Contents

List of Figures

List of Tables

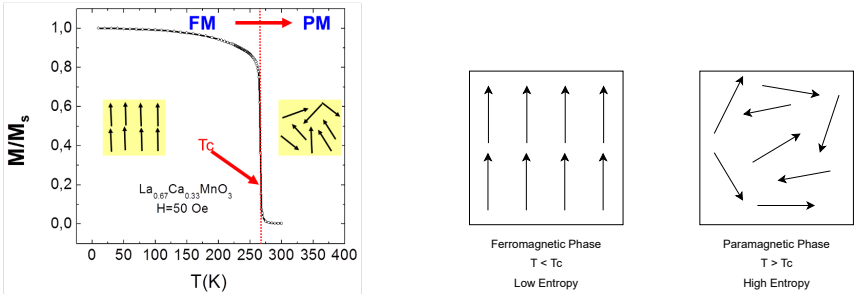
Chapter 1

Ferromagnetism and the Ising Model

In this Chapter a brief introduction to ferromagnetism is presented, from the history and significance of the Ising model, to the joint density of states and relevant thermodynamic relations and properties obtained from it.

1.1 Ferromagnetism

Magnetic materials are critical in our modern society, since they have very broad applications in our daily lives. For example, there are magnetic materials in every speaker and microphone, hard-drive disks on our computers, they play a big role in medicine where they are used in body scanners (usually in MRI machines) and together with applications in electric motors, transformers and generators [3]. Magnetic materials can be classified in terms of their magnetic properties, such as diamagnetism, paramagnetism, ferromagnetism, and so forth [4].



(a) (b)

Figure 1.1: (a) Magnetization curve for $La_{0.67}Ca_{0.33}MnO_3$ as a function of temperature;(b) Spins diagram for the ferromagnetic phase (left) and for the paramagnetic phase (right).

A ferromagnetic material exhibits spontaneous magnetization in the absence of an applied external magnetic field. This way, the magnetic moments of the atoms that compose the material are all naturally aligned along one direction [5]. One of the key features of ferromagnetic materials is that they only display this property below a certain well-defined critical temperature, T_C, usually called the Curie temperature. This temperature defines a phase transition between a ferromagnetic and a paramagnetic state, Figure 1.1(a). Above the Curie temperature, the entropy within the material becomes too strong and the magnetic moments start to dis-align, thus the materials looses its spontaneous magnetization, Figure 1.1(b). In other words, the effect due to thermal disorder overtakes the ferromagnetic order.

This kind of nature is common to various transition metal, Iron, Cobalt, Nickel, and so forth, and to some rare earth materials, such as Gadolinium. One application of ferromagnets is magnetic refrigeration. This is based on the magnetocaloric effect (MCE), which consists of a change in temperature of a magnetic material by the application and removal of an external magnetic field., known as the magnetocaloric effect (MCE). For a ferromagnetic material, the MCE is more prominent when the magnetic filed is applied for temperatures around the Curie temperature. High performance magnetic refrigerants include the LaFeSi 1:13 family [6] and MnFeP based intermetallics [7].

In the recent years, computational materials design is becoming the new norm to discover new materials by combining computer science with quantum and statistical mechanics. It is more advantageous than the experimental sciences since it is not as time-consuming and bounded by high costs in equipment as running an experiment [8, 9, 10].

1.2 Ising Model

In 1920 Wilhelm Lenz, a German physicist, gave Ernest Ising, his PhD student, an exercise that considered a one-dimensional chain of spin-1/2 particles that can be either pointing up or down. The spins can only interact with their neighbours. Later in 1925, Ising solved this problem, awarding him a doctorate in physics, and concluded that there was no phase transition in the one-dimensional case and wrongly extrapolated that there was no phase transition in higher dimensions [11]. It was not until 1944 that a Norwegian-born American physicist, Lars Onsager, solved the much harder two-dimensional case analytically, in a square lattice [12]. In this case, Onsager showed that there is a well defined phase transition from a ferromagnetic to a paramagnetic state at a critical temperature (T_C), therefore proving Ising's extrapolation wrong.

Since then, the Ising Model has become one of the most studied and published physical models, since it has a non-trivial phase transition while being able to have an analytical solution, at least for dimensions lower or equal to two. As of yet, there are no exact solutions for higher dimensions.We are only able to get properties from three dimensional lattices by numerical simulation.

The Ising Hamiltonian describes a system of lattice of atomic spins which can have two spin directions, up (+1) and down (−1) with neighbour to neighbour interactions. So, it is written as

$$\mathcal{H} = -\sum_{<i,j>} J_{ij} S_i S_j - H \sum_i S_i \tag{1.1}$$

where J is the interaction for constant between two neighbouring particles, S_i is the spin value of the particle at the site i, $< i, j >$ denotes that the sum is conducted over all neighbouring particles. The second term represents the interaction with an external magnetic field, H, and it is completely optional. If $J > 0$, the system will have a ferromagnetic behaviour since parallel spins are energetically more favourable, and if $J < 0$, the system will behave in an anti-ferromagnetic way.

In this work, I will not evaluate the Ising Model with an applied magnetic field, $H = 0$, and consider $J_{ij} = 1$ to simplify the computations. With this being said, the Hamiltonian treated in this work goes as follows,

$$\mathcal{H} = -\sum_{<i,j>} S_i S_j \equiv -\frac{1}{2} \sum_{ij} S_i S_j \tag{1.2}$$

To simulate real materials we can obtain the values of the interaction constant for each atom in our lattice through Density Functional Theory (DFT) calculations [13, 14].

1.2.1 Joint Density of States

It is mentioned in any undergraduate course in statistical mechanics that the density of states (DoS), $g(E)dE$, gives the number of states that have a certain energy between E and $E+dE$ [15]. For discrete systems, like the Ising Model, the DoS evaluated at a certain energy gives us the exact number of microstates that have a specific energy E. Through the DoS we can obtain the partition function $Z(T)$, as a function of temperature, and then compute energy related thermodynamic variables, such as the mean energy $E(T)$, specific heat $C(T)$, Helmholtz free energy, $F(T)$. As we are dealing with magnetic systems our natural interest is to study the magnetization throughout various temperatures and applied magnetic field intensity. For this, we need the number of microstates with a certain pair (E, M). This is the exact description of the Joint Density of States (JDoS), a multi-variable histogram with information about the number of microstates with a certain energy and another parameter, like density, ρ, number of particles, N, or, in this case, magnetization, M. This way we can compute the partition function as a function of both temperature and magnetization, $Z(T, M)$ and the Helmholtz free energy as $F(T, M)$.

Table 1.1 represents the JDoS for the Ising Model in a simple square (SS) lattice with length 2 and periodic boundary conditions (PBC). The extreme magnetization configurations appear only once in the JDoS, no matter the size of the lattice, since they represent all up or all down spins configurations.

Table 1.1: Joint Density of States for the SS lattice with $L = 2$ Ising Model. The rows are the values of energy, E, and the columns the values of magnetization, M.

E / M	-4	-2	0	$+2$	$+4$
-8	1	0	0	0	1
-4	0	0	0	0	0
0	0	4	4	4	0
$+4$	0	0	0	0	0
$+8$	0	0	2	0	0

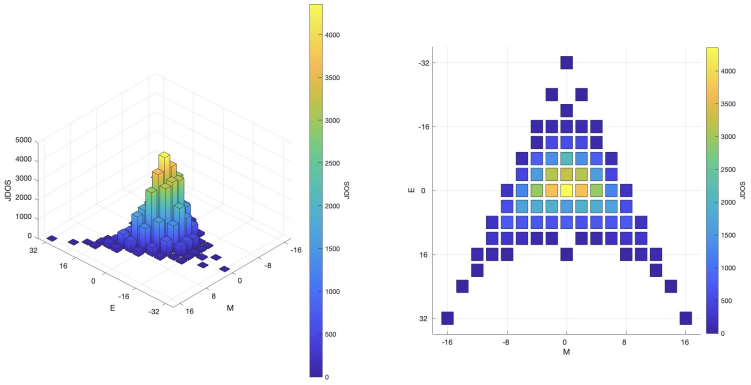


Figure 1.2: Plot of the exact JDoS for the SS L4 Ising Model with PBC. This was obtained by visiting each microstate available to the system.

