Contents

Because of its pedagogical popularity, instructional value and physical importance

Abstract

In this project, we studied the Monte Carlo methods to ~~simulate and analyze a physical process.~~ calculate statistical properties of a classical system. We carried out Monte Carlo simulations for Ising Model in 2D. We have considered ~~3~~ 2 algorithms to simulate Ising Model, they are Metropolis Algorithm and ~~Swendsen - Wang Algorithm~~ Wolff Algorithm and we compared these 2 algorithms.

// artırılacak içerik ...

After studying and demonstrating the Metropolis algorithm we discuss the so called "critical slowing down" problem towards the critical temperature. A solution to this problem is given by the Wolff algorithm which is discussed at the end.

Acknowledgements

I would like to thank my supervisor Asst. Prof. Dr. A. Levent Subaşı for his help at every stage of this study, Asst. Prof. Dr. Ş. dsadadadsaa from Koç University for helping me understand crucial theoretical points and informing me about practical sides of the calculations and finally the important people I met in ITU Physics Engineering Department.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 History of Monte Carlo Method

In 1945's a team which is adminstrated by John Mauchly and Presper Eckert was working on the first computer, ENIAC at University of Pensylvania. Also, John von Neumann and Edward Teller were interested in thermonuclear problem at Los Alamos. When ENIAC was almost completed, Neumann decided to set a model for a thermonuclear reaction to test the ENIAC and to realize it, he asked Stan Frankel and Nicholas Metropolis to work with him. Then, after getting some information about the ENIAC, they started to work for the model at Los Alamos. However, because of the World War during this time period, they could not finish their work. After that, Stanislaw Ulam joined the laboratory at Los Alamos. Ulam's extensive mathematical background made him aware that statistical sampling techniques had fallen into desuetude because of the length and tediousness of the calculations.[**?**] At the laboratory, he recognized the first electronic computer ENIAC and he was impressed by the speed of ENIAC. Ulam was interested in branching process and Neutron multiplication in fission device and Ulam wanted to solve Neutron diffusion, statistically by using ENIAC. At that time, Metropolis suggested an obsious name for the statistical method - a suggestion not unrelated to the fact that Ulam had an uncle who would borrow money from relatives because he " just had to go to Monte Carlo". [**?**].

1.2 Random Numbers for Monte Carlo Simulation

Monte Carlo method is an statistical approach to physical processes. To realize Monte Carlo method in any computer, random sampling and random decisions should be encountered. Hence, we want to get random numbers from a computer.

Actually, firstly we should ask that how can we get a random number in nature ? To get a randomness from the nature, we have to analyze probabilistic nature of Quantum Mechanics. According to the Quantum Mechanics, for an unstable nuclei, when the radioactive decay will occur, is random. Hence, if we save the times that decaying occurs, we get truly random numbers. Thus, using radioactive decay, we may create random numbers for our simulation. Also, in the computers, people generate these random numbers, with respect to an algorithm; therefore, numbers are predetermined. We expect that these numbers created by algorithm to have an uniform distribution, to be in a definite range and to have minimum correlation. These correlations in the numbers means that actually, we don't create numbers that are completely random. Thus, we label them as 'pseudo-random' numbers. Hence, this lack of randomness effect the Monte Carlo simulation and causes an error in the simulation.

Suppose we want to create pseudo-random integer numbers(k_n) between 0 and L. Most widely used algorithm for that is $k_n = f(k_{(n-1)})$, f is a function that returns integers. The number that is generated is dependent on the previous number. To generate pseudo-random integer we need a 'seed' to start the algorithm. After, we input a seed to algorithm, it continuously creates pseudo-random numbers. However, if we get a number that was created before, we enter a cycle, and we get same numbers, periodically. Hence, we want that the cycle is very long because when we are creating pseudo-random numbers, we should not enter the cycle twice. For a more complex algorithm, $k_n = f(k_{(n-1)}, k_{(n-2)}, ...)$ can be used. With using transformations on created numbers, we can arrange the range in which numbers are generated.

This algorithm that we used to create numbers are determined, hence, these numbers are deterministic not random. If we enter same seeds to the algorithm we get same numbers. Thus, there is no randomness in the algorithm for a given seed. Numbers that are generated only obey the rules of algorithm and mathematics. And if true randomness cannot be created in any mathematical operation, then it will have to come from some physical process. [**?**]

1.3 Monte Carlo Method

In the most general terms, Monte Carlo method is a statistical - almost experimental - approach to computing integrals using random positions, called samples, whose distribution is carefully chosen. [**?**]

In Monte Carlo method 2 basic sampling method exist;

One of them is direct sampling. In the direct sampling method, we take each samples

independently for each step. Hence, our new samples are independent of the previous ones. With taking samples again and again, we sweep out the states and by using these samples, we can calculate integrals.

Other sampling method is Markov-Chain sampling. However, in the Markov-Chain sampling, our sample depends on the previous sample. Fistly, we start sampling from an initial state that is given or where the last simulation ends. Then, from the initial state, we move to another state, that we sample, in any direction and distance. This direction and distance again random, but, distance is limited to a value δ. With using this procedure, we visit other states and if the resulting state is a state that we don't want to be in there, then we reject the move. Hence, this limitation for the moves δ affects the rejection rate. If δ is very large, rejection rate is so high, this means that we generally don't move to another state; thus, our traveled path is small. Also, if δ is very small, acceptance rate is so high, this means that we generally move from state to another state; however, in this case our δ is small; therefore , our traveled path is, again, small. Thus, when δ is on the edge , very large or small, we can't sweep out most of the states and calculation of the integral by samples may fail.

Instead of calculating the integral analitically, we are taking samples and using these samples we approximate the integrals in discrete manner. Thus, Monte Carlo method allows the evaluation of high dimensional integrals, such as appear in statistical physics and other domains, if only we can think of how to generate the samples. [**?**]

/ markov chain ne zaman tercih edilmeli direct sampling e

1.4 Basic Implementations of the Monte Carlo Method

To understand the basic structure of the Monte Carlo method, we will illustrate 2 basic examples to calculate π. To calculate it, we take random samples from inside a square and we count the samples that are inside the circle which is surrounded by the square. In these examples, we used direct sampling and Markov-Chain sampling methods.

In the first example, we have a square in the range 0 and 1 and we have a one fourth circle inside it, which has a center at origin and radius 1. If we take the ratio of samples that are inside of the circle (hits) to all samples, we expect that this ratio is $r = \frac{\pi r^2}{4}$. In this example, we used direct sampling method to take samples. We take N=1000,10000, ... , 10^7 samples, respectively and we calculated the ratio of hits to all samples. And we estimate the mean square deviation. If we plot mean square deviation, $E[(\frac{Nhits}{N} - \frac{\pi}{4})^2]$, with respect to N, we get,

Hence, with direct sampling method, we can calculate π with high precision.

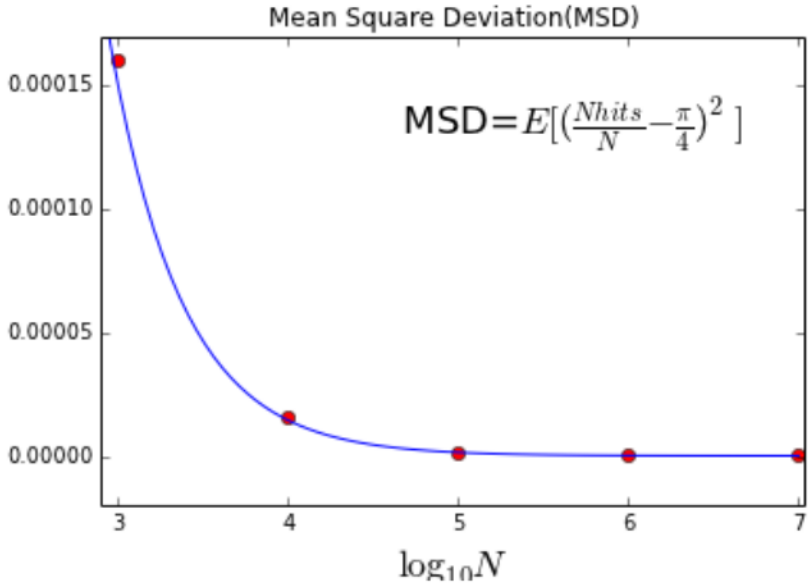


Figure 1.1: Mean Square Deviation for N = 1000,10000,... , respectively

/ fit in niye uydugunu acıkla. ?????

In the second example, we have a square centered at origin and length 2 and we have a circle inside it which has a center at origin and radius 1. Again, we expect that the ratio to be $r = \frac{\pi}{4}$. However, in this case we used Markov-Chain sampling instead of direct sampling. In this case we take N=$4 * 10^6$ samples. If we look at Mean square deviation,

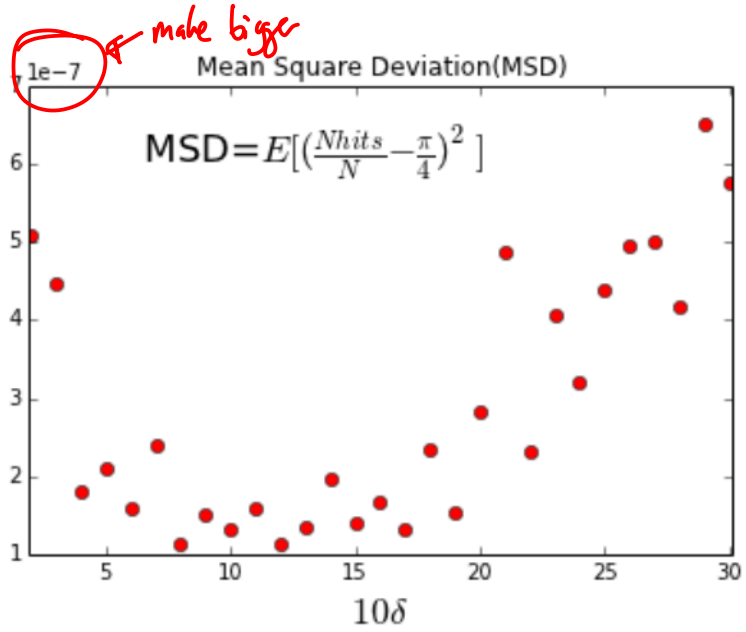


Figure 1.2: Mean Square Deviation for δ= 0.2,0.3,...,3.0 , respectively

We can see that when δ is 0.8 we get maximum precision.

If we look at rejection rate,

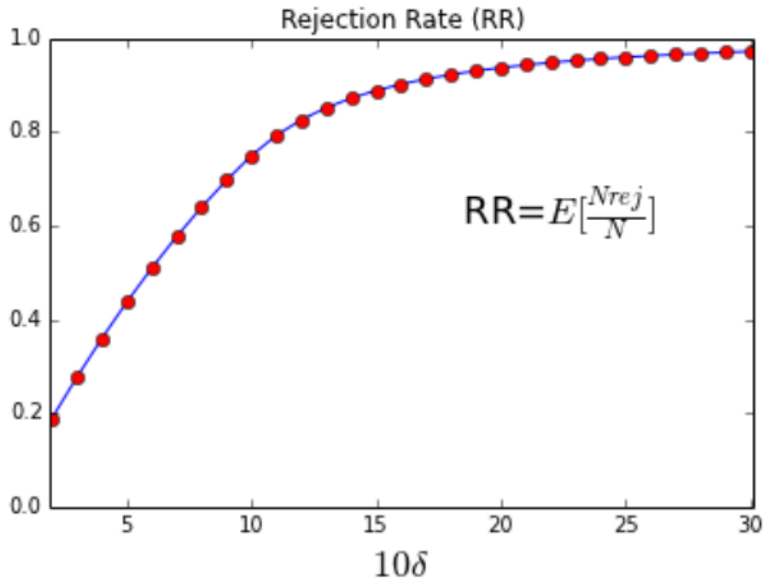


Figure 1.3: Rejection rate for δ= 0.2,0.3,...,3.0 , respectively

when δ is 0.8, we have a rejection rate is around 0.55.

