Monte Carlo Simulations of $\mathbb{Z}_2$ Lattice Gauge Theory.

Dae Heun Koh • March 14, 2019

Abstract

We discuss the monte-carlo analysis of the 3D Ising model, the 3D $\mathbb{Z}_2$ lattice gauge theory (LGT), and 4D lattice gauge theory. Using the Metropolis algorithm, we confirm the duality relations between 3D $\mathbb{Z}_2$ LGT and the 3D Ising model, and demonstrate clear evidence of continuous phase transitions. Using finite size scaling, we determine T_c and extract the critical exponents ν and α for the 3D Ising and 3D LGT, which show continuous phase transitions at the critical points. We examine, in particular, the low temperature behavior of the Wegner-Wilson loop operators, and show evidence for the perimeter law. We continue our simulations of bbZ_2 lattice gauge theory in 4 dimensions, and demonstrate strong evidence of a first-order phase transition accompanied by the existence meta-stable states.

Contents

1 Introduction

In this project report, we will study the Monte Carlo Simulations of Lattice Gauge Theories. Before proceeding to technical details of numerical simulations and calculations, here we present the theoretical background for the topic of the project which is *lattice gauge theory*. A good starting point for our discussion is the 3-dimensional Ising model.

1.1 The 3D Ising model

While the 1D and 2D Ising models have an exact analytic solution, the exact solution for the 3D Ising model remains an open question. The partition function for the 3D Ising model (at zero external magnetic field) is given by:

$$Z = \sum_{\{s_i=\pm 1\}}^{N} \exp\left[\beta J \sum_{\langle i,j\rangle} s_i s_j\right] \equiv \sum_{\{s_i=\pm 1\}}^{N} \exp\left[K \sum_{\langle i,j\rangle} s_i s_j\right], \tag{1.1.1}$$

where $J > 0$ for a ferromagnet, $s_i = \pm 1$, and $\langle i, j\rangle$ is the sum over all nearest neighbors of the lattice. For simplicity, we will define $K \equiv \beta J \equiv \beta \equiv 1/T$, after setting $J = 1$. Considering the low-temperature expansion of the 3D Ising model, one can show that the low temperature expansion of the 3D Ising model have the same mathematical structure as the high temperature expansion of the following theory:

$$Z = \sum_{\{U=\pm 1\}} \exp\left[\tilde{K} \sum_{\square} U_{ij} U_{jk} U_{kl} U_{lj},\right] \tag{1.1.2}$$

where the variables $U_{ij} = \pm 1$ indicate the bond from node s_i to s_j and $\square$ indicate the sum over all square plaquettes. The bond variables U_{ij} are multiplied around a single square plaquette, and then summed over. As demonstrated in [9], equation (1.1.2) describes a 3D $\mathbb{Z}_2$ *lattice gauge theory*. The $\mathbb{Z}_2$ lattice gauge theory is the simplest example of a lattice gauge theory, which we explain in the next section. In particular, for the model in (1.1.2), the following duality relations hold between the 3D Ising model and the $\mathbb{Z}_2$ lattice gauge theory:

$$e^{-2K^*} = \tanh \tilde{K}^*, \tag{1.1.3}$$

where K^* is the critical value of K for the 3D Ising model and $\tilde{K}$ is that of the $\mathbb{Z}_2$ lattice gauge theory. Using monte-carlo estimates of K and $\tilde{K}$ from finite-size scaling, we will confirm the above duality relation in the proceeding sections.

For the 3D Ising model, some interesting physical observables that characterize the phase transition include the *magnetization* m, *free energy* E, *specific heat* C, and *susceptibility* χ:

$$m \equiv \frac{1}{N_s} \sum_{i=1}^{N_s} s_i, \quad E \equiv -\frac{1}{Z}\frac{\partial Z}{\partial \beta}, \quad C \equiv -k_B \beta^2 \frac{\partial E}{\partial \beta}, \quad \chi \equiv \frac{\partial m}{\partial h}, \tag{1.1.4}$$

where h is an applied external field. Quoting [1], the susceptibility of a variable X with respect to Y measures "the strength of the response X to changes in Y." In practice, however, we will not use (with perhaps the exception of the magnetization m) the definitions enumerated in (1.1.4) in numerical calculations–our task is to simulate the lattice system via the monte-carlo method to estimate equilibrium expectation values of the variables (1.1.4). The problem of how to measure observables alongside with the error estimates will be discussed in section 2.3.

1.2 Lattice Gauge Theories

There is an important property of the $\mathbb{Z}_2$ lattice gauge theory (LGT) Hamiltonian that is not present in that of the Ising model. To see this, we must investigate the symmetries of the 3D Ising and the 3D $\mathbb{Z}_2$ lattice gauge Hamiltonians–transformations that leave $\mathcal{H}$ invariant. It should be clear that the Ising model has a global symmetry by which we invert each and every site of the lattice, $s_i \mapsto -s_i$, as $s_i s_j \mapsto (-s_i)(-s_j) = s_i s_j$ for the nearest-neighbor interaction terms. We call the transformation $s_i \to -s_i$ a $\mathbb{Z}_2$ symmetry, referring to the 2-element discrete group $\mathbb{Z}_2$ that acts on the cube lattice (via the one-dimensional representation $\{1, -1\}$) as $s_i \mapsto -s_i$.

It is manifest that the $\mathbb{Z}_2$ LGT also has a global $\mathbb{Z}_2$ symmetry $U_{ij} \to -U_{ij}$ that acts on each and every bond of the cube lattice. However, as mentioned in [9], the LGT Hamiltonian has a local symmetry for which the Hamiltonian is left invariant if all bonds emanating from a single vertex are inverted. As is the case for the $\mathbb{Z}_2$ LGT, a theory with a Hamiltonian (or a Lagrangian, in QFT) that is invariant under *local* transformations is called a *gauge theory*.

As an aside, we may extend our $\mathbb{Z}_2$ lattice gauge theory to a general case where at each bonds of the lattice we assign a element of a representation of a group G. For example, if G is the lie group $U(1)$, we assign $U_{ij} = e^{iA_{ij}} \in \mathbb{C}^*$ to each bond on the lattice, where $\mathbb{C}^*$ is the group of invertible complex numbers. This defines a lattice equivalent of electromagnetism, a continuum $U(1)$ gauge theory. We may also choose to assign to each bonds an element from a representation of a non-abelian group such as $SU(3)$, the group of 3×3 special unitary matrices. Such a theory is called a Yang-Mills or non-abelian gauge theory, and has been successful in describing the strong and weak interactions of the Standard Model of Particle Physics. Numerical simulations of such generalized lattice gauge theories are used extensively to study non-perturbative regimes of quantum chromodynamics (QCD), as written in [3].

The terms gauge invariance and gauge symmetry are, of course, not limited to discretized lattice models, and appear quite often in various different physical contexts such as electromagnetism, general relativity, and quantum field theory. Yet what do we mean precisely by a gauge symmetry? One understanding of gauge symmetries is provided in [20] by Witten, where gauge symmetries are said to be "redundancies in the mathematical description of a physical system rather than properties of the system itself." A theory with gauge symmetry is a redundant description of a physical system in the sense that the work done by the electric field (a physical observable) does not depend on the choice of a reference potential, or that the curvature of spacetime (again, an observable) is independent of our choice of spacetime coordinates. David Tong also gives an intuitive description of gauge symmetries in the introduction of [16], from which we quote: "[gauge symmetry] is not a property of Nature, but rather a property of how we choose to describe Nature."

Hence, as in the case for electromagnetism and gravitation, it is then manifest (or rather tautologically true) that any physical observable must necessarily be gauge invariant. This presents an additional complexity in defining which quantities are physical observables when studying gauge theories, even in the simpler, discretized model of the $\mathbb{Z}_2$ LGT. Observables such as the magnetization or the correlation length that were well-defined in the 3D ising model cannot be naively extended (for example, by defining the "magnetization" as the sum over all spin variable located on the bonds) to the 3D $\mathbb{Z}_2$ LGT due to gauge symmetry. For the $\mathbb{Z}_2$ lattice gauge theory, the free energy E or the gauge invariant correlation functions known as *Wegner-Wilson loops* are examples of gauge-invariant operators of which we can measure the expectations at equilibrium.

1.3 Phase Transitions in Theories with Local Gauge Symmetry

Although the additional local gauge symmetry for the 3D $\mathbb{Z}_2$ LGT presents us with an added complexity in determining observables, one immediately obtains a gauge invariant observable for free; by definition, the free energy of the system is a gauge-invariant observable:

$$-\beta\mathcal{H} = K\sum_{\square} UUUU, \quad E \equiv -\sum_{\square} UUUU. \tag{1.3.1}$$

We can then measure the specific heat from the fluctuations in the free energy, and the divergence in the specific heat will locate the critical point of the 3D $\mathbb{Z}_2$ LGT. The behavior of the specific heat C will be our probe for locating the critical point during the numerical simulations of 3D LGT.

Another qualitative difference between the Ising model and the 3D LGT is the presence and absence of a local order parameter, as written in [9]. From Kardar [9], in the 2D and 3D Ising model, the value of the magnetization m defined by

$$m(T) = \frac{1}{V}\lim_{h\to 0} M(h,T) \tag{1.3.2}$$

The two different phases of the ising model was then characterized by the order parameter m, with m having two degenerate ground states $m = \pm 1$ for low temperatures. Here, as we began from a disordered state $T \gg 1$ to an ordered state, the magnetization chooses either of the configurations $m = 1$ or $m = -1$, and the symmetry that was present in the disordered phase ($\langle m\rangle = 0$) was spontaneously broken in the low temperature (ordered) phase. The properties of the phase transition such as critical exponents and divergence in correlation length was then characterized by an analysis of the Landau-Ginzburg theory, using the coarse-grained order parameter $m(\vec{x})$ that characterized the phase transition. The Ising model is an example of a theory with an order parameter m which undergoes spontaneous symmetry breaking.

Yet by Elitzur's Theorem [6], any Hamiltonian with a local gauge symmetry such as the $\mathbb{Z}_2$ LGT cannot undergo spontaneous symmetry breaking. Since we know, from the duality relations, that a phase transition of the 3D Ising model necessarily entails a phase transition in the 3D lattice gauge partition function, we are left with a conundrum–a phase transition without a local order parameter such as the coarse-grained magnetization m. With some thought, one may able to convince oneself that any local order parameter arising from a coarse-graining scheme, such as the magnetization in the Landau-Ginzburg theory, cannot have a counterpart in theories with local gauge symmetry, as we may change the local order parameter freely without changing the physical properties of the system. As a solution, which is described in Kardar [9], Wegner [19] suggested that the two phases in a theory with local gauge symmetry (such as the 3D LGT) is differentiated by the asymptotic behavior of gauge-invariant correlation functions known as the *Wegner-Wilson* loops:

$$W = \prod_{U_{ij}\in C} U_{ij}, \tag{1.3.3}$$

where C is a closed loop and W is the product of bonds over the closed loop C. The expectation of W is then given by:

$$\langle W \rangle = \frac{\sum_{\{U_{ij}=\pm 1\}} \prod_{U_{ij} \in C} U_{ij} \exp\left[\tilde{K} \sum_{\square} UUUU\right]}{\sum_{\{U_{ij}=\pm 1\}} \exp\left[\tilde{K} \sum_{\square} UUUU\right]}. \tag{1.3.4}$$

Under the high temperature expansion, we have (rephrasing from Kardar [9] p. 132):

$$\ln \langle W \rangle \propto \text{Area of } C, \tag{1.3.5}$$

while under the low-temperature expansion, we have:

$$\ln \langle W \rangle = -2e^{-8\tilde{K}} \cdot (\text{Perimeter of } C) + \cdots . \tag{1.3.6}$$

The two asymptotic relations of $\langle W \rangle$ are known as *the area law* and *the perimeter law*. Since a site on the loop shares exactly two bonds with the loop, the above operator is manifestly gauge invariant. Later, we will observe the perimeter law using the monte-carlo simulation of 3D $\mathbb{Z}_2$ lattice gauge theory in section 3.

