December 2015

Understanding Disordered Systems Through Numerical Simulation and Algorithm Development

Sean M. Sweeney
Syracuse University

Abstract

Disordered systems arise in many physical contexts. Not all matter is uniform, and impurities or heterogeneities can be modeled by fixed random disorder. Numerous complex networks also possess fixed disorder, leading to applications in transportation systems [1], telecommunications [2], social networks [3, 4], and epidemic modeling [5], to name a few.

Due to their random nature and power law critical behavior, disordered systems are difficult to study analytically. Numerical simulation can help overcome this hurdle by allowing for the rapid computation of system states. In order to get precise statistics and extrapolate to the thermodynamic limit, large systems must be studied over many realizations. Thus, innovative algorithm development is essential in order reduce memory or running time requirements of simulations.

This thesis presents a review of disordered systems, as well as a thorough study of two particular systems through numerical simulation, algorithm development and optimization, and careful statistical analysis of scaling properties.

Chapter 1 provides a thorough overview of disordered systems, the history of their study in the physics community, and the development of techniques used to study them. Topics of quenched disorder, phase transitions, the renormalization group, criticality, and scale invariance are discussed. Several prominent models of disordered systems are also explained. Lastly, analysis techniques used in studying disordered systems are covered.

In Chapter 2, minimal spanning trees on critical percolation clusters are studied, motivated in part by an analytic perturbation expansion by Jackson

and Read [6] that I check against numerical calculations. This system has a direct mapping to the ground state of the strongly disordered spin glass [7]. We compute the path length fractal dimension of these trees in dimensions $d = \{2,3,4,5\}$ and find our results to be compatible with the analytic results suggested by Jackson and Read.

In Chapter 3, the random bond Ising ferromagnet is studied, which is especially useful since it serves as a prototype for more complicated disordered systems such as the random field Ising model and spin glasses. We investigate the effect that changing boundary spins has on the locations of domain walls in the interior of the random ferromagnet system. We provide an analytic proof that ground state domain walls in the two dimensional system are decomposable, and we map these domain walls to a shortest paths problem. By implementing a multiple-source shortest paths algorithm developed by Philip Klein [8], we are able to efficiently probe domain wall locations for all possible configurations of boundary spins. We consider lattices with uncorrelated disorder, as well as disorder that is spatially correlated according to a power law. We present numerical results for the scaling exponent governing the probability that a domain wall can be induced that passes through a particular location in the system's interior, and we compare these results to previous results on the directed polymer problem.

Understanding Disordered Systems Through Numerical Simulation and Algorithm Development

by

Sean Michael Sweeney

B.S. Physics, Syracuse University, 2010
B.S. Mathematics, Syracuse University, 2010

DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Physics.

Syracuse University
December 2015

Acknowledgements

I would like to express my gratitude to Alan Middleton for guiding me through my graduate career. Anything I have learned in physics or computer science has been due in large part to the stimulating and enthusiastic discussions I've had with Alan and his expertise in these fields. Even as an undergraduate research assistant, he has always made me feel like a colleague rather than a subordinate, and I am grateful for the relationship we've had.

I would also like to reflect on the fact that my graduate experience would not have been nearly as smooth or enjoyable without the help of the Syracuse University physics department faculty and staff. Between the professors that piqued my interest in physics and the administrative assistants and secretaries without whom the entire department would grind to a halt, there are many invaluable people to which I am indebted.

I want to thank Creighton Thomas for offering me his thesis template and providing me with advice on my career, my research, and thesis writing.

Lastly, I need to thank my family and friends for putting up with me while I ignored the outside world for months at a time and my body for managing to survive on such little sleep.

Contents

List of Figures

Chapter 1

Introduction

Physicists aim to explain the fundamentals of the material world around us. We write mathematical equations to quantify the laws governing the universe, considering everything from normal forces to magnetic fields to quantum mechanical wave-functions or time dilation. Newton, Maxwell, Schrodinger, Einstein and the like—they all improved the collective human understanding of these universal laws, bringing us as a species one small step closer to explaining the natural phenomena around us. Perhaps the most ambitious and impressive branch of physics is statistical physics, as it provides the tools to tackle problems whose complexity makes them them seem impossible to solve.

While most high school physics students could explain the kinematics behind a single body in projectile motion, and any undergraduate physics student worth her salt could write an analytical solution to a two-body problem; what about a many-body problem of N interacting bodies? While traditional classical mechanics provides little hope of solving a problem with three or more interacting bodies, statistical mechanics does so with ease. Where free-body

diagrams and force laws run out of steam, statistical physics excels through the use of ensemble averages, partition functions, thermodynamic quantities, and other tools that take advantage of probabilistic mathematics.

Through the methods provided by statistical mechanics, one can completely and accurately describe the macroscopic properties and behavior of a system of N interacting particles. Employing a probabilistic analysis of the possible microstates or allowed configurations of a system, one can fully describe the system's temperature, entropy, or free energy, for instance. In this way, pure systems are often well-understood, and some systems have been completely solved analytically. Systems that exhibit some form of disorder, however, are typically much tougher to study, and exact solutions are extremely rare.

It is worth making a clarifying note for the reader here on the dual usage of the term "disordered." Throughout this thesis we will used the term "disordered" to refer to systems with fixed random heterogeneities (in contrast to pure systems), and we will also use "disordered" to refer to a system that has undergone a phase transition from an ordered to a disordered phase, with the meaning determined by context.

Take, for example, a ferromagnet with Ising spins and nearest-neighbor interactions. We can express the total energy of the system by writing the Hamiltonian, summing over all nearest-neighbor pairs of spins:

$$H_{\text{pure}} = -J \sum_{\langle ij \rangle} s_i s_j \ . \tag{1.1}$$

Here s_i are Ising spins taking values of $\{+1, -1\}$, and J is the interaction strength of nearest-neighbor spin pairs. Since we are dealing with purely ferromagnetic interactions, $J > 0$, and spins "want" to align in order to reduce the system energy. In the pure case, we have a uniform interaction strength,

J, which allows the system to be easily well-understood. Due to the identical magnitude of interaction terms for all pairs of spins, we can trivially predict based on given boundary conditions (some set of + and − spins at the system boundary) exactly where domain walls will appear in the system, dividing regions of + and − spins. These paths will be straight, minimizing the cost of each domain wall by minimizing the number of unsatisfied couplings along that path.

If disorder is introduced in the form of impurities, however, the Hamiltonian can then be expressed as

$$H_{\text{RBIM}} = -\sum_{\langle ij \rangle} J_{ij} s_i s_j \ , \tag{1.2}$$

with the J_{ij} couplings being positive but random in magnitude. This is sometimes called a random bond Ising magnet (RBIM). In this disordered case, the locations of these domain walls become difficult to predict without iterating over all possible system configurations to find the ground state. This randomness associated with disordered systems is precisely what drives much of their rich behavior such as glassiness, power law scaling, absence of a characteristic length scale, and slow dynamics. However, this rich behavior also makes these systems much more difficult to study analytically. A simple RBIM system of 100 spins would have $2^{100} \approx 10^{30}$ possible system configurations to consider when searching for the ground state configuration, making exhaustive simulations of these configurations typically not feasible.

Given that the techniques used in statistical physics draw heavily on rigorous mathematics of combinatorics and probability theory, statistical physics truly is a hybrid field, offering interesting interdisciplinary applications. With biophysics, chemical physics, and social science coming to mind as areas of ex-

citing overlap in subject matter, the multi-faceted utility of statistical physics is apparent. Furthermore, due to the lack of translation invariance, disordered systems naturally lend themselves well to graph theory. Particularly, general graphs and non-uniform grids become useful tools in modeling disordered systems. Thus it seems natural to reach out to computer science for help in the problems where statistical physics often falls short—namely disordered systems. Whereas iterating system configurations isn't a feasible task by hand, if we ask a computer to perform this exhaustive search for us, we can in fact perform the search in a reasonable amount of time, provided the system is small enough. A typical computer processor can perform in a fraction of a microsecond operations that would take a human minutes or hours to complete.

Since the advent of computational physics, all one needs is a simple model (a system Hamiltonian, for example) for a physical system and an algorithm to allow the system to evolve or to iterate through the possible system configurations. This is, at the most fundamental level, the essence of numerical simulation. This clever fusion between computer science and theoretical physics has only really been in existence for the last fifty years or so, but has already provided tremendous advances in theoretical physics. And while rapid technological development is largely responsible for the success of computational physics, there are also integral contributions made by humans in this field, developing clever algorithms to increase the efficiency of simulations. Without innovative work to design techniques like Monte Carlo integration, molecular dynamics, simulated annealing, or shortest paths and max-flow algorithms, most of the work done in computational physics would not be possible. Thus we have a unique situation where humans and computers have to "work to-

gether," so to speak, in order to solve complex problems. Computers offer lightning fast simple operations, but careful human guidance must be present to provide the computers with the correct tasks, optimizing the overall search for a solution and cutting corners wherever possible. Especially in systems where the thermodynamic limit is relevant and where we want to examine large length scales, the need for efficient algorithms, both in terms of memory usage and computational complexity, is paramount.

This chapter first provides a brief overview of the physics behind disordered systems, discussing some real-life examples of disordered systems as partial motivation for studying such hard-to-study problems. The ideas of broken symmetry and quenched disorder are explained in some detail. I offer a brief overview of phase transitions and criticality, as well as some of the techniques used to study systems that display a phase transition—namely Landau's mean field theory and Wilson's renormalization group approach. Naturally, discussion of the renormalization group leads to talk of scale invariance, and I discuss the physical and mathematical implications of scale invariance in terms of power laws, fractal objects, critical exponents, and universality.

In order to impart a more robust understanding of the theoretical physics of disordered systems, I next cover several prominent models of disordered systems, including the Hamiltonians, basic results, and fundamental properties governing their behavior. This section serves to provide a sound background for the presentation of algorithms and project specifics in later chapters.

Following this is an examination of the statistical analysis techniques employed in this thesis, including the theory of scaling plots, chi-squared tests for goodness of fit, computation of error bars, and bootstrap resampling. This section serves to inform the numerical analysis carried out in the course of the

projects presented in later chapters.

Lastly, a more technical review of computational techniques is presented, covering computer languages and scientific tools used, data structures, and a review of computational complexity as a practical means of measuring algorithm efficiency.

1.1 Systems With Quenched Disorder

The difference between ordered and disordered systems is that ordered systems exhibit some type of long-range order that "breaks" the symmetry of the Hamiltonian in the disordered phase. For example, systems with crystalline ordering have repeated unit cells tiling space in a predictable fashion, so that translational invariance holds only for some discrete set of displacement vectors and linear combinations of these vectors. In the disordered state of the pure system, translational symmetry holds in all directions and the system is said to be isotropic. The disordered state of the ferromagnet has no long-range correlations between spins and maintains the spin-flip symmetry of the Hamiltonian. In the pure ferromagnet, magnetic ordering at low temperatures means that spins are strongly correlated and tend to all point in the same direction, giving way to a state of either all "up" or all "down" spins with a nonzero magnetization.

There are two varieties of disorder that physicists typically consider—annealed disorder and quenched disorder. With annealed disorder, random variables in the system fluctuate over time due to thermal noise. With quenched disorder, however, these random variables do not evolve over the scale of measurement time; rather, they are "frozen in" and remain fixed for a given realization of

the system, often making them very important to the evolution of that system. Whereas annealed disorder is susceptible to thermal noise and tends to average out over time, quenched disorder remains fixed and determines the behavior of the system, even at (and in fact especially at) long time scales. Quenched disorder is often much more difficult to study that annealed disorder, since thermal averages are not equivalent to averages over disorder, as is the case for systems with annealed disorder. While this makes systems with quenched disorder more difficult to study, it also provides much of the rich behavior that makes these systems interesting.

As additional motivation to study disordered systems, numerous systems in real life possess unchanging random elements in the form of quenched disorder. In magnetic spin systems such the random ferromagnet, fixed non-heterogeneities are caused by atoms being replaced by atoms of a different element (substitution alloys) or by missing atoms, where atoms don't rearrange on short timescales. Quenched disorder also arises in transportation networks [1], drainage networks (watersheds) [13, 14], epidemic modeling [5], the spread of computer viruses [15], as well as many other contexts. For example, a transportation network might be modeled by a graph of vertices (cities) and edges (roads), with a weight for each edge representing the total time it takes to traverse that road. This weight represents a disorder that, barring extreme traffic fluctuations or road closures, could be considered to be quenched or fixed in time.

For a more abstract example, one could look to work by Moore and Katzgraber [16] modeling political policies with a spin glass Hamiltonian:

$$H_{\mathrm{SG}} = -\sum_{i<j} J_{ij} s_i s_j + \sum_i h_i s_i \ . \tag{1.3}$$

Here the Ising spins s_i represent whether or not a party should adopt a given policy independently, and the couplings J_{ij} account for how correlated or anticorrelated any two given policies s_i and s_j are. For instance, a policy on increased funding for law enforcement might be strongly correlated to a policy aimed at crime reduction. Meanwhile the external field term h_i serves to align the choices of policies adopted with the given party's preexisting manifesto, since there is a certain voter popularity cost associated with flip-flopping in elections. The authors of this work suggest a format for questionnaire that could be given to voters in order to collect real-world data on this subject. Though the simulations presented in this work are simply mathematical models, interesting phenomena emerge such as the existence of a dominant policy (in some ranges of disorder) that is strongly correlated with most other policies. Perhaps future work with real voter data could confirm the common hypothesis that many voters make their decision based on one or two core issues.

The existence of quenched disorder in physical systems can lead to some quite interesting and unique behavior. When examining excitations in disordered XY model [17, 18], domain walls in spin glass ground states [19], or boundaries between drainage basins [14], for example, fractal objects emerge with self-similar structure at all length scales. This leads to systems being governed by power laws with critical exponents that can be universal, something that will be discussed in more detail later in this chapter.

Disordered systems raise important, subtle questions about the idea of connectivity and whether boundary conditions play a crucial role in determining the physics of a given system. Because of the quenched disorder, we can see somewhat non-intuitive behavior like fixed "islands" of spins in a random field

Ising magnet [20], independent of boundary conditions. This provides fuel for the discussion of single versus multiple thermodynamic states in spin systems such as the spin glass [21, 22]. For physical applications of these ideas of connectivity and the significance of boundary conditions, one can look to fracturing in granular materials [23, 24], force chains [25, 26], jamming [27, 28], resistor networks [4], social network theory [4, 29], or even citation networks of scientific papers [30, 31].

1.2 Phase Transitions and Criticality

One can better understand the nature of quenched disorder by examining phase transitions. For a useful text that covers a large portion of what will be discussed in this section, refer to Nigel Goldenfeld's "Lectures on Phase Transition and the Renormalization Group" [32]. In systems that exhibit a phase transition, there is an ordered phase for which some spatial or orientational symmetry of the Hamiltonian is broken, as well as a disordered phase where the system is isotropic with respect to a particular degree of freedom. The nature of these phases and precisely how the system passes from one phase to another has been a topic of considerable interest over the last fifty years.

1.2.1 Ising Model

Here we will present the two dimensional Ising model as a pedagogical example, with Hamiltonian given by

$$H_{\mathrm{IM}} = -J \sum_{\langle ij \rangle} s_i s_j - h \sum_i s_i \ . \tag{1.4}$$

Here s_i are Ising spins taking values of $\{+1,-1\}$ (up or down), and J is a uniform interaction strength for nearest-neighbor spins. This interaction is ferromagnetic, so $J>0$ and spins tend to align with their neighbors in order to minimize the Hamiltonian. The h term represents an applied external field with which the spins tend to align. If we look at the free energy $F=U-TS$, we see that at low temperature T, the energy U will dominate and behavior will be determined solely by the Hamiltonian. Hence, at low temperature we see an ordered phase where spins are all aligned either up or down depending on the external field h. Meanwhile, at high temperature, the entropy S dominates the free energy, and we see a disordered phase where spins are equally likely to be up as they are to be down. There will be some critical temperature T_c that separates these two phases. Temperature is the tunable parameter that governs which phase the system is in, as demonstrated in Fig. 1.1.

The net magnetization, $M=\frac{1}{N}\sum_{i=1}^{N}s_i$, is called the order parameter for the system, as it essentially measures the degree of order in the system. In the disordered phase, the average magnetization will be $M=0$, whereas in the ordered phase we will see either $M=1$ or $M=-1$, depending on whether the external field h is up or down. If we begin with the system in a disordered phase ($T>T_c$) and a nonzero external field and lower the temperature T, when we get below T_c the system will become ordered and spins will line up with the external field. If the external field is up, we will have a phase with all spins up and $M=1$. If the external field is in the opposite direction, we will have an ordered phase with all spins pointing down and $M=-1$. But what if we perform the same experiment with $h=0$, no external field? As it turns out, our system still becomes ordered, and must choose either the up or down state when the system is cooled below T_c, as see in Fig. 1.2. This spontaneous

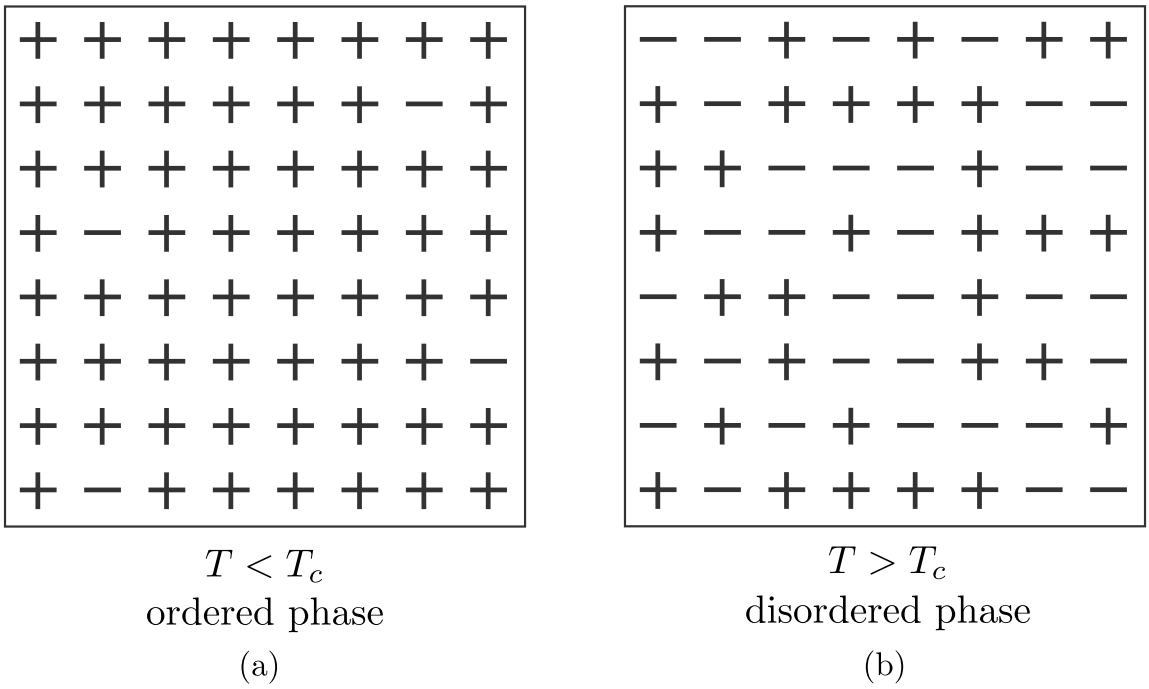

Figure 1.1: Ordered and disordered phases of the two-dimensional Ising ferromagnet. For temperature T below the critical temperature T_c, the system is in an ordered phase as shown in (a), with spins tending to align uniformly up or down (+ or −). For temperatures above the critical temperature, the system will exhibit a disordered phase, with spins randomly oriented both up and down, as seen in (b).

magnetization is known as spontaneous symmetry breaking, since the system "spontaneously" falls into an ordered state with spins aligned in one of two directions, despite having no external magnetic field to dictate this direction.

Rather than speaking of broken symmetry, one will often hear this process referred to as "ergodicity breaking." Ergodicity is the property of a system to have all of its microstates accessible. Essentially, a system is ergodic if it is allowed to explore all of its phase space as the system evolves over time, even if some of these states are less likely than others. This allows averaging over all phase space (an ensemble average) and averaging over time to be interchanged, which is convenient for the calculation of thermodynamic quantities. So in a system where symmetry breaking has occurred, the system has chosen one branch of the bifurcation that exists in phase space (either $M > 0$ or $M < 0$ in this case), restricting evolution of the system over time to only half of the possible microstates. Thus, the system is no longer ergodic in the ordered phase in the infinite system size limit.

1.2.2 Classification of Phase Transitions

Phase transitions are typically sorted into two broad categories: first order transitions and continuous (second order) transitions. Originally Ehrenfest postulated to classify phase transitions based on the lowest derivative of the free energy that was not continuous across the transition [33]. Although this classification is a bit outdated as it doesn't cover all possibilities and nuances for known phase transitions, it serves as a good starting point. Following this definition, for transitions at a critical point where distinct phases can coexist simultaneously, a first derivative of the free energy is discontinuous. For

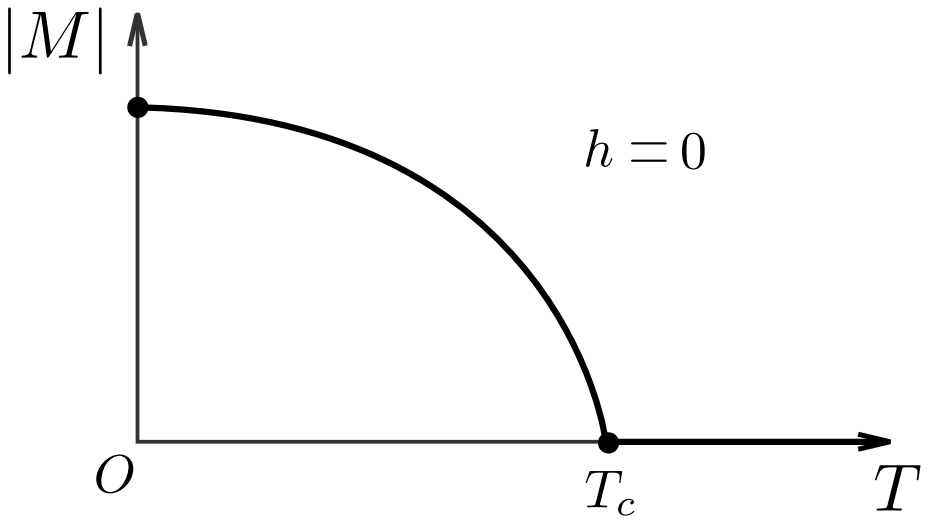

Figure 1.2: Net magnetization $M = \frac{1}{N}\sum_{i=1}^{N} s_i$ versus temperature T for a two-dimensional Ising ferromagnet with N spins and zero external field h. In the absence of an external magnetic field, spontaneous symmetry breaking occurs below T_c in the form of spontaneous magnetization, with spins tending to align either up or down ($M > 0$ or $M < 0$).

example, in the melting of ice to form liquid water, density—the first derivative of the free energy with respect to the chemical potential—is discontinuous across the transition. On the other hand, in second order or continuous phase transitions, first order derivatives of the free energy are continuous across the transition (hence the transition being "continuous") but may have a cusp or nonanalyticity representing a discontinuity in a second derivative of the free energy such as a specific heat capacity or susceptibility. In the Ising model, for example, the magnetization $M = \frac{1}{N}\frac{\partial F}{\partial h}$ is continuous across the ferromagnetic phase transition, but the magnetic susceptibility $\chi = \frac{\partial M}{\partial h}$ is discontinuous at $T = T_c$, as seen in Fig. 1.3. Since critical phenomena are typically associated with continuous phase transitions, we will restrict our discussion to systems that exhibit such continuous phase transitions.

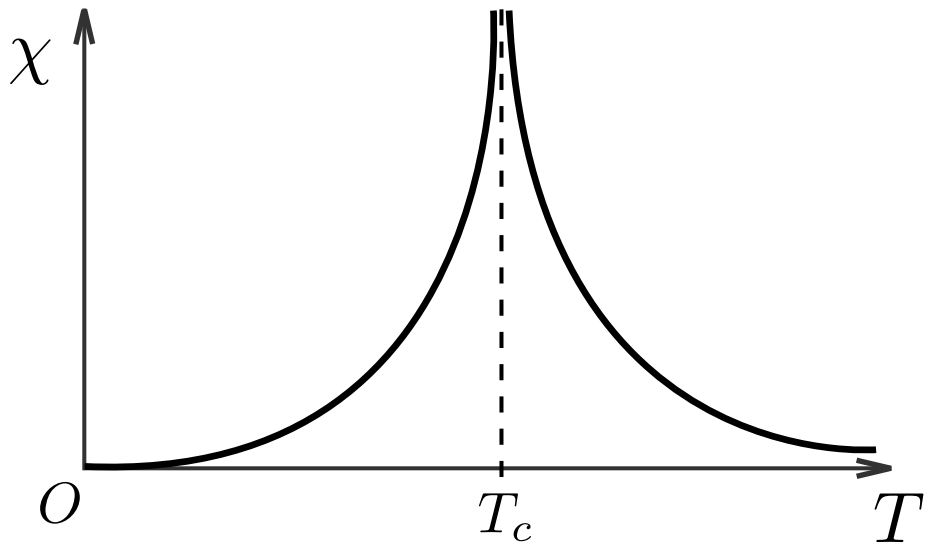

Figure 1.3: Magnetic susceptibility $\chi = \frac{\partial M}{\partial h}$ versus temperature T for a two-dimensional Ising ferromagnet with N spins. While the magnetization $M = \frac{1}{N}\frac{\partial F}{\partial h}$ is a first derivative of the free energy F and is continuous across the ferromagnetic phase transition, magnetic susceptibility is discontinuous across the transition. Thus, the ferromagnetic phase transition is a *continuous* or second order phase transition.

1.2.3 Mean Field Theory

When trying to understand systems with continuous phase transitions, one might first look at Landau's mean field approach [34, 35]. The basic idea is to replace the numerous degrees of freedom of a system with a single "mean field." In the Ising model, for example, one can replace all of the spin interactions with a mean field acting upon each individual spin. In essence, the sum total of the interactions between a spin s_i and all other spins $\{s_{j\neq i}\}$ is represented by an average field that is produced by the spins $\{s_{j\neq i}\}$. The approximation used to derive this mean field assumes the fluctuations of individual spins about the average magnetization in the system to be small.

Mean field theory predicts the behavior of various thermodynamic quantities as a function of $|T-T_c|$, the deviation of temperature away from the critical point. For example, in spin systems such as the Ising model, the specific heat capacity C can be written as

$$C \sim |T - T_c|^{-\alpha} \ , \tag{1.5}$$

the magnetization M follows the relationship

$$M \sim |T - T_c|^{\beta} \ , \tag{1.6}$$

the magnetic susceptibility χ obeys the relation

$$\chi \sim |T - T_c|^{-\gamma} \ , \tag{1.7}$$

and the correlation length ξ can be expressed as

$$\xi \sim |T - T_c|^{-\nu} \ . \tag{1.8}$$

These critical exponents $(\alpha, \beta, \gamma, \nu)$ govern the thermodynamic behavior of the system and can be calculated fairly easily. Unfortunately, however, it turns

out that the mean field results only hold true at or above some upper critical dimension d_u. For any dimension $d < d_u$, mean field results may not be correct. For instance, mean field theory predicts $\beta = 1/2$ and $\nu = 1/2$ for the Ising model in three dimensions, when in fact the values are much closer to $\beta = 1/3$ and $\nu = 2/3$ [36, 37]. The upper critical dimension for the Ising model turns out to be $d_u = 4$. Below four dimensions, the fluctuations that were assumed to be small in approximating the mean field become important to the dynamics of the system, meaning mean field results do not hold in less than four dimensions. Further, mean theory doesn't properly account for non-classical analytics in the free energy.

1.2.4 The Renormalization Group

The shortcomings of mean field theory necessitated the development of better techniques for handling critical phenomena with dimensionality below the upper critical dimension. Advancement in this field came largely through work done by Ken Wilson in the early 1970s [38, 39], work for which he was awarded the Nobel Prize in 1982.

At its core, the renormalization group approach to continuous phase transitions is a reflection of the fact that the correlation length ξ diverges to infinity at the critical point ($T = T_c$ in the case of the Ising model example). Wilson's idea was to iteratively coarse-grain the system in question, allowing for examination of the underlying physics at all length scales. The thought is that as one "zooms out" and examines larger and larger length scales, the microscopics of the system become unimportant, leading to universal behavior in critical systems due to the lack of a characteristic length scale. While the

ideas of scale invariance and universality will be discussed in more detail in the following section, this notion of the importance of the thermodynamic limit of infinite system size in critical systems remains central to renormalization group theory.

For a more concrete example of this renormalization group approach, we'll once again look the the Ising model example, following Leo Kadanoff's "block spins" method [40]. We'll begin with the standard Ising model Hamiltonian with no external field and ferromagnetic couplings between nearest-neighbor pairs of spins:

$$H_{\mathrm{IM}}(J) = -J \sum_{\langle ij \rangle} s_i s_j \ . \tag{1.9}$$

Next, we can imagine grouping each 2×2 block of spins, as shown in Fig. 1.4. We observe the net magnetization of each block (up, down, or zero) and replace the block of four spins with one coarse-grained spin s', which is set to be up, down, or randomly assigned a value depending on the net magnetization of the block of spins that s' replaces. Similarly, we replace the fine-grained couplings J with appropriate coarse-grained couplings J' to account for the new coarse-grained spins. Our rescaled Hamiltonian,

$$H_{\mathrm{IM}}(J') = -\sum_{\langle i'j' \rangle} J' s'_{i'} s'_{j'} \ , \tag{1.10}$$

will still be equivalent to the original Hamiltonian. The indices i and j have of course been changed to reflect the coarse-graining and the fact that we now have only one quarter the number of spins to consider.

This coarse-graining process is performed repeatedly, and as the blocks of spins become larger and larger (for some block size $b = \{1, 2, 4, 8, \ldots\}$), the observation scale with which we see the system increases with b. Eventually, as

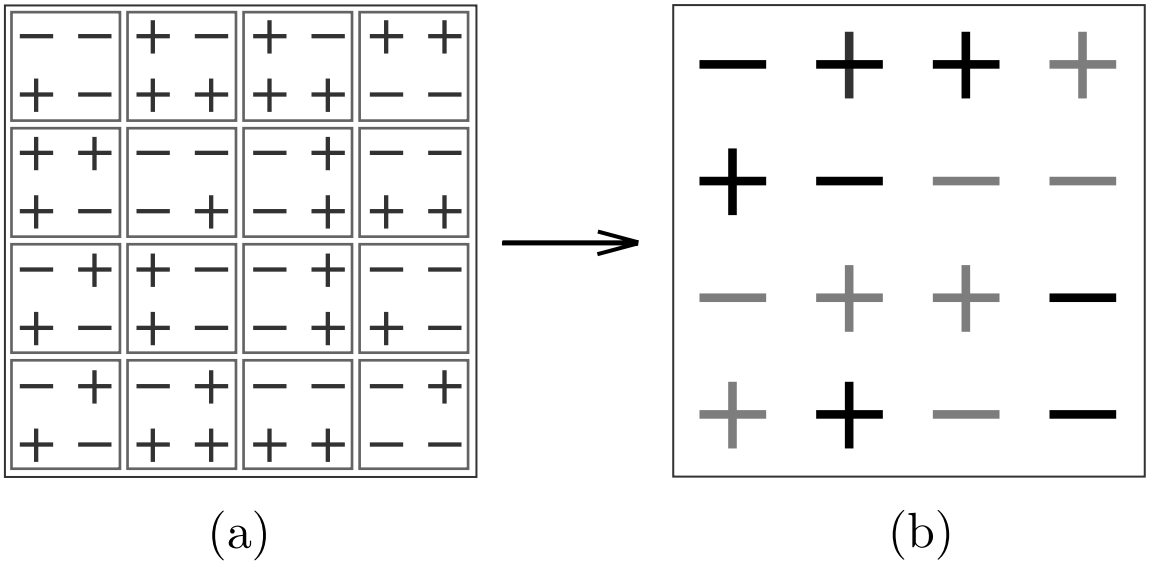

Figure 1.4: The block spin method for renormalization in the Ising ferromagnet is shown. In (a), 2×2 blocks of spins s are grouped together and replaced by a single block spin s' in (b). The orientation of s' is chosen according to the net magnetization of the four spins s making up the original block, as shown by the dark grey blocked spins in (b). If the net magnetization of a block is zero, the orientation of the blocked spin s' is chosen randomly. Those randomly chosen blocked spins are shown in lighter grey here for convenience. The example shown depicts a disordered state, so naturally there are a large number of blocks of spins with zero net magnetization. The couplings J between spins s are also adjusted to couplings J' between blocked spins s' to reflect this coarse-graining, so the Hamiltonian remains equivalent through the transformation.

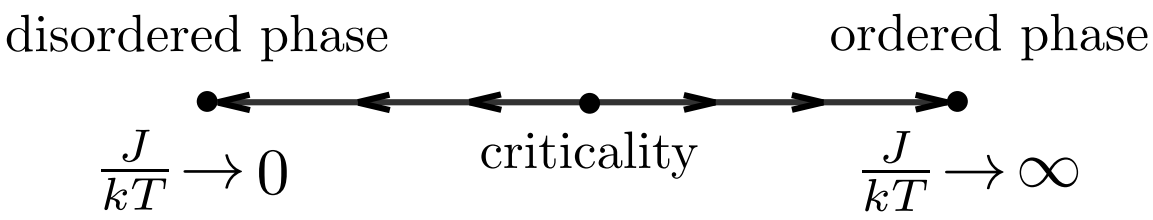

Figure 1.5: Renormalization group (RG) flow for the ferromagnetic phase transition in the Ising model. As the system is coarse-grained via the block spin method, the system will tend toward an ordered phase fixed point for low T or a disordered phase fixed point for high T. Only for $T = T_c$ will renormalization lead to criticality in the thermodynamic limit. In other words, for a given J, the RG flow for temperature T is away from the critical point, so temperature is a *relevant operator* in this RG scheme.

we approach the thermodynamic limit of infinite system size, this renormalization will lead us to a fixed point in phase space. This is referred to as *RG flow*, and is depicted in Fig. 1.5. In the two-dimensional Ising model, if $T < T_c$ this coarse-graining will lead to the $T = 0$ ordered phase fixed point, and if $T > T_c$ renormalization will give rise to the $T = \infty$ disordered phase fixed point. If, for a given J, the temperature is tuned precisely to $T = T_c$, coarse-graining will leave the system in a critical state, no matter how many iterations of renormalization are performed. Thus, temperature (or more accurately the ratio of the spin interaction strength to temperature, J/kT) determines the RG flow away from the critical point and is thus known as a *relevant operator*. Operators whose values do not matter in trying to tune the system to criticality—those operators whose RG flow leads to criticality no matter what their value—are called *irrelevant operators*.

In addition to the renormalization group, Ken Wilson and others studying critical phenomena began to use what is called an "epsilon expansion" to estimate the value of critical exponents in systems near the upper critical di-

mension d_u [41]. Sometimes referred to as a "$4-\epsilon$ expansion," since four is the upper critical dimension for the Ising ferromagnet and many other systems, an epsilon expansion is essentially a perturbation expansion in dimension d about the mean field solution in dimension d_u, using the parameter $\epsilon = d_u - d$. Thus, for dimensions near the upper critical dimension where ϵ is small, even first or second order perturbation expansions may give reasonable results. Also of significance is the fact that for any dimension $d \geq d_u$, mean field results will hold exactly. In effect, any dimension at or above the upper critical dimension will behave the same as if $d = \infty$. Worth noting here is that there is also typically defined a lower critical dimension d_ℓ, which is the dimension above which the continuous phase transition in question occurs. At or below d_ℓ, the system does not exhibit a phase transition. It should be clear that dimensionality plays a vital role in the underlying physics of these critical phenomena.

1.2.5 Implications of Scale Invariance

From renormalization group theory, we know that for critical phenomena, the thermodynamic limit becomes very important. To better understand this, we look at the correlation length. The correlation length ξ defines the characteristic length scale for a system, the length at which "things start to matter." More precisely, ξ denotes the spatial extent of correlations in the system. For a spin system, ξ will be an estimate of the linear size of the largest clusters of similarly oriented spins. Because the correlation length follows the scaling relation given in Eq. 1.8, as a system approaches criticality in a continuous phase transition, ξ increases and eventually diverges to infinity at criticality.

What this means is that at criticality there is no characteristic length scale, other than the global size of the system L if we are looking at a finite system. It is for this reason that the thermodynamic limit is so important. Only in infinite systems ($L \to \infty$) can we truly observe the full nature of criticality, a complete lack of length scales. This lack of length scales manifests itself in the form of scale invariance. Scale invariance or self-similarity in a critical system means that changing scale does not change the physics. This is precisely why repeated coarse-graining iterations of a system at criticality yields no ostensible change in the system's behavior and no RG flow away from the critical point.

One can picture this self-similarity manifesting itself in our example of the Ising ferromagnet near criticality. If we imagine an Ising ferromagnetic system at some low temperature below T_c, the system will be in an ordered phase with nearly all spins aligned. As we increase temperature toward T_c, small clusters of like spins will appear, roughly of linear size ξ. As we increase temperature further, these clusters will grow in size, and smaller clusters of spins will begin to grow inside those large clusters. In this way, we begin to have a recursive, nested structure for spin clusters, and when we finally reach T_c the entire system becomes one such cluster. In a finite system, $\xi = L$ since ξ is diverging to infinity but L is the largest length scale available to the system. If we change our observation scale and increase our total system size to $2L$, we again see that $\xi = 2L$, and the system and its physics look indistinguishable from the picture at the previous scale. We can repeatedly change our observation scale, but no matter how much we zoom in our out, the physics of the system will look the same, since ξ will be equal to the system size.

This scale invariance is a hallmark of criticality, and it has some significant

mathematical ramifications for the scaling of objects and the behavior of the system. Scale invariance uniquely leads to power law behavior and fractal objects in these critical systems. To justify this claim, we need to express scale invariance mathematically. Scale invariance for some function $f(x)$ means that if the input x is scaled by some factor λ, the value of the function is proportional to $f(x)$ itself:

$$f(\lambda x) = \lambda^{\Delta} f(x) \, . \tag{1.11}$$

This naturally leads to a power law ansatz,

$$f(x) = x^n \, , \tag{1.12}$$

since we can easily verify that a power law will have the scale invariant behavior described by equation Eq. 1.11:

$$f(\lambda x) = \lambda^n x^n = \lambda^{\Delta} f(x) \, . \tag{1.13}$$

Here $\Delta = n$ is the critical exponent governing this power law.

What this power law behavior means, among other things, is that we will see fractal objects cropping up in these systems that exhibit critical behavior. A fractal object is merely an object that has a fractal (non-integer) dimension, which follows directly from power law behavior. The interesting physics associated with these fractal objects makes the calculation of fractal dimensions and the other critical exponents for critical systems enticing to study. Further, because the thermodynamic limit plays such an important role in critical phenomena, these systems often display universal behavior, adding greatly to their merit as research topics. Universality means that the critical exponents of seemingly unrelated systems can be identical (and thus, the physics

is equivalent) if these two systems are in the same *universality class*. Universality is a reflection of the fact that microscopic differences disappear as a system is coarse-grained at criticality due to the divergence of the correlation length and the absence of a characteristic length scale.

1.2.6 Self-Organized Criticality

While systems at criticality display quite interesting behavior—namely, the power law behavior that comes with scale invariance due to the divergence of the correlation length—there remains a hurdle in that we need to be able to tune the system's parameters to capture the system precisely at criticality. While this is true for a large number of systems, there also exist systems that exhibit *self-organized criticality*. With self-organized criticality, there is no need to explicitly tune any parameters to a non-zero value to induce critical behavior. There is a subtle, implicit tuning of relaxation rates as these systems are slowly driven. As long as adequate relaxation time is allowed, these systems evolve toward critical behavior of their own accord, regardless of the initial conditions of their parameters. One interpretation is that these are systems in which the critical point is an attractor, and over time the system naturally tends to evolve into this critical state.

While this may sound unbelievable or far-fetched, there are actually numerous examples of self-organized criticality, many found in nature. Physicists have studied self-organized criticality in snow avalanches [42], earthquakes [43], ferromagnetic domain patterns [44], cloud formation [45], forest fires [46], formation of river networks via water erosion [47], evolution of species [48], fracturing in granular materials [23, 24], neuronal avalanches in

cortical networks [49], and traffic models [50]. So it is clear that self-organized criticality has numerous applications, but what is really going on? How can these systems tune themselves toward criticality?

Popular models for self-organized criticality involve cellular automata [51, 52] and avalanches via the sandpile model [53, 54]. In a nutshell, the sandpile model mimics the behavior of a pile of sand to which additional grains are constantly being added. As sand is added to the top of the pile, the slope of the pile gradually increases. If the slope becomes too large, an avalanche will occur, with sand falling from the top of the pile and being redistributed closer to the bottom of the pile. This self-correcting behavior will ensure that the slope of the sandpile remains near some critical value. Thus, the system maintains its own criticality, and the sizes of avalanches in this model are seen to follow a power law distribution, providing further evidence that this is indeed a critical system. The core idea behind this model is the slow accumulation of energy punctuated by periods of rapid energy redistribution that drives the system toward a critical state despite the lack of a tuning parameter [49].

Another landmark piece of work that follows in the same spirit as the sandpile model is self-organized Monte Carlo algorithm known as the invaded cluster (IC) algorithm [55]. The invaded cluster algorithm is adapted from invasion percolation [56]. The invaded cluster algorithm is an efficient way to sample spin systems to criticality without any a priori knowledge of tuning parameters, flipping invasion percolation clusters in order to equilibrate a spin system. To grow these clusters, a particular seed site is chosen and the bonds (couplings) of the system are given a random order. Growth of a cluster begins at the seed site and continues outward until the cluster spans or wraps around the system. At each step, the perimeter bond with the smallest ran-

domly assigned number is added to the growing cluster. After a large cluster is grown, the cluster is flipped. After a few iterations, a fractal structure is observed that follows power law behavior, as expected from a critical system.