/ veriler ne gosterdi , yorumlama gerekir, yeterli mi yapılanlar / ising ile karsılastırılabilir rejection rate

1.5 Central Limit Theorem (CLT)

/CLT nin niye yazıldıgını acıkla / bu bolum 2 ye ayrılabilir aslında ilk kısım olasılık kısmı definitionlar / expectation value, variance falan. ikinci kısım CLT

In various situations, histograms of measurements looks like a bell. This bell shaped curve is known as *Gaussian Distribution*. Gaussian distribution, which appears in a lot of practical applications, is a continuous probability distribution. Because Gaussian random variables arise in so many practical situations, staticians refer to them as normal random variables.[**?**].

Gaussian Distribution 1 *If X is an Gaussian Random Variable, the Probability Distribution Function (PDF) of X is*

$$f_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

where μ and σ are the parameters.

The parameter μ_x is the *Mean Value* or *Expected Value* of the distribution and σ_X is the *Standart Deviation* of the distribution and also ${\sigma_x}^2$ is the *Variance*. For a continuous probability distribution,

$$E[X] = \mu_x = \int_{-\infty}^{\infty} x f_X(x)\, dx \tag{1.1}$$

$$Var[X] = E[(X - \mu_x)^2] = E[X^2] - {\mu_x}^2 \tag{1.2}$$

Also there are 3 useful formulas for mean value and variance where $a, b \in \mathbb{R}$

$$E[aX + b] = aE[X] + b \tag{1.3}$$

$$Var[X + b] = Var[X] \tag{1.4}$$

$$Var[aX] = a^2 Var[X] \tag{1.5}$$

which can be easily proven.

When X and Y are 2 random variables

$$Cov[X, Y] = E[(X - \mu_x)(Y - \mu_y)] = E[XY] - \mu_x \mu_y \tag{1.6}$$

$$E[X + Y] = E[X] + E[Y] \tag{1.7}$$

$$Var[X + Y] = Var[X] + Var[Y] + 2Cov[X, Y] \tag{1.8}$$

again, they can be proven, directly.

Independence of 2 Random Variables *Random variables X and Y are independent if*

and only if

$$f_{X,Y}(x,y) = f_X(x)f_Y(y)$$

Hence, when X and Y are 2 independent random variables $Cov[X,Y] = 0$

Central Limit Theorem (CLT) explains why so many practical phenomena produce data that can be modeled as Gaussian random variables. CLT is used to calculate probabilities associated with the independent and identically distributed (i.i.d.) sum

$$W_n = X_1 + ... + X_n$$

In this case, $E[W_n] = n\mu_x$ and $Var[W_n] = nVar[X] = n{\sigma_x}^2$. As *n* approaches infinity, $E[W_n]$ and $Var[W_n]$ approach infinity, so there is no convergence in $E[W_n]$ and $Var[W_n]$. Therefore, to get rid of this divergence CLT will be explained in terms of standardized random variable

Standardized Random Variable 1

$$Z_n = \sum_{i=1}^{n} \frac{X_i - n\mu_x}{\sqrt{n{\sigma_x}^2}} = \frac{W_n - n\mu_x}{\sqrt{n{\sigma_x}^2}}$$

For standardized random variable Z_n, using above quantities

$$E[Z_n] = E\left[\sum_{i=1}^{n} \frac{X_i - n\mu_x}{\sqrt{n{\sigma_x}^2}}\right] = \frac{E\left[\sum_{i=1}^{n} X_i\right] - n\mu_x}{\sqrt{n{\sigma_x}^2}} = \frac{E[W_n] - n\mu_x}{\sqrt{n{\sigma_x}^2}} = 0 \qquad (1.9)$$

$$Var[Z_n] = Var\left[\sum_{i=1}^{n} \frac{X_i - n\mu_x}{\sqrt{n{\sigma_x}^2}}\right] = Var\left[\frac{\sum_{i=1}^{n} X_i}{\sqrt{n{\sigma_x}^2}}\right] = \frac{Var[W_n]}{n{\sigma_x}^2} = 1 \qquad (1.10)$$

Central Limit Theorem 1 *When $X_1, X_2, ...$ a sequence of i.i.d. random variables with μ_x and ${\sigma_x}^2$ the Cummulative Distribution Function (CDF) of $Z_n = \sum_{i=1}^{n} \frac{X_i - n\mu_x}{\sqrt{n{\sigma_x}^2}}$ satisfies*

$$\lim_{n\to\infty} F_{Z_n}(z) = \Phi(z)$$

where $\Phi(z)$ is CDF of standard normal distribution.

Standard normal distribution is a normal distribution with $\mu_x = 0$ and $\sigma_x = 1$.

Cummulative Distribution Function 1 *The cummulative distribution function of X is*

$$F_X(x) = P[X \leq x]$$

and

$$F_X(x) = \int_{-\infty}^{x} f_X(u)\, du$$

The sum of i.i.d. random variables W_n can be expressed in terms of Z_n with using inverse function of Z_n such that,

$$W_n = \sqrt{n{\sigma_x}^2} Z_n + n\mu_x \tag{1.11}$$

Then,

$$F_{Wn}(w) = P[W_n \leq w] = P[\sqrt{n{\sigma_x}^2} Z_n + n\mu_x \leq w] = F_{Zn}\left(\frac{w - n\mu_x}{\sqrt{n{\sigma_x}^2}}\right) \tag{1.12}$$

By using Central Limit Theorem,

$$\lim_{n\to\infty} F_{Zn}\left(\frac{w - n\mu_x}{\sqrt{n{\sigma_x}^2}}\right) = \Phi\left(\frac{w - n\mu_x}{\sqrt{n{\sigma_x}^2}}\right)$$

then for large numbers of n,

$$F_{Zn}\left(\frac{w - n\mu_x}{\sqrt{n{\sigma_x}^2}}\right) \approx \Phi\left(\frac{w - n\mu_x}{\sqrt{n{\sigma_x}^2}}\right) \tag{1.13}$$

The central limit theorem suggests that if the different components add to produce the measured data, the underlying random variable is Gaussian and CDF of W_n should approach a Gaussian CDF with the same mean and variance.[**?**]. This can be shown by using properties of expectation and variance.Also, CDF of Z_n should approach a Gaussian CDF with $\mu = 0$ and $\sigma = 1$.

$$E[Z_n] = E[\frac{W_n - n\mu_x}{\sqrt{n{\sigma_x}^2}}] = 0$$

$$E[W_n] = E[\sqrt{n{\sigma_x}^2} Z_n + n\mu_x] = n\mu_x$$

$$Var[Z_n] = Var[\frac{W_n - n\mu_x}{\sqrt{n{\sigma_x}^2}}] = 1$$

$$Var[W_n] = Var[\sqrt{n{\sigma_x}^2} Z_n + n\mu_x] = n{\sigma_x}^2$$

If Z_n is a continuous random variable, then the PDF of Z_n converges to a Gaussian

PDF. Also, When Z_n is a discrete random variable, although shapes of PDFs of Z_n and Gaussian are same, PDF values of Z_n differs from PDF values of Gaussian. There is a difference between PDF of Z_n and PDF of Gaussian, because for discrete PDF sum of heights equals to 1, while for continuous PDF area under PDF equals to 1.

If we draw a continuous curve through the heights of discrete PDF of Z_n, then we will find that area under that curve equals to the sum of heights of PDF, which equals to 1, multiplied by the distance between neighbouring PDF values, which equals to ϵ. Therefore,the area under that curve equals to ϵ. In order to make the area equals 1, we should multiply heights of discrete PDF by $\frac{1}{\epsilon}$. We know that,

$$Z_n = \sum_{i=1}^{n} \frac{X_i - n\mu_x}{\sqrt{n{\sigma_x}^2}}$$

thus

$$\epsilon = \frac{1}{\sqrt{n{\sigma_x}^2}}$$

Hence, by multiplying heights of discrete PDF of Z_n by $1/\epsilon$, we will get a discrete PDF approaches to Gaussian PDF for large n values.

For instance, let X_i s be a sequence of i.i.d. discrete random variables with uniform distribution in a range between 0 and 4. Also, $W_n = X_1 + X_2 + ... + X_n$. Hence, with respect to CLT, CDF of Z_n approximates to Gaussian CDF for large values of n, so are PDFs.

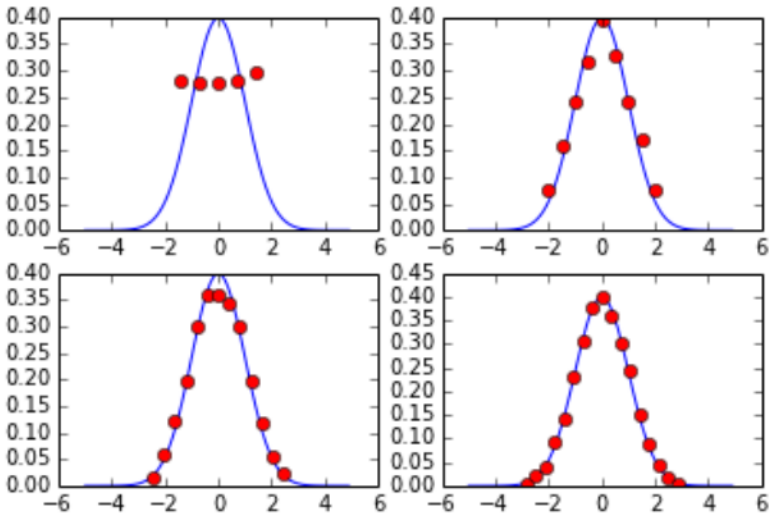
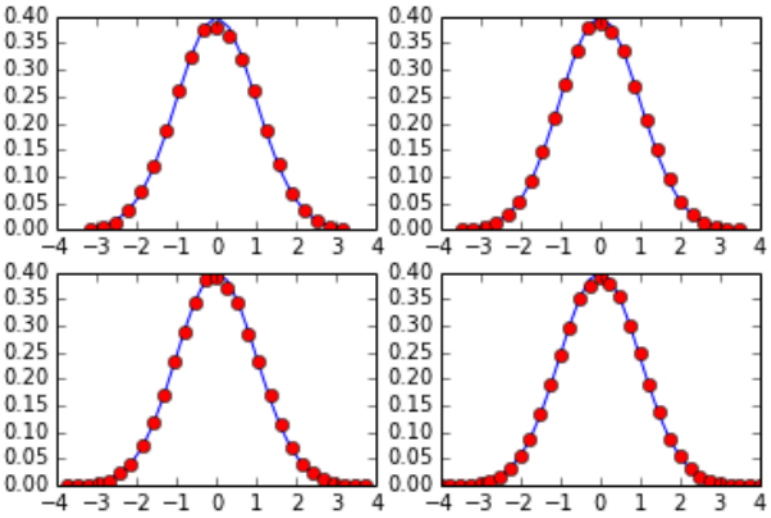


Figure 1.4: CLT for sampling with 1,2,3,4 states, respectively



axis labels

Figure 1.5: CLT for sampling with 5,6,7,8 states, respectively

Chapter 2

The Monte Carlo Method (MC)

2.1 Estimation of Expected Value

For a simulation of a system, the aim is to calculate expected value of a quantity. For instance, expected value of a quantity, Q, which follows Maxwell-Boltzmann distribution is

$$\langle Q \rangle = \frac{\sum_{\mu} Q_{\mu} e^{-\beta E_{\mu}}}{\sum_{\mu} e^{-\beta E_{\mu}}}$$

where μ is a state of the system and $\beta = \frac{1}{kT}$

By this method of calculating expected value of a quantity, all states of the system must be considered and this can be realized for systems with small number of states. However, when we consider larger systems, to calculate expected value of a quantity of system, a subset of these states must be used since all states cannot be considered. In this case, because of using small portion of states, there may be error in calculated expected value. By Monte Carlo method, expected value of a large system can be calculated with low error.