2 Monte Carlo Simulation Methods

Before we discuss the numerical results of our simulations, we will explain in detail the process from extracting statistically valid measurements from monte-carlo generated data. The first step in simulating lattice models such as the Ising or lattice gauge theory is to generate equilibrium states that obey the Boltzmann probability distribution for a given temperature.

2.1 Update Rules

From statistical mechanics, the expectation value of some observable $\mathcal{O}$, is given by [9]:

$$\langle \mathcal{O} \rangle = \sum_{\{\underline{s}\}} \mathcal{O}(\underline{s}) \exp\left[-\beta \mathcal{H}(\underline{s})\right]. \tag{2.1.1}$$

where the sum is over all configurations $\underline{s}$ of the system in concern. However, in most cases it is not practical to measure $\langle \mathcal{O} \rangle$ via the above exact method, as the number of configurations N of $\{\underline{s}\}$ increases exponentially with the system size. Hence, the whole purpose of the Monte-Carlo method is to measure observables from a sample of representative configurations $\{s_i\}$ of a system in equilibrium. More precisely, as in [9], we have

$$\langle \mathcal{O} \rangle \approx \overline{\mathcal{O}} = \frac{1}{n} \sum_{i=1}^{n} \mathcal{O}(s_i). \tag{2.1.2}$$

The fact that the sample estimate $\overline{\mathcal{O}}$ is a good estimate of the expectation value $\langle \mathcal{O} \rangle$ comes from the *central limit theorem* in probability theory. Note that by using the central limit theorem we

are assuming that our each measurement of $\mathcal{O}(s_i)$ from the samples $\{s_i\}$ are independent and identically distributed. The problem of statistical independence among sample measurements from monte-carlo simulations is a subtlety and will be addressed in a later section on measurements.

Hence, to measure observables using (2.1.2), we must have a method to generate configurations of our system of interest that are correctly weighted by the Boltzmann probability distribution given by (2.1.1). The two popular methods in simulating lattice theories are the Metropolis-Hastings algorithm and the Wolff cluster update algorithm, which we present here briefly, along with the C++ implementations.

2.1.1 The Metropolis-Hastings Algorithm

The Metropolis-Hastings Algorithm is a popular single-site update method used in monte-carlo simulations of lattice models. A single instance of a Metropolis update on a lattice with spin variables $s_i = \pm 1$ proceeds as follows:

1. Select a random site on the lattice.
2. Compute ΔE, the energy required to flip the spin variable.
3. If $E_f - E_i \equiv \Delta E < 0$, then we accept the spin flip; if $\Delta E > 0$, we generate a random number $r \in (0, 1)$ and accept the spin flip if $r < e^{-K\Delta E}$.
4. Repeat 1-3 for some given number of iterations that exceed the relaxation time.

Note that we do not know beforehand, in principle, the relaxation time of a given monte-carlo simulation for fixed temperature. Hence, (4) implies that if the relaxation time was determined (using the methods of (2.3.1)) to be greater than the full monte-carlo run, it is necessary to gather more monte-carlo history data for sensible measurement of observables.

There is some ambiguity in the definition of a single monte-carlo *sweep* among different implementations of the Metropolis algorithm. Newman and Barkema [1] and Jensen [8] defines one *sweep* as randomly choosing a site of the lattice and performing the Metropolis update N (the total number of spin sites) times, while Landau and Binder [10] defines it as performing the metropolis update once for every spin on the lattice (without random selection). However, it is easy to see that the difference is not in principle but due to how we define a single sweep, as both methods are nothing other than successive iterations of a single metropolis update, which have been demonstrated (for example, in [1]) to satisfy the mathematical constraints required for a correct simulation of the Boltzmann distribution.

Note that it is not necessary (and computationally inefficient) to compute the exponential weight $e^{-K\Delta E}$ at every instance of Metropolis update. In both the Ising model and $\mathbb{Z}_2$ LGT, all possible weights can be computed before a full simulation at a given temperature. For example, the 2D Ising model has 5 possible values for the acceptance weights: $\Delta E = -8, -4, 0, 4, 8$. For the 3D Ising model, we have $\Delta E = -12, -8, -4, 0, 4, 8, 12$. The same process may be done in the $\mathbb{Z}_2$ LGT case by considering the plaquettes (4 in 3D, 6 in 4D) adjacent to given bond. We then compute the values $e^{-12K}, e^{-8K}, e^{-4K}$, and store them inside a hash-table (or a hash-map) data structure such as dictionaries in Python or maps in C++. Details of the implementation are given in the C++ function `metropolis`. As the Metropolis algorithm is simple and easily generalizable to the $\mathbb{Z}_2$ LGT, we will mostly use the Metropolis update method to generate our monte-carlo history data.

2.2 Random Number Generator

Numerical algorithms used to simulate thermal systems rely heavily on random number generation to simulate thermal fluctuations, and often a poor choice of random number generator (RNG) may result in periodic behavior in the thermal fluctuations that may render any measurement of observables meaningless. In order to extract sensible results from monte-carlo methods, it is essential that the period of the random number generator used in simulation is longer than the full extent of the monte-carlo simulation. The most popular random number generator used in monte-carlo simulations is the Mersenne Twister (usually called MT19937) algorithm developed by Makoto Matsumoto and Takuji Nishimura [11], which we explain here briefly. Quoting [11], the MT19937 algorithm has a period of $2^{19937} - 1$ with an 623-dimensional equi-distribution up to 32 bits of accuracy.

In generating the monte-carlo history files (from the Metropolis algorithm), we used the MT19937 random number generator, which is included in the C++ Standard Template Library (STL).

2.3 Measurement of Observables

The ultimate goal for using the monte carlo simulation method for analyzing lattice theories is to calculate physical observables that may be of interest. These include physical observables such as magnetization, heat capacity, susceptibility, and correlation lengths, for example. In this section, we describe the route from generating monte carlo history data to calculating the magnetization, heat capacity, and other interesting quantities such as critical exponents.

To begin our discussion on measurements, we will derive how to perform measurements of observables defined in (1.1.4) using monte-carlo data. The expectation of the magnetization $\langle m \rangle$ is given by

$$\langle m \rangle = \frac{1}{N_s} \left\langle \sum_{i=1}^{N_s} s_i \right\rangle, \tag{2.3.1}$$

where N_s is the total number of sites on the lattice. The magnetization is an order-parameter for the 3D ising model, and we may apply equation (2.3.1) directly in computing the monte-carlo calculation of the magnetization (for a single step). The free energy per site may be computed in the same way:

$$\langle E \rangle = -\frac{1}{N_s} \left\langle \sum_{\langle i,j \rangle} s_i s_j \right\rangle, \tag{2.3.2}$$

where now the sum is done over all nearest-neighbor pairs. As a convention, we will consider the *negative* of the free energy per site given in equation (2.3.2), so that our measurements will be positive, and the negative of the free energy will be maximized at equilibrium.

The monte-carlo evaluation of the *fluctuations* such as the susceptibility and the specific heat requires some thought, however, although not too complicated. The derivation is a straightforward application of the Boltzmann probability distribution, and is illustrated in detail by Newman and Barkema in [1]. The expectation of C and χ are

$$\langle c \rangle = K^2 N_s [\langle E^2 \rangle - \langle E \rangle^2], \tag{2.3.3}$$

$$\langle \chi \rangle = K N_s [\langle m^2 \rangle - \langle m \rangle^2]. \tag{2.3.4}$$

Note that we have additional factors of N_s, since we have defined E and m as per-site variables. Using the above definitions, we can calculate observables for each step in our monte-carlo simulation. However, to extract actual estimates at each temperature, additional details must be accounted for, which we explain in the following sections.

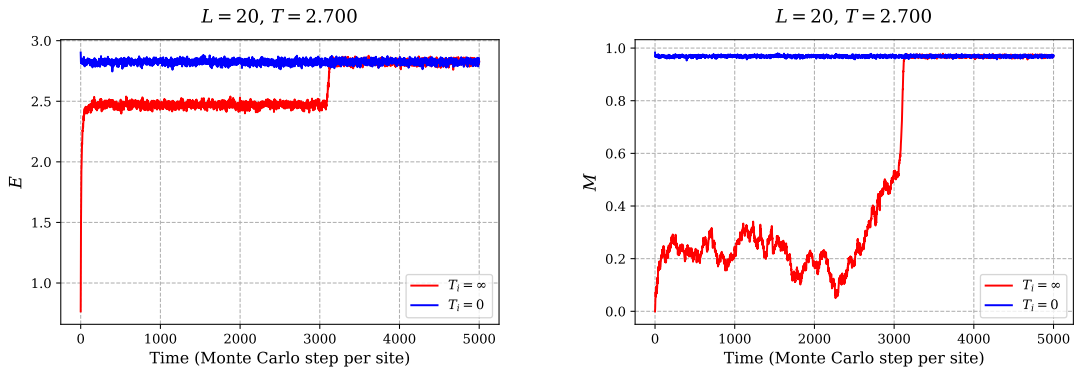
2.3.1 Relaxation Time and the Problem of Local Optima

Measurements of observables such as magnetization, energy, and specific heat are only defined at *thermal equilibrium*, or as David Tong illustrates in [17], when "the probability distribution is independent of time which ensures that the expectation values of the macroscopic observables are also time independent." This will also be true of monte-carlo simulations, where we require that our measurements be performed only after a sufficient number of monte-carlo steps have been performed to ensure that the system has reached equilibrium. We will define the number of monte-carlo steps needed for a system to reach its thermal equilibrium at a given temperature as the *relaxation time* or the *relaxation time*.

As discussed in Newman and Barkema [1], the best (or the most practical) way to judge whether a given monte-carlo simulation run has reached equilibrium is to plot the history of an observable such as E and look for a time t_0 for which the expectation value of E stays reasonably fixed. However, it is possible that a given monte-carlo run relaxes to a bad local optimum in which the expectation value of E seems time-independent while in truth there exists a global optimum that characterizes the true thermal equilibrium for a system in a given temperature. The existence of such local optima has been observed in the monte-carlo simulation of the 3D Ising model, which we present in Figure 1. To detect false equilibration to local optima, we must perform out monte-carlo run for different initial conditions, for example, as Newman and Barkema [1] suggests, the completely ordered state $s_i = 1$ for all i, and the completely disordered state $s_i = \pm 1$ randomly with equal probabilities. We then observe whether some observable such as the free energy relaxes to the same expectation value for the two different initializations of our lattice. Ideally, it is desirable to perform the simulation for different random number generator seeds and multiple initial conditions; however, in practice two initial conditions will suffice for our purposes. One may notice from Figure 1 the idiosyncratic behavior of the order parameter M during false relaxation to a local optima–yet investigations into such behavior is a topic for a different project.

