QUANTUM SIMULATION OF A TRANSVERSE-FIELD ISING MODEL

A Theses submitted to

African University of Science and Technology,

(AUST) Abuja, Nigeria.

in partial fulfilment for the award of

MASTER DEGREE IN THEORETICAL PHYSICS

By

Ayeni Babatunde

Supervised by:

Dr. Antonello Scardicchio

International Centre for Theoretical Physics(ICTP), Italy

December, 2010

Dedication

I dedicate this theses to GOD ALMIGHTY who gave me wisdom, power and courage to complete this work.

Secondly I dedicate this work to my caring, wonderful and loving parents - **Mr. Emmanuel Oluwadamisi Ayeni** and **Mrs.Olufunke Mary Ayeni** and also to my siblings; **David Adetura**, **Joshua Oluwatimilehin**, **Peace Oluwakonyinsola**, **Deborah Adunolaoluwa** for their love and support in every way.

Acknowledgement

First and foremost, I'm grateful to the Almighty God through his Son Jesus Christ who gave me wisdom, power and courage to complete my courses and this theses successfully. My admission into AUST was not without God, my ability to have completed the courses giving its rigour was through his strength, my visit to ICTP, Italy, where I did my theses with a great professor, was all by his grace. Thank you Jesus for the life you gave me. I love you.

Secondly, I want to appreciate my parents for their genuine support towards me during my program at AUST, Abuja. They have been there for me all my life. Life would have been useless for me if you were not there to support me in every way - through your instructions, your guidance, your prayers, your money, your hard work every day and all others which will make an endless list. I'm grateful dad, I'm grateful mum. I love you both with a genuine love that only comes from within a caring heart. Since you gave all away to make us have a life, my prayer for you is that, you will enjoy the fruit of your incessant labour till the end of life at a fruitful age in Jesus name. In addition, I want to thank my siblings who were there for me at all time. You are the greatest in the whole world. None like my brothers: Ade and Timi, none like my sisters: sis Funmi, Kanyin and Adun. My deep love and prayers goes to you all.

I would like to express my sincere gratitude to the Acting President of the African University

of Science and Technology, **Professor C.E. Chidume** for his transparent leadership and administration of the school and for supporting me in making my trip to ICTP possible.

I am also deeply grateful to **Doctor Boubou Cisse** for all his concerns for the Theoretical Physics stream. For all your advice, meetings, reply of mails, etc, I'm grateful sir. You were always accessible and ready to listen to any complain. Indeed you're an asset to the foundation of AUST. Thank you sir.

I wish to thank staffs and students at ICTP, Italy, most especially my supervisor prof. Antonello Scardicchio of the Condensed Matter Physics section. You are a brilliant scientist and a great teacher with high pedagogical skills. Also, my sincere gratitude goes to Dr. Shehu, Dr. Somesh with whom I shared office with, Mr. Philip, Duma, Sylvia - a very wonderful and brilliant lady who was once my colleague at AUST, Joddel, Yemi, the christian fellowship among so many others that I can't list, for their advice and help. Thank you all.

My next gratitude is to all the staffs and students at African University of Science and Technology(AUST), Abuja, and to all my friends. My sincere appreciation to Dr. Popov Todor for all his tutoring and advice and also my colleagues in Theoretical Physics: Salawu, Dandogbesi and Nsofini, my colleagues in other streams:Dammy, Sunday Aliu, Auphe Baba, Mustapha Kabiru, Aguye, Yahaya, Khunle, Ogbe, Gaba(my room mate), Sylvia, among so many others who have all been very nice and friendly to me. God bless you all. Also, I want to thank my friends who were not in AUST but still showed me their love and support: AJaz, Raphael, Sunday(atole), Lukman, Fasho, Margaret and all others. Thank you and God bless you in all your life endeavours.

Last but not the least, I wish to express my sincere appreciation to Mr. Sarki Ajuji of the Center for Energy Research and Training(CERT), ABU, who taught me and supervised my undergraduate project. You first introduce me to C++, Monte Carlo methods, Metropolis algorithm, philosophy, Dirac Delta distributions, and other challenging mathematical and computational problems, without which I won't have had the necessary skills to have used during my theses and even my program at AUST. You are indeed a brain. Thank you sir and God bless you.

Abstract

A popular model that have been used to study ferromagnetism is the Ising Model which is an arrangement of spins along a particular direction and with discrete values of ±1. 1-D Ising model doesn't show a phase transition to the paramagnetic phase as opposed to the 2-D Ising model which shows a transition at a critical temperature. In this work, I have used Monte Carlo simulation method to study the 1-D quantum Ising model in a transverse field at a finite temperature to obtain the critical field when a ferromagnetic material becomes paramagnetic.

Contents

Dedication **ii**

Acknowledgement **iii**

1 INTRODUCTION **1**

1.1 Magnetic Systems 1

1.2 A Brief History of Ising Model 2

1.3 Statement of the Problem 4

2 LITERATURE REVIEW **6**

2.1 Statistical Mechanics 6

2.2 Ising Models 8

2.2.1 1-D Ising Model 8

2.2.2 2-D Ising Model 11

3 MONTE CARLO METHODS **14**

3.1 Metropolis Algorithm 14

3.2 Simulation of Ising Models Using Metropolis Algorithm 16

3.2.1 1-D Model 16

3.2.2 2-D Ising Model 18

4 QUANTUM LATTICE MODEL **21**

4.1 Path Integral Formulation of Ising Model 21

4.2 Quantum Monte Carlo Simulation and Numerical Results 25

4.3 Conclusion 27

Bibliography **28**

A C++ Codes **29**

A.1 1-D Ising model 29

A.2 2-D Ising model 31

A.3 1-D Quantum Ising model in a transverse field 33

CHAPTER 1

INTRODUCTION

1.1 Magnetic Systems

Magnetism is a common phenomenon observed in some materials in nature. A magnetic material unlike an electric charge cannot exist as a monopole but as a dipole. The smallest unit of a magnet which is usually referred to as "magnons" exists as a dipole of north pole and south pole. Opposite poles of a magent attracts. A magnet is a material that produces a magenetic field.[1]

When magnetic systems enter into a magnetic field, they get magnetized, on withdrawing the materials from the field, some loses their magnetism which makes them to be referred to as **temporary magnet** and some retains their magnetism which made them to be called **permanent magnet**. Examples of permanent magnet are iron, nickel, cobalt, loadstone, etc. While temporary magnets are just any object attracted by a magnet, those materials will lose their magnetism once the permanent magnet is removed, although they may retain a very weak magnetic strength.

In understanding the origin of magnetism, one has to consider the atomic description of the material. Every matter is made up of atoms which are composed of nucleon(neutron+proton) and electrons on the shells around the atom. According to atomic theory, these electrons orbit round the shells of the atom. According to Lenz law, a moving electron create a magnetic field. As these electrons orbit around the nucleus, they also spin along their orbit which give the electrons a dipole moment. Therefore, generally there are some atoms that have a magnetic moment. We can consider a crystal that contains such atoms arranged in a regular pattern, such a crystal will become magnetic under suitable conditions of spins alignment and of external conditions such as temperature and external magnetic field.

Depending on the orientation of the spins of the electrons, the material can be categorized into different types among which are:

- Ferromagnets: These materials have most of its spins aligned and uniformly ordered. These materials are the only one that can retain magnetism and become permanent magnets.

- Paramagnets: When there is no uniform alignment of spins, the material can be referred to as paramagnetic. As such, they are weakly attracted to a magnet

- Diamagnets: These are considered as materials not possessing any form of magnetism. Every other substance like carbon, waste, plastic, etc are diamagnetic.

Different theoretical model have been used to describe the phenomenon of ferromagnetism, the simplest being the Ising model; “a model for ferromagnetism formulated as a problem by Wilhelm Lenz(1920) and gave it as a problem to his student Ernst Ising and was already solved by 1925 as his PhD theses work”[2].

1.2 A Brief History of Ising Model

There are different conditions that can affect the magnetism or magnetic strength of a material; i.e. there are “harsh” conditions that make a material loses it magnetism. Essentially, temperature and applied magnetic field affects the strength of a magnetic system. Increase in temperature within a material increases thermal fluctuations which dis-orient the spins from their aligned positions and make the material to lose its magnetism. Also, since the electrons of an atom respond to magnetic field, increase in the applied magnetic field strength decreases the fluctuations of the spins which tends to increase the magnetisation of the material. In an attempt to investigate how a ferromagnet respond to an external "controllable" constraints, the problem which led to the Ising model was formulated.

The 1-D Ising model considers N spins on a linear chain. These spins interacts with each other with an interaction energy denoted by J, which by convention can be (J>0) for ferromagnetism or (J<0) for antiferromagnetism. The spins, as was used by Ising were assigned discrete variables of +1 for spin-up and -1 for spin-down. A simple schematic diagram is shown below.

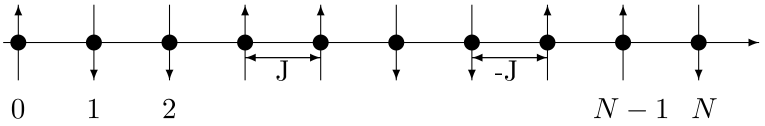

Figure 1.1: 1-D lattice of N spins with periodic boundary condition on the N^{th}spin. The interaction energy $\pm J$ between nearest neighbours pairs is shown for aligned and opposing spins.