There are three configurations worth noting.The first is the zero energy and zero magnetization state. This is the state with the most micro configurations since it represents the macrostate in which is likely to find the system in, when $T \gg T_C$(has the highest entropy). The second is the macrostate with the highest energy and zero magnetization. For every system size and lattice this macrostate has always 2 microstates, since the spins are sorted in a chess/checker board pattern. The last configuration is the state with the least energy and zero magnetization. Its value is always going to be equal to $2L$, since the spins are sorted in rows and columns of alternating positive and negative spins, similar to slices. Throughout this work these configurations will appear again and they will be referred as, Zero Zero, Checker Board and Slice, respectively.

The simplest way to compute the JDoS is to visit all of the possible microstates. It is easy to do this when we have 16 spins ($L = 4$ in a SS lattice, Figure 1.2), since the number of possible configurations is still quite small, $2^{16} \approx 65E3$ configurations. But, for instance, for systems with 256 spins ($L = 8$ in a SS lattice) it becomes computationally impossible to randomly or sequentially visit all of the microstates since we have to sample $2^{256} \approx 1E77$ configurations. Instead we can use various clever numerical methods methods to get an estimation of the JDoS in a much more reasonable time with fairly good precision. I will present widely studied methods in the next Chapter and a new unpublished method in the third Chapter.

1.2.2 Thermodynamics

From the JDoS we can obtain all of the thermodynamic quantities when the system is in an equilibrium state. In this section I will present some useful formulas to compute those variables from the JDoS. The probability of a given state with energy E_i, is given by

$$P_i = \frac{\sum_q g(E_i, M_q) \exp(-\beta E_i)}{Z}, \tag{1.3}$$

where β is defined as $\beta \equiv 1/k_B T$ and Z is the canonical partition function given by

$$Z = \sum_q Z(T, M_q) = \sum_q \sum_i g(E_i, M_q) \exp(-\beta E_i). \tag{1.4}$$

From this we can obtain mean thermodynamic variables,

$$\langle E \rangle = \frac{1}{Z} \sum_i \sum_q g(E_i, M_q) E_i \exp(-\beta E_i), \tag{1.5}$$

$$\langle M \rangle = \frac{1}{Z} \sum_q \sum_i M_q g(E_i, M_q) \exp(-\beta E_i) \equiv \frac{1}{Z} \sum_q M_q Z(T, M_q). \tag{1.6}$$

$\langle E \rangle$ is related to the specific heat by

$$\langle C \rangle = \frac{\langle E^2 \rangle - \langle E \rangle^2}{(k_B T)^2}, \tag{1.7}$$

and finally to obtain the mean entropy we use the second law of thermodynamics,

$$\langle S \rangle = \int \frac{\langle C \rangle}{T} dT. \tag{1.8}$$

However there is another way to estimate thermodynamic properties, from the computed JDoS. Using the Helmholtz free energy, defined as

$$F(T, M) = -k_B ln(Z(T, M)) \equiv U - TS \tag{1.9}$$

and the principle of minimum energy, we can extract $F_{min}(T) = \min(F(M, T))$, which is defined as the lowest temperature-dependent free energy that can be reached by the system. From it we can obtain the magnetization and energy for that value of free energy minima, $M_{F_{min}}$ and $E_{F_{min}}$, respectively.Through F_{min} we can use the following thermodynamic relations to compute the specific heat and entropy of the system:

$$C = -T\frac{\partial^2 F_{min}}{\partial T^2}, \tag{1.10}$$

$$S = -\frac{\partial F_{min}}{\partial T}. \tag{1.11}$$

1.2.3 Relevance

Despite its simplicity and age, the Ising Model is used in a multitude of research fields within the physical sciences and the social sciences.

Within the physical sciences the Ising Model is used to simulate not only magnetic materials but also systems that are characterized by nearest-neighbour interactions and undergo a phase transition Ising-like, this means systems that go from an ordered-low entropy phase to a disordered-high entropy phase at a specific critical temperature [16]. An example would be liquid vapour transitions, where the order parameter is the density, ρ, binary liquid mixtures, where the order parameter is the concentration and the transition corresponds to the mixing of the two liquids.

In the social sciences, the Ising Model has been used to describe a plethora of theoretical models of social behaviour and model financial markets [17]. Topics ranging from the study of racial segregation in certain communities [18], to a demonstration that a community that speaks only one language can start speaking another one without any outside bias [19].

In economics Monte Carlo methods are widely used to model financial markets due to its randomness, and the Ising Model is the standard model that those methods are applied to. In theses case studies the spins are the option of buying or selling a certain stock through neighbour to neighbour communication or external factors. The magnetization represents the average actions of the stock market agents [20, 21].

Chapter 2

Monte Carlo Methods Applied to the Ising Model

A short review of the Monte Carlo (MC) methods used to solve the Ising Model in the current paradigm is presented. There will be a focus on the famous Metropolis Method and the Wang-Landau sampling.

2.1 Metropolis Method

The classic Metropolis method, introduced in 1953 by Metropolis et al. [22], belongs to the Markov chain Monte Carlo (MCMC) class of algorithms. These algorithms exploit the fact that if we construct a Markov chain that has a specific equilibrium distribution one can obtain samples recording the states generated by the Markov chain.

In a brief fashion, a Markov chain is a stochastic model that describes a sequence of possible events, in this case microstates,in which the probability of transiting to another state depends on the current state. This way the probability of the next state, S_j, given the current state, S_i, can be written as

$$P(S_j, t) = \sum_i W(S_i \to S_j) P(S_i, t), \tag{2.1}$$

where $W(S_i \to S_j) \equiv W_{ij}$ is the transition probability to move from the state i to j. We require that

$$W_{ij} \geq 0 \qquad\qquad \sum_j W_{ij} = 1. \tag{2.2}$$

The master equation considers the change of the probability of the next state with time, t,

$$\frac{dP(S_j, t)}{dt} = \sum_i \left[W_{ij} P(S_i, t) - W_{ji} P(S_j, t) \right]. \tag{2.3}$$

In the equilibrium regime, the master equation has to equal 0, and we get the detailed balance condition for the equilibrium probability $P_{eq}(S_j)$,

$$W_{ji} P_{eq}(S_j) = W_{ij} P_{eq}(S_i). \tag{2.4}$$

The objective of Metropolis sampling is to generate canonical configurations with an equilibrium probability

$$P_{eq}(E_i) = \frac{\exp(-\beta E_i)}{Z}. \tag{2.5}$$

Here Z is the partition function, however this is usually not know before hand. When considering a Markovian process we generate each new configuration from the preceding one avoiding this problem. As a result the difference of energy between the two states is needed, $\Delta E = E_i - E_j$ and the transition probability of given as

$$W_{ij} = \begin{cases} \tau_0^{-1} \exp(-\beta \Delta E) & \text{if } \Delta E \geq 0 \\ \tau_0^{-1} & \text{if } \Delta E < 0 \end{cases} \tag{2.6}$$

where τ_0^{-1} is the time required to attempt a spin-flip. We often set this time unit to one.

This way the Metropolis method applied to the Ising Model [23], with a fixed external magnetic field and a fixed temperature, goes as follows:

1. Choose an initial state;
2. Choose a spin i and perform a spin-flip;
3. Calculate the energy change from that spin-flip, ΔE;
4. Accept the flip with a probability $\min(1, \exp(-\beta \Delta E))$;
5. When the system reaches equilibrium, measure any thermodynamic quantity needed;
6. Go back to (2) and repeat until there is enough samples of the thermodynamic variables.