For the example of Figure 1, a reasonable choice for equilibration time would be $t \approx 3500$, and we would proceed to measure observables using only those data that were gathered after $t = 3500$. In theory, however, one cannot state in complete certainty that the region where both curves (red and blue) converge to the same expectation value of E is not another local optima that the monte-carlo runs happened to encounter. A practical way to circumvent this issue is to inspect the relaxation plots for other monte-carlo runs with similar temperatures, and seeing if the observed plateau has a distinctly different expectation value of E than the other monte-carlo runs that were simulated with similar temperatures.

Note the we have somewhat implicitly assumed that the correct physical observable that we should use to determine sufficient relaxation was the free energy E, rather than some other variables such as the magnetization m. To justify this, recall that for an isolated system, the driving force towards thermodynamic equilibrium is *entropy*. In other words, the system will evolve in time so as to maximize its entropy. However, for the case of monte-carlo simulation of lattice theories on a fixed temperature T (such as the Ising or the Wolff algorithm), we are in effect assuming the existence of an infinite heat-capacity reservoir kept at constant temperature T, and dynamics



(a) Free energy E.

(b) The magnetization (per site) M.

Figure 1: An example of false equilibration to a local optima (3D Ising).

towards equilibrium is governed not by entropy but the Helmholtz free energy (as explained, for example, by Daniel Schroeder in [18]). In short, the correct observable that characterizes thermal equilibrium is the free energy E, and thermal equilibrium is hence *defined* by the minimization of E.

Yet again, the previous narrative on determining relaxation times assumes the uniqueness of the equilibrium value of E, for example, for any given configuration of initial conditions. For the 4D $\mathbb{Z}_2$ lattice gauge theory, however, we will demonstrate that at the critical point $K_c = \frac{1}{2}\ln(1+\sqrt{2})$ (said to be known analytically in [3]), the expectations $\langle E \rangle$ obtained by relaxing the system from a completely ordered and disordered state remains different (or, as Michael Cruetz states "[the two expectations] do not converge to each other in an observable time" [3]). In this case, the difference in expectations values of E are attributable to a metastable state of the system, characteristic of the (first-order) phase transition in 4D $\mathbb{Z}_2$ LGT.

2.3.2 Correlation Time and Critical Slowing Down

The increase in relaxation time near the critical point is not an exclusive attribute of first-order transitions, however, and is related to a phenomenon as *critical slowing down*. To see how this happens, we summarize the discussion in Chapter 9.2 in Kardar [9]. Consider the Landau-Ginzburg Hamiltonian:

$$\mathcal{H}[\vec{m}] = \int d^d x \left[\frac{r}{2}\vec{m}^2 + u\vec{m}^4 + \frac{K}{2}(\vec{\nabla}\vec{m})^2 + \cdots \right], \tag{2.3.5}$$

where, under the limit of $V \to \infty$ and the saddle-point approximation, we argued that as $r \to 0$ the system undergoes a phase transition. Here, however, we are interested in the *dynamics* of the magnetization $\vec{m}$, that is, the time-dependent behavior of $\vec{m}$ rather than its equilibrium behavior. The Langevin equation for the Landau-Ginzburg Hamiltonian is then given as

$$\frac{\partial \vec{m}}{\partial t} = -\mu r \vec{m} - 2\mu u \vec{m}^2 \vec{m} + \mu K \nabla^2 \vec{m} + \eta(x,t). \tag{2.3.6}$$

where η is a brownian random force driving the fluctuations of $\vec{m}$. Starting from a disordered state, since $t > 0$ we may assume that $u \approx 0$ (close to the critical point) and examine the time-dependent

behavior as we approach the phase transition from the disordered state. The Fourier modes $\vec{m}(\vec{q},t)$ for $\vec{m}(\vec{x},t)$ may be obtained as

$$\vec{m}(\vec{q},t) = \vec{m}(\vec{q},0)e^{-\gamma(q)t} + \int_0^t d\tau\ e^{-\gamma(q)(t-\tau)}\eta(\vec{q},t). \tag{2.3.7}$$

The Fourier modes of $\vec{m}$ contains information about the fluctuations of $\vec{m}(\vec{x},t)$, and the time-dependent expectation value of the modes $\vec{m}(\vec{q},t)$ is given as

$$\langle\vec{m}(\vec{q},t)\rangle = \vec{m}(\vec{q},0)e^{-t/\tau(q)}, \tag{2.3.8}$$

where the time constant $\tau(q)$ of the decay of each mode is given as

$$\tau(q) = \frac{1}{\mu(r+Kq^2)}. \tag{2.3.9}$$

The above equation for gives the relaxation time $\tau(q)$ for each Fourier mode q of the fluctuations in $\vec{m}$. Note that since η is a Gaussian noise with mean $\langle\eta(\vec{x},t)\rangle = 0$, we have $\langle\eta(\vec{q},t)\rangle = 0$ and similarly the expectation of the integral term of equation (2.3.7) also evaluates to 0. As we approach the critical point, the fluctuations in m happen at all length scales as the correlation length ξ diverges. Then for fluctuations over length scales larger than some large ξ such that $q \ll 1$ in frequency space, the relaxation time is given by $\tau(q) \sim 1/(\mu r)$. On these length scales (which we may take it as the whole system size of our finite lattice), the relaxation time diverges as $r \to 0$ near the critical point. This is the phenomenon known as *critical slowing down*, and the critical point is characterized by divergences in both length (ξ) and time (τ) scales.

Of course, in monte-carlo simulations, the correlation length and times will not diverge to infinity but attain a peak near the critical point. This is problematic in extracting sensible measurements from monte-carlo data, as the divergence in the relaxation time affects how many measurements from our monte-carlo simulation may be regarded as independent measurements. The formula for measuring any observable from monte-carlo data is contingent on the hypotheses of the central limit theorem (2.1.2), and divergences in the relaxation and correlation times for simulations near the critical point will render any measurement of observables meaningless if the extent of the simulation becomes shorter then the correlation length.

In practice, we calculate the correlation time of a stochastic time series as follows. Quoting Sokal [15], the *un-normalized autocorrelation function* for a real valued stationary stochastic process with mean $\langle f_t\rangle = \mu$ is given by:

$$C(t) \equiv \langle f_{s+t}f_t\rangle - \mu^2. \tag{2.3.10}$$

Whereas the correlation length ξ gives the estimate of how two points separated in space are correlation, the autocorrelation function $C(t)$ measures the correlation between two events at times $t' = s$ and $t' = s+t$, as stated in [1]. Intuitively, we expect the autocorrelation function $C(t,t_0)$ to be larger for t close to t_0 (with t_0 being the reference time for which we measure the correlations from), and decay to zero for sufficiently large values of t, since events separated by long time scales (of course, away from the critical point) will be independent. From [1], we expect the autocorrelation function to decay exponentially at long times: $C(t) \sim e^{-t/\tau}$. For our purposes, we are interested in the *integrated correlation time* τ:

$$\tau \equiv \int_0^\infty \frac{C(t)}{C(0)} \sim \int_0^\infty e^{-t/\tau}\ dt. \tag{2.3.11}$$

In Sokal's notes, we define the *normalized autocorrelation function* $\rho(t) = C(t)/C(0)$. For a discrete stochastic process such as our monte-carlo simulation, we estimate $C(t)$ as in equation (3.21) of [1]:

$$\hat{C}_m(t) = \frac{1}{t_{max} - t} \sum_{t'=0}^{t_{max}-t} m(t')m(t'+t) - \frac{1}{t_{max} - t} \sum_{t'=0}^{t_{max}-t} \frac{m(t')}{t_{max} - t} \sum_{t'=0}^{t_{max}-t} m(t'+t), \tag{2.3.12}$$

where m is an observable (such as the magnetization m) that we desire to analyze its autocorrelation, and t_{max} is the total extent of the stochastic process. Equation (2.3.12) gives a formula for the estimator of $C(t)$ when the exact mean μ is not known, as Sokal mentions in [15]. The unbiased estimator for the integrated correlation time τ is then given by Sokal as:

$$\hat{\tau} = \sum_{t=0}^{t_{max}} \frac{\hat{C}(t)}{\hat{C}(0)} \tag{2.3.13}$$

Note that in Sokal's notes, there is an extra factor of $\frac{1}{2}$ for the estimator for τ. This is due to the fact that Sokal is measuring the autocorrelation function over a symmetric interval. Since we are interested in computing correlations between events *after* relaxation from the initial state, if t is well beyond the relaxation time the two expressions will give the same results (if summed over some limiting value of $|t| < M$), as by definition of thermal equilibrium the correlations will be symmetric with respect to time.

In practice, for long times t' from our reference time t, the sample autocorrelations $\rho(t')$ will mostly be noise and we would like to truncate the sum in (2.3.13) after some window M. A detailed discussion of this issue is again given in Sokal's notes [15]. Hence, our estimator for τ is given as:

$$\hat{\tau} = \sum_{t=0}^{t_{max}} \hat{\rho}(t)\lambda(t), \tag{2.3.14}$$

where $\lambda(t)$ is a dynamic window defined by:

$$\lambda(t) = \begin{cases} 1 & \text{if } |t| < M, \\ 0 & \text{if } |t| < M, \end{cases} \tag{2.3.15}$$

and M is "the smallest integer M such that $M \geq c \cdot \hat{\tau}(M)$" [15], where c is some constant, which Sokal suggests using $c \approx 4, 10$. We have chosen $c = 5$ for our computation of the integrated correlation times. The variance in $\hat{\tau}$ is estimated as

$$\sigma^2(\hat{\tau}) = \frac{(2M+1)}{n}\hat{\tau}^2. \tag{2.3.16}$$

We will use equation (2.3.14) and (2.3.16) to compute the estimate and error in the integrated correlation times of the 3D Ising model and perform a rough analysis of the effects of diverging correlation times. Although we ensure that our monte-carlo history is sufficiently long so that multiple statistically independent samples could be drawn for a given integrated autocorrelation time, the majority of our calculations will use a more practical error estimation method known as *bootstrap resampling*.

2.3.3 Statistical Error Analysis

Once the integrated correlation time τ is known, we can estimate the uncertainties of our estimates $\overline{\mathcal{O}}$ for any observable $\mathcal{O}$ obtained from monte-carlo simulation. The expression for the standard error σ of our estimates $\overline{\mathcal{O}}$ is given in [1] equation (3.38) as

$$\sigma = \sqrt{\frac{1 + 2\tau/\Delta t}{n-1}(\overline{\mathcal{O}^2} - \overline{\mathcal{O}}^2)} \tag{2.3.17}$$

where $\overline{\mathcal{O}}$ and $\overline{\mathcal{O}}$ are our measurements from a single monte-carlo sweep, of which we have gathered for n sweeps in total (of course, after omitting events before relaxation from initial configuration).

However, in many cases, direct computation of the statistical errors via equation (2.3.17) is either impossible or unfeasible, and we must resort to other techniques for estimating the statistical errors of our measurements. This is particularly true for quantities such as the specific heat c or the susceptibility χ, since estimates of fluctuations does not make sense for a single value but must be averaged over some sample of the measured observables, as explained in detail in [1]. Our method of choice for estimating statistical errors is the *bootstrap resampling* method, given in [1], which we illustrate here briefly. From the whole history of our monte-carlo simulation data, we random sample n distinct data points. We then calculate the estimator for quantities such as the specific heat $\hat{c}$ from these n samples, and append the value to a list. We repeat many times (say N), allowing sampling with replacement and adding our estimates of c (or χ) to the list. Then quoting [1], from our list of estimators $l = \{\hat{c}_1, ..., \hat{c}_N\}$, the mean and standard deviation of the distribution l is an estimator for the c and the error in the value of c:

$$c_{est} = \sum_{i=1}^{N} \frac{\hat{c}_i}{N}, \quad \sigma_{c_{est}} = \sqrt{\overline{c^2} - \overline{c}^2}. \tag{2.3.18}$$

Note that, as Newman indicates, the standard error $\sigma_{c_{est}}$ must not contain any dependence on N, the number of $n-$sampling repetitions. We will use the bootstrap method to estimate errors for all of our measurements, due to its simplicity, making sure that for each monte-carlo run at a given temperature, we achieve relaxation from initial conditions. Also, note that even for the bootstrap resampling method, we need an estimate of the integrated correlation times to decide on the number of measurements n for each repeated resamplings from the full monte-carlo history, since the full monte-carlo time must be greater than our estimate of the integrated correlation time τ.