The lines connecting neighbouring spins are referred to as links. When an external magnetic field is applied to a magnetic system, the applied magnetic field orient the spins along its direction which increases the internal magnetic strength within the system. Ising solved the 1-D model analytically and found no phase transition from the ferromagnetic phase to the paramagnetic phase but that the magnetisation just reduces to zero at a particular temperature and remains zero for all higher temperatures.

However, 2-D model which is the square lattice Ising model, a more complicated model, was solved by Lars Onsager in 1944 and there, a phase transition was observed which renders Ising's initial assertion of "no phase transition for all dimensions" false. A schematic diagram for the 2-D Ising model is shown below.

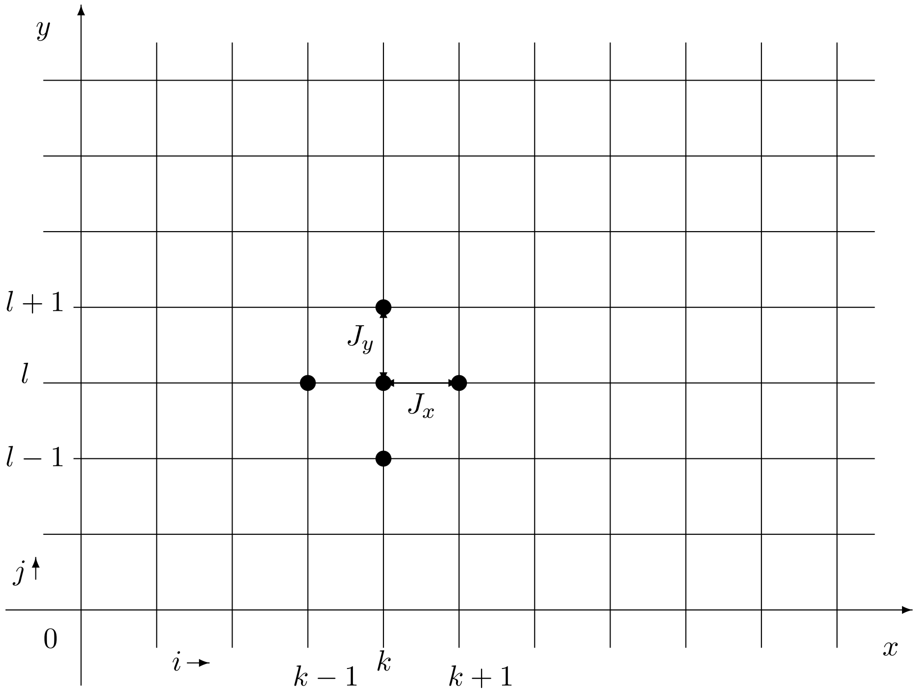

Figure 1.2: 2-D rectangular lattice arrangement of $m * n$ spins with periodic boundary conditions on the n^{th} and m^{th}spins with J_x as the interaction energy on the x axis and J_y as the interaction energy on the y axis

As opposed to the 1-D and 2-D Ising models, the 3-D Ising model hasn't received any exact solution even though some other approaches have been used to investigate phase transitions at higher dimensions. In an attempt to understand higher-D Ising model, approaches like mean-field theory, quantum field theory, computer simulations, etc, have been used.

1.3 Statement of the Problem

The Ising models outlined above are statistical models which can be referred to as classical statistical models because of the discrete nature of the spins. However, these models become quantum-mechanical when those spins are considered as spins of the famous Pauli spin matrices. The energies now are averages of the quantum Hamiltonian operators which describes the system.

In this work, we will solve the 1-D quantum Ising model in a transverse field which should show quantum phase transitions(i.e. quantum effects) when the temperature is kept very low and we vary the field to obtain the critical field. However, there are a number of challenges faced when one tries to simulate a quantum system as opposed to simulating a classical system. A successful approach to simulating quantum system is to find a path integral formulation which describes the problem in a classical way.

CHAPTER 2

LITERATURE REVIEW

2.1 Statistical Mechanics

Statistical physics is a mathematical formulation used in solving systems having many degrees of freedom. For example, like a box of gas containing so many particles of the order of $10^{24} \sim$ Avogadro's number. Basically a system consisting of a large number of particles can be describe with statistical mechanics, although this mathematical tool also work for systems of fewer number of particles. I will consider statistical mechanics as “building things bottom up” because it goes to the microscopic level of the system to compute the thermodynamic averages of the large system of particles. Thermodynamic values like internal energy U, specific heat capacity C, susceptibility χ, magnetisation M, etc can be derived from the framework of statistical mechanics.

Every thermodynamic system pass through phases which are different states of the system. In real experiments, these systems chooses their states accordingly as the system is driven towards equilibrium.

The essential assumption of statistical mechanics is that all accessible states are equally probable[4]. The probability of the system being in a particular state α_i is given by the

Boltzmann distribution2.1.1

$$P(\alpha_i) = \frac{exp(-E(\alpha_i)/k_BT)}{Z(T)} \tag{2.1.1}$$

where $\{\alpha_i\}$ will represent the different accessible states as the index $i = 1, 2, ..., k$ labels the acceptable configuration of the system, k_B is the Boltzmann constant, T is the temperature.

$$Z(T) = \sum_{\alpha_i} exp(-E(\alpha_i)/k_BT) = \sum_{\alpha_i} e^{-E_{\alpha_i}/k_BT} \tag{2.1.2}$$

where $Z(T)$ is a normalizing factor and is referred to as the partition function - this is a weighted sum over states.[5]

The partition function itself is like a core function of statistical physics yet it's not quite "interesting" since we only use it to obtain other statistical thermodynamic properties. Almost every other thermodynamic property can be obtained after deriving the partition function[6]

The expectation of a quantity Q for a system in equilibrium is

$$\langle Q\rangle = \sum_{\alpha_i} Q(\alpha_i)P(\alpha_i) = \frac{1}{Z}\sum_{\alpha_i} Q_{\alpha_i}e^{-\beta E_{\alpha_i}}$$

where $\beta = \frac{1}{k_BT}$ is called the inverse temperature or temperature parameter.

The expectation value of the energy $\langle E\rangle$ of a system is

$$U = \langle E\rangle = \frac{1}{Z}\sum_{\alpha_i} E_{\alpha_i}e^{-\beta E_{\alpha_i}} \tag{2.1.3}$$

From the above 2.1.3 it is obvious that the expectation energy can be written in terms of the derivative of the partition function as

$$U = -\frac{1}{Z}\frac{\partial Z}{\partial \beta} \tag{2.1.4}$$

the specific heat capacity is given as

$$C = \frac{\partial U}{\partial T} = \frac{\partial U}{\partial \beta}\frac{\partial \beta}{\partial T} = -k\beta^2\frac{\partial U}{\partial \beta} = k\beta^2\frac{\partial^2 \ln Z}{\partial \beta^2} \tag{2.1.5}$$

In deriving the entropy,

$$\begin{aligned} C &= \frac{\partial U}{\partial T} \qquad \text{but} \qquad \partial S = \frac{\partial Q}{T} \Rightarrow \partial Q = T\partial S \\ &= T\frac{\partial S}{\partial T} = -\beta\frac{\partial S}{\partial \beta} \end{aligned} \tag{2.1.6}$$

we can equate the two equations 2.1.5 and 2.1.6 for C and integrate with respect to β

$$k\beta^2 \frac{\partial^2 \ln Z}{\partial \beta^2} = -\beta \frac{\partial S}{\partial \beta} \qquad \Rightarrow \frac{\partial S}{\partial \beta} = k\beta \frac{\partial}{\partial \beta} \frac{\partial \ln Z}{\partial \beta}$$

$$S = -k \int \beta \frac{\partial}{\partial \beta} (\frac{\partial \log Z}{\partial \beta}) d\beta = -k(\beta \frac{\partial \log Z}{\partial \beta} - \int \frac{\partial \log Z}{\partial \beta} d\beta)$$

$$S = -k\beta \frac{\partial \log Z}{\partial \beta} + k \log Z \tag{2.1.7}$$

And now we can write an expression for the Helmholtz free energy F of the system

$$\begin{aligned} F = U - TS &= -\frac{\partial \log Z}{\partial \beta} + kT\beta \frac{\partial \log Z}{\partial \beta} - kT \log Z \\ &= -kT \log Z \end{aligned} \tag{2.1.8}$$

From all the above expressions, it is clearly shown how we can directly calculate all thermodynamic quantities from the partition function.

When considering a thermodynamic system, different thermodynamic properties of the system couples to different constraints or field. When a perturbation is introduced to the system, there is always an appropriate response which is considered as its conjugate variable. For example, increase of the thermodynamic temperature within a system increases the entropy within the system. Similarly, the magnetisation M of a magnetic material changes in response to an applied magnetic field B. The entropy S and magnetisation M are conjugate variables to the field temperature T and magnetic field B. From thermodynamics, values of the conjugate variables are derivatives of the free energy with respect to the constraints.[6]

Therefore the magnetisation is given as

$$M = \frac{\partial F}{\partial B} \tag{2.1.9}$$

2.2 Ising Models

2.2.1 1-D Ising Model

In the spirit of Ernst Ising, the spins arranged on a 1-D lattice are assumed to have only one of two directions, either parallel to the z-axis or anti-parallel to it. In order words, we can consider that the magnetic moment has only a z-component. If the magnetic moment is denoted by μ, the two discrete values of the magnetic moment are $m_z = \pm\mu$ which represent a spin up$(+\mu)$ and a spin down$(-\mu)$ respectively. If we consider an isotropic magnetic field applied along the z-axis, the Hamiltonian can be written as

$$H_{ham} = -J\sum_{i=1}^{N} s_i s_{i+1} - \mu B\sum_{i=1}^{N} s_i \tag{2.2.1}$$

where $-J\sum_{i=1}^{N} s_i s_{i+1}$ is the interacting energy between spins on neighbouring spins as shown in fig 1.

