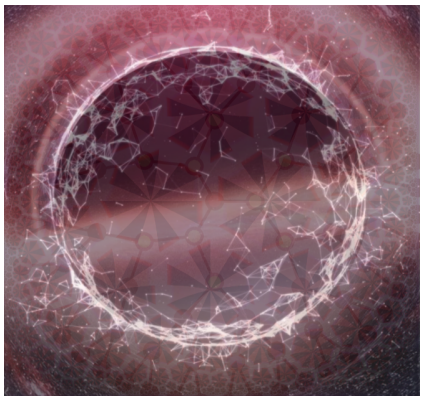
Creating a Holographic Statistical Mechanical-Network Model with the Ising Model and Tree Networks



THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE
in
PHYSICS

Author :	H.B.J.J. Salaris
Student ID :	s1440756
Supervisors :	K. Schalm
	D. Garlaschelli
2^{nd} corrector :	H. Schiessel

Leiden, The Netherlands, June 19, 2020

Creating a Holographic Statistical Mechanical-Network Model with the Ising Model and Tree Networks

H.B.J.J. Salaris

Huygens-Kamerlingh Onnes Laboratory, Leiden University
P.O. Box 9500, 2300 RA Leiden, The Netherlands

June 19, 2020

Abstract

Adopting some key ideas of the AdS/CFT correspondence, such as the geometrization of the RG formalism and having an AdS background spacetime, mappings of the 1D and 2D Ising model onto a network model were developed. The mappings primarily serve to engineer a 2D phase transition into a higher dimensional tree network and show what holographic properties are obtained by merely invoking some conceptual 'ingredients' from the holographic duality. The networks were studied by MC simulation of the Ising model and subsequent construction. This thesis then further reports on efforts to describe the network ensemble seeded off the Ising model independently, by a(n) (exponential) random graph model.

Contents

Chapter 1

Introduction

With the first direct 'real' image of a black hole, a little over a year ago [1], it has become more clear than ever, that what was once considered a mere mathematical curiosity of Einstein's theory of general relativity, is truly a part of nature. Of the many peculiar aspects of black holes, perhaps its most significant feature to physicists is the 'event horizon', the boundary surface that marks the point of no return for anything that falls in. In particular, the effort to reconcile the laws of thermodynamics with the presence of such a phenomenon has strongly impacted the attempts of formulating a theory of quantum gravity. In the 1970's, it was conjectured by Bekenstein [2] and supported by Hawking [3], from thermodynamic and quantum mechanical considerations, that black holes are thermodynamic objects with entropy that is proportional to the surface area of the event horizon. Moreover, black holes are considered maximum entropy objects. This implies that there is an upper bound on the entropy that a finite region of space can contain, and the upper bound is proportional to the *area* of the region.

This non-extensiveness of the entropy led 't Hooft to propose the 'holographic principle' publicly in 1993 [4], where he argued that any model of quantum gravity in a volume of space should reduce to a description by degrees of freedom in the lower dimensional boundary. Shortly after, it was worked out by Susskind how to create a string-theoretical realization [5]. Then, towards the end of that decade, arguably the most successful realization of the holographic principle was conjectured by Maldacena: 'the anti-de Sitter/conformal field theory (AdS/CFT) correspondence' [6], also known as the 'holographic duality', or 'gauge/gravity theory'. The AdS/CFT correspondence relates quantum gravity theories within the framework of string theory as dual to quantum field theories with conformal

symmetry. The gravity theory descibes the geometry of spacetimes of d dimensions that are asymptotically anti-de Sitter (AdS), while the conformal field theories (CFT) are defined on the boundary of these spacetimes, i.e. a spacetime with one dimension lower, $d-1$. Upon imposing certain symmetries on the conformal field theories, one can reduce the quantum gravity theory to that of classical general relativity. Hence, one has found a way to describe general relativity in terms of quantum field theory.

The AdS/CFT correspondence has been recognized as powerful mathematical machinery, finding applications beyond its original framework of string theoretical quantum gravity. Following the trend of a lot of interdisciplinary activity surrounding AdS/CFT, we have been inspired to adopt some of its core ideas to *create a network model with holographic properties*. Network theory is a young and active scientific discipline, whose popularity has been propelled by the fact that a wide range of both physical and societal systems can be described in terms of a network. That is, a collection of entities (nodes/vertices) and their connections (links/edges). Our aim is show that by copying some key characteristics from AdS/CFT into a network model, one can already find a relatively simple holographic statistical mechanical system. In particular, we have attempted to *encapture the 2D phase transition of the Ising model in the topology of a seemingly higher dimensional network model*. Before going over what this thesis reports and how it is structured, let us briefly discuss two parts of AdS/CFT that we have invoked to construct a network model.

The first is, the geometric representation of the renormalization group (RG) flow. Conceptually speaking, the main idea of the AdS/CFT correspondence is loosely the following. One has a field theory on 'the boundary' of an higher dimensional space, 'the bulk'. The field theory is renormalizable, i.e. it can be viewed from different (energy) scales. One can identify the ability of being able to 'zoom in' or 'zoom out' and observe the boundary system at a different scale as moving along the extra dimension of the bulk. More formally, the flow of the couplings in the boundary field theory, as prescribed by the renormalization group, corresponds to the radial coordinate of the bulk. So, essentially, the bulk consists of layers which can be considered to be copies of the boundary system at different scales. If one now alters the field theory, such that its RG flow is affected, e.g. changing a coupling locally, the correspondence prescribes that the geometry of the bulk space changes accordingly. One then finds a duality, if the change in geometry is alternatively described by the gravitational theory. Thus, analogous to a hologram, all information of the gravitational bulk is encoded in the theory of the boundary.

The second aspect of AdS/CFT that we translate into our networks,

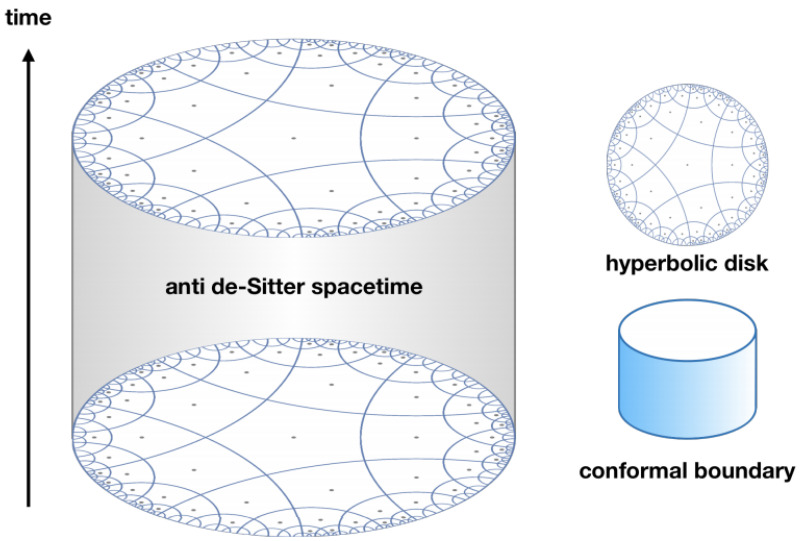


Figure 1.1: *Visualization of AdS_3 spacetime. Figure obtained from [7].*

is the hyperbolic nature of AdS spacetime. As mentioned earlier, in AdS/CFT, the geometry of the bulk is that of d-dimensional spacetimes that are asymptotically anti-de Sitter (AdS), i.e. they should be solutions to Einstein's equations with negative cosmological constant. The bare AdS spacetime is the maximally symmetric (simplest) vacuum solution. It corresponds to hyperbolic geometry, i.e. it is the generalization of the hyperboloid to spacetimes with Lorentzian signature. Hence, the restriction the Einstein equation imposes on the spacetimes can be seen as an hyperbolicity constraint. A visualization of AdS_3, so with dimension $d = 3$, is displayed in figure 1.1. In this picture the spatial geometry (at a given time) is represented by the hyperbolic disk. In the hyperbolic space of the disk each semi-circle is of the same size. So, in a sense, with this hyperbolic geometry, space itself is exponentially expanding as one moves towards the boundary. Note that the boundary cilinder has flat geometry and here it represents minkowski spacetime, in which the conformal field theory 'lives'.

With the above in mind, we designed and applied procedures related to coarse graining (real-space RG) on the Ising model, to generate networks. The Ising model plays the role of the boundary field theory, while the network is analogous to spacetime, where each configuration of spins results in a particular topology for the network. We translate the restriction of spacetimes having to be asymptotically AdS, to the requrement that all networks are trees (or very tree-like). Though there is no general precise

mathematical definition of a 'hyperbolic network', we have followed [8] in asserting that trees are the most hyperbolic examples of networks. Note that with hyperbolic spaces and trees, the boundary makes up a siginificant part of the whole, one can almost say that the volume of the space/-graph *is* the boundary. Therefore, it is interesting to see how much one gets, in terms of holography, from merely imposing hyperbolic geometry on the networks. Furthermore, the Ising phase transition was engineered into the network model, such that it loosely resembles the Hawking-Page phase transition [9].

The mapping of the Ising model to a network model serves as a starting point for creating a holographic network model. On its own, it does not signify anything very profound, as it remains nothing but the Ising model in disguise. It inherits the Ising thermodynamics trivially. Therefore, we have looked into ways of extending this Ising-network model, by first studying the networks seeded by the Ising model using a MC simulation and then weighting them with a network-specific measure. Chapter 3 describes the mappings and compares the thermodynamics of this semi-independent network model, to that of the Ising/boundary model. Alternatively, in chapter 4, we develop a way to create the network ensemble seeded off the Ising model independently. This is done by means of a(n) (exponential) random graph model.

Chapter 2 reviews a topic interesting in it of itself, namely the Ising model defined on the Cayley tree/Bethe lattice. Dependent on the order of when the thermodynamic limit is taken, this model exhibits completely different thermodynamics (hence the two names). The network is not dynamical here, and this chapter serves to briefly review the Ising model and familiarize the reader with the tree network.