Note that we accept the spin flip if a given uniformly random number $r, r \in [0, 1]$, is less or equal to the acceptance criteria. Typically in Monte Carlo simulations we define the MC time as the amount of trial flips equal to the number of spins in our lattice, N.

2.1.1 Success and Limitations

The Metropolis sampling proposed by Metropolis et al. [22], can be successfully applied to an array of models, ranging from widely studied quantum ensembles and gases simulations to state-of-the-art protein and peptide simulations and to machine learning and neural networks. The following paragraphs will be directed to magnetic systems, like the Ising model, but they can be extrapolated to others physical systems.

When estimating thermodynamic variables the simulation must reach the equilibrium stage, where the probability distribution takes the form of Equation 2.5. Then we take a measurement at each MC time, resulting in R total values for that variable. At the end the average of that variable, A, is taken $\langle A \rangle = \frac{1}{R} \sum_i A_i$. For large systems the time taken to reach equilibrium stage is often very long thus making the simulation time consuming. This is worsened by the fact that to study how some thermodynamic variable A changes over a wide range of temperatures or applied fields intensities, we need to run multiple Metropolis simulations for each temperature and field intensity values, making this process very time-consuming.

Lastly there is another shortcoming known as critical slowing down. In short, for computations where the temperature is near the critical temperature, T_C, the sampling slows down, meaning that it is more time-consuming for the computations to reach the equilibrium stage, thus slowing down the overall simulation.

2.2 Wang-Landau Sampling

Since its introduction, the Metropolis sampling was the go-to method to study phase transitions and critical phenomena in condensed matter physics and statistical mechanics. In the final decades of the 20th century scientists were committed to develop new methods that could overcome the shortcomings of the Metropolis sampling. Various methods were proposed such as the cluster flip algorithms, where Swendsen and Wang where pioneers, and the multicanonical ensemble method [24]. The first solved the critical slowing down present in the Metropolis and the second could sample rough energy landscapes with ease.

In 2001, Fugao Wang and David P. Landau [24, 25] proposed a new Monte Carlo method, now called the Wang-Landau (WL) method or sampling. The goal of this method diverges from the goal of previous methods. Instead of generating configurations with a canonical probability, this method tries to estimate the canonical partition function

$$Z = \sum_E g(E) \exp(-\beta E), \tag{2.7}$$

through the estimation of the density of states $g(E)$ from a flat histogram in the phase space.

The main difficulty of a simple random walk in the energy space is that the walker would spend most of its time in the highest probable states thus making the random walk ineffective. The idea of the Wang-Landau sampling is to do a random walk,Metropolis like, and accepting the new states with a probability proportional to the inverse to their DoS $\frac{1}{g(E)}$. Doing this we obtain a flat histogram meaning that each macrostate has the same probability of being visited. Since $g(E)$ is unknown *à priori*, in each step of the random walk, $g(E)$ is also being constructed.

The proposed algorithm computed the DoS but, it can also estimate the JDoS by performing the random walk in the phase space composed by energy, E, and the second order parameter, in our case, the magnetization of the system, M. This comes with a downside, since the JDoS has much more more information than the DoS it takes much longer compute. Later in 2006, Landau et al. [26], presented a modification of the WL called the global updates method. This is a much harder version to implement but is much more efficient when sampling 2D discrete phase spaces and continuous phase spaces [27].

2.2.1 Algorithm

First we start with an arbitrary configuration of spins and a guess for the density of states. Usually,this guess is $g(E, M) = 1$. Choosing a random site in our lattice, we perform a trial flip and compute the energy-magnetization pair before the trial, (E_i, M_i), and after, (E_j, M_j). The new configuration is accepted with a probability

$$P((E_i, M_i) \to (E_j, M_j)) = \min\left(1, \frac{g(E_i, M_i)}{g(E_j, M_j)}\right). \tag{2.8}$$

Whether the configuration is accepted or rejected, we have to update the histogram and refine the DoS estimation. This way, being the system in the state (E, M),

$$H(E, M) = H(E, M) + 1,$$

and we multiply the current value of the DoS by a modification factor, $f > 1$,

$$g(E, M) = f \times g(E, M).$$

A reasonable choice for the initial value of the modification factor if $f_0 = e$. If f_0 is too small, the simulation will take a very long time to reach all of the possible macrostates, (E, M). If it is too large, then we will have large statistical errors. This process is repeated until the histogram is considered "flat", all of the possible energies have been visited the same number of times. As it is impossible to obtain a 100% flat histogram, we define a rule for flatness as $\min(H(E,M)) > \langle H(E,M)\rangle \times p$; p is chosen according to the size of the problem. For small cases, such as the two dimensional Ising model, p can be set as high as 0.95, but for larger systems the flatness condition may never be satisfied if p is near unity. Once a flat histogram is reached, we set $H(E,M) = 0$, keep the estimation of the DoS and reduce the modification factor, $\sqrt{f_i} \rightarrow f_{i+1}$ and continue the random walk. We stop the simulation if $f < f_{final}$, where f_{final} is a number very close to one (often $f_{final} \sim 1 + 1E - 8$).

At the end, the method gives us the relative density of states. To determine the normalized DoS we can use the fact that

$$\sum_{E,M} g(E,M) = 2^N. \tag{2.9}$$

We can also use number of configurations that have a designated magnetization $\Omega(M)$ to normalize the JDoS. This way we normalize the JDoS in a sequential manner magnetization through magnetization. $\Omega(M)$ is just the possible combinations that exist given the number of spins down $N_\downarrow$ and the number of spins up $N_\uparrow$ ($N_\uparrow + N_\downarrow = N$). So $\Omega(M)$ cam be easily calculated by

$$\Omega(M) = \frac{N!}{N_\downarrow!(N - N_\downarrow)!}, \tag{2.10}$$

with

$$N_\downarrow = \frac{N - M}{2}. \tag{2.11}$$

2.2.2 Success and Limitations

The modification factor controls the accuracy and the steps taken to reach a flat histogram in our computations. As it approaches unity, the number of iterations goes to infinity. This way, at the beginning of the simulation, when the modification factor is large, the estimation of the DoS is quite bad, since there we are taking fewer samples with a high weight. At the later stages, the modification factor is lower therefore we have more samples of each macrostate with a lower overall weight. Therefore, the initial stages of the simulation are characterized by the accumulation of low precision statistics and the later stages by the refining of the firsts samples.

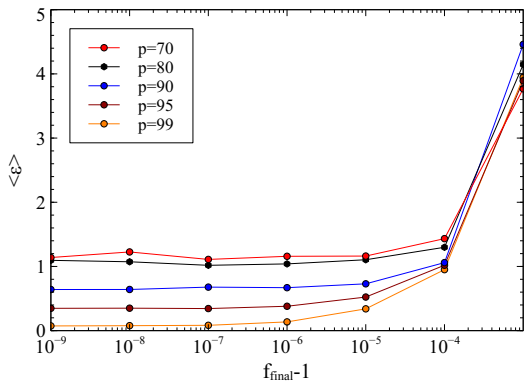
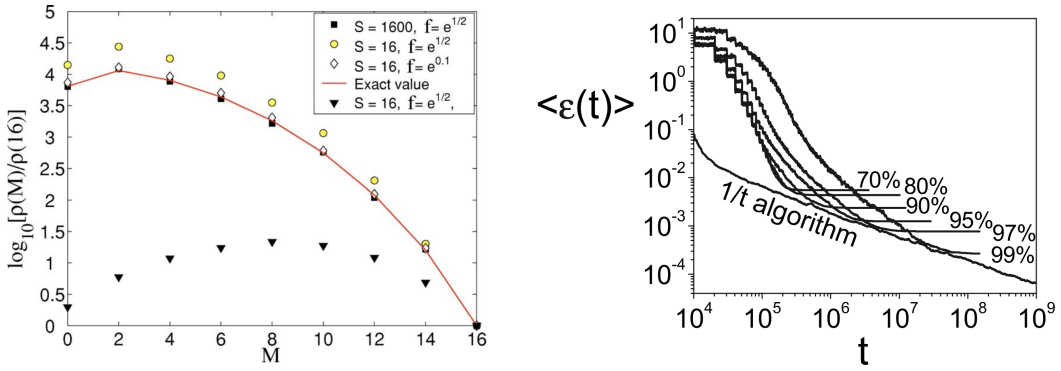


Figure 2.1: Mean absolute error of the JDoS for the Ising model computed by the Wang-Landau sampling for a L4 SS lattice plotted against $f_{final} - 1$. The values were averaged over 100 simulations.