If we consider an arrangement of spins without interaction the energy of the whole system is given as

$$E = -MB$$

where M = total magnetisation and B = magnetic field

$$M = \sum_{i=1}^{N} m_z = \mu\sum_{i=1}^{N} s_i = \sum_{i=1}^{N} s_i \qquad\qquad (\text{if } \mu = 1) \tag{2.2.2}$$

the spins s_i are assigned values of $+1$ or -1. The Hamiltonian for the 1-D Ising model is written as

$$H_{ham} = -J\sum_{i=1}^{N} s_i s_{i+1} - B\sum_{i=1}^{N} s_i \tag{2.2.3}$$

recalling 2.1.2 the partition function can be written as

$$Z = \sum_{\mu} e^{-\beta E_\mu} \tag{2.2.4}$$

where μ denote the magnetic moment of our system. The energy of the system is given as

$$E_\mu = \langle H_{ham}(s_i)\rangle = -J\sum_{i=1}^{N} s_i s_{i+1} - B\sum_{i=1}^{N} s_i$$

therefore, the partition function can be written as

$$Z = \sum_{\mu} exp(\beta J\sum_{i=1}^{N} s_i s_{i+1} + \beta B\sum_{i=1}^{N} s_i)$$

$$Z = \sum_{s_1=\pm1} \dots \sum_{s_N=\pm1} exp(\beta J\sum_{i=1}^{N} s_i s_{i+1} + \beta B\sum_{i=1}^{N} s_i) \tag{2.2.5}$$

which is the sum over all the states of the system. Since each spin sum has two terms, the total number of terms represented by the spins sums is 2^N.

Without repeating what has been solved many years back, the magnetisation formula for the 1-D model which is a standard class-room work, is given as

$$M = \frac{Ne^K \sinh(b) + e^{2K}\cosh(b)\sinh(b)/\sqrt{e^{2K}\cosh^2(b) - 2\sinh(2K)}}{e^K\cosh(b) + \sqrt{e^{2K}\cosh^2(b) - 2\sinh(2K)}} \tag{2.2.6}$$

where $K = \beta J$ and βB.

In arriving at this equation, periodic boundary condition was taken into account i.e. the spin at the $(N+1)^{th}$ position in considered as equivalent to the spin at the 1^{st} position, i.e. $s_{N+1} = s$, which make the 1-D arrangement of spins as a “belt” of chains of spins.

The magnetisation per spin is given as

$$m = \frac{M}{N} = \frac{Ne^K \sinh(b) + e^{2K}\cosh(b)\sinh(b)/\sqrt{e^{2K}\cosh^2(b) - 2\sinh(2K)}}{e^K\cosh(b) + \sqrt{e^{2K}\cosh^2(b) - 2\sinh(2K)}} \tag{2.2.7}$$

$K = \beta J$ and $b = \beta B$

A stated earlier, the 1-D Ising model doesn’t show any phase transition. From the above formula, the mean magnetisation can be plotted against temperature. Using the values of $B = 0.15T$ which is a very weak field, and $T = 0.001$ to $10J$, the graph of m against T from the exact equation of magnetization is shown below.

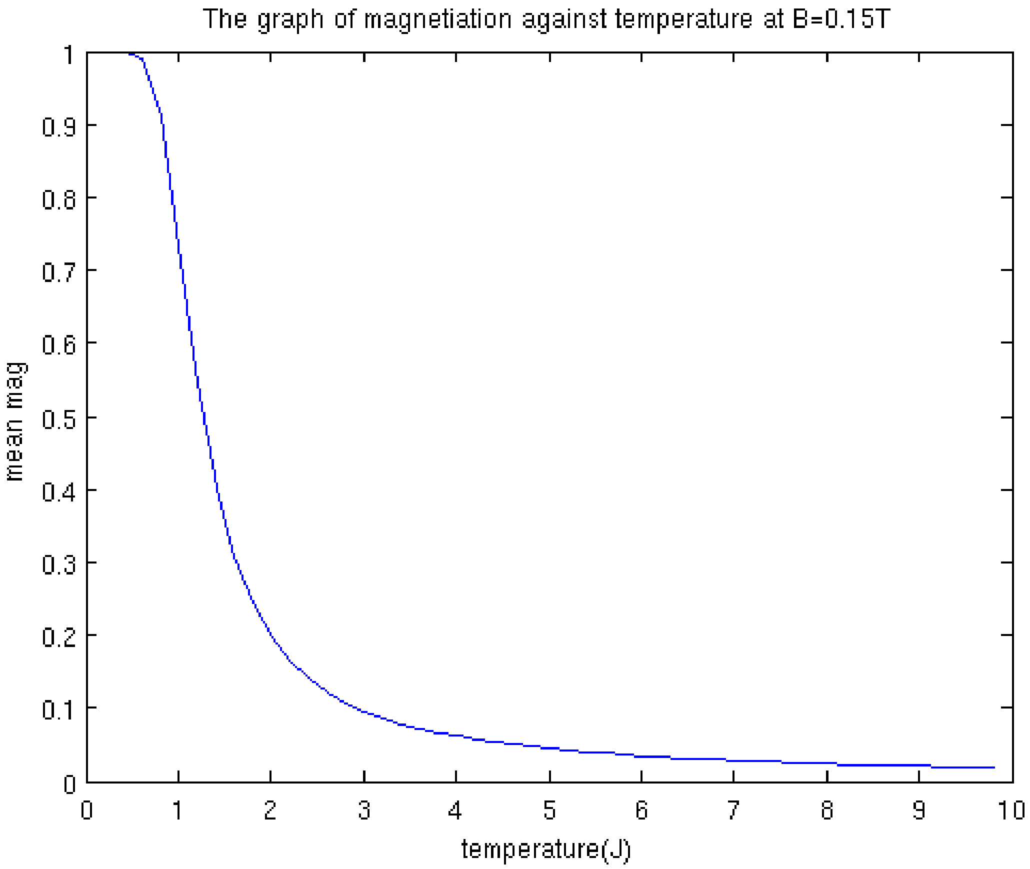

Figure 2.1: exact magnetization against temperature in 1-D Ising model

From the above diagram, it is observed that the magnetization decreases as temperature increases but no phase transition was obtained at any temperature which is true only for this 1-D model. It wil be shown later that the 2-D model shows a phase transition at a particular temperature known as a **critical temperature**

2.2.2 2-D Ising Model

This is a much more complicated model in terms of the dimension and even the analytic solution. The problem was first worked out exactly by Lar Onsager in 1944[3] for the case of zero magnetic field. Since this is a highly non-trivial problem as opposed to 1-D model, much is understood through computer simulations especially if an external magnetic field is applied or if one consider some other complex geometries other than cartesian.

As shown in 1.2, a generalized 2-D Ising lattice model would be to consider a rectangular lattice of spins with two different interaction energies along the two axis. The exact magnetization formula for the square lattice at zero field magnetic field as derived by Onsager is given below as

$$m = \begin{cases} 0 & T > T_c \\ \{1 - [\sinh(2\beta J)]^{-4}\}^{\frac{1}{8}} & T < T_c \qquad kT_c \approx 2.269185J \end{cases} \tag{2.2.8}$$

A graph of m against T in the 2-D Ising model as shown below shows a transition to a phase where $m = 0$ at all temperatures greater than T_c.