Monte Carlo method selects states of system, that is used to calculate expected value of a quantity, from a probabilty distribution, P_{μ} which we specify. With respect to Monte Carlo method, expected value of a quantity is

$$Q_M = \frac{\sum_{i=\mu}^{M} Q_{\mu_i} {P_{\mu_i}}^{-1} e^{-\beta E_{\mu_i}}}{\sum_{j=\mu}^{M} {P_{\mu_j}}^{-1} e^{-\beta E_{\mu_j}}} \tag{2.1}$$

where states of system are $\{\mu_1, ..., \mu_M\}$ and Q_M is *estimator of* Q. In the formulation, we see ${P_{\mu_i}}^{-1}$ because we choose state Q_{μ_i} by a probabilty P_{μ_i} from all of the states.

Now, for an accurate estimation of expected value, P_μ must be defined. For instance, P_μ may be chosen as equal for all states. Then,

$$Q_M = \frac{\sum_{i=1}^{M} Q_{\mu_i} e^{-\beta E_{\mu_i}}}{\sum_{j=1}^{M} e^{-\beta E_{\mu_j}}}$$

This is a poor choice for P_μ. If probabilities of some states are greater than others, to get a good estimation of expected value, these states with high probabilities must be used. Therefore, equal P_μ for all states are not appropriate for this system, P_μ corresponding these states of high probabilities arranged to be high with respect to others. Hence, it is obvious that, if P_μ is chosen in the form of Maxwell-Boltzmann distribution, then better approximation to expected value will occur.

On the other hand, if the number of states used to calculate expected value of a quantity by Monte Carlo method is increased, then estimator of Q converges to $\langle Q \rangle$.

$$\lim_{M \to \infty} Q_M = \langle Q \rangle$$

2.2 Sampling

To estimate expected value of a system which has high number of states, Monte Carlo method should be used. For a system with states which has not equal probabilities, an appropriate P_μ must be defined, in order to approximate expected value of system. We have specified that if we choose P_μ in the form of Maxwell-Boltzmann distribution, then we get better approximation to expected value.

In this case, if we choose P_μ in the form,

$$P_\mu = Z^{-1} e^{-\beta E \mu} \tag{2.2}$$

then

$$Q_M = \frac{1}{M} \sum_{i=1}^{M} Q_{\mu_i} \tag{2.3}$$

This sampling is called *Importance Sampling*. Z is a normalization constant for probability

P_μ. In the following chapter, it is called *Partition function*; since, if Z is known, then all macroscopic properties of a system can be calculated.

2.3 Markov Process and Markov Chain

~~How to pick states to get a good approximation of expected value of a system is defined.~~ Now, the question is *how* exactly we pick our states so that each one appears with its correct Maxwell-Boltzmann Probability. [**?**]. Hence, the problem is the generation of a random set of states according to Maxwell-Boltzmann distribution.

Markov Process

Markov Process is a mechanism found by the Russian Mathematician Andrey Markov. In Markov Process, for a given state μ, mechanism creates a new random state ν. Probability to move state from μ to ν is *Transition Probability*, $P(\mu \to \nu)$.

Transition probabilities must satisfy two conditions: Transition probabilities should be time independent and depend only on initial and final states, μ, ν, respectively. Also, it is obvious that,

$$\sum_{\nu} P(\mu \to \nu) = 1 \tag{2.4}$$

which says that, sum of probabilities of transition from the state μ to other states ν and probability of staying in same state is equal to 1, as we expected.

Markov Chain

When simulating a system with Monte Carlo method, Markov Process is used repeatedly, as a chain, and a *Markov Chain* is achieved. Simulation moves through the states, $\mu \to \nu \to \delta \to \lambda \to ....$

The Markov Process is chosen specially so that when it is run for long enough starting from any state [redacted] it will eventually produce a succession of states which appear with probabilities given by the Maxwell-Boltzmann distribution (We call the process of reaching Maxwell-Boltzmann distribution "coming to equilibrium", since it is exactly the process that a real system goes through with its "analogue computer" as it reaches equilibrium at the ambient temperature.)[**?**].

To achieve equilibrium there are 2 conditions:

1. Ergodicity:

With respect to condition of ergodicity, some of transition probabilities may be zero, however through any two states, which has transition probabilty equals to zero, there must be at least one path of non-zero probability. Thus, all states in the system must be reachable at least one path, it means that all states must be a part of the system.

2. Detailed Balance:

   Using detailed balance, after achieving the equilibrium condition, we obtain Maxwell-Boltzmann distribution rather than any distribution.

   If we run the simulation for a long time, we will reach equilibrium condition and equilibrium condition states that, transitions into a state and out of that state must be equal. Thus, *equilibrium condition* can be written as following,

$$\sum_{\nu} P_{\mu} P(\mu \to \nu) = \sum_{\nu} P_{\nu} P(\nu \to \mu) \tag{2.5}$$

   Therefore,

$$P_{\mu} = \sum_{\nu} P_{\nu} P(\nu \to \mu) \tag{2.6}$$

   Hence, if above equation is satisfied, equilibrium is achieved. However, this equilibrium condition is not sufficient to get Maxwell-Boltzmann distribution, after run the simulation for a long time although we choose probabilities appropriately.

   To get Maxwell-Boltzmann distribution, after arranging appropriate probabilities, we should reach *simple equilibrium*, by avoiding *dynamic equilibrium*.

   / simple equilibrium ve dynamic equilibrium daha düzgün acıkla

   (a) Simple Equilibrium:

   Let **P** is a matrix that consists of transition probabilities $P(\mu \to \nu)$, **P** is called *Markov Matrix* or *Stochastic Matrix* and let $\omega_{\mu}(t)$ is probability of system is in a state μ at time t. Then,

$$\omega_{\nu}(t+1) = \sum_{\mu} P(\mu \to \nu) \omega_{\mu}(t)$$

   With using matrix notation

$$\underline{\omega}(t+1) = \mathbf{P}\underline{\omega}(t) \tag{2.7}$$

$\underline{\omega}(t)$ is an array consists of $\omega_\mu(t)$.

If simulation reaches simple equilibrium, then

$$\underline{\omega}(\infty) = \mathbf{P}\underline{\omega}(\infty) \tag{2.8}$$

In this case, $\underline{\omega}(\infty)$ equals Maxwell-Boltzmann distribution, by choosing appropriate probabilities.

(b) Dynamic Equilibrium:

Dynamic equilibrium is in which $\underline{\omega}$ rotates around different values. This rotation is *limit cycle*.

Hence, in order to get Maxwell-Boltzmann distribution we must get rid of dynamic equilibrium. Thus, we need an additional condition, *Detailed Balance*:

$$P_\mu P(\mu \to \nu) = P_\nu P(\nu \to \mu) \tag{2.9}$$

/detailed balance kullanılınca dynamic equilibriumdan uzak durulduğunun kanıtını yap. ya da kitabı referans göster.

explain

If we look at detailed balance condition, it also includes equilibrium condition which is described above. Hence, if detailed balance is satisfied, then equilibrium condition will be satisfied and we will avoid to get dynamic equilibrium and limit cycles. Then, after reaching equilibrium, we can get Maxwell-Boltzmann distribution by arranging probabilities. ?

Once we remove limit cycles in this way (by using detailed balance), it is straightforward to show that the system will always tend to the probabilty distribution P_μ as $t \to \infty$. [**?**]. It means that $\omega_\mu(\infty) = P_\mu$.

Let's show that the system tend to P_μ as $t \to \infty$;

It is known that, $\underline{\omega}(t+1) = \mathbf{P}\underline{\omega}(t)$. If t is chosen to be equal to zero, we will get $\underline{\omega}(1) = \mathbf{P}\underline{\omega}(0)$ and by making some iterations, then

$$\underline{\omega}(t) = \mathbf{P}^t\underline{\omega}(0) \tag{2.10}$$

Also, $\underline{\omega}(0)$ can be expressed as a linear combination of eigenvectors $\underline{v_i}$ or $\mathbf{P}$,

$$\underline{\omega}(0) = \sum_i a_i \underline{v_i} \tag{2.11}$$

/ nasıl yazıldı? kanıtla

Then

$$\underline{\omega}(t) = \mathbf{P}^t \sum_i a_i \underline{v_i}$$

Also, we know that $\mathbf{P}\underline{v_i} = \lambda_i \underline{v_i}$ where λ_i is eigenvalue corresponding eigenvector $\underline{v_i}$. Then,

$$\underline{\omega}(t) = \sum_i a_i {\lambda_i}^t \underline{v_i} \tag{2.12}$$

Besides, we know that $\sum_\nu P(\mu \to \nu) = 1$. Thus, sum of all elements in each column of Markov matrix equals to 1. On the other hand, since elements of Markov matrix describes probabilities, they are in the range [0,1].

Let's assume a 2x2 Markov Matrix **P**;

$$\begin{bmatrix} a & b \\ 1-a & 1-b \end{bmatrix}$$

and transpose of **P** is $\mathbf{P}^T$

$$\begin{bmatrix} a & 1-a \\ b & 1-b \end{bmatrix}$$

To find its eigenvectors and eigenvalues $\mathbf{P}^T v_i = \lambda_i \underline{v_i}$ then, $(\mathbf{P}^T - \lambda_i I)\underline{v_i} = 0 = \mathbf{K}\underline{v_i}$ then **K** is

$$\begin{bmatrix} a-\lambda & 1-a \\ b & 1-b-\lambda \end{bmatrix}$$

Hence, for generally for transpose of a n by n Markov Matrix **P**, eigenvector $\underline{v_i}$ is

$$\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

and for this eigenvector which has elements of 1, there is an eigenvalue of 1. To see this, using same 2x2 Markov Matrix, and $\mathbf{P}^T\underline{v_i} = \lambda_i \underline{v_i}$; $a - \lambda + 1 - a =$

0 $b + 1 - b - \lambda = 0$ then $\lambda = 1$.

Markov Matrix , **P**, and its transpose have same determinant, then $\mathbf{P} - \lambda I$ and $\mathbf{P}^T - \lambda I$ have same determinant. Hence, eigenvalues of **P** and $\mathbf{P}^T$ are same (because determinants are equal). Therefore, Markov Matrix **P** has an eigenvalue of 1.

Now, let's assume v_i is an eigenvector corresponding to eigenvalue $|\lambda| > 1$ and $\mathbf{P}^n \underline{v_i} = |\lambda|^n \underline{v_i}$, so its lenght grows exponentially for n goes to ∞. Hence, for a large n, an element of $\mathbf{P}^n$ must be larger than one (to satisfy equality), which is impossible. So, all eigenvalues of **P** is small or equal to 1.

Also, we have found

$$\underline{\omega}(t) = \sum_i a_i {\lambda_i}^t \underline{v_i}$$

as t goes to ∞ eigenvalues which has an absolute value is smaller than 1 vanishes and right hand side of equation dominated by eigenvalue $\lambda_0 = 1$. Hence, for a long time $\underline{\omega}(t)$ is proportional to $\underline{v_0}$ which is eigenvector corresponding λ_0 that has elements of 1. Hence,

$$\underline{\omega}(\infty) = a_0 \underline{v_0} \tag{2.13}$$

Besides, in equilibrium, we have known that $P_\mu = \sum_\nu P_\nu P(\nu \to \mu)$ with vector notation, it is equal to $\underline{p} = \mathbf{P}\underline{p}$. $\underline{\text{p}}$ is a vector whose elements are P_μ. For an eigenvalue and a corresponding eigenvector of matrix **A**, $\mathbf{A}\underline{v} = \lambda \underline{v}$ Hence, if we write

$$\mathbf{P}\underline{p} = \lambda \underline{p}$$

, then $\underline{\text{p}}$ is a normalized eigenvector of **P** which has eigenvalue 1. It means that $\underline{v_0} = \underline{p}$

Therefore, $\underline{\omega}(\infty)$ is $\underline{p}$, because of that, when t goes to infinity, $\underline{\omega}(t)$ goes to $\underline{p}$.

$$\lim_{t \to \infty} \underline{\omega}(t) = \underline{p} \tag{2.14}$$

By carefully choosing transition probabilities which satisfy detailed balance condition, we can reach any probability distribution $\underline{p}$ in equilibrium.