Chapter 2

Example of a Statistical Mechanical System on a Hyperbolic Network: The Ising model on the Cayley Tree/Bethe Lattice

The Cayley tree, named in honor of Arthur Cayley who introduced the mathematical notion of a tree [10], is a a relatively simple example of a network that can be considered as a discretized version of a hyperbolic space. The study of the Ising model on the Cayley tree [11–14] showcases the importance of the boundary when working with hyperbolic objects. The model had gathered interest after it was discovered that it matters if one defines the Ising model on the finite Cayley tree and then takes the thermodynamic limit, or if one takes the Cayley tree to be infinite from the outset–a property that has much to do with the boundary. The graph resulting from the latter procedure has become known as the 'Bethe lattice' *, and it has been shown to be a lattice where the Bethe-Peiers approximation becomes exact [16, 17].

After giving a brief review of the Ising model, this chapter treats the Ising model on the Cayley tree as well as on the Bethe lattice.

*There is some ambiguity in older literature as to when one refers to a Cayley tree or a Bethe lattice, which has resulted in confusion to the present day. An attempt has been made relatively recently [15] to consolidate the nomenclature and clarify the distinction.

2.1 The Ising Model: A Brief Review

Originally devised as a model for magnetization in the early1920's [18], the Ising model has become known as the prototype model of statistical mechanics. Its prominence is due to the fact that despite its simplicity[†], it exhibits many interesting features of complex statistical mechanical systems (e.g. possible phase transition, cooperative and critical phenomena).

The model considers discrete variables that can be in one of two states +1 or -1, usually intepreted as the spin polarisation of an atom (spin up or spin down). If we have N of these spins σ_i, the particular set of values

$$\sigma = \{\sigma_1, \sigma_2, ..., \sigma_N\} \tag{2.1}$$

specifies a microstate of the system and one has 2^N states. The spins are arranged on a lattice, thus a microstate corresponds to a particular configuration of the lattice. Only nearest neighbour interactions and a coupling to an external field are incorporated in the energy of a configuration, as prescribed by the Hamiltonian:

$$E(\sigma) = -J \sum_{\langle i,j \rangle} \sigma_i \sigma_j - H \sum_i \sigma_i \tag{2.2}$$

where J is the coupling constant, $\langle i, j \rangle$ means the sum is over nearest neighbours of spin σ_i and H is an external magnetic field. We consider the ferromagnetic case where $J > 0$, i.e. it is energetically favourable for the spins to be aligned with their nearest neighbours.

The canonical probability of finding the system in a state σ is:

$$P(\sigma) = \frac{1}{Z} e^{-\beta E(\sigma)} \tag{2.3}$$

with $\beta = 1/k_b T$ and the normalizing factor is the partition function

$$Z = \sum_\sigma e^{-\beta E(\sigma)} \tag{2.4}$$

Then ensemble averages of observables $X(\sigma)$ of the system are:

$$\langle X \rangle = \frac{1}{Z} \sum_\sigma X(\sigma) e^{-\beta E(\sigma)} \tag{2.5}$$

[†]We mean that the Ising model is simple in the conceptual sense. Solving the model exactly can be a formidable task, depending on i.a. the dimension of the lattice on which it is defined. For example to date an exact solution for the Ising model defined on a 3-dimensional lattice has not been found.

by which one can calculate thermodynamic quantities statistical mechanically.

By the usual methods of statistical mechanics one can in principle obtain all equilibrium thermodynamic functions of interest (e.g. energy, specific heat, magnetization, susceptibility etc.) from the partition function. Therefore, for our purposes it will be sufficient to limit ourselves predominantly to the derivation of the partition function in the next sections.

2.2 Ising Model on the Cayley Tree

The Cayley tree is defined as a simple connected undirected (no self-loops, no isolated vertices and the edges are not directed) graph $G = (V, E)$, where V is the set of vertices (also called nodes, sites or points) and E is the set of edges (also called links, connections or bonds) that has no cycles (closed paths). One constructs it as follows: start with a root node 0 and connect q nodes to it. These q nodes constitute the first shell. Next connect each node of the first shell to $q-1$ new nodes, in this way constructing the second shell. Iterate this process to construct n shells. The result is a finite spherical tree like in figure 2.1, where each node has degree q (the degree of a node is its number of links to other nodes) except for at the boundary. At the boundary (i.e. the leaf vertices), located at shell n, the nodes have only degree one. Finally we define the Ising model on the Cayley tree by associating the spins with the nodes of the Cayley tree: $\sigma_i = \pm 1$, $i \in V$ (see figure 2.1).

From (2.2) and (2.4) the partition function reads:

$$Z = \sum_{\sigma} \exp\{K \sum_{\langle i,j \rangle} \sigma_i \sigma_j - h \sum_i \sigma_i\} \tag{2.6}$$

with $K = J/k_b T$ and $h = H/k_B T$. First let us consider the case $h = 0$. Then using a derivation adapted from [12, 15], we obtain a recursion relation for Z.

Consider the bond variables $\theta_\alpha = \sigma_r \sigma_s = \pm 1$, where σ_r and σ_s are spins sitting at the end of the bond. Note then that instead of specifying states by $\sigma = \{\sigma_i\}$, we can equally well specify them by the set $\{\sigma_0, \{\theta_\alpha\}\}$, as there is a unique correspondence between the two. If we divide the Cayley tree into $l = 1, 2, ..., n$ shells as shown in figure 2.2 and label the bond variables accordingly, we can write the partition function for a Cayley tree

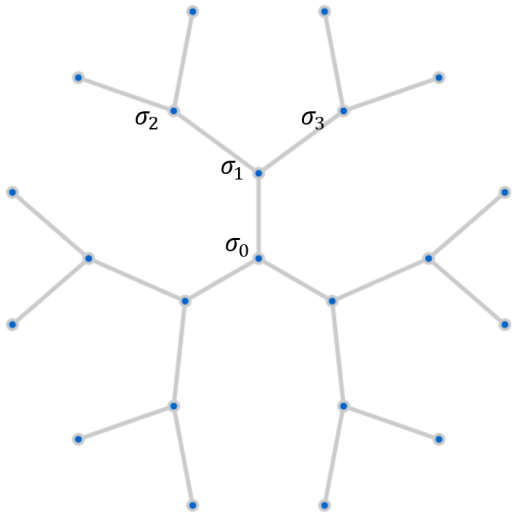


Figure 2.1: *A Cayley tree with degree $q = 3$ and having $n = 3$ shells. The root node 0 in the middle, is connected to q nodes. These q nodes in turn are in addition connected to $q - 1$ other nodes and the subsequent $q(q - 1)$ nodes are again in addition connected to $q - 1$ other nodes etc. The Ising model is defined on the lattice by associating the spins σ with the nodes.*

consisting of n shells as:

$$Z_n = \sum_{\{\sigma_0,\{\theta_\alpha\}\}} \exp\left\{K \sum_{l=1}^{n} \sum_{m=1}^{n_l} \theta_m^{(l)}\right\} \tag{2.7}$$

where n_l is the number of bonds or equivalently the number of nodes, belonging to shell l. Now note that each bond θ_α is independent as there are no closed loops in a Cayley tree. Hence the partition function factorizes nicely:

$$\begin{aligned} Z_n &= \sum_{\sigma_0} \left(\sum_{\theta_1^{(1)}} \sum_{\theta_2^{(1)}} \dots \sum_{\theta_{n_1}^{(1)}} \right) \dots \left(\sum_{\theta_1^{(n)}} \sum_{\theta_2^{(n)}} \dots \sum_{\theta_{n_n}^{(n)}} \right) \left(e^{K\theta_1^{(1)}} e^{K\theta_2^{(1)}} \dots e^{K\theta_{n_1}^{(1)}} \right) \dots \left(e^{K\theta_1^{(n)}} e^{K\theta_2^{(n)}} \dots e^{K\theta_{n_n}^{(n)}} \right) \\ &= \sum_{\sigma_0} \prod_{l=1}^{n} \prod_{m=1}^{n_l} \sum_{\theta_m^{(l)}=-1}^{1} e^{K\theta_m^{(l)}} \\ &= \left(\sum_{\sigma_0} \prod_{l=1}^{n-1} \prod_{m=1}^{n_l} \sum_{\theta_m^{(l)}=-1}^{1} e^{K\theta_m^{(l)}} \right) \times \left(\prod_{m=1}^{n_n} \sum_{\theta_m^{(n)}=-1}^{1} e^{K\theta_m^{(n)}} \right) \end{aligned} \tag{2.8}$$

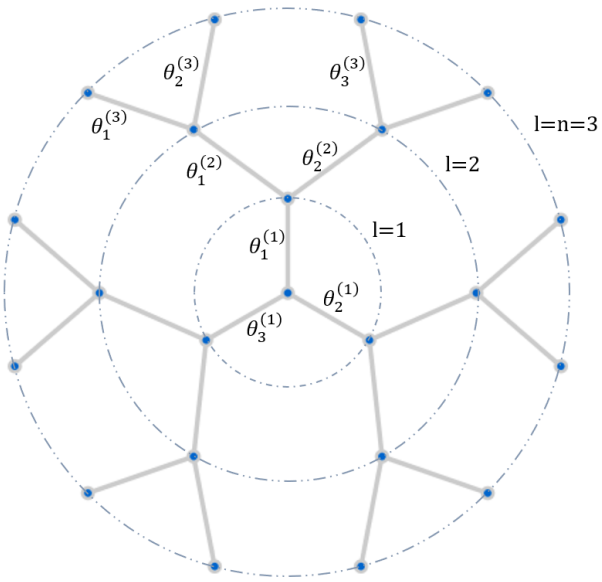


Figure 2.2: *A Cayley tree with $q = 3$ divided into $n = 3$ shells. The bond variables $\theta_m^{(l)} = \pm 1$ are labeled accordingly, i.e. shells are indexed by l and bonds (or equivalently nodes) in the shell are indexed by m.*

In the last line Z_n is factorized into the sum at the boundary (between the second pair of brackets) and the rest. Performing the sum over bonds at the boundary:

$$\prod_{m=1}^{n_n} \sum_{\theta_m^{(n)}=-1}^{1} e^{K\theta_m^{(n)}} = \prod_{m=1}^{n_n} 2\cosh(K) = [2\cosh(K)]^{n_n} \tag{2.9}$$

Note that the rest is just the partition function of the Cayley tree with $n-1$ shells:

$$Z_{n-1} = \sum_{\sigma_0} \prod_{l=1}^{n-1} \prod_{m=1}^{n_l} \sum_{\theta_m^{(l)}=-1}^{1} e^{K\theta_m^{(l)}} \tag{2.10}$$

So one finds the recursive equation:

$$Z_n = Z_{n-1} \left[2\cosh(K)\right]^{n_n} \tag{2.11}$$

and iteration gives

$$Z_n = [2\cosh(K)]^{n_n + n_{n-1} + \ldots + n_1} Z_0 \tag{2.12}$$

with $Z_0 = \sum_{\sigma_0} 1 = 2$. Now $n_n + n_{n-1} + ... + n_1$ sums up to the total number of bonds, which is equal to the number of edges $|E|$ of the graph. Furthermore for any finite tree one has for the total number of vertices $|V| = |E| + 1$. Hence

$$Z_n = 2\left[2\cosh(K)\right]^{|V|-1} \tag{2.13}$$

This partition function is identical to that of the one-dimensional Ising model. And the Ising chain is known to show no spontaneous magnetization, moreover in taking the thermodynamic limit one can check that the free energy per site $-\beta f = \lim_{|V|\to\infty} |V|^{-1} \ln(Z_n)$ is analytic in β. Hence we can conclude that for zero field the Ising model on the Cayley tree does not exhibit a phase transition and there is no spontaneous magnetization.