With invasion percolation, if one studies the distribution of random numbers for selected bonds as a function of time (the number of algorithm steps), a structure of *ponds* and *outlets* emerges [57, 58]. In this framework, outlets are successive maximal bonds which separate the chain of added bonds into pools of non-maximal bonds. We can imagine the value of bonds as heights for pieces of land, and much in the same way that sand is slowly added to the sandpile, water is poured continuously onto the seed site [59]. As the water spills outward from the seed site following the invasion percolation mechanism, small pools of low value bonds and outlets (high value bonds connecting two pools) emerge. The distribution of pool sizes and outlet values can be studied, with values tuning themselves to criticality without any a priori knowledge of critical parameters like a percolation threshold. In the thermodynamic limit, the outlet values approach the critical threshold for percolation [60, 61]. Invasion percolation is perhaps the most illustrative example of how global optimization can lead to self-organized critical behavior, and this will be discussed further in the next section on percolation.

Critical behavior is also commonly found in many systems with quenched disorder. The fixed nature of quenched disorder often allows for direct mappings to well-studied problems in combinatorial optimization, which means that efficient global optimization algorithms from combinatorics and computer science can be applied to probe the rich, power law behavior of these critical systems extremely efficiently. Though this optimization is typically the product of local interactions, the global greedy nature of optimization algorithms

can lead to critical behavior and the scale invariance and power law distributions with which it is associated. In some sense, because we're optimizing globally (over the entire system), it seems logical that the only relevant length scale is the system size L. As we examine larger systems and approach the thermodynamic limit, we push this length scale toward infinity, leaving scale invariant systems with no characteristic length scale—in other words, criticality.

For instance, minimal spanning trees can be used to study highly disordered spin glass ground state [7], telecommunications networks connecting computer terminals [2], efficient circuit design [62], taxonomic reconstruction of evolutionary trees [63], or pattern recognition in image analysis [64]. The minimal spanning tree problem is one that is widely studied in computer science and forms the basis for the project covered in Chapter 2 of this thesis that examines minimal spanning trees on critical percolating clusters.

Similarly, a mapping between random ferromagnet domain walls and the a shortest paths problem forms the basis for the work discussed in Chapter 3. The shortest paths problem can also be used to find the lowest energy path of a vortex line in a disordered superconductor [65, 66], calculate efficient travel routes and optimize transportation in traffic systems [67, 68], study social networks [3, 4], model connections in world-wide web [69–71], examine current flow in resistor networks [4, 72], or optimize internet traffic delivery [73]. The goal of my work is to contribute to a better understanding of disordered systems, as well as develop better and faster algorithms for numerical simulations of disordered systems.

1.3 Prominent Models

As a brief primer for the more detailed work presented in the following chapters, we next present a few of the most iconic and fundamental models of disordered systems. These powerful models appear in countless piece of physics literature and have been used to study a wide variety of disordered systems. They are deserving of study both from a physical standpoint as well as a standpoint of efficient algorithm development and optimization. These models form a vital basis for much of the content discussed in the following chapters and the subject of disordered systems as a whole.

1.3.1 Percolation

Percolation is one of the most well-studied phenomena in condensed matter physics, being easy to simulate but with quite interesting results in the form of a percolation phase transition. In addition, invasion percolation represents one of the most well-understood examples of self-organized criticality. For a useful general-purpose reference on percolation, see Ref. [74].

Percolation comes in two varieties, bond and site percolation. We'll focus here on bond percolation, as it is a central topic in Chapter 2. In bond (site) percolation, bonds (sites) in a graph are occupied with some occupation probability $p \in [0,1]$ and left unoccupied with probability $1-p$. In practice, we can either occupy each bond independently with probability $p \in [0,1]$ or independently assign each bond a random value in $[0,1]$ and then occupy all that are less than p. These two methods are equivalent, but the latter is typically preferable from an algorithm standpoint, though this of course depends on the implementation.

The principal result of this simple model is the connectivity transition known as the percolation phase transition. If we examine a square lattice whose bonds are occupied with probability p according to bond percolation, at low p we see only small connected clusters of bonds and a generally disconnected system. However, once we raise p above some critical value p_c, we start to observe clusters that span or *percolate* across the lattice and have a linear dimension roughly equal to the system size L. Clearly, this critical percolation threshold p_c separates two distinct phases that differ in the degree of long-range order they exhibit, as illustrated in Fig. 1.6. This should be reminiscent of the Ising ferromagnet discussed previously. Instead of temperature T as a tunable parameter that governs the phase transition, the occupation probability p controls the percolation transition. Just as with the Ising ferromagnet, we can express thermodynamic quantities like the correlation length ξ in terms of how far p is from its critical value:

$$\xi \sim |p - p_c|^{-\nu} \, . \tag{1.14}$$

As with other critical phenomena, we see the divergence of the correlation length and power law behavior in critical percolation systems. As a result, critical percolating clusters have self-similar (fractal) structure. One can look at the mass M (aggregate number of sites) of clusters and see that the mass scales as a power law of the linear size ℓ of clusters:

$$M \sim \ell^{d_{\mathrm{f}}} \, . \tag{1.15}$$

Here d_{f} is called the mass fractal dimension. In two dimensions, this exponent has been proven to be exactly $91/48$, which is remarkable since exact results are so rare in disordered systems. We can also examine minimal paths on

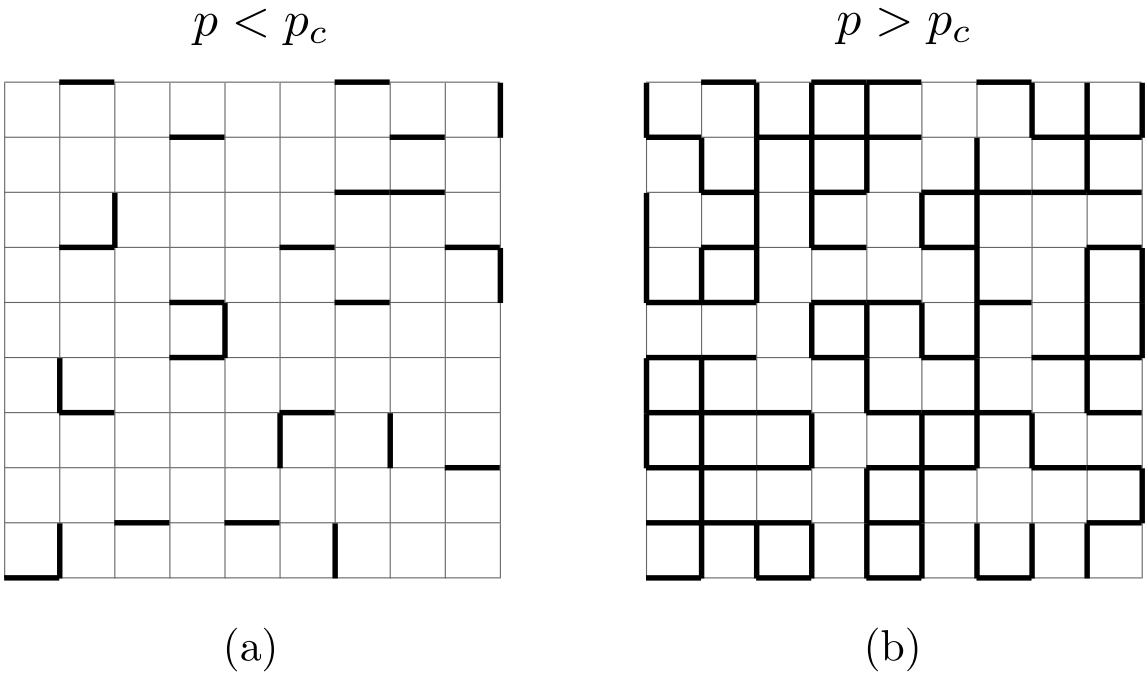

Figure 1.6: Bond percolation for a 10×10 square lattice. Occupied bonds are bolded. In (a), the occupation probability p is below the critical value p_c and we see only small, disconnected clusters. In (b), p is above p_c, and at least one system spanning or *percolating* cluster can be seen with linear size comparable to the system size $L = 10$.

percolating clusters to investigate the path length fractal dimension $d_{\min}$:

$$s \sim r^{d_{\min}} . \tag{1.16}$$

Again, r is some linear distance such as the radius of the percolation clusters.

The project covered in Chapter 2 of this thesis is primarily focused with the calculation of the critical exponent for minimal paths on percolation clusters, and will be discussed in great detail in Chapter 2. Also discussed in Chapter 2 is the relationship between Bernoulli bond percolation and invasion percolation, and the implications of invasion percolation as a form of self-organized criticality. We'll outline the salient points of this discussion here.

In Bernoulli (bond) percolation [75, 76], after determining the occupation of each edge independently, one inspects the graph to check for long-range connectivity in the form of a cluster of connected vertices that percolates, i.e., spans the graph. Examining larger and larger systems on a macroscopic scale, this percolation transition becomes clear; below some critical percolation probability p_c only small clusters are seen, but at $p = p_c$ large clusters that span the system begin to emerge. In the exploration of Bernoulli percolation, this occupation probability p is finely tuned in order to observe this critical transition and the clusters that are formed at criticality.

With invasion percolation [56], edges are assigned a weight $w \in [0,1]$ to serve as a basis for occupation if $w < p$. Edges of low weight are occupied greedily, and the invasion percolation algorithm has a termination condition. The invasion percolation algorithm consists of a growing cluster from an initial single occupied seed site. Additional sites are "invaded" by choosing the lowest weight edge from those adjacent to the growing cluster and expanding the invaded region to include this edge. This invasion percolation process can

be repeated until long-range connectivity is observed (i.e., until the invaded region percolates across the system). Because clusters formed by invasion percolation and Bernoulli bond percolation have the same thermodynamic scaling, invasion percolation allows for the simulation of critical percolating systems without having any knowledge of what the value of p_c is for a particular system. Thus invasion percolation is an example of self-organized criticality [57, 58].

1.3.2 Directed Polymer

Another prominent model with quenched disorder for which exact results are known is the directed polymer. With a directed polymer in a random potential, there exists a pinned phase which is governed by disorder rather than thermal fluctuations in dimension $d \leq 3$ or in dimensions $d > 3$ in the low temperature limit [66, 77]. The directed polymer is essentially a string in the x direction that is allowed to wander or fluctuate in the transverse $\vec{y}$ directions subject to elastic stretching energy considerations. With quenched disorder in the form of a random potential, the Hamiltonian for the directed polymer looks like

$$H_{\mathrm{DP}} = \int \left[\frac{\kappa}{2} \left(\frac{\partial \vec{y}}{\partial x} \right)^2 + V\left(x, \vec{y}\right) \right] , \tag{1.17}$$

where x is the longitudinal direction, $\vec{y}$ represents the transverse directions, κ is the an elastic energy constant, and V is a random potential [77]. This model is illustrated in Fig. 1.7 for two dimensions (one longitudinal plus one transverse direction). In the pinned phase, the behavior of the free energy is decided by minimizing this Hamiltonian, with competition between the elastic energy of the transverse fluctuations and the contributions from the random potential term.

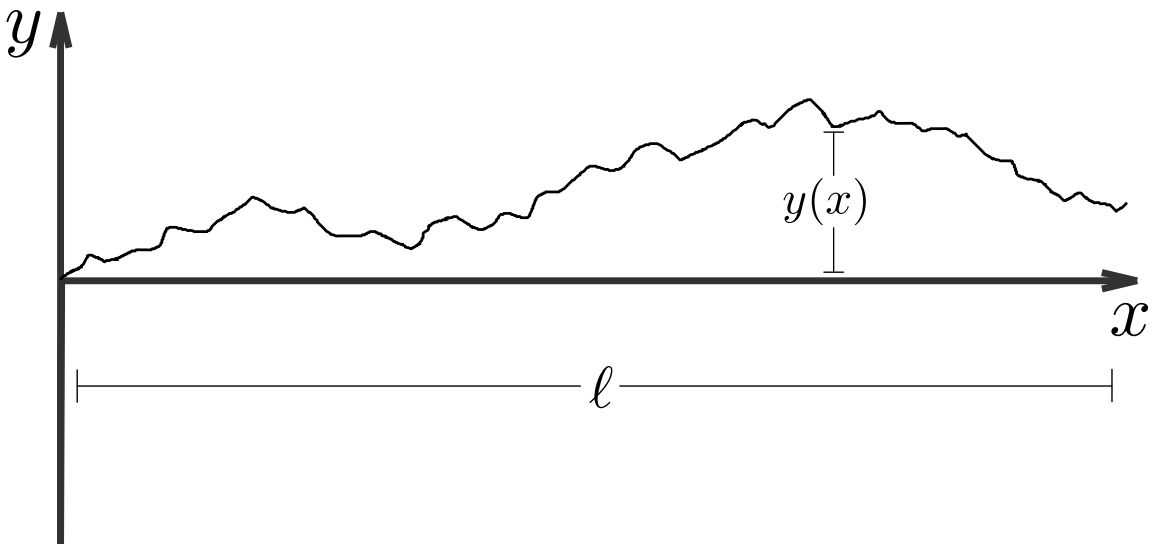

Figure 1.7: A sketch of a directed polymer of length ℓ in a random potential in two dimensions (one longitudinal direction x plus one transverse direction y).

Like many problems governed by quenched disorder and global optimization, the directed polymer exhibits power law behavior, with fluctuations in the free energy scaling as

$$\Delta F \sim \ell^{\theta} \tag{1.18}$$

for paths of longitudinal length ℓ. Similarly, fluctuations in the transverse direction scale with some wandering exponent ζ as

$$\overline{\left\langle \left| \vec{y}(\ell) - \vec{y}(0) \right| \right\rangle} \sim \ell^{\zeta} \,, \tag{1.19}$$

where the transverse fluctuations are averaged thermally as well as averaged over disorder realizations.

Amazingly enough, in two $(1+1)$ dimensions exact results are known. It was predicted that $\theta = 1/3$ [78], and this value has been proven to be exact.

The critical exponents θ and ζ have been shown to obey the scaling relation

$$2\zeta - \theta = 1 \tag{1.20}$$

in two dimensions [66, 77]. Combined with the fluctuation-dissipation theorem shown to be applicable by Huse, Henley, and Fisher [79],

$$2\theta = \zeta \ , \tag{1.21}$$

we see that $\theta = 1/3$ is exact and that $\zeta = 2/3$ as well [80–82]. This exact result is rare for disordered systems and is extremely helpful in examining related systems and drawing conclusions about universality classes. The directed polymer model is often used as a simplified basis for understanding behavior in other systems such as disordered Ising spin models, as we'll discuss in the next section.

1.3.3 Random Ferromagnet

Another well-studied disordered system is the random ferromagnet, often called the random bond Ising magnet (RBIM). As mentioned on previously in this chapter, the random ferromagnet with Ising spins and nearest-neighbor interactions can be described by the following Hamiltonian:

$$H_{\mathrm{RBIM}} = -\sum_{\langle ij \rangle} J_{ij} s_i s_j \ . \tag{1.22}$$

Here we sum over nearest neighbor pairs of Ising spins s_i and s_j, with random interaction strengths J_{ij}. Since we are dealing with purely ferromagnetic interactions, $J_{ij} > 0$ so spins tend to align with their neighbors. Because of the random quenched disorder the couplings J_{ij} represent, the random ferromagnet has rich physics.

One topic of interest in spin systems with quenched disorder such as the RBIM is predicting and studying the locations of domain walls, the interfaces separating regions of up spins (+) from regions of down (−) spins. While the ferromagnetic nature of the couplings would normally cause a trivial ground state with spins either all up or all down, by fixing spins at the system boundaries to a particular configuration, we can force one or more domain walls to be present in the system. As shown in Fig. 1.8, any location along the boundary where two adjacent spins have opposite orientations is precisely the location where a domain wall will start or end. By imposing various combinations of boundary spins, we can explore different ground state configurations of the system, thereby gaining a glimpse into the finite temperature behavior of the system and probing the multiplicity of thermodynamic states.

In minimizing the Hamiltonian to find the ground state, we can be sure that no cyclic domain walls (domain walls that surround a cluster of spins and form a closed loop) will be present, since flipping this cluster of spins (and thus removing the domain wall) would lower the total system energy. Therefore, all domain walls in the ground state will both start and end somewhere along the system boundary. This fact allows us to map domain walls in the random ferromagnet ground state to the shortest path problem [66]. A global optimization problem from graph theory, the shortest path problem seeks to find the minimal-cost path between two vertices in a random graph by seeking out the low cost edges along the path between the two vertices [83]. In this case, paths between the locations of domain wall endpoints (+/− or −/+ interfaces along the boundary) are considered, with the interaction strengths J_{ij} providing the cost for edges along these paths. This allows us to find the precise location of a domain wall between two specified boundary locations in polynomial time

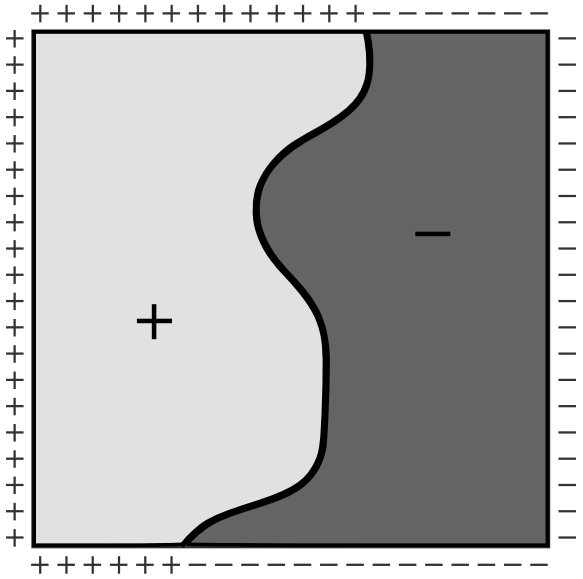

Figure 1.8: Schematic of a random ferromagnet in two dimensions with fixed boundary conditions. Having regions of differing spins on the boundary induces a domain wall, shown here as a solid black line running through the system. The domain wall separates a region of up (+) spins (shows in light grey) from a region of down (−) spins (shown in dark grey).

by using any number of efficient shortest path algorithms. This mapping is central to the work presented in Chapter 3 of this thesis, as we seek to probe domain walls for all possible boundary conditions in the two-dimensional ferromagnet.

As suggested by Huse and Henley in their work on directed polymers, the directed polymer result ($\zeta = 2/3$ in two dimensions) may potentially be applied to disordered spin systems [66]. Directed polymers are shortest paths with the stipulation that overhangs are not allowed. Through our work in Chapter 3 we investigate whether the same wandering exponent ζ from the directed polymer problem can be applied directly to the domain walls of the random ferromagnet in two dimensions. In Chapter 3 we also offer a proof that domain

walls are fully decomposable in two dimensions, as illustrated in Fig. 1.9. This decomposability is important, as it allows us to examine all 2^{4L} possible combinations of boundary conditions by merely investigating the $(4L)(4L-1) \approx 16L^2$ possible decompositions. In this work, we also delve into the discussion of thermodynamic states.

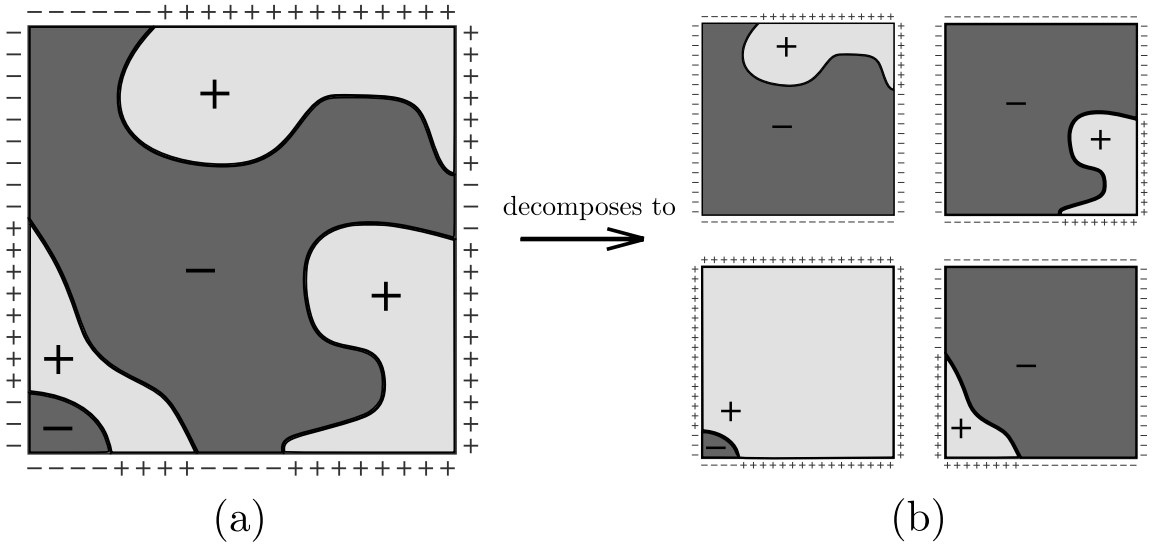

Figure 1.9: Schematic of the decomposition of domain walls in a two-dimensional random ferromagnet. The set of fixed boundary spins in (a) give rise to complicated domain walls shown as solid black lines through the system, separating regions of like spins. The regions of up (+) spins are shown in light grey, while the regions of down (−) spins are shown in dark grey. In (b) a potential decomposition of the domain walls from (a) is shown. This figure is meant merely to illustrate the concept of domain wall decomposition. A rigorous proof of the decomposability of these domain walls is provided in Chapter 3 of this thesis.

1.3.4 Random Field Ising Model

Another disordered spin system that has garnered much attention in the past forty years is the random field Ising model (RFIM), sometimes called the ran-

dom field Ising ferromagnet. The RFIM is a theoretical model whose complexity and interesting physics have piqued the interest of the scientific community. Many of the analysis techniques described here that were developed for the RFIM also have applications in spin glass systems, which will be discussed in the next section. In addition, there exists a mapping from the theoretical RFIM to the experimentally realizable diluted antiferromagnet in a field (DAFF) [84]. For a helpful overview of this model and much of what is discussed in this section, see Refs. [9, 85, 86]. The Hamiltonian for the RFIM is identical to that of the pure Ising model (Eq. 1.4) but with an added random field term h_i acting on each individual spin:

$$H_{\text{RFIM}} = -J \sum_{\langle ij \rangle} s_i s_j - \sum_i h_i s_i \ . \tag{1.23}$$

Here J is the uniform interaction strength between nearest neighbor spin pairs, and h_i is a random field that can take on both positive and negative values. Typically, one takes h from a Gaussian distribution with a mean of 0 and a variance of $\Delta^2 J^2$ so that the parameter Δ governs the degree of disorder in the system.

As is clear from the Hamiltonian, there is competition between ferromagnetic couplings and random external fields, leading to *frustration* as spins attempt to satisfy both their ferromagnetic couplings to neighboring spins and their couplings to the random external field. This means that minimizing the Hamiltonian is not straightforward, and ground state calculations are nontrivial. In fact, because of the complexity of the model and the effects of frustration, computing the partition function of the RFIM is seen to be NP-hard [87, 88]. While the details of NP-hardness will be discussed in more detail toward the end of this chapter, let us explain briefly that NP-hard problems are

the "hardest" class of optimization problems to solve. NP-hard problems have no known solutions that can be computed in polynomial time (polynomial in the size of the input problem).

While computing the partition function—which governs the behavior of the system at any temperature T—is NP-hard, the finite temperature RFIM is thought to be in the same universality class as the zero-temperature RFIM, governed by a $T = 0$ fixed point and a disorder-induced phase transition. This means that studying ground state configurations can allow us to effectively study some of the finite low temperature behavior of the RFIM where the physics is governed by the quenched disorder rather than thermal fluctuations. Luckily, the problem of calculating ground states in the RFIM proves to be a tractable problem due to a mapping to the min-cut max-flow problem of combinatorial optimization [89–91]. In the max-flow problem, a random graph of nodes and connecting bonds is considered. The bonds have flow capacities (costs), and the maximum flow that can be pushed from specially designated source node s to the designated sink node t is optimized [83]. Because of the equivalence of the max-flow and min-cut problems [92], computing the maximum $s-t$ flow is equivalent to calculating the minimum "cut" in the graph that separates s and t. The value of a cut is the total cost of the bonds whose removal from the graph partitions the graph into two disjoint sets such that s and t belong to distinct sets.

As outlined in Ref. [9], the RFIM can be directly mapped to a min-cut problem by introducing two non-physical spins $s^+ \equiv +1$ and $s^- \equiv -1$ into our spin system. Each physical spin s_i is then connected to either s^+ or s^- via a bond having cost $|h_i|$, with the choice of whether to connect to s^+ or s^- for a given spin s_i determined by the sign of h_i. If $h_i > 0$ ($h_i < 0$) then s_i is connected to s^+

(s^-). The physical spins are connected to their nearest neighbors by bonds of cost J to represent the ferromagnetic spin couplings. The non-physical spins s^+ and s^- represent the source and sink (s and t) for the min-cut problem. As illustrated in Fig. 1.10, the minimum cut can then be computed. The location of this cut marks precisely the location of domain walls in the RFIM ground state, and ground state spins orientations can be assigned based on whether each spin remains connected to s^+ or s^- after the minimum cut has been performed. Physically, the problem of computing the minimum cut becomes the problem of minimizing the energy of "broken bonds" (unsatisfied ferromagnetic couplings) along this domain wall.

Due to this mapping and efficient methods for computing maximum flows such as Goldberg and Tarjan's push-relabel algorithm [93], the RFIM has been studied extensively, especially in two and three dimensions. A continuous phase transition between a low temperature, low disorder ferromagnetic phase and a high temperature, high disorder paramagnetic phase is seen to exist in dimensions $d > 2$ due to the presence of the random field as predicted by Imry and Ma for disordered systems with random fields [94]. Below three dimensions, no low temperature ordered phase is seen (and thus no phase transition).

In three dimensions, the ferromagnetic phase exhibits rough domain walls that, akin to pinned directed polymers, have a roughness that is governed by the scaling relation

$$w \sim \ell^{\zeta} , \tag{1.24}$$

where w represents transverse fluctuations at some length scale ℓ. This roughness exponent is seen to be $\zeta \approx 2/3$ [20, 66, 95, 96] and is examined by inducing

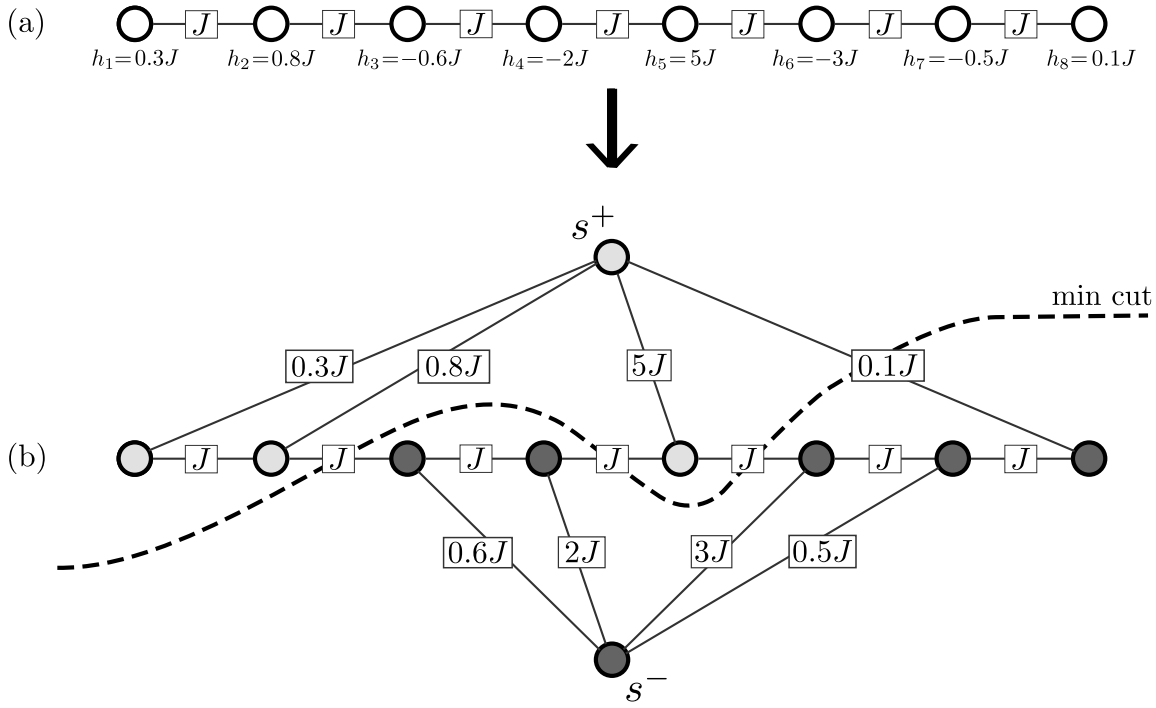

Figure 1.10: Illustration of the direct mapping from the RFIM to the min-cut problem, based on a useful diagram in Ref. [9]. In (a), the spins of a one dimensional RFIM are shown, along with the values of their external random field h_i. Based on the sign of the external field for a given spin, a bond is added connecting that spin to one of two artificially added non-physical spins, s^+ and s^-, as seen in (b). If $h_i > 0$, a bond is drawn between s_i and s^+ with value h_i. If $h_i < 0$, a bond is drawn between s_i and s^- with value $-h_i$, so that all bonds in the graph have non-negative cost values. The minimum cut is the set of bonds with the lowest total cost that, when removed, partitions the graph in (b) into two disjoint sets such that s^+ and s^- are no longer in the same set (no longer connected). This minimum cut (shown as a dashed line here) corresponds precisely to the location of a domain wall in the RFIM ground state, and spins can then be assigned up (+) or down (−) based on their connectivity to s^+ or s^- after the minimum cut has been performed. Here up spins are shown in light grey, while down spins are shown in dark grey. Though this mapping is shown here for a one-dimensional RFIM for simplicity, the procedure is analogous for higher dimensionality.

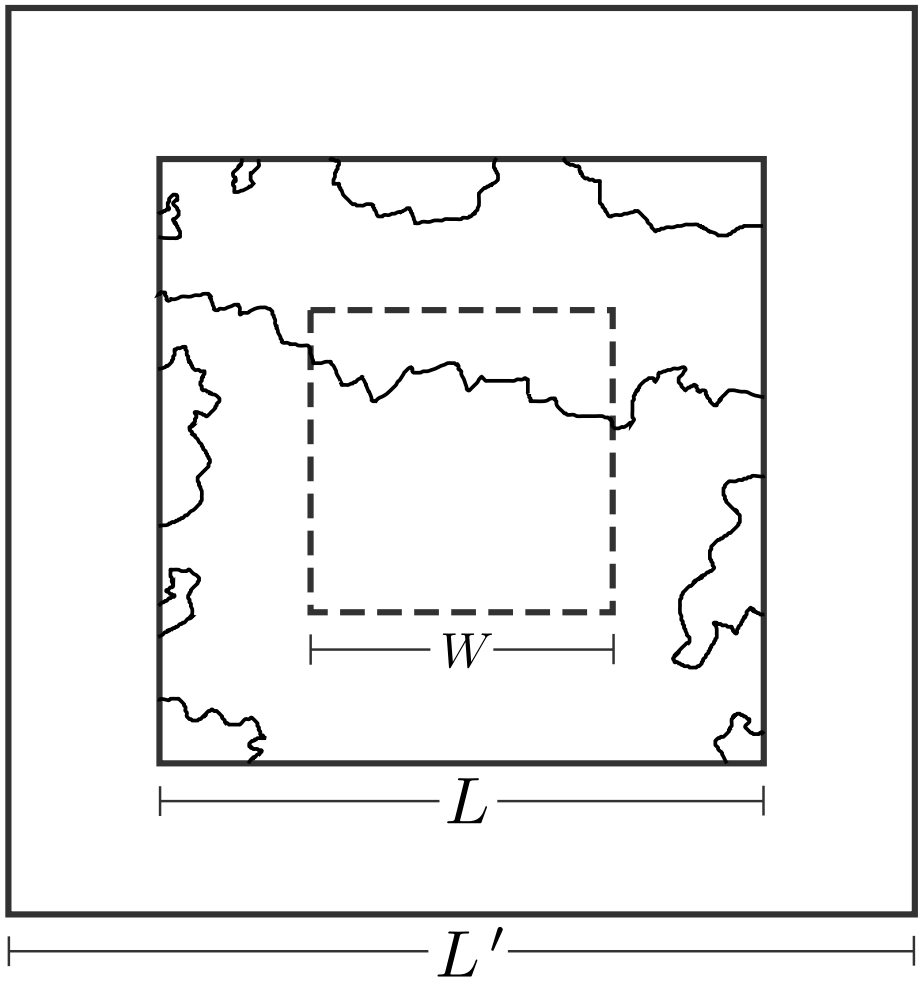

Figure 1.11: An illustration of the procedure for examining domain walls in in a fixed volume of linear size W (shown as a dashed box) as the linear size of the spin system (shown as a solid box) is increased from L to L', as depicted in Refs. [9, 10]. Deflection of domain walls away from the center window as system size approaches the thermodynamic limit implies convergence to a single thermodynamic state. This procedure is useful for investigating the multiplicity of thermodynamic states in disordered spin systems and has been applied to the RFIM, RBIM, and spin glasses.

domain walls via a change in boundary conditions [20]. Through the examination of domain wall wandering at the interior of RFIM systems, there has

been sophisticated analysis performed to investigate the multiplicity of thermodynamic states by looking at "windows" of fixed volume in the center of disordered spin systems as system sizes are increased [11, 20, 97–99]. While useful in examining the RFIM, this technique becomes paramount in characterizing the multiplicity of thermodynamic states in spin glasses, as will be discussed in the next section. In essence, spin configurations and correlation functions in a small window of fixed volume W^d are monitored as the spin system is repeatedly increased in size (from linear size L to a larger L' in a given iteration), which effectively changes the boundary conditions for the finite-volume window. This procedure is illustrated in Fig. 1.11. A convergence of these spin configurations as $L \to \infty$ implies a single thermodynamic state. This can be seen by a deflection of domain walls away from the center window as the system size is increased toward infinity. For the three dimensional RFIM, a single pair of thermodynamic states (related by a global spin flip) is observed in the low temperature ordered phase, while a single thermodynamic state is observed in the high temperature disordered phase [9, 20].

1.3.5 Spin Glass

Many of the analysis techniques developed to examine the RFIM can also be applied to another widely studied disordered system, the Ising spin glass. Immensely popular among condensed matter theorists since its inception in the 1970s, the spin glass has been something of a hot topic, with rich behavior due to its prominent features of quenched disorder and frustration. In fact, spin glasses have such complex dynamics that they are still today not well understood, and many claims about the physics of spin glasses remain controversial.

This is due in no small part to the "glassy," slow-to-equilibrate nature of the spin glass phase and its complex free energy landscape which makes numerical computations difficult. Spin glasses remain one of the most exciting and challenging topics in condensed matter, at the forefront of contemporary research. For an informative overview on spin glasses and much of the content discussed in this section, see Newman and Stein's "Spin Glasses and Complexity" [100].

The fundamental model of the Ising spin glass is given by the Edwards-Anderson Hamiltonian [101]:

$$H_{\mathrm{EA}} = -\sum_{\langle ij \rangle} J_{ij} s_i s_j \ . \tag{1.25}$$

Here s_i and s_j are Ising spins and J_{ij} are couplings between nearest neighbor spin pairs. The only difference between the spin glass Hamiltonian and the random ferromagnet Hamiltonian in Eq. 1.2 is that the couplings are allowed to be both ferromagnetic ($J_{ij} > 0$) and antiferromagnetic ($J_{ij} < 0$). The distribution for couplings is typically taken to be either Gaussian or bimodal ($\pm J$).

Competing ferromagnetic and antiferromagnetic terms in the Hamiltonian leads to frustration (similar to that of the RFIM), as spins cannot always satisfy all of the couplings linking them to their neighbors. Hence, globally greedy algorithms cannot be used to calculate ground states, and even at zero temperature these systems are difficult to study. In general, spin glasses in the low temperature glassy phase have complex free energy landscapes with many metastable states that represent local (but not necessarily global) minima. This means that equilibration times for spin glasses vary greatly, making numerical computations exceedingly difficult.

Fortunately, there are some special cases of spin glass systems that can

be studied effectively using mappings to optimization problems. For example, Newman and Stein demonstrated that invasion percolation can be used to study spin glass ground states in the strongly disordered limit [7], using this result to gain insight into the multiplicity of thermodynamic states in this limit. The ground state of the two-dimensional spin glass in zero external field can also be found in polynomial time using a mapping to a non-bipartite weighted matching problem [102] and employing an efficient matching algorithm such as the Blossom IV algorithm of Cook and Rohe [103]. This powerful mapping has stimulated numerical work and led to numerous results for two-dimensional spin glass models. Unfortunately, in three or more dimensions or in the presence of an external field, the problem of computing spin glass ground states is thought to be NP-hard [102]. To this day, the largest three-dimensional spin glass models that have been studied are of linear size $L \approx 12$ [104].

Using the short-range Edwards-Anderson model, Fisher and Huse developed their "droplet theory" to describe excitations in a thermodynamic state in Ising spin glasses [105]. In essence, droplets are clusters of spins of linear size ℓ that can flip with an energy cost that scales like

$$\Delta F \sim \ell^{\theta} \ . \tag{1.26}$$

These excitations govern the finite temperature behavior of the spin glass. Because θ is seen to be positive for $d = 3$ [106–108], droplet theory predicts a phase transition between a high temperature paramagnetic phase and a low temperature glassy phase for $d = 3$. However, the fact that θ is seen to be negative for $d = 2$ [109, 110] prohibits such a phase transition in two dimensions, since large droplets with low energy destroy the possibility of an ordered phase

at large length scales.

As a consequence of this scaling analysis, the droplet picture of spin glasses predicts a single pair of thermodynamic states, related by a global spin flip, for the low temperature spin glass phase. The operative definition for a thermodynamic state in an infinite system is a spin configuration that marks a minimum in the free energy such that no flip of a *finite* number of spins can move the system to a lower energy configuration. A useful measure for characterizing spin states is the Edwards-Anderson order parameter,

$$q_{\mathrm{EA}} = \frac{1}{N}\sum_i \langle s_i \rangle^2 \ , \tag{1.27}$$

where $\langle s_i \rangle$ denotes a time-averaged spin value. This order parameter functions as one might expect, with $q_{\mathrm{EA}} = 0$ in the high temperature paramagnetic phase and $q_{\mathrm{EA}} > 0$ in the low temperature glassy phase, which exhibits broken spin flip symmetry when no external field is applied. Since its initial proposal, there has been much support for the two-state picture of spin glasses [21, 111–114], but conclusive proof of the correctness of this theory remains to be seen.

In contrast to the droplet/scaling picture of spin glasses, an alternative and potentially conflicting mean field approach can be considered. In the infinite-range Sherrington-Kirkpatrick Hamiltonian [115],

$$H_{\mathrm{SK}} = -\frac{1}{\sqrt{N}}\sum_{i\neq j} J_{ij} s_i s_j \ , \tag{1.28}$$

each spin interacts with every other spin. Using his replica method, Parisi predicted the free energy of this model [116], and he was later proved to be correct [117]. In short, the replica method or "replica trick" employs identical copies of a system (replicas) to compute an ensemble average over n such replicas. After some calculation, the limit as $n \to 0$ is taken. In performing this

calculation, Parisi theorized that the symmetry that is broken in the spin glass phase transition is that of the replicas themselves, causing this theory to be termed “replica symmetry breaking” (RSB). After much work in the RSB picture [22, 116, 118], it is agreed that a spin glass phase transition about some critical temperature T_c correctly describes spin glass behavior in the mean field limit ($d = \infty$). Whether these RSB mean field results are applicable to finite-dimensional short-range spin glass models and whether a spin glass phase transition persists in finite dimensions is a topic of much debate, and careful numerical work is ongoing [119–121]. To add to this debate, Almeida and Thouless predict that with the Sherrington-Kirkpatrick model of Ising spin glasses, a phase transition exists even in the presence of a nonzero external field [122]. This conflicts directly with the predictions of Fisher and Huse’s droplet theory.

RSB theory also makes a bold prediction that the glassy phase is characterized by infinitely many thermodynamic states separated by infinite energy barriers, a huge divergence from the predictions of droplet theory. This means that for a given disorder realization, once a spin glass system equilibrates into one of these many possible states, it will remain in this state and cannot transition to another thermodynamic state. Clearly, this aligns with the idea of replica symmetry being broken, since a system with identical disorder realization (a replica) may find itself equilibrating to a completely different thermodynamic state. In order to better characterize the spin configurations in the RSB picture, Parisi developed an order parameter for quantifying spin overlaps [123]:

$$q_{\alpha\beta} = \frac{1}{N}\sum_i \langle s_i\rangle_\alpha \langle s_i\rangle_\beta \ . \tag{1.29}$$

This Parisi order parameter effectively measures correlation between spin states α and β. The indices α and β are called "replica indices." Is is clear from Eqs. 1.27 and 1.29 that the Edwards-Anderson spin overlap measures the self overlap of a particular spin state α with itself, as $q_{\mathrm{EA}} = q_{\alpha\alpha}$. Similarly, a value of $-q_{\mathrm{EA}}$ indicates the spin overlap of a spin state with its flipped state (related by a global spin flip).

Examining the probability distribution of this spin overlap, $P(q)$, is instructive in differentiating between the droplet and RSB pictures of spin glasses. Intuitively, $P(q)$ indicates the probability of the spin system to be in any particular thermodynamic state at a given time, averaged over disorder realizations. Thus a peak in $P(q)$ represents a pure thermodynamic state. Fig. 1.12 shows a sketch of the spin overlap distribution for both the droplet/scaling theory and RSB pictures. In droplet theory, the single pair of thermodynamic states is reflected in a pair of delta functions at $\pm q_{\mathrm{EA}}$. In RSB, large peaks at $\pm q_{\mathrm{EA}}$ appear, along with infinitely many (when averaged over disorder realizations) smaller peaks between $-q_{\mathrm{EA}}$ and $+q_{\mathrm{EA}}$, shown as a smooth curve in Fig. 1.12. This curve is representative of the infinitely many thermodynamic spin states predicted by RSB. Numerical investigation into the shape of this spin overlap distribution is one way to provide support for these spin glass pictures [124].