2.4 Determination of Critical Temperature and Exponents

In continuous phase transition such as the Ising model, we are often interested in the behavior of thermodynamic quantities near the critical point, which are captured by *critical exponents*, defined as:

$$M \propto |T - T_c|^{\beta}, \quad \chi \propto |T - T_c|^{-\gamma}, \tag{2.4.1}$$

$$C_V \propto |T - T_c|^{-\alpha}, \quad \xi \propto |T - T_c|^{-\nu}. \tag{2.4.2}$$

as in [13]. These critical exponents are universal in the sense that models belonging to the same universality class will share the same critical exponents. Universality is perhaps most vividly demonstrated by the theory of renormalization group flow, where the position of the critical fixed

point may depend on microscopic parameters while the critical exponents were determined by the eigenvalues of the RG flow near the fixed point, as in [9].

However, as discussed in class, the fixed point of the renormalization group flow corresponding to the observed 3D Ising phase transition is not the Gaussian fixed point (giving mean-field theory exponents) but the fixed point known as *the Wilson-Fisher fixed point*. The critical exponent near the Wilson-Fisher fixed point is known as the *Ising critical exponents*, which we measure in section 3.

While the critical temperature T_c and divergences of observables are features of infinite-lattice systems, in monte-carlo simulations we always work with a finite number of lattice sites. Hence, for difference lattice sizes L of our simulations, the location of T_c will be different, and the various divergences of thermodynamic quantities will be cutoff due to finite size effects. As in [13], we define the pseudo-critical point for a lattice with linear size L as the value of K at which the system reaches the maximum value of C or χ. We will then use a technique known as *finite size scaling* to estimate the actual location of T_c and the values for the exponents α, β, γ, etc. Quoting [13], the procedure for locating T_c and the critical exponent associated with, for example, the susceptibility χ is given by:

1. Using various lattice sizes L, we locate the maximum value of χ for each lattice size L.
2. We then perform a power-law fit:

$$\beta_{max} = \beta_c - c_1 L^x \tag{2.4.3}$$

   where β_{max} is the pseudo-critical point for the lattice–the value of β (in our notation, K) of which the maxmimum χ_{max} occurs for a given lattice size L, and $x \equiv -1/\nu$.
3. Once β_c is known, the exponent γ (for the case of susceptibility) may be estimated via a (different) power-law fit $\chi_{max} \propto L^{\gamma/\nu}$.

We may of course repeat the above analysis to other thermodynamic quantities such as the specific heat, and we will explicitly use C (the specific heat) to determine the critical point and the exponents ν and α. Note that the above analysis only applies to continuous phase transitions.

3 Results

Here we present our results for the monte-carlo simulation of the 3D Ising and 3D, 4D $\mathbb{Z}_2$ lattice gauge theory. As always, we will refer to *monte-carlo history* (MC history) as the raw data obtained from a full monte-carlo run at fixed temperature using the Metropolis algorithm. We then generate relaxation plots (E vs t) for each MC history to roughly determine relaxation times. Note that relaxation times may differ from run to run depending on the proximity to the critical point and the existence of a metastable state (local optimum). After dropping all rows before the relaxation time, we will use the bootstrap method to calculate observables, and extract critical exponents by finite size scaling relations. A conservative estimate of relaxation times for temperatures away from the critical point is around $t \approx 500$ monte-carlo time, although the relaxation times are larger for simulations near the critical point, as usual.

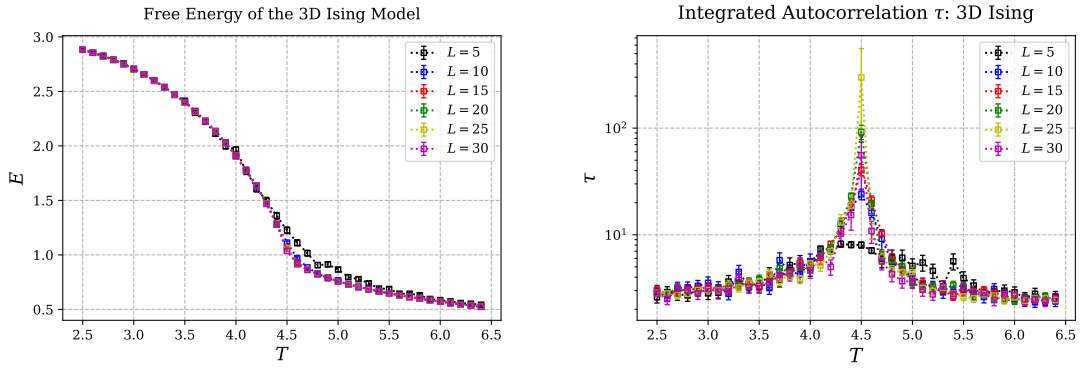
Each run consists of iterations greater (3000 10000) than the measured autocorrelations (which vary from $O(1) \sim O(100)$) to ensure sufficient statistical independence of monte-carlo data, and each data point (monte-carlo run at a given temperature) was checked for relaxation by starting

the simulation for two distinct initial conditions: $T = 0$ start where all sites are assigned $s_i = +1$ or $T = \infty$ where all sites are assigned $s_i = \pm 1$ with equal probability. We then selected one simulation (typically starting with $T = 0$) to generate measurements for each temperature, using the bootstrap method. If not stated otherwise, the integrated autocorrelation plots show values of τ calculated from using formula (2.3.12) using the variable E, the free energy, instead of m (magnetization per site) which is not defined for LGTs. [1]

Generating the monte-carlo history files was done via the C++ programming language due to its advantage in speed, and post-generation data analysis was done with the Python Language. All codes used to generate monte-carlo data and some auxiliary Python scripts used to generate plots are uploaded at Github.[2] To make sure that our monte-carlo runs are at least significantly greater than the integrated autocorrelation time τ, we include plots of the integrated autocorrelations for each data we have used, unless stated otherwise.

3.1 3D Ising Model

The 3D Ising model shows clear evidence of a continuous phase transition where $\log Z \propto E$ changes continuously as it crosses the critical point, as it could be seen from Figure 2. Each data point corresponds to a bootstrap estimate of an single monte-carlo history generated at a given temperature, consisting of at least 3000 sweeps.



(a) Free energy E.

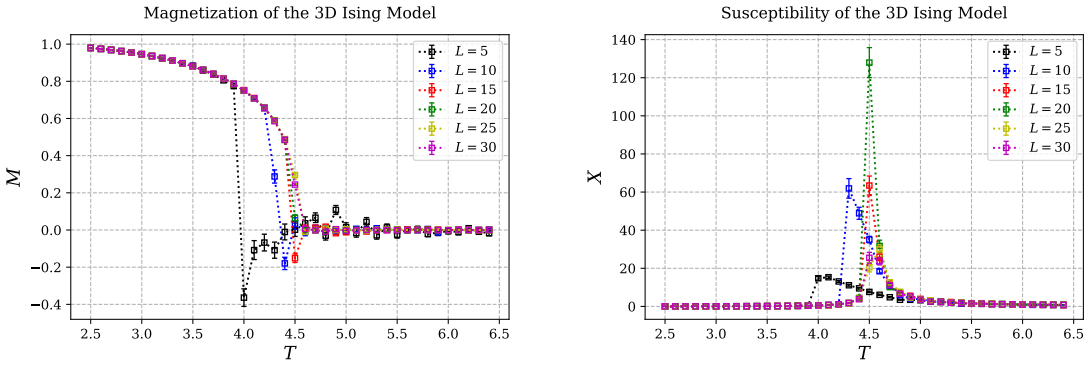
(b) The integrated autocorrelations τ_{int}.

Figure 2: The Ising model shows a second-order (continuous) phase transition around $T_c \approx 4.5$. The measured integrated autocorrelation for each monte-carlo run of (a) (τ_{int}) is given in the figure right. Note that the error-bars on (a), which are obtained via the bootstrap method, are included but too small to be visible.

The phase transition is clearly visible if one also examines the specific heat and the susceptibility.

[1] Due to time constraints, the autocorrelations were not measured for the fine-grid search (see Figure 4 (b)). It is possible and quite likely that the integrated autocorrelations around these range of temperature values are considerably greater than those measured in Figures 2 or 6. This may have introduced a systematic error in our measurements of K_c and the critical exponents ν, α. For each analysis, however, we will include our calculations of the integrated correlation times for reference.

[2] https://github.com/dkoh0207/Statistical-Mechanics



(a) The magnetization (per site) M.

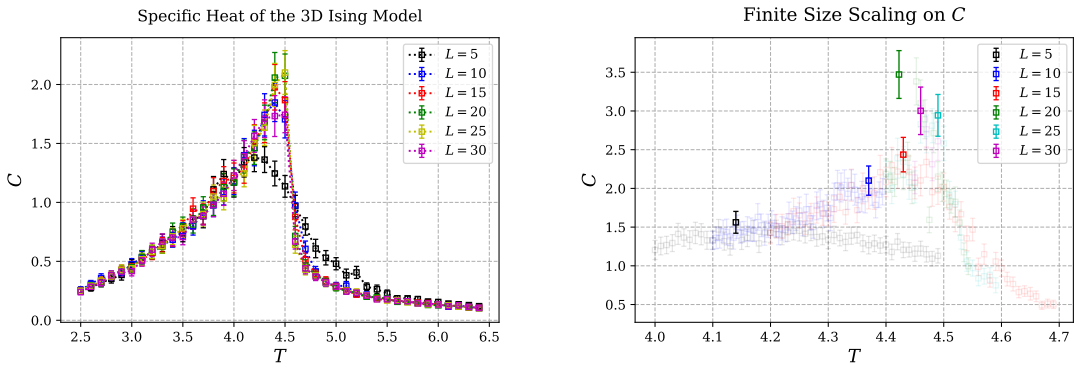
(b) The susceptibility χ.

Figure 3: Behavior of the magnetization M and the susceptibility χ.

It is possible to observe hints of finite size scaling effects for both C and χ. However, for a rigorous determination of the critical exponents one needs a finer-grid search near the critical point to locate the pseudo-critical points (the value of K at which χ or C attains a maximum) for varying lattice sizes L.[3]

We will compute the critical exponents ν and α explicitly using finite-size scaling of the specific heat. Note that although Figure 3 gives a good idea on the behavior of the 3D Ising system over a wide-range of temperatures, to extract information about critical exponents it is necessary to run the simulations with a finer-spacing near the critical point.

[3]Unfortunately, this is exactly the region where the Metropolis algorithm is highly inefficient, due to critical slowing down. Although by measuring the autocorrelation lengths we ensured that our simulation lengths were sufficiently longer than the integrated autocorrelation time, obtaining accurate estimates in this regime via the Metropolis method is quite difficult and time-consuming, and it is better to the Wolff algorithm or other cluster-update methods to extract measurements near the critical point.



(a) The specific heat C.

(b) Fine-grid search near T_c.