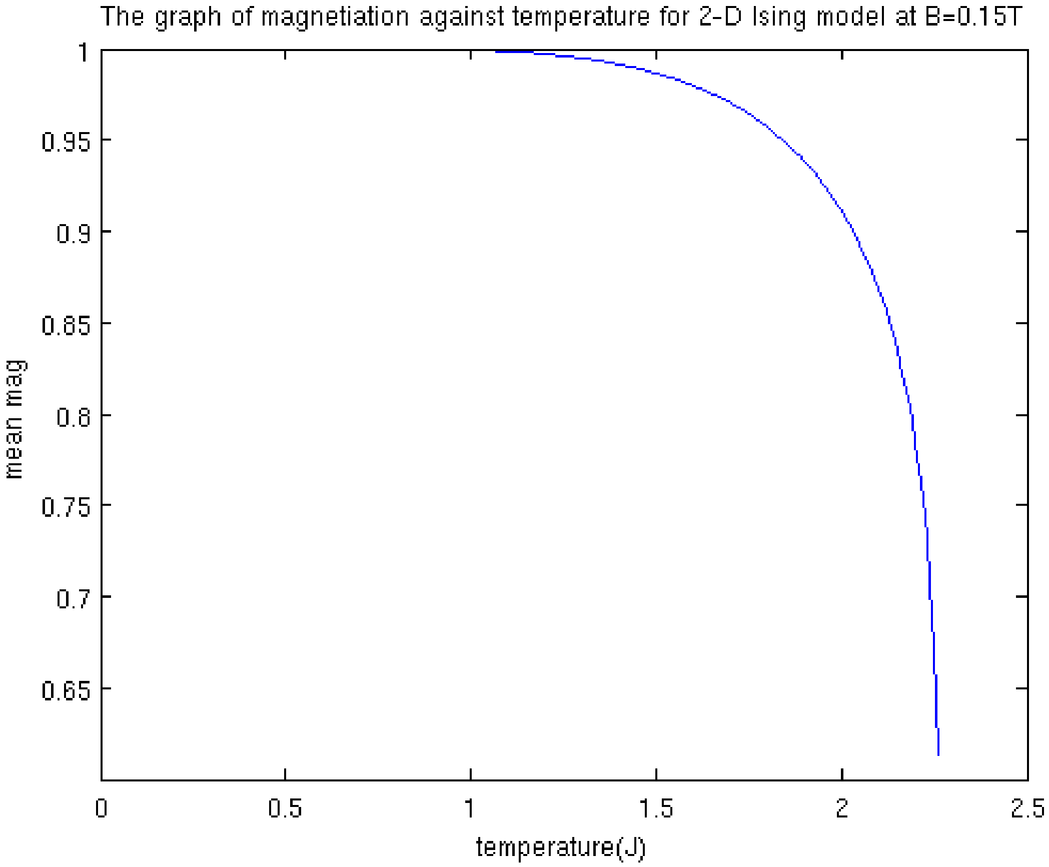

Figure 2.2: exact magnetization against temperature in 2-D square lattice

Below the critical temperature, the 2-D model shows spontaneous magnetization, but above the critical temperature the magnetization disappeared giving a zero magnetization and inherently means an observed phase transition in 2-D.

The above result is for a square lattice model with zero applied magnetic field. A generalized 2-D Ising model will consider a rectangular lattice arrangement with different interaction energies along the two axis. Computer simulations will prove useful here if one desire to understand the phase transition in 2-D Ising model.

The Hamiltonian for this model is given as

$$H_{ham} = -\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[J_y\sigma_{i,j}\sigma_{i,j+1} + J_x\sigma_{i,j}\sigma_{i+1,j}] - 2h\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\sigma_{i,j} \tag{2.2.9}$$

where J_x is the interaction energy along the x axis, J_y is the interaction energy along the y axis and h as the applied magnetic field.

CHAPTER 3

MONTE CARLO METHODS

3.1 Metropolis Algorithm

This algorithm survives inside simulation models such as Monte Carlo(MC) simulation. Monte Carlo simulation is essentially the only known numerical method for calculating the partition function of a model such as Ising model on a large lattice[6]. By the use of MC methods, we can simulate the random thermal fluctuations of a system over different acceptable and accessible configurations chosen according to Metropolis algorithm. Simulating a system on a computer would mean that, we allow our system to transit from one initial state to another acceptable state. Since in real experiment, the probability of the system being in any state μ is $P = \frac{1}{Z}e^{-\beta E_\mu}$, similarly, the acceptable states in our simulation will be given weights $w_\mu(t)$ according to Boltzmann probability. The difficult task is how to choose our states according to Boltzmann's probability.

Without delving much into the gory details of the theory of Monte Carlo simulations, our algorithm for choosing our states must satisfy some properties. Usually, for an effective MC simulation, we should have a set of states that forms a Markov chain. A Markov process guarantees that, given a system in one state μ, a new state ν can be generated and then feed that state into the process again to generate another state λ and so on. In addition, our Markov process must satisfy two other conditions, which are "ergodicity" and "detailed balance".

- Ergodicity: This condition ensures that our Markov process is able to reach every other state from a particular state.

- Detailed Balance: This is the condition that ensures that the Boltzmann probability distribution which we generate will bring our system to equilibrium.

In principle, the condition of detailed balance necessarily ensures equilibrium. If we denote the probability of the system being in a particular state μ as p_μ and the probability of the system making a transition from state μ to state ν as $P(\mu \to \nu)$ which is referred to as the "transition probability", therefore, the system will come to equilibrium after all the states have been visited which is represented below by the equation

$$\sum_\nu p_\mu P(\mu \to \nu) = \sum_\nu p_\nu P(\nu \to \mu) \tag{3.1.1}$$

with a condition $\sum_\nu P(\mu \to \nu) = 1$, being satisfied. Therefore,

$$p_\mu = \sum_\nu p_\nu P(\nu \to \mu)$$

If making a transition from state μ to state ν is possible, then from 3.1.1, we have

$$p_\mu P(\mu \to \nu) = p_\nu P(\nu \to \mu) \tag{3.1.2}$$

$$\frac{P(\mu \to \nu)}{P(\nu \to \mu)} = \frac{p_\nu}{p_\mu} = \frac{e^{-\beta E_\nu}}{e^{-\beta E_\mu}} = e^{-\beta(E_\nu - E_\mu)} \tag{3.1.3}$$

If we denote the probability of choosing a state as $C(\mu \to \nu)$ and the probability of accepting that state as $A(\mu \to \nu)$. Therefore, the probability of the system making a transition from one initial state to a new state is written as

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

from 3.1.3

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

If we assume that every state generated are equally probable to be selected, we have

$$\frac{A(\mu \to \nu)}{A(\nu \to \mu)} = e^{-\beta(E_\nu - E_\mu)} \tag{3.1.4}$$

To arrive at our Metropolis algorithm, we can assume that state ν has a higher energy than state μ i.e. $E_\nu > E_\mu$ so that transition from state ν to state μ is always possible with a probability of 1 i.e. $A(\nu \to \mu) = 1$, therefore

$$A(\mu \to \nu) = \begin{cases} e^{-\beta(E_\nu - E_\mu)} & \text{if } E_\nu - E_\mu > 0 \\ 1 & \text{otherwise} \end{cases} \tag{3.1.5}$$

The above equation is our Metropolis algorithm.

In practice, we always accept a new state ν if the energy of that state is lower than the energy of the initial state. Even if the new state has an energy greater than the initial state, we don't just reject that move out of hand, but accept it with a probability $e^{-\beta(E_\nu - E_\mu)}$ which should be greater than some random number which can be generated from the computer through the use of a good random number generator. That random number will be set during our simulation.

The metropolis algorithm, being one of the most famous algorithm used in MC simulation [6], is designed such that our system will come to equilibrium after a long run, thereby mimicking real experiments.

3.2 Simulation of Ising Models Using Metropolis Algorithm

3.2.1 1-D Model

The 1-D Ising model is a very simple model to solve both analytically and numerically. In order to justify any numerical scheme applied to solving statistical systems, the results of the numerical solution should match with known analytical solution to within some error limits. For the 1-D model whose exact results we know as shown in figure 2.1, I implemented the Metropolis algorithm inside a MC code to simulate an arrangement of 400 spins within the temperature range of $T = 0.01$ to $10J$ and at a magnetic field of $B = 0.15T$ to justify if the algorithm can "copy" the exact behaviour of the exact solution. The simulation was ran for 200 Monte Carlo cycle to thermalize the system and for 1000 Monte Carlo cycles to calculate averages of the magnetization. The figure below is a graph of magnetization against temperature of both the exact and simulation results.

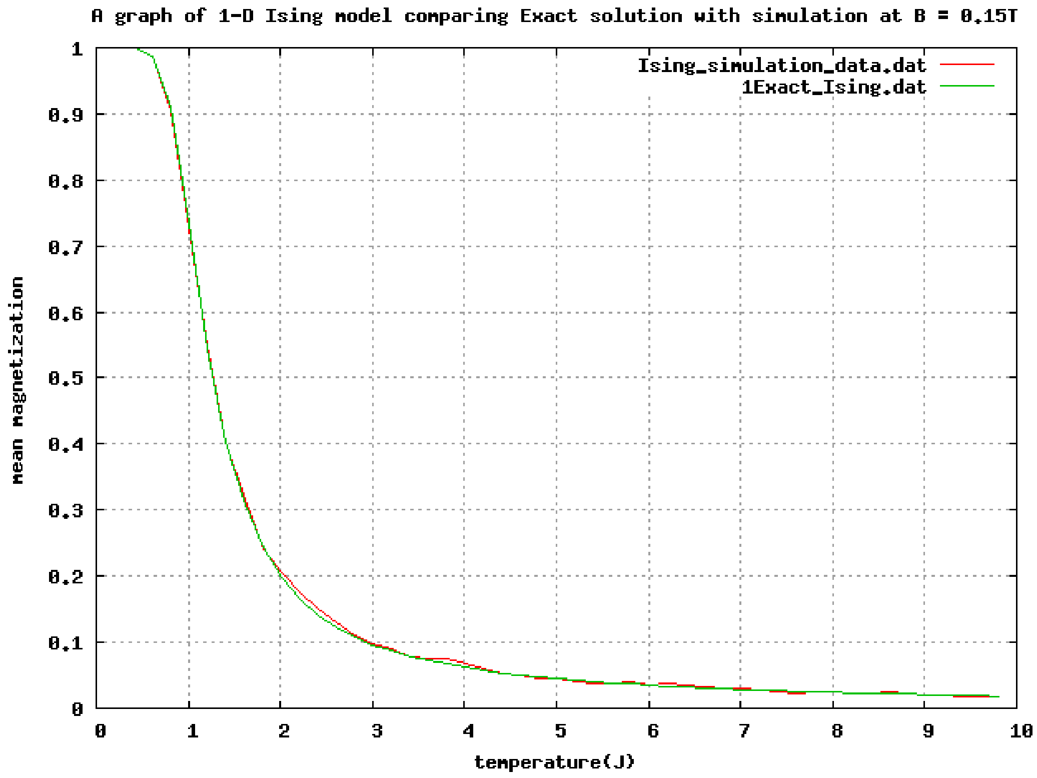

Figure 3.1: magnetization against temperature for both simulation and exact solution of 1-D Ising model

From the figure above, the result of our simulation almost perfectly match our exact result. We can therefore conclude that Monte Carlo simulation will be a good way to proceed if one desire to study systems that are intractable analytically.