Due to the biased samples at the initial stages of the computations, the final estimation of the JDoS will converge, with high precision, to a wrong solution independent of the flatness criteria and f_{final} value used, shown in Figure 2.1.

A few modifications to the original WL were proposed throughout the years, but the most noteworthy are the modifications from Chenggang Zhou et al. [1] and Belardinelli et al. [2]. Zhou has studied the method in great detail [1] and was a co-author of the global updates method [26]. He proposed the introduction of a parameter S, defined as the separation between successive records in the histogram. Meaning that in our random walk we would only update the histogram each S steps. This will diminish the correlation between successive samples and improve accuracy. The modification proposed by Belardinelli et al. is more complex and introduces a new way of changing the modification parameter during the computations, they called it the $1/t$ time-dependent algorithm. The method starts as the WL does, however, if $f_{i+1} \leqslant 1/t$, where t is the MC time, the modification factor becomes $f_{i+1} = f(t) = 1/t$ and we discard the histogram and update f each MC time step. The simulation only stops when $f(t) < f_{final}$. This ensures that the mean error vanishes, Figure 2.2(b).



(a) (b)

Figure 2.2: (a) $g(M)$, the density of states for magnetization M of L4 SS Ising lattice normalized. The solid line connects the exact values, and the symbols were obtained with different parameters f and S. Data was averaged over 100 measurements, so the statistical errors are smaller than the symbols. Taken from [1]. (b) Comparison between the mean error computed by the original WL for different flatness criteria and the mean error calculated using the $1/t$ time-dependent algorithm for a L8 SS Ising lattice. Taken from [2].

Either in its original forms, or using these more recent improvements, the WL method has become the go-to method for DoS and JDoS estimation, because of its efficiency and ability to sample the whole phase space even if the estimation of the JDoS is not the most precise [1, 2]. It has been applied to magnetic systems like the Ising model, Lennard-Jones fluid simulations, biologic processes, etc [28, 29, 30].

Chapter 3

Flat Scan Sampling

Background for the Flat Scan Sampling (FSS) method and a general description are presented along with the writers C++ implementations and an analysis of the single core and parallel performance.

3.1 Background

The author of FSS, João Amaral, had previously, in 2014, proposed a new method to estimate the JDoS, Random Path Sampling (RPS). The RPS was implemented in high performance languages and extensively studied by Nuno Fortunato. (citar)

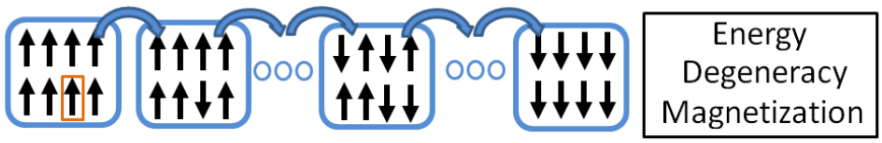


Figure 3.1: Scheme of how the Random Path Sampling method works.

The RPS method departs from the premise that by starting on an extreme magnetization point in the phase space, generally all spins up (M), and successively flipping one spin down at each step of the random walk, we arrive at the other end of the phase space ($-M$). Process illustrated in Figure 3.1. Performing R sweeps of the phase space generate a histogram that is flat in magnetization and we can obtain the JDoS by

$$H(E,M)/R = P(E,M), \tag{3.1}$$

$$\Omega(M) \times P(E,M) = g(E,M). \tag{3.2}$$

$\Omega(M)$ is defined in Equation 2.10.

The idea for FSS departs from the basic mechanism of RPS, in the sense that it is a method that estimates the JDoS by a sequential sweep of the phase space magnetization by magnetization. However the way of that both methods sample the phase space is completely different. The FSS takes a similar approach to the WL method, in the sense that a random walk with probability proportional to the inverse of the DoS is performed, a flat energy random walk.

3.2 Algorithm

The Flat Scan Sampling method stems from the observation that if the DoS at a certain magnetization M_q is known, by performing a random walk the energy space (E, M_q), with a probability proportional to the inverse of the DoS $\frac{1}{g(E)}$, called a flat energy random walk, and by sampling a set number of statistically diverse configurations for each energy value the DoS at the next magnetization M_{q+1} can be estimated. This is possible because at each step of the random walk we perform a scan, i.e., in a sequential manner, we flip and unflip each spin in our configuration obtaining information about the DoS at the next magnetization, $g(E, M_{q+1})$. The value of $g(E_j, M_{q+1})$ is then computed through the value of $g(E_i, M_q)$ by the following equation

$$g(E_j, M_{q+1}) = g(E_i, M_q) \times \text{fraction of configurations.} \tag{3.3}$$

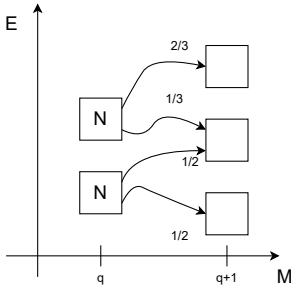


Figure 3.2: Scheme of how the Flat Scan Sampling works.

Shown in Figure 3.2, the fraction of configurations corresponds to the fraction of the scanned configurations in the random walk that contributed to the estimation of the DoS at the next magnetization. This way, the JDoS is computed sequentially by starting at a known DoS in the phase space, such as all of the spins up (M), and sweeping the whole phase space magnetization by magnetization until we arrive at the configuration where all of the spins are down $(-M)$.

The principal parameter of the method is the maximum number of samples for each point in the energy space, known as REP. In later section new parameters will be introduced to try to make the estimation more accurate while sacrificing some performance. There will be also an extensive study of how this parameters affects the precision of the JDoS and wall time.

The algorithm can be written in the following steps:

1. Choose a magnetization value where $g(E, M_q)$ is known, usually magnetization where the spins all up/down;
2. Generate a certain configuration in that magnetization and compute its energy, E_i;
3. Choose a spin to flip down and another one to flip up and compute the energy of the new configuration, E_j;
4. Accept the new configuration with a probability $\min(1, g(E_i)/g(E_j))$;
5. Sequentially flip each spin in the configuration, taking the system from the state (E_i, M_q) to (E_j, M_{q+1}) and accumulate a histogram $H(E_i, E_j)$ and unflip the spins;
6. If the all of the number of sampled states per (E, M_q) pair is equal to REP, stop the simulation and compute the DoS at $q+1$ by using Equation 3.3. Where the fraction of configurations is now equal to $H(E_i, E_j)/\sum_j(H(E_i, E_j))$.

3.3 CPU Implementation

For single core and message passing interface (MPI) implementations, C++ was the preferred because of its speed and optimization over python or MatLab, and modularity, over C. The random number generator (RNG) used was the xoshrio256**.