Figure 4: Pseudo-critical point search using C_{max} near T_c. The highlighted data points refer to C_{max}, the maximum value of C for each lattice size L. The faded data points refer to the entire grid-search over the region, for separate values of L. The total number of iterations ranged from at least 3000 to 5000.

Figure 4 shows a fine-step search near the critical region to determine the pseudo-critical points for each lattice size L. Note that theoretically the value of C_{max} must grow with increasing lattice size L and the value of the pseudo-critical point β_{max} converges to β_c as $L \to \infty$. This again renders the search for C_{max} increasingly difficult for larger lattice sizes near the critical point, since as L is increased one needs to vary T with a smaller step ΔT near the critical point.

Once the temperatures corresponding to C_{max} for each L has been determined, we use the scaling relation (2.4.3) to estimate the true critical point β_c. Figure 5 shows the least-squares fit of the scaling relation to the critical temperature and the specific heat. The error-bars in the inverse-temperatures (denoted K or β) are calculated as

$$\Delta K = \Delta\left(\frac{1}{T}\right) = \frac{\Delta T}{T^2}, \tag{3.1.1}$$

where ΔT is the spacing of our grid-search over the range of temperatures given in Figure 4. We will refer to the term *grid search* as the process of locating C_{max} in a given interval of T, usually near the critical point. The critical point $K_c \equiv \beta_c$ is estimated as:

$$K_c = 0.222 \pm 0.001 \tag{3.1.2}$$

The values we obtain for ν and α may be summarized as:

$$\nu = 0.6 \pm 0.1, \quad \alpha = 0.24 \pm 0.05. \tag{3.1.3}$$

The literature values of K_c, ν and α for the 3D Ising model is given in [7] and [5] (respectively) as:

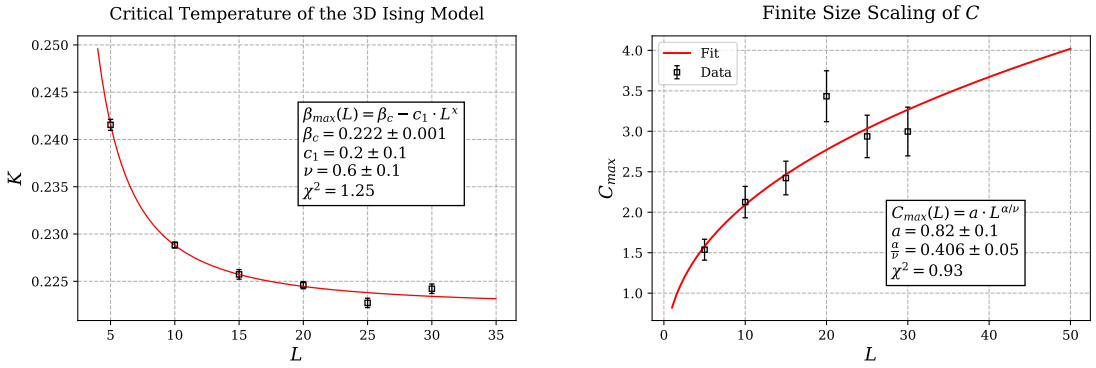
$$K_c^{lit} = 0.221655 \pm 0.000001 \tag{3.1.4}$$

$$\nu^{lit} = 0.625(1), \quad \alpha = 0.15 \pm 0.02 \tag{3.1.5}$$

where we have propagated the uncertainties if necessary, using the standard technique:

$$\Delta f(x_1, ..., x_n) = \sqrt{\left(\frac{\partial f}{\partial x_1}\Delta x_1\right)^2 + \cdots + \left(\frac{\partial f}{\partial x_n}\Delta x_n\right)^2}. \tag{3.1.6}$$

We see that our estimates of the exponents are around $\sim 2\sigma$ away from the accepted values, while we have good agreement with literature for K_c. The discrepancies in the exponents are most likely caused by systematic errors due to the inefficiencies of the Metropolis algorithm near the critical point (more precisely, the effective number of independent measurements near the critical point).



(a) Power law fit to the scaling relation (2.4.3).

(b) Estimating the critical exponent α.

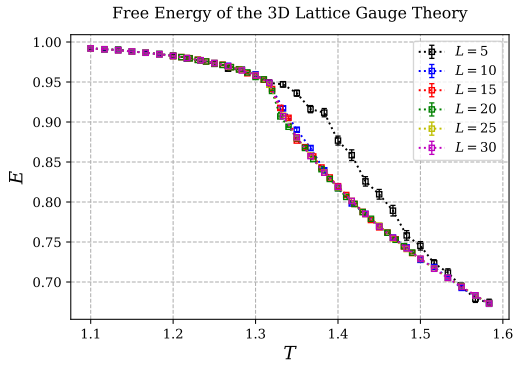
Figure 5: Using finite size scaling to determine K_c, ν, and α.

3.2 3D $\mathbb{Z}_2$ Lattice Gauge Theory

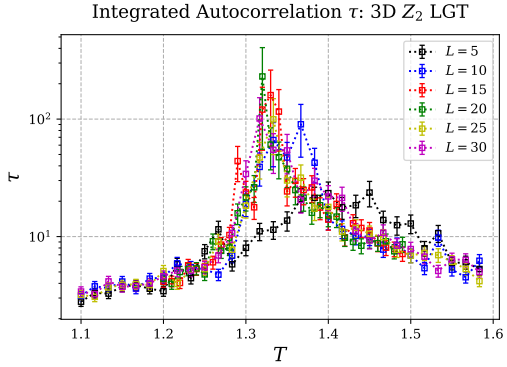
Solving the duality relation, we expect that the 3D Lattice Gauge Theory has a phase transition at $\tilde{K}_c$ defined by:

$$\tilde{K}_c = -\frac{1}{2}\ln(\tanh K_c) \approx 0.761 \implies \tilde{T}_c \approx 1.314. \tag{3.2.1}$$

where $K_c = 0.221655$ is the 3d Ising critical point. The duality relation also imply that the 3d $\mathbb{Z}_2$ lattice gauge theory has a continuous phase transition, albeit without any order parameters due to Elitzur's theorem. As we discussion in section 1, the two phases for theories with local gauge symmetry are differentiated by the behavior of Wilson loops. However, locating the critical point for the $\mathbb{Z}_2$ LGT may be done by employing the same finite-size scaling argument for the specific heat divergence near the critical point, as was done in the 3d Ising case. In Figure 6, we see clear evidence of a continuous phase transition near $T \approx 1.31$, marked by a divergence in integrated autocorrelations and the specific heat.



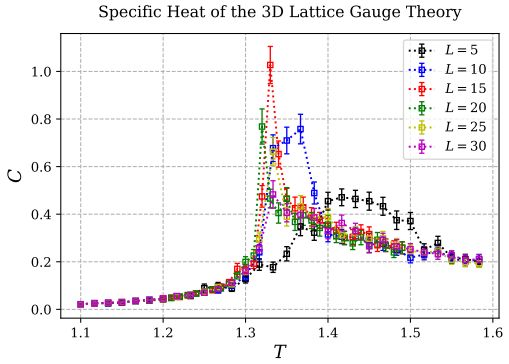
(a) Free energy per site for the 3D $\mathbb{Z}_2$ LGT.



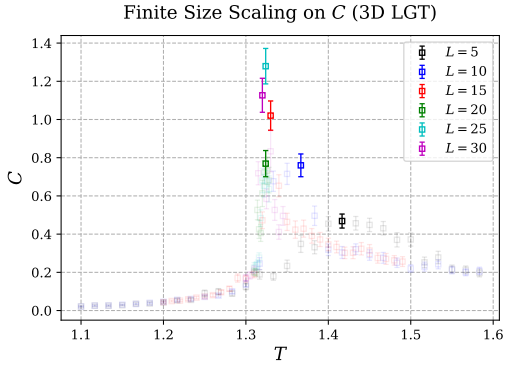
(b) Integrated auto-correlations for 3D $\mathbb{Z}_2$ LGT.

Figure 6: Measurements of the free energy and the integrated auto-correlations for the 3D $\mathbb{Z}_2$ LGT. Note that the discontinuity in the derivative of E is visible simply by observing the free energy plot of (a), which we can confirm in the next figure on the specific heat.

To employ finite-size scaling to locate the true critical point and the values of the exponents ν and α, we again analyze the behavior of C near the critical point. The relevant plots are provided in Figure 7.

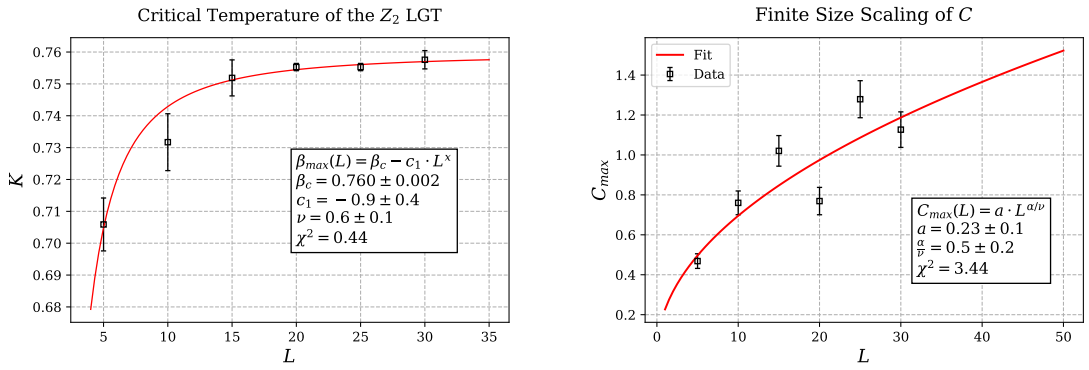


(a) Specific heat C for the 3D $\mathbb{Z}_2$ LGT.



(b) Fine-grid search of C_{max} near T_c.

Figure 7: Measurements of the specific heat and pseudo-critical points for the 3D $\mathbb{Z}_2$ LGT. Again, the highlighted points for Figure (b) (C_{max}) were used to estimate the true critical point K_c. For $L = 5, 10, 15$ we have used the same C_{max} obtained from plot (a), while for $L = 20, 25, 30$ a finer grid-search near K_c was performed.



(a) Determination of K_c and ν.

(b) Fine-grid search of C_{max} near T_c.

Figure 8: Power law fits to the finite-size scaling relations of (2.4.3).