A recent method for probing the multiplicity of thermodynamic states involves examining windows of fixed volume W^d as boundary conditions are changed and the spin system is brought toward the thermodynamic limit of infinite system size [9–11, 97, 99]. As described in the previous section concerning the RFIM and illustrated in Fig. 1.11, one monitors the convergence of spin configurations in a fixed-volume window of linear size W at the center of the spin system as the linear system size is increased (from L to L' in a

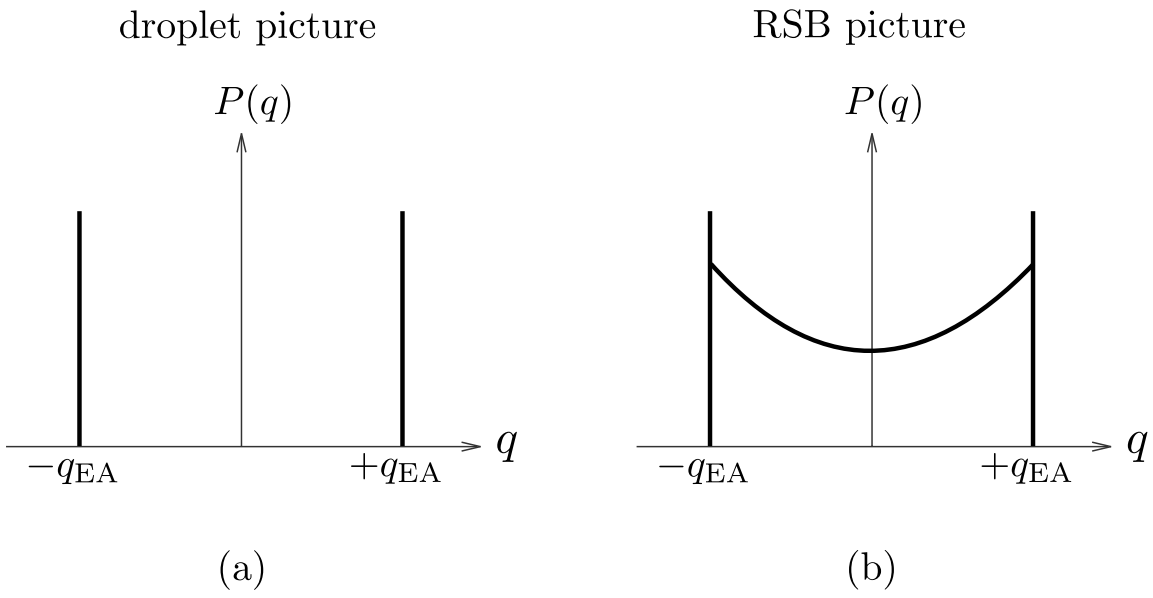

Figure 1.12: A sketch of the spin overlap distributions for the droplet picture and the RSB picture, as depicted in Ref. [11]. Here the spin overlap distribution $P(q)$ is averaged over disorder realizations. In (a), a pair of delta function peaks at $\pm q_{\mathrm{EA}}$ can be seen, corresponding to the single pair of thermodynamic states that is predicted by droplet theory. In (b), a smooth curve between these peaks reflects the infinitely many thermodynamic states predicted by the RSB picture.

given step). Convergence to a single spin configuration (up to global spin flip symmetries) indicates the presence of a single thermodynamic state (pair of states). By inducing domain walls between ground state configurations for periodic and antiperiodic boundary conditions, one can study the degree to which these domain walls wander. Of particular interest is whether these induced domain walls intersect with the window as $L \to \infty$ or whether they deflect to infinity, which would indicate the convergence of the window to a single spin configuration. Domain walls crossing the window as $L \to \infty$ would indicate that spin configurations in the window continue to change as the system size L is increased, with no convergence in the $L = \infty$ limit. This has been termed "chaotic size dependence" by Newman and Stein [125] and is in line with the RSB picture of many thermodynamic states. This type of analysis and concern for the location of domain walls at the center of an infinite system is central to the arguments and simulations on the RBIM presented in Chapter 3 of this thesis.

1.4 Statistical Analysis

Due to the complexity of disordered systems and their power law critical behavior, they are often difficult to study. Even if systems can be simulated exactly using computational methods, careful numerical analysis methods are essential. Here we'll briefly outline several analysis techniques we use and some of the theory behind them. For useful references detailing many of these statistical techniques, see Refs. [126–128].

1.4.1 Scaling Plots

When attempting to analyze critical systems with power law behavior, the thermodynamic limit is important. Thus is becomes necessary to study data taken from systems of various sizes (linear size L) and examine behavior in the limit that L is large. We require a reliable method for comparing data sets drawn from systems of different sizes, taking into account the finite size scaling that is inherent when working in a finite system rather than an infinite one, where the true scaling forms and divergence of the correlation length manifest. The general methodology of a scaling plot is to scale the x and y axes each by a different scaling factor such that we plot some functional form that is independent of system size. We do this by examining the expected scaling form for the data and developing scaling parameters to take system size L into account. Visually, this means that all of the data sets we plot should "collapse" onto same curve. Often, there is some tunable parameter (a critical exponent if our data is thought to have a power law scaling form), and the curves will converge better as this parameter approaches its true value. This is of course of central importance in determining critical exponents, which is why these scaling plots are so essential.

Suppose, for example, we're studying the mass (aggregate number of sites) of critical percolation clusters, which are expected to scale like

$$M(\ell) \sim \ell^{d_f} \ , \tag{1.30}$$

where ℓ is the linear size of a cluster. Perhaps the data we've collected is a distribution of masses of various clusters as a function of their linear size ℓ, and we're confined to a finite system of linear size L. The ratio ℓ/L is a natural choice for a scaling parameter, as this marks the point at which clusters have

linear size comparable to that of the system boundaries, which is where finite size effects should become important. Using this scaling parameter, we can expect that the mass of clusters $M(\ell)$ will scale as ℓ^{d_f} times some unknown function of the dimensionless scaling parameter ℓ/L:

$$M(\ell) \sim \ell^{d_f} f\left(\frac{\ell}{L}\right) . \tag{1.31}$$

While its exact form is not apparent (and is unimportant in this analysis), the function $f(\ell/L)$ is expected to have the same form no matter which data set is plotted. If we define new scaled parameters

$$\widetilde{M}(\ell) = \frac{M(\ell)}{\ell^{d_f}} \tag{1.32}$$

and

$$\widetilde{\ell} = \frac{\ell}{L} , \tag{1.33}$$

the nature our scaling plot becomes clear. Plotting the scaled parameters $\widetilde{M}$ versus $\widetilde{\ell}$ will provide a nice collapse for data sets drawn from various system sizes. As the parameter d_f is tuned closer to its true value, this collapse will improve. This provides a reliable method for estimating the value of critical exponents from data using finite size scaling.

1.4.2 Chi-Squared Test

In order to test and quantify how well two data sets agree, a χ^2 or "goodness of fit" parameter is often introduced (not to be confused with χ as the magnetic susceptibility for spin systems). This is useful for comparing experimental data to a proposed theoretical model, or it can be used to compare scaling plots for a variety of differently-sized systems, as will be seen in Chapter 2 of this thesis.

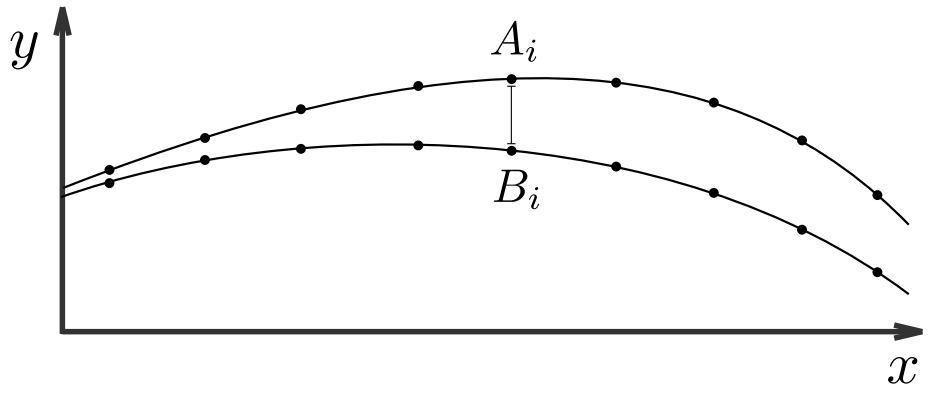

Figure 1.13: A sketch of a standard chi-squared test to quantify the degree of discrepancy between two data sets, A and B. Each data set is interpolated at a set of points $\{i\}$, and the difference between the values of the two data sets at these points is calculated. Eq. 1.34 is then used to sum the squares of these differences over all points sampled, normalized appropriately by the variances in the data sets.

To compute χ^2 for two data sets A and B, each data set is interpolated and values extracted at a set of points $\{i\}$. Here we assume the points $\{i\}$ to be independent, although we explore a form of the χ^2 measure for correlated data points as a part of our analysis in Chapter 2. The square deviation of the values of the data sets is normalized by the variance in the data sets, σ^2, and this quantity is summed over all the N points selected. We can write this χ^2 measure as:

$$\chi^2 = \sum_{i=1}^{N} \frac{(A_i - B_i)^2}{\sigma_A^2 + \sigma_B^2} \ . \tag{1.34}$$

Here σ_A^2 (σ_B^2) represents the variance in the data set A (B) at the i^{th} point. This procedure is illustrated in Fig. 1.13. The chi-squared test is a standard and widely used tool in statistics and is a powerful addition to our arsenal of analysis techniques.

1.4.3 Normal Error Bars

When making an experimental measurement or estimating a value from a numerical simulations, it is vital to quantify the degree of uncertainty associated with the measurement. Here we outline a standard method for computing the error bars of independent measurements. For a good reference on basic practice of error analysis, see John Taylor's "An Introduction To Error Analysis: The Study Of Uncertainties In Physical Measurements" [128].

The principal idea behind the standard computation of normal error bars is the central limit theorem. In essence, the average of N independent variables will follow a Gaussian or *normal* distribution provided N is large enough. This powerful statement means that no matter what distribution governs the physics of the problem, as long as the random variables are independent and identically distributed, normal error bars can be used to quantify the degree of uncertainty in numerical averages. In short, the Gaussian probability distribution has the form:

$$g(x) \sim \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right] , \tag{1.35}$$

with μ being the true mean of x and σ^2 being the variance. The standard deviation, σ, represents the root mean square of fluctuations about the mean. In essence, σ denotes the width of the distribution.

Using this fact, we estimate normal error bars about the average of N independent measurements by first calculating the sample mean $\overline{x}$ for our finite sample,

$$\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i , \tag{1.36}$$

and the sample variance s^2,

$$s^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})^2 \ . \tag{1.37}$$

From here we can easily quantify the normal error bars based on the rms fluctuations about the calculated mean of our finite sample:

$$\text{err} = \frac{s}{\sqrt{N-1}} \ . \tag{1.38}$$

1.4.4 Bootstrap Resampling

A powerful resampling method we employ heavily is called bootstrapping or bootstrap resampling. Owing its name to the idiom "pulling yourself up by your bootstraps" [129], this resampling technique proves to be extremely useful when a sample size is too small for other typical analysis methods, as it seems to produce statistics beyond the reach of the given sample size. Nonetheless, this technique does have sound mathematical basis and is widely accepted in the scientific community.

At its core, bootstrapping is repeated random sampling of a data set *with replacement*. A sample with n points is repeatedly sampled with replacement to form a new data set of n points, called a "resample." Bootstrap resampling presumes that the data set we have is the best estimate of the true underlying distribution for the data. Because of the random nature of the resampling, in a given resample some points from the original data set are sampled multiple times and some not at all, by random chance. Once N of these resamples are generated, the mean for each resample is calculated, and from the fluctuations in these N means normal error bars can be calculated according to Eq. 1.38. The bootstrap resampling process is sketched in Fig. 1.14. With computer

algorithms for bootstrap resampling, one can very quickly perform thousands and thousands of bootstrap resamplings quite easily. Since N is large, the statistics of this bootstrap resampling technique can be quite good.

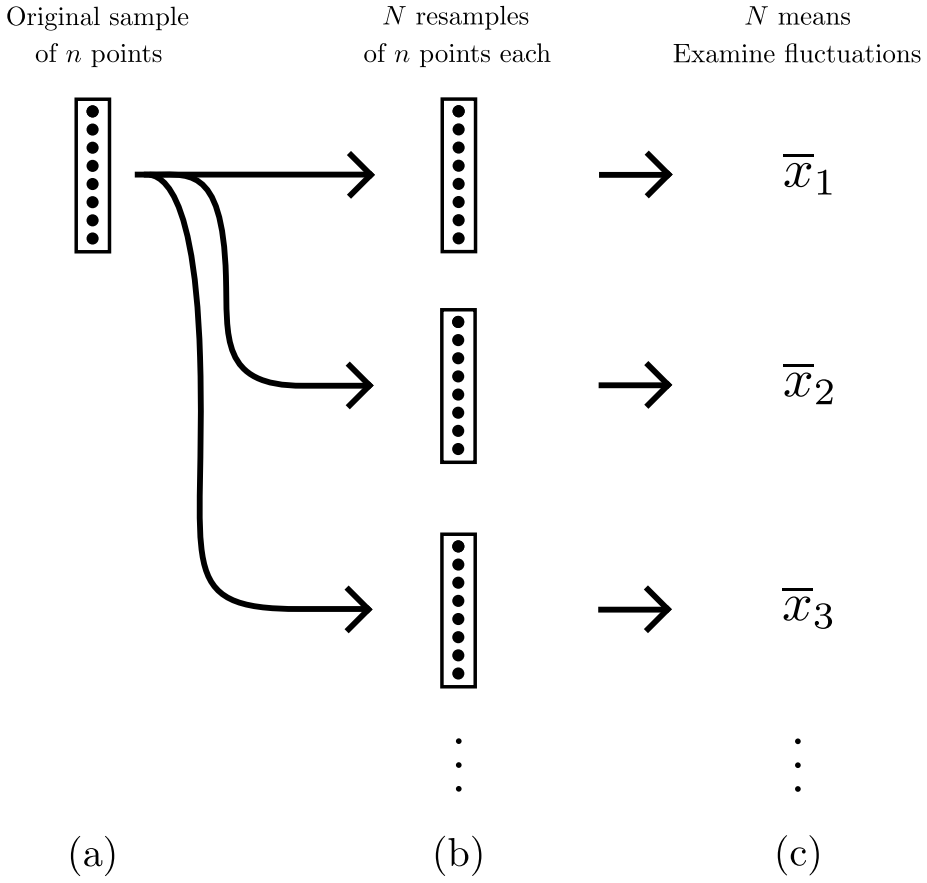

Figure 1.14: An illustration of the bootstrap resampling method. The original sample of n points in (a) is resampled with replacement n times to form a new data set, called a "resample." N such resamples are constructed as shown in (b). The mean of each of the resamples is calculated to give N means as in (c), and the fluctuations in these resampling means provides statistics for the computation of error bars.

1.5 Basics of Numerical Simulations

When studying complex systems, careful analysis techniques are essential but are rendered almost useless without efficient simulations. The ability to generate good statistics via high quantity of samples and large system sizes is of the utmost importance in precise numerical calculations. Though discussion of project-specific production algorithms will be reserved for Chapters 2 and 3, here we will review the fundamental coding languages, tools, libraries, and techniques we employ in an effort to make our implementation of algorithms as efficient as possible. We also briefly discuss the idea of computational complexity, as this will serve as a metric for benchmarking algorithm running time.

1.5.1 Programming Languages

For production code, the code that actually generates samples from which we draw statistics, we use the C++ programming language. C++ is rigorous, efficient, good for low-level precision coding, and lends itself well to objected-oriented design. For a general C++ reference, see Ref. [130]. We also used a handful of other languages such as Perl, Python, and Bash shell scripting (general unix scripting) for processing, organizing, and analyzing data files.

For plotting data and some additional analysis, we used GNU Octave [131] and gnuplot [132], both of which are offered for free under the GNU General Public License. These plotting tools allow for a high degree of customization in creating both two and three-dimensional plots of functions and data sets. Octave also has a fairly advanced scripting language replete with data fitting, built-in support for vectors and matrices, and customizable classes and func-

tions. For a comprehensive guide on scientific plotting and fitting, refer to Hartmann's "A Practical Guide To Computer Simulation" [127].

1.5.2 OrangeGrid

Instrumental in the generation of a high quantity of samples was Syracuse University's OrangeGrid. Using the high-throughput Condor computing system, the OrangeGrid boasts unmatched parallelization by offering shared resources from idle processors in computer labs across the university. With over 10,000 cores routinely available for use, computation jobs that would require several years of CPU time on a single processor can be run in a matter of days. As long as simulations are properly parallelized—typically by separating batches of different random number generator seeds to be sent to different processors—OrangeGrid allows much better statistics than would otherwise be possible with traditional computing clusters.

1.5.3 Object-Oriented Programming

Object-oriented programming employs a collection of loosely-coupled, often self-sustaining objects (realized class instances) to model a system. This is especially appropriate in studying complex system with some collective behavior or solving a problem in combinatorial optimization. For example, graphs can be composed of an aggregate of node objects and edge objects that connect or disconnect, interact, or take on random weights. Optimization algorithms can then be called by the aggregate graph objects and performed on the node and edge members. Object-oriented programming offers benefits of customizability, modularity, extensibility, and abstraction; high-level programming sim-

plifies the flow of a project, and the high degree of modularity enables easy changes to the implementation without forcing a complete code rewrite. Code becomes more portable and easier to maintain.

A set of 26 software design patterns form a standard template for object-oriented programming [133], using the concepts of data hiding, privacy, inheritance (subclasses), polymorphism (base class typing for subclasses), and composition. High-level object code that is not language-specific, these design patterns outline standard methods to efficiently solve typical coding problem. These patterns can be adapted and customized to provide effective project organization in a variety of situations. For example, the *Strategy* pattern allowed us to easily swap between various minimal spanning tree algorithms during the development of the algorithm presented in Chapter 2. The *Observer* pattern was used in the production code outlined in Chapter 3 to query the graph object when prompted and to perform analysis on the shortest paths while the system is in a given state. *Private Class Data* is a pattern that was used throughout all of our production codes to protect private data from being unintentionally manipulated by other classes during code runtime, greatly reducing the potential for bugs in the code.

1.5.4 Libraries and Tools

In order to maximize the utility of a production code, robust and sometimes complicated data structures are often needed to ensure efficient implementation. There are several freely available code libraries that we found to be especially useful in this regard. By allowing us to import fully-functional data structures rather than writing our own from scratch, code-writing overhead

is reduced considerably. In some cases, it may be necessary to modify library-provided data structures or abandon them all together, depending on the degree of customization that is needed for a given task. Here we enumerate a few of these libraries that were instrumental to our code:

C++ Standard Library - A code library that provides useful and robust versions of standard data structures such as arrays, strings, vectors, and maps (hash tables). Among its other numerous features are support for file input and output, error handling, and built-in sorting algorithms.

Boost C++ Libraries [134] - Comprehensive and well-supported libraries of C++ data structures and algorithms that are widely used in scientific research. The Boost libraries are essentially an extension of the C++ Standard Library, but a bit more advanced and customizable. For instance, the C++ Standard Library offers Fibonacci heap priority queues which we needed to implement Dijkstra's algorithm in the work presented in Chapter 3. However, to implement the more advanced Klein algorithm, we required mutable priority queues, which are provided by Boost but not the C++ Standard Library.

GNU Scientific Library (GSL) [135] - A GNU library aimed at scientific computing, complete with its own data structures and routines well-suited to scientific computing tasks such as interpolation, linear and nonlinear curve fitting, fast Fourier transforms, and random number generation. Random number generation in particular requires careful attention when performing precise numerical calculations and large numbers of iterations, as done in our numerical simulations. The GSL random number generators are thoroughly vetted and prove to be sufficient for large-scale numerical simulations.

David Eisenstat's dtree Library [136] - A template library of "dtrees" (dynamic trees), adapted from Sleator-Tarjan self-adjusting binary search trees [137]. Crucial to the efficient running time of the Klein algorithm presented in Chapter 3, these binary trees offer efficient ancestor and descendant searches, as well as updating of member values and eversion (reorienting the tree to change its root).

In addition to these code libraries, there are also a few external tools we found to be pivotal during debugging phases:

GNU Debugger (gdb) [138] - A general C++ debugging tool, useful for runtime debugging. gdb offers capabilities for manually stepping through lines of code, setting breakpoints to pause code execution at user-defined locations, and querying values during runtime (particularly useful for examining and dereferencing pointers to check for pointer errors).

Valgrind [139] - A multi-purpose debugging tool that can access and monitor system memory during runtime. This is extremely useful in detecting memory leaks, as Valgrind automatically flags potential issues and refers the user to specific lines of code that may be causing memory problems.

1.5.5 Computational Complexity

In analyzing algorithm performance for numerical simulations, it is useful to discuss "big O" notation, which denotes the asymptotic scaling of amortized running time as a function of the size of the input problem, N. This notation indicates how the running time of a particular algorithm will change as the size of the input problem is changed. For example, if doubling the size of the

input problem results in an increase in running time by a factor of four, this algorithm would have a runtime of order N^2, or $O(N^2)$. For physics simulations, N typically represents the number of sites or particles in the system being simulated, L^d for a square lattice. Because this notations denotes the asymptotic scaling, it deals only with the order of the scaling, and constant factors are ignored.

The running time scaling of a solution to a problem can be used to classify problems based on their inherent computational complexity, a fundamental topic in computer science [83, 140]. If a decision problem (a problem that asks for a "yes" or "no" answer to a posed question) of size N can be answered with a proof that can be computed in polynomial time (running time order is some polynomial function of N), the problem is said to be in P. Meanwhile, decision problems that cannot be solved in polynomial time with a deterministic algorithm but whose "yes" solutions can be verified in polynomial time are said to belong to NP (nondeterministic polynomial time).

The hardest problems in NP—those with the highest degree of computational complexity—are called NP-complete. Any problem in NP is *reducible* to any of the NP-complete problems, which have all been proven to be equivalent [141]. If problem A is reducible to some other problem B, there exists some mapping from A to B that can be performed in polynomial time, and this implies that B is *at least as hard* as A, if not harder. So if a polynomial time solution for B exists, a polynomial time solution for A also must exist. Frequently studied NP-complete problems include the Traveling Salesman Problem, the problem of finding Hamiltonian cycles, 3-coloring, and 3-SAT (satisfiability). NP-completeness has strong implications, as a polynomial time solution for any of these NP-complete problems would imply that *all* NP problems have

polynomial time solutions, definitively answering the question of whether or not P equals NP.

The hardest complexity class is called NP-hard. Problems that are NP-hard are those to which NP-complete problems are reducible, meaning NP-hard problems are at least as hard as NP-complete problems. NP-hard problems are frequently not in NP because they're not decision problems. NP-hard problems are referred to as *intractable*, and it is thought that no polynomial time solution exists to solve these problems. Many optimization problems that arise in physical contexts are seen to be NP-hard, such as the max cut problem, the computation of three-dimensional spin glass ground states, and the calculation of the partition function in the RFIM, to name a few. The intractability of these problems makes them exceedingly difficult to study properly with good statistics, although ongoing research in quantum computing may offer some hope in making certain intractable physics problems tractable [142].

Because we're concerned with the thermodynamic limit, computational complexity and algorithm running time take top priority in numerical simulations of disordered systems. Even in systems with polynomial time solutions, a critical slowing down of equilibration algorithms is seen in many systems with quenched disorder, behaving as the computational analog of the diverging correlation length at criticality and mimicking the glassy behavior of the physical systems being studied [9, 143]. For this reason, it is crucial to develop optimized algorithms, as any reduction in complexity represents a marked improvement in the number of samples of a system that can be studied as well as the size of the system that can be simulated.

1.6 Overview of Thesis

Now that we have established a historical, physical, and computational framework to serve as a backdrop for our original research, we present two projects studying disordered systems.

In Chapter 2 we present a project studying the path length fractal dimension of minimal spanning trees on critical percolation clusters. We attempt to verify a perturbation expansion developed by Jackson and Read [6] for this fractal dimension. We employ a variety of minimal spanning tree algorithms and develop our own algorithm that combines two of the most prominent of these such algorithms. A proof of the correctness of our algorithm is also presented. We address finite size scaling and provide careful analysis of numerical data. By developing our custom chi-squared test to handle correlated data, we are able to compute the fractal dimension of paths with high precision. The work presented in this chapter was published as Ref. [144].

In Chapter 3 we discuss a research project that studies the RBIM and probes the effects of boundary conditions on the deep interior of the system. We utilize a direct mapping from RBIM domain walls to a shortest paths optimization problem. Taking advantage of the decomposability of domain walls in the two-dimensional RBIM (which we also prove in this chapter), we employ a sophisticated multiple-source shortest paths algorithm developed by Philip Klein [8] to rapidly test all possible combinations of boundary conditions in a given sample. Quantities of interest are the spacing of domain walls on the interior of the system and the probability that a domain wall can be induced to pass through a particular location by changing only boundary spins. This probability is expected to follow a power law, and the critical exponent gov-

erning this behavior is calculated. The work presented in this chapter is in motion to be submitted for publication in Physical Review E.

Chapter 2

Minimal Spanning Trees on Critical Percolation Clusters

2.1 Overview of Project

In this project, the fractal dimension of minimal spanning trees on percolation clusters is estimated for dimensions d up to $d = 5$. A robust analysis technique is developed for correlated data, as seen in such trees. This should be a robust method suitable for analyzing a wide array of randomly generated fractal structures. The trees analyzed using these techniques are built using a combination of Prim's and Kruskal's algorithms for finding minimal spanning trees. This combination reduces memory usage and allows for simulation of larger systems than would otherwise be possible. The path length fractal dimension d_s of MSTs on critical percolation clusters is found to be compatible with the predictions of the perturbation expansion developed by T. S. Jackson and N. Read [6].

2.2 Motivation

The statistical physics and dynamics of disordered physical systems naturally leads to the study of fractal geometric objects. Physical systems with quenched disorder, i.e., those with fixed random heterogeneities, often have power law correlations at large scales or interfaces that are fractal or self-affine. These structures can result from some global optimization problem that has connections with graph theory. For example, Dijkstra's shortest path algorithm [83, 145] can be used to find the lowest energy path of a vortex line in a disordered superconductor [65, 66], and these paths are self-affine. Excitations in a disordered XY model in two dimensions [17, 18], domain walls in spin glasses [19], and boundaries between drainage basins [14] are examples of physical objects with fractal dimension that are found by global optimization.

A structure with fractal scaling that arises in physical contexts is the minimal spanning tree (MST). One such context is a highly disordered Ising spin glass. Newman and Stein showed that for the strongly disordered limit, the problem of finding a ground state can be directly mapped to finding an MST [7]. This mapping can be used to investigate the multiplicity of ground states in the thermodynamic limit. Minimal spanning trees have other applications such as in transportation networks connecting cities [1], telecommunications networks connecting remote computer terminals [2], efficient circuit design [62], taxonomic reconstruction of evolutionary trees [63], and pattern recognition in image analysis [64].

A minimal spanning tree is a structure that connects a set of nodes with minimum total cost. This structure is defined for a weighted graph $G = (V, E, w)$ where V is a set of vertices (or nodes), E is a set of edges that connect vertices,

and w is a weight function, with each edge $e \in E$ having weight $w(e)$. A spanning tree is a loopless connected set of edges that includes all the vertices in V. The minimal spanning tree is the spanning tree T that minimizes the total weight

$$w(T) = \sum_{e \in T} w(e) \,. \tag{2.1}$$

This is a well-known problem in computer science and combinatorics. See Ref. [146] for a general overview of MSTs and MST-finding algorithms.

A notable fact about the MST is that the minimal tree is determined only by the numerical ordering of the weights, i.e., it is otherwise independent of their value. So there is a large invariance or universality for these structures; their geometry is independent of the disorder distributions. As long as the weights $w(e)$ are independently drawn from the same distribution (independent identically distributed or i.i.d. weights), the edges can be sorted in order of increasing weight. This ordering alone determines the tree. Two physical problems with wholly different distributions of quenched disorder have MSTs with equivalent statistics.

Recently, Jackson and Read carried out an analytical calculation for the fractal dimension d_s of paths in MSTs [6, 147]. They developed a perturbation expansion for d_s for MSTs on critical percolation clusters in d dimensions, obtaining the result

$$d_s = 2 - \frac{\epsilon}{7} + \mathcal{O}(\epsilon^2) \,, \tag{2.2}$$

where $\epsilon = 6 - d$ and $d = 6$ is the upper critical dimension [6]. In general, disordered systems are difficult to analyze and rarely yield quantitative analytic results. This prediction therefore provides a strong motivation for more precise computation of fractal dimensions in spanning trees, in particular, di-

mensions of the trees on spanning percolation clusters. We note that it is unclear whether the fractal dimension of these trees is affected by being constructed on spanning percolation clusters as opposed to a whole lattice. The work presented in this paper seeks to numerically compute d_s in dimensions $2 \leq d \leq 5$ for comparison with Eq. (2.2). This calculation employs a combination of memory-saving techniques to simulate large systems as well as careful data fitting procedures to obtain precise estimates for d_s in the limit of infinite system size.

Our final calculations for d_s yield values of 1.216(1) for $d = 2$, 1.46(1) for $d = 3$, 1.65(2) for $d = 4$, and 1.86(4) for $d = 5$ (refer to Table 2.1). We develop and utilize a χ^2 test that accounts for the scale-invariant correlations found in the data, allowing for an improved χ^2 measure and robust estimates of the uncertainty in the effective exponent at scale L, $d_s(L)$. We then use linear and nonlinear least squares fitting to extrapolate to the infinite system size limit. We find the numerical results to be of higher precision than previous calculations and compatible with Eq. (2.2), though more conclusive confirmation requires improved numerical statistics and higher order analytic work. We emphasize that the analysis procedure used here is generalizable and could be useful for other work dealing with disordered systems.

2.3 Model and Algorithms

To model MSTs on percolation clusters, we simulated hypercubic lattices with L vertices per side with periodic boundary conditions, giving L^d total vertices and dL^d total edges. Edges are independently given a weight randomly distributed on the interval $(0,1)$, where $w(e)$ is represented by a double precision

number. Very rarely two edges are assigned the same weight. In this case, a new random weight is generated for one of these edges until a weight is generated that does not match any previously assigned weight. An important distinction to note is that rather than seeking to find a tree that spans all of the vertices in these hypercubic lattices, we seek to find the MST on a percolation cluster that wraps around the periodic lattice (in any of the d directions) at the threshold of percolation. Thus the MST-finding algorithms are stopped when the tree contains a subset of vertices that wraps around the system instead of including all L^d vertices of the graph. The final object of interest, the MST on a critical percolation cluster, contains only a small subset of the set of vertices V in the lattice.

In order to avoid confusion, we first note a dual usage of the term spanning. For an MST, spanning means that the tree includes all vertices in the graph for which the MST is found. In the context of percolation theory, the term refers to a cluster that is spanning or percolating around the lattice. We use spanning in both senses, with the correct sense implied by the context.

One naive approach to finding the MST on a graph is to iterate through the list of all spanning trees and select the tree with the lowest total weight. This approach might work on a small finite graph, but the number of trees grows exponentially with L^d. In order to analyze properties of MSTs in the thermodynamic limit of infinite system size, a more efficient MST-finding algorithm (both in terms of running time and maximum memory requirements) is needed.

2.3.1 Bernoulli Percolation Versus Invasion Percolation

As background for the algorithms used to find MSTs, it might be helpful to briefly review the relationship between invasion percolation and Bernoulli percolation. In Bernoulli (bond) percolation [75, 76], edges in a random graph have a probability p to be occupied, and a probability $1-p$ to be unoccupied. After determining the occupation of each edge independently, one inspects the graph to check for long-range connectivity in the form of a cluster of connected vertices that percolates, i.e., spans the graph. On a periodic hypercubic lattice, one definition of a percolating or spanning cluster is a cluster that wraps around the lattice along one or more of the d spatial dimension axes. Such a cluster contains a loop that cannot be contracted to a point. Note that Bernoulli percolation is equivalent to assigning weights $w(e)$ on the interval $(0,1)$ and occupying only those edges with weights $w(e) \leq p$. We will refer to this as the alternative definition of Bernoulli percolation.

Examining larger and larger systems on a macroscopic scale, this percolation transition becomes clear; below some critical percolation probability p_c only small clusters are seen, but at $p = p_c$ large clusters that span the system begin to emerge. Aizenman refers to these large critical clusters as incipient spanning clusters (ISCs), a term which is closely related to, but distinct from, the incipient infinite cluster (IIC) [148, 149]. In the exploration of Bernoulli percolation, this occupation probability p is finely tuned in order to observe this critical transition and the clusters that are formed at criticality.

An alternate approach to percolation is invasion percolation [56]. Invasion percolation is a procedure that greedily occupies edges of low weight w and has a termination condition. Invasion percolation consists of a growing cluster C

that is a subset of E. This cluster has a set of adjacent edges ∂C. The cluster C begins as a single occupied seed site. Additional sites are "invaded" by choosing from ∂C the lowest weight edge, $e_l(\partial C)$, and expanding the invaded region to include this edge, by adding $e_l(\partial C)$ to C. This invasion percolation process can be repeated until long-range connectivity is observed (i.e., until the invaded region percolates across the system).

There has been much discussion [57, 58, 150] on how to relate the clusters of invasion percolation to those of ordinary Bernoulli percolation. There seems to be a strong reason to believe that in the limit of infinite system size, infinite connected clusters created using both of these percolation methods should obey the same scaling relations, meaning that the fractal path length dimension d_s should be the same for both methods. Invasion percolation is extremely useful because it allows for the simulation of critical percolation systems without having any knowledge of what the value of p_c is for a particular system. Thus invasion percolation is an example of self-organized criticality [57, 58].

In our particular case, we're interested in analyzing the MSTs that lie on an ISC. In other words, we want to find the MST for the ISC. We'll refer to this final object as the MTISC (minimal tree for an incipient spanning cluster). Here we use two algorithms, Kruskal's algorithm [151] and Prim's algorithm [1]. While Prim's algorithm is similar to the invasion percolation process described above [152], Kruskal's algorithm is related to Bernoulli percolation due to the global nature of the algorithm. However, neither algorithm actually requires the choice of an occupation probability p as a parameter, and consequently both processes exhibit self-organized critical behavior, as the growth is stopped when a tree is found that wraps [55]. For a more precise definition of these algorithms, refer to Appendix A. We next present less formal descrip-

tions of these algorithms.

2.3.2 Kruskal's Algorithm

Kruskal's algorithm is an MST-finding algorithm that considers all edges of a graph. In Kruskal's algorithm, we grow a forest of many small trees, merging small groups of trees into larger trees and eventually identifying one of these large trees as the MTISC, terminating the algorithm. At the start of Kruskal's algorithm, each vertex is its own isolated tree. All of the edges that are not yet part of a tree are sorted, and the edge with minimal weight is selected, excluding edges that would form non-wrapping loops. When an edge is selected, the trees containing each of its vertices are joined into a single tree. This process continues until a tree grows big enough that it wraps around the periodic lattice. At this time, this tree is identified as the Kruskal's MTISC, T_K. This process is illustrated in Fig. 2.1.

Kruskal's algorithm bears similarity to Bernoulli percolation in that it is a non-local algorithm that requires information about all edges of the graph. If we consider a graph with edges whose weights are distributed evenly on $(0,1)$ and run Kruskal's algorithm until we first examine an edge with weight $w > p_c$, the sites of T_K are identical to those obtained from performing Bernoulli percolation on the same graph with occupation probability p_c. Thus T_K is the MST on the Bernoulli percolation cluster. Given the Bernoulli percolation cluster, this MST is that formed by using the weights $w(e)$ in the alternative definition of Bernoulli percolation.

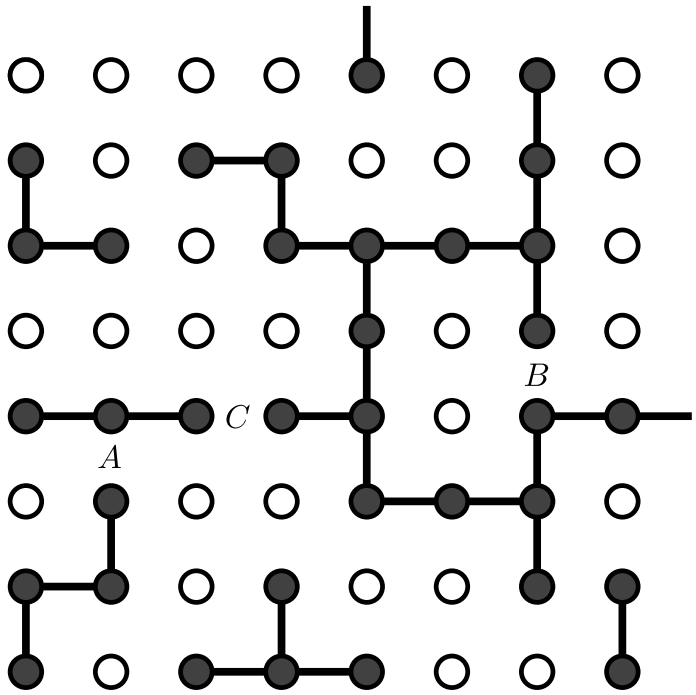

Figure 2.1: A sample iteration of Kruskal's algorithm on an eight by eight periodic square lattice ($d = 2$). At a given step, the state is a forest of trees, which includes isolated sites (open circles) and larger trees (connected solid circles). During each step of the algorithm, the edge with the lowest weight is selected from the remaining unselected edges. For example, if edge A is selected, the trees containing either endpoint of edge A are merged into a single tree. If edge B is selected, its addition would form a non-wrapping loop (forbidden cycle), so edge B is not added to any tree and is removed from future consideration. If edge C is selected, its addition would form a wrapping loop (allowed cycle), so the algorithm is terminated. The tree containing the endpoints of edge C is then the Kruskal's MTISC, T_K.

2.3.3 Prim's Algorithm

Prim's algorithm is equivalent to algorithms for loopless invasion percolation [152]. At the start of the algorithm, the MST consists of a single seed site. This tree grows outward from this vertex through the examination and conditional addition of adjacent edges, as edges adjacent to the growing tree are examined. At any given iteration, the minimal weight adjacent edge is selected and incorporated into the tree, excluding edges that would lead to non-wrapping loops. The check for which edges form a loop is simple: if both ends of the edge are in the current tree, a loop would be formed. To check whether a loop is wrapping or non-wrapping, the algorithm assigns to each vertex a a displacement $\vec{r}_a$ from the origin. This displacement is found for a newly added vertex b by adding a vector displacement $\vec{e}_{ab}$ for the edge to the vector displacement of the end of the edge (site) that is already in the tree $\vec{r}_a$. Thus for a newly added site b we have $\vec{r}_b = \vec{r}_a + \vec{e}_{ab}$. When a loop is encountered, this new site b will already have a vector displacement $\vec{r}'_b$ defined, since this site is already in the tree. If the relative displacement $\left|\vec{r}_b - \vec{r}'_b\right|$ between these two labelings is greater than L, the loop formed is a wrapping loop. When this wrapping edge is examined, the algorithm is terminated, and this final tree is identified as the Prim's MTISC, T_P, as shown in Fig. 2.2. T_P is thus the MST on the invasion percolation cluster, given an initial seed site. Note that this MST or invasion cluster may depend on the seed site.

To increase efficiency when generating MTISCs, Kruskal's and Prim's algorithms can be engineered to save memory by creating edges and weights only as needed during the execution of the growth algorithm rather than at the start [153, 154]. The memory saved in the Prim's method is much larger and

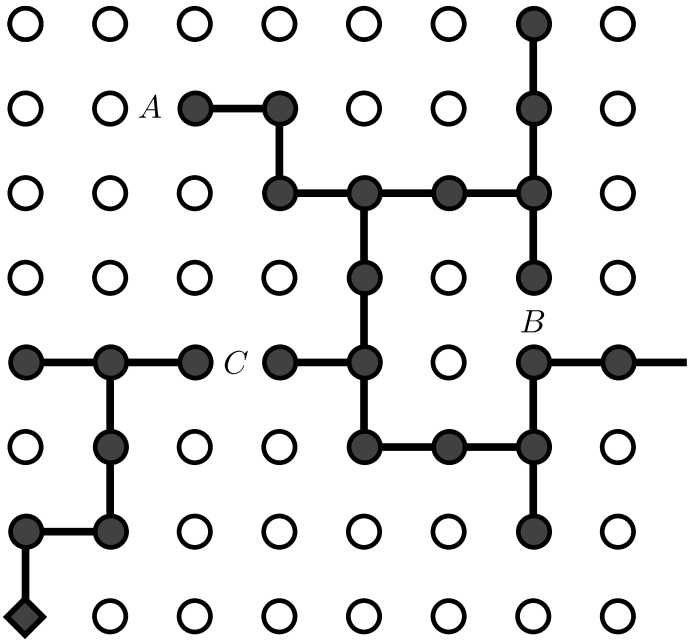

Figure 2.2: A sample iteration of Prim's algorithm on an eight by eight periodic square lattice ($d = 2$). The tree is represented by connected solid circles and lines, while unoccupied sites are shown as open circles. The diamond-shaped vertex at the bottom left of the lattice is the initial seed site v_0. During each step of the algorithm, the edge adjacent to the current Prim's tree with the lowest weight is selected. If edge A is selected, it is added to the Prim's tree, along with the endpoint of A that is not already in the Prim's tree. If edge B is selected, its addition to the Prim's tree would form a non-wrapping loop (forbidden cycle), so edge B is not added to the tree and is removed from future consideration. If edge C has the lowest weight and is selected, its addition would cause the Prim's tree to wrap around the periodic lattice (adding an allowed cycle), so the algorithm is terminated, leaving the current Prim's tree as the Prim's MTISC, T_P.

leads to an overall more memory efficient method because Prim's algorithm only requires the growing of a single tree rather than a large forest of trees. Due to this decreased memory usage, it is possible to simulate larger systems with Prim's algorithm than with Kruskal's algorithm.