The graph of the magnetization against temperature above doesn't show any phase transition which conforms to the 1-D exact solution. However, we should expect our Monte Carlo simulation of the 2-D model to undergo a phase transition from ferromagnetic to paramagnetic as in the exact solution.

3.2.2 2-D Ising Model

Since in our Metropolis algorithm, we have to always evaluate the difference in the energies of two states, it's not shrewd to always compute the energies directly as given in 2.2.9 and taking the difference each time we evaluate the metropolis algorithm. It is easy to see that each time a single spin is flipped, all other spins retain their values, therefore the difference of the sums of other spins interaction is zero and leaving only an effective difference based on the spins that flip and their nearest neighbours.

To derive an expression for the energy difference, assume we make a move from state μ to state ν, then the difference in their energies is given as,

For a particular state μ the energy is given as

$$E^{\mu} = -\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[J_y\sigma^{\mu}_{i,j}\sigma^{\mu}_{i,j+1} + J_x\sigma^{\mu}_{i,j}\sigma^{\mu}_{i+1,j}] - 2h\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\sigma^{\mu}_{i,j}$$

For a new state ν

$$E^{\nu} = -\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[J_y\sigma^{\nu}_{i,j}\sigma^{\nu}_{i,j+1} + J_x\sigma^{\nu}_{i,j}\sigma^{\nu}_{i+1,j}] - 2h\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\sigma^{\nu}_{i,j}$$

choose a random spin $\sigma_{k,l}$ from state μ for flipping i.e. when $i = k$, $j = l$. Only the four(4) nearest neighbours are affected

$$\text{the spin at position } (k,l) \text{ of the state } \nu \text{ is now } \sigma^{\nu}_{k,l} = -\sigma^{\mu}_{k,l} \tag{3.2.1}$$

let $\triangle E = E^{\nu} - E^{\mu}$, therefore

$$\triangle E = \sum_{<i,j>}[J_y\sigma^{\mu}_{i,j}\sigma^{\mu}_{i,j+1} + J_x\sigma^{\mu}_{i,j}\sigma^{\mu}_{i+1,j}] - \sum_{<i,j>}[J_y\sigma^{\nu}_{i,j}\sigma^{\nu}_{i,j+1} + J_x\sigma^{\nu}_{i,j}\sigma^{\nu}_{i+1,j}] + 2h(\sum_{<i,j>}\sigma^{\mu}_{i,j} - \sum_{<i,j>}\sigma^{\nu}_{i,j}) \tag{3.2.2}$$

in state μ , $i = 1, 2, \ldots, k-2, k-1, k, k+1, \ldots, m$ and in state ν, $j = 1, 2, \ldots, l-2, l-1, l, l+1, \ldots, n$. When we consider this in 3.2.2, up to $i = k-2$, and from $i = k+1$ upward nothing changes between interaction of spins $\sigma_{i's}$ and $\sigma_{j's}$ in the two configuration μ and ν and so their difference just simply cancels out. Up to $j = l-2$, and from $j = l+1$ upwards, spins interaction of the two different states μ and ν are the same and they cancels out in 3.2.2 leaving the expression shown below:

$$\triangle E = \sum_{i=k-1}^{k} \sum_{j=l-1}^{l} [J_y\sigma^{\mu}_{i,j+1}\sigma^{\mu}_{i,j+1} + J_x\sigma^{\mu}_{i,j}\sigma^{\mu}_{i+1,j} - J_y\sigma^{\nu}_{i,j+1}\sigma^{\nu}_{i,j+1} - J_x\sigma^{\nu}_{i,j}\sigma^{\nu}_{i+1,j}] + 2h(\sigma^{\mu}_{k,l} - \sigma^{\nu}_{k,l}) \tag{3.2.3}$$

when we now apply 3.2.1 in 3.2.3 we have the simple equation below:

$$\triangle E = 2\sigma^{\mu}_{k,l}[J_x(\sigma^{\mu}_{k-1,l} + \sigma^{\mu}_{k+1,l}) + J_y(\sigma^{\mu}_{k,l-1} + \sigma^{\mu}_{k,l+1}) + 2h] \tag{3.2.4}$$

Therefore in calculating the energy difference between states μ and ν, we simply apply the above relation to know if we should accept the move or reject the move. This relation is very elegant because we can determine if flipping a particular spin will lower the energy of our system before actually flipping that particular spin. Using 3.2.4 in our MC code will optimize our simulation for speed.

Simulating the 2-D Ising model at a field of $h = 0.001T$(i.e. a very weak field) and at $kT = 0.05$ to $2.2J$ (i.e. the critical temperature) with the c++ code that I wrote for the 2-D lattice , I obtained a “copy” of the analytical solution which shows a transition at about $T_c \approx 2J$ to a paramagnetic phase where the magnetization remained zero as shown in the graph below.

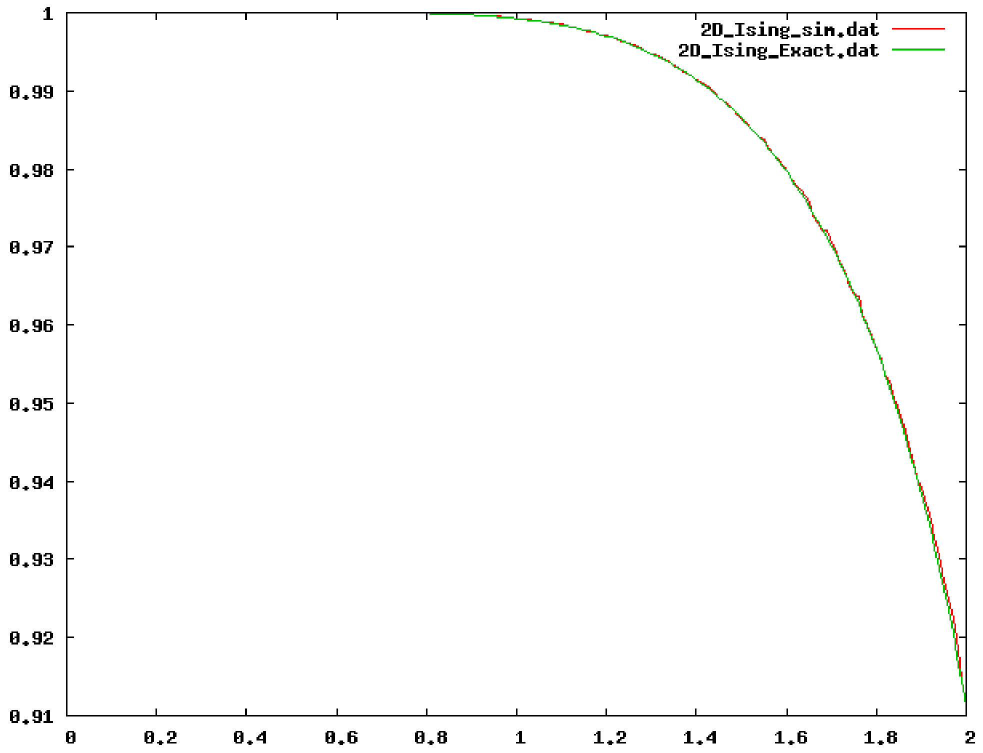

Figure 3.2: magnetization against temperature for both exact solution and simulation for 2-D square lattice

From the above graph of the magnetization against temperature, there is a phase change at a temperature near $T_c \approx 2J$ from ferromagnetic(ordered spins) to paramagnetic(scattered spins). At very high temperatures, thermal fluctuations now dominates over the interaction of the spins.

Conclusively, using the Metropolis algorithm, one can simulate just any system at suitable conditions to obtain thermodynamic variables which are invaluable yet unobtainable analytically and this has been an invaluable tool in simulating quantum systems.

CHAPTER 4

QUANTUM LATTICE MODEL

4.1 Path Integral Formulation of Ising Model

In this final chapter which is conveys the main work, I will discuss the 1-D Ising model in a transverse field and the numerical results obtained if one vary the applied magnetic field which is transverse to the orientation of the spins. The Hamiltonian for the 1-D Ising chain of spins in a transverse field is given as

$$\hat{H} = -J\sum_{i=1}^{N} \sigma_i^z \sigma_{i+1}^z - \mathit{\Gamma} \sum_{i=1}^{N} \sigma_i^x \tag{4.1.1}$$

where $\mathit{\Gamma}$ is the transverse field.