In summary, using the power law fit statistics of 8, we estimate the critical point $\tilde{K}_c$ of the 3D $\mathbb{Z}_2$ lattice gauge theory as:

$$\tilde{K}_c = 0.760 \pm 0.002, \tag{3.2.2}$$

which agrees quite well with the value of $\tilde{K}_c$ computed via the duality relation (3.2.1) using the literature value of the 3D Ising critical point $K_c = 0.221655$. The exponent ν and α are given as

$$\nu = 0.6 \pm 0.1, \quad \alpha = 0.3 \pm 0.1, \tag{3.2.3}$$

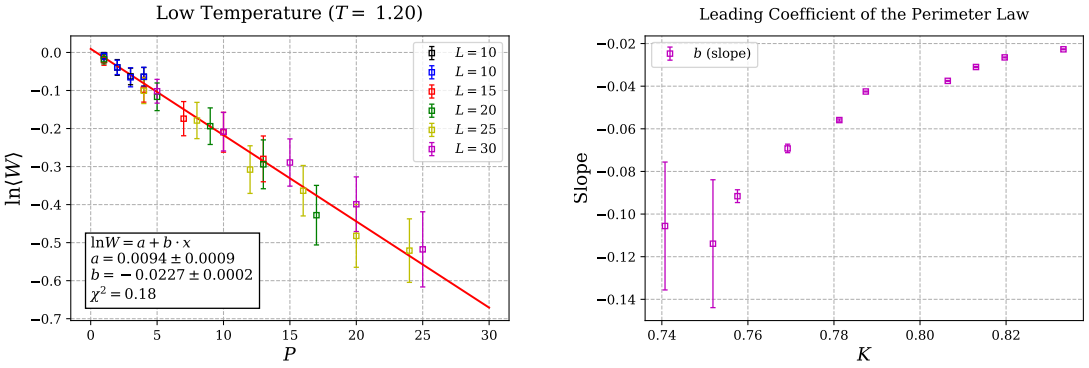
which is again roughly $\sim 2\sigma$ away from accepted values. The estimate of α obtained from the 3D Ising model and the $\mathbb{Z}_2$ lattice gauge theory (obtained from our monte-carlo run) agree with in 1σ yet differ from literature by around 2σ, which indicate sources of systematic errors, especially in estimating the values and errors of the specific heat C. We will discuss possible sources of systematic errors in the last section of the report.

Lastly, from Wegner [19], the two phases in theories with local gauge symmetry is described by the asymptotic behavior of gauge-invariant correlation functions known as *Wegner-Wilson loops*, as described in Chapter 7 in [9]. Unfortunately, although it is possible to see (in Figure 10) a distinct transition of the behavior of the Wegner-Wilson loops across the critical point, it is quite difficult to confirm the area law due to large statistical error obtained via the bootstrap method, most likely arising from high fluctuations of the Wegner-Wilson loops for $T > T_c$. Yet it is possible to confirm the low-temperature behavior of the Wilson loop given in [9] by:

$$\ln \langle W \rangle \approx -2e^{-8\tilde{K}} P_W \tag{3.2.4}$$

where $\langle W \rangle$ is the expectation value of the Wilson loop W computed via monte-carlo and P_W is the perimeter of the loop W. Computation of Wilson loops of various sizes in different lattice sizes are given in Figure 9. As it could be seen, in temperatures below the critical point, log of the expectation of the Wilson loops decreases linearly with the number of bonds on its side (1/4 of its perimeter, using square-shaped loops). Fitting a linear function to each plot generated at different temperatures, we can plot the value of the slope $b \approx -2e^{-8\tilde{K}}$ and see what happens near the critical point. In Figure 9, as $K \to K_c$ the perimeter coefficient defined by b decreases, with

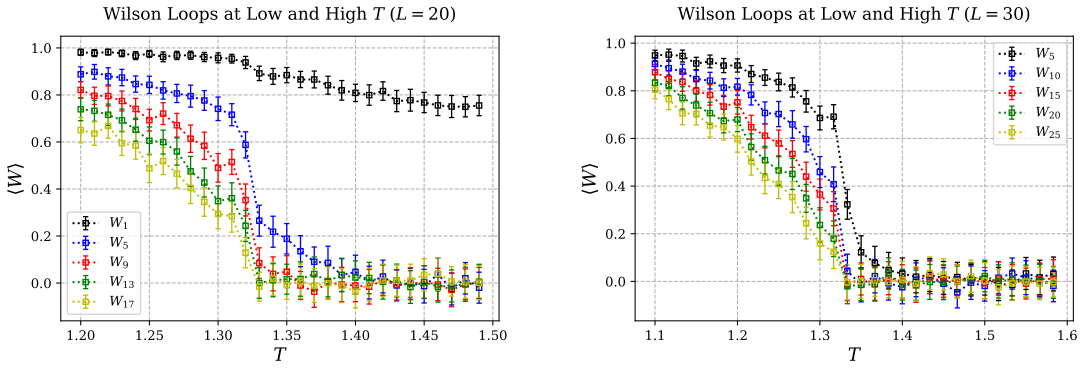
a hint of exponential behavior suggested from equation (3.2.4) for larger values of K. Near the critical point $K_c \approx 0.76$, we also demonstrate a jump in the statistical uncertainties obtained from the least-squares optimizing fit, which indicates that a linear model is a poor choice for the relation P vs. $\ln(W)$ for $K < K_c$. In any case, the overall low-temperature behavior predicted by equation (3.2.4) is consistent, at least at the qualitative level, with the simulated results shown in Figure 9.



(a) Perimeter law at low temperatures.

(b) The values of the slope b.

Figure 9: Testing the perimeter law for $T = 1.20$ (Fig. (a)), and trend in the slope estimate b as $K \to K_c = 0.76$ (Fig. (b)). L denotes linear dimension of the lattice in which the Wilson loops were calculated, and P denotes the perimeter divided by 4, which is the side length of the Wilson loop W. For convenience, we have used square-shaped Wilson loops.



(a) Wilson loops at $L = 20$.

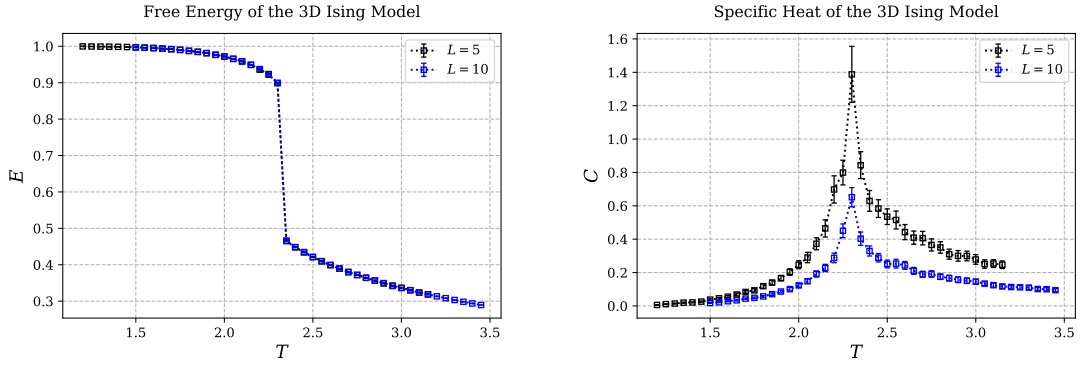
(b) Wilson loops at $L = 30$.

Figure 10: Behavior of the expectation value of Wilson-loop operators for low and high temperatures. The notation "W_n" refers to a square-shaped Wilson-loop with side length n. As we compute Wilson-loops at high temperatures, it becomes increasingly difficult to control the statistical uncertainties on $\langle W\rangle$.

3.3 4D $\mathbb{Z}_2$ Lattice Gauge Theory

From [3], the 4D $\mathbb{Z}_2$ lattice gauge theory is claimed to have a first-order transition demonstrating strong hysteresis effects–the dependence of the current state of a system on the path along the phase-space (parametrized by macroscopic variables. A *first-order phase transition*, contrary to a continuous phase transition, is characterized by a discontinuous jump in $E \propto \log Z$, To summarize, the 4D lattice gauge theory is characterized by:

1. First-order phase transition at (discontinuity in E) $T_c = \frac{2}{\ln(1+\sqrt{2})} \approx 2.27$, from [3].
2. Hysteresis effects that may be demonstrated via a time-evolution of the system at varying temperatures.
3. Strong meta-stability effects near the critical point, demonstrated on lattice sizes $L = 3$ to $L = 8$ in a paper by Micheal Cruetz [4].

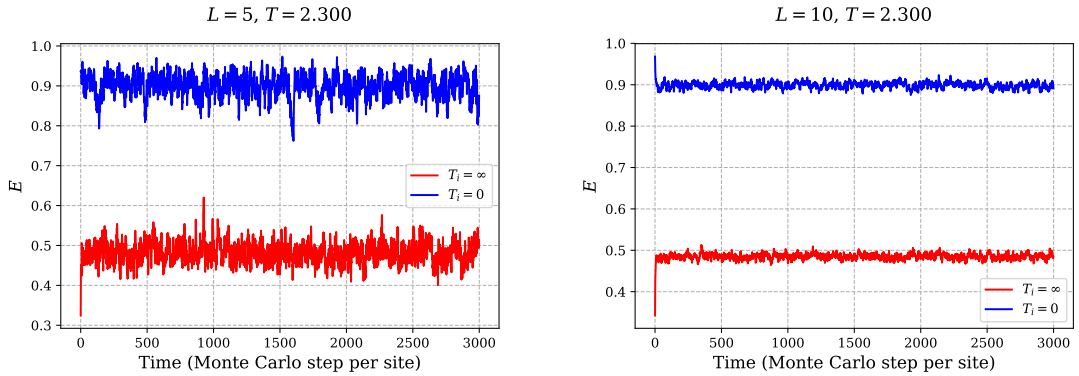


(a) Discontinuous change in the free-energy E.

(b) Peak in specific heat.

Figure 11: Clear evidence of a first-order phase transition for the 4D $\mathbb{Z}_2$ LGT.

Although there are theories of finite-size scaling of first-order transitions, such as [2], we will instead focus on the qualitative analysis of 4D lattice gauge theory and first-order transitions. As we see in Figure 12, even for small lattice sizes $L = 5, 10$ we observe strong evidence for a first-order phase transition, marked by a discontinuous change in the free energy E. The relaxation plots near the critical point $T_c \approx 2.27$ show evidence of pronounced meta-stability effects that fail to converge to equilibrium within the observed monte-carlo time frame. As noted in [5], the coexistence of two stable states at the critical point is expected for a first-order phase transition.



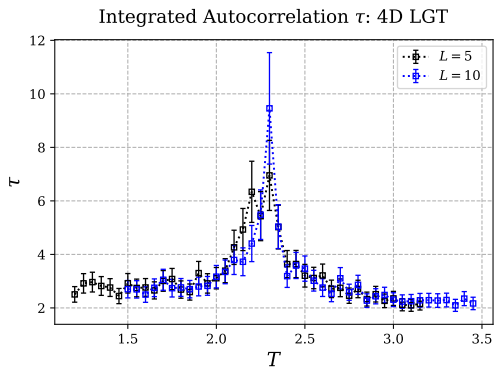
(a) Monte-carlo history for 4D LGT at $L = 5$.

(b) Peak in specific heat.

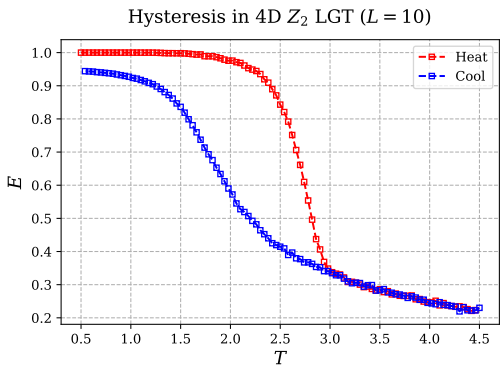
Figure 12: Monte-carlo history for 4D LGT at $L = 5$ and $L = 10$. Blue line indicates relaxation from $T = 0$ initial condition, and red line indicates relaxation from $T = \infty$ initial condition.

The difference in free energy between the two meta-stable states at the critical temperature is referred to as the *latent heat*–the free energy (per site) required for a phase transition at fixed temperature. From [5], the latent heat (per spin variable) is reported (in the Figure 3 of [5]) to be around ≈ 0.4, which agrees reasonably well with our observed values for the latent heat (see Figure 12).