Before the description of the implementation let us define the function that handles the scan. This should be performed each step of the simulation. The scan is defined as flipping each spin in the configuration, measuring the new energy and registering the change in state $E_i \to E_j$ in the histogram. Thus, this operation can be implemented with a for cycle running from 0 to N, the number of spins.

```
1: function SCAN(configuration)
2:     for idx = 0,1,..., N-1 do
3:         flip the spin idx spin down
4:         compute the new energy Ej
5:         H(Ei, Ej)++
6:         flip the spin idx spin up
7:     end for
8: end function
```

The actual implementation follows the base-line algorithm described in the last section. By knowing that the JDoS is symmetric, to save computing time, we can estimate only half of the JDoS and after the simulation mirror it. Here `qmax` is the index of the last magnetization in our computation. The variable `hist(E)` is used to count how many configurations were sampled in each point of the energy space. We only stop when every point has sampled REP microstates. In pseudo-code it can be written as follows

```
1: for q=0,1,...,qmax do
2:     set hist(E) = 0 and H(Ei,Ej)=0
3:     generate random configuration with M=Mq and compute its energy Ei
4:     scan(configuration)
5:     hist(Ei)++
6:     while min(hist(E) < REP) do
7:         flip one random spin down
8:         flip one random spin up
9:         compute the energy of the new configuration Ej
10:        set ratio = min(g(Ei)/g(Ej))
11:        if rand() < ratio then
12:            accept new configuration
13:        else
14:            reject new configuration
15:        end if
16:        if hist(Ei) < REP then
17:            hist(Ei)++
18:            scan(configuration)
19:        end if
20:    end while
21:    set g(E,Mq+1) = g(E,Mq) * H(Ei, Ej) / sum(H(:,Ej))
22: end for
```

This implementation has the same problem as the original WL method. We sample successive configurations, thus reducing statistical accuracy and increasing correlation between scans. This way, a new parameter that reduces correlation between scanned configurations is proposed. It is called skip and acts the same way as the parameter S in the WL. We only sample configurations that are distanced by skip steps in the random walk. Only line 17 is modified by the addition of "and `k % skip = 0`". Usually this value is set equal to the number of spins in the system, N.

3.4 Validation and Convergence

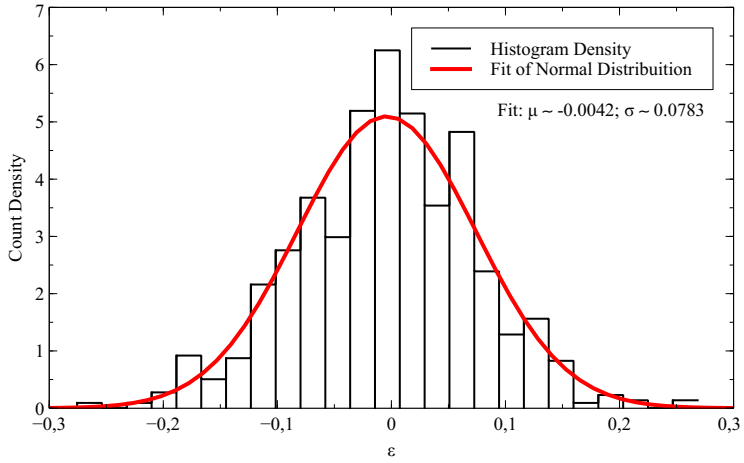


Figure 3.3: Scheme of how the Flat Scan Sampling works.

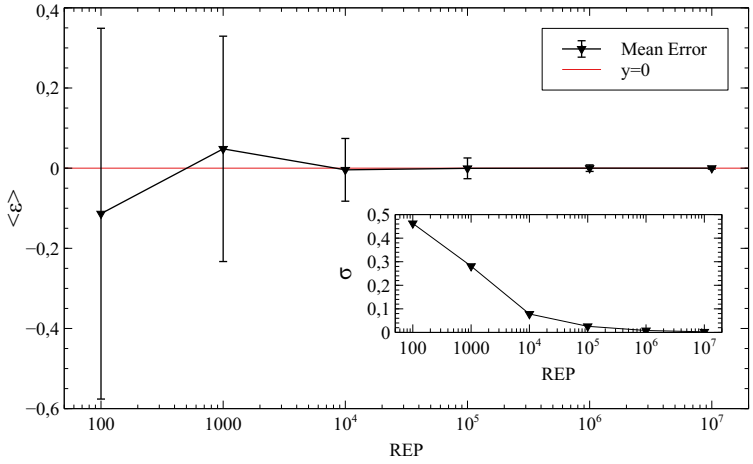


Figure 3.4: Scheme of how the Flat Scan Sampling works.

3.5 Comparison with Wang-Landau

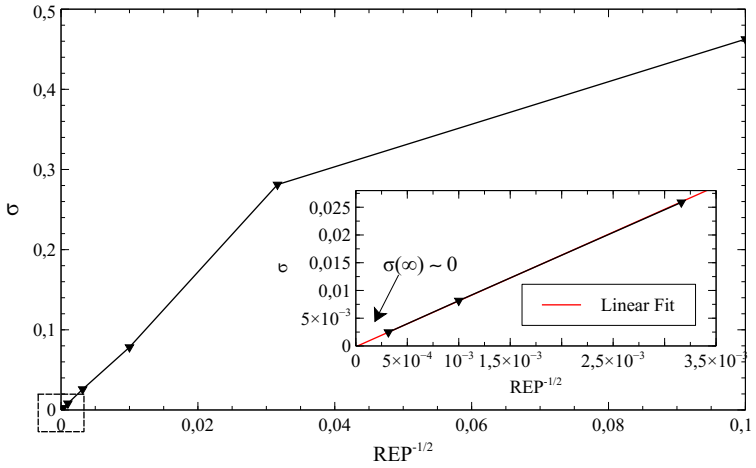


Figure 3.5: Scheme of how the Flat Scan Sampling works.

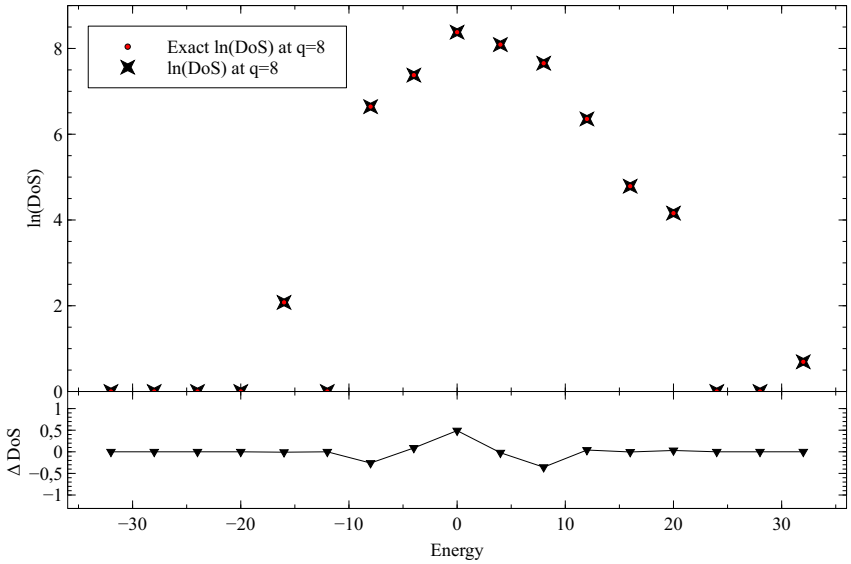


Figure 3.6: Scheme of how the Flat Scan Sampling works.