We can decompose the Hamiltonian above into two parts, i.e.

$$\hat{H} = \hat{H}_1 + \hat{H}_2$$

where $\hat{H}_1 = -J\sum_{i=1}^{N} \sigma_i^z \sigma_{i+1}^z$, $\hat{H}_2 = -\mathit{\Gamma}\sum_{i=1}^{N} \sigma_i^x$. The $\sigma_i^{(\gamma=x,y,z)}$ are the Pauli spin matrices at lattice i which are given as

$$\sigma^x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad , \quad \sigma^y = \begin{pmatrix} o & -i \\ i & 0 \end{pmatrix} \quad , \quad \sigma^z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{4.1.2}$$

Recalling the formaula from 2.2.4 , the partition function for this model can be written as,

$$\begin{aligned} Z &= \sum_{\{s_1, s_2, \ldots, s_N\}} exp(-\beta \hat{H}) \\ &= \sum_{s_{1,2,\ldots,N}} \langle s|exp(-\beta \hat{H})|s\rangle = Tr[exp(-\beta \hat{H})] \end{aligned} \tag{4.1.3}$$

where $|s\rangle = |s_1, s_2, \ldots, s_N\rangle$ is any particular state of N spins. The space of all the configurations is 2^N

The application of Metropolis algorithm to simulating statistical systems such as Ising models has proved useful as shown in the previous chapter. However the model that we have considered are referred to as classical Ising model since the spins are just discrete variables as opposed to this present Ising model of linear chains in a transverse field. The spins in this model are quantum mechanical operators which are the Pauli spins matrices. It has been pointed out that MC methods can not be directly applied to simulating Quantum Statistical systems [7] since the exponential of matrix operators are required and also because the presence of zero or negative matrix elements prevents us from directly applying our sampling algorithm. However, the quantum mechanical problem in d-dimensions can be transformed into an equivalent classical statistical problem in $d+1$-dimensions through the use of path integral[7]. In this work, I have used the discrete analogue of the Feynmann path integral for the numerical simulation of the quantum lattice model by applying the Trotter-Suzuki product formula[8]. The Suzuki-Trotter product is given as

$$exp(\hat{U} + \hat{V}) = \lim_{P \to \infty} [exp(\hat{U}/P) exp(\hat{V}/P)] \tag{4.1.4}$$

where P is the number of the different paths our system can evolve through. Using this in the partition function, we have,

$$\begin{aligned} Z &= \sum_s \langle s| \lim_{p \to \infty} [e^{-\hat{H}_1/k_B TP} e^{-\hat{H}_2/k_B TP}]^P |s\rangle \\ &= \lim_{p \to \infty} \sum_s \langle s|[e^{-\hat{H}_1/k_B TP} e^{-\hat{H}_2/k_B TP}]^P |s\rangle \\ &= \lim_{p \to \infty} Z_P \end{aligned} \tag{4.1.5}$$

$$\Rightarrow \qquad Z_P = \textstyle\sum_s \langle s|[e^{-\hat{H}_1/k_BTP} e^{-\hat{H}_2/k_BTP}]^P|s\rangle$$

where the Z_P above is the approximation for Z only if P is large enough.

If we introduce a complete set of states which can be chosen to be the eigen basis of σ^z into the above equation, we have

$$Z_P = \sum_s \langle s|(e^{-\hat{H}_1/k_BTP} e^{-\hat{H}_2/k_BTP})(e^{-\hat{H}_1/k_BTP} e^{-\hat{H}_2/k_BTP}) \ldots (e^{-\hat{H}_1/k_BTP} e^{-\hat{H}_2/k_BTP})|s\rangle$$

$$Z_P = \sum_s \langle s|(\Omega_1\Omega_2)(\Omega_1\Omega_2)\ldots(\Omega_1\Omega_2)|s\rangle \tag{4.1.6}$$

where Ω_1 and Ω_2 are $e^{-\hat{H}_1/k_BTP}$ and $e^{-\hat{H}_2/k_BTP}$ respectively.

$$\begin{aligned} Z_P &= \sum_s \langle s|(\Omega_1 \sum_k |s_k\rangle\langle s_k|\Omega_2)(\Omega_1 \sum_k |s_k\rangle\langle s_k|\Omega_2)\ldots(\Omega_1 \sum_k |s_k\rangle\langle s_k|\Omega_2)|s\rangle \\ &= \sum_s \langle s| \sum_k (\Omega_1|s_k\rangle\langle s_k|\Omega_2) \sum_k (\Omega_1|s_k\rangle\langle s_k|\Omega_2) \ldots \sum_k (\Omega_1|s_k\rangle\langle s_k|\Omega_2)|s\rangle \end{aligned}$$

where $\Omega_1|s_k\rangle$ gives

$$e^{-\hat{H}_1/k_BTP}|s^k\rangle = e^{\frac{J}{k_BTP}\sum_{i=1}^N \sigma_i^z\sigma_{i+1}^z}|s^k\rangle$$

but $\quad e^x = \sum_{k=0}^n \frac{x^k}{k!} = 1 + x + \frac{x^2}{2!} + \ldots + \frac{x^n}{n!}$

$$\Rightarrow \qquad \sigma_i^z\sigma_{i+1}^z|s^k\rangle = s_i^k s_{i+1}^k|s_{i+1}^k\rangle$$

therefore,

$$\begin{aligned} e^{-\hat{H}_1/k_BTP}|s^k\rangle &= e^{\frac{J}{k_BTP}\sum_{i=1}^N \sigma_i^z\sigma_{i+1}^z}|s_i^k\rangle = e^{\frac{J}{k_BTP}\sum_{i=1}^N s_i^k s_{i+1}^k}|s_{i+1}^k\rangle \\ &= \prod_{i=1}^N e^{\frac{J}{k_BTP}s_i^k s_{i+1}^k}|s_{i+1}^k\rangle \end{aligned}$$

$$\begin{aligned} Z_P &= \sum_s \langle s_i| \prod_{i=1}^N e^{\frac{J}{k_BTP}s_i^k s_{i+1}^k} \sum_k (|s_{i+1}^k\rangle\langle s_i^k|\Omega_2) \ldots \prod_{i=1}^N e^{\frac{J}{k_BTP}s_i^k s_{i+1}^k} \sum_k (|s_{i+1}^k\rangle\langle s_k|\Omega_2)|s_i\rangle \\ &= \sum_s \langle s_i^k| \sum_{k=1}^P \prod_{i=1}^N e^{\frac{J}{k_BTP}s_i^k s_{i+1}^k}\Omega_2 \ldots \sum_{k=1}^P \prod_{i=1}^N e^{\frac{J}{k_BTP}s_i^k s_{i+1}^k}\Omega_2)|s_i^{k+1}\rangle \\ &= \sum_{\{s_i^k\}} \prod_{k=1}^P \prod_{i=1}^N \exp(\frac{J}{k_BTP}s_i^k s_{i+1}^k)\langle s_i^k| \exp(\frac{\Gamma\sigma_i^x}{k_BTP})|s_i^{k+1}\rangle \end{aligned} \tag{4.1.7}$$

where periodic boundary condition have been taken along the Trotter direction, i.e. $s_i^{P+1} = s_i^1$. To evaluate $\langle s_i^k|\exp(\frac{\Gamma\sigma_i^x}{k_BTP})|s_i^{k+1}\rangle$, we will use the series expression of exponential function shown above,

$$\Rightarrow \qquad \sigma_i^x|s_i^{k+1}\rangle = |-s_i^{k+1}\rangle \qquad \text{and} \qquad \sigma_i^x(\sigma_i^x|s_i^{k+1}\rangle) = |s_i^{k+1}\rangle$$

$$\begin{aligned}\langle s_i^k|\exp(\frac{\Gamma\sigma_i^x}{k_BTP})|s_i^{k+1}\rangle &= \langle s_i^k|[\frac{\Gamma}{k_BTP} + \frac{1}{3!}(\frac{\Gamma}{k_BTP})^3 + \ldots)]|-s_i^{k+1}\rangle \\ &+\langle s_i^k|[1 + \frac{1}{2!}(\frac{\Gamma}{k_BTP})^2 + \frac{1}{4!}(\frac{\Gamma}{k_BTP})^4 + \ldots)]|s_i^{k+1}\rangle\end{aligned}$$

$$\begin{aligned}\langle s_i^k|\exp(\frac{\Gamma\sigma_i^x}{k_BTP})|s_i^{k+1}\rangle &= \sinh(\frac{\Gamma}{k_BTP})\delta_{s_i^k,-s_i^{k+1}} + \cosh(\frac{\Gamma}{k_BTP})\delta_{s_i^k,s_i^{k+1}} \\ &= \cosh(\frac{\Gamma}{k_BTP})s_i^k s_i^{k+1} \\ &= [\frac{1}{2}\sinh(\frac{2\Gamma}{k_BTP})]^{1/2}\exp(\frac{1}{2}\log\coth\frac{\Gamma}{k_BTP})s_i^k s_i^{k+1} \qquad (4.1.8)\end{aligned}$$

Therefore, using 4.1.8 in 4.1.7 and assuming that P is large such that Z_P approximates the partition function Z we have,

$$Z = C\sum_{\{s_i^k\}}\exp[\sum_{k=1}^{P}\sum_{i=1}^{N}(K_P s_i^k s_i^{k+1} + \frac{J}{k_BTP}s_i^k s_i^{k+1})]$$

where $\quad C = [\frac{1}{2}\sinh(\frac{2\Gamma}{k_BTP})]^{PN/2}$ and $K_P = \frac{1}{2}\log\{\coth(\frac{\Gamma}{k_BTP})\}$

$$\Rightarrow \qquad Z = \sum_{\{s_i^k\}}\exp[\sum_{k=1}^{P}\sum_{i=1}^{N}(K_P s_i^k s_i^{k+1} + \frac{J}{k_BTP}s_i^k s_i^{k+1}) + \ln C]$$

since the partition function is given as $Z = e^{-\beta H}$, therefore

$$H = -\frac{1}{\beta}\sum_{k=1}^{P}\sum_{i=1}^{N}(K_P s_i^k s_i^{k+1} + \frac{J}{k_BTP}s_i^k s_i^{k+1}) - \frac{1}{\beta}\ln C \qquad (4.1.9)$$

$$H = -\sum_{k=1}^{P}\sum_{i=1}^{N}(\frac{K_P}{\beta}s_i^k s_i^{k+1} + \frac{J}{P}s_i^k s_i^{k+1}) - \frac{1}{\beta}\ln C \qquad (4.1.10)$$

Comparing this Hamiltonian with that in equation 2.2.9, we can conclude that the Hamiltonian for the 1-D quantum problem has been successfully transformed into a 2-D classical

Hamiltonian which we can now easily simulate with the use of Metropolis algorithm run inside a Monte Carlo cycle. It now appears that the 1-D quantum Ising model resembles a 2-D rectangular lattice with two directions of spins arrangement each with its own interaction energy.