2.3.4 Two-step Method

We used a two-step method, combining Prim's and Kruskal's algorithms in order to take advantage of the increased efficiency afforded by Prim's algorithm when selecting a typical MTISC. We first generate a tree T_P using Prim's algorithm and then apply Kruskal's algorithm to this tree, ensuring that the MTISC obtained has the same scaling as the MTISC that would be obtained by increasing p large enough to form a forest, one of whose trees is the final tree T_K. The data presented in this paper comes from analysis of MTISCs generated by the two-step method, as well as comparisons with the intermediate data from T_P, before Kruskal's algorithm is applied to T_P.

While Prim's algorithm in general takes less time and memory to find an MTISC than Kruskal's algorithm, the two algorithms do not find exactly the same MTISC [155]. T_K and T_P are not completely equivalent because T_K is an MST on the Bernoulli percolation cluster, while T_P is an MST on an invasion percolation cluster. As shown in detail in Appendix B, either T_K is a subset of T_P, or they do not intersect. We are interested in constructing an MTISC that is either identical to T_K or scales the same as T_K. The non-local greedy edge selection of Kruskal's algorithm guarantees T_K to have the same sites as a Bernoulli percolation cluster.

A technical detail to note about the two-step method is that in the final

stage of Prim's algorithm when a wrapping edge is selected, we add this edge to the final Prim's tree T_P. This is done so that when Kruskal's algorithm is applied to T_P, there will be a wrapping edge to satisfy the termination condition of Kruskal's algorithm. While the inclusion of this edge temporarily destroys the tree-like nature of T_P, its inclusion is essential.

Because Prim's growth begins at a random seed site that does not necessarily belong to T_K, often the T_K is a subset of T_P. This of course depends which vertex v_0 is used as the seed site for Prim's growth, as the Prim's MTISC is a function of its seed site, $T_P(v_0)$. Executing Kruskal's algorithm in the second step of this method serves to "trim off" the part of $T_P(v_0)$ that includes the seed site v_0 and that does not belong to T_K. The tree derived from this procedure is termed the two-step MTISC, T_2. This method is illustrated in Fig. 2.3.

We show in Appendix B that this two-step procedure yields the Kruskal's MTISC, i.e., $T_2 = T_K$, in all cases except when there naturally arise multiple disjoint ISCs on the lattice at criticality. By direct simulation of these systems, we observed that $T_2 \neq T_K$ about 0.2% of the time in two dimensions, 1% in three dimensions, and 5% in four dimensions. Five-dimensional samples were not compared due to the large memory demands of simulating minimal spanning forests in five dimensions.

We suppose by standard scaling that in the case where $T_2 \neq T_K$, T_2 and T_K should have the same path length fractal dimension d_s. We simulated samples in this case for systems up to size 512^2, 64^3, and 32^4. Examining scaling collapses as in Fig. 2.4, we observed similar values of d_s for T_2 and T_K in two, three, and four dimensions. Data for T_K in five dimensions was not obtained due to the large memory demands of simulating minimal spanning forests in five dimensions. The memory requirements for the two-step method allowed

us to simulate systems of size 2048^2 within roughly 2GB of memory, 256^3 within 1.5GB, 64^4 within 1GB, and 48^5 within 2GB.

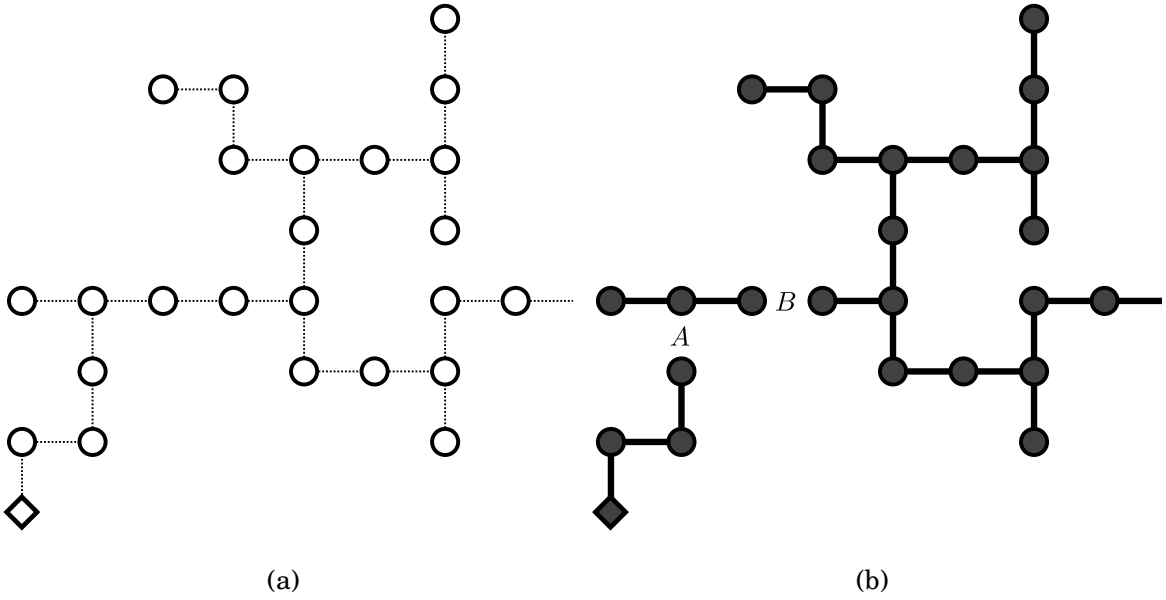

Figure 2.3: Illustration of the two-step method. This method grows a candidate MTISC using the Prim's method and then trims the tree using Kruskal's algorithm. (a) depicts T_P (plus the final wrapping edge) which is given as input to Kruskal's algorithm. During construction of this tree, only edges adjacent to the tree are tested. The potential connectivity of the unoccupied sites is explicitly shown using open circles and thin lines, and for reference the Prim's seed site v_0 at the bottom left of the lattice is represented by a diamond. (b) shows the state of the graph after running 23 steps of Kruskal's algorithm on T_P, with trees being represented by connected solid circles. If edge B is selected before edge A, i.e., $w(B) < w(A)$, the Prim's seed site v_0 will not be part of T_K, and the disconnected portion of the graph containing the Prim's seed site v_0 will be trimmed off.

2.4 Numerical Results

2.4.1 Methods for Scaling Analysis

To find the fractal dimension of the minimal trees on spanning percolation clusters, we compute the Euclidean distance r and path length s between some origin on the MTISC and other points on the MTISC. Accurately determining the scaling in the limit of large clusters requires taking into account lattice effects, finite size effects, and statistical uncertainties. Given any two points on a tree, there is a unique path between the two points, so that the path length is easily defined. The choice of endpoints for the paths is chosen in a natural fashion for each tree.

For trees constructed using Prim's algorithm, the origin is taken to be the seed site where cluster growth begins. The set of paths connecting the origin to all other points in the tree is used in the statistics. For each tree T_P, we find $n_P(s)$, the number of paths of length s that start at the origin. We use $r_P(s)$ to indicate the average over all paths of length s of the Euclidean distance $|\vec{r}|$.

After then running Kruskal's algorithm on the Prim's tree to find the trimmed MTISC T_2, a random origin is chosen on the trimmed tree. All paths from this new origin are used to find both $r(s)$ and $n(s)$, the averaged Euclidean distance and number of paths. In order to reduce the amount of data stored, the full set of N samples is grouped into sets of N_b batches of uniform size N/N_b. The batch-averaged quantities of $r(s)$ and $n(s)$ for these trimmed trees are calculated and stored. Here we use $r(s)$ and $n(s)$ to refer to data generated by the two-step algorithm. As a comparison, we also looked at the averages for Prim's trees, before trimming.

We assume that the paths on the tree are well described as fractal. The scaling of the sample-averaged $r(s)$ will then follow the relation

$$r(s) \sim s^{1/d_s} \ . \tag{2.3}$$

If we write L as the length of one side of a hypercubic system, then s/L^{d_s} is a natural scaling parameter, and we expect to see finite size effects near $s/L^{d_s} = 1$, as paths of this length approach the size of the system. The standard one parameter finite size scaling assumption is then that $r(s)$ will scale like s^{1/d_s} multiplied by some unknown function of the argument $s^{1/d_s} L^{-1}$. The form of the scaling hypothesis is that

$$\begin{aligned} r(s) &\approx s^{1/d_s} f\left(\frac{s^{1/d_s}}{L}\right) \\ &\approx s^{1/d_s} g\left(\frac{s}{L^{d_s}}\right) \ , \end{aligned} \tag{2.4}$$

where the scaling functions $f(\omega)$ and $g(\omega)$ behave as $f(\omega) \approx c_1$ for some constant c_1 for $\omega \ll 1$, $f(\omega) \approx 0$ as $\omega \to \infty$, $g(\omega) \approx c_2$ for some constant c_2 for $\omega \ll 1$, $g(\omega) \approx 0$ as $\omega \to \infty$. For more compact formulas, we define the scaled dimensionless variables

$$\rho = \frac{r(s)}{s^{1/d_s}} \ , \tag{2.5}$$

$$\omega = \frac{s}{L^{d_s}} \ . \tag{2.6}$$

Using this scaling hypothesis, we can make estimates for d_s by plotting data for multiple system sizes on the scaled axes of ρ vs. ω. If we tune the parameter d_s, we see that the curves for various sizes L_i can be made to collapse well near an estimated best value of d_s (see Fig. 2.4). While this method

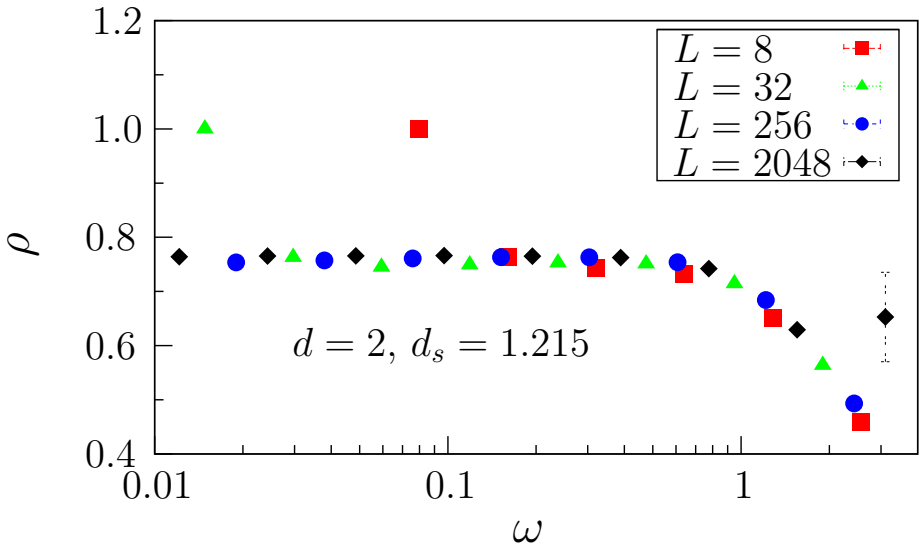

Figure 2.4: Scaling collapse for $d = 2$ with $d_s = 1.215$, which appears to be moderately close to the true value of d_s judging by eye from the goodness of the collapse. Note that error bars are smaller than the symbol size for all points except the right-most point.

allows us to get a fair idea of this exponent d_s, it relies on subjective estimations of curve collapse and for that reason is not ideal when attempting precise estimates.

To have a better estimate for d_s and to estimate the uncertainty in this estimate, we have implemented an automated fitting procedure. This procedure determines an effective exponent $d_s(L)$ derived from data for samples of size L and of larger size $2L$ by minimizing a "goodness of fit" parameter χ^2 at each scale L. The only input to this procedure is an estimate of s_l, the small path data cutoff, which is discussed in Appendix C. We then extrapolate $d_s(L)$ for $L \to \infty$ to get our best estimate for d_s.

The key part of this calculation is to choose a robust and reliable measure for χ^2. This allows us to quantitatively measure how well the data for

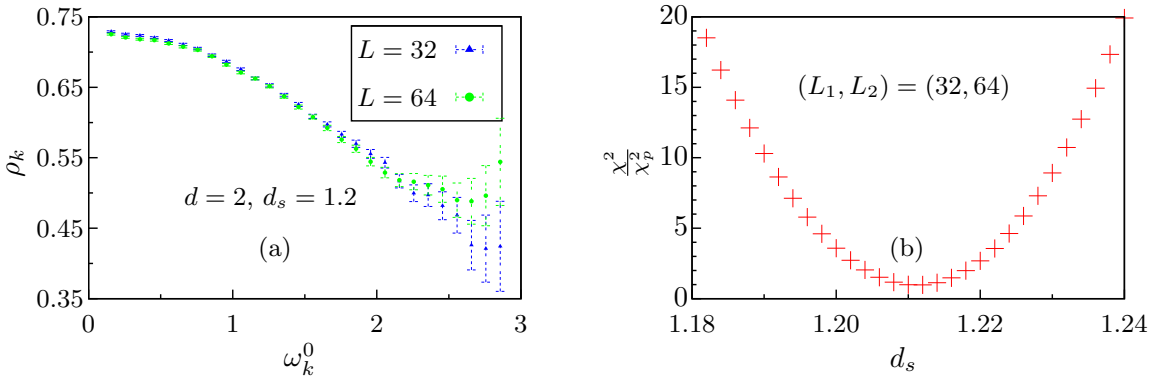

Figure 2.5: A sample collapse for two systems of size $L = 32$ and $L = 64$ in dimension $d = 2$. (a) Shows the comparison of the two systems at interpolated points ω_k^0 for a value of $d_s = 1.2$. Comparing the value of $\rho_k(L_1 = 32)$ indicated by the blue (dark gray) triangles and $\rho_k(L_2 = 64)$ indicated by the green (light gray) circles at each of these points allows for the calculation of $\chi^2(d_s; L_1, L_2)$. (b) Shows χ^2 compared with an expected estimate χ_p^2 as a function of the fitting parameter d_s for the same pair of systems. Though we determine final error bars by resampling, the χ^2 model is shown for comparison.

a given pair of system sizes collapses as a function of the parameter d_s. We seek a value of d_s for which this χ^2 is minimized (Fig. 2.5), though the magnitude of our final error bars are determined by resampling. Our definition of χ^2 must allow for non-uniform correlations in fluctuations of $r(s)$ between different values of s, discrete lattice effects (small s lower data cutoff), and statistical uncertainties (large s upper data cutoff).

2.4.2 Dealing with Correlated Data

In order to define a useful χ^2 statistic, we first focus on the correlations we observed in the $r(s)$ data for the spanning trees. In summary, we find that these correlations have a range in s that grows linearly with increasing path

length s, in dimensions $d = 2,3,4,5$. We then modify the standard χ^2 test for uncorrelated data to account for these correlations.

To describe correlations in the averaged $r(s)$ data, it will be helpful to first define how our data is averaged over samples, since we use multiple groupings of data for calculating averages. Let $r_i(s)$ denote an average of Euclidean distance $r = |\vec{r}|$ over all points on tree i that are at a chemical (path length) distance s from the origin for each tree $i = 1,\ldots,N$ in the N samples. For faster analysis, the N samples are organized into N_b batches. The batch index α ranges over $1 \leq \alpha \leq N_b$. We will use $r_\alpha(s)$ to denote an average over the $m = N/N_b$ samples in batch α:

$$r_\alpha(s) = \frac{1}{m} \sum_{i \in \alpha} r_i(s) \; . \tag{2.7}$$

The global average over all N samples will be represented by

$$\overline{r(s)} = \frac{1}{N_b} \sum_{\alpha=1}^{N_b} r_\alpha(s) = \frac{1}{N} \sum_{i=1}^{N} r_i(s) \; . \tag{2.8}$$

To initially examine correlations over s of $r_i(s)$ within a single tree, we plot the fluctuations of the batch averages $r_\alpha(s)$ about the global average $\overline{r(s)}$. That is, we plot the variations of the average $\delta r_\alpha(s)$, where

$$\delta r_\alpha(s) = r_\alpha(s) - \overline{r(s)} \; , \tag{2.9}$$

vs. path length s. Fig. 2.6 displays these correlations for a typical batch of data in a system of size 64^2.

Note that the form of the correlations should be independent of batch size, up to a multiplicative scaling factor. We can see this explicitly by examining $\overline{\delta r_\alpha(s) \delta r_\alpha(t)}$ for path length values s and t. For $i \neq j$, i and j are independent

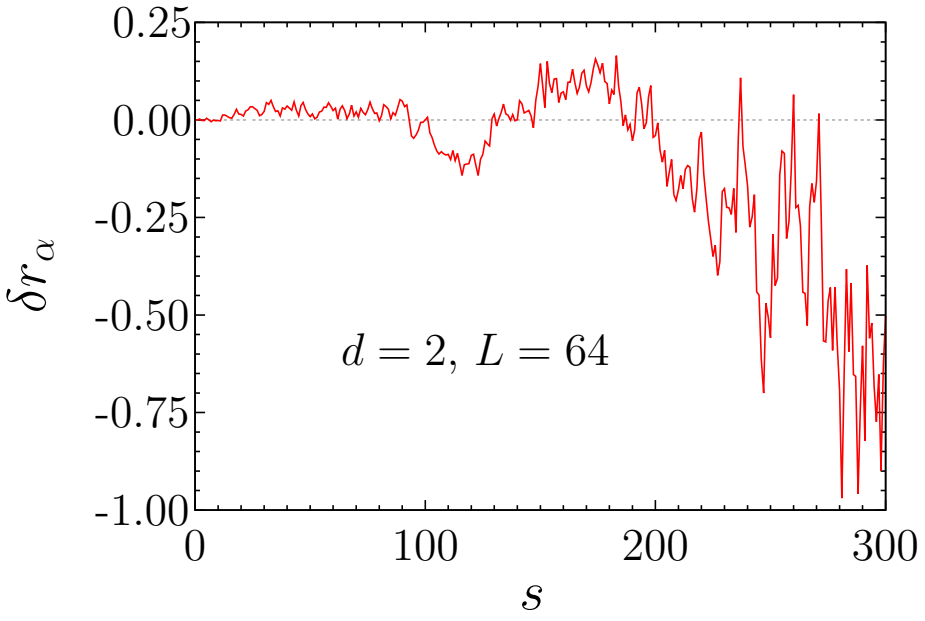

Figure 2.6: Variations from the mean for a typical batch α of data for a system of size $L=64$ in dimension $d=2$. The difference between the mean Euclidean distances $\delta r_\alpha(s) = r_\alpha(s) - \overline{r(s)}$ is plotted vs. path length s.

samples, so we can use the relation

$$\overline{\delta r_i(s)\delta r_j(t)} = \overline{\delta r_i(s)}\ \overline{\delta r_j(t)} \tag{2.10}$$

and of course

$$\overline{\delta r_i(s)} = \overline{r_i(s) - \overline{r(s)}} = 0\ . \tag{2.11}$$

By standard computation all of the $i \neq j$ cross terms are zero, and we are left with

$$\overline{\delta r_\alpha(s)\delta r_\alpha(t)} = \frac{1}{m}\overline{\delta r_i(s)\delta r_i(t)}\ , \tag{2.12}$$

showing that the choice of grouping the data into batches should not affect the form of the correlations in $r(s)$.

To quantitatively examine these correlations we compute the correlation matrix $c_{s,t}$ defined as

$$c_{s,t} = \frac{1}{N_b} \sum_{\alpha=1}^{N_b} \frac{\delta r_\alpha(s) \delta r_\alpha(t)}{\gamma(s)\gamma(t)} \,. \tag{2.13}$$

Here $\gamma(s)$ is the root mean square fluctuation in $\delta r_\alpha(s)$ computed over the N_b batches of data and is used to normalize the entries $c_{s,t}$ of the correlation matrix:

$$\gamma^2(s) = \frac{1}{N_b} \sum_{\alpha=1}^{N_b} \delta r_\alpha^2(s) \,. \tag{2.14}$$

A sample correlation matrix for $L = 64$ is plotted in Fig. 2.7. The data suggests that the correlation length increases with increasing s. To construct a model for the scaled correlation length, we measured the width along the diagonal of the peak in the correlation matrix for various values of s. We used different measures for measuring this width, including a measure of the full width at half maximum (how many "steps" away from the diagonal before $c_{s,t}$ falls to a value of $1/2$), a measure of one decay length (how many steps from the diagonal until $c_{s,t}$ drops to a value of $1/e$), and a measure using an exponential decay model $c_{s,t} = \exp(-|s-t|/\ell')$ assuming an unscaled correlation length ℓ'. We calculate ℓ' using the zeroth moment (integral) of $c_{s,t}$, allowing us to then obtain the scaled correlation length $\ell = \ell'/L^{d_s}$, given a rough estimate for d_s.

We plot in Fig. 2.8 the scaled correlation length ℓ vs. ω for multiple system sizes for dimension $d = 2$, using the zeroth moment of $c_{s,t}$ to estimate the unscaled correlation length ℓ'. We see that ℓ increases linearly with ω up until roughly $\omega = 1$, at which point the scaled correlation length levels off to a constant value. An alternate example of a measure of correlation length ℓ' for $d = 4$ is displayed in Fig. 2.9. Here we used the full width at half maximum to measure correlation length. There are larger fluctuations in the curve. Al-

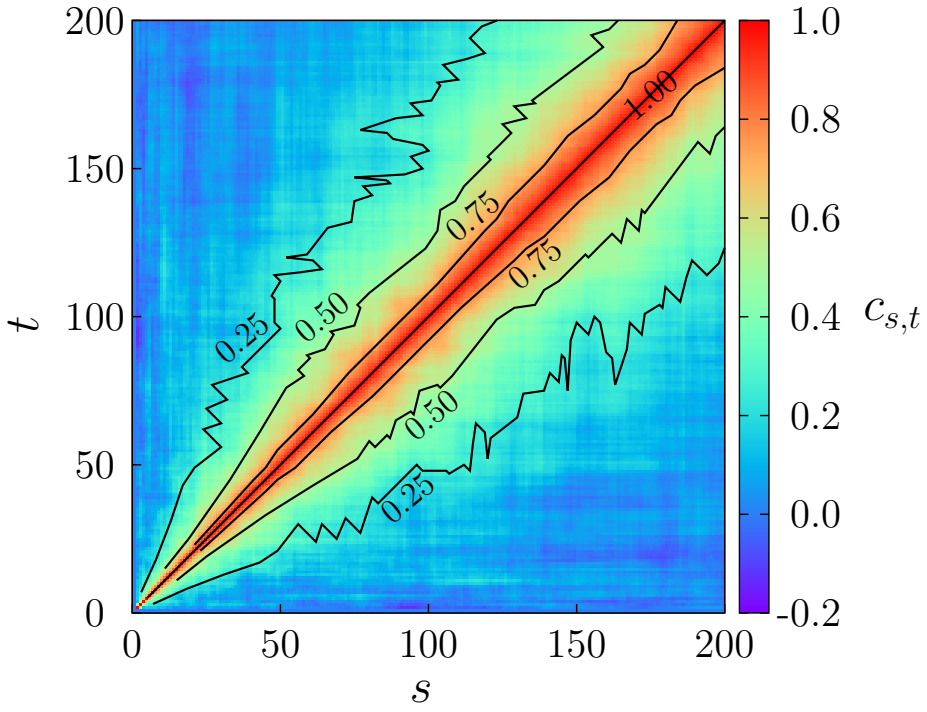

Figure 2.7: Correlation matrix $c_{s,t}$ averaged over all data for systems of size $L = 64$ in dimension $d = 2$. Selected contours are shown.

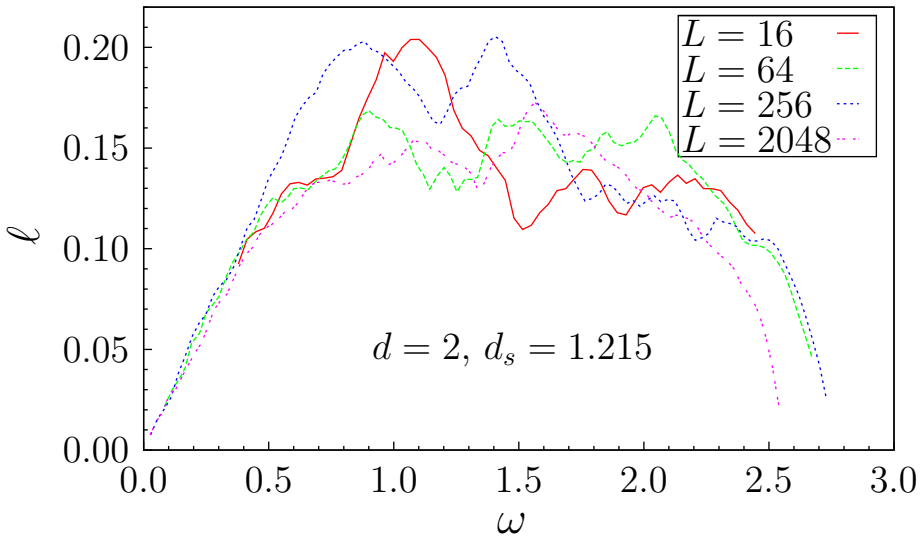

Figure 2.8: Scaled correlation length ℓ vs. ω in dimension $d = 2$ with $d_s = 1.215$ for systems $L = 16, 64, 256, 2048$ (red solid, green dashed, blue dotted, and magenta dash-dotted lines, respectively). In this case ℓ was measured assuming exponential decay $c_{s,t} = \exp(-|s-t|/\ell')$ and integrating $c_{s,t}$ to obtain the unscaled correlation length ℓ', with the relation to the scaled correlation length ℓ being $\ell = \ell'/L^{d_s}$.

though the proportionality constant differs depending on which method is used to measure the correlation length, we observed a relationship between ℓ and ω consistent with linear behavior for ω up to about 1 in all cases for dimensions $2 \leq d \leq 5$. The crossover generally appears broader when measuring correlation length with the full width half maximum than when measuring using the integral of $c_{s,t}$. The broadness of this crossover doesn't seem to depend on dimensionality. Based on this empirical observation, as well as recent results [61] showing that invasion percolation has strong mixing over geometrically

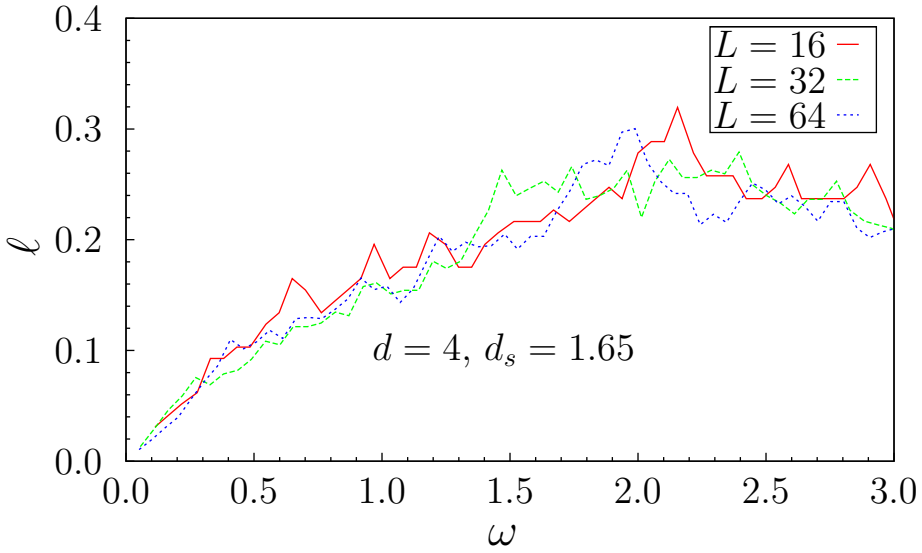

Figure 2.9: Scaled correlation length ℓ vs. ω in dimension $d = 4$ with $d_s = 1.65$ for systems $L = 16, 32, 64$ (red solid, green dashed, and blue dotted lines, respectively). In this case the unscaled correlation length ℓ' was measured using the full width at half maximum (how many "steps" in s away from the diagonal before $c_{s,t}$ falls to a value of $1/2$). To obtain the scaled correlation length ℓ, we use the relation $\ell = \ell'/L^{d_s}$.

separated scales, we choose an ansatz for the scaled correlation length.

$$\ell(\omega) \propto \begin{cases} \omega & : \omega \leq 1 \\ 1 & : \omega > 1 \, . \end{cases} \tag{2.15}$$

The $\omega = 1$ cutoff for the linear regime in this model is a rough estimate based on the empirical data as seen in Figs. 2.8 and 2.9, and we find this model to work well with our data. Furthermore, this cutoff is unimportant to some extent since high ω (large path length) data points are weighted less overall, as long paths are less frequent. For the exponential decay model used in Fig. 2.8, the proportionality is $\ell \approx 0.24\omega$. Physically, this means that on average, for any given growing path in a spanning tree, the path must grow about 24% longer than its current length before the correlations in the $r(s)$ data for this path decay, using the exponential decay model $c_{s,t} = \exp(-|s-t|/\ell')$ for the unscaled correlation length ℓ'.

Now that we have a consistent model for the correlation length, we incorporate this model into a χ^2 goodness of fit measure. We use subscripts 1 and 2 to indicate data sets for systems of size L_1 and L_2 being compared. We use $L_2 = 2L_1$. A general goodness of fit measure for quantifying how well two curves collapse in variables ρ vs. ω starts with choosing a set of independent points ω_k and at each point calculating the difference between $\rho_1(\omega_k)$ and $\rho_2(\omega_k)$, $\Delta_{12}(\omega_k) = \rho_1(\omega_k) - \rho_2(\omega_k)$.

For a χ^2 test with uncorrelated data this difference $\Delta_{12}(\omega_k^0)$ at the point ω_k^0 would then be squared and normalized by the sum of the variances, $\sigma_1^2(\omega_k^0)$ for the system of size L_1 and $\sigma_2^2(\omega_k^0)$ for the system of size L_2. This gives

$$\chi_0^2(d_s; L_1, L_2) = \sum_k \frac{\Delta_{12}^2(\omega_k^0)}{\sigma_1^2(\omega_k^0) + \sigma_2^2(\omega_k^0)} \, , \tag{2.16}$$

where we would sum over discrete points ω_k^0 chosen with uniform spacing [126].

To incorporate the observed structure of the correlations into our definition of χ^2, we compare the spacing of data points with the correlation length. First consider grouping the data points into bins with a representative point ω_k at the center of each bin, with the ω_k points spaced out in such a way that they are effectively independent. To this end, we consider centering bins around the terms of the sequence $\omega, q\omega, q^2\omega, \ldots, 1$, where q is some constant factor. Our assumption is that data points in bins centered at ω and $q\omega$ are correlated to the same extent as data points in bins centered at $q^k\omega$ and $q^{k+1}\omega$. So the correlation length is effectively constant if we choose bins logarithmically spaced, i.e., $\omega_{k+1} = q\omega_k$, for $\omega_k \leq 1$ and for some constant factor q. Furthermore, changing the value of q will change the value of this correlation length, and there will be some choice of q for which this correlation length is equal to 1, allowing us to use the form of Eq. (2.16):

$$\chi^2 = \sum_k \frac{\Delta_{12}^2(\omega_k)}{\sigma_1^2(\omega_k) + \sigma_2^2(\omega_k)} \,. \tag{2.17}$$

If we consider taking the continuum limit of this equation, we see that

$$\begin{aligned} \chi^2 &= \int \frac{\Delta_{12}^2}{\sigma_1^2 + \sigma_2^2} \, d(\log_q \omega) \\ &= \int \frac{\Delta_{12}^2}{\sigma_1^2 + \sigma_2^2} \left(\frac{1}{\ln(q)\omega} \right) d\omega \\ &\propto \int \frac{\Delta_{12}^2}{\ell_q(\sigma_1^2 + \sigma_2^2)} \, d\omega \,, \end{aligned} \tag{2.18}$$

where we have defined $\ell_q = \ln(q)\omega$. If we revert back to the discrete form, we see that our generalized goodness of fit measure becomes

$$\chi^2 = \sum_k \frac{\Delta_{12}^2(\omega_k^0)}{\ell(\omega_k^0)[\sigma_1^2(\omega_k^0) + \sigma_2^2(\omega_k^0)]} \,, \tag{2.19}$$

where $\ell(\omega_k^0) \propto \ell_q$ is the scaled correlation length at point ω_k^0 [156]. Dividing by the scaled correlation length ℓ will effectively weight each "box" or bin of data points (whose width is equal to ℓ) as one independent data point in the χ^2 sum. We use the form of ℓ given by Eq. (2.15). The proportionality constant in Eq. (2.15) affects the scale of χ^2 but not the fitted d_s. See Appendix C for details.

For our analysis, all runs ($N = 4 \times 10^6$ samples per system size) were split into $N_b = 400$ uniform batches (histograms) of $N/N_b = 10^4$ samples apiece. This was done in part because storing data for all 4×10^6 samples was impractical and running the bootstrap analysis over samples individually would be too time consuming to be feasible. To determine a useful batch size, we ran some preliminary tests by varying the batch size and plotting scaling collapses like Fig. 2.4. We saw similar estimated values of d_s based on these collapses, independent of batch size. A batch size of 400 seemed balanced because it allowed for error bars of order $1/\sqrt{N_b} \approx 5\%$. As we have shown in Eq. (2.12), the form of correlations should be independent of batch size chosen.

2.4.3 Extrapolation of Thermodynamic Limit

Next we must consider exactly what region in s of the data we want to fit. We address lattice effects by examining a lower (small s) data cutoff. The upper (large s) cutoff is also considered due to low statistics (large uncertainties) for $s \gg L^{d_s}$, though in the end we find that no upper cutoff is necessary. To determine reasonable cutoffs, we examine how the measured χ^2 and d_s respond to changes in these cutoffs. A more detailed discussion is included in Appendix C. Once we've decided upon fair cutoffs, this collapse procedure is run N_b times

(once for each batch of data for each pair of sizes L_1, L_2 with $L_2 = 2L_1$). Each time, the value of d_s for which χ^2 is minimized is found, giving N_b independent estimates $d_s(\alpha, L_1, L_2)$ for a given pair of systems L_1 and L_2. The final estimate of d_s for this pair of systems is the average of the N_b independent estimates:

$$\overline{d_s(L_1, L_2)} = \frac{1}{N_b} \sum_{\alpha=1}^{N_b} d_s(\alpha, L_1, L_2) \; . \tag{2.20}$$

Then the sample variance $S^2(L_1, L_2)$ in these N_b estimates is used to estimate the statistical error bars in the final estimate,

$$\sigma_{d_s}(L_1, L_2) = \frac{S(L_1, L_2)}{\sqrt{N_b - 1}} \; . \tag{2.21}$$

At this point we have obtained a single estimate of d_s (with error bars) for each consecutive pair of systems simulated.

Next we look to see if this $\overline{d_s(L_1, L_2)}$ converges to some value as L_1 and L_2 tend to infinity. We extrapolate the infinite system size limit using a variety of least squares fitting routines adapted from the GNU Scientific Library [135]. Standard scaling suggests that d_s as a function of system size L will exhibit power law scaling behavior. But since we have no knowledge of the expected value for the exponent in this power law, we can allow this exponent to vary as a parameter in our fit, plotting $\overline{d_s}$ vs. $L^{-\lambda}$, where we take L to be the geometric mean $L = \sqrt{L_1 L_2}$. Allowing λ to vary, fitting the data gives an estimate for d_s as well as the correction to scaling exponent λ. See Fig. 2.10 for an example of one extrapolation attempt, where λ is seen to be roughly 0.5 and linear least squares fitting for $\overline{d_s}$ vs. $L^{-\lambda}$ is used for the four largest system sizes in $d = 2$ dimensions.

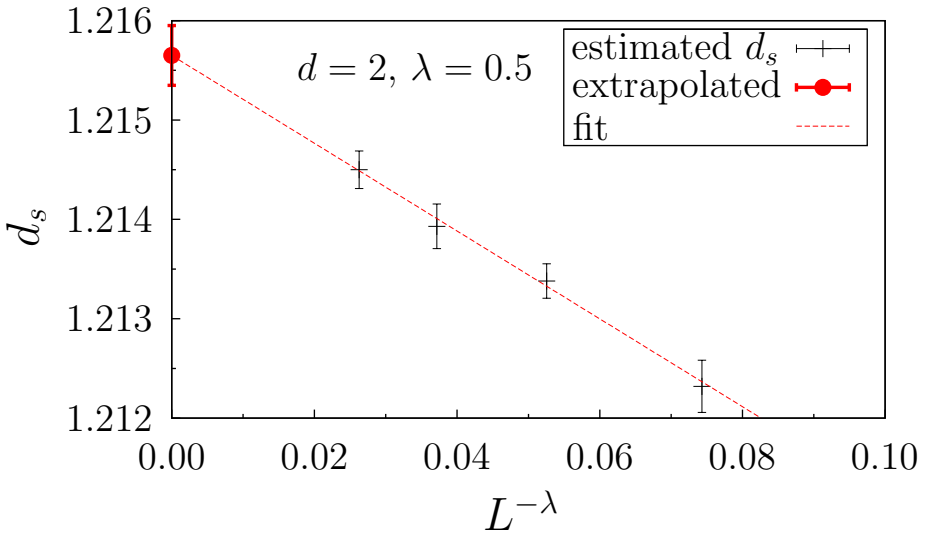

Figure 2.10: An illustration of a linear least squares fitting method being used to extract the value of d_s in the infinite system size limit in dimension $d = 2$. The fit uses the form $d_s(L) = AL^{-\lambda} + d_s(\infty)$ where λ is a correction to scaling exponent and A and $d_s(\infty)$ are fitting parameters. Here $\lambda = 0.5$, and $L = \sqrt{L_1 L_2}$, where L_1 and L_2 are the two system sizes used to produce a given data point. The fit found gives $d_s(\infty) = 1.216(1)$.

2.4.4 Blind Test of Analysis Method

To test this data analysis procedure, we applied the procedure to the similar problem of the uniform spanning tree (UST) in dimension $d = 2$. Whereas a minimal spanning tree seeks to minimize the total weight of a tree that spans all the vertices of the lattice, a uniform spanning tree is simply any tree that spans the vertices of the lattice, chosen with equal weight from all possible spanning trees. It can be thought of as a generalization of the minimal spanning tree problem to a system where all edges have the exact same weight. The reason for using this system as a test case for the analysis method is that one can examine $r(s)$ data to look at d_s for paths on the UST, just as one would examine such paths on an MST. The analysis should be completely analogous, and we observed directly that the correlations in the $r(s)$ data for the UST have a similar structure to that of the MST, allowing us to use the form of Eq. (2.15) for ℓ. To ensure no bias in this test, one of us was not made aware of the nature of the system at the time of this test but was only given the prepared $r(s)$ data on which to blindly run the analysis. The final result from this blind test, using a range of systems of size 128^2 to 1024^2, was $d_s = 1.2499(4)$ for the UST, which agrees well with the known exact result $d_s = 5/4$ in two dimensions [157, 158]. This provides confidence in the data analysis method.

2.4.5 Comparison of Results to Literature

Table 2.1 displays our final numerical estimates for d_s. The values in dimensions $d = 2$ and $d = 3$ agree with previous results [159–162] and have error bars that are similar or smaller. Our result for d_s in dimension $d = 4$ is a bit higher than the result 1.59(2) by Cieplak, Maritan, and Banavar [159]. The

Table 2.1: d_s calculated using MST algorithms

Dimension d	Two-step	Prim's (no trimming)
2	1.216(1)	1.216(1)
3	1.46(1)	1.46(1)
4	1.65(2)	1.66(2)
5	1.86(4)	1.86(4)

fractal dimensions computed from the two-step MTISC agree with those of the intermediate, untrimmed Prim's MTISC, to within our error estimates.

Our results for the path length dimension d_s can be directly compared with the $\mathcal{O}(\epsilon = 6-d)$ expansion of Jackson and Read. A graphical comparison of the data is shown in Fig. 2.11. Our results are not in conflict with the first order perturbation theory calculations. Some previous comparisons of $\mathcal{O}(\epsilon)$ calculations with numerical results for disordered materials show differences of similar magnitude [163].

We investigate possible $\mathcal{O}(\epsilon^2)$ calculations using nonlinear fitting routines adapted from the GNU Scientific Library [135] to fit d_s vs. ϵ. Using a two parameter fit,

$$d_s = 2 + a\epsilon + b\epsilon^2 \ , \tag{2.22}$$

and allowing a,b to vary, we find a chi-squared of 3.8 for two d.o.f., suggesting consistency with a quadratic fit to within our errors. For this fit, we find $a = -0.142(8)$, which is near to the $-1/7$ suggested by Jackson and Read. We find for this fit the value $b = -0.014(2)$ for the second order prefactor.

Fixing $a = -1/7$, a one parameter fit,

$$d_s = 2 - \frac{\epsilon}{7} + b\epsilon^2 \,, \tag{2.23}$$

gives $b = -0.0133(1)$ with a chi-squared of 3.8 for three d.o.f. Presuming that the Jackson and Read result is correct to first order, this gives us a more precise numerical prediction for the second order term. This fit, Eq. (2.23), is plotted in Fig. 2.11.