Now, we want to satisfy Maxwell-Boltzmann distribution in equilibrium condition, therefore we choose p_μ as the Maxwell-Boltzmann probabilities. By using detailed

balance condition;

$$\frac{P(\mu \to \nu)}{P(\nu \to \mu)} = \frac{P_\nu}{P_\mu} = e^{-\beta(E_\nu - E_\mu)} \tag{2.15}$$

So we have to satisfy 3 conditions:

(a) $\frac{P(\mu \to \nu)}{P(\nu \to \mu)} = \frac{P_\nu}{P_\mu} = e^{-\beta(E_\nu - E_\mu)}$

(b) $\sum_\nu P(\mu \to \nu) = 1$

(c) Ergodicity

If these 3 conditions are satisfied, then we get Maxwell-Boltzmann distribution as an equilibrium distribution.

There are a lot of transition probabilities to satisfy these conditions. There is a lot of algorithms to realize it such as *Metropolis algorithm*. However, a purpose-built algorithm can often give a much faster simulation than an equivalent standard algorithm, and the improvement in efficiency can easily make the difference between finding an answer to a problem and not finding one. [**?**]

2.4 Acceptance Ratio and Selection Probabilities

We have 2 equations to satisfy, one is the ratio of transition probabilities and the other is the sum of transition probabilities to move a state to other states.

Let's consider "stay-at-home" condition $\nu = \mu$, then ratio of transition probabilities equals 1, so first condition (detailed balance) is satisfied, independent of $P(\mu \to \mu)$. So, we can choose other transition probabilities more easily. This flexibilty comes from the $\sum_\nu P(\mu \to \nu) = 1$, we can adjust any $P(\mu \to \nu)$. For a adjustment in $P(\mu \to \nu)$, we can adjust $P(\nu \to \mu)$ to keep the ratio constant. So, by changing $P(\mu \to \mu)$, we can use any value of $P(\mu \to \nu)$.

Let's assume

$$P(\mu \to \nu) = g(\mu \to \nu) A(\mu \to \nu)$$

where, $g(\mu \to \nu)$ is the *Selection probability*, which is the probabilty of generating new state and $A(\mu \to \nu)$ is the *Acceptance Ratio*. It represents that if our algortihm creates a new state ν, then with a fraction of time $A(\mu \to \nu)$ we will accept that new state ν and we will stay in first state at the rest of time. Acceptance ratios are about the "stay-at-home"

conditions, for example if we choose $A(\mu \to \nu) = 0$ then $P(\mu \to \mu) = 1$. So we are free to choose any value of $A(\mu \to \nu)$ in between 0 and 1.

If we look at the equation $\frac{P(\mu\to\nu)}{P(\nu\to\mu)} = \frac{P_\nu}{P_\mu} = e^{-\beta(E_\nu - E_\mu)}$, then we have a constarint in

$$\frac{P(\mu \to \nu)}{P(\nu \to \mu)} = \frac{g(\mu \to \nu)A(\mu \to \nu)}{g(\nu \to \mu)A(\nu \to \mu)}$$

We can choose any value of $\frac{A(\mu\to\nu)}{A(\nu\to\mu)}$ and this gives us a freedom in the value of $g(\mu \to \nu)$ and $g(\nu \to \mu)$. Hence, we will create a MC algorithm that creates new states with probabilities $g(\mu \to \nu)$ and then we will accepts it with probabilities $A(\mu \to \nu)$.

However, there is also a requirement in the values of $A(\mu \to \nu)$. If we choose acceptance ratios to be low, then we cannot generate enough states. Thus, we want to choose $A(\mu \to \nu)$ as close as to unity. For chosen selection probabilities, we only have a constraint on the ratio $\frac{A(\mu\to\nu)}{A(\nu\to\mu)}$. Hence, we choose largest acceptance ratios as 1, that implies $P(\mu \to \mu) = 1$, and the other is fixed by the ratio of acceptance probabilities.

Chapter 3

The Ising Model and Algorithms

3.1 The Ising Model

The Ising model is the model of magnetization of a material. The magnetization of a material is formed by magnetic dipole moments of spins. Ising models material as a lattice which includes spins. Spin can be +1 or -1 and it is represented by s_i. +1 or -1 spins stand for up-pointing or down-pointing dipoles. For a system which has N spins, there are 2^N total states. These spins interact with each other. We consider the simplest case, all interactions has the same strength, which is represented by J (interaction energy), and the interactions occurs only between the neighbouring spins. Also, we can deal with an external magnetic field acting on spins in this simplest case. In this case Hamiltonian (H_E) is;

$$H_E = -J \sum_{<i,j>} s_i s_j - H \sum_i s_i \tag{3.1}$$

We know that, Hamiltonian of Ising model represents Energy of system. Partition function of the Ising model is

$$Z = \sum_{\{s_i\}} e^{-\beta H_E} \tag{3.2}$$

$\{s_i\}$ means that we actualize the sum for all spins and for all values of spins, which are $+1$ and -1. Hence, $\{s_i\}$ represents all possible microstates of the Ising model.

If a large system, which has a fixed volume and fixed number of particles, is in interaction with environment, then we call this system *cannonical ensemble*. Canonical partition function for an cannonical ensemble is $Z = \sum_{\{i\}} e^{-\beta E_i}$ where i is the microstate of the

system and E_i is the energy at that state. Also, $\beta = \frac{1}{kT}$.

As previously defined, Z is a normalization constant for P_μ and macroscopic properties of system can be calculated by using Z.

We know that expectation of a quantity is $\langle Q \rangle = \frac{\sum\limits_{\mu} Q_\mu e^{-\beta E_\mu}}{\sum\limits_{\mu} e^{-\beta E_\mu}}$ which follows Maxwell-Boltzmann distribution. We can write it in terms of Z

$$\langle Q \rangle = \frac{1}{Z} \sum_{\mu} Q_\mu e^{-\beta E_\mu} \tag{3.3}$$

Expectation value of energy is internal energy U.

$$\langle E \rangle = U = \frac{1}{Z} \sum_{\mu} E_\mu e^{-\beta E_\mu} = -\frac{1}{Z} \frac{\partial Z}{\partial \beta} = -\frac{\partial log Z}{\partial \beta} \tag{3.4}$$

Specific heat, C, is the required heat to increase the temp one degree. Thus,

$$C = \frac{\partial U}{\partial T} = \frac{\partial U}{\partial \beta} \frac{\partial \beta}{\partial T} = \left[-\frac{\partial^2 log Z}{\partial \beta^2} \right] \left[-k\beta^2 \right] = k\beta^2 \frac{\partial^2 log Z}{\partial \beta^2} \tag{3.5}$$

Entropy, S, is defined as

$$C = T \frac{\partial S}{\partial T} = -\beta \frac{\partial S}{\partial \beta} \tag{3.6}$$

$$S = -k\beta \frac{\partial log Z}{\partial \beta} + k log Z \tag{3.7}$$

(There is also a constant in S but according to the 3rd law of Thermodynamics it is 0.) Helmholtz free energy, F, is defined as,

$$F = U - TS = -kT log Z \tag{3.8}$$

Pressure, p, is conjugate variable of volume V. Hence,

$$p = -\frac{\partial F}{\partial V} \tag{3.9}$$

Also, magnetization, M, is conjugate variable of magnetic field. Thus,

$$M = \frac{\partial F}{\partial H} \tag{3.10}$$

Then, we can calculate magnetic susceptibility, χ by

$$\chi = \frac{\partial \langle M \rangle}{\partial H} \tag{3.11}$$

We will simulate the Ising model with Monte Carlo Method with using different algorithms.

3.2 Macroscopic Properties of Ising Model

In this section, we will look at the properties of Ising model that will be studied in Monte Carlo simulations. In the Monte Carlo simulations, we will calculate:

1. Magnetization per site

   Instead of calculating magnetization with using free energy, we will calculate it more easily.

   Mean magnetization per site can be calculated using,

   $$\langle m \rangle = \frac{1}{N} \left\langle \sum_i s_i \right\rangle \tag{3.12}$$

2. Internal Energy per site

   Instead of calculating internal energy with using partition function, we will calculate it with using definition of energy.

   Internal energy per site can be calculated using,

   $$u = \frac{1}{N} \langle H_E \rangle = \frac{1}{N} \langle -J \sum_{<i,j>} s_i s_j - H \sum_i s_i \rangle \tag{3.13}$$

3. Specific Heat per site

   Specific heat per site is defined as $c = \frac{1}{N}\frac{\partial U}{\partial T} = \frac{k\beta^2}{N}\frac{\partial^2 log Z}{\partial \beta^2}$

   $$c = \frac{k\beta^2}{N}\frac{\partial}{\partial \beta}\left(\frac{1}{Z}\frac{\partial Z}{\partial \beta}\right) = \frac{k\beta^2}{N}\left[\frac{1}{Z}\frac{\partial^2 Z}{\partial \beta^2} - \frac{1}{Z^2}\left(\frac{\partial Z}{\partial \beta}\right)^2\right] = \frac{k\beta^2}{N}[\langle E^2 \rangle - \langle E \rangle^2] \tag{3.14}$$

~~This is Fluctuation Dissipation Theorem and~~

$$\langle E^2 \rangle = \frac{1}{Z} \sum_{\mu} {E_\mu}^2 e^{-\beta E_\mu} = \frac{1}{Z} \frac{\partial^2 Z}{\partial \beta^2} \tag{3.15}$$

// formulu kontrol et hata var 2. terimde ????

$$c = k\beta^2 N[\langle E_{ps}^2 \rangle - \langle E_{ps} \rangle^2] \tag{3.16}$$

where E_{ps} is energy per site.

4. Magnetic Susceptibility per site

   Magnetic susceptibility per site is defined as $\chi = \frac{1}{N} \frac{\partial \langle M \rangle}{\partial H}$

   If we use Fluctuation Dissipation Theorem for Magnetic susceptibility we get

$$\chi = \frac{\beta}{N}[\langle M^2 \rangle - \langle M \rangle^2] \tag{3.17}$$

$$\chi = \beta N[\langle m^2 \rangle - \langle m \rangle^2] \tag{3.18}$$

3.3 How to Analyze Properties with Monte Carlo Method

Now, we know how to calculate magnetization per site and energy per site for Ising model in a Monte Carlo simulation for any state of system with considering only spins in that state. After calculation magnetization per site and energy per site, we can calculate mean of them. Also, other 2 properties of the system, specific heat and magnetic susceptibility, are depend on the fluctations on the magnetization and energy. Hence, to calculate them, we have to analyze magnetization and energy, previously.

3.3.1 Equilibration Time

To ensure that our states follow Maxwell-Boltzmann distribution, we have to make sure that we are in equilibrium. Hence, after the simulation started, to make observations of system properties, we should wait a period of time for system to get equilibrium. This

time period is called equilibration time τ_{eq}. When equilibrium is achieved, probability of the system being in a state is proportional to Maxwell-Boltzmann distribution.

As previously defined, when equilibrium is reached, system spends most of the time in states which have small range of energies. Hence, to see the system to get equilibrium, we can observe energy or magnetization.

3.3.2 Autocorrelation Function and Correlation Times

If we want to analyze system more efficiently, we should use independent datas in our simulation. Hence, to get independent datas from the simulation, we should use states which are independent from each other. To get independent states, we have to wait a period of time between 2 independent state, this period of time is called *Correlation time* τ of simulation after system achieve the equilibrium. Correlation time is a measure of how long it takes the system to get from one state to another one which is significantly different from the first. [**?**]. To calculate correlation time, time displaced autocorrelation function should be considered.