4.2 Quantum Monte Carlo Simulation and Numerical Results

In the Hamiltonian derived in the previous section, the K_P is referred to as the coupling constant in the "Trotter direction P" which depends on temperature T, the size of the P and on the applied magnetic field Γ. By the using the Metropolis algorithm3.1.5, we can now simulate the 1-D quantum Ising model to obtain thermodynamic quantities of interest such as free energy, magnetization, ground-state energy, etc. It is worth stressing the fact that, the Trotter index which is sometimes called the imaginary time axis has to be very large so as to approximate our partition function and to display the quantum effect which is expected.

With the use of the c++ programs that I wrote for the 2-D classical problem, I simulated the 1-D quantum problem at a finite temperature of $0.5J$ for various system sizes and at a varied magnetic field of 0.05 to $6T$. The graph of magnetization is shown below,

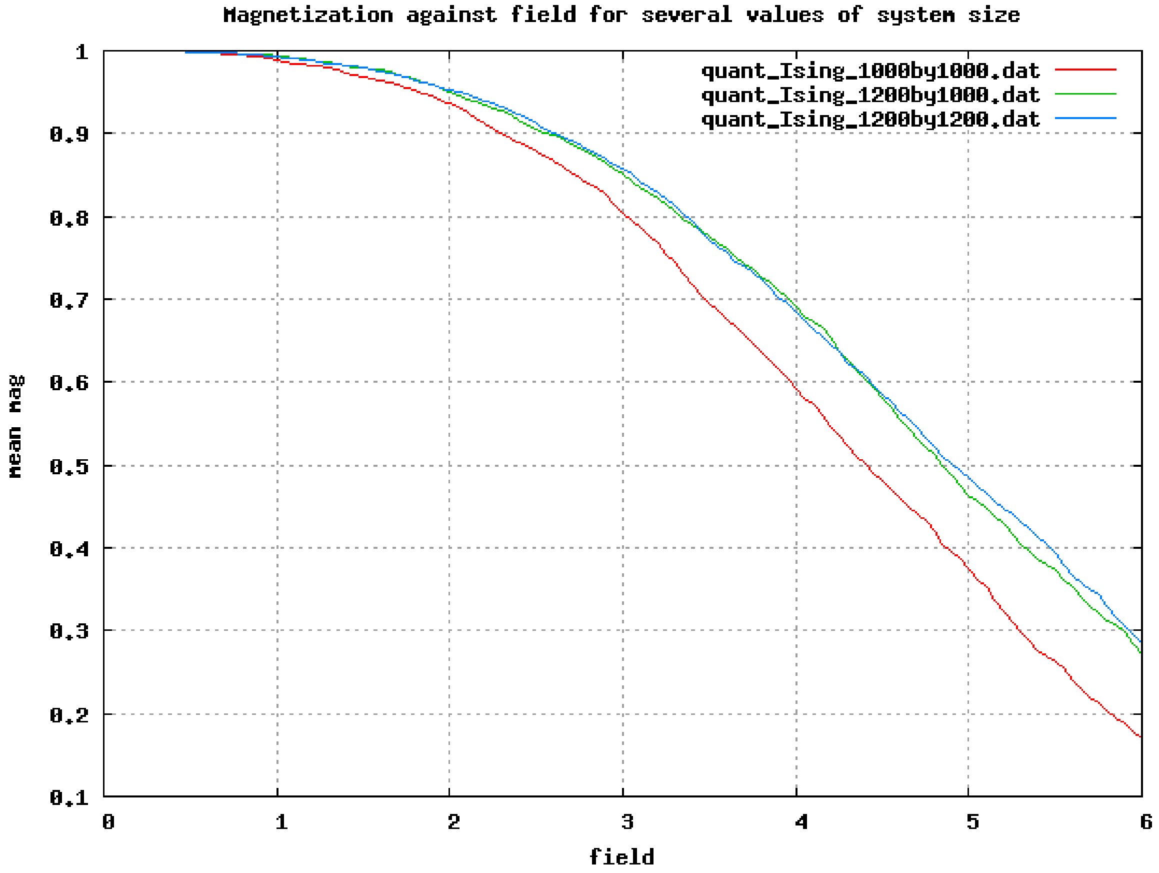

Figure 4.1: magnetization against magnetic field for different system size

From the figure above, we can conclude that this “resembles” the 2-D classical Ising graph of the magnetization against temperature which showed a phase transition at a critical temperature for a weak magnetic field. Although, as opposed to 2-D classical model, the 1-D quantum Ising model in a transverse field, show a phase transition at critical field of about $1T$ as shown in the above graph. The extra dimension of the 1-D quantum problem can be said to be due to quantum effects displayed by the spins.

4.3 Conclusion

In this work, I have used path-integral and Monte Carlo method to study the 1-D Ising model in a transverse field, showing that there is a critical magnetic field when the material will undergo a phase transition to lose all its magnetism. At a very small temperature, the material remains magnetic not until the applied magnetic field becomes very strong and the coupling of the spins becomes weak giving away its magnetism due to quantum effects.

Bibliography

[1] http://en.wikipedia.org/wiki/magnet.

[2] http://en.wikipedia.org/wiki/Ising_model.

[3] http://en.wikipedia.org/wiki/Square-lattice_Ising_model.

[4] Tony Guenault, *Statistical Physics.*

[5] Rubin H. Landau and Manuel J. Páez, *Computational Physics. Problem Solving with Computers.*

[6] M.E.J Newman and G.T Barkema, *Monte Carlo Methods in Statistical Physics.*

[7] Hans De Raedt and Ad Lagendijk, *Phys. Rep. 127 (1985), p.233.*

[8] K. Binder and D.W Heermann, *Monte Carlo Simulation in Statistical Physics.*

[9] Tao Pang, *An Introduction to Computational Physics*

[10] M.J. de Oliveira and J.R.N. Chiappin, *Physica A 238 (1997), p.312.*

APPENDIX A

C++ Codes

A.1 1-D Ising model

```
/*
    This program for 1-D Ising model simulates the Ising arragement of spins
    to compute average magnetisation

    Written by Ayeni Babatunde, Theoretical Physics stream
*/

#include<iostream>
#include<ctime>
#include<cstdlib>
#include<fstream>
#include<math.h>

using namespace std;

const int N = 64;  // no. of spins
int J = 1;   //interaction energy for a ferromagnet
double min_temp = 0.01, max_temp = 10.0, Tint = 0.2;     //for temperature
double temp = min_temp;
double B = 0.15;   // the magnetic field applied
int accept;   // no. of accepted configuration at this temperature

void initialize(int []);
bool metropolis(int []);
void oneMC(int []);
double mag_per_spin(int []);

int main()
{
    //open a file for writing output
    ofstream outfile("Ising_simulation_data.dat");
    outfile<<"temp"<<'\t'<<"mag_per_spin"<<endl;
    int MCsteps = 1000;    //no. of Monte Carlo step
    double mag_sum;    // variable for magnetisation
    //initializing my array of spins
    int s[N];
    initialize(s);

   while(temp < max_temp)          //start from the min temp
    {
        accept = 0;    // no. of accepted configuration at this temperature
        int seed = time(NULL);          //new seed
        srand(seed);
      int thermSteps = 0.2*MCsteps;     //steps for system to thermalize
        for(int i = 0; i < thermSteps; ++i)
            oneMC(s);

        //we now assume our system is in thermal equilibrium
        //we have to now calculate our mean magnetisation
        mag_sum = 0.0;
        for(int i = 0; i < MCsteps; ++i)
        {
            oneMC(s);    //geneartes a configuration around the equilibrium
            mag_sum += mag_per_spin(s);
        }

        //write data to file
        outfile<<temp<<'\t'<<mag_sum/double(MCsteps)<<endl;
        temp += Tint; //increase my temperature value
   }

    return 0;
}

void initialize(int s[])
{
    for (int i = 0; i < N; ++i)
        s[i] = 1;    //uniform start
}

bool metropolis(int s[])
{
    int k = rand()%N;    // choose site k for flipping
    double r = rand()/double(RAND_MAX);
    double dE;  // for my change in energy
    dE = 2*s[k]*(J*(s[(k==0)? N-1:k-1] + s[(k == N-1)? 0:k+1]) + B);
    if(exp(-dE/temp) > r)
    {
        s[k] *= -1;
        return true;
    }
    else return false;
}

void oneMC(int s[])
{
    for(int i = 0; i < N; ++i)
        if(metropolis(s)) ++accept;     //call my metropolis algorithm
}

double mag_per_spin(int s[])
{
    int sum = 0;
    for(int i = 0; i < N; ++i)
```

```
        sum += s[i];
    return sum/double(N);
}
```