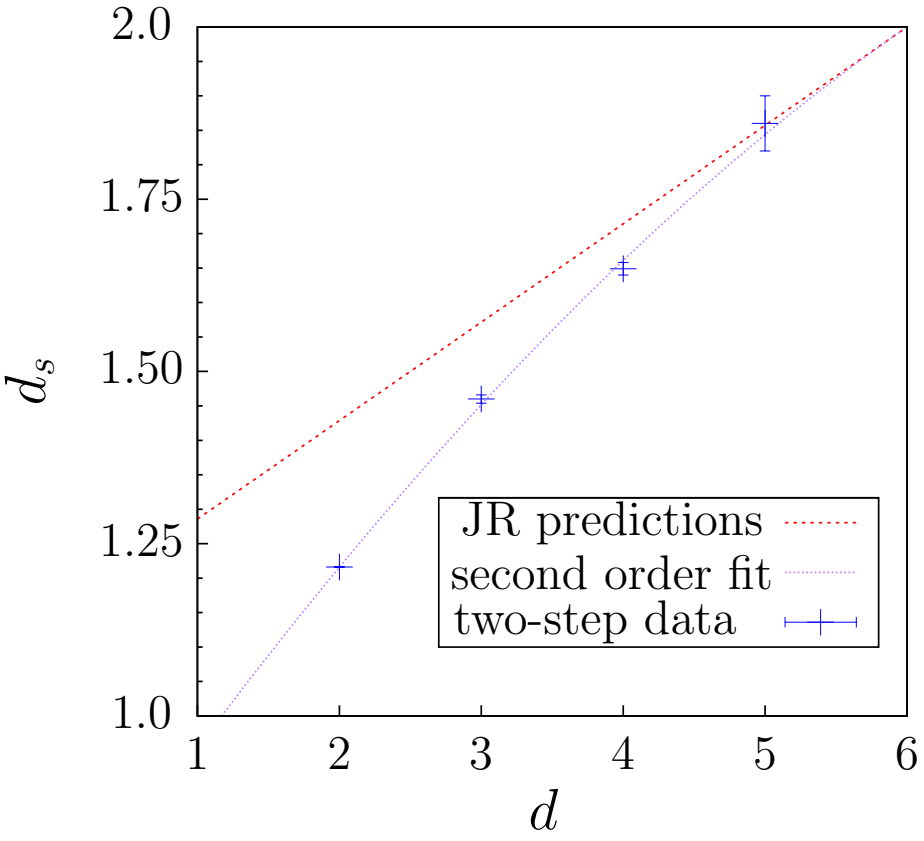

Figure 2.11: A plot comparing numerical results for d_s using the two-step method (the blue points) against predictions from the $\mathcal{O}(\epsilon)$ perturbation expansion theorized by Jackson and Read (the red dashed line). Also included is an example of a compatible $\mathcal{O}(\epsilon^2)$ fit (the purple dotted line), $d_s = 2 - \frac{\epsilon}{7} + b\epsilon^2$, with a best fit value $b = -0.0133$.

2.5 Summary

The intention of the two-step method was to allow the simulation of larger systems than were previously possible, reducing memory requirements of MST-finding algorithms by combining Prim's algorithm with Kruskal's algorithm. In this regard the work was successful, allowing for precise calculations of d_s. The trimming of the Prim's algorithm tree to possibly improve scaling of MTISCs constructed appears to have been unnecessary. Calculations using Prim's algorithm alone yielded almost identical results to those that used the two-step method.

We developed a data analysis method that allowed taking nonuniform correlations into account in order to obtain more accurate estimates for d_s. This analysis method should be applicable to a wide array of disordered systems with scale invariance and may be useful for future work.

The results for d_s calculated in this work are compatible with the perturbation expansion result proposed by Jackson and Read. Fitting $d_s = 2 + a\epsilon + b\epsilon^2$, we find $a = -0.142(8)$, compatible with the predicted $a = -1/7$ [6]. Fixing the first order result to the Jackson and Read result, we used an $\mathcal{O}(\epsilon^2)$ fit, $d_s = 2 - \frac{\epsilon}{7} + b\epsilon^2$, yielding $b = -0.0133(1)$. This could be checked if a higher order analytic calculation could be computed.

This work was made possible in part by NSF Grant No. DMR-1006731 and by the Syracuse University Gravitation and Relativity computing cluster, which is supported in part by NSF Grant No. PHY-0600953. This work was carried out largely using the Syracuse University HTC Campus Grid, a computing resource of approximately 2000 desktop computers supported by Syracuse University. Some of this work was discussed at the Aspen Center for

Physics (NSF Grant No. 1066293).

2.6 Appendices

In these appendices, we present more precise definitions of the algorithms used to generate MTISCs, as well as details of the connection between T_P, T_K, and T_2. We also discuss the region in s used for the χ^2 fitting procedure to estimate the fractal dimension d_s.

2.6.1 Appendix A: Definitions and Algorithms

Here we present a more detailed discussion of Kruskal's and Prim's algorithms. We consider these MST-finding algorithms in the context of an undirected weighted graph $G = (V, E, w)$, where V is a set of vertices, E is a set of edges connecting these vertices, and we define a weight function $w : E \mapsto \mathbb{R}$, with each edge $e \in E$ having weight $w(e)$. We consider the case of unique weights; no two edges in E have exactly the same weight.

In order to precisely describe the algorithms, it will be helpful to define some terms. A cycle or loop is a closed path such that the removal of any single edge from this path will result in the path becoming an open path, a connected set of edges with no vertex shared by more than two edges. When constructing trees, the notion of allowed and forbidden cycles is useful. Every cycle in the finite set $\mathcal{C}$, the set of all possible cycles for graph G, is chosen to belong to $\mathcal{C}_F$ (the set of forbidden cycles) or $\mathcal{C}_A$ (the set of allowed cycles), where $\mathcal{C}_F \cup \mathcal{C}_A = \mathcal{C}$ and $\mathcal{C}_F \cap \mathcal{C}_A = \emptyset$. In a typical application on a periodic lattice, the allowed cycles correspond to loops that wrap around the lattice, while forbid-

den cycles correspond to non-wrapping loops that can be deformed, plaquette by plaquette, to a point. A set of two or more edges is considered connected if every edge in the set shares a vertex with at least one other edge in the set. A cluster T is set of connected edges and the vertices that these edges connect. A cluster may contain cycles, whereas a tree is an acyclic cluster. T_E denotes the set of edges in the cluster and T_V denotes the set of vertices in the cluster.

Consider edge $e = (u, v)$ where $e \in E$ and $u, v \in V$. Adding e to a cluster T means that the edge set for the cluster, T_E, becomes $T_E \cup \{e\}$, and the set of vertices in the cluster, T_V, becomes $T_V \cup \{u, v\}$. A forbidden edge for a cluster T is an edge that, if added to T, would cause T to gain a forbidden cycle. An edge that is not a forbidden edge is an allowed edge. An allowed terminating edge for a cluster T is an allowed edge that, if added to T, would cause T to gain an allowed cycle (a wrapping loop). The addition of this allowed cycle will be used as the termination condition for both Kruskal's and Prim's algorithms, which are described below. ∂T is the set of adjacent edges for a cluster T, consisting of all edges that have one (but not both) endpoints in cluster T. As this set of edges forms a border or frontier on the outer regions of the cluster, we call ∂T the frontier of cluster T. The forbidden frontier of this cluster, $\partial_F T$, is the subset of the adjacent edges that are forbidden edges for T. The allowed frontier of this cluster, $\partial_A T$, is the subset of the adjacent edges that are allowed edges for T.

To examine an edge in Kruskal's or Prim's algorithm is to select this edge during a step of the algorithm and decide whether to accept or reject this edge into a cluster based on the conditions of the algorithm (usually dealing with the weight of the edge and whether this edge is allowed or forbidden for a particular cluster).

Next we will present Kruskal's and Prim's algorithms, as well as our two-step method, step by step, using the notation we have outlined above.

Kruskal's Algorithm:

1. Sort E by increasing weight w to form a list of edges L.

2. Initialize each vertex in the graph to be its own tree, not connected to any other vertices and containing no edges.

3. Select the first (lowest weight) edge $e=(u,v)\in L$. Remove this edge from L. If e is a allowed edge, join into a single tree the tree containing u with the tree containing v, adding edge e. If e is a forbidden edge (that is, adding edge e to to the tree(s) containing its endpoints u and v would introduce a forbidden cycle), edge e is disregarded and not added to any trees. Edge e has now been examined.

4. Repeat step (3) until a allowed terminating edge, i.e., one that introduces an allowed (wrapping) cycle, is selected. The tree that contains both the vertices of this edge is identified as the Kruskal's MTISC T_K, and the allowed terminating edge that is examined in this final step is the Kruskal's wrapping edge, $e_K\in E$.

Prim's Algorithm:

1. Initialize the growing Prim's tree T_g to have one site, called the Prim's origin, $v_0\in V$. Note that T_g is both a function of the v_0 and time (number of iterations of Prim's algorithm), since T_g "grows" from the origin v_0 by adding edges and vertices as Prim's algorithm proceeds.

2. Sort the frontier of the current Prim's tree, ∂T_g, by increasing edge weight w to form the Prim's queue (a function of T_g), excluding any edges that have already been examined.

3. Select the first (lowest weight) edge $e = (u, v)$ in the Prim's queue. If e is a allowed edge, add it to T_g, and if either u or v is not already in T_g, add it to T_g as well. Otherwise, if e is a forbidden edge, disregard the edge and do not add it to T_g. Edge e has now been examined.

4. Repeat steps (2)-(3) until a allowed terminating edge, i.e., one that introduces an allowed (wrapping) cycle, is selected. At this time, the growing Prim's tree T_g is identified as the Prim's MTISC T_P, and the allowed terminating edge examined in this final step is the Prim's wrapping edge, $e_P(v_0) \in E$. Note that T_P and e_P are both functions of the Prim's origin v_0.

two-step Algorithm:

1. Perform Prim's algorithm on initial seed site v_0 to obtain the Prim's MTISC T_P.

2. Using T_P as the input graph, run Kruskal's algorithm *on* T_P. Thus the set of edges L that is sorted and considered in Kruskal's algorithm will be the edges in T_P. This is the "trimming" step that may prune off some parts of the Prim's MTISC near the seed site v_0, yielding the two-step MTISC, T_2.

3. For analsysis of paths, a new origin is selected randomly among the vertices in T_2, and both Euclidean distance r and path length s for all points in T_2 are calculated relative to this random origin.

2.6.2 Appendix B: Proof of Validity for Two-step Method

The two-step method first applies Prim's algorithm to find $T_P \cup e_P$. Kruskal's algorithm is then executed using the edges in $T_P \cup e_P$, serving as a trimming procedure for T_P and yielding what we will term the two-step MTISC, T_2. Here we will prove that the two-step method for MTISC generation yields $T_2 = T_K$ in all cases except in the case of multiple disjoint ISCs. All results of this method can be divided into three cases. The first is the case where T_P is exactly equal to T_K and no trimming is needed, yielding $T_P = T_2 = T_K$. In the second case, T_K is a subset of T_P, and the trimming procedure of the two-step method yields $T_2 = T_K$. In the third case, there are multiple disjoint ISCs, and T_P and T_K share no common edges or vertices, meaning that $T_2 \neq T_K$. In other words, we will prove that

$$(T_P \supseteq T_K) \vee (T_P \cap T_K = \emptyset) \tag{2.6.2.1}$$

and subsequently that $T_2 = T_K$ in all cases except when the Prim's MTISC and the Kruskal's MTISC do not intersect. Here it will be useful to establish a few lemmas to aid in verifying these three cases we have outlined.

Lemma 1 Edges in the allowed frontier of T_K have higher weight than edges in T_K.

Proof Because Kruskal's algorithm examines edges that are monotonically increasing in weight as the algorithm proceeds, at the time Kruskal's algorithm terminates, the edges that are not examined must be higher weight than edges that have been examined. Since edges in the allowed frontier $\partial_A T_K$ cannot be in T_K, this means that edges in T_K have been examined at the time of Kruskal's algorithm termination, whereas edges in $\partial_A T_K$ have not. Thus edges in $\partial_A T_K$ must be higher weight than edges in T_K.

Lemma 2 For any given cycle $c \in \mathcal{C}$ for which all edges of c are examined in Kruskal's algorithm, the edge in this cycle that is examined last must be the highest weight edge in this cycle.

Proof Because Kruskal's algorithm examines edges that are monotonically increasing in weight as the algorithm proceeds, for the finite set of edges in c, the highest weight edge in this set will be examined last in Kruskal's algorithm.

Corollary 3 As a corollary, the Kruskal's wrapping edge $e_K \in E$ has higher weight than any edge in the Kruskal's MTISC T_K.

Proof Because e_K is the last edge examined during the running of Kruskal's algorithm, it must have a weight higher than every other edge that is examined during the running of the algorithm, including every edge in T_K.

Lemma 4 From lemma 2, for any edge e in the forbidden frontier of T_K, where adding e to T_K would form a forbidden cycle c_e, e must be the highest weight edge in the forbidden cycle c_e.

Proof The criteria for an edge e to be in the forbidden frontier $\partial_F T_K$ is that adding e to T_K would form a forbidden cycle c_e. This means that every edge in $c_e \backslash \{e\}$ is in T_K and is examined before edge e during Kruskal's algorithm. Thus, edge e will have higher weight than any other edge in cycle c_e.

Using these lemmas that we have established, we will show that each realization of the two-step method can be categorized into one of three distinct cases, with cases 1 and 2 yielding $T_2 = T_K$. In the third case, we see that there are no common edges or vertices between T_P and T_K, so $T_2 \neq T_K$ in this case.

Case 1 If the Prim's origin v_0 is a vertex in the Kruskal's MTISC T_K, then the Prim's MTISC T_P will be identical to the Kruskal's MTISC; $T_P = T_K$.

Proof

Since Prim's algorithm grows a tree T_g by examining and adding edges adjacent to T_g, it will be crucial for us to pay close attention to the frontier of T_g, ∂T_g. This frontier is sorted to form the Prim's queue from which edges are selected and considered for addition to T_g. In this first case, as long as we are only adding to T_g vertices that are also in T_K, the Prim's queue will consist entirely of edges that are either in T_K or ∂T_K. The fulfillment of this condition,

$$\partial T_g \subseteq T_K \cup \partial T_K \ , \tag{2.6.2.2}$$

will be shown in the remainder of the proof.

From lemma 1, when the frontier of the Prim's tree is sorted in Prim's algorithm to form the Prim's queue, any edges that are also in T_K will be placed before (having lower weight than) those edges in the allowed frontier of T_K. This means that all edges in T_K will be examined and added to T_g earlier in Prim's algorithm than edges in the allowed frontier of T_K. As the Prim's origin v_0 is in T_K, we are guaranteed to have at least one edge from T_K added to the Prim's queue at the start of the Prim's growth. Furthermore, since T_K is connected, as edges from T_K are added to T_g through the Prim's growth, more edges from T_K and its frontier ∂T_K will be added to the Prim's queue, ensuring that T_g has more edges from T_K to add during further steps of Prim's algorithm. T_g begins as a single vertex v_0 and grows to include more and more edges and vertices from T_K as Prim's algorithm proceeds.

Though we have shown that edges in T_K will be examined during Prim's algorithm before any edges in the allowed frontier of T_K, we cannot say the same about edges in its forbidden frontier, $\partial_F T_K$, which may also be in the Prim's queue and in consideration for selection during the Prim's growth. If edge $x = (u_x, v_x) \in \partial_F T_K$ is in the Prim's queue, from lemma 4 we know that both vertices u_x and v_x are in T_K. If we call c_x the forbidden cycle that would be created in T_K by adding edge x to T_K, we also know from lemma 4 that x has higher weight than every other edge in the forbidden cycle c_x.

Thus, all edges in $c_x \backslash \{x\}$ will be examined in Prim's algorithm and added to T_g before x is examined. So when x is finally examined in Prim's algorithm, x will be a forbidden edge for T_g and will not be added to T_g. This ensures that any forbidden cycles for T_K will be handled in the same manner in both Kruskal's and Prim's algorithms; no edges in the forbidden frontier of T_K will be added to T_g.

Using corollary 3, we can see that until the Kruskal's wrapping edge e_K is examined in Prim's algorithm, Eq. (2.6.2.2) will remain satisfied and all of the edges and vertices in T_K will be added to T_g before e_K is examined. Furthermore, as we have shown, none of the edges (allowed or forbidden) in the frontier of T_K will be added to the growing Prim's tree T_g before the Kruskal's wrapping edge e_K is examined. In this case, e_K will also serve as the Prim's wrapping edge e_P. Since the condition (2.6.2.2) is satisfied at the time $e_K = e_P$ is examined in Prim's algorithm (terminating the algorithm by adding an allowed cycle), $T_P = T_K$. In this case, the Kruskal's trimming step of the two-step method is unnecessary, and the two-step method will yield $T_2 = T_K$.

Case 2 If the Prim's origin v_0 is not in the Kruskal's MTISC T_K but the Prim's

tree T_g "grows" to include any vertices of T_K, then the Prim's MTISC T_P will include all of the Kruskal's MTISC T_K, i.e., $T_P \supset T_K$.

Proof

If we start Prim's algorithm from an origin v_0 that is not a vertex in the Kruskal's MTISC T_K, we can define bridge edge $b = (u_b, v_b)$ as the first edge added to the growing Prim's tree T_g that introduces a vertex of T_K into T_g. So $b = (u_b, v_b)$ is the first edge added to T_g for which either u_b or v_b are in T_K.

At the point in Prim's algorithm when b is added to T_g, b has a weight lower than any edge in the Prim's queue. Otherwise, edge b would not have been selected for addition to the Prim's tree at that time.

Examining edge b in the frame of the Kruskal's MTISC T_K, we know that edge b is in the allowed frontier of T_K since only one of u_b or v_b is in T_K. This means that b cannot be in T_K itself, nor can b be in the forbidden frontier of T_K. Because b is in the allowed frontier of T_K, it has a weight higher than any edge in T_K, by lemma 1. Thus, Prim's algorithm will continue as in case 1, with growth continuing from this first vertex of T_K (either u_b or v_b) that is introduced to T_g. Any edges from T_K that are added to the Prim's queue will be added to the front (lower weight end) since these edges will all have a weight lower than $w(b)$, whereas every edge in the Prim's queue thus far has weight higher than $w(b)$.

Prim's algorithm will add edges from the Kruskal's MTISC T_K until reaching termination with the examination of the Kruskal's wrapping edge e_K, which in this case will also be the Prim's wrapping edge e_P. At this point $T_P \supset T_K$. In this case, the Kruskal's algorithm portion of the two-step method will serve to trim

from the Prim's tree the region near the origin v_0 up to the bridge b, leaving $T_2 = T_K$.

Case 3 If the Prim's origin v_0 is not in the Kruskal's MTISC T_K and the growing Prim's tree T_g does not "grow" to include any vertices of T_K before Prim's algorithm terminates, then the Prim's MTISC and the Kruskal's MTISC will not intersect. In other words, $T_P \cap T_K = \emptyset$.

Proof

In this case, during Prim's algorithm, the Prim's wrapping edge e_P is examined before any vertices of the Kruskal's MTISC T_K are examined or added to the Prim's tree. The algorithm terminates with $T_P \cap T_K = \emptyset$. In this case, the two-step method will yield $T_2 \neq T_K$. However, this case is uncommon, and it is seen by direct observation that T_2 and T_K have similar scaling properties.

Taking all three cases for the two-step method, we can say that either the Prim's MTISC T_P contains or is equal to the Kruskal's MTISC T_K, or the Prim's and Kruskal's MTISCs do not intersect. Symbolically, $(T_P \supseteq T_K) \vee (T_P \cap T_K = \emptyset)$. In cases 1 and 2, where T_P and T_K do overlap, the two-step MTISC T_2 will be equal to the Kruskal's MTISC T_K. So in these cases the two-step method yields the same result as that obtained from running Kruskal's algorithm on the entire graph G. We see that $T_2 = T_K$ in all cases except the case of multiple disjoint ISCs, where the Prim's MTISC T_P (and by construction the two-step MTISC T_2 as well) does not intersect with the Kruskal's MTISC T_K.

2.6.3 Appendix C: Fitting Region for χ^2

Here we will discuss the data cutoffs we impose in order to restrict exactly what region in s of data we want to fit. These data cutoffs allow us to reduce overfitting errors in our χ^2 fitting routine (Eq. (2.19)) for estimating the fractal dimension d_s. We consider a small s (lower) cutoff by implementing s_l as the minimum s value allowed into the fitted data. We also consider a large s (upper) data cutoff by enforcing an ω_u as the maximum allowed value for any of the scaled data in the fit. Introducing an ω_u cutoff seemed natural because $\omega \gg 1$ corresponds to paths that are much longer than L^{d_s} and are rare. Since these long paths are less frequent, data points with a large ω value have higher statistical uncertainties than those with smaller ω values. For small s we consider lattice effects. It made sense to use s_l as a low cutoff since we have small discrete s values (steps) in the paths. Using ω_l would also have been a viable option, but it is easier to interpret the physical meaning of s_l, since one unit of s corresponds to the lattice spacing for all system sizes.

To determine reasonable values to use for the upper and lower data cutoffs, we tested our χ^2 fitting routine with various values of these cutoffs and assessed how well the minimum value of χ^2 agreed with a predicted estimate χ^2_p. To estimate χ^2_p, it is instructive to envision comparing data sets from two different system sizes on scaled axes ρ vs. ω, as in Fig. 2.5(a). We use a discrete set $\{\omega^0_k\}$, consisting of n uniformly spaced values of ω, to calculate a set of $\rho(\omega^0_k)$ values for each of the two data sets. The comparison between these ρ values goes into the calculation of χ_2, as outlined in Eq. (2.19).

One can imagine breaking the ω axis up into b segments or "boxes" of length equal to the scaled correlation length ℓ. Each box will contain some

number n_b of data points from the set of n points in the $\{\omega_k^0\}$ discrete points we use to calculate the goodness of the collapse. In this way, we can estimate χ_p^2 piece by piece, calculating separately the contribution χ_b^2 from each of the b boxes:

$$\chi_p^2 = \chi_b^2 b \ . \tag{2.6.3.1}$$

Further, we can estimate n_b, the number of data points that fall in each of these boxes. This allows us to subdivide the χ_b^2 into contributions from each individual data point, χ_1^2:

$$\chi_p^2 = \chi_1^2 n_b b \ . \tag{2.6.3.2}$$

Now that we have partitioned χ_p^2 by this relation, we can calculate χ_1^2, n_b, and b individually. First, we can estimate χ_1^2, the χ^2 contribution from a single data point in the set of $\{\omega_k^0\}$ values, by its expectation value $E(\chi_1^2)$. We can write, using the form of Eq. (2.19),

$$\begin{aligned} \chi_1^2 &\approx E(\chi_1^2) \\ &\approx \frac{E([\rho_1 - \rho_2]^2)}{\ell[\sigma_1^2 + \sigma_2^2]} \\ &\approx \frac{E(\rho_1^2) + E(\rho_2^2)}{\ell[\sigma_1^2 + \sigma_2^2]} \\ &\approx \frac{\sigma_1^2 + \sigma_2^2}{\ell[\sigma_1^2 + \sigma_2^2]} \\ &\approx \frac{1}{\ell} \ , \end{aligned} \tag{2.6.3.3}$$

where $\rho_1(\rho_2)$ is the ρ value taken from data set 1(2), and $\sigma_1^2(\sigma_2^2)$ is the variance at this point for data set 1(2).

Next we can write an expression for the number of data points per box, n_b, by multiplying the length of each box, ℓ, by the density of data points to obtain

$$n_b = \ell \left(\frac{n}{\omega_u - \omega_l} \right) \ , \tag{2.6.3.4}$$

where again n is the total number of data points in the set $\{\omega_k^0\}$ being used for the χ^2 calculation and $\omega_u - \omega_l$ gives the full allowed range of ω_k^0 values.

Finally, we compute the total number of boxes, b, by dividing the range $\omega_u - \omega_l$ into two regions, $\omega < 1$ and $\omega > 1$. This allows us to separately compute the number of boxes in each of these regions, $b_<$ and $b_>$ respectively. Of course, $b = b_< + b_>$. Expressing $b_>$ is trivial, since in this region the scaled correlation length is constant at $\ell = A$ (as given by our model in Eq. (2.15)), where A is the proportionality constant relating ℓ and ω by $\ell = A\omega$. Thus we write

$$b_> = \frac{\omega_u - 1}{A} \ . \tag{2.6.3.5}$$

Writing $b_<$ is not quite as simple. It may help to begin at $\omega = 1$ and think of stepping left toward lower values of ω and eventually to ω_l, counting up the number of boxes we step across. The boundary of the first box will occur at $\omega = (1 - A)$, since the scaled correlation length ℓ is equal to A around $\omega = 1$. Likewise, the edge of the second box we approach will fall at $\omega = (1-A)^2$, the i^{th} box will reach to $\omega = (1-A)^i$, and the final $b_<^{\text{th}}$ box will fall at the lowest allowed ω value, ω_l. So we can write

$$\omega_l = (1 - A)^{b_<} \ , \tag{2.6.3.6}$$

from which is follows that

$$\begin{aligned} b_< &= \frac{\ln(\omega_l)}{\ln(1-A)} \\ &\approx \frac{\ln(1/\omega_l)}{A} \ . \end{aligned} \tag{2.6.3.7}$$

Here an approximation is made in the denominator since A is small.

Combining Eqs. (2.6.3.2), (2.6.3.3), (2.6.3.4), (2.6.3.5), and (2.6.3.7), we can write an expression for χ_p^2 by adding the contributions from both of the two

regions we examined, $\omega < 1$ and $\omega > 1$:

$$\begin{aligned}\chi_p^2 &= \frac{1}{\ell}\left[\ell\left(\frac{n}{\omega_u-\omega_l}\right)\right]\left[\frac{\ln(1/\omega_l)}{A}\right] \\ &\quad + \frac{1}{\ell}\left[\ell\left(\frac{n}{\omega_u-\omega_l}\right)\right]\left[\frac{\omega_u-1}{A}\right] \\ &= \frac{1}{A}\left[\frac{n}{\omega_u-\omega_l}\right]\left[\ln\left(\frac{1}{\omega_l}\right)+\omega_u-1\right] .\end{aligned} \tag{2.6.3.8}$$

In general, this proportionality constant A will depend on what method is used to measure correlation length. For this reason, we look at the ratio χ_p^2/χ^2. Since both χ_p^2 and χ^2 have a $1/A$ dependence, the ratio of the two is independent of A.

We use this ratio χ_p^2/χ^2 to get an idea of the effect various upper and lower data cutoffs have on our fit. We plot an example case of this analysis for systems of size $L=512$ and $L=1024$ in dimension $d=2$. In Figs. 2.12(a) and 2.12(b) it is apparent that below a certain s_l threshold, lattice effects skew the value of χ^2 as well as the value of d_s. Meanwhile, Figs. 2.12(c) and 2.12(d) indicate that we need not impose an upper cutoff on the data. The high variance of points in the large ω region of the data already appropriately weights these points. For our final analysis, we chose $s_l = 425$ for two dimensions, $s_l = 375$ for three dimensions, and $s_l = 100$ for four and five dimensions.

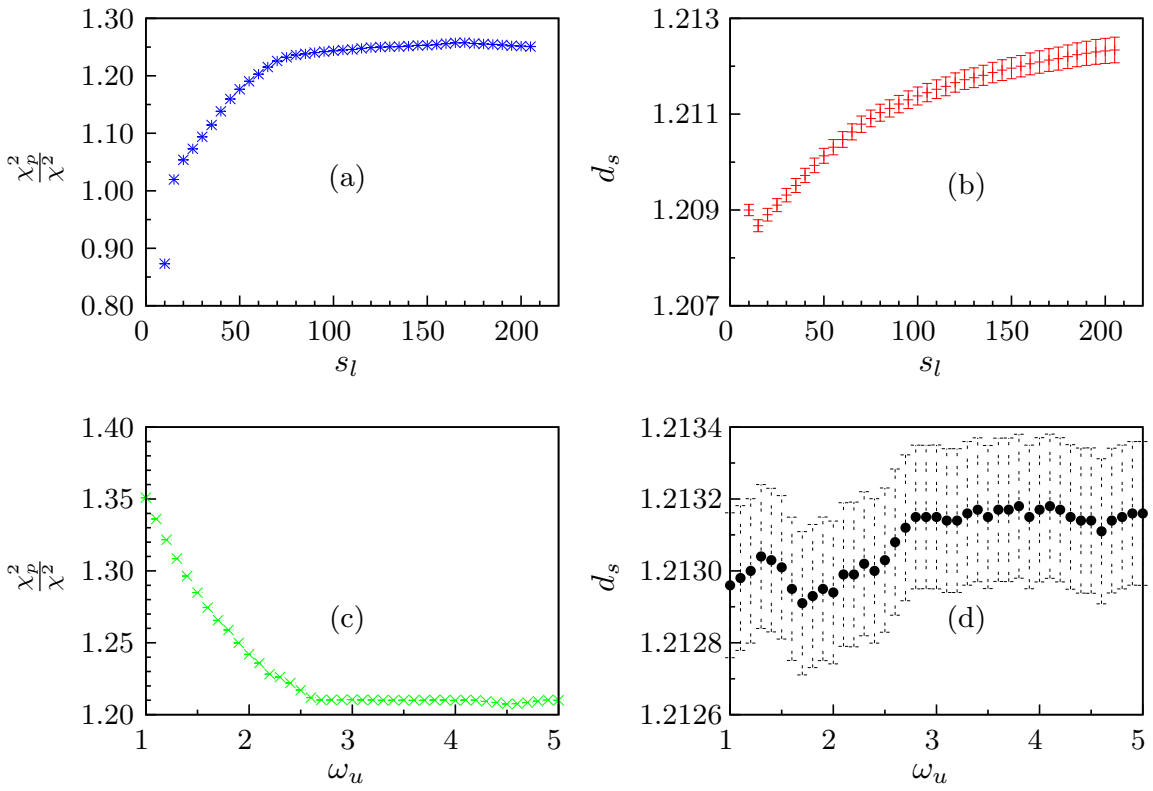

Figure 2.12: Examining the effects of the lower and upper data cutoffs on a collapse of two systems of sizes $L = 512$ and $L = 1024$ in dimension $d = 2$. (a) and (b) display χ_p^2/χ^2 and d_s as functions of the lower (small s) data cutoff s_l, while (c) and (d) show χ_p^2/χ^2 and d_s as functions of the upper (large s) data cutoff ω_u for $s_l = 100$. Both χ_p^2 and χ^2 are measured at the value of d_s for which χ^2 is minimized.

Chapter 3

Effects of Boundary Conditions in the Two Dimensional Random Ferromagnet

3.1 Overview of Project

An outstanding problem in understanding disorder at $T = 0$ is the thermodynamic limit. Is there only one ground state or many? This question is aided by the study of the effect of boundary conditions on a system's microscopic configuration.

In this project we contribute to this understanding by considering the random bond Ising ferromagnet in two dimensions, investigating the locations of domain walls given various choices of boundary spins. We simulate square lattices of $L \times L$ spins $\{s_i\}$ having randomly generated ferromagnetic couplings $\{J_{ij}\}$. We provide a proof that domain walls in the random ferromagnet are

fully decomposable in two dimensions, and we utilize a direct mapping from the random ferromagnet to a multiple-source shortest paths problem to probe all possible boundary conditions. Employing an efficient multiple-source shortest paths algorithm developed by Philip Klein [8, 164], we achieve a running time $O(N \log N)$ per sample of $L \times L = N$ vertices to solve the all boundary pairs shortest paths problem. Traditional methods using Dijkstra's algorithm require a running time $O(N^{3/2} \log N)$.

We examine the amount of control we have over the deep interior of the system, calculating a scaling exponent γ for the probability of being able to precisely control ground state domain wall locations at the center of the system by fixing boundary spins. This probability of controllability scales with system size like $P \sim L^{-\gamma}$, and we find $\gamma = .33(1)$ for systems with random, uncorrelated spin couplings. We also consider systems with spin couplings that are spatially correlated according to a power law, and find the γ for correlated disorder to be different than that of uncorrelated disorder. For a summary of the numerical results for γ with correlated disorder, see Table 3.2 and Figure 3.19. We find results that are not inconsistent with those of Schorr and Rieger [12], but an issue with the part of our correlation procedure where we ensure non-negative couplings makes any conclusive statement about these results difficult.

3.2 Motivation

The random bond Ising ferromagnet (RBIM) serves as an excellent prototype for better understanding the properties of more complex disordered systems, having scaling properties similar to those of spin glasses, the random field Ising model, percolation systems, directed polymers, and jamming systems.

The RBIM itself also has numerous physical applications, such as modeling a randomly diluted resistor lattice [165]. Conductances of the resistors are given by the spin couplings in the RBIM, and locations of dilutions (resistors missing from the lattice) are modeled by a conductance of zero. The RBIM has additionally been used to study criticality in avalanches [166], investigate fracturing and force-chains in granular materials [167, 168], and model networks of noninteracting fermions [169, 170] and disordered superconductors [171].

The RBIM can be described by the following Hamiltonian:

$$H_{\mathrm{RBIM}} = -\sum_{\langle ij \rangle} J_{ij} s_i s_j \ . \tag{3.2.0.1}$$

Here the s_i are Ising spins taking values of $\{+1, -1\}$, and J_{ij} are interaction strengths for neighboring spins. Since we are dealing with purely ferromagnetic interactions, $J_{ij} > 0$. Also called the "random ferromagnet," the RBIM physically represents a ferromagnetic material with random bond strengths due to impurities or spatial disorder in the location of the Ising spins.

The statistics of the locations of domain walls, the interfaces separating regions of up spins (+) from regions of down spins (−), is important to the phenomenological description of disordered spin systems. While the ferromagnetic nature of the RBIM couplings would normally cause a trivial ground state with spins either all up or all down, by fixing spins at the system boundaries to a particular orientation, we can force one or more domain walls to be present in the system. Values of disorder near the system boundary have the potential to significantly influence the properties and scaling of domain walls in the interior of the system. Of particular interest is the effect that changing boundary conditions—flipping some of the spins at the outer boundary of a

non-periodic system—has on the location of domain walls in the deep interior of the system, particularly in the thermodynamic limit of infinite system size. By exploring configurations of boundary spins, what is the probability that we induce a domain wall that passes through a particular location in the interior of the system? And how does this probability scale with system size?

In order to precisely discuss domain walls, it helps to classify them as either cyclic or terminated at the system boundary. A cyclic domain wall is a domain wall surrounding a cluster of spins of a given orientation embedded in a region of spins having the opposite orientation. Because this represents an excitation from the ground state, there can be no cyclic domain walls in a ground state spin configuration. All ground state domain walls start and end at the locations along the boundary where two adjacent boundary spins have opposite orientations. By imposing various combinations of boundary spins, we can explore different ground state configurations of the system, and there will be a unique ground state for each configuration of boundary spins. This fact allows us to map domain walls in the random ferromagnet to the shortest paths problem [66].

In the graphs representing materials and having N vertices, the total number of paths is $\sim e^N$. A shortest path is simply defined as the minimal-cost path between two vertices [83]. In the case of the RBIM, paths between the locations of domain wall endpoints ($+/-$ or $-/+$ interfaces along the boundary) are considered, and the interaction strengths J_{ij} provide the costs for edges along these paths. To map to shortest paths, consider the lattice dual to the RBIM, where nodes μ and ν are connected by an edge with weight $\widetilde{J}_{\mu\nu} = J_{ij}$. The edge $\langle\mu\nu\rangle$ in the dual crosses the edge $\langle ij\rangle$ in the RBIM. Domain walls then become paths with minimal total weight. This allows us to find the precise location of

a domain wall between two specified boundary locations in polynomial time by using any number of efficient shortest path algorithms.

The shortest paths problem has numerous significant applications. In traffic systems routing can be optimized efficiently by modeling roads by edges with associated weights given by the length of time it takes to traverse each road [67, 68]. In social networks, analytics to study the connections between individuals and the subsequent clustering of communities are performed using shortest path finding algorithms [3, 4]. Shortest paths can be used to find the lowest energy path of a vortex line in a disordered superconductor [65, 66], study connectivity in the world-wide web [69–71], optimize internet traffic delivery [73], model current flow in resistor networks [4, 72], more efficiently design VLSI chips [172, 173], and guide robot navigation in unknown terrain [174].

As suggested by Huse and Henley, directed polymers are useful for studying disordered spin systems [66, 77, 79]. Directed polymers are shortest paths with the stipulation that overhangs are not allowed, though overhangs are seen to be irrelevant to the asymptotic scaling of these paths. Like pinned directed polymers in random media, the locations of domain walls may wander depending on the potential (edge weights) of the underlying lattice. The transverse deviation y of paths of longitudinal length L is governed by the wandering exponent ζ, scaling as $y \sim L^{\zeta}$ when averaged thermally as well as over disorder realizations. We investigate whether the same wandering exponent $\zeta = 2/3$ from the two-dimensional directed polymer problem can be applied directly to the domain walls of the random ferromagnet in two dimensions when we consider all domain walls induced by all possible boundary spin configurations. We examine the probability P of being able to induce a domain wall that

passes through a particular location at distance $L/2$ from the system boundary. We conjecture that this probability scales as $P \sim L^{-\gamma}$, and we find $\gamma = .33(1)$ via examining scaling collapses and extrapolating to the infinite system size limit. This result is compatible with the scaling relation $\gamma = 2\zeta - 1$ that we derive in Section 3.4.2. We also examine lattices with edge weights that are spatially correlated according to a power law. Fourier weights are drawn randomly from a Gaussian distribution and then spatial correlations are added via a modified Fourier filtering method developed by Makse et. al [175, 176] to generate correlated disorder in real space. The results for the scaling with correlated disorder and with uncorrelated disorder are summarized in Table 3.2.

3.3 Model and Algorithms

In this section we briefly outline the random ferromagnet model we simulate, define useful terms, and explain the mapping to the shortest paths problem. A more detailed discussion of the model and definitions of terms is included in Appendix A. We also review Dijkstra's single-source shortest paths algorithm, which we use to initialize Klein's algorithm and to benchmark runtime performance. Next we discuss the multiple-source shortest paths algorithm we employ [8, 164], detail the data structures used to implement it [136], and examine the running time performance of both algorithms. See Appendix C for pseudocode for the shortest paths algorithms presented here.

For the sake of clarity, we'll briefly remark here about the dual usage of the terms "primal" and "dual," as well as the indices we use throughout. Because of the one-to-one mapping from primal edges to dual edges, the dual to the dual graph is the primal graph. Graph duality is a symmetric relation, and saying

that “graph B is dual to graph A” or “graph A is dual to graph B” are equivalent statements. Thus, the terms “primal” and “dual” simply become labels for the principal graph of interest and its dual graph, respectively. As our focus shifts, so does our terminology, with a change of notation to reflect that. We begin by discussing the mapping from the RBIM to shortest paths, while we later focus on shortest paths and the implementation of Klein’s algorithm.

When discussing the mapping from the RBIM spin system to the shortest paths problem, we consider the spin system with spins indexed by $\{i, j\}$ to be the primal (principal) graph. The lattice of nodes indexed by $\{\mu, \nu\}$ that forms the shortest paths problem is referred to as the dual graph, as it is dual to the RBIM spin system. The edges $\langle \mu\nu \rangle$ in the dual cross the edges $\langle ij \rangle$ in the primal graph, and there is a one-to-one mapping from primal edges to dual edges, from spins to polygons, and from spin plaquettes (faces) to nodes in the dual graph.

When discussing Klein’s shortest paths algorithm and the details of its implementation, the shortest paths lattice is the principal graph with which we are concerned. We refer to it as the primal, denoting its vertices by indices $\{u, v\}$ (rather than $\{\mu\nu\}$) to reflect this distinction, even though edges $\langle u\, v \rangle$ are equivalent to edges $\langle \mu\nu \rangle$. To denote elements dual to this lattice, we simply append an asterisk for convenience. Though subtle, this change of indices is also a reflection of the fact that Klein’s algorithm can be applied to any planar graph and is wholly unrelated to the mapping from the RBIM spin system. The mapping from the RBIM merely serves to motivate and inform the lattice upon which we choose to run Klein’s algorithm, the equivalence of course being in the mapping of Ising coupling strengths to edge weights: $J_{ij} = \widetilde{J}_{\mu\nu} = \ell(u\, v)$, where $\ell(u\, v)$ is the weight of edge $\langle u\, v \rangle$ in the shortest paths problem.

3.3.1 RBIM Model and Simulation Overview

We consider an $L \times L$ square lattice of $N = L^2$ sites, simulating one million samples apiece for systems up through 2048^2 sites, with independent identically distributed (i.i.d.) edge weights $\widetilde{J}_{\mu\nu} = \ell(u\,v)$ drawn uniformly from the interval $(0,1)$. While we are primarily concerned with shortest paths on a lattice with these i.i.d. random weights, we also tested lattices with edge weights having spatial correlations that decay according to a power law, using a modified Fourier filtering method to achieve these long-range correlations [175, 176]. We note that our discussion and results should not be unique to a square lattice. We merely choose the square lattice for simulations due to convenience.

To study domain walls, one can consider a random ferromagnet that has one contiguous section of boundary spins all facing up (+) and the remaining boundary spins all facing down (−). With a boundary spin configuration of this type, there are only two locations on the boundary where two adjacent spins have opposite orientations. Thus we induce a domain wall whose endpoints are these two boundary locations where an up-down interface exists. This process is illustrated in Fig. 3.1.

The question of interest is whether we can find an induced domain wall that passes through a specific node in the center of the system by adjusting only the endpoints of the domain wall. In the case of the pure ferromagnet at $T = 0$, the answer is yes. Consider a pure ferromagnet with no disorder, meaning that all spin pair coupling strengths are equal, $J_{ij} = J$. If we impose a single domain wall via the boundary conditions described above, the lowest energy domain wall will be a straight line separating a region of all up spins (+) and a region of all down spins (−). The location of this wall can be moved

in unit lattice steps. However, when we add an element of random disorder into the system in the form of random ferromagnetic coupling strengths J_{ij}, we find we have less control over the exact location of this domain wall, particularly at the center of the system. The domain wall begins to wander with some roughness as it extends further and further from the system boundaries. Some bonds have a high coupling strength and thus cost a great deal of energy to include in a domain wall. This means that some spins are more heavily coupled and tend to flip together due to the strong ferromagnetic interaction between them, making it unlikely that a domain wall will appear between those strongly coupled spins.