2.2.1 Alternative Method

Here we consider an alternative recursive relation for the partition function, which can also be used when $h \neq 0$ (only pertubatively) and in the treatment of the Bethe lattice. We follow the outline as prescribed in [13, 19].

Note that if one were to make a cut at the root of the Cayley tree and split σ_0, the result would be q rooted trees. That is, q disconnected identical pieces as shown in figure 2.3 for $q = 3$. This means that we can factor

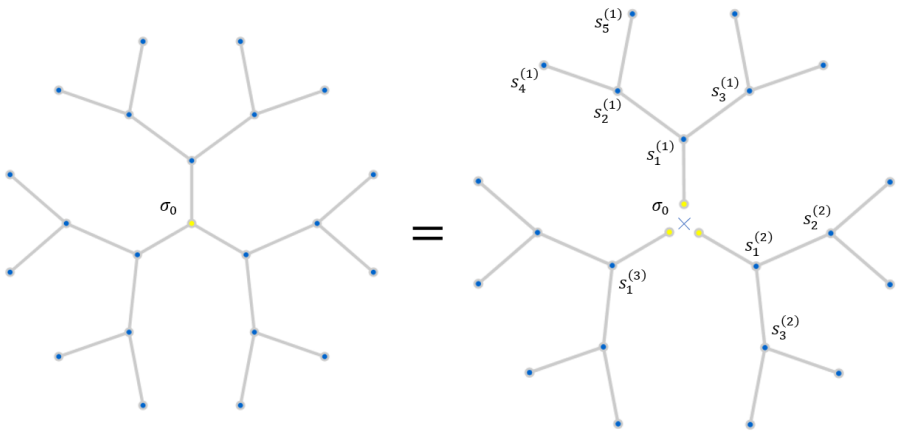


Figure 2.3: *The result of cutting the Cayley tree at its root. The Cayley tree is equivalent to the disconnected branches with their root nodes identified. Thus one can factorize the partition function into conditional partition functions of each branch.*

out the partition function (2.4) into conditional partition functions with respect to σ_0 of each branch. So,

$$Z = \sum_{\sigma_0} \exp\{h\sigma_0\} \prod_{m=1}^{q} Z_n^{(m)}(\sigma_0) \tag{2.14}$$

with

$$Z_n^{(m)}(\sigma_0) = \sum_{s^{(m)}} \exp\left\{ K\sigma_0 s_1^{(m)} + K \sum_{\langle i,j \rangle} s_i^{(m)} s_j^{(m)} - h \sum_i s_i^{(m)} \right\} \tag{2.15}$$

being the conditional partition function for the m-th branch. The set $\{s_i^{(m)}\}$ denotes the spins on the m-th branch (σ_0 not included) and the sums in (2.15) are defined appropiately. Now as the sum over all states on identical systems should give identical results, the partition function of each branch is the same. So we can lose the superscript and (2.14) becomes:

$$Z = \sum_{\sigma_0} \exp\{h\sigma_0\} \left[Z_n(\sigma_0) \right]^q \tag{2.16}$$

Next note that we can also factorize $Z_n(\sigma_0)$ in conditional partition functions of each branch, now starting from s_1:

$$\begin{aligned} Z_n(\sigma_0) &= \sum_{s} \exp\left\{ K\sigma_0 s_1 + K \sum_{\langle i,j \rangle} s_i s_j - h \sum_i s_i \right\} \\ &= \sum_{s_1} \exp\{K\sigma_0 s_1 + h s_1\} \left[\sum_{t} \exp\left\{ K s_1 t_1 + K \sum_{\langle i,j \rangle} t_i t_j - h \sum_i t_i \right\} \right]^{q-1} \end{aligned} \tag{2.17}$$

where the expression between square brackets is the conditional partition function with respect to s_1 and the set $\{t_i\}$ are the spins on a subbranch emanating from s_1. Then the full partition function (2.16) can be evaluated using the recursive equation for the conditional partition function:

$$Z_n(\sigma_0) = \sum_{s_1} \exp\{K\sigma_0 s_1 + h s_1\} \left[Z_{n-1}(s_1) \right]^{q-1} \tag{2.18}$$

Note the distinction between this recurrence relation and (2.11). With the latter, the partition function of the entire Cayley tree with n shells is

expressed in terms of the partition function of the Cayley tree with $n-1$ shells. On the other hand (2.18) is the result of decomposing a branch with n generations[‡] into branches with $n-1$ generations. Nonetheless both methods lead to the same result for $h = 0$. One has $Z_0 = 1$ and (2.18) yields

$$Z_n = [2\cosh(K)]^{\frac{1-(q-1)^n}{2-q}} \tag{2.19}$$

by which (2.16) becomes

$$Z = 2\,[2\cosh(K)]^{q\frac{1-(q-1)^n}{2-q}} = 2\,[2\cosh(K)]^{|E|} \tag{2.20}$$

That is, we retrieve again the Ising chain model (2.13), as we should.

For arbitrary $h \neq 0$, no closed form expression has been found and in the weak field limit one usually uses expansion or numerical methods. For instance one can expand (2.16) in h [13] and obtain some very unusual properties such as a diverging susceptibility without spontaneous magnetization [11, 13].

2.3 Ising Model on the Bethe Lattice

Essentially the Bethe lattice is the Cayley tree that goes on forever, i.e. it is infinite and there are no boundary nodes. It defined as an infinte cycle-free graph where each vertex is connected to the same number of neighbours. Thus one has a tree lattice with coordination number q, i.e. each node has degree q.

As the system is taken to be infinite from the start, we cannot peform a finite sum for the partition function. We can however use self-similarity to solve the system [15, 19]. First consider the partition function (2.16) and the recurrence relation (2.18) obtained for the finite Cayley tree. As in the Bethe lattice the branches corresponding to $Z_n(\sigma_0)$ and $Z_{n-1}(s_1)$ are infinite, one actually has:

$$Z_n(\sigma) = Z_{n-1}(\sigma) \tag{2.21}$$

which makes (2.18) a self-similarity equation for the conditional partition function $Z(\sigma)$. It turns out that it is more convenient- especially if one

[‡]One can still call them shells, but generations seems to be more appropiate as we are considering a branch and not the whole spherical Cayley tree. Ofcourse whatever you call it, the value of n is the same.

wants to study the magnetization, to first sum over $s_1 = \pm 1$ in (2.18)

$$Z_n(1) = e^{K+h}\left[Z_{n-1}(1)\right]^{q-1} + e^{-K-h}\left[Z_{n-1}(-1)\right]^{q-1} \tag{2.22}$$

$$Z_n(-1) = e^{-K+h}\left[Z_{n-1}(1)\right]^{q-1} + e^{K-h}\left[Z_{n-1}(-1)\right]^{q-1} \tag{2.23}$$

and then consider the ratio

$$x_n = Z_n(-1)/Z_n(1) = \frac{e^{-K+h} + e^{K-h}x_{n-1}^{q-1}}{e^{K+h} + e^{-K-h}x_{n-1}^{q-1}} \tag{2.24}$$

for which one has the self-similarity equation

$$x = y(x) \text{ with } y(x) = \frac{e^{-K+h} + e^{K-h}x^{q-1}}{e^{K+h} + e^{-K-h}x^{q-1}} \tag{2.25}$$

As $K > 0$, the function $y(x)$ increases monotonically from e^{-2K} to e^{2K} for $-\infty < x < \infty$. The solution to (2.25) can be found graphically by simultaneously plotting $y = x$ and $y = y(x)$. In doing so, two cases are found depending on the value of K (i.e. temperature T): one finds either one intersection point or three, as shown in figure 2.4. These two cases are in