Time Displaced Autocorrelation Function

For example, let's consider the quantity magnetization. Time-Displaced Autocorrelation $\chi(t)$ of magnetization is;

$$\chi(t) = \int dt'[m(t') - \langle m \rangle][m(t'+t) - \langle m \rangle] = \int dt'[m(t')m(t'+t) - \langle m \rangle^2] \quad (3.19)$$

where, m(t) is magnetization at time t. $\chi(t)$ gives correlation of 2 different values of magnetization with t times difference. If $\chi(t) \neq 0$, then on average, the fluctations are correlated, otherwise, uncorrelated.

Autocorrelation is expected to fall off exponential; hence,

$$\chi(t) \sim e^{-t/\tau} \quad (3.20)$$

where τ is the correlation time.

When t = τ , $\chi(\tau) = 1/e = 0.367879$, this is still an high value. Hence, the time needed for independent samples is greater then correlation time. For some considerations, this time is chosen as 2τ. This is came from some analysis in error calculation.

There are 3 basic ways to determine correlation time;

Autocorrelation function is an exponential decaying function, Hence it has its maximum value when t=0. If the autocorrelation function is normalized with its value at t=0, then the max value of this function is 1, at t=0. Then, if $\frac{\chi(t)}{\chi(0)}$ is plotted with respect to t; then, correlation time can be found by searching the value $\frac{\chi(t)}{\chi(0)} = \frac{1}{e}$.

Also, correlation time may be calculated with using *Integrated Correlation time*

$$\int_0^\infty \frac{\chi(t)}{\chi(0)} dt = \int_0^\infty e^{-t/\tau} dt = \tau \tag{3.21}$$

Another way to find correlation time is to plot logarithm of normalized autocorrelation function with respect to t, logarithm of normalized autocorrelation function is $log(\frac{\chi(\tau)}{\chi(0)}) = \frac{-t}{\tau}$. Then, by finding the slope of the plot, we can measure correlation time. Slope is equal to $\frac{-1}{\tau}$.

To practise autocorrelation function in the simulation, autocorrelation function should be calculated in discrete time domain.

$$\chi(t) = \frac{1}{t_{max} - t} \sum_{\nu=0}^{t_{max}-t} m(t')m(t'+t) - \frac{1}{t_{max} - t} \sum_{\nu=0}^{t_{max}-t} m(t') * \frac{1}{t_{max} - t} \sum_{\nu=0}^{t_{max}-t} m(t'+t) \tag{3.22}$$

Then $\chi(t)/\chi(0)$ should be plotted. Then, with above analysis, we can determine correlation time. However, this formula to calculate autocorrelation function takes time proportional to n^2 where n equals to number of samples.

To decrease this time, Fourier Transform of the autocorrelation function should be calculated. Then, by taking inverse Fourier Transform, autocorrelation function can be calculated.

Fourier Transform of autocorrelation function;

$$\chi_{FT}(\omega) = \int dt e^{i\omega t} \int dt' [m(t') - \langle m \rangle][m(t'+t) - \langle m \rangle] \tag{3.23}$$

With using *Cross-Correlation Theorem* of Fourier Transform

$$\chi_{FT}(\omega) = \mid m'_{FT}(\omega) \mid^2 \tag{3.24}$$

where $m'_{FT}(\omega)$ is Fourier Transform of $m'(t) = m(t) - \langle m \rangle$

Fourier Transform of $m'(t)$ can be calculated using *Fast Fourier Transform (FFT)*. FFT takes time proportional to $nlogn$, where, n equal to number of samples. Hence,

firstly FFT of $m'(t)$ should be taken and $\chi_{FT}(\omega)$ should be calculated with given function. Then taking Inverse FFT of $\chi_{FT}(\omega)$, $\chi(t)$ can be found. After that, correlation time can be calculated with normalizing $\chi(t)$ and using above one of the methods to find correlation time.

Also, if $\langle m \rangle$ is calculated over a much longer run than $\chi_{FT}(\omega)$, then correlation time can be found without taking inverse FFT.[**?**]. From the definiton of the Fourier Transform;

$$\chi_{FT}(0) = \int_0^\infty \chi(t)dt = \tau\chi(0) \tag{3.25}$$

then,

$$\tau = \frac{\chi_{FT}(0)}{\chi(0)} \tag{3.26}$$

where, $\chi(0) = \langle m^2 \rangle - \langle m \rangle^2$, which is variance in magnetization.

Correlation Times and Markov Matrices

There are a lot of correlation times which correspond to states of the system and the interaction between correlation times may cause, the method of finding correlation time above, be inaccurate.

As previously mentioned, $\underline{\omega}(t+1) = \mathbf{P}\underline{\omega}(t)$ where , $\underline{\omega}(t)$ is a vector of probabilities of finding the states. By iteration $\underline{\omega}(t) = \mathbf{P}^t\underline{\omega}(0)$. Also, $\underline{\omega}(0)$ can be expressed as a linear combination of eigenvectors of $\mathbf{P}$, $\underline{\omega}(0) = \sum_i a_i\underline{v_i}$. Then $\underline{\omega}(t) = \mathbf{P}^t \sum_i a_i\underline{v_i}$. Also, we know that $\mathbf{P}\underline{v_i} = \lambda_i\underline{v_i}$ where λ_i is eigenvalue corresponding eigenvector $\underline{v_i}$. Then, $\underline{\omega}(t) = \sum_i a_i{\lambda_i}^t\underline{v_i}$.

As t goes to ∞, largest eigenvalue, λ_0 of $\mathbf{P}$ dominates. Then, $\underline{\omega}(t)$ is proportional to v_0 which is eigenvector corresponding the largest eigenvalue. Also, $\underline{\omega}(t)$ tends to Maxwell-Boltzmann distribution.

$$\underline{\omega}(\infty) = a_o\underline{v_0} \tag{3.27}$$

Expectation value of a quantity Q at time t is

$$Q(t) = \sum_\mu \omega_\mu(t)Q_\mu = \underline{q}\underline{\omega}(t) \tag{3.28}$$

Then,

$$Q(t) = \sum_i a_i {\lambda_i}^t \underline{q v_i} = \sum_i a_i {\lambda_i}^t q_i \tag{3.29}$$

where, $q_i = \underline{q v_i}$, q_i is the expectation value of Q in the i^{th} eigenstate.

As t goes to ∞, $Q(\infty) = a_0 q_0$, $Q(\infty)$ is proportional to q_0 denoted by λ_0. By defining,

$$\tau_i = \frac{-1}{log \lambda_i}, i \neq 0 \tag{3.30}$$

then,

$$Q(t) = a_0 q_0 + \sum_{i \neq 0} a_i q_i e^{\frac{-t}{\tau_i}} = Q(\infty) + \sum_{i \neq 0} a_i q_i e^{\frac{-t}{\tau_i}} \tag{3.31}$$

where, τ_i's are the correlation times for system. Also, $Q(\infty)$ is the equilibrium expectation $\langle Q \rangle$.

Autocorrelation function of Q can be written as,

$$\chi(t) = [Q(0) - Q(\infty)][Q(t) - Q(\infty)] = \sum_{i \neq 0} a_i q_i \sum_{i \neq 0} a_i q_i e^{\frac{-t}{\tau_i}} \tag{3.32}$$

Then,

$$\chi(t) = \sum_{i \neq 0} b_i e^{\frac{-t}{\tau_i}} \tag{3.33}$$

where, $b_i = \sum_{j \neq 0} a_i a_j q_i q_j$

This is the generalization of $\chi(t) \sim e^{\frac{-t}{\tau}}$ for all times, not only for long times. Hence, there are as many correlation times as the eigenvalues, or eigenstates.

//// oku bu kısmı.

Hence, to find correlation time τ for any temperature, autocorrelation function of a quantity may be examined after the equilibrium is achieved. Hence, calculating auto-correlation function for all quantities that we are interested and using maximum of the correlation times is a good choice for us.

Now, we have independent data for quantities after reaching the equilibrium. Hence, with using these, we can examine fluctations in these quantities to analyze other macroscopic properties of the system. However, previously, we should determine our

error ranges in these quantities.

3.3.3 Errors in Calculation

To make a complete analysis, we have to determine tolerances in our data. Errors on Monte Carlo results can be divided into 2 classes: Statistical errors and Systematic errors. Statistical errors are result of randomness in Monte Carlo simulations and these errors can be decreased taking a lot of samples. Systematic errors are caused by the alghorithm that is used to make measurements.

Statistical Errors

In a Monte Carlo calculation, value of a quantity changes from step to step, as a result of randomness. It is often straightforward to estimate the statistical error in a measured quantity, since the assumption that the error is statistical implies that we can estimate the true value by taking the mean of several different measurements, and that the error on that estimate is simply the error on the mean. [**?**]

The basic error analysis of a statistical measurement is direct calculation of standart deviation of measurement:

For example, take n measurements of a quantity q_i. Mean of that quantity is $\langle q \rangle$. Then, standart deviation is

$$\sigma = \sqrt{\frac{\frac{1}{n}\sum_{i=0}^{n}(q_i - \langle q \rangle)^2}{n-1}} = \sqrt{\frac{1}{n-1}(\langle q^2 \rangle - \langle q \rangle^2)} \tag{3.34}$$

// CLT ile acıkla 1/n kısmını

It is known that $n = \frac{t_{max}}{2\tau} >> 1$ then,

$$\sigma = \sqrt{\frac{2\tau}{t_{max}}(\langle q^2 \rangle - \langle q \rangle^2)} \tag{3.35}$$

Also, there are 3 error analysis methods for an statistical measurement:

1. The Blocking Method

   In the Blocking Method, data is divided in to several blocks with several samples

and for each block, mean of that quantity is calculated. Then, putting them into the

$$\sigma = \sqrt{\frac{\frac{1}{n}\sum_{i=0}^{n}(q_i - \langle q \rangle)^2}{n-1}} = \sqrt{\frac{1}{n-1}(\langle q^2 \rangle - \langle q \rangle^2)} \tag{3.36}$$

where, this time n is block number and q is the mean of the quantity.

2. The Bootstrap Method

   When calculating a quantity of the system, independent samples are chosen in datas of that quantity. In the Bootstrap method, we resample these quantity datas, again we choose n samples from that data values but this time, these n samples are chosen randomly; also, these samples can be same, and using these n samples we calculate that quantity. In this case, n can be any value, for example, number of independent samples in datas of that quantity. This procedure is repeated for i times and for each time we calculate the quantity q_i. Then, error in that quantity is the standart deviation in the values q_i, which is

$$\sigma = \sqrt{\langle q^2 \rangle - \langle q \rangle^2} \tag{3.37}$$

3. The Jackknife Method

   When calculating a quantity, n independent samples are used. In the Jackknife method, that quantity is calculated again with using n-1 samples by removing first, second, ... ith independent sample, respectively, if quantities are labeled as q_i with ith sample removed, then

$$\sigma = \sqrt{\sum_{i=1}^{n}(q_i - q)^2} \tag{3.38}$$

   where, q is the quantity calculated with n samples.

Systematic Errors

For systematic errors, there is no general method of estimate. There are 2 obvious systematic errors in Metropolis simulation of the Ising Model. Firstly, to reach the equilibrium state, we wait a finite time. Actually, we get Maxwell-Boltzmann distribution when

t goes to infinity. The other error is that not running the simulation long enough after equilibrium to take independent samples.

3.4 Metropolis Algorithm

To derive Metropolis algorithm, firstly, we choose $g(\mu \to \nu)$ and in oder to satisfy detailed balance condition we choose appropriate $A(\mu \to \nu)$ and the algorithm chooses a new state ν and then accepts or rejects it with respect to the acceptance ratio, and if it accepts, we move to the new state, else we stay at the current state and the algorithm repeats itself. ~~continually.~~ $g(\mu \to \nu)$ should be arranged to satisfy Ergodicity condition.

