole cluster of spins.

1. Start from an initial configuration (in our report, random spins)
2. Choose a random site, flip it and add it to a stack
3. While the stack is not empty:
   (a) Pop an element from the stack
   (b) For all neighbors that point in the opposite direction, add them to the stack with probability $1 - \exp(-2\tilde{\beta}J)$, and flip all added spins

The algorithm terminates after a finite time, since all spins can be added to the stack only once (after that they are flipped).

B. Simulation strategies

Thermalisation time First we need to ensure that the system has enough time for thermalisation. We thought that a good indicator would be to simulate an ensemble of configurations, and plot the standard deviation of their magnetizations. At the beginning we start from random spins, where the magnetization is close to zero, so this standard deviation is small. But then there are two equally likely directions for the spins to align, so the standard deviation should increase, and converge to a value.

Correctness of algorithms It is a good practice to check our implementation. Therefore we plotted multiple values obtained by Metropolis and Wolff algorithms to see if the plots coincide. The comparison of the critical temperature with literature values is also a good indicator, if the two values are close.

C. Error analysis

For estimating errors we will do multiple measurements. These measurements will be started from random initial configurations, therefore we will not need to worry about correlation time. Assuming that the measurement values follow a normal distribution, we can estimate the error with 95% probability as

$$\sqrt{\frac{\mathrm{var}(x)}{N-1}} \cdot t_{N-1}$$

N is the number of measurements, $\mathrm{var}(x)$ is the variance of the measured values, and t_{N-1} is the inverse CDF of the student-t distribution at $N-1$ degrees of freedom, evaluated at 0.975.

algorithm	square grid	cubic grid
Metropolis	**10000** ($L \leq 120$)	**500** ($L \leq 20$)
Wolff	**6000** ($L \leq 140, \tilde{\beta} \geq 0.3$)	**3000** ($L \leq 25, \tilde{\beta} \geq 0.2$)

TABLE I. sufficient warm-up steps for different scenarios

D. Fitting

Critical temperature We use the scaling behavior of the susceptibility/cluster size in equation (3). Therefore, we:

1. Simulate the susceptibility's dependence on the temperature at different grid sizes.
2. Restrict attention to the part which is not significantly cut down by the finite size of the grid.
3. Simulate the largest grid on this parameter range.
4. Since $\chi \sim T_r^{-\gamma}$, we also have

   $\log \chi = \text{const.} - \gamma \log(T_r) = \text{const.} - \gamma \log(T - T_c)$

   where $T_r = \frac{T - T_c}{T_c}$.
5. Try different values of T_c, and try to fit a line on the $\log \chi$-$\log(T - T_c)$ plot.
6. The T_c where the R-value of the line is closest to ± 1 is the critical temperature.
7. Repeat steps 3-6 to get a bunch of T_c values for error estimation.

Dynamical exponent We use the relation in equation (4). Therefore, we simulate a system for some time, then calculate the normalized autocorrelation function of m and H. Then we fit a line on the log-plot of this autocorrelation function, and get the autocorrelation time. In the case of the Wolff algorithm we scale the time by $\langle C \rangle / N$, where $\langle C \rangle$ is the average cluster size. Finally, we fit a line on the log-log plot of the $\tau - N$ plot, and from the slope we get z.

IV. RESULTS

A. Thermalisation time

The results are in table I. Generally we got that the Wolff algorithm converges the slowest when there is a large grid and small $\tilde{\beta}$. This can be explained with the fact that at small $\tilde{\beta}$ values the clusters are small, so more clusters are needed to move away from the initial configuration. The Metropolis algorithm is the slowest around the critical temperature.

algorithm	calculated from m	calculated from H
Metropolis	1.5(72)	0.5(49)
Wolff	-0.77(1)	0.33(8)

TABLE II. dynamical exponents

B. Critical temperature

For the 2D Ising model we got $\tilde{\beta}_c = 0.4415(54)$. For the 3D Ising model we got $\tilde{\beta}_c = 0.221(74)$.

C. Dynamical exponent

The different dynamical exponents are in table II.

V. DISCUSSION

A. Critical temperature

The measured values are close to the literature values: the relative errors are 0.2% for the 2D case and 0.04% for the 3D case. However, in the 2D case the exact value is outside of the error bounds, that is, we have systematic errors. One likely explanation is that we used a large $\tilde{\beta}$ range for fitting the exponential curve, but the scaling relations (II) only hold very close to the critical temperature.

B. Dynamical exponent

We can see from the results that the dynamical exponent is smaller for the Wolff algorithm. However, only one value is similar to the literature value: for the Wolff algorithm, we got 0.33(8) and the literature value is 0.25.

VI. SUMMARY

We demonstrated that near the critical temperature the Wolff algorithm scales better than the Metropolis algorithm. However, our result is not close to the literature value.

[1] U. Wolff, "Collective Monte Carlo Updating for Spin Systems", Phys. Rev. Lett. 62 (1989) 361.

[2] Advanced Monte Carlo Methods notes by E. Carlon – Academic year 2016/2017

[3] P. J. Meyer, "Computational Studies of Pure and Dilute Spin Models", thesis, chapter 3
[4] M. Dílaver, S. Gündüç, M. Aydın, Y. Gündüç, "A New Measurement of Dynamic Critical exponent of Wolff Algorithm by Dynamic Finite Size Scaling", arXiv:cond-mat/0409696