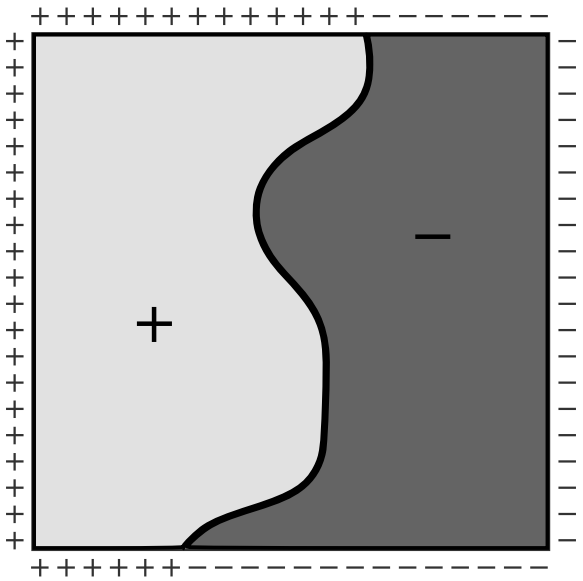

Figure 3.1: Schematic of a random ferromagnet in two dimensions with fixed boundary conditions. Having regions of differing spins on the boundary induces a domain wall, shown here as a solid black line running through the system. The domain wall separates a region of up (+) spins (shows in light grey) from a region of down (−) spins (shown in dark grey).

Rather than exhaustively iterating over all $2^{4(L-1)}$ possible combinations of

boundary spins, we take advantage of the fact that in two dimensions, complicated domain walls in the random ferromagnet can be decomposed into combinations of simple domain walls (boundary-to-boundary across the system, given by the shortest path between pairs of boundary points in the dual mapping). We include a proof of this decomposability in Appendix B, and Fig. 3.2 illustrates this concept. Because domain walls are decomposable, we effectively probe all $2^{4(L-1)}$ possible configurations of boundary spins by merely investigating the $(4L)(4L-1) \approx 16L^2$ possible decompositions of single boundary-to-boundary domain walls (shortest paths).

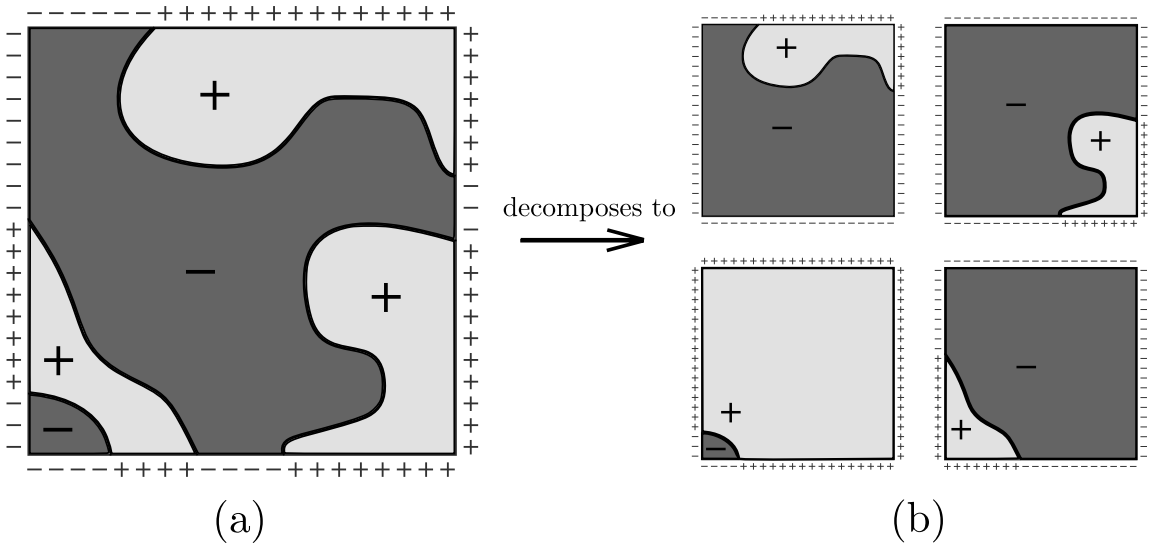

Figure 3.2: Schematic of the decomposition of domain walls in a two-dimensional random ferromagnet. The set of fixed boundary spins in (a) give rise to complicated domain walls shown as solid black lines through the system, separating regions of like spins. The regions of up (+) spins are shown in light grey, while the regions of down (−) spins are shown in dark grey. In (b) a potential decomposition of the domain walls from (a) is shown. This figure is meant merely to illustrate the concept of domain wall decomposition. A rigorous proof of the decomposability of these domain walls is provided in Appendix C.

To compute all possible boundary-to-boundary shortest paths, we employ a multiple-source shortest paths (MSSP) algorithm developed by Philip Klein [8, 164] which affords us a running time of order $N \log N$ per lattice of N vertices. This is a marked improvement over the traditional method of using Dijkstra's single-source shortest paths (SSSP) algorithm [145] repeatedly, which would require a running time of $N^{3/2} \log N$. In order to achieve the improved running time with Klein's algorithm, it is vital that we use Sleator-Tarjan dynamic trees (self-adjusting binary search trees) [137], which we implement using David Eisenstat's code library [136].

3.3.2 Mapping to shortest paths

We consider a random ferromagnet with a square grid of $L \times L$ Ising spins, where each spin i has a spin value s_i on $\{+1, -1\}$ (up or down). Each nearest neighbor pair of spins is connected by a bond $\langle ij \rangle$. The strength of ferromagnetic interaction between nearest neighbor pair of spins $\langle ij \rangle$ is given by $J_{ij} > 0$ so that the Hamiltonian is given by Eq. 3.2.0.1. The primal spin system graph $G(S, B)$ consists of vertices (spins) $s_i \in S$ and undirected edges (bonds) $\langle ij \rangle \in B$ connecting adjacent spins. We are interested in fixing the outermost boundary spins, and allowing the rest of the interior spins to vary to minimize the total energy. We note that the mapping to shortest paths is general and is not restricted to a square lattice. We simply use a square lattice here to reflect the geometry of our simulations.

In order to discuss the mapping to the shortest paths problem, we need to consider the graph that is dual to the array of Ising spins we have defined. On the dual graph $G^*(V, E)$, each plaquette or face of the primal graph becomes a

vertex μ. These dual vertices form a lattice, and nearest neighbors μ and ν in this dual lattice are then connected by undirected dual edges $\langle\mu\nu\rangle \in E$. To the exterior of this lattice of dual vertices we also add $4(L-1)$ dual vertices, which we will identify as the boundary vertices $V_b \subseteq V$ of the dual graph. Each of these boundary vertices has exactly one incident dual edge that connects it to a vertex on the outermost layer of vertices in $V \setminus V_b$, such that each vertex in V_b is connected to a different vertex in $V \setminus V_b$. Each bond $\langle ij\rangle$ between spins i and j maps to an edge $\langle\mu\nu\rangle \in E$ that crosses the bond and connects dual vertex μ to dual vertex ν. We define a length (energy cost), $\widetilde{J}_{\mu\nu}$ for each bond $\langle\mu\nu\rangle$ in the dual. There is a one-to-one mapping for bonds in the primal to edges in the dual, and $J_{ij} = \widetilde{J}_{\mu\nu}$ for each mapped bond. This dual mapping is depicted in Fig. 3.3.

In the primal graph, a cluster boundary is a set of adjacent unsatisfied bonds that forms a boundary for a connected cluster of identical spins. In the dual graph, this list of bonds becomes a list of edges forming a path, which is called a domain wall. Thus a domain wall represents a path in the dual graph only containing edges that map to unsatisfied bonds. For a cluster boundary Λ, equivalent to a path P on the dual graph, the total energy of Λ (or equivalently the total length ℓ of path P) is given by

$$H_\Lambda = \sum_{\langle ij\rangle\in\Lambda} J_{ij} = \sum_{\langle\mu\nu\rangle\in P} \widetilde{J}_{\mu\nu} = \ell(P)\,. \tag{3.3.2.1}$$

Minimizing the total energy of a cluster boundary is equivalent to minimizing the total length of a path in the dual graph (finding a shortest path).

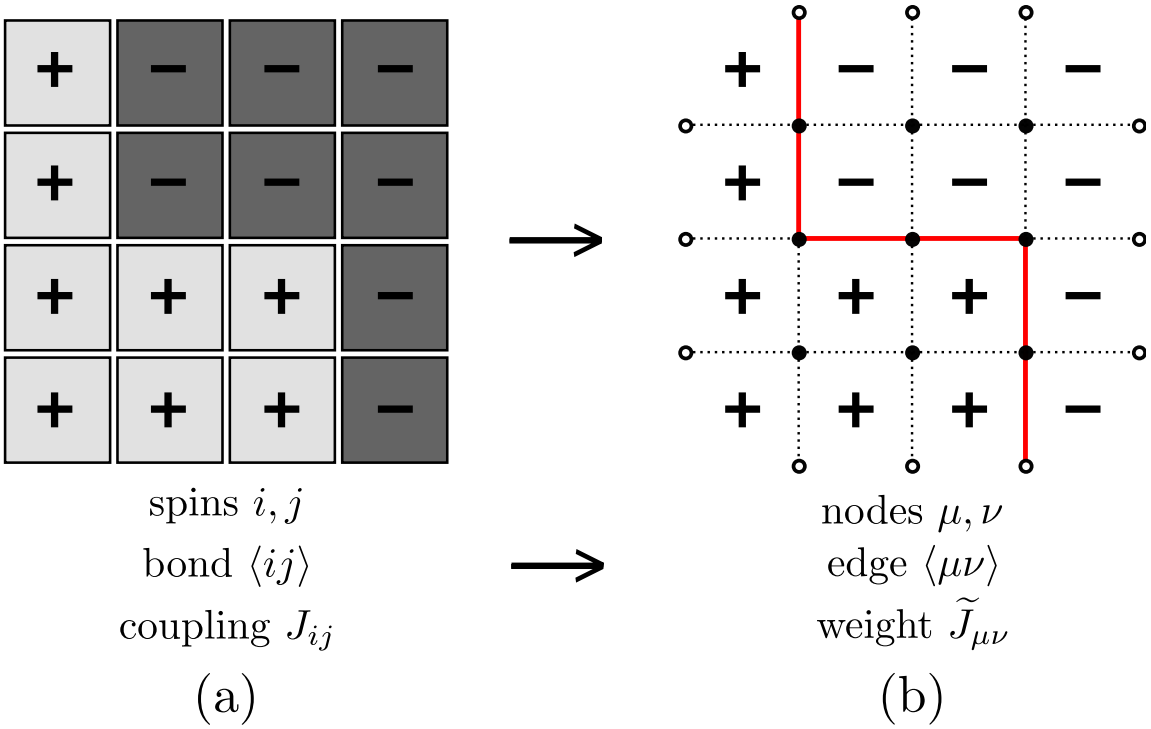

Figure 3.3: An illustration of the mapping from domain walls in the RBIM to shortest paths. In (a), neighboring Ising spins i, j share a bond of ferromagnetic coupling strength J_{ij}. In (b), the dual graph is shown, with nodes μ, ν sharing an edge with weight $\widetilde{J}_{\mu\nu}$. Because bonds in the RBIM are mapped directly to edges in the dual graph in a one-to-one fashion, $\widetilde{J}_{\mu\nu} = J_{ij}$ for each mapped bond. In (b), the boundary nodes are depicted as open circles, and the shortest path is shown by the solid red line, making the equivalence between shortest paths and domain walls visually apparent.

3.3.3 Dijkstra's SSSP algorithm

Because of the mapping from domain walls in the random ferromagnet to shortest paths on the dual graph, in order to probe all possible configurations of boundary spins in the random ferromagnet, we must compute the shortest path between all pairs of boundary vertices in the dual graph. For a square lattice of size L by L, there will be $4(L-1)$ such boundary vertices. The most

naive method would be to check all possible paths between pairs of boundary vertices and select the shortest ones. While this brute force method would certainly provide the correct results, the running time would be exponential in the number of vertices, $N = L^2$.

A more reasonable method to compute these paths would be to repeatedly use Dijkstra's single-source shortest paths (SSSP) algorithm [145] to compute the shortest paths from each of the boundary vertices. Dijkstra's algorithm efficiently computes shortest paths from a single source, constructing a shortest paths tree rooted at the given source node. With a mutable Fibonacci heap priority queue implementation such as the one provided in the BOOST C++ libraries [134], this can be done in $O(N \log N)$ time.

Dijkstra's algorithm begins with the shortest paths tree T only containing the source node. At the start of the algorithm, the source node is given a distance of 0, while all other nodes are given a starting distance of ∞. Growth continues outward, as the weights of edges adjacent to T are considered in order to update the minimum distance to all nearby reachable nodes that are not yet in T. At any given step of the algorithm, the closest reachable node not in T is added to T. This criterion automatically excludes loops from being introduced to T, and the greedy nature of this selection ensures that globally optimal (shortest) paths are included in T when the algorithm is complete. As the algorithm progresses and T grows, previously unexamined edges are considered, allowing the reachable distance to nearby nodes to be lowered (often multiple times) before they are eventually added to T. An example is depicted in Fig. 3.4.

Using Dijkstra's algorithm repeatedly to compute all of the boundary-to-boundary shortest paths in an $L \times L$ square lattice would require $4(L-1)$ calls to

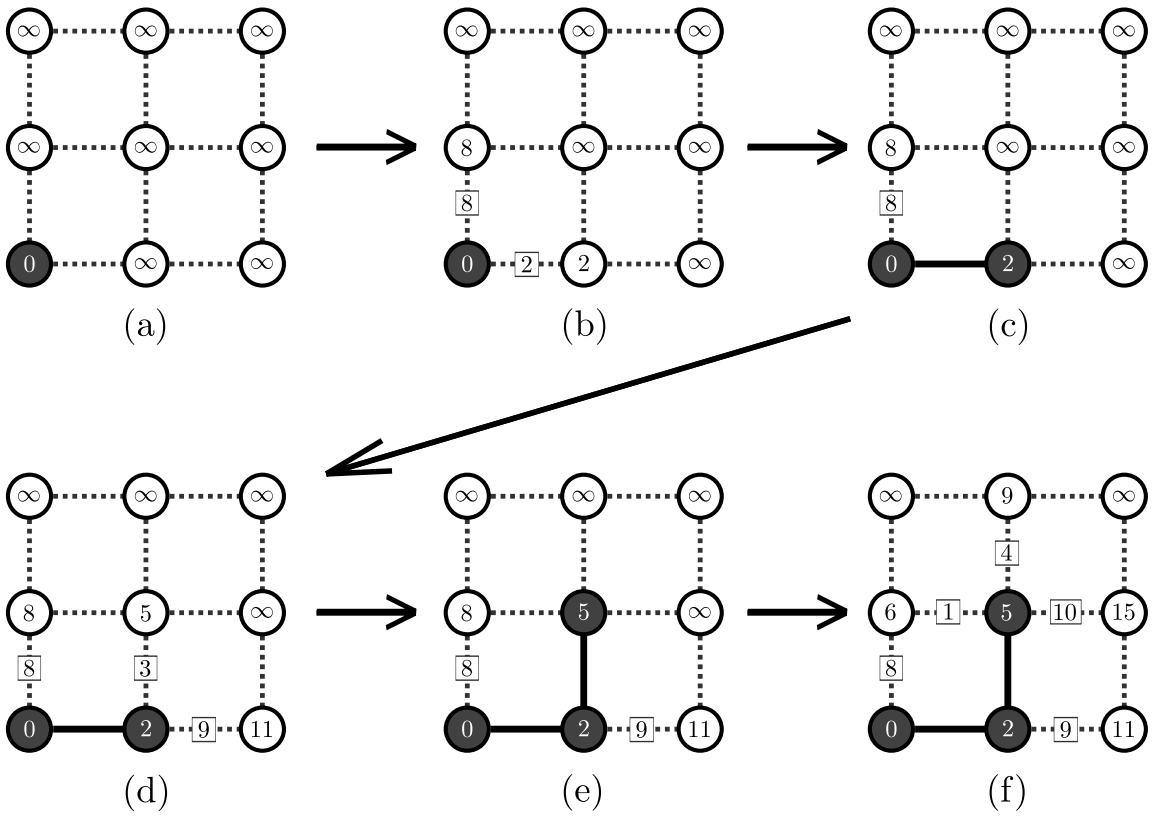

Figure 3.4: In Dijkstra's algorithm, the distance to all nodes is initially set to be infinite while the distance to the source or root node is set to zero, as shown in (a). Here the current minimum reachable distance to nodes at a given step is shown as a number inside each node, and the connectivity of the graph is shown with dotted lines denoting edges. As nodes are added to the tree (shown as the shaded circles), the weight of edges adjacent to the growing tree are used to update the minimum reachable distance to nearby nodes. Once all reachable distances are updated, the closest reachable node not currently in the tree is selected and added to the tree. The process is then repeated until all nodes are in tree and both the shortest path and minimum distance to every node from the given source is known, forming a shortest paths tree for the given source. Loops are avoided, and as can be seen from steps (e) to (f), the minimum reachable distance to a node not yet in the shortest paths tree may be updated multiple times before that node gets added to the tree. The greedy nature of the selection process for adding nodes to the tree ensures that globally minimum paths are found.

Dijkstra's algorithm (one for each boundary node). This gives a total running time of $O(N^{3/2}\log N)$. However, we see that by using a dynamically updated tree in Philip Klein's multiple-source shortest paths algorithm, we can achieve a much shorter running time.

3.3.4 Klein MSSP algorithm

Klein's algorithm is a multiple-source shortest paths algorithm, so we can use it to construct the shortest paths trees rooted at each of boundary nodes in a planar lattice. Whereas we would have to run Dijkstra's single-source shortest paths algorithm once for each of the $4(L-1)$ boundary nodes in a square lattice, Klein's algorithm computes all of these trees in one cohesive routine. It has a total amortized running time of order $N\log N$ (for a system of $N=L^2$ vertices), which is impressively the same asymptotic running time as a *single* call to Dijkstra's algorithm. Note that a planar input graph is required for Klein's algorithm, so we are restricted to two dimensional magnets in this case.

The algorithm begins with a single-source shortest paths tree T_0 rooted at the first boundary node r_0. To create T_0, we simply run Dijkstra's algorithm once with r_0 as the source. Then, rather than recomputing the entire tree T_i for each successive root (source) r_i along the system boundary, this MSSP algorithm maintains a dynamic tree that is repeatedly altered to become a shortest paths tree for each successive boundary source, progressing in a counter-clockwise fashion around the system boundary. This procedure of changing roots is sketched in Fig. 3.5. The series of alterations that transform one shortest paths tree to the next are called "pivots," as each alteration adds one edge into the tree and removes another nearby edge. Often, trees T_i and

T_{i+1} whose roots are adjacent to each other only differ by a few edges, and only a a handful of pivots is required. By reusing the tree structure from one root to the next, we avoid having to recompute a large number of paths in each tree.

The upper bound on running time relies on the fact that each each undirected edge is only pivoted into or out of the tree at most twice (or once in the case of directed edges) and the fact that we use dynamic tree data structures that can perform searches, updates, and accumulations along paths in $O(\log N)$ time. In order to efficiently implement Klein's algorithm, special care must be taken in choosing data structures. Throughout the course of the algorithm, we must maintain not only the current tree T that we are dynamically altering, but also its dual or interdigitating tree T^*. At any given time, T^* contains the edges in the dual graph that map to edges *not* in T. As can be seen in Fig. 3.6, between T and T^*, all edges in the square lattice are accounted for. Since it contains information about all the edges not in T, the dual tree T^* is of central importance when searching for edges to pivot into T.

The choice for which edges to pivot into T is made by examining the reduced length or *slack cost* for primal edges, $\hat{\ell}(u\,v) = d(u) + \ell(u\,v) - d(v)$. Here $\ell(u\,v)$ is the length (weight) of edge $\langle u\,v\rangle$ connecting vertices u and v. $d(u)$ and $d(v)$ are the current from-root distances to u and v. Slack costs are maintained for all edges in T and in T^* at all times. Since our lattice has undirected edges, the slack costs for both the forward and reverse direction for edges must be stored. Intuitively, an edge's slack cost represents the amount that a path length can be shortened by adding that edge to T. Of course, edges already in T will have a slack cost of zero (in the forward direction). For pivots, edges with a forward slack cost less than zero are selected. In simplest terms, if an edge $\langle u\,v\rangle$ has a forward slack cost $\hat{\ell}(u\,v) < 0$, this means that adding that $\langle u\,v\rangle$ (the edge *from* u

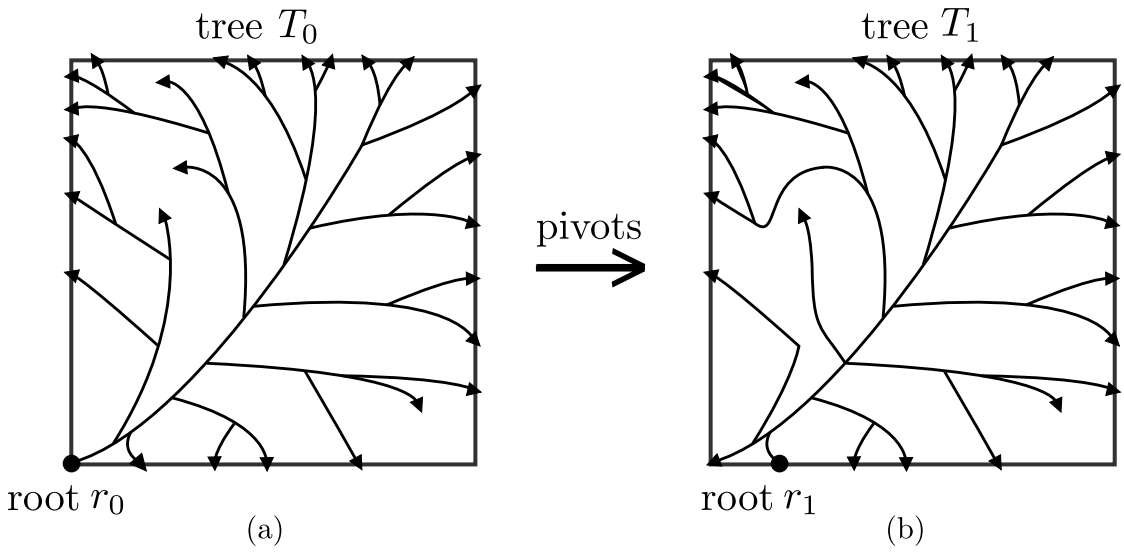

Figure 3.5: In Philip Klein's multiple-source shortest paths algorithm, Dijkstra's single-source shortest paths algorithm is first used to initialize a shortest paths tree T_0 for some initial source or root r_0, as is depicted in (a) with arrows pointing away from the tree's root. Then, the tree is re-rooted at the next root along the system boundary, r_1, and a series of pivots are carried out to update the tree so that contains shortest paths with r_1 as the source. We then call this new tree T_1, which is identical to the tree we would construct if we performed Dijkstra's algorithm with r_1 as the source. In this way, the same tree is dynamically updated to become a shortest paths tree for each of the $4(L-1)$ roots along the system boundary in succession. As can be seen by comparing (a) and (b) in this sketch, the shortest paths trees for sources near each other often share much of the same structure, with only a few differing edges. Intuitively, reusing the common structure of these shortest paths trees is what allows for the speedup that Klein's algorithm offers.

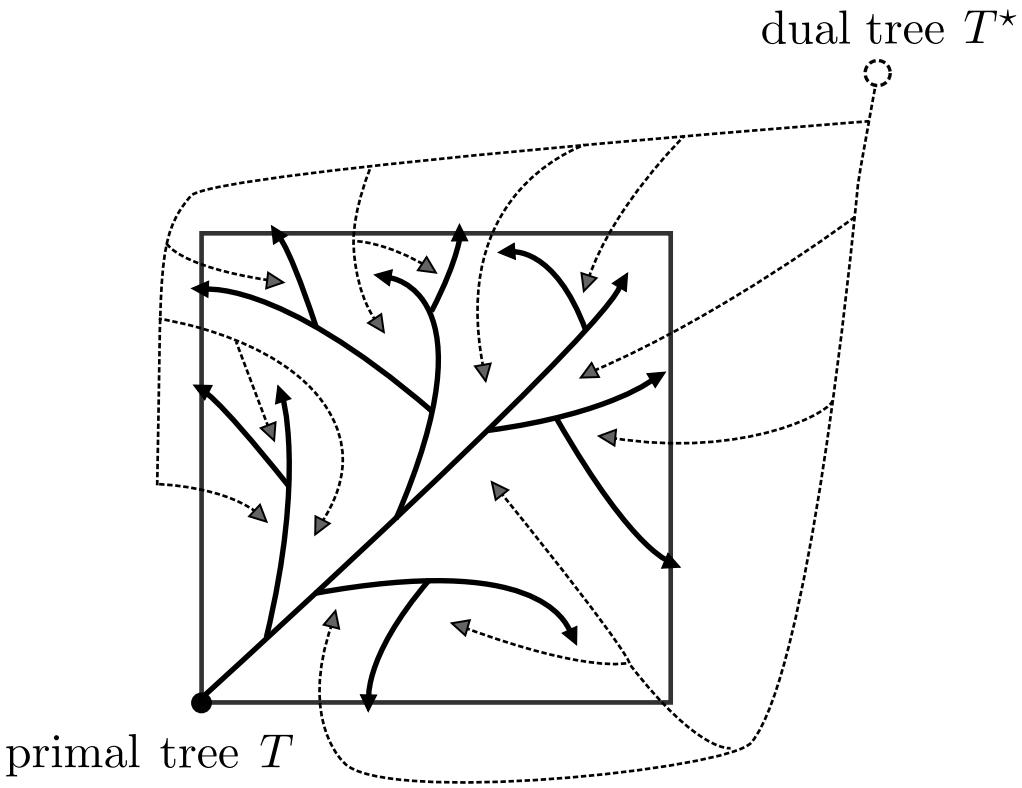

Figure 3.6: Shown here is a sketch of a shortest paths tree T on a square lattice (shown as solid lines) and its interdigitating dual tree T^* (shown as dashed lines). As shown here, faces in the primal graph become vertices in the dual graph, and the root of T^* is the infinite face outside the outer system boundaries. Edges in the dual graph cross edges in the primal graph, so edges can be mapped from dual to primal (and vice-versa) in a one-to-one fashion. All of the edges in T^* will map to edges in the primal lattice that are *not* in tree T. Maintaining this dual tree structure is convenient (and more importantly, efficient) when searching for edges to pivot into tree T.

to v) to the tree T would make the root-to-v path shorter, resulting in a more correct shortest paths tree for the given source. When an edge is added to T, another edge is ejected from the tree, as each node can only have a single parent, and the number of edges in T remains constant.

As can be seen in Fig. 3.7, the process of changing roots from r_i to r_{i+1} begins with a "special pivot," whereby the edge $\langle r_{i+1} r_i \rangle$ is included in the tree T, causing the entire tree to be effectively re-rooted at the new root, r_{i+1}. In the figure, r_i and its descendants are colored in blue, while the remaining nodes are colored red. The red-to-blue edges are searched for the edge with the most negative slack cost, and this edge is pivoted into T. Recall that an edge with negative slack cost is one that would make from-root paths in T shorter if added to T. Adding blue-to-red edges would not make any of these from-root paths in T shorter, so we only need to consider adding red-to-blue edges here. The dual tree T^* is used to search for this minimal slack cost edge efficiently. With the "dtree" data structure provided in David Eisenstat's library [136], we can maintain, update, and search slack costs along paths in $O(\log N)$ time, which makes the usage of this dual tree T^* crucial to achieving an efficient running time with this algorithm. As this pivot process is carried out iteratively, nodes are re-colored as they switch from blue to red. After enough iterations, the tree T becomes the shortest paths tree T_{i+1} for root r_{i+1}.

When this sequence of pivots is complete and the shortest paths tree T_{i+1} is fully updated, we can take statistics of this tree, which is identical to the tree that would be created using Dijkstra's algorithm, modulo any path degeneracies (paths of equal length). The correctness of this algorithm is proven in Refs. [8, 164]. We have also exhaustively verified the agreement between Dijkstra's algorithm and our implementation of Klein's algorithm (modulo de-

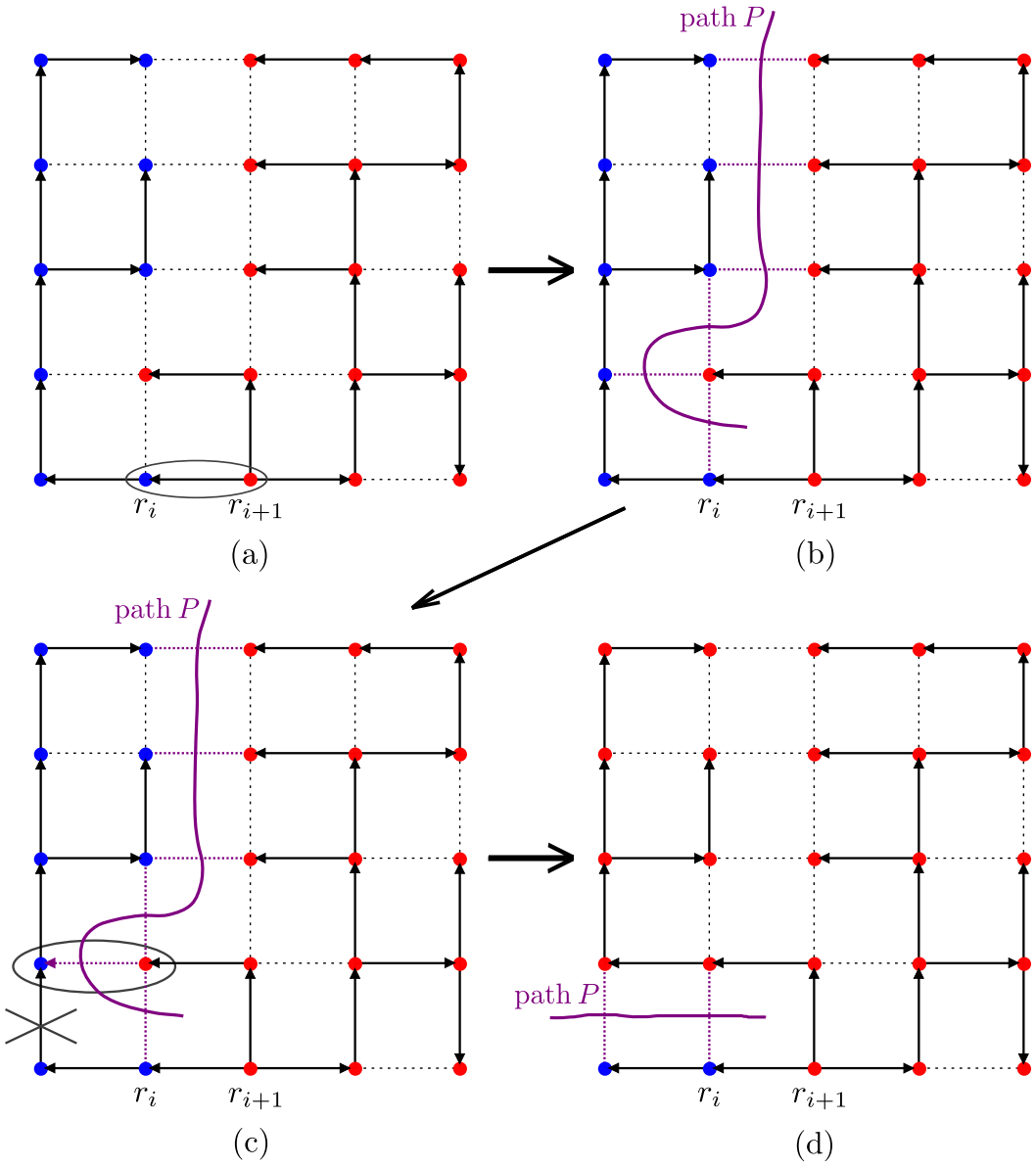

Figure 3.7: When updating from tree T_i to T_{i+1}, the tree is first re-rooted at the new root, r_{i+1}, as depicted in (a). This is accomplished by adding the edge from r_{i+1} to r_i to the tree. This entails either performing a special pivot or merely reversing the direction of that particular edge. The subtree containing the old root r_i and its descendants is colored blue, and the remaining nodes form a subtree rooted at the new root r_i that is colored red. As shown in (b), path P in the dual tree (shown in magenta here) is searched for those red-to-blue edges with negative slack cost. Once the minimum negative slack edge along P is found, it is pivoted into the tree. Because each node in a tree can only have one parent, when an edge is pivoted into the tree, another edge must be pivoted out of the tree. The process then repeats until the shortest paths tree for root r_{i+1} is established. As seen in (c) and (d), this pivoting process results in blue nodes turning red as they are removed from the blue subtree and added into the red subtree.

generacies for paths of equal length) for a large number of samples. We investigated the frequency of these path length degeneracies and found that we could avoid all degeneracies with very high probability if we used 64-bit length values (for systems as large as 2048^2). Due to the extreme rarity of these degeneracies, they should not have any significant influence on our scaling results.

3.3.5 Algorithm Performance

Repeating the process of pivots for each of the $4(L-1)$ boundary sources, the entirety of Klein's MSSP algorithm takes running time of order $N \log N$, the same computational complexity as a single call to Dijkstra's algorithm. Thus Klein's algorithm marks a tremendous improvement over traditional methods and allows for the simulation of larger systems, more samples, and better statistics. For comparison, see Table 3.1 for timing information on the MSSP problem, both for a Dijkstra implementation and an implementation using Klein's algorithm. For these timing estimates, we used an Intel Core i7-2600K CPU @3.40 GHz with 8 cores and 16GB of RAM. It is easy to see from this table that when using Dijkstra's algorithm we were extremely limited by running time (rather than memory constraints) and were only able to simulate systems as large as 1024^2. Even then, we only had enough CPU time to simulate a few thousand of these large samples. On the other hand, using Klein's algorithm, we were able to simulate one million samples apiece of systems as large as 2048^2 in a reasonable amount of time.

As can be seen in Fig. 3.8, the speedup offered by Klein's algorithm is significant, especially in the limit of large system size. The fit lines in this figure confirm our theoretical assumptions about the asymptotic complexity of Di-

Table 3.1: Runtime information on boundary pairs shortest paths production code

System	Memory (MB)	Dijkstra (s/Sample)	Klein (s/Sample)
8^2	.2	.003	.001
16^2	.3	.03	.006
32^2	.6	.3	.03
64^2	1.7	2.6	.1
128^2	6.3	24	.5
256^2	25	220	2
512^2	90	2000	8.5
1024^2	360	16400 (4.5 hrs)	36
2048^2	1500	estimated 32 hrs	160 (2.7 mins)

jkstra's and Klein's algorithms. Namely, using Dijkstra's algorithm for every boundary node has a complexity of order $N^{3/2}\log N$, while the complexity of Klein's algorithm is only of order $N\log N$. This speedup by a factor of $N^{1/2} = L$ is considerable when L is large.

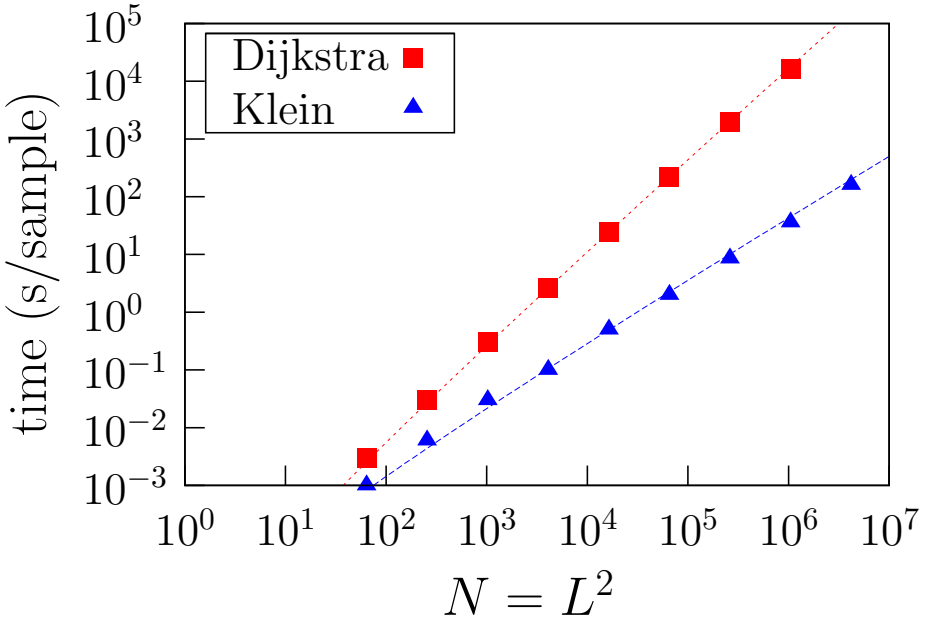

Figure 3.8: Running time estimates for calculating all $4(L-1)$ boundary-rooted shortest paths trees for systems with a total number of nodes $N = 8^2$ through $N = 2048^2$. Here we observe the speedup gained by using Klein's multiple-source shortest paths algorithm versus using Dijkstra's single-source shortest paths algorithm repeatedly. The dotted lines show the expected asymptotic running time behavior, $O(N \log N)$ for Klein's algorithm and $O(N^{3/2} \log N)$ for repeated uses of Dijkstra's algorithm. Note the logarithmic scale on both axes. With fitting coefficients, the asymptotic running times plotted here are $(1.2 \times 10^{-6}) \times N^{3/2} \log N$ for Dijkstra and $(3.1 \times 10^{-6}) \times N \log N$ for Klein.

3.4 Analysis Methods and Numerical Results

Here we outline the analysis performed after using Klein's algorithm to find the shortest paths for all boundary sources. Using the raw output data, we determine whether a given node is part of a system-spanning path and is therefore "controllable" via changing boundary conditions. We find this probability to scale with system size like $P \sim L^{-\gamma}$. We introduce scaling exponents ρ and

ω to compare data across different system sizes and use a chi-squared fit to estimate the value of γ for each pair of consecutive system sizes. We extrapolate γ to the thermodynamic limit ($L \to \infty$). For uncorrelated disorder, we find $\gamma = 0.33(1)$. We also discuss the implications of our results in terms of the multiplicity of ground states in the random ferromagnet. Lastly, we explore lattices with edge weights that have built-in spatial correlations and compare these results with those of Schorr and Rieger [12]. See Table 3.2 and Figure 3.19 for a summary of these results. Unfortunately, as we'll discuss, an issue with the part of our correlation procedure where we ensure non-negative weights for the shortest paths algorithm causes these results to be fairly inconclusive.

3.4.1 Output

During the course of Klein's algorithm, every time a shortest paths tree T_i for some boundary root r_i is created, we take statistics for a set of nodes in this tree. Of primary interest is whether a given node u belongs to a system-spanning path for the given tree T_i, which would mean that this point is *controllable* via changing boundary conditions—that is, a domain wall could be induced that passes through the corresponding location in the random ferromagnetic spin system. By accumulating statistics of whether a node is controllable for each of the roots $\{r_i\}$ throughout the course of the algorithm, we can create a table for a given node that displays precisely the trees $\{T_i\}$ which pass through a node, i.e., the trees for which the node is controllable by a domain wall terminated at boundary point r_i.

Since each (directed) edge is added and removed from the dynamic shortest paths tree at most once during the running of Klein's algorithm, the roots for

which each node is controllable will form a finite number of contiguous "intervals" of boundary roots [164]. Specifically, each directed edge will have at most one continuous interval of roots for which it is part of a system-spanning path, so each undirected edge will have at most two such intervals. Since each internal node (not on the boundary) has a coordination number of four, each node will have at most eight intervals for which it is controllable. Fig. 3.9 shows a visualization of these intervals, color-coded by roots along the boundary. Qualitatively, this gives us some intuition about the uncontrollable regions of spins in the interior of the system (the black regions in the figure), as well as some insight into the orientation of system-spanning paths (whether paths span across the system horizontally or vertically).

Since it takes $O(\log N)$ time to query a point, in order to maintain short run times and limit output data file size, we only query and record intervals for a limited set of points. As sketched in Fig. 3.10, we chose to query points in a "union jack" formation—along the diagonals of the system, as well as horizontal and vertical lines across the system. In order to further reduce output file size, which became a concern when storing intervals for one million samples per system size, we only record every n^{th} node along these diagonal, horizontal, or vertical lines, increasing n by a factor of two for each larger system in order to keep output file size constant.