Mean of energy is U. $E[E] = \mu_E = \langle E \rangle = U$ Then, variance in energy is $\sigma^2 = E[(E - \mu_E)^2] = E[E^2] - {\mu_E}^2$. We have shown that

$$U = E[E] = -\frac{\partial \log Z}{\partial \beta}$$

and

$$E[E^2] = \frac{1}{Z}\frac{\partial^2 Z}{\partial \beta^2}$$

Hence,

$$Var[E] = E[E^2] - (E[E])^2 = \frac{\partial^2 \log Z}{\partial \beta^2} \tag{3.39}$$

Then, standart deviation σ is

$$\sigma = \sqrt{\frac{\partial^2 \log Z}{\partial \beta^2}} \tag{3.40}$$

The variation of the actual value of U around the expectation value μ_E is tiny by comparison with the kind of energies we are considering for the whole system, and probably not within the resolution of our measuring equipment.[**?**]

So, energy fluctations are the very small portion of the total energy. This means that, system mostly in states with a narrow range of energy.Thus, in the simulation, we want to be in that states, mostly. An algorithm which has *Single-spin-flip dynamics* is an algorithm for this case.

Single-spin-flip dynamics state that, maximum energy difference between new state ν and current state μ is 2J for each bond between spin we flip and its neighbours. For example, in 2D structre a site has 4 neighbours then maximum difference between energies

of 2 states is 8J. Therefore, in general, maximum energy difference between states is $2zJ$ where z is the lattice coordination number, which means that number of neighbours of a site. Also, single-spin-flip dynamics guarantees the ergodicity condition. For any state, there is always a chance to go to that state from current state.

Let's look at the Metropolis Algortihm. According to the Metropolis Algorithm, $g(\mu \rightarrow \nu)$ is same for all possible states ν. For instance, for a system with N states, that we can access from current state μ, $g(\mu \rightarrow \nu) = \frac{1}{N}$ In this case, to satisfy detailed balance condition:

$$\frac{A(\mu \rightarrow \nu)}{A(\nu \rightarrow \mu)} = e^{-\beta(E_\nu - E_\mu)} \tag{3.41}$$

There is no condition for the individual acceptance ratios. Therefore, for example, we can choose $A(\mu \rightarrow \nu) = A_0 e^{-\frac{1}{2}\beta(E_\nu - E_\mu)}$ and we can set A_0 to be any value with considering acceptance ratio is a probabilty value. However, this choice of acceptance ratio is inefficient. Because, we have low probabilities to accept a move and in this case we generally do not move to another site.

To maximize acceptance ratio, as earlier mentioned, we set larger of two acceptance ratios to 1, by changing A_0 and adjust the other to satisfy detailed balance condition.

For insance, let $E_\nu > E_\mu$, in this case $A(\mu \rightarrow \nu) = 1$ and $A(\nu \rightarrow \mu) = e^{-\beta(E_\nu - E_\mu)}$.

Hence, we will choose acceptance ratios as

$$A(\mu \rightarrow \nu) = \begin{cases} e^{-\beta(E_\nu - E_\mu)}, & \text{if } E_\nu > E_\mu \\ 1, & \text{otherwise} \end{cases}$$

Thus, simulation always accepts to move to a new state with lower energy.

By using this acceptance ratios, we will simulate Ising Model via Metropolis Algorithm with Single-spin-flip dynamics.

3.5 Implementation of Metropolis simulation of the Ising Model

In the simulation of the Ising model with Metropolis algorithm, instead of βJ we used $\hat{J}$ and we take J as 1. After that we let $J = \hat{J}$. Before simulation starts, we should define 3 properties in the program:

1. Initial Spins

Firstly, initial spins of the system should be defined. If T = 0 is chosen as the initial temperature of the system, then, all spins are up in the initial state. If T = ∞ is chosen as the initial temperature of the system, then, all spins are randomly distributed. Also, we can choose any spin configuration that help us to get equilibrium state, fastly.

2. Acceptance Probabilities

   For flipping a spin, we have to know the acceptance probabilities. According to the Metropolis Algorithm of the Ising Model, flipping a spin is accepted with a probabilty;

$$A(\mu \to \nu) = \begin{cases} e^{-J(E_\nu - E_\mu)}, & \text{if } E_\nu > E_\mu \\ 1, & \text{otherwise} \end{cases}$$

   Hence, we can calculate energy difference between 2 states:

$$E_\nu - E_\mu = -\sum_{<ij>} {s_i}^\nu {s_j}^\nu + \sum_{<ij>} {s_i}^\mu {s_j}^\mu = -\sum_{i.neig.k} {s_i}^\mu ({s_k}^\nu - {s_k}^\mu) \tag{3.42}$$

   Also, it is clear that ${s_k}^\nu - {s_k}^\mu = -2{s_k}^\mu$ since, our spin can be either 1 or -1. Then,

$$E_\nu - E_\mu = 2{s_k}^\mu \sum_{i.neig.k} {s_i}^\mu \tag{3.43}$$

   This energy difference is independent of new state ν. Therefore, it can be calculated at the beginning of the program.

   If a spin k in the lattice has K neighbours, then $\sum_{i.neig.k} {s_i}^\mu$ may be one of the following values :

$$-K, -K+2, -K+4, ..., 0, ..., K-4, K-2, K$$

   , there are K+1 possible values. However, if $E_\mu \geq E_\nu$, selected spin is always flipped. Hence, if sum is 0 or has opposite sign of ${s_k}^\mu$, we don't need to calculate it. Then, this acceptance probability is needed for only K/2 values.

   If 2D lattice is the case, as in our simulation, K = 4, then $\sum_{i.neig.k} {s_i}^\mu = 4, 2, 0, -2, -4$ and if ${s_k}^\mu$ is 1, then $0, -2, -4$ is not required according to acceptance probabilities

to flip a spin. Also, if ${s_k}^{\mu}$ is -1, then $0, 2, 4$ is not required according to acceptance probabilities to flip a spin. Hence, we need only the cases that ${s_k}^{\mu} \sum_{i.neig.k} {s_i}^{\mu} = 2, 4$. Then, at the beginning of the program, acceptance probabilities e^{-4J} and e^{-8J} should be calculated, and in the simulation it should be used to flip or not to flip a spin.

3. Neighbouring spins

   To get rid of loops to determine neighbours of a spin, all neighbours of any spin should be numbered and stored in a matrix.

After, defining initial properties we can start the simulation. Simulation goes like this:

1. Select a spin randomly.
2. Calculate energy difference, ΔE of states when this spin is flipped.
3. If $\Delta E \leq 0$, then flip the spin. If $\Delta E > 0$, then flip the spin according to acceptance probability.
4. Repeat the process

This is the Metropolis simulation of the Ising Model.

/// programa referans verilecek....

3.5.1 Calculation of Energy and Magnetization

Actually, the most important macroscopic properties of the system is energy and magnetization. Hence, energy and magnetization should be calculated throughout the simulation, in an efficient way.

Energy per site

Energy of a state can be calculated by using Hamiltonian and replacing the values of spins at that state, but this is not an efficient way to calculate energy. Since, at every state a sum must be calculated and values of all spins must be checked. By considering energy change between states, which is calculated to implement simulation of Ising Model, a more efficient way can be used. If energy of current state μ is calculated, then energy of next state, which is created by single spin flip, is

$$E_{\nu} = E_{\mu} + \Delta E \tag{3.44}$$

To implement it, energy of initial state can be calculated at the beginning of the simulation. Then, using ΔE's which is calculated to simulate the Ising Model, energy of next state can be calculated without considering values of all spins. Then, in each step by dividing them to the number of sites, we can get energy per site at each state.

Magnetization per site

Magnetization of a state is

$$M_\mu = \sum_i {s_i}^\mu \tag{3.45}$$

again magnetization can be calculated with replacing values of all spins and calculating the summation. But, again, this is inefficient way to do it. More efficiently, to calculate magnetization of a system, magnetization difference between states can be considered.

$$M_\nu = M_\mu + \Delta M \tag{3.46}$$

$$\Delta M = M_\nu - M_\mu = \sum_i {s_i}^\nu - \sum_i {s_i}^\mu = {s_k}^\nu - {s_k}^\mu \tag{3.47}$$

where, s_k is the spin that is flipped. If ${s_k}^\mu$ equals 1 then ${s_k}^\nu$ equals -1, so, difference is -2. Also, if ${s_k}^\mu$ equals -1 , then ${s_k}^\nu$ is 1, so, difference is 2. Hence, this can be generalized to by this way,

$$\Delta M = 2{s_k}^\nu \tag{3.48}$$

Again, by considering the magnetization difference, it can be calculated more efficiently. Magnetization of the initial state can be calculated at the beginning of the simulation, then by adding it to $2{s_k}^\nu$, magnetization of the new state can be calculated.

$$M_\nu = M_\mu + 2{s_k}^\nu \tag{3.49}$$

Then, in each step by dividing them to the number of sites, we can get magnetization per site at each state.

3.5.2 Equilibration Time

Now, we know that how to calculate energy and magnetization per site when using Metropolis algorithm. To ~~make~~ analyze of these quantities, we have to get equilibrium, ~~previously~~. Because, only when equilibrium is achieved, we have a Maxwell-Boltzmann distribution. As we previously defined, when equilibrium is reached, system spends most of the time in states which have small range of energies. Hence, to see the system to get equilibrium, we can observe energy or magnetization.

To decrease equilibration time, we can arrange initial spins. For example, if we want to analyze the Ising model at T = 2.0 ($\hat{T}$ = kT) (we wrote T=2.0 instead of $\hat{T}$=2.0), we can choose a spin configuration that has energy close to the energy when T=2.0, ~~and we get equilibrium earlier~~. Also, if we start with all spin up configuration, then we will wait a longer time to get equilibrium.

Let's look at magnetization to observe equilibration times for 2 initial spin configurations:

1. Initial spins are random, corresponds to T=∞

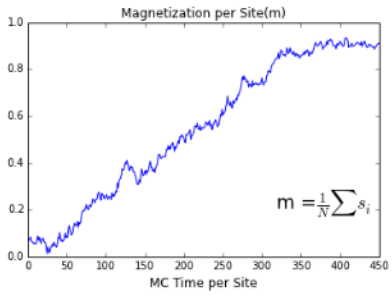


Figure 3.1: Magnetization per Site vs MC Time per Site when T = 2.0

2. Initial spins are all-up, corresponds to T=0

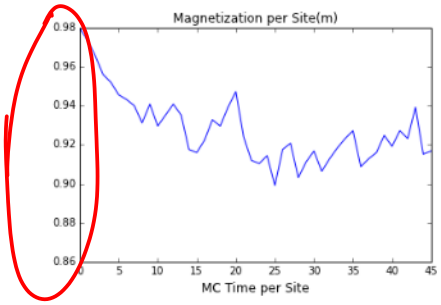


Figure 3.2: Magnetization per Site vs MC Time per Site when T = 2.0

Our system's temperature is 2.0. If we start simulation with all spins to be random, which corresponds to T=∞ case, then, we will get equilibrium about 350 MC Time per site. Also, if we start simulation with random spins, which corresponds to T=0 case, then, we will get equilibrium about 20 MC Time per site. Therefore, we can see that if our initial temperature, that we start the simulation, is close to our system's temperature we will get equilibrium earlier.

We want to analyze Ising Model for a lot of temperatures; hence, starting simulation with spin configuration at the end of the simulation for previous temperature, will decrease our equilibration time. If we are sure about that equilibrium is achieved, then we can analyze Ising model properties.

3.5.3 Autocorrelation Function and Correlation times

Now, we have to take independent samples from the properties after equilibrium is achieved. Thus, in the simulation, autocorrelation of quantities should be examined in the equilibrium state. With using fft() and ifft() functions in the program, autocorrelation function $\chi(t)$ after equilibrium is calculated. Autocorrelation function is proportional to $e^{\frac{-t}{\tau}}$ at early time steps, so, if $\chi(t)$ is normalized with $\chi(0)$, then $\frac{\chi(t)}{\chi(0)} = e^{\frac{-t}{\tau}}$. Now, with fitting this normalized autocorrelation function at early time steps with an exponential curve, autocorrelation time can be calculated. To fitting it, taking logarithm of both sides, then $log(\frac{\chi(t)}{\chi(0)}) = -t/\tau$ then $\tau = \frac{-1}{slope}$. Also, as mentioned before, we should take a sample in each 2τ.