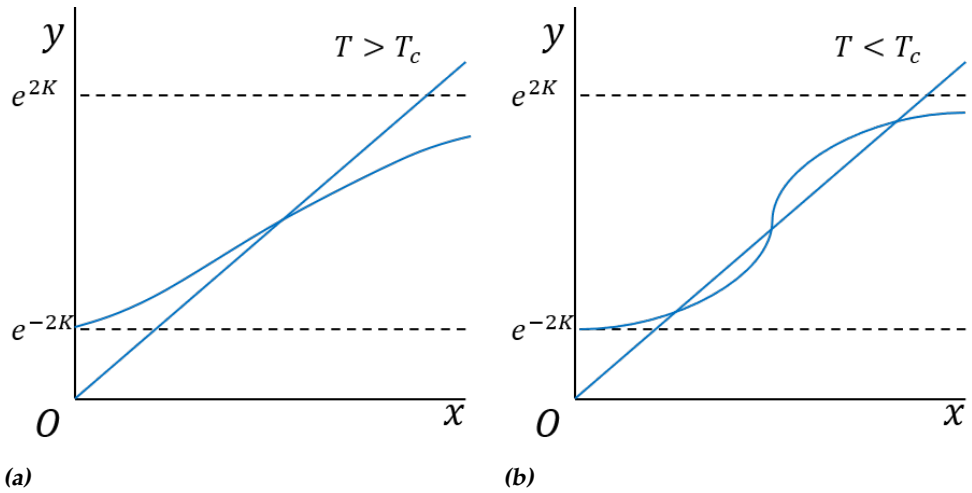


Figure 2.4: *Sketches of graphical solutions for (2.25). One finds either one solution (a) or three (b), corresponding to the paramagnetic phase and the ferromagnetic phase respectively.*

agreement with the typical behaviour of a ferromagnet (paramagnetic and ferromagnetic phase respectively). So the Ising model on the Bethe lattice exhibits a phase transition with spontaneous magnetization. By considering the variation of x (or equivalently the magnetization) with the external

field H the critical temperature at which the phase transition takes place can be found, see [19]. It is entirely dependent on the coordination number q and given by:

$$K_c = J/k_b T_c = \tfrac{1}{2} \ln (q/(q-2)) \tag{2.26}$$

2.3.1 Bethe lattice vs Cayley tree

The solution to the Ising model on the Bethe lattice is found to differ significantly from the model defined on the Cayley tree. As mentioned earlier, the Bethe lattice vs the Cayley tree is an example of a difference in limiting procedures: defining a model on a system that is infinite and then study its physical properties or considering the system to be finite, study its properties and then take the limit of large system size. Usually the latter procedure shows equivalence with the former, such as on the regular d-dimensional lattice $\mathbb{Z}^d$. The reason that this is not the case for the Cayley tree/Bethe lattice lies in the fact that the contribution from the boundary sites is non-negligible, even in the thermodynamic limit. The ratio of the number of boundary nodes n_n with respect to the total number of nodes is shown *not* to approach zero by a simple calculation. Note the number of nodes in shell l is given by $n_l = q(q-1)^{l-1}$ and the total number of nodes for a Cayley tree with n shells is:

$$N = 1 + \sum_{l=1}^{n} q(q-1)^{l-1} = \frac{2 - q(q-1)^n}{2-q} \tag{2.27}$$

Thus:

$$\lim_{N\to\infty} \frac{n_n}{N} = \lim_{n\to\infty} \frac{(2-q)q(q-1)^{n-1}}{2-q(q-1)^n} = \frac{q-2}{q-1} \neq 0 \tag{2.28}$$

for $q > 2$. Therefore it must come as no surprise that the Ising model on the Bethe lattice yields different results, as with the Bethe lattice there are no boundaries.

By another simple calculation one can find a heuristic argument as to why the Cayley tree does not exhibit spontaneous magnetization. From (2.26) one sees that $q = 2$ represents a threshold for the number of neighbours for which a phase transition can occur, i.e. spontaneous magnetization occurs when $q > 2$ in the Bethe lattice. One can also read this condition as that the average degree $\langle k \rangle$ should be larger than 2 (obiously for the Bethe lattice $\langle k \rangle = q$). The average degree for the Cayley tree is:

$$\langle k \rangle = 2|E|/|V| = 2 - 2/|V| \tag{2.29}$$

where the tree property $|V| = |E| + 1$ is used. Thus for the Cayley tree the average degree does not exceed the critical value of 2, irrespective of how large q- the degree of the nodes in the interior is. And so the Ising model defined on it has no phase transition (technically one could say a phase transition still occurs when $T = 0$).

2.3.2 Relation to the Bethe-Peierls approximation

Historically, the study of the Ising model on the Cayley tree began with the finding that the solution on the infinite Cayley tree was exactly the same as that of the Bethe-Peierls approximation [16, 17]. The authors Kurata et al. therefore coined the infinite Cayley tree (infinite from the outset) 'Bethe lattice', after Hans Bethe. The Bethe-Peierls approximation is a mean field approximation which incorporates first order interactions, first introduced by Bethe [20] and then applied to the Ising model by Peierls [21]. We give an outline of the approximation following [17, 22].

One starts by considering a cluster of a regular lattice with coordination number q. So a central spin σ_0 and its q surrounding neighbors, as shown in figure 2.5 for $q = 3$.

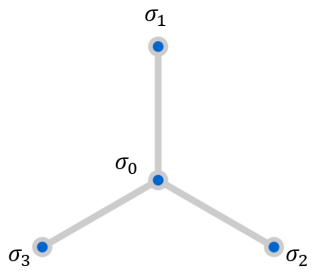


Figure 2.5: *Spin cluster for $q = 3$, consisting of a central spin σ_0 and its neighbours.*

The Hamiltonian of the cluster is written as:

$$E_c = -J\sigma_0 \sum_{i=1}^{q} \sigma_i - H\sigma_0 - H_e \sum_{i=1}^{q} \sigma_i \tag{2.30}$$

Apart from the usual interaction with nearest neighbors and an external field H, it approximates the interaction of the surounding spins with the rest of the lattice as a coupling to an effective field H_e. Then the partition

function of the cluster is straightforwardly calculated:

$$\begin{aligned} Z_c &= \sum_{\sigma} \exp\left\{ K\sigma_0 \sum_{i=1}^{q} \sigma_i + h\sigma_0 + h_e \sum_{i=1}^{q} \sigma_i \right\} \\ &= \sum_{\sigma_0}\sum_{\sigma_1}\ldots\sum_{\sigma_q} \exp\{h\sigma_0\} \prod_{i=1}^{q} \exp\{K\sigma_0\sigma_i + h_e\sigma_i\} \\ &= \sum_{\sigma_0} \exp\{h\sigma_0\} \left[2\cosh(K\sigma_0 + h_e)\right]^q \end{aligned} \tag{2.31}$$

Unless one prefers to work with hyperbolic trigonometric identities it is more convenient to make a change of variables

$$z = e^{-2K} \ , \ \mu = e^{-2h} \ , \ \mu_1 = e^{-2h_e} \tag{2.32}$$

which gives

$$Z_c = \mu^{-\frac{1}{2}} \left[z^{-\frac{1}{2}} \mu_1^{-\frac{1}{2}} + z^{\frac{1}{2}} \mu_1^{\frac{1}{2}} \right]^q + \mu^{\frac{1}{2}} \left[z^{\frac{1}{2}} \mu_1^{-\frac{1}{2}} + z^{-\frac{1}{2}} \mu_1^{\frac{1}{2}} \right]^q \tag{2.33}$$

The magnetization of the central spin can be found through the usual relation:

$$m_0 = \frac{\partial}{\partial h} \ln Z_c = -2\mu \frac{\partial}{\partial \mu} \ln Z_c \tag{2.34}$$

Similarly the magnetization of a neighbour spin can be obtained from:

$$m_{nb} = \frac{1}{q} \frac{\partial}{\partial h_e} \ln Z_c = -2 \frac{\mu_1}{q} \frac{\partial}{\partial \mu_1} \ln Z_c \tag{2.35}$$

Now the main assumption of the approximation is made; a self-consistency condition is introduced:

$$m_0 = m_{nb} = m \tag{2.36}$$

This gives the equation of state,

$$\frac{\mu}{\mu_1} = \left(\frac{\mu_1 + z}{1 + \mu_1 z} \right)^q \tag{2.37}$$

which can be solved graphically, similar to the self-similarity equation of section 2.3. In fact, if one defines $x^{q-1} = \mu_1/\mu$. And recalls (2.32), equation (2.37) becomes:

$$x = \frac{e^{-K+h} + e^{K-h} x^{q-1}}{e^{K+h} + e^{-K-h} x^{q-1}} \tag{2.38}$$

i.e. one retrieves exactly (2.25). Hence the Bethe-Peierls description is completely equivalent to the exact solution of the Ising model on the Bethe lattice.

One might wonder how an approximation can be exactly valid (it is an approximation right?). A rationale for this remarkable fact can be found by considering the *dimensionality* of the Bethe lattice. In general, dimensionality greatly influences the validity of a mean field theory (MFT). Usually MFT prescribes a replacement of all interactions to any given one body by an average or effective interaction. It follows that the more interactions are present the better results MFT will give, as fluctuations are averaged out. Thus MFT yields better and better results with increasing dimension, as the number of interactions automatically increases with dimension. In fact, the MFT can become exactly valid if one lets the dimension d go to infinity. This happens for example with the (Weiss) MFT of the Ising model on a regular d-dimensional cubic lattice.

Herein then lies an explanation for the equivalence of the Bethe-Peierls treatment to the exact solution. Consider the number of sites V_l within l steps of any given site on a flat d-dimensional regular lattice[§], and S_l the number of sites at step l of the same site (think of V_l and S_l as the volume and the surface area of a sphere with radius l respectively). V_l will be propotional to l^d, while S_l will be proportional to l^{d-1}. Hence:

$$S_l \propto V_l^{1-\frac{1}{d}} \tag{2.39}$$

In contrast, from (2.28) one sees that for the Bethe lattice the surface scales linearly with volume, $S_l \propto V_l$, which happens only in (2.39) when $d \to \infty$. Also, one can look at the dimensional dependence of the probability of finding loops in a regular lattice. With increasing dimension this probability decreases and loops become even irrelevant when $d \to \infty$, thus high dimensional regular lattices become like trees.

The above arguments indicate that the Bethe lattice is in some sense infinite-dimensional. That is, the number of neighboring sites and thus interactions surrounding any given site, increase with distance to that site like they would in an infinite dimensional regular lattice. From this perspective it is not surprising that the Bethe-Peierls MFT can be exact.

You might ask why all of the same arguments do not hold for the Cayley tree. The answer lies in the fact that the contribution from the boundary sites *is* taken into account with the Cayley tree. Obviously, the spatial dependence of the number of interactions surrounding a site at the

[§]By flat we mean that the d-dimensional lattice can tile the d-dimensional euclidean space in a regular way. With $d = 2$ one has the square, triangular and hexagonal lattice.

boundary is very different from that of a site in the interior. For example, the number of nearest-neighbours is one for a site at the boundary as opposed to $q \geq 3$, for a site in the interior. Hence the similarity to an infinite-dimensional regular lattice is lost.