As will be explored in the following section, from these intervals we can bin the points at a given distance D from the system boundaries and compute an effective probability $P(D)$ for these points to be controllable. The distance D is measured using concentric squares layered inward, starting at $D = 0$ at the boundary and reaching at $L/2$ in the center of the system. At each layer, we check to see how many nodes and edges are controllable compared to the

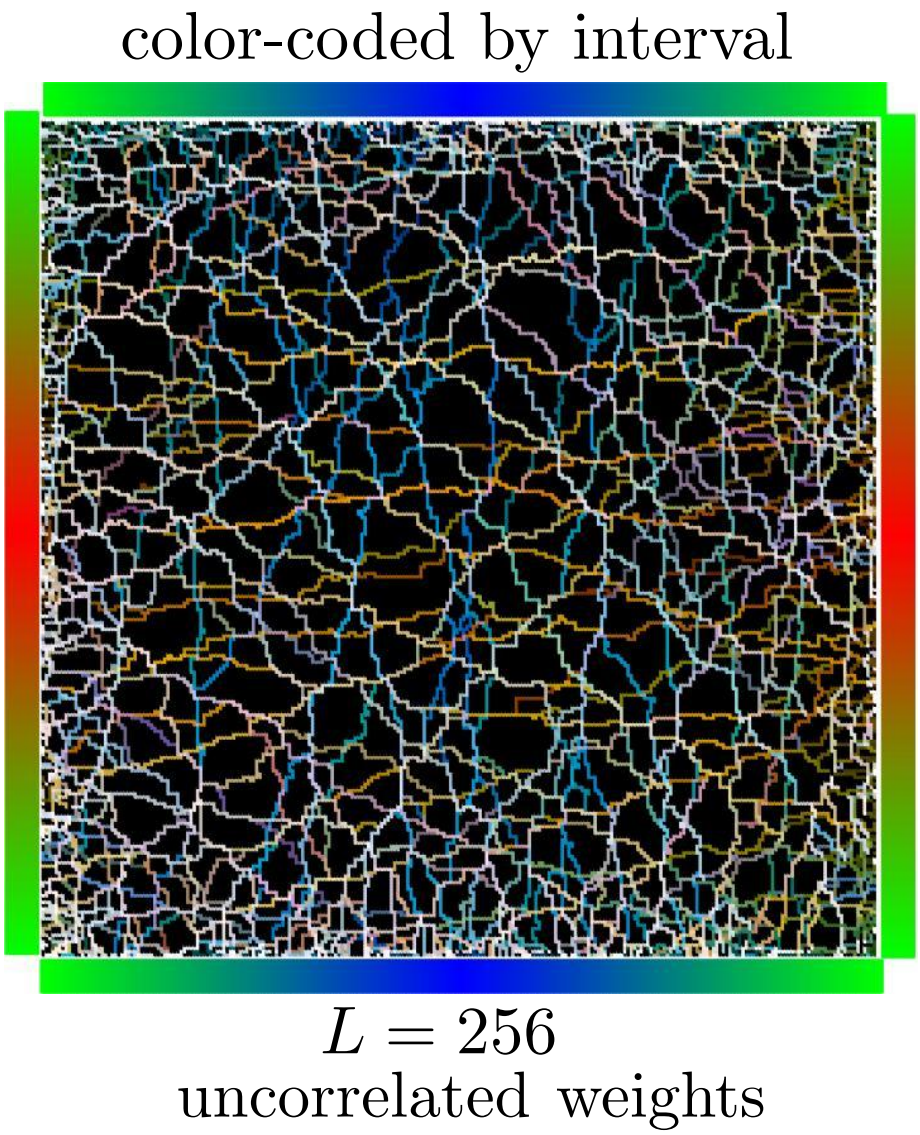

Figure 3.9: Presented here is a visualization of an sample of size $L = 256$. For this visualization, every point in the system is queried for each tree T_i with root r_i on the system boundary. In this way a set of intervals are formed for each node, denoting for which trees (which roots) the point is controllable. These roots are color-coded as seen in the color key around the system boundary, and each point is then colored according to the designated color of the roots in its interval. For example, the points colored in mostly red are controllable for those shortest paths trees rooted midway up the left or right side of the system boundary, where red is shown in the color key surrounding the system.

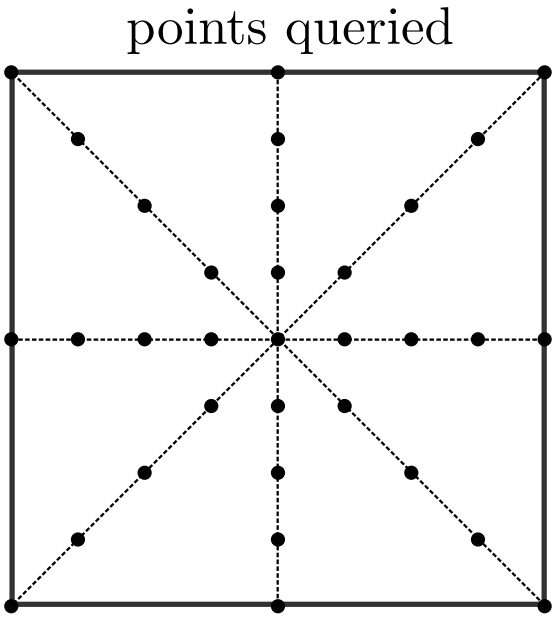

Figure 3.10: This diagram shows the points we queried to check for controllability, including points along the diagonals of the system as well as horizontal and vertical lines across the system's center. Querying every point in the $L \times L$ system would bottleneck running time and effectively destroy the drastic speedup afforded by Klein's algorithm, so an abridged set of queried points was necessary in the production code.

total number queried at that layer. That gives the effective probability P for each D value. The $P(D)$ data for all runs (10^6 samples per system size) is split into $N_b = 400$ uniform batches (histograms) of 2500 samples apiece. From here, an average over all samples is easily computed, with error bars determined via bootstrap resampling of the batches. The data in split into batches in part because running the bootstrap analysis over all 10^6 samples individually is time consuming. A batch size of 400 allows for error bars of order $1/\sqrt{N_b} \approx 5\%$.

3.4.2 Scaling

The primary output we examined quantitatively was $P(D)$, the probability for nodes a distance D from the system boundary to be controllable (part of a system-spanning shortest path). This probability decreases as distance from the boundary increases. Nodes in the center of the system are further removed from the boundary and more difficult to influence due to the tendency of domain walls to wander with some roughness. But how does this probability scale with system size L?

Our assumption is that at long length scales our shortest paths should asymptotically resemble directed polymers, since overhangs are seen to be irrelevant. From the directed polymer result, independent paths of length L tend to separate by L^{ζ}, where $\zeta = 2/3$ in two dimensions [66, 77, 79, 82]. So the chance of hitting the center of the system with a single attempt (single source) is

$$P_1 \sim L^{-\zeta} \ . \tag{3.4.2.1}$$

With many sources, yielding m *independent* tries, the probability to hit the center with one or more of these m tries, P_m, is equal to 1 minus the probability to miss with all independent tries:

$$P_m = 1 - P_{\text{miss all}} = 1 - (P_{\text{miss single try}})^m = 1 - (1 - P_1)^m \ . \tag{3.4.2.2}$$

Here, for all pairs of boundary points, our estimate for the number of independent tries m is equal to

$$m \sim (\frac{L}{L^{\zeta}}) = L^{1-\zeta} \ . \tag{3.4.2.3}$$

This is based on the conjecture that L^{ζ} gives the amount of separation needed between boundary sources needed to have roughly independent attempts, along

the boundary of our system that has total length $4L \sim L$. Assuming P_1 is small in the thermodynamic limit, we can write:

$$\begin{aligned} P_m &= 1-(1-P_1)^m \\ &\approx 1-(1-mP_1) \\ &\sim mL^{-\zeta} \\ &\sim L^{1-2\zeta} , \end{aligned} \tag{3.4.2.4}$$

where we have used Eqs. 3.4.2.1 and 3.4.2.3. Thus we have our scaling relation for P, the total probability of central nodes being controllable in a system of size L:

$$P \sim L^{-\gamma} , \tag{3.4.2.5}$$

where we have defined $\gamma = 2\zeta - 1$.

Using this scaling relation, $2D/L$ is a natural scaling parameter, since at the center of the system where paths are furthest from the system boundaries, $D/(L/2) = 1$. The standard one parameter finite size scaling assumption is then that $P(D)$ will scale like $L^{-\gamma}$ multiplied by some unknown function of the argument $2D/L$. The form of the scaling hypothesis is that

$$\begin{aligned} P(D) &\approx L^{-\gamma} f\left(\frac{D}{L/2}\right) \\ &\approx L^{-\gamma} f\left(\frac{2D}{L}\right) , \end{aligned} \tag{3.4.2.6}$$

where the scaling function $f(\omega)$ behaves as $f(\omega) \approx c_1$ for some constant c_1 for $\omega \ll 1$, $f(\omega) \approx 0$ as $\omega \to \infty$. For more compact formulas, we define the scaled dimensionless variables

$$\rho = P(D)L^{\gamma} , \tag{3.4.2.7}$$

$$\omega = \frac{2D}{L} . \tag{3.4.2.8}$$

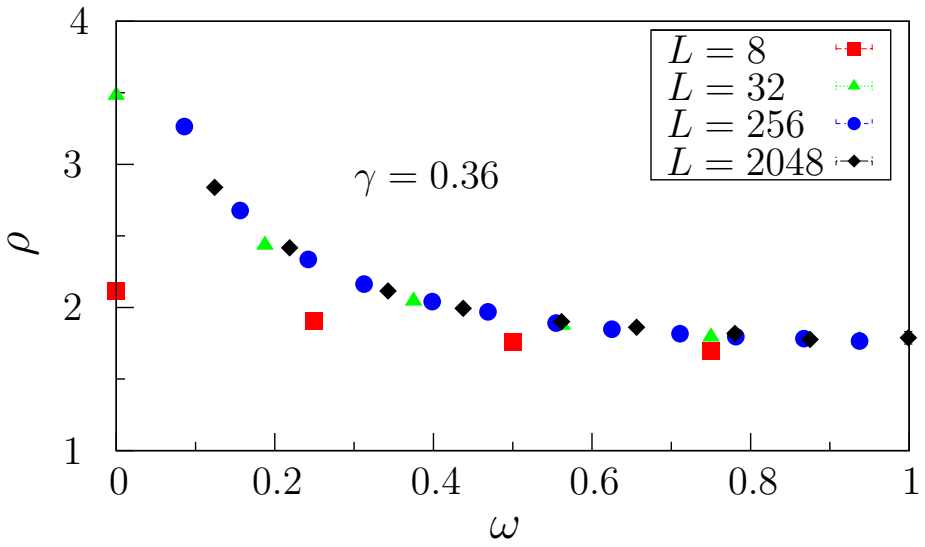

Figure 3.11: Scaling collapse for various two-dimensional RBIM systems and uncorrelated bond couplings. In this plot, $\gamma = 0.36$, which appears to be moderately close to the true value of the fitting parameter γ, judging by eye from the goodness of the collapse. Note that error bars are smaller than the symbol size for all points except the right-most point.

so that our scaling hypothesis becomes

$$\rho \approx f(\omega) \ . \tag{3.4.2.9}$$

Using these scaled variables ρ and ω, we can plot data for different system sizes on the same set of axes, as in Fig. 3.11. The parameter γ can be tuned to adjust the goodness of the collapse in this scaling plot. Due to the scaling relation underlying the choice of these scaled variables, the data for different system sizes will collapse best when γ is tuned to be near its true value.

In order to estimate γ and to quantify the uncertainty in this estimate, we employ an automated fitting procedure. This procedure determines an effective exponent $\gamma(L)$ derived from data for samples of size L and of larger size $2L$ by minimizing a "goodness of fit" parameter χ^2 at each scale L. The only input

to this procedure is an estimate of ω_l, the small path data cutoff, which is discussed in the following section. In essence, this cutoff allows us to fit only data points near the center of the system, since this is the region we are interested in. We then extrapolate $\gamma(L)$ for $L \to \infty$ to get our best estimate for γ.

We employ a standard χ^2 goodness of fit measure. We use subscripts 1 and 2 to indicate data sets for systems of size L_1 and L_2 being compared. We use $L_2 = 2L_1$. A general goodness of fit measure for quantifying how well two curves collapse in variables ρ vs. ω starts with choosing a set of independent points ω_k and at each point calculating the difference between $\rho_1(\omega_k)$ and $\rho_2(\omega_k)$, $\Delta_{12}(\omega_k) = \rho_1(\omega_k) - \rho_2(\omega_k)$. This difference $\Delta_{12}(\omega_k)$ at the point ω_k is then squared and normalized by the sum of the variances, $\sigma_1^2(\omega_k)$ for the system of size L_1 and $\sigma_2^2(\omega_k)$ for the system of size L_2. This gives

$$\chi^2(\gamma; L_1, L_2) = \sum_k \frac{\Delta_{12}^2(\omega_k)}{\sigma_1^2(\omega_k) + \sigma_2^2(\omega_k)} , \tag{3.4.2.10}$$

where we sum over discrete points ω_k chosen with uniform spacing [126]. As depicted in Fig. 3.12, this χ^2 test allows us to quantitatively measure how well the data for a given pair of system sizes collapses as a function of the parameter γ. We seek a value of γ for which this χ^2 is minimized (Fig. 3.12), though the magnitude of our final error bars are determined by resampling.

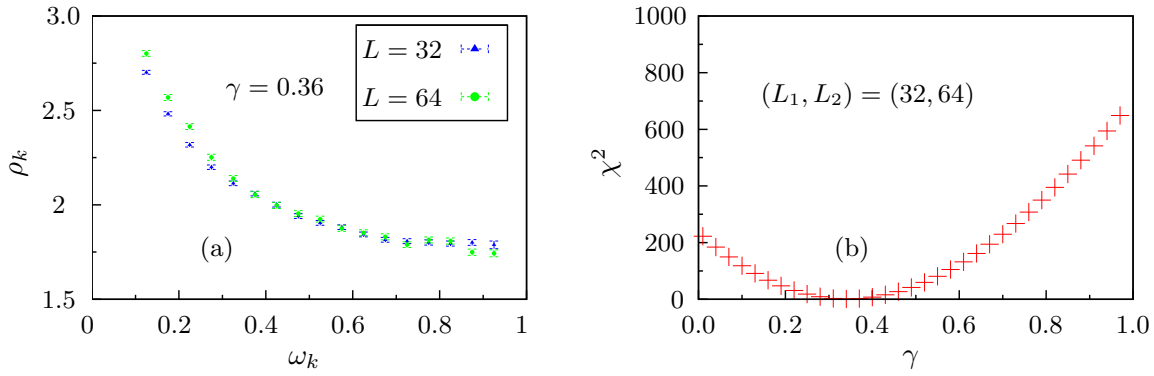

Figure 3.12: A sample collapse for two systems of size $L=32$ and $L=64$ (a) shows the comparison of the two systems at interpolated points ω_k for a value of $\gamma=0.36$. Comparing the value of $\rho_k(L_1=32)$ indicated by the blue triangles and $\rho_k(L_2=64)$ indicated by the green circles at each of these points allows for the calculation of $\chi^2(\gamma; L_1, L_2)$. (b) shows χ^2 as a function of the fitting parameter γ for the same pair of systems. From here we can extract the best estimate of γ by examining where χ^2 is at a minimum.

3.4.3 Extrapolation to Thermodynamic Limit

Now that our automated χ^2 fitting procedure has been established, we must consider exactly what region in ω of the data we want to fit. As we're primarily concerned with behavior near the center of the system (far from the boundaries, where $\omega \to 1$), no maximum cutoff in ω is needed. On the other hand, paths behave differently in the center of the system than they do near the system boundaries, and nodes are much more easily influenced and likely to be controllable closer to the system boundaries. Thus a minimum (lower end) cutoff, ω_l, for the fitting region is appropriate in order to study the physics of the system's deep interior. Examining computed values of γ for various ω_l cutoffs, we determined that a value of $\omega_l = 7/8$ to be reasonable. Above this value there is no change in the computed value of γ within error bars. Using this

cutoff, we effectively fit a region in the center of the system of size $L/8 \times L/8$.

Now that we've decided upon what region to fit, the collapse procedure is run N_b times (once for each batch α of data for each pair of sizes L_1, L_2 with $L_2 = 2L_1$). Each time, the value of γ for which χ^2 is minimized is found, giving N_b independent estimates $\gamma(\alpha, L_1, L_2)$ for a given pair of systems L_1 and L_2. The final estimate of γ for this pair of systems is the average of the N_b independent estimates:

$$\overline{\gamma(L_1, L_2)} = \frac{1}{N_b} \sum_{\alpha=1}^{N_b} \gamma(\alpha, L_1, L_2) \; . \tag{3.4.3.1}$$

Then the sample variance $S^2(L_1, L_2)$ in these N_b estimates is used to estimate the statistical error bars in the final estimate,

$$\sigma_\gamma(L_1, L_2) = \frac{S(L_1, L_2)}{\sqrt{N_b - 1}} \; . \tag{3.4.3.2}$$

At this point we obtain a single estimate of γ (with error bars) for each consecutive pair of systems simulated.

Next we look to see if this $\overline{\gamma(L_1, L_2)}$ converges to some value as L_1 and L_2 tend to infinity. We extrapolate the infinite system size limit using a variety of least squares fitting routines adapted from the GNU Scientific Library [135]. Standard scaling suggests that γ as a function of system size L will exhibit power law scaling behavior. But since we have no knowledge of the expected value for the exponent in this power law, we can allow this exponent to vary as a parameter in our fit, plotting $\overline{\gamma}$ vs. $L^{-\lambda}$, where we take L to be the geometric mean $L = \sqrt{L_1 L_2}$. Allowing λ to vary, fitting the data gives an estimate for γ as well as the correction to scaling exponent λ. See Fig. 3.13 for an example of one extrapolation attempt, where λ is seen to be 0.5 and linear least squares fitting for $\overline{\gamma}$ vs. $L^{-\lambda}$ is used for the three largest system sizes. Due to

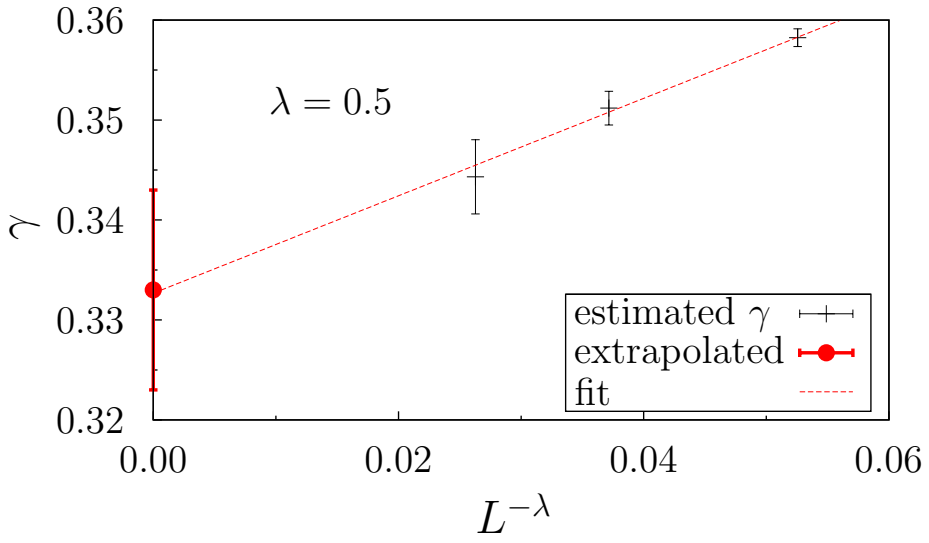

Figure 3.13: An illustration of a linear least squares fitting method being used to extract the value of γ in the infinite system size limit. The fit uses the form $\gamma(L) = AL^{-\lambda} + \gamma(\infty)$ where λ is a correction to scaling exponent and A and $\gamma(\infty)$ are fitting parameters. Here $\lambda = 0.5$, and $L = \sqrt{L_1 L_2}$, where L_1 and L_2 are the two system sizes used to produce a given data point. The fit found gives $\gamma(\infty) = 0.33(1)$.

the slow convergence in our scaling form, data from systems other than the three largest pairs (that is, smaller than $L = 256$) were excluded from the final fit. This exclusion underscores the vital need for large system sizes in simulations where the thermodynamic limit is relevant. Ultimately, we find that in the thermodynamic limit, $\gamma = 0.33(1)$, which is compatible with our hypothesis that γ should be related to the directed polymer result by the relation $\gamma = 2\zeta - 1 = 1/3$.

3.4.4 Multiplicity of Ground States

Through the examination of domain wall wandering at the interior of spin systems, there has been sophisticated analysis performed to investigate the multiplicity of thermodynamic states by looking at "windows" of fixed volume in the center of disordered spin systems as system sizes are increased [11, 20, 97–99]. This technique is paramount in characterizing the multiplicity of thermodynamic states, particularly in spin glasses. This type of analysis explores the idea of connectivity, probing how strongly spin domains are coupled to each other (and to boundary spins) and investigating the occurrence of "islands" of spin domains on the system's interior that are completely uncoupled from surrounding spins. In this analysis, spin configurations and correlation functions in a small window of fixed volume W^d are monitored as the spin system is repeatedly increased in size (from linear size L to a larger L' in a given iteration), which effectively changes the boundary conditions for the finite-volume window. This procedure is illustrated in Fig. 3.14. A convergence of these spin configurations as $L \to \infty$ implies a single thermodynamic state. This can be seen by a deflection of domain walls away from the center window as the system size is increased toward infinity.

For our analysis, we once again examine a box in the center of the system of linear size $W = L/8$, keeping in line with our choice for ω_l. The difference is that now we are interested in computing the *total* probability of being controllable, P_W, for all points within the central region of size W, rather than being parameterized by distance D from the system boundary. Computing P_W for each of the one million samples simulated for a given system size, we can observe the distribution of these probabilities, as seen in Fig. 3.15. Based on

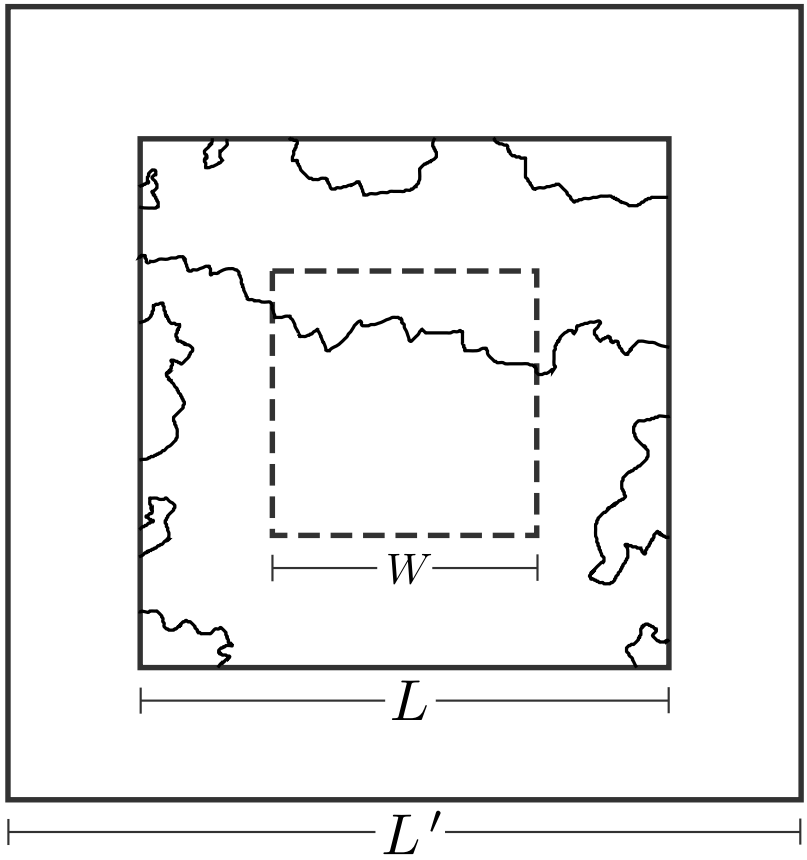

Figure 3.14: An illustration of the procedure for examining domain walls in in a fixed volume of linear size W (shown as a dashed box) as the linear size of the spin system (shown as a solid box) is increased from L to L', as depicted in Refs. [9, 10]. We use $W = L/8$ and $L' = 2L$. Deflection of domain walls away from the center window as system size approaches the thermodynamic limit implies convergence to a single thermodynamic state (or pair of states related by a global spin flip). This procedure is useful for investigating the multiplicity of thermodynamic states in disordered spin systems and has been applied to the RFIM, RBIM, and spin glasses.

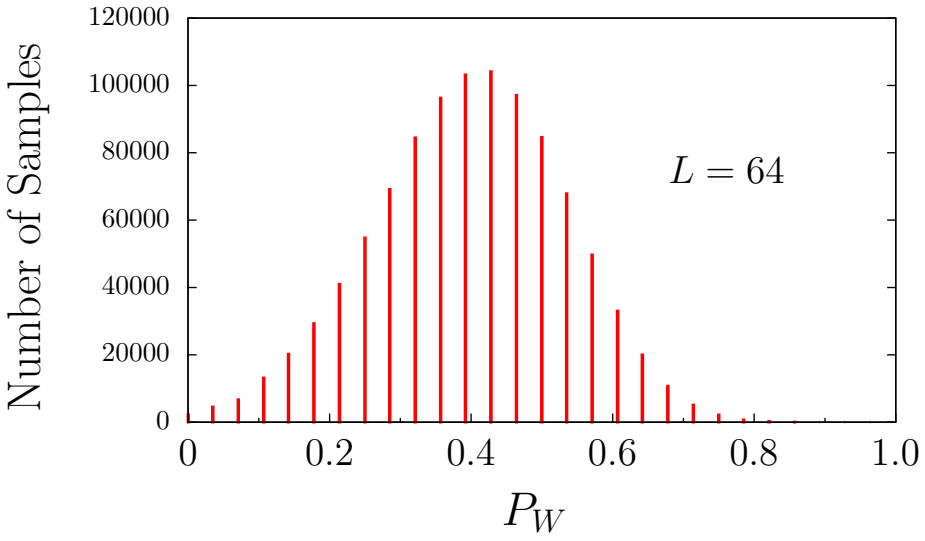

Figure 3.15: Histogram displaying the distribution of P_W for one million samples of size $L = 64$. P_W represents the total probability of being controllable for all points in a box (window) of size $W = L/8$ at the center of the system.

these histograms, the distribution appears to be close to Gaussian, justifying the use of root-mean-square fluctuations to determine error bars for the estimate of P_W averaged over all one million samples.

At this point we've computed an estimate for $P_W(L)$ for each system size. Following the similar scaling form

$$P_W \sim L^{-\gamma_W} \,, \tag{3.4.4.1}$$

extracting the slope of a log-log plot for $P_W(L)$ versus L should produce an estimate of γ_W. As displayed in Fig. 3.16, we perform a fit $P_W(L) = AL^{-\gamma_W} + P_W(\infty)$, where A, γ_W, and $P_W(\infty)$ are scaling parameters. Here we find $\gamma_W = 0.33(1)$, which is compatible with our previous result of $\gamma = 0.33(1)$. We also see that $P_W(\infty) = -0.1(1)$, meaning that in the thermodynamic limit, the probability of the center window W being controllable (P_W) decays to zero (within error bars).

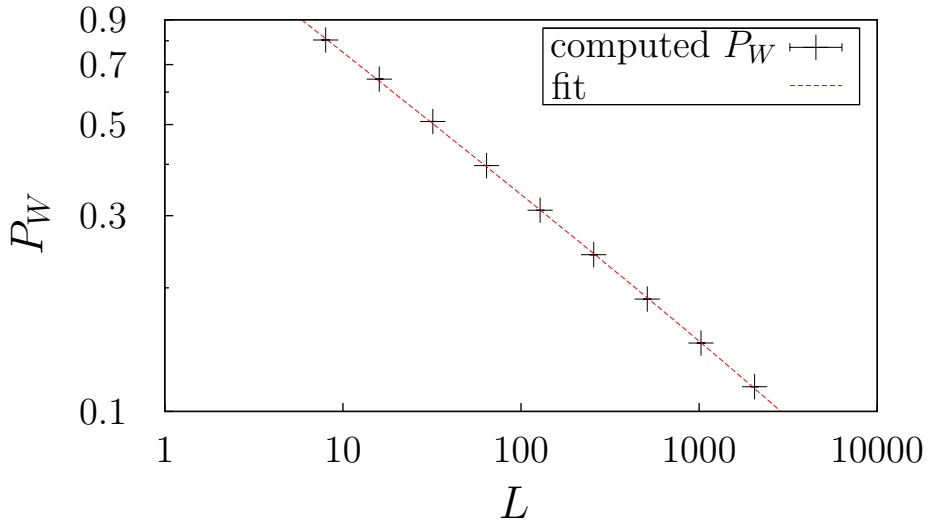

Figure 3.16: Log-log plot of P_W versus L for various system sizes. P_W represents the total probability of being controllable for all points in a box (window) of size $W = L/8$ at the center of the system. Performing a fit with $P_W(L) = AL^{-\gamma_W} + P_W(\infty)$ and fitting parameters A, γ_W, and $P_W(\infty)$ in order to extract the slope of this plot, we find $\gamma_W = 0.33(1)$, which is compatible with our previous result of $\gamma = 0.33(1)$.

This means that as $L \to \infty$, domain walls in the RBIM deflect away from the window of size W, implying the existence of a single pair of ground states (related by a global spin flip). This corroborates well-known results for the two-dimensional RBIM [97, 177].

3.4.5 Correlated Edge Weights

In addition to lattices with edge weights drawn randomly and uniformly on $(0, 1)$, we also examine systems with built-in spatial correlations that decay as a power law $C_n \sim (1 + n^2)^{-\eta/2}$ for two points separated by spatial distance n. This is achieved through a modified Fourier filtering method [175, 176].

The general process we use to construct correlated data sets follows fairly

closely with the process laid out in the 1995 work by Makse-et-al [175]. Starting with some uncorrelated two-dimensional Gaussian data $\{u_j\}$, with a mean of zero and standard deviation of one, one can apply a Fourier transform to generate $\{\hat{\mathcal{U}}_q\}$. Similarly, we take the correlation function $C_n = (1+n^2)^{-\eta/2} = (1+n_x^2+n_y^2)^{-\eta/2}$ and Fourier transform this to obtain $\hat{\mathcal{C}}_q$. Next, taking advantage of the convolution theorem, we convolve u and C in Fourier space (since it's more efficient than convolving in real space) to obtain what we'll call

$$\hat{\mathcal{G}}_q = \sqrt{\hat{\mathcal{C}}_q} \cdot \hat{\mathcal{U}}_q \ . \tag{3.4.5.1}$$

From here, we can perform an inverse Fourier transform on $\hat{\mathcal{G}}_q$ to obtain $\{g_j\}$, the final correlated data set. In practice, we actually generate random Gaussian values for the Fourier weights $\{\hat{\mathcal{U}}_q\}$ directly rather than generating random Gaussian values for the real space weights $\{u_j\}$ in order to avoid an extra Fourier transform step, thus improving efficiency.

Fig. 3.17 shows computed correlations in one of these such data sets for several values of our correlation strength parameter η. These correlations in g_j follow the power law form set by C_n and fall close to the expected fit lines. This agreement should be expected, as the convolution theorem allows us to write:

$$\begin{aligned} C_n = \langle g_j g_{j+n} \rangle &= \mathbf{IFT}([\mathbf{FT}(g_j)]^2) \\ &= \mathbf{IFT}([\hat{\mathcal{G}}_q]^2) \\ &= \mathbf{IFT}([\sqrt{\hat{\mathcal{C}}_q} \cdot \hat{\mathcal{U}}_q]^2) \\ &= \mathbf{IFT}(\hat{\mathcal{C}}_q \cdot \hat{\mathcal{U}}_q^2) \\ &= \mathbf{IFT}(\hat{\mathcal{C}}_q) \\ &= C_n \ . \end{aligned} \tag{3.4.5.2}$$

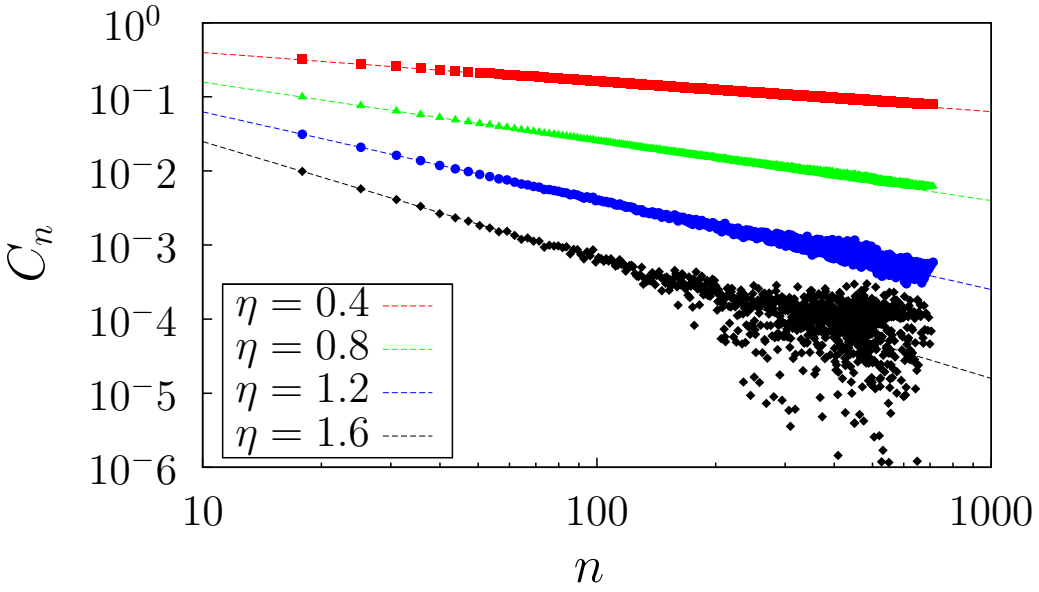

Figure 3.17: Computed two-point correlation function C_n for points separated by spatial distance n in a system of size $L = 1024$ over 1000 samples. Edge weights have built-in spatial correlations that decay according to a power law, $(1+n^2)^{-\eta/2}$. The data points for computed values of C_n fall close to expected fit lines for four different values of η.

In the final steps we have used the fact that $\mathcal{U}$ is Gaussian distributed with a variance of one, meaning that $\hat{\mathcal{U}}_q^2 = 1$ on average. These results are identical whether the correlations are computed exhaustively in real space (with a complexity of $O(N^2)$) or quickly ($O(N \log N)$) in Fourier space, taking advantage of the convolution theorem and fast Fourier transforms offered by libraries such as the GNU Scientific Library [135]. Of course, fast Fourier transforms are ideal for our purposes, as the $O(N \log N)$ complexity will not bottleneck the running time of Klein's algorithm.

There is one caveat to this procedure for imparting correlations that we must address when using the procedure to generate correlated edge weights for shortest paths. Due to the construction using random Gaussian data sets, the final correlated data will contain both positive and negative values. This is problematic for our purposes of equating these values to Ising spin couplings, as we only deal with ferromagnetic (positive) couplings. Additionally, we must avoid negative weights in order to use our algorithms for computing shortest paths. Thus, we must first apply a uniform shift to the correlated data set before we can use the values as edge weights. By adding some constant h to all values in the final correlated data set g_j, we can ensure that all values will be non-negative, as long as we choose h to be large enough. We explored several options for a choice of shift h, but the simplest method which we chose was to shift all $\{g_j\}$ by the minimum negative value among the $\{g_j\}$ for that particular set. Thus, we bump the lowest value in the set up to zero, and everything else in the set will be at least as high as that. This choice of shift is the smallest h we can possibly choose to still ensure non-negative weights and thus represents the smallest perturbation to the correlation structure that is built into the weights. Despite our shift being the minimum required for non-negative

weights, we still have concerns about the shift procedure diluting the degree to which the correlated disorder in the edge weights affects the locations of optimal paths. In the square lattice, if the correlated disorder is small compared to the typical edge weight, paths will tend to be straighter as overhangs (paths wandering) become more costly. The shift procedure effectively introduces a surface tension term in this case, and thus the geometric length of paths becomes more important than the disorder in the edge weights. The consequence of this shift procedure is skewed values of γ, and we believe our results for the case of correlated disorder on the square lattice reflect this.

Introducing a uniform shift decreases the significance of both the correlations and the disorder in edge weights, pushing all edge weights closer to uniform and causing the system to more closely resemble a pure (ordered) ferromagnet with uniform couplings. Fig. 3.18 displays the color-coded controllability intervals for systems of size $L = 256$ for four different values of the correlation strength parameter η. Qualitatively, we do see that stronger correlations (smaller η) lead to larger uncontrollable domains in the system's interior, as high weight edges are more likely to be located within some proximity to each other, creating high weight areas that system-spanning shortest paths tend to avoid. The smaller the value of η, the longer the spatial reach of these correlations, and the larger these regions tend to be. On the other hand, for large values of η, the system begins to resemble its uncorrelated counterpart, as correlations are weak enough that they do not play a significant role in determining the location of system-spanning shortest paths. In either case, however, paths look suspiciously straighter than in the uncorrelated system. Comparing with Fig. 3.9, it seems apparent that even for $\eta = 5.0$ where correlations are weak enough to be insignificant, paths are straighter and closer to-

gether than in the uncorrelated (and more importantly, *un-shifted*) case. This closeness of paths is reflected in the fact that we estimate $\gamma(\eta = 5.0) \approx 0.26(3)$, lower than the expected value of $1/3$ and our estimated value of $\gamma = 0.33(1)$ for the uncorrelated case. We note that we also estimate $\gamma = 0.26(3)$ for uncorrelated Gaussian disorder with a shift applied to ensure non-negative weights. This simply reflects the fact that for large values of η, correlations are weak enough that the system is virtually uncorrelated, and the shift procedure to ensure non-negative weights is the factor having the most significant impact on the value of γ.

Table 3.2 summarized our results for γ at various of the correlation strength parameter η. In Figure 3.19 we compare these results to results computed by Schorr and Rieger for an optimal paths problem in a similar correlated disorder. As can be seen in this figure, our results aren't entirely inconsistent with those of Schorr and Rieger, but they aren't in strong agreement either. Due to the skewing of γ by the shift procedure we used to ensure non-negative correlated edge weights (necessary to compute shortest paths), it is difficult to make any conclusive claims about our results for the correlated disorder. Future refinement of the correlation routine so as to avoid this shift procedure could greatly improve the accuracy of these results.

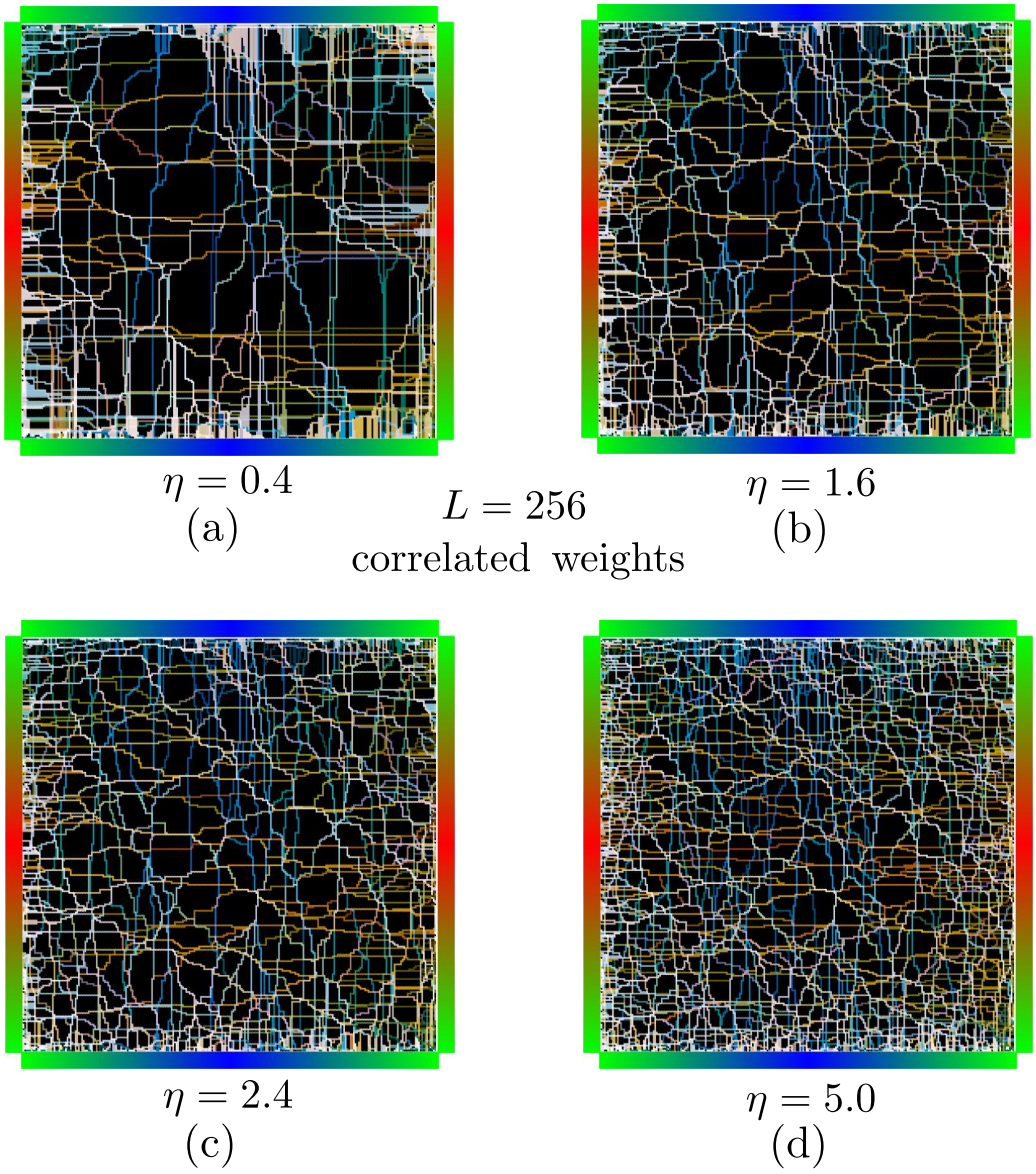

Figure 3.18: Color-coded visualizations (similar to Fig. 3.9) for a system of size $L = 256$, with edge weights having spatial correlations that fall off as $(1+n^2)^{-\eta/2}$ for points separated by spatial distance n. Every point in the system is queried for each tree T_i with root r_i on the system boundary. An interval is formed that denotes for which trees (which roots) the point is controllable. These roots are color-coded as seen at the system boundary, and each point is then colored according to its interval. Subfigures (a) through (d) show the color-coded systems for various values of the parameter η that governs the strength of the built-in correlations in edge weights. For smaller values of η, edge weights are more strongly correlated, and larger regions of uncontrollable points (black regions in this figure) are seen. Higher values of η (weaker correlations) lead to figures that look more reminiscent of uncorrelated edge weights, as in (d).