In the simulation, we consider properties magnetization per site, energy per site and squares of them. Hence, we will evaluate autocorrelation for these 4 properties and we will find 4 correlation times and we will use maximum of correlation times.

Let's plot autocorrelation function with respect to MC time per Site:

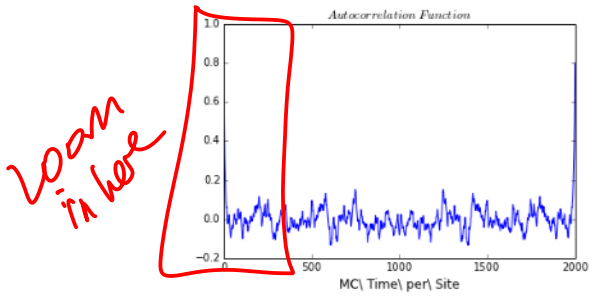


Figure 3.3: Autocorrelation function vs MC Time per Site when T = 2.0

// fft ve normal çizimdeki farkı acıkla. finite size durumu

// programda yapılan fft düzeltmesini acıkla.

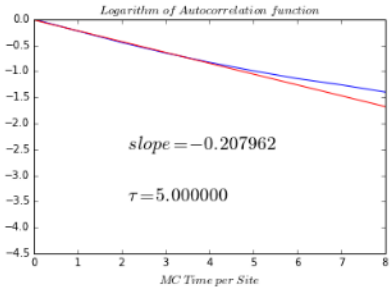


Figure 3.4: $log(\frac{\chi(t)}{\chi(0)})$ vs MC Time per Site when T = 2.0

At T=2.0, we have found correlation time $\tau = 5$ [time per site] by fitting normalizing autocorrelation function with an exponential. Hence, by taking a sample in each 2τ time per site, we have independent data for energy and magnetization and squares of them after reaching the equilibrium. Also, correlation time should be maximum of correlation times which are found by considering magnetization, energy and squares of them.

3.5.4 Analysis of Macroscopic System Properties

Now, we have independent samples for energy and magnetization and squares of them. As previously mentioned, with using these independent samples, we can calculate their mean values. Also, after finding mean values, we should calculate errors in them.

With using these mean values, we can calculate magnetic susceptibility and specific heat, which are fluctuations in magnetization and energy.

As we defined,

$$c = kJ^2N[\langle E_{ps}^2\rangle - \langle E_{ps}\rangle^2] \tag{3.50}$$

where E_{ps} is energy per site.

$$\chi = JN[\langle m^2\rangle - \langle m\rangle^2] \tag{3.51}$$

We embed k in T, so in our simulation k = 1 and $J = \frac{1}{T}$. Also, to get a complete analysis we should also add error bars to these properties. We can observe dependence of specific heat per site, magnetic susceptibility per site, internal energy per site and mean magnetization per site, correlation time, equilibraiton time and acceptance ratio to the temperature.

// Tc etrafinda hata cok cok fazla degil. Tc den uzakken 15-20 dk run. Tc etrafinda 2 gunluk run.

First, lets look at mean magnetization per site vs temperature plot:

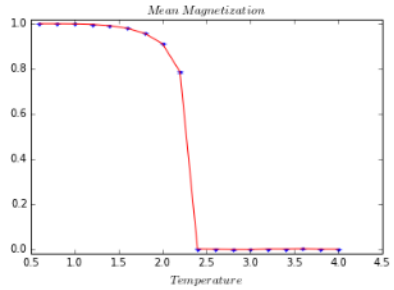


Figure 3.5: Mean magnetization vs Temperature

If we look at the plot, mean magnetization is 1 when T=0 and 0 when T=∞ as we expected. Also, if we increase the temperature from 0, we see that magnetization is about 1 until T is around 2.2. Also, if we keep on increasing the temperature, we will see a sharp decrease in magnetization to 0. This sharp change around when T is around 2.2 is called *Phase Transition*. There is a lot of properties that should be examined at the phase transition. These are studied in the next section.

We can also look to internal energy of system to see the phase transition:

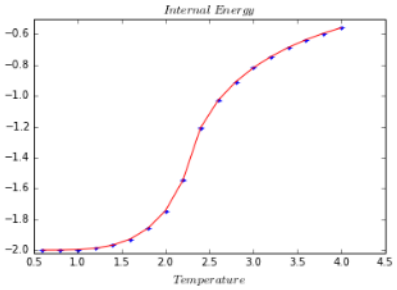


Figure 3.6: Internal energy vs Temperature

Again, we see that a change in internal energy of the system when T is around 2.2. This is also caused by phase transition.

Let's look at fluctuations in magnetization and energy, which are magnetic susceptibility and specific heat of the system.

Again, we can see that fluctuations in the magnetization is diverged at the temperature of phase transition. Hence, we see a peak at that temperature. Also, because of this increased fluctuations at the phase transition and increased correlation times at phase transition our error bars are increased. If we look at fluctuations in the energy;

Over again, fluctuations (hence, specific heat) is increased at the phase transition.

Now look at how correlation time acts at the phase transition.

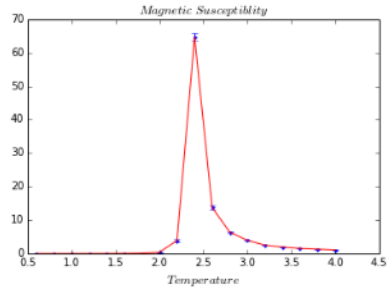


Figure 3.7: Magnetic susceptibility vs Temperature

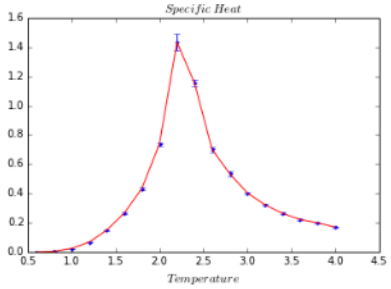


Figure 3.8: Specific heat vs Temperature

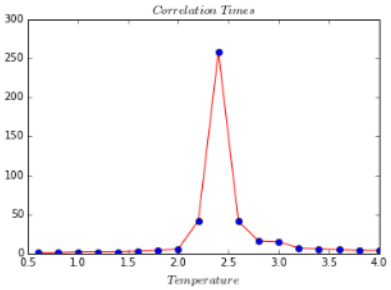


Figure 3.9: Correlation time vs Temperature

When phase transition occurs, our correlation times, states times between independent states, are diverged. Hence, because of that it is hard to simulate system around phase transition. Because, to get a lot of independent samples, we have to wait for a long time. Also, these increase in correlation time causes us to get a small number of samples; hence, causes increased error bars at the phase transition.

Also, we can look at how many attempts to flip a spin is accepted in the simulation, this is acceptance ratio.

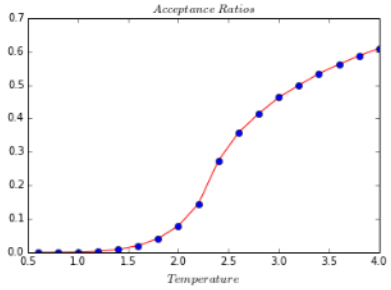


Figure 3.10: Acceptance ratio vs Temperature

We can see that our acceptance ratio is increasing with temperature. Our probability to flip a spin in Metropolis simulation of Ising model is depends on energy difference between states. If our new state's energy is lower than previous state's energy, we always accept flipping. However, if it is greater, we accept it with a probabilty proportional to energy difference between states. We know that at low temperatures spins are tend to same direction, hence, a states of low temperature energy of system is so low and if we flip a spin, we generally increase the energy of the system, because to flip a spin we have to break to bonds between same oriented spins. Hence, it is hard to flip a spin at low temperatures. If we look at high temperatures, spin are like randomly oriented, hence energy of the system is low. Thus, to flip a spin we have to broke less bonds, so, flipping a spin is more likely to occur.

// mean field çözümü.

// phase transition formulu ile phase transition için sıcaklık bulma

//error progpagation

3.6 Phase Transitions

We have seen that at phase transition, our correlation times diverge, there is a sharp change in magnetization and energy, hence there is a divergence in the magnetic susceptibility and specific heat at the phase transition. This phase transition is a property of Ising model. The change in phases of system is occurs at *Critical temperature*, T_c. For 2D Ising model,

$$T_c \approx 2.269J \tag{3.52}$$

in our case J is 1. Above critical temperature all spins are tend to random direction hence this phase is called *paramagnetic phase*, also below critical temperature all spins are tend to same direction thus this phase is called *ferromagnetic phase*.

Now, suppose we are at high temperatures where spins are random and uncorrelated. If we decrease temperature, interaction between spins forces spins to be in same direction. Hence, spins become correlated. Spin groups which are correlated because of these effect are called clusters and size of clusters are ξ, correlation length. This is because of the nature of the Ising model. Hence, at the phase transition, we have a lot of spin groups, seperately. If we continue to decrease temperature, spins chose a direction and generate non-zero magnetization. When T goes to 0 , the most of the spins are in same direction and absolute magnetization per spin goes to 1.

When we are in critical region, around T_c , weird events occur, these are called critical phenomena. We know that when temperature is high our spins are random when we are approach critical temperature from there, clusters occur, clusters includes spins that are correlated. Hence, when we flip a spin at critical temperature, these clusters flip, because spins inside it are correlated, hence, we see fluctuations in magnetization and energy, these fluctuations are called critical fluctuations. Thus, when we are approaching critical temperature from a high temperature, ξ increases and fluctuations increase, hence magnetic susceptibility and specific heat increases. One of the error sources in critical region is these fluctuations. Actually, in thermodynamic limit these fluctuations diverge, but in our case, where finite size exists, fluctuations become very large. Hence, errors increase. The other source of error is correlation time. At critical temperature, most of the spins are correlated to each other and with Metropolis algorithm we flip spins one by one. Hence, to get a state which is independent of current state, we have to wait a long correlation time. These long correlation time causes errors in calculation since we need a lot of independent measurements in simulation and when correlation times increase, we get lower measurements and because of that, error becomes higher. Actually, in thermodynamical limit correlation time diverges, but in our case, where finite size exists,

correlation time becomes very high. This increase in correlation time is called *Critical slowing down*. Hence, this error source depends on our algorithm whereas, fluctuation is a property of system. Hence, we can find a way to decrease correlation times to increase accuracy.

3.7 Critical Exponents

Now, we know that when we approach to phase transition correlation length increases. To measure our distance from phase transition, let us define reduced temperature, t.

$$t = \frac{T - T_c}{T_c} \tag{3.53}$$

For Ising model, divergence of correlation length near phase transition then goes like

$$\xi \propto |t|^{-\nu} \tag{3.54}$$

[**?**] ν is critical exponent and is a property of Ising model, independent of J or lattice size, algorithm, ... This is known as universality. Hence, ξ depends only on nature of Ising model.

As a result of divergence of correlation length, we get divergences in magnetic susceptibility and specific heat. To describe these divergences, we have 2 universal critical exponents,

$$\chi \propto |t|^{-\gamma} \tag{3.55}$$

$$c \propto |t|^{-\alpha} \tag{3.56}$$

Also, we can describe divergence in correlation time per site

$$\tau \propto |t|^{-z\nu} \tag{3.57}$$

where z is dynamic exponent.

Hence, with using dynamic exponent, we can analyze critical slowing down effect. Also, we know that critical slowing down depends on our algorithm. Hence, z depends on our algorithm, then, we want small z values in our algorithm. If z = 0, there is no critical

slowing down and we can simulate Ising model around phase transition, efficiently.

If we know critical temperature analitically, to measure critical exponents, we should look at finite size scaling.

3.8 Finite Size Scaling

Now, we know that

$$\tau \propto \xi^z \tag{3.58}$$

we know that ξ diverges near phase transition; hence, correlation times near phase transition are getting bigger.