Chapter 3

Mapping of the Ising model onto a Network model

This chapter treats our mappings of the Ising model onto a network model. In addition to the key ideas from AdS/CFT, we have used the Hawking-Page phase transition [9] as a leitmotiv in designing the procedure by which we construct a network from a configuration of spins. The Hawking-Page phase transition describes a transition between AdS spacetime and a spacetime geometry of a black hole that is asymptotically AdS. Both spacetime geometries are solutions to the (vacuum) Einstein equation with negative cosmological constant, which can be seen as a hyperbolicity constraint on the admitted solutions, and in the context of AdS/CFT they correspond to the low and high temperature phase of the boundary field theory. So, the black hole is interpreted as resulting from the disordered (high temperature) phase of the boundary field, whereas in the ordered (low temperature) phase of the boundary, there is no black hole but just the 'bare' AdS spacetime.

More concretely, the mappings were designed with the following in mind: the resulting network model should undergoe a phase transition, should be 'hyperbolic' regardless of the phase and should ultimately be described by a dual theory defined on its boundary. Moreover, with the aim of letting the networks loosely reflect the two phases of the Hawking-Page phase transition, we wanted to see the following property: starting with a pertubation at the boundary of the network (say a random walk), there is a significant increase in the time for it to reach the boundary again in the disordered phase as compared to the ordered phase. This was translated to having networks resembling trees in the 'AdS phase', while in the 'black hole phase' the networks are still tree-like, but have some added

structure (e.g. loops, or more nodes). The two phases of the network topology correspond to the low temperature and high temperature phase of the Ising model respectively.

The specific construction procedures by which we map a configuration of spins σ onto a network G_σ:

$$M : \sigma \to G_\sigma \tag{3.1}$$

developed with all of the above in mind, are layed out in detail in sections 3.2 and 3.3. But in short they can be summarized as follows: we define spins of the Ising model on the boundary of the to-be-constructed network. Then by a set of rules related to coarse graining, the bulk of the network is constructed. The procedure is designed such that the resulting network is (approximately) a simple fully connected tree with higher complexity for more disordered spin configurations. It should be mentioned that the mappings we present here in this thesis are not one-to-one, but two-to-one, due to inversion symmetry. That is, if one inverts all spins of configuration σ, the resulting configuration $-\sigma$, will be mapped to exactly the same graph: $G_\sigma = G_{-\sigma}$. This should not pose any significant problem as the entropy of the created network ensemble will only slightly differ from that of the Ising model (to be precise the difference is given by $\ln 2$, which is neglible for a large number of degrees of freedom).

The networks generated from the Ising model trivially inherit the Ising thermodynamics. Therefore, we look to extend the network model by weighting them with a network specific measure. In order to preserve the thermodynamics of the Ising model, we want the networks G_σ to be weighted by roughly the same weight as given in the Ising model ensemble:

$$P_I(\sigma) = \frac{1}{Z_I} e^{-\beta H_I(\sigma)} \tag{3.2}$$

where we consider the Hamiltonian that only incorporates nearest neighbour interactions

$$H_I(\sigma) = -J \sum_{\langle i,j \rangle} \sigma_i \sigma_j \tag{3.3}$$

(see section 2.1 for a brief review of the Ising model). Yet, to obtain a true network model that stand on its own, the weights we put on the networks should be determined from the network itself, without knowledge of the corresponding spin configuration. This amounts to finding a network Hamiltonian $H_{Nw}(G)$ that is (roughly) equal to the Ising energy, but

only makes reference to properties of the network. Additionally, we want H_{Nw} to be a function of some property of the whole network, i.e. the 'boundary' and the 'bulk'. We will see that it is easy to find a network attribute near the boundary that tracks the Ising energy. However, we would like to show that although H_{Nw} receives a contribution from the boundary as well as the bulk, the boundary contribution dominates due to the hyperbolicity of the networks.

In sum, we consider a network ensemble Ω that consists of the networks $\{G_\sigma\}$ weighted by the measure:

$$P_{Nw}(G_\sigma) = \frac{1}{Z_{Nw}} e^{-\beta H_{Nw}(G_\sigma)} \tag{3.4}$$

The network model's partition function is given by:

$$Z_{Nw} = \sum_{G} \delta_{G,G_\sigma} e^{-\beta H_{Nw}(G)} = \sum_{G_\sigma} e^{-\beta H_{Nw}(G_\sigma)} \tag{3.5}$$

Note that we have first written the partition function as a sum over all possible networks G. The kronecker delta reflects the fact that only networks resulting from our construction procedure are part of the ensemble. It can be seen as the *hyperbolicity constraint* of the model, as it restricts the networks of the ensemble to have hyperbolic structure. The kronecker delta also highlights the point that in addition to weighting networks with a network Hamiltonian, in order to have a fully fledged network model, one needs to provide a way of obtaining the networks of our ensemble $\{G_\sigma\}$ without knowledge of the Ising spins. In the next chapter we discuss a model that attempts to establish the ensemble independently of the Ising model.

While one could try to find an analytic correspondence between the Ising model and the network models resulting from our mappings, we have chosen to take a more straightforward route in simulating the Ising model and subsequently constructing the networks from the generated configurations of spins. The Ising model is simulated by means of a Monte Carlo (MC) method- a well-known numerical method used in computational physics to study statistical models. In particular, we employ the Metropolis [23] and Wolff algorithm [24]. From the generated networks G_σ (that occur in accordance with $P_I(\sigma)$), we try to identify an appropriate network Hamiltonian H_{Nw}. Then using the newly found H_{Nw}, we perform a separate simulation, where the networks occur in accordance with $P_{Nw}(G_\sigma)$. Note that this still amounts to generating configurations of spins, as we lack a direct way of generating the networks. Finally, we

obtain from simulation the free energy of the network model:

$$F_{Nw} = \frac{\ln Z_{Nw}}{\beta} \tag{3.6}$$

and compare it to that of the Ising model with the aim of showing that the two models exhibit the same thermodynamics. Our strategy is schematically displayed in figure 3.1.

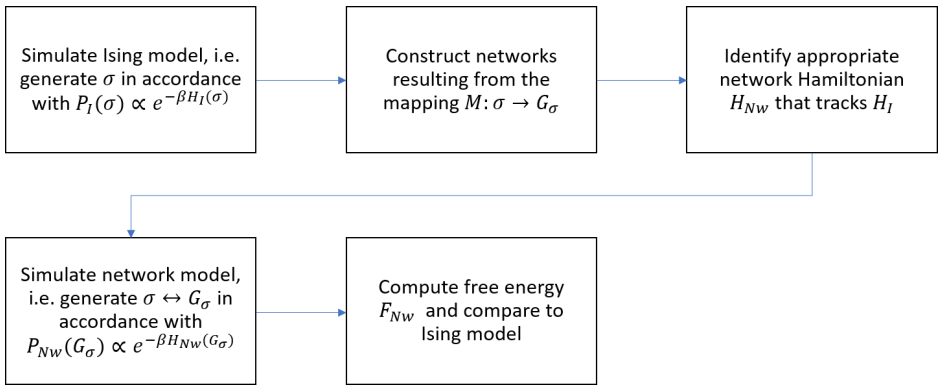


Figure 3.1: *Schematic displaying our strategy of showing that we have successfully mapped the thermodynamics of the Ising model onto a network model.*

We devote the first section of this chapter to the MC simulation of the Ising model and the calculation of thermodynamic quantities, such as the free energy. For the most part an outline is followed as given in [25] and [26]. Subsequent sections describe and discuss our mappings of the 1D and 2D Ising model onto a network model along with the results they yield.

3.1 Monte Carlo simulation of the Ising Model

The idea of simulating the Ising model in thermal equilibrium, is to sample configurations σ that occur in accordance with the Boltzmann distribution $P(\sigma)$ as given by (3.2). More generally, if one has a system consisting of an ensemble of states $\{X\}$ with probabability distribution $P(X)$, the goal of simulating the system is to sample states with a frequency that corresponds to that distribution. This allows for the calculation of ensemble averages of an observable A:

$$\langle A \rangle = \sum_X A(X)P(X) \tag{3.7}$$

which are often the quantities of interest.

Perhaps the most apparent method of simulation that comes to mind is to randomly generate states X (with uniform distribution) and accept them with probability $P(X)$ (or with a probability proportional to $P(X)$). Unfortunately, for the Ising model, or in general for systems where the target distribution is of the form of (3.2), this method is computationally very inefficient. The degeneracy $g(E)$ of a given energy E, i.e. the number of states that have that energy, increases drastically with energy. With increasing energy the acceptance probability becomes smaller and smaller. Thus, practically most of the computational effort is alloted to generating states that will be rejected anyway.

Therefore, one would like to generate a set of *representative states*, i.e. limiting the simulation to generating states that have a significant weight, while at the same time maintaining the right relative proportions in the generated frequencies (these should correspond to the desired distribution $P(X)$).

This delicate feature is accomplished by the Metropolis Monte Carlo method [23]. Hereby, as opposed to uniform sampling, states are generated by means of a Markov chain- a sequence of states where the probability for a given state is solely dependent on the previous one in the sequence. Essentially, one creates a system with artificial dynamics, whereby the probability for a state X to occur is now 'time-dependent': $P(X,t)$. With time t we obviously do not mean physical time, but the parameter that keeps track of the number of steps in the simulation.

Let $T(X \rightarrow X')$ denote the probability of having state X transitioning into X', then it is straightforward to find the equation governing the dynamics- the so-called master equation:

$$P(X,t+1) - P(X,t) = -\sum_{X'} T(X \rightarrow X')P(X,t) + \sum_{X'} T(X' \rightarrow X)P(X',t) \tag{3.8}$$

The idea is to find transition probabilities $T(X \rightarrow X')$ such that for large t the time-dependent probability: 1) equilibrates towards a stationary distribution, 2) the stationary distribution is the desired target distribution. Thus we have

$$\begin{aligned} &P(X,t+1) = P(X,t) \\ &P(X,t) \Rightarrow P(X) \end{aligned} \tag{3.9}$$

for $t \rightarrow \infty$, and (3.8) becomes

$$\sum_{X'} T(X \rightarrow X')P(X) = \sum_{X'} T(X' \rightarrow X)P(X') \tag{3.10}$$

For simplicity's sake, one often tries to find solutions that satisfy (3.10) term for term, that is:

$$T(X \to X')P(X) = T(X' \to X)P(X') \tag{3.11}$$

for every pair X, X'. This equation is called a *detailed balance* condition, and it simply says that the probability flux from state X to X' should be equal to that of X' to X.