A.2 2-D Ising model

```
 /*
    This program is for 2-D Ising model. It simulates the Ising
    arrangment of spins on a rectangle of m by n to computes avarage
    magnetisation and to check for a phase transition.

    Author: Ayeni Babatunde. Theoretical Physics stream(AUST)

*/

#include<iostream>
#include<ctime>
#include<cstdlib>
#include<fstream>
#include<math.h>

using namespace std;

const int m = 30;  // no. of spins on horizontal axis
const int n = 25;   //no. of spins on vertical axis
double Jx = 1;   //interaction energy on horizontal axis
double Jy = 1;   //interaction on vertical axis
double h = 0.0015;   // the magnetic field applied
double min_temp = 0.01, max_temp = 2.0, Tint = 0.01;     //for temperature
double temp = min_temp;
int accept;   // no. of accepted configuration at this temperature

void initialize(int [][n]);
bool metropolis(int [][n], double*);
void oneMC(int [][n], double*);
double mag_per_spin(int [][n]);
double energy(int [][n]);
int main()
{
    //open a file for writing output
    ofstream outfile("2D_Ising_sim_mag.dat");
    ofstream outfile1("2D_Ising_energy.dat");
    outfile<<"temp"<<'\t'<<"mag_per_spin"<<endl;
    int MCsteps = 10000;   //no. of Monte Carlo step
    double mag_sum, old_energy;   // variable for magnetisation and energy

    //initializing my array of spins
    int s[m][n];   //create a 2-dim array
    initialize(s);
    old_energy = energy(s);
    double* new_energy = &old_energy;

   while(temp < max_temp)           //start from the min temp
    {
        accept = 0;   // no. of accepted configuration at this temperature
        int seed = time(NULL);         //new seed
        srand(seed);
        //steps ssumed for our system to thermalize
        int thermSteps = 0.2*MCsteps;
        for(int i = 0; i < thermSteps; ++i)
            oneMC(s,new_energy);

        //we now assume our system is in thermal equilibrium
        //we have to now calculate our mean magnetisation
        mag_sum = 0.0;
        for(int i = 0; i < MCsteps; ++i)
        {
            oneMC(s, new_energy);    //genearte a new statearound the equilibrium
            mag_sum += mag_per_spin(s);
        }

        //write data to file
        outfile<<temp<<'\t'<<mag_sum/double(MCsteps)<<endl;
        outfile1<<temp<<'\t'<<*new_energy<<endl;
        temp += Tint; //increase my temperature value

    }

    return 0;
}

void initialize(int s[][n])
{
    for (int i = 0; i < m; ++i)
        for(int j = 0; j < n; ++j)
            s[i][j] = 1;    //uniform start
}


bool metropolis(int s[][n], double* new_energy)
{
    int k = rand()%m;    // choose site k on x axis for flipping
    int l = rand()%n;    //choose site l on y axis for flipping
    double r = rand()/double(RAND_MAX);
    double delta_E;
    //time to work-- just kidding!!\

    delta_E = 2*s[k][l]*(Jx*(s[(k==0)? m-1:k-1][l] + s[(k==m-1)? 0:k+1][l])
                      + Jy*(s[k][(l==0)? n-1:l-1] + s[k][(l==n-1)? 0: l+1]));
    if (exp(-delta_E/temp) > r)
    {
        *new_energy = delta_E + *new_energy;
        s[k][l] *= -1;
        return true;
    }

    else return false;

}

void oneMC(int s[][n], double* new_energy)
{
    for(int i = 0; i < m; ++i)
        for(int j = 0; j < n; ++j)
            //call my metropolis algorithm
            if(metropolis(s,new_energy)) ++accept;
}

double mag_per_spin(int s[][n])
```

```
{
    int sum = 0;
    for(int i = 0; i < m; ++i)
        for(int j = 0; j < n; ++j)
            sum += s[i][j];
    return sum/double(n*m);
}

double energy(int s[][n])
{
    double sum = 0;
    for(int i = 0; i < m-1; ++i)
        for(int j = 0; j < n-1; ++j)
            sum += Jx*s[i][j]*s[i+1][j] + Jy*s[i][j]*s[i][j+1] + 2*h*s[i][j];
    return -1*sum/double(n*m);    //return energy per spin
}
```

A.3 1-D Quantum Ising model in a transverse field

```
/*
    This program is for 1-D Quantum Ising model in a transverse field.
    It simulates the Ising arrangment of spins to determine
    the critical field.

    Author: Ayeni Babatunde, Theoretical Physics stream(AUST).
    E-mail:ayeni_babatunde2006@yahoo.com

*/

#include<iostream>
#include<ctime>
#include<cstdlib>
#include<fstream>
#include<math.h>

using namespace std;

const int p = 1200;  // no. of time-slices which is the Trotter number y-axis
const int n = 1200;   //no. of spins on the horizontal axis
double J=1;
double Jx = J/p;   //interaction energy on horizontal axis
double temp = 0.5;  // the temperature

//double Jy = 1;
double min_h = 0.05, max_h= 6, h_int = 0.05;      //for temperature
double h = min_h;
int accept;    // no. of accepted configuration at this temperature

void initialize(int [][n]);
bool metropolis(int [][n], double*, double);
void oneMC(int [][n], double*, double);
double mag_per_spin(int [][n]);
double energy(int [][n], double);
int main()
{
    //open a file for writing output
    ofstream outfile("2D_Ising_Trot_mag.dat");
    //ofstream outfile1("2D_Ising_Trot_energy.dat");
    outfile<<"field"<<'\t'<<"mag_per_spin"<<endl;
    int MCsteps = 100;    //no. of Monte Carlo step
    double mag_sum, old_energy;    // variable for magnetisation and energy

    //initializing my array of spins
    int s[p][n];    //create a 2-dim array
    initialize(s);

    //interaction on vertical axis - Trotter time axis
    double Jy = 0.5*temp*log(1/tanh(min_h/(temp*p)));
    old_energy = energy(s,Jy); //energy of initial configuration
    double* new_energy = &old_energy;

    int i=1;
   while(h< max_h)            //start from the min temp
    {
        //interaction on vertical axis - Trotter time axis
        Jy = 0.5*temp*log(1/tanh(h/(temp*p)));
        accept = 0;    // no. of accepted configuration at this temperature
        int seed = time(NULL);          //new seed
        srand(seed);

        //steps ssumed for our system to thermalize
        int thermSteps = 0.2*MCsteps;
        for(int i = 0; i < thermSteps; ++i)
            oneMC(s,new_energy, Jy);

        //we now assume our system is in thermal equilibrium
        //we have to now calculate our mean magnetisation
        mag_sum = 0.0;
        for(int i = 0; i < MCsteps; ++i)
        {
            oneMC(s, new_energy, Jy);    //geneartes a state near  equilibrium
            mag_sum += mag_per_spin(s);
        }

        //write data to file
        outfile<<h<<'\t'<<mag_sum/double(MCsteps)<<endl;
        //outfile1 <<h<<'\t'<<*new_energy<<endl;
        h += h_int; //increase the field

    }

    return 0;
}

void initialize(int s[][n])
{
    for (int i = 0; i < p; ++i)
        for(int j = 0; j < n; ++j)
            s[i][j] = 1;    //uniform start
}


bool metropolis(int s[][n], double* new_energy, double Jy)
{
    int k = rand()%n;    // choose site k on x axis for flipping
    int l = rand()%p;    //choose site l on y axis for flipping
    double r = rand()/double(RAND_MAX);
    double delta_E;
    //time to work—— just kidding!!\
    delta_E = 2*s[l][k]*(Jx*(s[l][(k==0)? n-1:k-1] + s[l][(k==n-1)? 0:k+1])
                + Jy*(s[(l==0)? p-1:l-1][k] + s[(l==p-1)? 0: l+1][k]));
    if (exp(-delta_E/temp) > r)      //metropolis algorithm
    {
        //*new_energy = delta_E + *new_energy;
        s[l][k] *= -1;
        return true;
    }

    else return false;

}

void oneMC(int s[][n], double* new_energy, double Jy)
{
    for(int i = 0; i < p; ++i)
        for(int j = 0; j < n; ++j)
             //call metropolis algorithm for the job. oops! kidding!
            if(metropolis(s,new_energy,Jy)) ++accept;
}

double mag_per_spin(int s[][n])
{
    int sum = 0;
    for(int i = 0; i < p; ++i)
        for(int j = 0; j < n; ++j)
            sum += s[i][j];
    return sum/double(n*p);
}

double energy(int s[][n], double Jy)
{
    double sum = 0.0;
    for(int i = 0; i < p-1; ++i)
        for(int j = 0; j < n-1; ++j)
            sum += Jx*s[i][j]*s[i][j+1] + Jy*s[i][j]*s[i+1][j];
                sum += log(pow(0.5*sinh(2*h/(temp*p)),double(p*n)/2.0));
    return -1*sum/double(n*p);    //return energy per spin
}
```