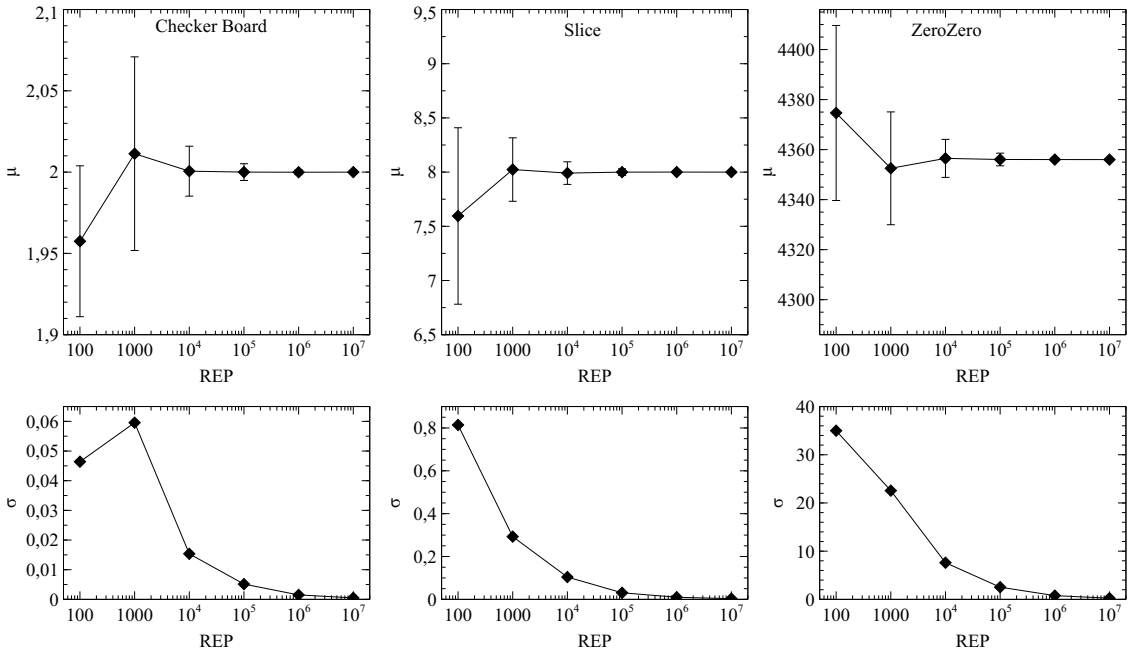


Figure 3.7: Scheme of how the Flat Scan Sampling works.

Chapter 4

Parallel Implementation

4.0.1 Parallel Implementation

Before addressing the approach used, a small introduction to two different parallel paradigms will be given. As the majority of modern CPUs have at least 4 physical processing cores, CPU code parallelization has become more important then ever to exploit all of the performance provided by modern CPUs. Not only that but as the problem that we are trying to solve becomes more complex or when we want to simulate bigger systems, single core computations may not be able to solve the problem in a reasonable amount of time. This way there is a need for parallelization. GPU computing is also an alternative but, generally, is a more complex approach.

In computer science there are two main paradigms for parallel applications. One based on threads, shared memory parallel programming, and another based on processes, distributed memory parallel programming. A shared memory program is executed in multiple threads that coexist in the same memory space. Thus each one has access to the other memory and vice-versa. Therefore communication between concurrent executions of the code within the process is easy. However with multi-threading the code can only be executed in the same computer since it is only run in a single process. Multiple threads running within the same process can reduce parallel performance,. In C/C++, there are many libraries that implement this style of programming, the ones worth mentioning are OpenMP and PThreads. On the other hand, a distributed memory concurrent program is executed through various processes each in their own memory space. Communication between processes is harder than thread communication. Multi process communication is more performance taxing since they live in different memory spaces. Excessive communication can lead to performance downfalls, known as communication overhead. This downside, however, comes with better parallel performance and the ability to execute computations in various computing nodes. The default C/C++ library for distributed memory parallel programming is the message passing interface (MPI) having many implementations, such as OpenMPI, MPICH or MVAPICH.

Due to the nature of FSS algorithm we can have various independent walkers sampling their own histogram. Each walker performs an independent random walk with access to the whole energy space and constructs a histogram for a magnetization M_q. All of the sampled histograms are combined to compute the DoS at M_{q+1}. The next iteration is performed using the computed DoS by joining all of the histogram contributions in the last iteration. This way, in one iteration there is only need for 2 communications. One is the broadcasting of the computed DoS in the last iteration to all of the walkers, the other is the collection of the sampled histograms by all of the walkers. There is a need for one walker to execute all of the serial computations such as the outputting to the terminal and to files, and performing the computations of the DoS at the end of an iteration and broadcasting it to all of the other

walkers. Considering this approach, distributed memory parallelism was the better option for a high performant implementation and MPI was used to carry the process to process communications.

There are two ways of dividing the computations between n walkers. Each performs a random walk with $\text{REP}_{walker} = \text{REP}/n$ giving us the wall time of a single core computation with REP samples per energy divided by the number of walkers or having each walker sample REP configurations per energy giving us the wall time of a single core REP computation.

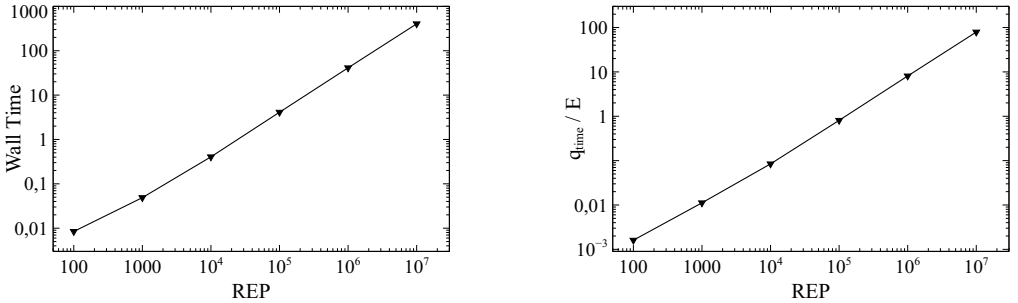
The MPI implementation starts by assigning each walker a different seed for the RNG, so the results come from different streams of random numbers. This is achieved by multiplying a defined seed by the number assigned to that walker. Then, the first iteration is computed by the manager process and broadcasted to all of the walkers. When all of the walkers have sampled the assigned number of configurations, they send the histogram to the manager and it computes the DoS at the third magnetization finally broadcasts it to the walkers. This is repeated until $q \geqslant q_{max}$. At the end, the manager writes the output files.

Two scenarios of this implementation were explored. One where the manager also performed is own random walk through the energy space, acting like a manager and a walker, and another were the manager only performed the single core operations.Here the advantage is on the side of the first scenario, since there is one more walker and having that walker also performing single core tasks gives an overall better performance than having one idle processes during 95% of the computation.

4.1 Performance

Now let us discuss the performance of FSS method of both implementations. The following single core benchmarks were performed in a computer equipped with a Ryzen 9 5950X at stock speeds.

In the following figure, we can observe the wall time and the time spent sampling per energy value as a function of the parameter REP. We can see that the method scales linearly with the parameter REP.



(a) (b)

Figure 4.1: (a) LogLog plot of the single core wall time of FSS as a function of the number of samples per each energy point, REP. (b) LogLog plot of the time spent sampling for each energy value, q_{time}/E.These are computations for a system with 16 spins. Data was averaged over 1000 computations to reduce statistical error.

Each iteration of the simulation there is an output to the terminal where the q_{time}/E is given. This value is computed by dividing the wall time spent on that iteration by the number of sampled energy points. The time spent sampling configurations for each energy value is a very important metric. It can tell us if the computations are stable or the number of samples,

REP, must be increased. For a stable calculation, this time must remain approximately the same throughout the computation. However, if it is increasing each iteration then usually the value of the parameter REP is insufficient to explore all of the possible energies in the next magnetization. Which causes the estimation of the of the DoS at M_{q+1} to be wrong. In the next iteration the method does not have enough information about the DoS so the time spent sampling will increase. This will propagate until the end of the computations. The obvious solution would be to increase the value of REP, but increasing the skip value is enough, sometimes.