Another feature of the 4D $\mathbb{Z}_2$ lattice gauge theory is the phenomenon of Hysteresis, which we can demonstrate using the $L = 10$ 4D LGT monte-carlo simulation. We include an example of a simulated heating-cooling cycle, starting with the initial condition $U_{ij} = +1$ for all bonds U_{ij}. Starting from $T = 0.5$ and $E = 1$ (free energy per site), we heat the system by varying the temperature and performing 10 monte-carlo sweeps over the lattice via the Metropolis method at each temperature. Although the formation of a Hysteresis loop in Figure 13 is clearly visible, if the system at each temperature step is allowed to cool for a longer period of time (i.e., greater number of monte-carlo sweeps per temperature), then the curves in Figure 13 will tend to converge, except possibly near the critical point where strong meta-stability effects suppress relaxation to a single ground state.



(a) Integrated autocorrelations for Figure 12.



(b) Hysteresis curve for $L = 10$ 4D LGT.

Figure 13: Measured integrated autocorrelation times τ for data used in Figure 12 included for reference. Plot (b) shows heating and cooling of a $L = 10$ 4D LGT system, with 10 monte-carlo sweep per site at each temperature. This experiment was suggested by Michael Cruetz on [3]. Note that we have omitted error-bars for Figure (b), as we are interested in overall qualitative behavior when the system is cooled down and heated up rapidly.

4 Discussion

Although our calculations of the critical point K_c are quite accurate, our values of the critical exponents ν and α obtained from the 3D Ising and 3D lattice gauge simulations differs from those reported in literature by roughly $\sim 2\sigma$. The 2σ difference is not a statistical impossibility; however, it is always better to identify and understand sources of systematic errors in our measurements of the critical exponents.

Our fit to the finite-size scaling relations of the specific heat C is determined by the values C_{max} near the critical point. As our lattice sizes increases, it becomes considerably difficult to locate the pseudo-critical point K_{max} where the specific heat C reaches the true maximum value for the given lattice size L. The value of K at which the pseudo-critical point occurs does not change drastically as $L \gg$ is varied, since as $L \to \infty$, we expect that the pseudo-critical point converges asymptotically to the true critical point K_c, and for $L \gg 1$, a small mis-identification of the pseudo-critical point near K_c will have diminishing effects on the quality of the power-law fit of Figures 5 (a) and 8 (a). However, this is not the case for the estimates for the specific heat, as a small deviation from the pseudo-critical point will tend to magnify the difference in the specific heat, due to the divergent behavior of C near the critical point. In other words, the estimates of C is highly unstable near the critical point–magnifying small errors in K_c' (the pseudo-critical point) to large discrepancies in C.

We must then ask where the systematic errors in locating K_c' originates from, and this is most probably caused by a combination of large spacing between the temperature sweeps (that is, the value $\Delta T = T_{i+1} - T_I$ used to scan over the range of temperatures) and the inefficiencies of the Metropolis algorithm near the critical point. More precisely, as we increase our lattice sizes, we expect that the pseudo-critical point approaches closer the critical point. To locate the pseudo-critical point in greater precision, we in turn require finer steps in temperature near K_c, and as we discussed in the footnotes, this region is precisely where the Metropolis algorithm fails to

provide decent statistics on $\langle C \rangle$. For instance, it is quite possibly likely that the values of C_{max} obtained for lattice sizes $L \geq 20$ in Figure 8 are gross underestimates, as our measurements show $C_{max}(L = 20) > C_{max}(L = 30)$ yet we expect that C_{max} diverges with increasing lattice sizes. Underestimation of C_{max} may drastically affect the fit parameters and their errors in 8.

Near the critical point, each measurement of our monte-carlo simulations become highly correlated, and we saw that the critical point is characterized by a divergence in the integrated correlation time that controls our statistical accuracy and uncertainty. The Metropolis algorithm, which is by definition a local algorithm that updates the configuration of the system by successive applications of local updates, is highly inefficient near the critical point–marked by a divergence in autocorrelations. Hence, to obtain decent statistics near the critical regime, ones needs an update algorithm more robust to the problems of critical slowing down and correlation between time-separated measurements. A popular algorithm used in Ising model simulations is the Wolff algorithm, where a cluster of spins are flipped simultaneously at each step. The Wolff algorithm, in contrast to the Metropolis algorithm, is inherently non-local, involving changing the configuration of the system over an extended range of spins. It is known that the Wolff algorithm "almost entirely removes the problem of critical slowing down" [1], and using non-local update algorithms such as the Wolff method is a promising path to a more accurate and precise determination of the critical exponents. Although we do not discuss the Wolff Algorithm in detail, a rough implementation using a *queue* data structure is provided in the 3D ising simulation source code `ising_3d.cpp` uploaded at Github: `https://github.com/dkoh0207/Statistical-Mechanics`.

Also, devising and applying non-local update algorithms for monte-carlo simulations of gauge theories is another direction for which an interested reader may pursue. A detailed survey of available algorithms for gauge theories are given, for example, in [12] and [14].

To summarize, using the Metropolis algorithm, we estimated the critical exponents ν and α for the 3D Ising and 3D lattice gauge theory, and demonstrated evidence of the duality relation. We also confirm the validity of the perimeter law for Wilson loops at low temperatures, and extend our simulations to 4D lattice gauge theory and show clear evidence of first order transition. A fruitful direction would be to use an algorithm more efficient and reliable near the critical regime to extract the critical exponents. As modern computational power grow and computer algorithms become ever-more efficient and sophisticated, using numerical simulations to probe many body systems, especially in strongly-correlation phases where perturbation theory cannot be applied, seems promising.

5 Appendix

Here, we include some example C++ implementations of the Metropolis algorithm for the 3D Ising and 3D lattice gauge theory. Since many of the code is reused in different simulations (ex. 3D LGT, 4D LGT) with a change in dimensions or lattice size, we will only include "representative" functions at each step of the monte-carlo simulation. Also, I have borrowed many of the ideas and implementations of monte-carlo simulations explained in detail in Hjorth Jensen's *Computational Physics* lecture notes [8] and Michael Cruetz's implementations of 4D $\mathbb{Z}_2$ lattice gauge theory in [3], which I explicitly quote in places if I have used them. Again, a full documented code is available at Github: `https://github.com/dkoh0207/Statistical-Mechanics`.

5.1 Imported Libraries

```
...
using namespace std;
ofstream ofile;
// The boost linear algebra C++ library is used to implement
// multidimensional arrays and tuples efficiently and reliably.
typedef boost::multi_array<int, 3> Lattice;
typedef Lattice::index coord;
typedef boost::tuple<int, int, int > site;

// Set a different seed for the random number generator
// at every run using the current time.
random_device rd;
// The Mersenne Twister 19937 Random Number Generator
mt19937_64 mt(rd());

inline int periodic(int i, const int limit, int add) {
  // This implementation of periodic boundary condition is from
  // the 2D ising model example by Hjorth Jensen.
  return (i + limit + add) % (limit);
}
...
```

5.2 Initializing the 3D Ising Lattice

```
void initialize( Lattice &lattice, map<int, double> &weights, double &K,
double &M, double &E, const int &coldstart) {
  /*
  Function for setting initial state, Boltzmann weights, and observables.

  Inputs:
    lattice: A 3D multidimensional boost array
    weights: A dictionary (ref to std::map) for saving pre-calculated
        weights.
    K: The value of the current temperature.
    M: (double) Magnetization
    E: (double) Energy
```

```
  coldstart: 1 for setting all sites = 1, -1 for setting all sites
      randomly
  between -1 and 1.

Returns:
  None (a void function)
*/
static uniform_int_distribution<int> u(0,1);
unsigned int lsize = 0;
lsize = lattice.size();
// Refresh Magnetization and Energy
M = 0;
E = 0;
// Set initial state for new MC run
for (auto i = 0; i != lsize; ++i) {
  for (auto j = 0; j != lsize; ++j) {
    for (auto k = 0; k != lsize; ++k) {
        if (u(mt) == 1) {
          lattice[i][j][k] = 1;
        } else {
          lattice[i][j][k] = coldstart;
        }
        // Adjust magnetization accordingly
        M += (double) lattice[i][j][k];
    }
  }
}
// After initializing lattice, compute initial energy:
for (auto i = 0; i != lsize; ++i) {
  for (auto j = 0; j != lsize; ++j) {
    for (auto k = 0; k != lsize; ++k) {
        E += lattice[i][j][k] * (
            lattice[periodic(i, lsize, 1)][j][k] +
            lattice[i][periodic(j, lsize, 1)][k] +
            lattice[i][j][periodic(k, lsize, 1)]);
    }
  }
}
/*
Initializing boltzmann weights to be used in each metropolis step.
This allows us to avoid computing exp(dE) at every step, which is
computationally expensive due to the exponential function.
*/
weights.insert(pair<int, double>(-12, exp(-12.0 * K)));
weights.insert(pair<int, double>(-8, exp(-8.0 * K)));
weights.insert(pair<int, double>(-4, exp(-4.0 * K)));
weights.insert(pair<int, double>(0, 1.0));
weights.insert(pair<int, double>(4, 1.0));
weights.insert(pair<int, double>(8, 1.0));
weights.insert(pair<int, double>(12, 1.0));
```

```
  // We check the initial magnetization (per site) and energy.
  cout << "Initial M = " << M / ((double) lsize*lsize*lsize) << endl;
  cout << "Initial E = " << E / ((double) lsize*lsize*lsize) << endl;
}
```

For 3D lattice gauge theory, only $-8, -4, 0, 4, 8$ are possible values for the exponential weights used in the Metropolis algorithm, since a randomly chosen bond shares itself with four square plaquettes. It is easy to check that the weights for the 4D lattice gauge theory is the same as that of the 3D Ising model.

5.3 Auxiliary Functions

```
...
site choose_random_site(const unsigned int lsize) {
  /*
  A helper function for choosing a random site on the lattice,
  using periodic boundary conditions.

  Inputs:
    lsize: Linear dimension of the lattice.

  Returns:
    s (site): A randomly selected site object (3D tuple) from the lattice
        .
    A site object consists of three coordinate indices.
  */
  static uniform_int_distribution<int> u(0, lsize-1);
  int idx, idy, idz = {0};
  // Call three random integers, which correspond to coordinates on the
      lattice.
  idx = u(mt);
  idy = u(mt);
  idz = u(mt);
  site s(idx, idy, idz);
  return s;
}
...
...
vector<site> find_neighbors(const site s, const unsigned int lsize) {
  /*
  A helper function for finding the six neighboring sites, given
  a specific site on the lattice.

  Inputs:
    site s: A site object indicating the current site.
    lsize: Linear dimension of the lattice

  Returns:
    neighbors: A std::vector of sites containing the site objects
    for the six neighbors of site s.
  */
```

```
  vector<site> neighbors;
  int idx, idy, idz = {0};
  idx = s.get<0>();
  idy = s.get<1>();
  idz = s.get<2>();
  for (auto i = 0; i != 3; ++i) {
    if (i == 0) {
      site n1(periodic(idx, lsize, 1), idy, idz);
      site n2(periodic(idx, lsize, -1), idy, idz);
      neighbors.push_back(n1);
      neighbors.push_back(n2);
    }
    if (i == 1) {
      site n1(idx, periodic(idy, lsize, 1), idz);
      site n2(idx, periodic(idy, lsize, -1), idz);
      neighbors.push_back(n1);
      neighbors.push_back(n2);
    }
    if (i == 2) {
      site n1(idx, idy, periodic(idz, lsize, 1));
      site n2(idx, idy, periodic(idz, lsize, -1));
      neighbors.push_back(n1);
      neighbors.push_back(n2);
    }
  }
  return neighbors;
}
...
...
int deltaE(site s, Lattice& lattice, const unsigned int lsize, const
    double& K) {
  /*
  Helper function for computing the change in free energy that would
  occur if a site is to be flipped.

  Inputs:
    site s: The current site in consideration
    lattice: 3D multidimensional boost array modeling the lattice.
    lsize: Linear dimension of the lattice.
    K: Current temperature

  Returns:
    dE: (double) The change in free energy E_f - E_i.
    The convention is that E > 0, so that the ``free energy'' is
    actually given by -E.
  */
  int dE = 0;
  vector<site> neighbors = find_neighbors(s, lsize);
  for (auto n : neighbors) {
    dE += lattice[n.get<0>()][n.get<1>()][n.get<2>()];
```

```
  }
  dE = dE * lattice[s.get<0>()][s.get<1>()][s.get<2>()];
  dE = -dE * 2;
  return dE;
}
...
```

Note that we are actually calling the negative of the free energy as "the free energy." If we define the free energy such that $E < 0$ and E is minimized at equilibrium, we must multiply dE by -1 at the last step.