In designing a practical implementation of the transition probability $T(X \to X')$, the detailed balance condition is usually one of the first things to check. Note that for the Ising model, one can write the detailed balance condition (3.11) as:

$$\frac{T(\sigma' \to \sigma)}{T(\sigma \to \sigma')} = \frac{P(\sigma)}{P(\sigma')} = e^{-\beta(E(\sigma)-E(\sigma'))} \tag{3.12}$$

Hence, only the ratio of the transition probabilities are fixed and it only depends on the energy difference between configurations before and after the transition $\Delta E = E(\sigma) - E(\sigma')$. This leaves quite some freedom in choosing a particular transition probability.

Another important property one usually requires for the simulation scheme to be valid, is referred to as *ergodicity*. The Markov chain is called ergodic if: 1) every state in the ensemble under consideration is attainable from any other state within a finite number of steps and 2) it is aperiodic. Ergodicity is needed in addition to the detailed balance condition, as one can think of transition probabilities that satisfy (3.11), but exclude states of the ensemble X apriori (e.g. $T(X \to X') = 0$ for all X, X'). We want to sample states in a biased way, but not totally exclude any state. Moreover, one wants the simulation to work regardless of what initial state is used.

In the traditional Metropolis algorithm applied to the Ising model, one uses a single spin flip updating scheme to implement transitions. A spin on the lattice is selected at random and the transition between the present configuration σ and the configuration with this spin flipped σ' is considered. If this transition lowers the energy, the trial state σ' is always accepted. If however, the energy inreases, σ' is accepted with a probability given by the boltzman factor that corresponds to the change in energy: $e^{-\beta(E(\sigma)-E(\sigma'))}$. In appendix A.1 we give a more detailed description of the Metropolis algorithm applied to the Ising model and show that it satisfies the detailed balance condition.

Though the Metropolis algorithm is suitable for our purposes, we prefer to use, where possible, another updating algorithm, the *Wolff algorithm* [24]. Here, instead of a single spin, a whole cluster of spins are updated in

parallel. The cluster is grown by starting off with a randomly selected spin and adding *aligned* nearest neighbours with probability $1 - e^{-2\beta J}$. Once no more spins are added to the cluster, all the spins in the cluster are flipped with probability 1. See appendix A.2 for a more detailed description of the Wolff algorithm and that it satisfies detailed balance. The main advantage of the Wolff algorithm over the traditional Metropolis- and other local update algorithms is that it does not suffer (or at least very weakly) from 'critical slowdown' in simulation of systems that undergo a phase transition. Critical slowdown refers to the phenomenon that when the system is critical, the autocorrelation time diverges and one needs to generate a lot more configurations to obtain reliable results. In short, at criticality a very long simulation time is needed. Therefore, unless explicitly stated otherwise we use the Wolff algorithm. Unfortunately, the Wolff algorithm is specifically designed for Ising- and other spin models where the Hamiltonian is given by nearest-neighbour interactions. It is not (easily) applicable to models with a different hamiltonian. Hence, in simulations where we go beyond the Ising model we use the much more widely applicable Metropolis algorithm.

We have then a simulation of the Ising model in equilibrium by means of a stochastic trajectory through phase space. Hence, in observing the generated sequence we can approximate ensemble averages (3.7) through time averages:

$$\bar{A} = \frac{1}{m} \sum_{t=1}^{m} A(\sigma(t)) \tag{3.13}$$

By m we will denote the number of sampled configurations; with the metropolis algorithm the 'time' t is expressed in units of Monte Carlo steps per spin (MCS), being equal to N trials for a system with N spins. When the Wolff algorithm is used, m will be given by the number of moves (cluster updates).

To get an indication of the error in these time averages we can calculate the standard deviation:

$$\bar{\sigma} = \sqrt{\bar{A}^2 - \bar{A}^2} \tag{3.14}$$

However, for this to represent a true statistical error, it should be obtained from samples that are *independent*. Obviously, our Markov chain simulation generates *correlated* configurations, so we need to correct for this. There are different ways of calculating the statistical error from correlated data. We have chosen a method that involves the calculation of the autocorrelation time τ, a quantitiy that gives an indication of how many simulation steps configurations are correlated with one another. We refer to

appendix B for a full description of our calculation of the correlation time and error analysis. In short, the standard deviation obtained from correlated data can be related to the standard deviation for the uncorrelated case via [27, 28]:

$$\sigma = \bar{\sigma}\sqrt{1 + 2\tau} \tag{3.15}$$

Finally, note that as the simulation of the Ising model is obviously always performed on a finite lattice, observables calculated from (3.13) will suffer from a finite size effect. That is, they differ from the ensemble averages of the Ising model in the thermodynamic limit $N \to \infty$, which is the macroscopic model we are interested in. The problem is somewhat alleviated by implementing periodic boundary conditions, which we do. However, to make real qualitative statements on anything related to the macroscopic Ising model we employ finite size scaling. The results are obtained for varying system sizes. If a trend is observed with increasing scale, one can extrapolate the result or statement to the macroscopic model.

3.1.1 Free Energy Calculation

In addition to ensemble averages, we would like to obtain the free energy from simulation. The free energy, related to the partition function by

$$F = -\frac{\ln Z}{\beta} \tag{3.16}$$

essentially encodes all the relevant information of the system in the canonical ensemble. Hence, we can claim that we have successfully mapped the Ising model onto a network model if we can show that the free energy of the latter converges to the former. Moreover, in combination with the average energy obtained from (3.13), we can calculate the entropy S from the standard thermodynamic relation:

$$F = \langle E \rangle - TS \tag{3.17}$$

Calculation of the free energy is a difficult assignment for conventional MC methods. The difficulty lies in the fact that a quantity like the free energy cannot be formulated as an ensemble average of a function of the degrees of freedom (in contrast to e.g. the energy of the system). Rather,

it is a phase space integral. And although a MC simulation samples the dominant contribution of phase space to the free energy, it begs the question if it provides a good estimate of the full phase space volume integral. (See [29] for a nice analogy of trying to estimate the surface area of a river by measuring the average depth.) Even more troublesome, the simulation samples states with a probability proportional to the target distribution, where the proportionality factor is unknown. This is not an issue for the calculation of ensemble averages, as it cancels out. However for the calculation of the partition function (free energy), this factor is required.

It is worth mentioning that ofcourse there exists methods to compute the free energy of the Ising model which do not involve MC simulation. Most notably the transfer matrix method, whether it be analytical or numerical. However, the point is that we would like a method that can be incorporated in our MC simulation and subsequently also be applied to the network model. Therefore, a free energy calculation method like the numerical transfer matrix method is not suitable for our purposes, as it is restricted to lattice spin models.

Fortunately, what *is* feasible with MC simulations- is to compute free energy differences. That is, if one knows the free energy for particular values of the system parameters (e.g. at a given temperature), one can obtain the difference to the free energy at different values for those system parameters (e.g. at a different temperature). A large variety of methods have been developed for this purpose, to name a few: expressing the difference as an ensemble average [30–32], histogram analysis methods [33–36], entropic sampling [37], Wang-Landau method [38] or straightforward thermodynamic integration [25, 29]. The entropic sampling and Wang-Landau method require a simulation in energy space as opposed to configuration space and thus cannot be incorporated into our MC simulation. Probably, the methods that can be implemented most easily are a histogram method or thermodynamic integration. We choose to use a histogram method, developed relatively recently by Sheng Bi and Ning-Hua Tong [39]. Their method has been implemented in combination with a standard MC simulation of the Ising model and the results show it to be very accurate. We describe the method in what follows.

As the configuration probability for the Ising model is given by (3.2), the energy probability distribution is:

$$P(E) = \frac{g(E)}{Z} e^{-\beta E} \tag{3.18}$$

where $g(E)$ is the degeneracy of the energy level E. With our MC simulation we can then estimate $P(E)$ by making a histogram of the encounterd

energies:

$$p(E) \approx \frac{N(E)}{m} \tag{3.19}$$

with $N(E)$ being the number of configurations with energy E encountered during simulation. m is the total number of sampled configurations (simulation steps) and obviously the accuracy of the estimation increases with increasing m. The simulation is done for a specific temperature, so $\beta = 1/k_bT$ is an input parameter. Then rewriting (3.18), we see that if the degeneracy $g(E)$ is known, we can estimate the free energy from simulation:

$$\begin{aligned} F(T) &= -\frac{\ln Z}{\beta} = -\frac{1}{\beta} \ln \left(g(E) e^{-\beta E} \frac{m}{N(E)} \right) \\ &= E - k_bT \ln \left(g(E) \frac{m}{N(E)} \right) \end{aligned} \tag{3.20}$$

Usually, the groundstate degeneracy is known; for the Ising model one has $g(E_g) = 2$. So theoretically speaking, the free energy can be calculated for arbitrary temperature directly from the groundstate histogram $N(E_g)$.

However, this does not work in practice. For any given T, the energy distribution is sharply peaked at an energy $E(T)$. The peak position moves away from the groundstate energy E_g towards higher energy levels for increasing temperature. And for finite m one cannot accurately sample $P(E_g)$, as it becomes vanishingly small. Figure 3.2 illustrates this. It shows $P(E) \approx N(E)/m$ for different temperatures obtained from our simulation of the Ising model on a $L \times L = 20 \times 20$ square lattice. For high temperatures $P(E_g)$ becomes zero and so $F(T)$ cannot be calculated from $N(E_g)$.