Back to the single core performance. The FSS method scales linearly with the repetitions, so given the wall time of that system with a certain REP value we can easily estimate how much time the computations will take for a another REP value. When running the algorithm for a new system we can not have a precise estimation for the wall time. However, knowing only the q_{time}/E for the first iterations and assuming that the REP value is sufficient so that this time does not increase, we can estimate the wall time. Assume that the system has NE values of energy and NM values of magnetizations available.The JDoS will have $NE \times NM$ points, but since we only compute half of the JDoS, $(NE \times NM)/2$. About 1/3 to 1/2 of those points will be different than 0, so the estimation for the wall time comes as

$$\text{Wall Time} \approx (q_{time}/E)\frac{NE \times NM}{2} \times \text{fraction of non-zero points.} \tag{4.1}$$

In Table 4.1 different wall times for single core computations can be seen with the respective estimation through Equation 4.1 for two different fractions of non-zero energy points, 1/3 and 1/2.

Table 4.1: Wall Time of some Ising systems with their respective estimations by Equation 4.1 with two different values for the fraction of non-zero energy points.

Lattice	N	$NE \times NM$	q_{time}/E	Wall Time	Estimation 1/3	Estimation 1/2
SS	16	289	$0.11s$	$4s$	$5s$	$8s$
SS	64	4225	$0.85s$	$728s$	$600s$	$900s$
SS	256	66049	$1s$	$18942s$	$10898s$	$16512s$
SC	64	6305	$1.1s$	$1148s$	$1144s$	$1733s$
FCC	256	197633	$20s$	$871118s$	$652188s$	$988165s$

4.1.1 Parallel Scalability

Tests for parallel scalability were performed in the LIP super computer....

Consider a certain task that takes a time T to be solved sequentially by one core. Using N workers, ideally, it would only take T/N to complete the task. This is called a speedup on N. Ideally, this division of the task assumes that the task can be divided into N pieces of equal complexity, that take the same amount of time to finish. In reality this is not true, the N pieces could have little differences in complexity resulting in some workers having to wait for the other to finish. This is known as load imbalance and induces serialization in the computations.

Let us now construct a model for scalability. Considering the overall problem size is $s + p = 1$, where s is the nonparallelizable part and p is the perfectly parallelizable part.In theory, we would want $s = 0$ and $p = 1$. In reality, there are many reasons for a nonvanishing serial part. There can be algorithmic limitations meaning that there might be operations that can just not be performed in parallel, bottlenecks in the computer system, such as shared paths to memory between different cores, and communications overhead meaning that in

order for workers to inter-communicate there must be some serialization. This last point is often the cause for poor parallel performance.

The serial wall time can be written as

$$T_s = s + p, \tag{4.2}$$

while the wall time of solving the same problem with N workers comes to

$$T_p = s + \frac{p}{N}. \tag{4.3}$$

This way, application speedup or parallel scalability can be define as the quotient of parallel and serial performance. Serial performance is the serial work done over time

$$P_s = \frac{s+p}{T_s} = 1, \tag{4.4}$$

and parallel performance is

$$P_p = \frac{p+s}{T_p} = \frac{1}{1-p+\frac{p}{N}}. \tag{4.5}$$

The application speedup is now

$$S = \frac{P_p}{P_s} = \frac{1}{1-p+\frac{p}{N}}. \tag{4.6}$$

This last equation is known as Amdahl's Law, derived by Gene Amdhal in 1967. This limits the speeup when $N \to \infty$ to $1/(1-p)$. This famous equations tells us how much faster can a program run when using N CPU cores. In Figure 4.2 the Amdahl's Law speedup is plotted against the number of cores for different values of p.

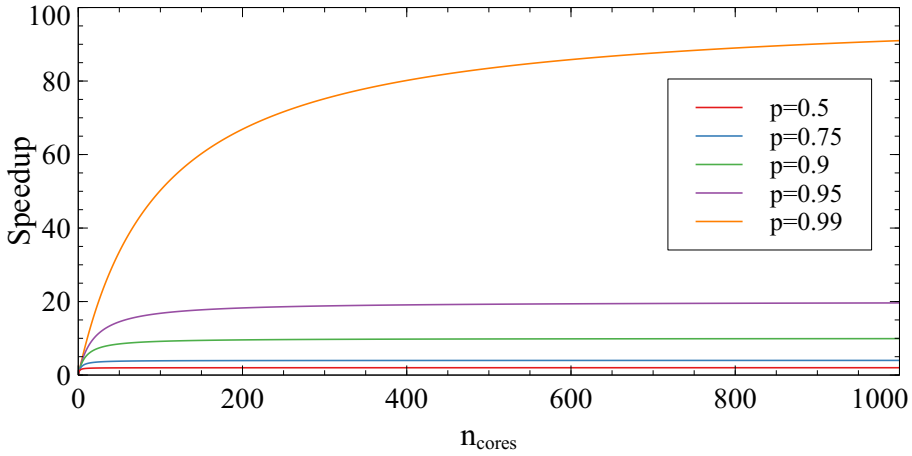


Figure 4.2: Speedup from Amdahl's Law as a function of the number of cores for five different values of the parallel portion of the code p.

There are modifications to this law to accommodate system bottlenecks, parallel synchronization and inter-core communications. The most noteworthy is the Hill and Marty's model, which introduces a new way of defining the work and performance. An asymmetric architecture has one manager that does not work and a symmetric architecture has a manager that also does work. The performance of the Hill and Marty's model depends on the type of architecture considered.

$$P(N,r) = \begin{cases} \frac{N-r}{r}\sqrt{r} & \text{if symmetric} \\ n-r & \text{if assymetric} \end{cases} \tag{4.7}$$

where r is the size of the sequential core used during the parallel execution.

We can now fit this model to the wall time versus core count data taken from the computations at ...

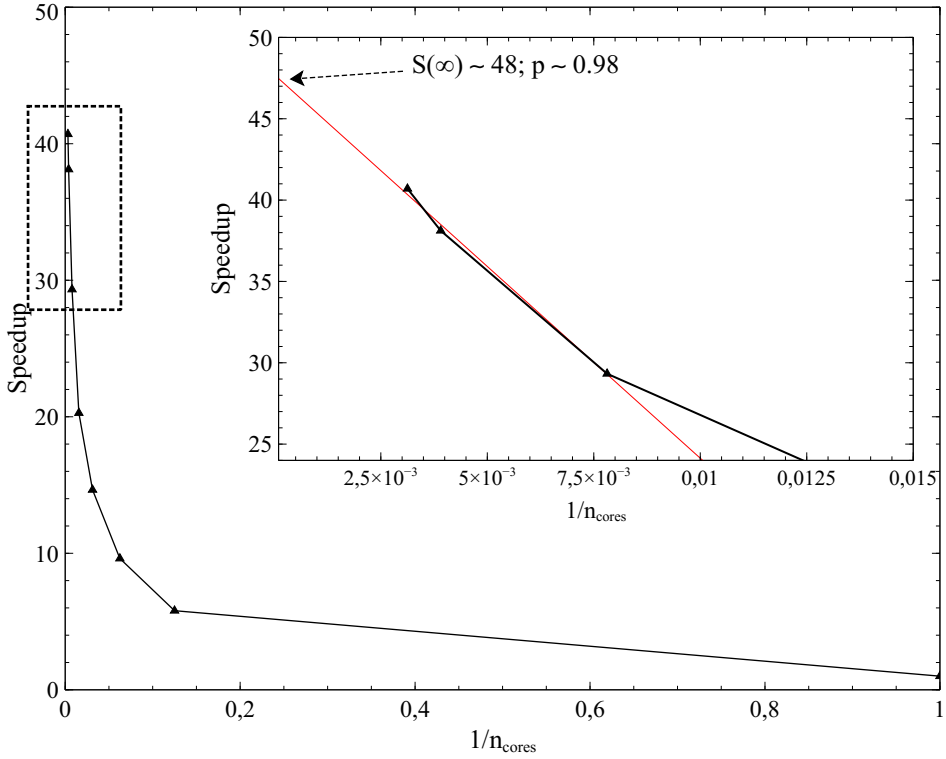


Figure 4.3: Scheme of how the Flat Scan Sampling works.