In Monte Carlo simulations, we simulate models within a finite size. Because of finite size, correlation times actually can never really diverge. In a simulation, maximum value of correlation length is L^d, where d is dimension of our system. Therefore, when we are at critical temperature, our correlation length is approximatelly L^2 for an 2D Ising model.

$$\tau \propto L^z \tag{3.59}$$

At $T = T_c$, if we plot correlation times with respect to L on logarithmic scale, we can get z.

For Metropolis Algorithm:

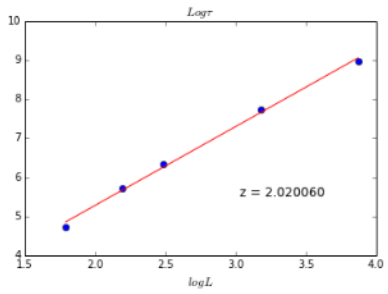


Figure 3.11: $log\tau$ vs $logL$

z is found as 2.02. This z is a high value for Monte Carlo simulations. Hence, it causes a great critical slowing down. Now, with using different algorithms we want to decrease dynamic exponent values, hence, want to decrease correlation times. If we look at Wolff algorithm, we get a lower value of z, hence, we expect that Wolff algorithm is better than Metropolis algorithm near phase transition. When, we are away from critical temperature

(T_c) Metropolis algorithm is a good choice to simulate Ising model. When we approach to T_c, correlation times are getting bigger and it makes hard to get sufficient independent values. Hence, because of this effect errors increase when we are approaching the phase transition.

3.9 Wolff Algorithm

Correlation times increase with L^z at critical temperature. CPU time to perform Monte Carlo steps per site increase with L^d because we have L^d sites. Hence, CPU time to simulate one correlation time for 2D Ising model is

$$\tau_{CPU} \propto L^{2+z} \tag{3.60}$$

Also, we found z is around 2, hence $\tau_{CPU} \propto L^4$. This is a big time for simulating the system around critical temperature.

The reason of that z is high values in Metropolis algorithm is because of divergence of correlation length ξ and critical fluctations near phase transition. When T is closer to critical temperature, correlation length is getting bigger and large regions of spins are in same direction. This regions are called domains. When $T = T_c$, $J = 0.44$; hence, $T_c = 2,269J$, $A(\mu \to \nu) \approx e^{-8J/T_c} \approx 0.03$ This is about 3% to flip a spin inside a domain. Therefore, for Metropolis algorithm it is difficult to flip a spin because it tries to flip spin by spin. As a result of this, acceptance ratio is very low around critical temperature and generally, we don't change state. Because of this, simulation time diverges around critical temperature.

Also, chance of flip a spin at the edge of a domain is high because it has opposite spins in its neighbourhood, hence lower energy cost. The basic idea of Wolff algorithm is to look for clusters of similarly oriented spins and then flip them in their entirely all in one go, rather than trying to turn them over spin by painful spin.[**?**] These algorithms that flips clusters are called Cluster-Flipping algorithms or simply Cluster algorithms. With using Cluster algorithms we can get rid of critical slowing down.

To find clusters, a spin is picked at random and we look at the neighbourhood of this spin to find a spin in same direction. If we find a spin in same direction, then we use the same procedure for this spin. By iterating in this way, we can form cluster. We don't want to flip all neighbouring spins in same direction, hence, this flipping a spin depends on temperature. If T is high, spins are generally uncorrelated and we want to flip little clusters. When T is around critical temperature, cluster size increases. When T is lower

than critical temperature, ferromagnetism come in action and most of the spins in same direction, like one cluster. Hence, we understand that if T decreases, cluster size to flip increases. Because of this, we add a spin to cluster to flip it with a probability P_{add} and we expect that P_{add} is negatively proportional to the temperature.

After sweeping the lattice and adding spins to cluster with a probability, we flip the cluster with an acceptance ratio which depends on energy cost to flip cluster.

With choosing these probabilities, we want to satisfy detailed balance for a given P_{add} and we want acceptance ratio as close to 1 by choosing P_{add}.

Suppose that we have chosen a cluster in state μ and with flipping these spins in cluster, we went to the state ν. If there is a spin neighbouring to the cluster, to flip the cluster we have to break these bonds. Let's create the cluster with choosing a random spin and adding spins to cluster in the same direction with P_{add}. Also, suppose we have rejected m spins to add to cluster with $1 - P_{add}$; hence, there are m bonds to broke. Hence, selection probability $g(\mu \to \nu) = 1 - P_{add}^{m}$. Also, suppose that for the same cluster there are n bounds to broke for a reverse move, $\nu \to \mu$. Thus, $g(\nu \to \mu) = 1 - P_{add}^{n}$ To satisfy detailed balance,

$$\frac{P(\mu \to \nu)}{P(\nu \to \mu)} = \frac{g(\mu \to \nu)A(\mu \to \nu)}{g(\nu \to \mu)A(\nu \to \mu)} = \frac{P_\nu}{P_\mu} = e^{-\beta(E_\nu - E_\mu)} = (1 - P_{add})^{m-n}\frac{A(\mu \to \nu)}{A(\nu \to \mu)} \tag{3.61}$$

E_ν and E_μ depends on bonds will broken. To break m bonds we need energy $m2J$ and to make n bonds we need energy $-n2J$.

$$E_\nu - E_\mu = 2J(m - n) \tag{3.62}$$

Hence,

$$\frac{A(\mu \to \nu)}{A(\nu \to \mu)} = ((1 - P_{add})e^{\beta 2J})^{n-m} \tag{3.63}$$

If we choose $P_{add} = 1 - e^{-2\beta J}$, then acceptance ratios are equal and independent of anything. With choosing acceptance ratios as 1 and $P_{add} = 1 - e^{-2\beta J}$, we will satisfy detailed balance condition and accept to flip the cluster always.

Also, ergodicity conditions is satisfied. We have always a chance to accept a neighboring spin to the cluster and flipping the cluster we get a new state. Hence, we have always a probability to access a state.

P_{add} is inversely proportional to temperature. When T=0, P_{add} is 1 and when $T = \infty$, P_{add} is 0. This means that size of cluster is inversely proportional to temperature.

3.10 Implementation of Wolff simulation of the Ising Model

After, defining initial properties we can start the simulation. Simulation goes like this:

1. Select a spin randomly.
2. Look at the neighboring spins. If they are in same direction, add the cluster with probability P_{add}
3. For a spin that is added to the cluster, look at the neighboring spins of this spin and again add the spins with same direction which are not in the cluster with probability P_{add}
4. Repeat the process and complete the cluster
5. Flip the cluster

This is the Wolff simulation of the Ising Model.

/// programa referans verilecek....

Now, we expect that correlation times to decrease when Wolff algorithm is used. To see, it we can compare dynamic exponents of Wolff algorithm and Metropolis algorithm.

In Wolff algorithm we flip spins in a cluster. If there is n spins in a cluster, we flip n spins. Also, in Metropolis algorithm we flip 1 spin. Hence, to make a fair description of correlation time, this difference must be considered. We calculated correlation times for Metropolis algorithm in MC time per site. Thus, to achieve an independent state, we have to wait τ trials of flipping 1 spin, per spin, in scrictly speaking. Also, again if we calculate correlation times for Wolff algorithm as in Metropolis algorithm, again we have to wait $\frac{\tau}{\langle n \rangle / L^d}$ trials of flipping 1 spin, per spin. Hence, if we multiply which is found in Wolff algorithm with $\frac{\langle n \rangle}{L^d}$, we make a fair calculation of correlation time. Therefore,

$$\tau = \tau_{steps} \frac{\langle n \rangle}{L^d} \tag{3.64}$$

Where τ_{steps} is found in Wolff algorithm and $\langle n \rangle$ is mean value of cluster sizes.

Chapter 4

Conclusion and Future Plan

A simplified version of the DMRG method is capable to find low lying states of the single particle problem. We have applied DMRG on a tight binding chain problem which upon decreasing lattice spacing gives the continuum limit. DMRG results for the single particle with a harmonic potential or infinite-well potential are very precise. In this part of the project, we have done wave function transformation to speed up the diagonalization process. Secondly, we have carried out tDMRG based on Suzuki-Trotter approach on the harmonic oscillator problem and demonstrated the classical oscillator-like time evolution of a coherent state. The wave function transformation is crucial in tDMRG.

Calculation of expectation values of observables is also feasible via DMRG. We calculated the $\langle x \rangle$, $\langle x^2 \rangle$, $\langle x^4 \rangle$, $\langle p \rangle$ and $\langle p^2 \rangle$ for the harmonic oscillator during the sweeping process. Expectation values of the observables are obtained precisely by the converged state vectors of the DMRG.

The content of this project is the application of the DMRG on single particle problem, but the DMRG is actually developed to treat numerical solutions of many body problems. In the many body case, we are interested in a physical system [15] consisting of a few photon pulses and a two level atom interactions, where atom is coupled to a single mode wave guide. As a future work time dependence of photons in this problem could be studied by tDMRG.

We have prefered to write DMRG code in MATLAB. Another important work we want to do in the future is transforming the DMRG code to an object oriented language such as Python or C++. Advantage of these kind of programming languages could be doing a better organization on DMRG code.

Bibliography

[1] U. Schollwöck, "The density-matrix renormalization group," *Rev. Mod. Phys.*, vol. 77, pp. 259–315, Apr 2005.

[2] R. Noack and S. R. White, "The Density Matrix Renormalization Group," *Lecture Notes In Physics*, vol. 528, pp. 27–66, 1999.

[3] M. A. Martín-Delgado, G. Sierra, and R. M. Noack, "The density matrix renormalization group applied to single-particle quantum mechanics," *Journal of Physics A Mathematical General*, vol. 32, pp. 6079–6090, Aug. 1999.

[4] R. A. Bertlmann, "Theoretical physics t2 quantum mechanics, lecture notes."

[5] T. J. Osborne and M. A. Nielsen, "Entanglement, quantum phase transitions, and density matrix renormalization," *eprint arXiv:quant-ph/0109024*, Sept. 2001.

[6] G. Barcza, Ö. Legeza, K. H. Marti, and M. Reiher, "Quantum-information analysis of electronic states of different molecular structures," *Physical Review A*, vol. 83, p. 012508, Jan. 2011.

[7] G. Kin-Lic Chan, J. J. Dorando, D. Ghosh, J. Hachmann, E. Neuscamman, H. Wang, and T. Yanai, "An Introduction to the Density Matrix Renormalization Group Ansatz in Quantum Chemistry," *ArXiv e-prints*, Nov. 2007.

[8] S. Pittel and B. Thakur, "The Density Matrix Renormalization Group and the Shell Model," *ArXiv e-prints*, Apr. 2009.

[9] K. A. Hallberg, "New trends in density matrix renormalization," *Advances in Physics*, vol. 55, pp. 477–526, Oct. 2006.

[10] K. G. Wilson, "The renormalization group: Critical phenomena and the kondo problem," *Rev. Mod. Phys.*, vol. 47, pp. 773–840, Oct 1975.

[11] S. R. White, "The Density Matrix Algorithms for Quantum Renormalization Groups," *Physical Review B*, vol. 48, no. 14, pp. 10345–10356, 1993.

[12] S. R. Manmana, "Time evolution of one-dimensional Quantum Many Body Systems," in *AIP Conference Proceedings*, vol. 789, pp. 269–278, AIP, Feb. 2005.

[13] U. Schollwoeck and S. R. White, "Methods for Time Dependence in DMRG," May 2006.

[14] T. Kato, "On the Adiabatic Theorem of Quantum Mechanics," *Journal of the Physical Society of Japan*, vol. 5, p. 435, Nov. 1950.

[15] E. Rephaeli, J.-T. Shen, and S. Fan, "Full inversion of a two-level atom with a single-photon pulse in one-dimensional geometries," *Physical Review A*, vol. 82, pp. 1–4, Sept. 2010.

Appendix A

Programs

Here is the tDMRG program written in MATLAB for single particle problem. The program includes main script and subroutines which perform specific tasks. The global variables in the main and the sub-scripts are defined to allow data communication between them.

Here is the subroutine in which block operators are initialized and saved.