Therefore, [39] prescribes a scheme to calculate unkown values of $g(E)$ from known ones. Suppose one knows the degeneracy at energy level E_1 and one wants to know the degeneracy at energy E_2. The probabilities $P(E_1)$ and $P(E_2)$ for these energy values to occur are given by (3.18) and at the same temperature the partition function $Z = Z(T)$ is the same. Using this and plugging in (3.19) we find:

$$\begin{aligned} g(E_2) &= g(E_1) \frac{P(E_2)}{P(E_1)} e^{-\beta(E_1 - E_2)} \\ &\approx g(E_1) \frac{N(E_2)}{N(E_1)} e^{-\beta(E_1 - E_2)} \end{aligned} \tag{3.21}$$

Thus if we obtain $N(E_1)$ and $N(E_2)$ from simulation, we can calculate $g(E_2)$ from $g(E_1)$. The only requirement is that $N(E_1)$ and $N(E_2)$ are

(a) **(b)**

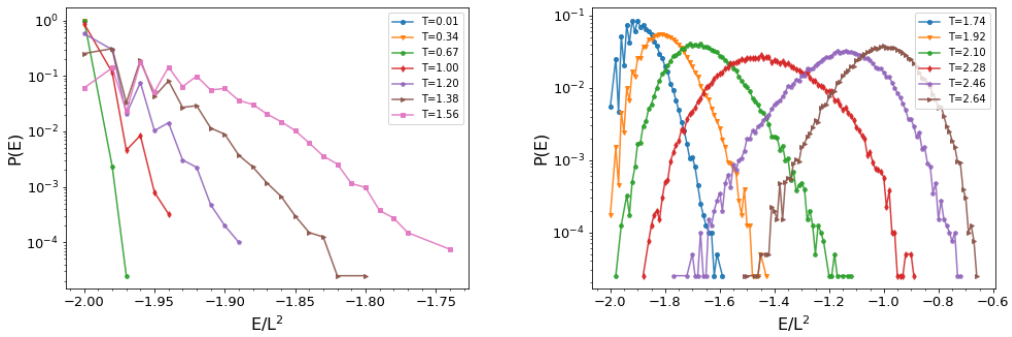


Figure 3.2: *Energy probability distribution $P(E) \approx N(E)/m$ for various temperatures plotted on a logarithmic scale. These are obtained from simulation of the Ising model on a $L \times L = 20 \times 20$ square lattice with number of samples $m = 4 \times 10^4$. The energy is given by (3.3) and the temperature is expressed in reduced units of J/k_b. (a) For low temperatures $P(E)$ is peaked around the ground-state energy ($E_g/L^2 = -2.0$). (b) For higher temperatures the peak is shifted towards higher energies and $P(E_g)$ is zero.*

nonzero. This means that the simulation should be done at a temperature for which it produces a histogram where $N(E)$ is significantly large for E_1 and E_2. More generally, the simulation at temperature T produces an energy window W_T, where $N(E)$ is significantly nonzero. Then by (3.21), knowledge of $g(E)$ can be transferred to all $E \in W_T$.

The free energy $F(T)$ is then obtained by a relay-like scheme. Knowing the degeneracy for a particular energy E_i, we do a MC simulation at a temperature T_i, such that we produce a histogram for which $E_i \in W_{T_i}$. Thus with $g(E_i)$ known and $N(E_i)$ obtained, we can calculate $F(T_i)$ by (3.20). Next we do a simulation at temperature T_{i+1} producing a histogram with energy window $W_{T_{i+1}}$. Temperature T_{i+1} is chosen such that $W_{T_{i+1}}$ and W_{T_i} overlap, i.e. T_{i+1} is close to T_i. We choose an energy $E_{i+1} \in W_{T_i} \cap W_{T_{i+1}}$ and use the data of the T_i simulation to obtain $g(E_{i+1})$ from $g(E_i)$ by (3.21). As $g(E_{i+1})$ is temperature independent, we can use it in combination with $N(E_{i+1})$- obtained from the T_{i+1} simulation, to calculate $F(T_{i+1})$. In this way, starting with T near zero where we know $g(E_g)$, and incrementing the temperature appropiately, $g(E)$ and $F(T)$ are found for consecutively higher E and T respectively.

We follow [39] in choosing the common energy value $E_{i+1} \in W_{T_i} \cap W_{T_{i+1}}$ to be the crossing energy E_c, given by the intersection $P(E_c, T_i) = P(E_c, T_{i+1})$, as shown in figure 3.3. Then in order to guarantee $P(E_c)$ is not too small,

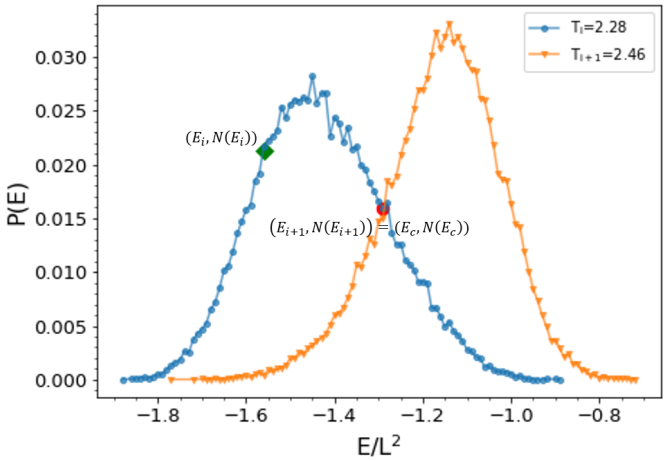


Figure 3.3: *Energy histograms for two adjacent temperatures T_i and T_{i+1}, illustrating the procedure to obtain $F(T)$. Suppose we know $g(E_i)$, then with $N(E_i)$ we can calculate $F(T_i)$ using (3.20). For E_{i+1} we choose the crossing energy E_c, as this is where $N(E)$ is largest for both histograms. Then the knowledge of $g(E_i)$ can be transferred to $g(E_{i+1})$ by (3.21). With $g(E_{I+1})$ known, $F(T_{i+1})$ can be calculated. The histograms were obtained from simulation of the Ising model on a $L \times L = 20 \times 20$ square lattice with number of samples $m = 4 \times 10^4$.*

one needs to choose an increment interval $\Delta T = T_{i+1} - T_i$ that is not too large. Bi and Tong found the optimal value to lie within $0.06 - 0.1$ for different values of m in their simulations. They tested their method on a version of the Ising model (two-state Potts model) where the phase transition occurs at $T = 1.181$, while we consider a model where the phase transition occurs at $T_c = 2.269$. Hence, conversion by this factor $2.269/1.181$ gives the optimal range relevant for our simulation: $\Delta T = 0.12 - 0.19$. From our simulations, we find that we have to use a larger ΔT at the high end of the temperature spectrum. The problem is that for high temperatures we find the energy histograms of two adjacent temperatures ΔT to practically be the same. Perhaps this is due to the fact that in general we simulate on smaller lattices than used in [39]. In the end we decided to use a varying temperature mesh for the calculation of $F(T)$, where ΔT is increased for higher temperatures.

This thesis lacks a full error analysis of the free energy calculation. Bi and Tong have given an estimate for the error in their method by repeatedly calculating F (400 times). From the independent data they calculated the standard deviation and compared the average to the exact value. We

have chosen not to perform such an analysis, as it proved to be rather costly. Our MC simulation typically samples the system $m = 1 \times 10^4$ times for each temperature, and then this would have to be carried out several hundreds of times, in the worst case amounting to a simulation time of several months with our simulation apparatus. To still give some indication of the error we can calculate the difference between F and the exact value of the Ising model per spin:

$$\epsilon = \frac{|F - F_{Exact}|}{N} \tag{3.22}$$

Bi and Tong found their error to be small in the ordered phase and increase linearly with temperature in the disordered phase. We expect that the quantity ϵ will behave similarly. Obviously, ϵ can only be seen as a direct indication of the error for the numerically found free energy of the Ising model $F = F_I$. However, we can use $\epsilon_I = |F_I - F_{Exact}|/N$ as a benchmark for the accuracy of the free energy calculation method. And if ϵ_I is small, we can be confident that the numerically calculated network free energy F_{Nw} will also be close to its true value.

3.2 First Mapping Scheme: Averaging over Spin Domains

Here we shall present our first mapping scheme, which we coin 'Averaging over Spin Domains' (AoSD). Before going over to the 2D Ising model, we start with the mapping of the 1D Ising model onto a network model. Though it does not feature a phase transition- the property we would like to see reflected in our network model, the added benefit is that it is easier to picture and thus easier to illustrate our mapping scheme. In addition, it can be used to verify our results obtained in the high temperature phase of the 2D Ising model, as we expect these to be very similar.

3.2.1 AoSD in 1D

Construction Procedure

A MC simulation as described in the previous section is done, generating configurations of N spins on a 1D lattice of size L (so $N = L$). We impose periodic boundary conditions, so the lattice has the topology of a chain. The input parameter of the simulation is the temperature $T = 1/\beta$ (in reduced units of $[J/k_B]$), by which the probability for each configuration to

occur is controlled.