4.2 Comparison with Wang-Landau Sampling

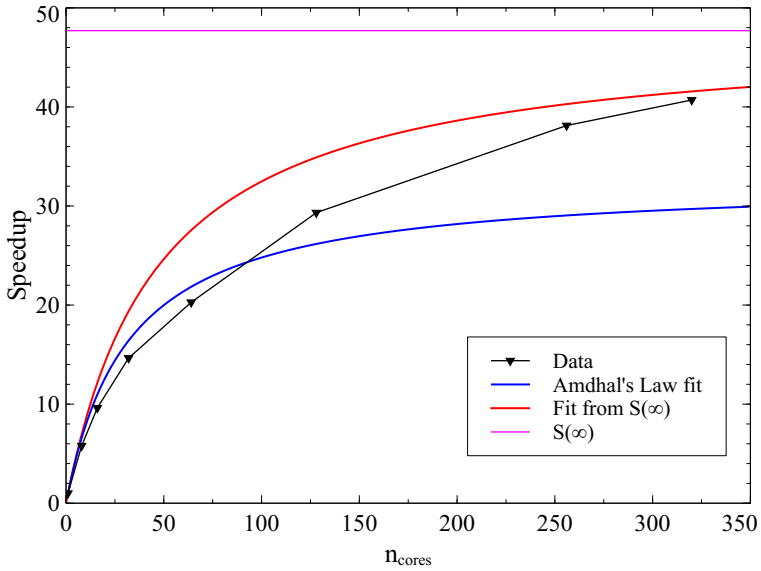


Figure 4.4: Scheme of how the Flat Scan Sampling works.

Chapter 5

Thermodynamics and Finite Size Scaling

Chapter 6

Conclusion and Future Work

Bibliography

[1] C. Zhou and R. N. Bhatt, "Understanding and improving the Wang-Landau algorithm," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 72, no. 2, pp. 1–4, 2005.

[2] R. E. Belardinelli and V. D. Pereyra, "Fast algorithm to calculate density of states," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 75, no. 4, pp. 1–5, 2007.

[3] O. Gutfleisch, M. A. Willard, E. Brück, C. H. Chen, S. G. Sankar, and J. P. Liu, "Magnetic materials and devices for the 21st century: Stronger, lighter, and more energy efficient," *Advanced Materials*, vol. 23, no. 7, pp. 821–842, 2011.

[4] D. J. Griffiths, *Introduction to Eletrodynamics*. Cambridge University Press, 4th ed., 2013.

[5] Stephen Blundell, *Magnetism in Condensed Matter*. 2001.

[6] A. Fujita, S. Fujieda, Y. Hasegawa, and K. Fukamichi, "Itinerant-electron metamagnetic transition and large magnetocaloric effects in La(FexSi1-x)13 compounds and their hydrides," *Physical Review B - Condensed Matter and Materials Physics*, vol. 67, no. 10, pp. 1044161–10441612, 2003.

[7] O. Tegus, E. Brueck, K. H. J. Buschow, and F. R. de Boer, "Transition-Metal-Based Magnetic Refrigerants for Room-Temperature Applications.," *ChemInform*, vol. 33, no. 14, pp. no–no, 2010.

[8] S. Curtarolo, G. L. Hart, M. B. Nardelli, N. Mingo, S. Sanvito, and O. Levy, "The high-throughput highway to computational materials design," *Nature Materials*, vol. 12, no. 3, pp. 191–201, 2013.

[9] W. Chen, J. George, J. B. Varley, G. M. Rignanese, and G. Hautier, "High-throughput computational discovery of In2Mn2O7 as a high Curie temperature ferromagnetic semiconductor for spintronics," *npj Computational Materials*, vol. 5, no. 1, 2019.

[10] S. Sanvito, C. Oses, J. Xue, A. Tiwari, M. Zic, T. Archer, P. Tozman, M. Venkatesan, M. Coey, and S. Curtarolo, "Accelerated discovery of new magnets in the Heusler alloy family," *Science Advances*, vol. 3, no. 4, pp. 1–10, 2017.

[11] E. Ising, "Beitrag zur Theorie des Ferromagnetismus," *Zeitschrift für Physik*, vol. 31, no. 1, pp. 253–258, 1925.

[12] L. Onsager, "A Two-Dimensional Model with an Order-Disorder Transition," *Physical Review Letters*, vol. 65, no. 3 and 4, pp. 117–149, 1944.

[13] G. Bihlmayer, *Density Functional Theory for Magnetism and Magnetic Anisotropy*. 2020.

[14] N. Marzari, A. Ferretti, and C. Wolverton, "Electronic-structure methods for materials design," *Nature Materials*, vol. 20, no. 6, pp. 736–749, 2021.

[15] R. K. Pathria and P. D. Beale, *Statistical Mechanics*. 3rd ed.

[16] A. Pelissetto and E. Vicari, "Critical phenomena and renormalization-group theory," *Physics Report*, vol. 368, no. 6, pp. 549–727, 2002.

[17] D. Stauffer, "Social applications of two-dimensional Ising models," *American Journal of Physics*, vol. 76, no. 4, pp. 470–473, 2008.

[18] P. Abell and P. Abell, "The Journal of Mathematical Sociology Some Aspects of Narrative Method," no. July 2015, pp. 37–41, 2010.

[19] D. Nettle, "Is the rate of linguistic change constant?," *Lingua*, vol. 108, no. 2-3, pp. 119–136, 1999.

[20] L. Damodaran and K. M. Udayanandan, "Dynamics of stock market , using Ising model," pp. 71–78.

[21] P. dvorak, "Ising Model in Finance From Microscopic Rules to Macroscopic Phenomena," pp. 1–56, 2012.

[22] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[23] D. P. Landau and K. Binder, *A Guide to Monte Carlo Simulations in Statistical Physics*. 2005.

[24] F. Wang and D. P. Landau, "Efficient, Multiple-Range Random Walk Algorithm to Calculate the Density of States," *Physical Review Letters*, vol. 86, pp. 2050–2053, mar 2001.

[25] D. P. Landau, S.-H. Tsai, and M. Exler, "A new approach to Monte Carlo simulations in statistical physics: Wang-Landau sampling," *American Journal of Physics*, vol. 72, no. 10, pp. 1294–1302, 2004.

[26] C. Zhou, T. C. Schulthess, S. Torbrügge, and D. P. Landau, "Wang-landau algorithm for continuous models and joint density of states," *Physical Review Letters*, vol. 96, no. 12, pp. 8–11, 2006.

[27] P. Poulain, F. Calvo, R. Antoine, M. Broyer, and P. Dugourd, "Performances of Wang-Landau algorithms for continuous systems," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 73, no. 5, pp. 1–11, 2006.

[28] K. A. Maerzke, L. Gai, P. T. Cummings, and C. McCabe, "Simulating phase equilibria using wang-landau-transition matrix monte carlo," *Journal of Physics: Conference Series*, vol. 487, no. 1, 2014.

[29] Q. Yan, R. Faller, and J. J. De Pablo, "Density-of-states Monte Carlo method for simulation of fluids," *Journal of Chemical Physics*, vol. 116, no. 20, pp. 8745–8749, 2002.

[30] Q. Yan and J. J. de Pablo, "Fast Calculation of the Density of States of a Fluid by Monte Carlo Simulations," *Physical Review Letters*, vol. 90, no. 3, p. 4, 2003.