Table 3.2: γ calculated using Klein's algorithm

uncorrelated	γ
uncorrelated	.33(1)
η	$\gamma(\eta)$
0.4	0.62(4)
0.75	0.53(3)
0.9	0.49(3)
1.0	0.47(3)
1.1	.44(3)
1.25	0.42(3)
1.6	0.36(3)
1.9	0.33(3)
2.1	0.32(3)
2.4	0.30(3)

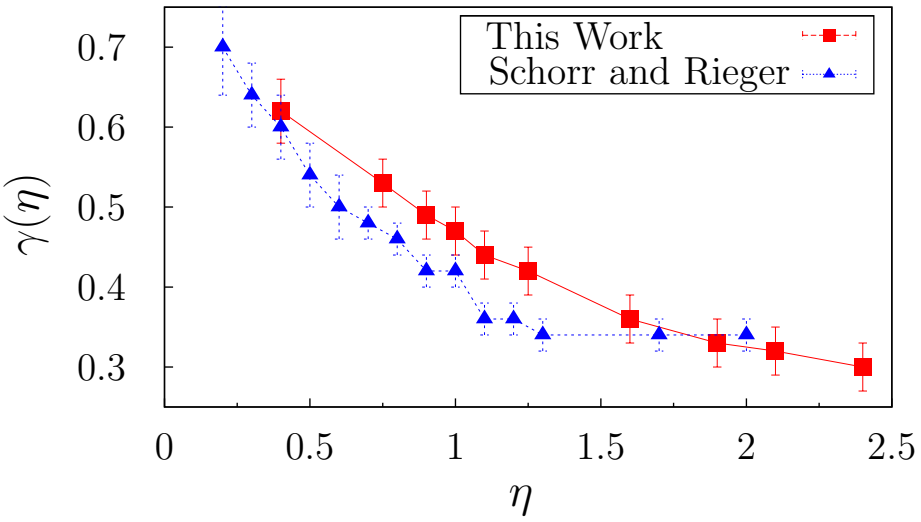

Figure 3.19: A comparison of our computed γ for various values of the correlation strength parameter η, compared with results for optimal paths computed by Schorr and Rieger [12]. The shift procedure we employed to ensure non-negative weights makes it difficult to make any conclusive claims about the agreement (or lack thereof) between our results and theirs.

3.5 Summary

The primary purpose of this work was to apply Klein's algorithm to investigate domain walls in ground states of the random bond Ising magnet, via a direct mapping to the shortest paths problem. By offering a total complexity of $O(N \log N)$, Klein's algorithm is a marked improvement over traditional methods whose complexity is $O(N^{3/2} \log N)$. This speedup allowed for the simulation of larger systems and more samples than previously possible, which is essential when exploring a system in the thermodynamic limit.

Also vital to the feasibility of exploring large systems was the decomposability of domain walls in two dimensions. By greatly reducing the number of boundary conditions that must be checked, this decomposability converted

the complexity of the problem from exponential time to polynomial time. It is our hope that the proof included for this decomposability is of general use for future work.

Through our simulations, we probed the probability P that nodes in the interior of an $L \times L$ system are controllable via changing boundary conditions. We showed that this probability has the asymptotic scaling form

$$P \sim L^{-\gamma} ,$$

and we computed $\gamma = .33(1)$, which is compatible with the scaling relation $\gamma = 2\zeta - 1$ that we derived to relate γ to the wandering exponent of directed polymers, ζ. We also confirmed the known result that the RBIM has a single pair of ground states.

We also explored RBIM systems whose edge weights were spatially correlated according to a power law, using a Fourier filtering method [175, 176] to achieve these long-range correlations. We set out to compute γ for various correlation strengths η and compare our results with work by Schorr and Rieger [12]. Unfortunately our procedure for generating correlated edges proved to be problematic, and in ensuring non-negative weights we effectively made the disorder less relevant to the dynamics of the system. Our results for these systems with correlated weights, while not inconsistent with results by Schorr and Rieger, are not of high enough precision or quality to provide a conclusive statement of their merit.

This work was made possible in part by NSF Grant No. DMR-1006731 and by the Syracuse University Gravitation and Relativity computing cluster, which is supported in part by NSF Grant No. PHY-0600953. This work was carried out largely using the Syracuse University OrangeGrid, a computing

resource of approximately 10,000 desktop computers supported by Syracuse University.

3.6 Appendices

In these appendices, we present more precise definitions of the mapping from the RBIM to the shortest paths problem. We also include a proof for the decomposability of the RBIM domain walls in two dimensions. Lastly, we provide detailed pseudocode of the algorithms we used to solve the shortest paths problem.

3.6.1 Appendix A: Definitions and Mapping to Shortest Paths

We consider a random ferromagnet model defined on a square lattice of $L \times L$ Ising spins forming a set S of spins. Each spin i can take spin values $\{+1,-1\}$, where we call $+1$ *up* and -1 *down*. We write this by defining a spin function $s(i) = s_i : i \to \{+1,-1\}$. We denote the set of all bonds by B, where each nearest neighbor pair of spins is connected by bond $\langle ij \rangle \in B$. The strength of ferromagnetic interaction between nearest neighbor pair of spins $\langle ij \rangle$ is given by the function $J(\langle ij \rangle) = J_{ij} : \langle ij \rangle \to \mathbb{R}_{>0}$ so that the Hamiltonian is

$$H = -\sum_{\langle ij \rangle} J_{ij} s_i s_j \tag{3.6.1.1}$$

The primal graph $G(S,B)$ consists of vertices (spins) $s_i \in S$ and undirected edges (bonds) $\langle ij \rangle \in B$ connecting adjacent spins. We will be interested in fixing the

outermost spins, *boundary spins*, and allowing the rest of the interior spins to vary.

We also need to consider the dual graph to this square array of Ising spins. On the dual graph $G^*(V,E)$, each plaquette or face of the primal graph becomes a vertex μ. These dual vertices form a square lattice, and nearest neighbors μ and ν in this dual lattice are then connected by undirected dual edges $\langle\mu\nu\rangle \in E$. To the exterior of this lattice of dual vertices we also add $4(L-1)$ dual vertices, which we will identify as the *boundary vertices* $V_b \subseteq V$ of the dual graph. Each of these boundary vertices has exactly one incident dual edge that connects it to a vertex on the outermost layer of vertices in $V \setminus V_b$, such that each vertex in V_b is connected to a different vertex in $V \setminus V_b$.

Each bond $\langle ij\rangle$ between spins i and j maps to an edge $\langle\mu\nu\rangle \in E$ that crosses the bond and connects dual vertex μ to dual vertex ν. We define a length function $\widetilde{J}(\langle\mu\nu\rangle) = \widetilde{J}_{\mu\nu} : \langle\mu\nu\rangle \to \mathbb{R}_{>0}$ There is a one-to-one mapping for bonds in the primal to edges in the dual, and $J_{ij} = \widetilde{J}_{\mu\nu}$ for each mapped bond. This dual mapping is depicted in Fig. 3.3. A *realization* is an assignment of J_{ij} to all nearest neighbor pairs $\langle ij\rangle$ in the primal graph. Equivalently, a realization is an assignment of all lengths $\widetilde{J}_{\mu\nu}$ in the dual graph.

Given a spin assignment s_i, a bond $\langle ij\rangle$ is *satisfied* if $s_i s_j = 1$ and *unsatisfied* if $s_i s_j = -1$. In the primal graph, a *cluster boundary* is a set of adjacent unsatisfied bonds that forms a boundary for a connected cluster of identical spins. In the dual graph, this list of bonds becomes a list of edges forming a path, which we'll call a *domain wall*. Thus a domain wall is a path in the dual graph and only contains edges that map to unsatisfied bonds. This path can either be a cycle or a connected path terminating at exactly two boundary vertices. We'll call a domain wall that forms a cycle in the dual graph a *contractible domain*

wall, and we'll call a domain wall that forms a path terminating at two boundary vertices in the dual graph a *boundary-terminated domain wall*. Note that on the square lattice, there can often be multiple ways to partition the set of all edges mapped from unsatisfied bonds into several domain walls. For a cluster boundary Λ, equivalent to a path P on the dual graph, the total energy of Λ (or equivalently the total length ℓ of path P) is given by

$$H_\Lambda = \sum_{\langle ij\rangle \in \Lambda} J_{ij} = \sum_{\langle \mu\nu\rangle \in P} \widetilde{J}_{\mu\nu} = \ell(P)\,. \tag{3.6.1.2}$$

Minimizing the total energy of a cluster boundary is equivalent to minimizing the total length of a path in the dual graph (finding a shortest path).

A *domain wall endpoint* is a boundary vertex whose incident edge is part of a domain wall. For a given spin assignment, we'll define the set of all unsatisfied bonds by $B_u \subseteq B$ and the set of all satisfied bonds by $B_s \subseteq B$ such that $B_u \cup B_s = B$ and $B_u \cap B_s = \emptyset$. Similarly, we'll define E_u as the dual edges that map to bonds in B_u and E_s as the dual edges that map to bonds in B_s. While the *energy* of a given spin assignment s_i is given by the Hamiltonian (Eq. 3.6.1.1), we define the *cost* C of a spin assignment by the sum of the unsatisfied bonds in that spin assignment,

$$C = \sum_{\langle ij\rangle \in B_u} J_{ij} = \sum_{\langle \mu\nu\rangle \in E_u} \widetilde{J}_{\mu\nu} = \sum_{P} \ell(P)\,, \tag{3.6.1.3}$$

where the last sum is over all of the domain walls, P. So the cost of a domain wall is precisely equivalent to the energy of a domain wall. Decomposing the

Hamiltonian, we see that

$$\begin{aligned} H &= -\sum_{\langle ij\rangle} J_{ij} s_i s_j \\ &= -\sum_{\langle ij\rangle \in B_s} J_{ij} + \sum_{\langle ij\rangle \in B_u} J_{ij} \\ &= -\sum_{\langle ij\rangle \in B_s} J_{ij} + \sum_{\langle ij\rangle \in B_u} J_{ij} + \left(\sum_{\langle ij\rangle \in B_u} J_{ij} - \sum_{\langle ij\rangle \in B_u} J_{ij} \right) \\ &= -\sum_{\langle ij\rangle} J_{ij} + 2 \left(\sum_{\langle ij\rangle \in B_u} J_{ij} \right) \\ &= E_0 + 2C \ , \end{aligned} \tag{3.6.1.4}$$

where we have defined the minimal ground state energy

$$E_0 = -\sum_{\langle ij\rangle} J_{ij} \ , \tag{3.6.1.5}$$

and we can easily see that

$$C = \frac{E - E_0}{2} \ . \tag{3.6.1.6}$$

Physically, we can think of C as a measure (exactly half) of the energy cost paid in order to change from a spin assignment of all up spins (where all the J_{ij} are satisfied) to a spin assignment where some bonds are unsatisfied.

We are interested in fixing the boundary conditions—choosing up or down for all spins on the boundary of the system and requiring all spin assignments to abide by these choices. We define a choice of boundary conditions by choosing a value of $+1$ or -1 for each spin on the boundary. Since the boundary is a cycle, the number of sign changes in S_i along the boundary is even, so there are an even number of domain wall endpoints fixed by the given choice of boundary conditions. We'll denote a choice of boundary conditions by $S_b = \{s_i | i \in \text{boundary}\}$.

For a given realization and for given choice of boundary conditions S_b, the *ground state* is the spin assignment that minimizes the total system energy (the Hamiltonian given by Eq. 3.6.1.1) subject to S_b. Note that a ground state cannot have contractible domain walls, since flipping the spins within a contractible domain wall P lowers the system cost by $\ell(P)$ without affecting the choice of boundary conditions S_b. Thus, we only consider boundary-terminated domain walls in the ground state.

Any boundary-terminated domain wall $P_{\alpha\beta}$ (having ends $\alpha, \beta \in V_b$) divides S into two clusters, $S_L(P_{\alpha\beta})$ and $S_R(P_{\alpha\beta})$. $S_L(P_{\alpha\beta})$ will be the cluster of spins bounded by the bonds mapped from the edges in $P_{\alpha\beta}$ and the system boundary clockwise from a to b. Meanwhile $S_R(P_{\alpha\beta})$ will be the cluster of spins bounded by the bonds mapped from the edges in $P_{\alpha\beta}$ and the system boundary clockwise from b to a. It is easy to see that $S_L(P_{\alpha\beta}) \cup S_R(P_{\alpha\beta}) = S$ and $S_L(P_{\alpha\beta}) \cap S_R(P_{\alpha\beta}) = \emptyset$.

3.6.2 Appendix B: Proof of Domain Wall Decomposability

As a note, since Klein's algorithm requires planarity to work, we restrict ourselves to the two-dimensional RBIM. While a proof of decomposability may potentially be constructed for more than two dimensions, domain walls can only be represented as paths in the two-dimensional case. In three dimensions, the interfaces between regions of up and down spin manifest as two-dimensional surfaces, which falls outside the scope of this proof.

For any set of paths $\{P_n\}$ on the dual graph, we define the *XOR sum* of these paths as the set of edges formed by the "exclusive or" edge-wise addition $\bigoplus$ of paths P_n:

$$\bigoplus_{n=1}^{N} P_n = P_1 \bigoplus P_2 \bigoplus \cdots \bigoplus P_N \ . \tag{3.6.2.1}$$

Lemma 1 Given a set of *end-disjoint* (sharing no domain wall endpoints in common) domain walls, the XOR sum of these domain walls forms a set of unsatisfied bonds corresponding to spin assignment (there are no "dangling ends" of domain wall paths).

Proof For a single path, each vertex in $V \setminus V_b$ will have zero, two, or four incident edges that are part of the XOR sum. Each of the boundary vertices V_b which will have exactly one incident edge, which is why the domain walls being end-disjoint is important. As we add additional paths to the XOR sum, this number of incident selected edges for vertices in $V \setminus V_b$ will increase by two (modulo two), since edges appear and disappear in pairs due to the XOR nature of the addition. Thus each vertex will always have an even number of incident edges that are part of the XOR sum, and we will have no dangling edges.

Lemma 2 For any set of domain walls $d = \{P_n\}$,

$$C(\bigoplus_n P_n) \leq \sum_n C(P_n) \,, \tag{3.6.2.2}$$

with the inequality becoming an equality in the case that the domain walls in set d are edge-disjoint and a strict inequality in the case that they are not edge-disjoint.

Proof If the domain walls of d are edge-disjoint, each edge in d appears exactly once in the set, and so the XOR sum of d is the union of all the edges in the domain walls of d, meaning that the cost of the set can be decomposed exactly into a sum over each individual domain wall:

$$C(\bigoplus_n P_n) = \sum_n C(P_n) \,. \tag{3.6.2.3}$$

If the domain walls share edges, we can have one or more edges from a domain wall that do not appear in the XOR sum, or we can have an edge appear in the XOR sum and be triple-counted (or worse) in the sum over individual domain walls. Both of these scenarios serve to lower the total cost of the XOR sum compared to the sum of costs over the individual domain walls. In this case we have

$$C(\bigoplus_n P_n) < \sum_n C(P_n)\ . \tag{3.6.2.4}$$

There is no way for the XOR sum of d to contain any edges that are not in one of the domain walls of d, and so there is no way for the cost of the XOR sum of d to be higher than the sum of the costs of the domain walls in d.

Lemma 3 For a ground state with $2N$ domain wall endpoints, the set of edges $g = \{\langle \mu\nu \rangle\}$ that map to unsatisfied bonds can be decomposed into a set of N edge-disjoint boundary-terminated domain walls $d = \{P_n\}$.

Proof Suppose we have a ground state with $2N$ domain wall endpoints forming a set b, and the set of edges g that map to unsatisfied bonds in this ground state. The unsatisfied bonds divide the system into $2N$ or fewer contiguous regions of up or down spins, which we'll call *spin clusters*. Because the ground state cannot have contractible domain walls, all of these spin clusters include some number of boundary spins, which determine whether the spin cluster is up or down (to align with the fixed spins on the boundary). In between these spin clusters are the $2N$ domain wall endpoints and the N domain walls connecting them. If we begin with an empty set of domain walls $d = \{\}$, we can populate this list by repeating the following steps until b is empty:

- Let the dual vertex v be the first element currently in b.

- Starting vertex v, trace a path using only edges in g and using the rightmost edge if there are multiple choices for unsatisfied edges leaving a vertex.
- Once this path terminates at another domain wall endpoint u (which it necessarily must), the total unsatisfied edges in this path forms a domain wall $P_n \subseteq g$.
- Add P_n to d.
- Remove v and u from b.
- Flip $S_R(P_n)$.

Flipping $S_R(P_n)$ serves to remove the edges of P_n from g, while also aligning that spin cluster with its neighbors, and removing two domain wall endpoints, the ends of P_n. It is easy to see that once these steps are repeated N times, there will be no domain wall endpoints remaining, and every spin cluster will be aligned with its neighbors, leaving the system in a state of either all up or all down spins (and thus no unsatisfied bonds remaining). At this time, the set of domain walls d will be edge-disjoint and will contain precisely all of the edges in g.

Lemma 4 Given a ground state with $2N$ domain wall endpoints and a decomposition into a set d of N edge-disjoint boundary-terminated domain walls $d = \{P_n = P_{\alpha_n\beta_n}\}$ (as discussed in Lemma 3), each of these N domain walls $P_n = P_{\alpha_n\beta_n}$ is the shortest α_n-to-β_n path on the entire $L \times L$ system.

Proof Suppose one of these domain walls $P_k = P_{\alpha_k \beta_k}$ is not the shortest α_k-to-β_k path. Then there exists another path P_k' that is shorter, meaning that

$$C(P_k') < C(P_k)\ . \tag{3.6.2.5}$$

Then if we look at the cost of the state with P_k' instead of P_k, calling this cost C', we can write:

$$\begin{aligned} C' &= C([\bigoplus_{n\neq k} P_n] \bigoplus P_k') \\ &\leq C(\bigoplus_{n\neq k} P_n) + C(P_k') \\ &< C(\bigoplus_{n\neq k} P_n) + C(P_k) \end{aligned} \tag{3.6.2.6}$$

where we have used Lemma 2. We can rewrite the final expression on the right side of the inequality:

$$\begin{aligned} C(\bigoplus_{n\neq k} P_n) + C(P_k) &= C(\bigoplus_{n} P_n) \\ &= \sum_n C(P_n) \\ &= C_{\text{ground state}}\ , \end{aligned} \tag{3.6.2.7}$$

where in the final step of this equality we have used Lemma 2 and the fact that the ground state consists of N edge-disjoint domain walls. Combining these two equations, we see that

$$C' < C_{\text{ground state}}, \tag{3.6.2.8}$$

which is a contradiction since the ground state is defined as the minimal cost spin assignment. Thus we cannot have a domain wall in d that is not shortest on the full system $L \times L$.

3.6.3 Appendix C: Algorithms (Pseudocode)

Here we provide detailed pseudocode for Klein's multiple-source shortest paths algorithm and subroutines, including Dijkstra's algorithm which is used to initialize the shortest paths tree. For implementation of the pseudocode outlined in this Appendix, see David Eisenstat's helpful "dtree" coding library [136] based on Sleator-Tarjan dynamic trees [137].

Dijkstra: Dijkstra's Algorithm

assert all primal nodes $u \in V$ have $dist(u) = \infty$
choose origin $o \in V$, $dist(o) \leftarrow 0$
min-*dist* priority queue $Q \leftarrow \{o\}$
tree $T \leftarrow \emptyset$
while Q not empty **do**
    node $v = Q.pop()$
    add v to tree T
    for all *nbrs()* w of v, w not already in T **do**
        if $dist(w) == \infty$ **then**
            add w to Q since it was previously unreachable
        if $dist(v) + length(v,w) < dist(w)$ **then**
            $dist(w) \leftarrow dist(v) + length(v,w)$

Klein: Klein's Algorithm

1: root list $R \leftarrow \{r_0, r_1, \ldots, r_{\max}, r_0\}$

2: **comment:** make sure R lists the roots on the boundary in order going counter-clockwise around the system

3: $oldroot \leftarrow r_0$

4: Analyze tree T_0 where T_0 is the tree T with root r_0

5: **repeat**

6: $\quad newroot = R.next(oldroot)$

7: $\quad$ ChangeRoot($oldroot$,$newroot$)

8: $\quad$ Analyze tree T

9: **until** $newroot == r_0$

ChangeRoot(r_0, r_1): Change T_0 to T_1 by changing its root from r_0 to r_1 and updating via pivots

special pivot to add edge (r_1, r_0) to T
comment: now the tree T is rooted at r_1, although it is not longer a shortest paths tree
let d_{10} be the directed edge $d(r_1, r_0)$ and d^*_{10} its dual
let dual node s be the head of directed dual edge (d^*_{10}), the face in G^* of d_{10} that is not the root of the dual tree T^*
$t \leftarrow -\ell(d_{10})$
repeat
    let P be the path in T^* from s to the root of T^*
    find a directed edge $\hat{d}$ in P whose $\hat{\ell}$ is minimum
    $\Delta \leftarrow min\{\hat{\ell}(\hat{d}), \ell(d_{10}) - t\}$
    comment: $\hat{\ell}(u,v) = dist(u) + \ell(u,v) - dist(v)$
    subtract Δ from the $\hat{\ell}$ values of the rootward edges in P
    add Δ to the $\hat{\ell}$ values of the leafward edges in P
    add Δ to t
    if $\Delta < \ell(d_{10}) - t$ **then**
        let f be the primal directed edge in T whose *head* is $head(\hat{d})$
        pivot $\hat{d}$ into T, ejecting f from T
until $t == \ell(d_{10})$
comment: the tree T is now a shortest paths tree rooted at r_1

Bibliography

[1] R. C. Prim, Bell Syst. Tech. J. **36**, 1389 (1957).

[2] W. Chou and A. Kershenbaum, in *Proceedings of the third ACM symposium on Data communications and Data networks: Analysis and design* (ACM, New York, NY, USA, 1973), DATACOMM '73, pp. 148–156.

[3] S. Wasserman and K. Faust, *Social network analysis: Methods and applications*, vol. 8 (Cambridge university press, 1994).

[4] M. E. J. Newman and M. Girvan, Phys. Rev. E **69**, 026113 (2004).

[5] J. L. Cardy and P. Grassberger, Journal of Physics A: Mathematical and General **18**, L267 (1985).

[6] T. S. Jackson and N. Read, Phys. Rev. E **81**, 021131 (2010).

[7] C. M. Newman and D. L. Stein, Phys. Rev. Lett. **72**, 2286 (1994).

[8] P. N. Klein, in *SODA* (2005), vol. 5, pp. 146–155.

[9] A. A. Middelton, in *New optimization algorithms in physics*, edited by A. K. Hartmann and H. Rieger (John Wiley & Sons, 2006).

[10] A. A. Middleton, Physical review letters **83**, 1672 (1999).

[11] C. M. Newman and D. L. Stein, Journal of Physics: Condensed Matter **15**, R1319 (2003).

[12] R. Schorr and H. Rieger, The European Physical Journal B-Condensed Matter and Complex Systems **33**, 347 (2003).

[13] R. Glantz and M. Hilpert, Phys. Rev. E **77**, 031128 (2008).

[14] E. Fehr, D. Kadau, N. A. M. Araújo, J. S. Andrade, and H. J. Herrmann, Phys. Rev. E **84**, 036116 (2011).

[15] R. Pastor-Satorras and A. Vespignani, Phys. Rev. Lett. **86**, 3200 (2001).

[16] M. A. Moore and H. G. Katzgraber, Phys. Rev. E **90**, 042117 (2014).

[17] H. Luo, M. Schulz, L. SchÃijlke, S. Trimper, and B. Zheng, Physics Letters A **250**, 383 (1998), ISSN 0375-9601.

[18] A. A. Middleton, Phys. Rev. B **61**, 14787 (2000).

[19] A. C. Carter, A. J. Bray, and M. A. Moore, Phys. Rev. Lett. **88**, 077201 (2002).

[20] A. A. Middleton and D. S. Fisher, Phys. Rev. B **65**, 134411 (2002).

[21] D. S. Fisher and D. A. Huse, Journal of Physics A: Mathematical and General **20**, L1005 (1987).

[22] M. Mezard, M. A. Virasoro, and G. Parisi, *Spin glass theory and beyond* (World scientific, 1987).

[23] R. Holtzman and R. Juanes, Phys. Rev. E **82**, 046305 (2010).

[24] R. Holtzman, M. L. Szulczewski, and R. Juanes, Phys. Rev. Lett. **108**, 264504 (2012).

[25] J. F. Peters, M. Muthuswamy, J. Wibowo, and A. Tordesillas, Phys. Rev. E **72**, 041307 (2005).

[26] P. de Gennes, Physica A: Statistical Mechanics and its Applications **261**, 267 (1998), ISSN 0378-4371.

[27] R. S. Farr, J. R. Melrose, and R. C. Ball, Phys. Rev. E **55**, 7203 (1997).

[28] M. E. Cates, J. P. Wittmer, J.-P. Bouchaud, and P. Claudin, Phys. Rev. Lett. **81**, 1841 (1998).

[29] Guimera, R. and Danon, L. and Diaz-Guilera, A. and Giralt, F. and Arenas, A., Phys. Rev. E **68**, 065103 (2003).

[30] M. E. J. Newman, Phys. Rev. E **64**, 016131 (2001).

[31] M. E. J. Newman, Phys. Rev. E **64**, 016132 (2001).

[32] N. Goldenfeld, *Lectures On Phase Transition And The Renormalization Group* (Sarat Book House, 1992), ISBN 9788187169567.

[33] P. Ehrenfest, in *Proc Acad Sci Amsterdam* (1933), vol. 36, pp. 153–157.

[34] L. Landau and E. LIFSCHITZ, Phys. Z. Sowjet **11**, 545 (1937).

[35] L. Landau and E. Lifshitz, *Statistical physics, 1969* (1969).

[36] K. G. Wilson, Reviews of Modern Physics **55**, 583 (1983).

[37] A. Pelissetto and E. Vicari, Physics Reports **368**, 549 (2002).

[38] K. G. Wilson, Phys. Rev. B **4**, 3174 (1971).

[39] K. G. Wilson, Rev. Mod. Phys. **47**, 773 (1975).

[40] L. P. Kadanoff, Physics **2**, 263 (1966).

[41] K. G. Wilson and J. Kogut, Physics Reports **12**, 75 (1974).

[42] J. Faillettaz, F. Louchet, and J.-R. Grasso, Physical review letters **93**, 208001 (2004).

[43] J. M. Carlson and J. Langer, Physical Review Letters **62**, 2632 (1989).

[44] X. Che and H. Suhl, Physical review letters **64**, 1670 (1990).

[45] K. Nagel and E. Raschke, Physica A: Statistical Mechanics and its Applications **182**, 519 (1992).

[46] S. Clar, B. Drossel, and F. Schwabl, Journal of Physics: Condensed Matter **8**, 6803 (1996).

[47] H. Takayasu and H. Inaoka, Physical review letters **68**, 966 (1992).

[48] P. Bak and K. Sneppen, Physical review letters **71**, 4083 (1993).

[49] L. de Arcangelis, C. Perrone-Capano, and H. J. Herrmann, Physical review letters **96**, 028107 (2006).

[50] O. Biham, A. A. Middleton, and D. Levine, Phys. Rev. A **46**, R6124 (1992).

[51] J. v. Neumann and A. W. Burks (1966).

[52] K. E. Wolff, in *Conceptual Structures: Integration and Interfaces* (Springer, 2002), pp. 341–353.

[53] K. Babcock and R. Westervelt, Physical review letters **64**, 2168 (1990).

[54] P. Bak, C. Tang, and K. Wiesenfeld, Physical review letters **59**, 381 (1987).

[55] J. Machta, Y. S. Choi, A. Lucke, T. Schweizer, and L. V. Chayes, Phys. Rev. Lett. **75**, 2792 (1995).

[56] D. Wilkinson and J. F. Willemsen, Journal of Physics A: Mathematical and General **16**, 3365 (1983).

[57] J. Goodman, Journal of Statistical Physics **147**, 919 (2012), ISSN 0022-4715.

[58] M. Damron, A. Sapozhnikov, and B. VÃągvÃűlgyi, Annals of Probability **37**, 2297 (2009).

[59] J. Van Den Berg, A. A. Jarai, and B. Vagvolgyi, Electron. Comm. Probab **12**, 411 (2007).

[60] D. Stein and C. Newman, Physical Review E **51**, 5228 (1995).

[61] M. Damron and A. Sapozhnikov, The Annals of Probability **40**, 893 (2012), ISSN 0091-1798.

[62] H. Loberman and A. Weinberger, J. ACM **4**, 428 (1957), ISSN 0004-5411.

[63] A. W. F. Edwards and L. L. Cavalli-Sforza, Systematics Association Publication. **6**, 67 (1964).

[64] R. Osteen and P. Lin, SIAM Journal on Computing **3**, 23 (1974).

[65] V. Petäjä, M. Sarjala, M. Alava, and H. Rieger, Phys. Rev. B **73**, 094517 (2006).

[66] D. A. Huse and C. L. Henley, Phys. Rev. Lett. **54**, 2708 (1985).

[67] F. Schulz, D. Wagner, and K. Weihe, *DijkstraâĂŹs algorithm on-line: An empirical case study from public railroad transport* (Springer, 1999).

[68] G. Yan, T. Zhou, B. Hu, Z.-Q. Fu, and B.-H. Wang, Phys. Rev. E **73**, 046108 (2006).

[69] A.-L. Barabási and R. Albert, science **286**, 509 (1999).

[70] R. Albert, H. Jeong, and A.-L. Barabasi, Nature **401**, 130 (1999).

[71] M. Faloutsos, P. Faloutsos, and C. Faloutsos, in *ACM SIGCOMM Computer Communication Review* (ACM, 1999), vol. 29, pp. 251–262.

[72] Z. Wu, E. López, S. V. Buldyrev, L. A. Braunstein, S. Havlin, and H. E. Stanley, Phys. Rev. E **71**, 045101 (2005).

[73] P. Echenique, J. Gómez-Gardeñes, and Y. Moreno, Phys. Rev. E **70**, 056105 (2004).

[74] D. Stauffer and A. Aharony, *Introduction to percolation theory* (CRC press, 1994).

[75] S. R. Broadbent and J. M. Hammersley, Proceedings of the Cambridge Philosophical Society **53**, 629 (1957).

[76] H. Frisch and J. Hammersley, Journal of the Society for Industrial and Applied Mathematics **11**, 894 (1963).

[77] D. S. Fisher and D. A. Huse, Physical Review B **43**, 10728 (1991).

[78] M. Kardar, Phys. Rev. Lett. **55**, 2235 (1985).

[79] D. A. Huse, C. L. Henley, and D. S. Fisher, Physical review letters **55**, 2924 (1985).

[80] M. Kardar and Y.-C. Zhang, Phys. Rev. Lett. **58**, 2087 (1987).

[81] J. Bouchaud and H. Orland, Journal of statistical physics **61**, 877 (1990).

[82] K. Johansson, Communications in mathematical physics **209**, 437 (2000).

[83] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson, *Introduction to Algorithms* (MIT Press, Cambridge, MA, USA, 2001), 2nd ed., ISBN 0070131511.

[84] S. Fishman and A. Aharony, Journal of Physics C: Solid State Physics **12**, L729 (1979).

[85] D. Belanger and A. Young, Journal of magnetism and magnetic materials **100**, 272 (1991).

[86] D. C. Hambrick, J. H. Meinke, and A. A. Middleton, arXiv preprint cond-mat/0501269 (2005).

[87] J. A. d'Auriac, M. Preissmann, and R. Rammal, Journal de Physique Lettres **46**, 173 (1985).

[88] S. Istrail, in *Proceedings of the thirty-second annual ACM symposium on Theory of computing* (ACM, 2000), pp. 87–96.

[89] J.-C. Picard and H. D. Ratliff, Networks **5**, 357 (1975).

[90] F. Barahona, Journal of Physics A: Mathematical and General **18**, L673 (1985).

[91] A. T. Ogielski, Physical review letters **57**, 1251 (1986).

[92] L. Ford and D. R. Fulkerson, Princeton U. Press, Princeton, NJ (1962).

[93] A. V. Goldberg and R. E. Tarjan, Journal of the ACM (JACM) **35**, 921 (1988).

[94] Y. Imry and S.-k. Ma, Physical Review Letters **35**, 1399 (1975).

[95] G. Grinstein and S.-k. Ma, Phys. Rev. B **28**, 2588 (1983).

[96] D. S. Fisher, Phys. Rev. Lett. **56**, 1964 (1986).

[97] C. Newman and D. Stein, Physical Review E **57**, 1356 (1998).

[98] M. Palassini and A. Young, Physical Review Letters **83**, 5126 (1999).

[99] C. Newman and D. Stein, Journal of statistical physics **106**, 213 (2002).

[100] D. Stein and C. M. Newman, *Spin glasses and complexity.* (Princeton, NJ: Princeton University Press, 2013), ISBN 978-0-691-14733-8/pbk; 978-1-400-84563-7/ebook.

[101] S. F. Edwards and P. W. Anderson, Journal of Physics F: Metal Physics **5**, 965 (1975).

[102] F. Barahona, Journal of Physics A: Mathematical and General **15**, 3241 (1982).

[103] W. Cook and A. Rohe, INFORMS Journal on Computing **11**, 138 (1999).

[104] F. Barahona, M. Grotschel, M. Junger, and G. Reinelt, Operations Research **36**, 493 (1988).

[105] D. S. Fisher and D. A. Huse, Physical review letters **56**, 1601 (1986).

[106] M. Palassini, F. Liers, M. Juenger, and A. Young, Physical Review B **68**, 064413 (2003).

[107] A. Bray and M. Moore, Journal of Physics C: Solid State Physics **17**, L463 (1984).

[108] A. K. Hartmann, Physical Review E **59**, 84 (1999).

[109] A. K. Hartmann, A. J. Bray, A. Carter, M. Moore, and A. Young, Physical Review B **66**, 224401 (2002).

[110] I. Campbell, A. K. Hartmann, and H. G. Katzgraber, Physical Review B **70**, 054429 (2004).

[111] W. McMillan, Journal of Physics C: Solid State Physics **17**, 3179 (1984).

[112] A. J. Bray and M. A. Moore, Phys. Rev. B **31**, 631 (1985).

[113] A. Bray and M. Moore, Physical review letters **58**, 57 (1987).

[114] D. S. Fisher and D. A. Huse, Phys. Rev. B **38**, 386 (1988).

[115] D. Sherrington and S. Kirkpatrick, Physical review letters **35**, 1792 (1975).

[116] G. Parisi, Physical Review Letters **43**, 1754 (1979).

[117] M. Talagrand, Annals of Mathematics pp. 221–263 (2006).

[118] E. Marinari, G. Parisi, F. Ricci-Tersenghi, J. J. Ruiz-Lorenzo, and F. Zuliani, Journal of Statistical Physics **98**, 973 (2000).

[119] A. T. Ogielski, Physical Review B **32**, 7384 (1985).

[120] A. T. Ogielski and I. Morgenstern, Physical review letters **54**, 928 (1985).

[121] N. Kawashima and A. Young, Physical Review B **53**, R484 (1996).

[122] J. De Almeida and D. J. Thouless, Journal of Physics A: Mathematical and General **11**, 983 (1978).

[123] G. Parisi, Physical Review Letters **50**, 1946 (1983).

[124] A. P. Young, Phys. Rev. Lett. **51**, 1206 (1983).

[125] C. M. Newman and D. Stein, Physical Review B **46**, 973 (1992).

[126] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing* (Cambridge University Press, New York, NY, USA, 1988), 1st ed., ISBN 0-521-35465-X.

[127] A. K. Hartmann, *A Practical Guide To Computer Simulation* (World Scientific Publishing Co., Inc., River Edge, NJ, USA, 2009), ISBN 9789812834157, 981283415X.

[128] J. Taylor, *An Introduction to Error Analysis: The Study of Uncertainties in Physical Measurements*, A series of books in physics (University Science Books, 1997), ISBN 9780935702750.

[129] B. Efron, The annals of Statistics pp. 1–26 (1979).

[130] The C++ Resources Network, *cplusplus.com* (2013), URL `http://www.cplusplus.com/`.

[131] J. W. Eaton, *GNU Octave* (2013), URL `http://www.gnu.org/software/octave/`.

[132] T. Williams and C. Kelley, *gnuplot homepage* (2015), URL `http://www.gnuplot.info/`.

[133] A. Shvets, G. Frey, and M. Pavlova, *Design Patterns* (2013), URL `http://sourcemaking.com/`.

[134] B. Dawes and D. Abrahams, *Boost C++ Libraries* (2013), URL `http://www.boost.org`.

[135] M. Galassi, J. Davies, J. Theiler, B. Gough, G. Jungman, M. Booth, and F. Rossi, *Gnu Scientific Library Reference Manual* (Network Theory Ltd., http://www.gnu.org/software/gsl, 2003), 3rd ed.

[136] D. Eisenstat, *dtree: dynamic trees a la carte (C++)* (2014), URL `http://http://www.davideisenstat.com/dtree/`.

[137] D. D. Sleator and R. E. Tarjan, Journal of the ACM (JACM) **32**, 652 (1985).

[138] The GDB Developers, *GDB: The GNU Project Debugger* (2013), URL http://www.gnu.org/software/gdb/.

[139] The Valgrind Developers, *Valgrind* (2013), URL http://valgrind.com.

[140] S. Mertens, Computing in Science & Engineering **4**, 31 (2002).

[141] S. A. Cook, in *Proceedings of the Third Annual ACM Symposium on Theory of Computing* (ACM, New York, NY, USA, 1971), STOC '71, pp. 151–158, URL http://doi.acm.org/10.1145/800157.805047.

[142] P. W. Shor, SIAM journal on computing **26**, 1484 (1997).

[143] A. A. Middleton, Physical review letters **88**, 017202 (2001).

[144] S. M. Sweeney and A. A. Middleton, Physical Review E **88**, 032129 (2013).

[145] E. W. Dijkstra, Numerische Mathematik **1**, 269 (1959).

[146] T. T. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms* (MIT Press, Cambridge, MA, USA, 1990), ISBN 0-262-03141-8.

[147] T. S. Jackson and N. Read, Phys. Rev. E **81**, 021130 (2010).

[148] M. Aizenman, IMA Vol. Math. Appl. **99**, 1 (1996).

[149] M. Aizenman, Nuclear Physics B **485**, 551 (1997), ISSN 0550-3213.

[150] J. van den Berg, Y. Peres, V. Sidoravicius, and M. E. Vares, Annales de l'Institut Henri PoincarÃľ - Probability and Statistics = Probabilities and Statistics **44**, 1173 (2008), ISSN 0246-0203.

[151] J. B. Kruskal, Proc. Am. Math. Soc. **7**, 48 (1956).

[152] A.-L. Barabási, Phys. Rev. Lett. **76**, 3750 (1996).

[153] G. Paul, R. M. Ziff, and H. E. Stanley, Phys. Rev. E **64**, 026115 (2001).

[154] P. Grassberger, Phys. Rev. E **67**, 036101 (2003).

[155] P. Duxbury and R. Dobrin, Physica A: Statistical Mechanics and its Applications **270**, 263 (1999), ISSN 0378-4371.

[156] G. E. P. Box and D. A. Pierce, Journal of the American Statistical Association **65**, pp. 1509 (1970), ISSN 01621459.

[157] A. Coniglio, Phys. Rev. Lett. **62**, 3054 (1989).

[158] S. N. Majumdar, Phys. Rev. Lett. **68**, 2329 (1992).

[159] M. Cieplak, A. Maritan, and J. R. Banavar, Phys. Rev. Lett. **76**, 3754 (1996).

[160] S. V. Buldyrev, S. Havlin, E. López, and H. E. Stanley, Phys. Rev. E **70**, 035102 (2004).

[161] A. P. Sheppard, M. A. Knackstedt, W. V. Pinczewski, and M. Sahimi, Journal of Physics A: Mathematical and General **32**, L521 (1999).

[162] B. Wieland and D. B. Wilson, Phys. Rev. E **68**, 056101 (2003).

[163] K. Dahmen and J. P. Sethna, Phys. Rev. Lett. **71**, 3222 (1993).

[164] P. Klein and S. Mozes, *optimization algorithms for planar graphs*, book draft, URL `http://http://planarity.org`.

[165] D. J. Frank and C. J. Lobb, Phys. Rev. B **37**, 302 (1988), URL http://link.aps.org/doi/10.1103/PhysRevB.37.302.

[166] G.-P. Zheng and M. Li, Phys. Rev. E **66**, 036108 (2002), URL http://link.aps.org/doi/10.1103/PhysRevE.66.036108.

[167] M. Kardar, J. Charmet, S. Roux, and E. Guyon, *Disorder and fracture* (1990).

[168] J. P. Bouchaud, E. Bouchaud, G. Lapasset, and J. Planès, Phys. Rev. Lett. **71**, 2240 (1993), URL http://link.aps.org/doi/10.1103/PhysRevLett.71.2240.

[169] T. Schultz, Rev Mod. Phys **36**, 856 (1964).

[170] F. Merz and J. T. Chalker, Phys. Rev. B **65**, 054425 (2002), URL http://link.aps.org/doi/10.1103/PhysRevB.65.054425.

[171] J. T. Chalker, N. Read, V. Kagalovsky, B. Horovitz, Y. Avishai, and A. W. W. Ludwig, Phys. Rev. B **65**, 012506 (2001), URL http://link.aps.org/doi/10.1103/PhysRevB.65.012506.

[172] J. Cong, A. B. Kahng, and K.-S. Leung, Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on **17**, 24 (1998).

[173] S. Peyer, D. Rautenbach, and J. Vygen, Journal of Discrete Algorithms **7**, 377 (2009).

[174] S. Koenig and M. Likhachev, in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on* (IEEE, 2002), vol. 1, pp. 968–975.

[175] H. Makse, S. Havlin, H. E. Stanley, and M. Schwartz, Chaos, Solitons & Fractals **6**, 295 (1995).

[176] H. A. Makse, S. Havlin, M. Schwartz, and H. E. Stanley, Physical Review E **53**, 5445 (1996).

[177] M. Aizenman, Communications in Mathematical Physics **73**, 83 (1980).

Biographical Data

NAME OF AUTHOR: Sean Michael Sweeney

PLACE OF BIRTH: Syracuse, NY

DATE OF BIRTH: May 12, 1988

DEGREES AWARDED:

High School Diploma, 2006

East Syracuse-Minoa High School, East Syracuse, NY

Bachelor of Science in Physics, 2010

Bachelor of Science in Math, 2010

Syracuse University, Syracuse, NY

HONORS AND AWARDS:

Archimedes Prize for Mathematics, May 2010

Neil F. Beardsley Prize for Physics, May 2010

Coronat Scholarship at Syracuse University, Fall 2006 to May 2010

NYS Scholarship for Academic Excellence, Fall 2006 to May 2010

Robert Byrd Scholarship, Fall 2006 to May 2010

George Wiley Award for Exceptional Performance in Organic Chemistry, 2007

Valedictorian of East Syracuse-Minoa High School, class of 2006