5.4 The Metropolis Algorithm

```
...
void metropolis(Lattice &lattice, const unsigned int lsize,
const double &K, double& M, double& E, map<int, double>& weights) {
  /*
  Implements the Metropolis Algorithm

  Inputs:
    lattice: A 3D Boost multidimensional array modeling the Ising Lattice
        .
    lsize: Linear dimension of the lattice.
    K: Inverse temperature (reference)
    M: Magnetization (reference)
    E: Free Energy (reference)
    weights: A c++ map (key-value pair) use to efficiently calculate
    the Boltzmann weights.

  Returns:
    None (a void function)

  */
  unsigned int count_flipped = 0;
  unsigned int n = lsize*lsize*lsize;
  double w = 0.0;
  // The static qualifier is to ensure that each distribution is
      initialized
  // only once throughout the program.
  static uniform_real_distribution<double> u(0,1);
  int dE = 0;
  for (auto i = 0; i != n; ++i) {
    int key = 0;
    site s = choose_random_site(lsize);
    dE = deltaE(s, lattice, lsize, K);
    w = weights[dE];
    if (u(mt) < w) {
      count_flipped += 1;
      lattice[s.get<0>()][s.get<1>()][s.get<2>()] *= -1;
      M +=  2 * lattice[s.get<0>()][s.get<1>()][s.get<2>()];
      E += ((double) dE);
```

```
    }
  }
}
...
```

5.5 Lattice Gauge Simulations

We use the same Metropolis algorithm used above, but with a different lattice definition:

```
...
typedef boost::multi_array<int, 4> Lattice;
typedef Lattice::index coord;
typedef boost::tuple<int, int, int, int> site;
...
```

We need 4 indices for the 3D lattice gauge theory: three for the coordinates and one for the bond directions. For each site, we have three bonds, so the 3D lattice gauge theory as a total of $3L^3$ number of lattice sites. The 4D lattice gauge theory needs 5 indices with 4 labeling the coordinates and one labeling the four bond directions.

It is essential to check that the updating the energy via the dE function gives the same value if we brute-force compute the plaquette energy of the lattice. The following function computes the plaquette energy via the brute-force method.

```
...
double compute_free_energy(Lattice &lattice,
const unsigned int lsize, const unsigned int n_bonds, const double & K) {

    double E_temp = 0.0;
    double plaquettes = 0.0;

    for (auto i = 0; i != lsize; ++i) {
        for (auto j = 0; j != lsize; ++j) {
            for (auto k = 0; k != lsize; ++k) {
                plaquettes = 0.0;
                // Square plaquette in the (1,2) direction.
                plaquettes += (double) (lattice[i][j][k][0] * lattice[i][
                    j][k][1]
                * lattice[periodic(i, lsize, 1)][j][k][1]
                * lattice[i][periodic(j, lsize, 1)][k][0]);
                // Square plaquette in the (2,3) direction.
                plaquettes += (double) (lattice[i][j][k][1] * lattice[i][
                    j][k][2]
                * lattice[i][periodic(j, lsize, 1)][k][2]
                * lattice[i][j][periodic(k, lsize, 1)][1]);
                // Square plaquette in the (1,3) direction.
                plaquettes += (double) (lattice[i][j][k][0] * lattice[i][
                    j][k][2]
                * lattice[periodic(i, lsize, 1)][j][k][2]
                * lattice[i][j][periodic(k, lsize, 1)][0]);
                E_temp += plaquettes;
            }
        }
```

```
    }
    return E_temp;
}
...
```

However, as was the case for the 3D Ising simulation, it is more efficient to compute the initial energy once and update the energy using dE as we proceed to evolve the system, as suggested in Jensen's notes [8]. In computing dE for the lattice gauge theory, we quote Michael Cruetz's article *Simulating Quarks* [3]. A C++ implementation of the same method is given below.

```
...
double deltaE(Lattice &lattice, site &bond,
const unsigned int lsize, const unsigned int n_bonds, const double &K) {
    double dE = 0.0;
    double plaq_sum = 0.0;
    vector<int> ind(3);
    int b = 0;
    ind[0] = bond.get<0>();
    ind[1] = bond.get<1>();
    ind[2] = bond.get<2>();
    b  = bond.get<3>();
    for (int m = 0; m != n_bonds; ++m) {
        double plaq = 0.0;
        plaq = 0;
        // This particular implementation of computing the energy
        // is borrowed from Michael Cruetz's z2 lattice gauge simulation.
        if (m != b) {
            plaq = lattice[ind[0]][ind[1]][ind[2]][b];
            ind[m] = periodic(ind[m], lsize, -1);
            plaq *= lattice[ind[0]][ind[1]][ind[2]][m] *
            lattice[ind[0]][ind[1]][ind[2]][b];
            ind[b] = periodic(ind[b], lsize, 1);
            plaq *= lattice[ind[0]][ind[1]][ind[2]][m];
            plaq_sum += plaq;
            ind[m] = periodic(ind[m], lsize, 1);
            plaq = lattice[ind[0]][ind[1]][ind[2]][m];
            ind[m] = periodic(ind[m], lsize, 1);
            ind[b] = periodic(ind[b], lsize, -1);
            plaq *= lattice[ind[0]][ind[1]][ind[2]][b];
            ind[m] = periodic(ind[m], lsize, -1);
            plaq *= lattice[ind[0]][ind[1]][ind[2]][m] *
            lattice[ind[0]][ind[1]][ind[2]][b];
            plaq_sum += plaq;
        }
    }
    dE = -2 * plaq_sum;
    return dE;
}
...
```

A pictorial representation of the algorithm may help understand how dE is calculated. For the 4D

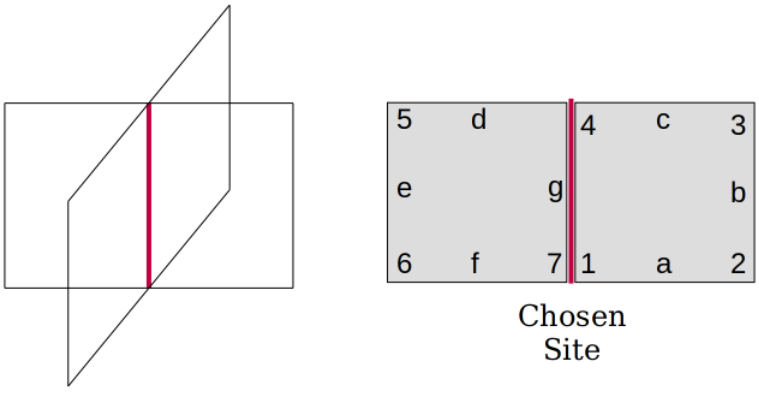


Figure 14: Contributions to dE for a chosen random bond (left) in the 3D LGT. In Cruetz's method for calculating dE, one starts from the site 1 and traverses the plaquette in numeric order labeled in the right diagram, multiplying the bonds on the boundary of the plaquette as it proceeds.

lattice gauge theory, we have one more pair of plaquettes in addition to the four plaquettes drawn on the left of Figure 14, and the total number of plaquettes is given by $L^4 * \binom{4}{2} = 6L^4$. That is, at each site, we may choose two directions that define a plaquette. This normalization factor is need if one aims to normalize the free energy to 1. The following function computes the Wilson loops:

```
...
int compute_wilson_loop(Lattice &lattice, const unsigned int l) {
    /*
    Function for computing a Wilson loop of side length l.
    For simplicity, we will use the Wilson loop oriented
    on the xy-axis with z = 0.
    */
    int wilson_loop = 1;

    for (auto i = 0; i < l; ++i) {
        wilson_loop *= lattice[i][0][0][0];
    }
    for (auto i = 0; i < l; ++i) {
        wilson_loop *= lattice[l][i][0][1];
    }
    for (int i = l-1; i >= 0; --i) {
        wilson_loop *= lattice[i][l][0][0];
    }
    for (int i = l-1; i >= 0; --i) {
        wilson_loop *= lattice[0][i][0][1];
    }

    return wilson_loop;
}
...
```

References

[1] M.E.J. Newman & G.T. Barkema, *Monte carlo methods in statistical physics*, Oxford Uinversity Press, 1999.

[2] K. Binder and D. P. Landau, *Finite-size scaling at first-order phase transitions*, Phys. Rev. B **30** (1984Aug), 1477–1485.

[3] Michael Creutz, *Simulating quarks*, Computing in Science and Engineering **6** (200404), 80–85.

[4] Michael Creutz, Laurence Jacobs, and Claudio Rebbi, *Monte carlo computations in lattice gauge theories*, Phys. Rept. **95** (198304), 201.

[5] Michael Creutz, P. Mitra, and K. J. M. Moriarty, *Computer investigations of the three-dimensional ising model*, Journal of Statistical Physics **42** (1986Mar), no. 5, 823–832.

[6] Shmuel Elitzur, *Impossibility of spontaneously breaking local symmetries*, Phys. Rev., D, v. 12, no. 12, pp. 3978-3982 **12** (197512).

[7] Rajan Gupta and Pablo Tamayo, *Critical exponents of the 3-d ising model* (1996), available at `arXiv:cond-mat/9601048`.

[8] Hjorth Jensen, *Computational physics lecture notes fall 2011*, 2011.

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

[10] David P Landau and Kurt Binder, *A guide to Monte Carlo simulations in statistical physics*, Cambridge Univ. Press, Cambridge, 2000.

[11] Makoto Matsumoto and Takuji Nishimura, *Mersenne twister: A 623-dimensionally equidistributed uniform pseudorandom number generator*, 1998.

[12] Stephen M Pickles, *Algorithms in lattice qcd* (1998).

[13] Kari Rummukainen, *Simulation methods in physics lecture notes*, 2011.

[14] Alan David Simpson, *Algorithms for lattice qcd* (1991).

[15] Alan D. Sokal, *Monte carlo methods in statistical mechanics: Foundations and new algorithms*, 1996.

[16] David Tong, *Gauge theory*, Cambridge Lecture Notes in Physics, Cambridge, 2018.

[17] ———, *Statistical physics*, University of Cambridge Part II Mathematical Tripos, Cambridge, 2018.

[18] Daniel V. Schroeder, *An introduction to thermal physics*, 2019.

[19] F. J. Wegner, *Duality in Generalized Ising Models and Phase Transitions without Local Order Parameters*, Journal of Mathematical Physics **12** (October 1971), 2259–2272.

[20] Edward Witten, *Symmetry and emergence* (2017), available at `arXiv:1710.01791`.