Let us now go over the procedure by which we construct a network G from a particular configuration $\sigma = \{..,\sigma_i,..\}$. See figure 3.4 for an accompanying schematic. Firstly, we regard the lattice sites of the Ising model as the boundary nodes of the to-be-constructed network. Next, for each domain of equal spins in the configuration a new node is added and all the sites in the domain are connected to the new node by edges. These new nodes constitute now the network's next to outer shell- let us label the shells of the network starting from the outer shell, so the outer shell is labelled by $l = 0$ and the next to outer shell by $l = 1$, etc. Spin values are assigned to each node in the $l = 1$ shell, given by the sum of spin values in the domain they are connected to. This bundling of domain sites constitutes step 1 of the construction procedure. Next comes step 2, which creates the $l = 2$ shell. The nodes in the $l = 1$ shell are coarse grained

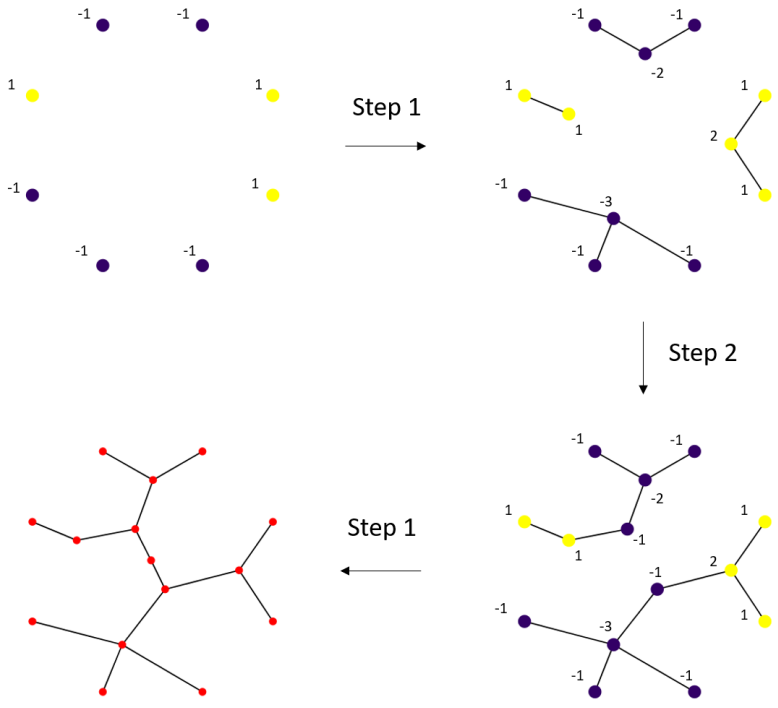


Figure 3.4: *Different stages of the 'Averaging over Spin Domains' procedure. Starting from the configuration of spins in the 1D Ising model, defined here on a lattice with size $L = 8$, the network is constructed by alternately applying step 1 (bundling of domains) and 2 (coarse graining).*

with a branching factor of two, so two neighbouring nodes in the $l = 1$ shell are connected to a new node. By construction, a node j in the newly created $l = 2$ shell is connected to two nodes of the $l = 1$ shell, let us call them k_1 and k_2, which have spin values that have opposing signs, let us refer to these as $\sigma_{k_1}^{(1)}$ and $\sigma_{k_2}^{(1)}$. Node j receives a spin value of $\sigma_j^{(2)} = \pm 1$, where the sign is determined by whichever spin value between k_1 and k_2 is larger in magnitude (hence by majority). So, if $|\sigma_{k_1}^{(1)}| > |\sigma_{k_2}^{(1)}|$, j inherits the sign of $\sigma_{k_1}^{(1)}$ and vice versa. If the spin values of k_1 and k_2 are equal in magnitude, j receives a spin value of $+1$ or -1 *randomly*. This concludes step 2, where we have effectively averaged over spin domains. Step 1 and 2 are then repeated alternately untill a new shell is created that contains only one node. This last node constitutes the root node of the created tree network and finalizes the construction procedure.

The cautious reader will have noticed that by assigning the spin value randomly in the case of a tie, our mapping is no longer two-to-one. In appendix C the effect of this 'added randomness' is explored. In short, it is shown that although the frequency of ties occurring is considerable, the added randomness is subleading and the effect of the random tie breaker on the obtained results is insignificant.

In programming the AoSD procedure, the nontrivial part is to scan and decompose the lattice of spins into its domains. We have adopted and modified a recursive routine from [25], originally designed to implement the Swendsen-Wang algorithm [40] (a cluster MC algorithm similar to the Wolff algorithm). See appendix A.3 for the routine.

The AoSD construction procedure is practically implemented in a MC simulation of the 1D Ising model via Python in a Jupyter Notebook. We refer to A.4.1 for the code.

Results

Through the above mapping we generated networks from the MC simulation of the Ising model. Figure 3.5 shows some representative configurations of spins along with their corresponding networks, generated by the simulation for different temperatures $T[J/k_B]$. For T close to zero one finds the groundstate of the network model to be a star graph. With increasing temperature the network becomes an ever deeper rooted tree with more and more nodes and edges.

Observing the generated networks, our first and simple guess for the network hamiltonian that tracks the Ising model energy E_I (3.3) is the number of nodes n. The average of both quantites obtained from sim-

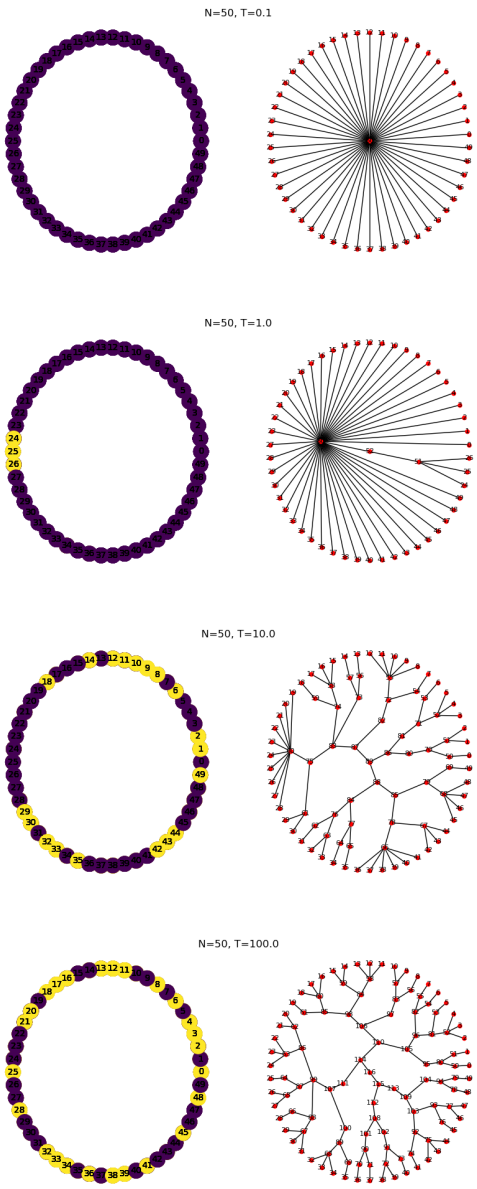
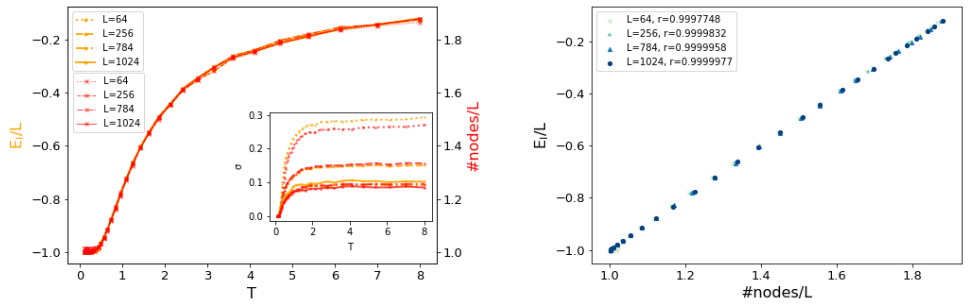


Figure 3.5: *Configurations of the 1D Ising model, along with the corresponding networks that were construced by the AoSD mapping procedure. The configurations/networks are representative for the ones generated by the simulation at a given temperature $T[J/k_b]$. The spins/nodes are labeled by numbers and the node colors (purple/yellow) indicate the spin value.*

ulation at different T and for various system sizes L, are shown in the same plot of figure 3.6a. Both were calculated as given by (3.13), with



(a) *(b)*

Figure 3.6: *Comparison between the Ising energy E_I and number of nodes, obtained from the MC simulation of the 1D Ising model and the correspondingly constructed networks. The simulation was done for increasing system size L and both quantities are averages over $m = 2.5 \times 10^4$ samples for each temperature $T[J/K_B]$. (a) Ising energy (yellow, left vertical axis) and the number of nodes (red, right vertical axis) are shown as a function of T. The inset shows the corresponding standard deviation σ. (b) The Ising energy plotted against the number of nodes. In the legend the Pearson correlation coefficient r between the two quantities is shown.*

$m = 2.5 \times 10^4$ samples for each T. The inset shows the corresponding standard deviation (3.15), where we see that the fluctuation in the mean decreases with increasing scale. In figure 3.6b the same datapoints for the Ising energy and the number of nodes are shown, but now plotted against each other. Both figures show that the variation of the Ising energy and the number of nodes, as a function of temperature, practically overlap for all system sizes. In the legend of figure 3.6b, we have included the Pearson correlation coefficient $r \in [-1, 1]$ between the two datasets, giving a more quantitative measure to the overlap. For all system sizes r is found to be close to 1, indicating a strong positive linear correlation.

Having established that the number of nodes and the Ising energy are linearly related, we define the network Hamiltonian as follows:

$$H_{Nw} = J(-2L + n) \tag{3.23}$$

where we include J the coupling constant, so that the model has the same units of energy as the Ising model. Note that the number of boundary nodes is fixed and in 1D equal to L, so $n = L + n_{int}$, where n_{int} are the internal nodes. It is then perhaps more instructive to write H_{Nw} in terms of n_{int}:

$$H_{Nw} = J(-L + n_{int}) \tag{3.24}$$

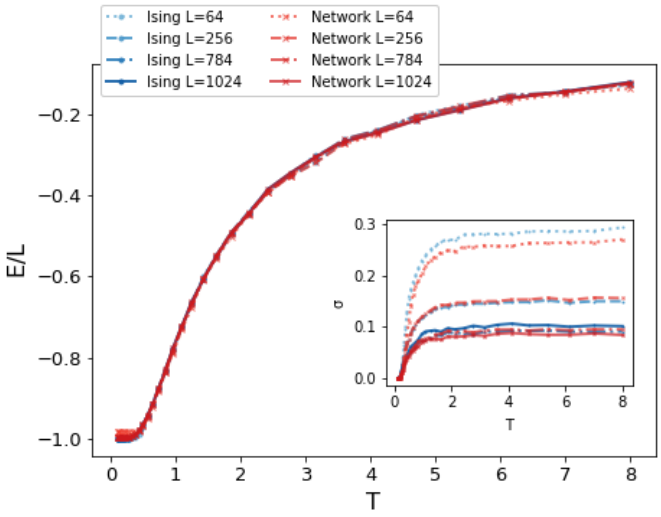


Figure 3.7: *The network energy (red), as defined by (3.23), and the 1D Ising energy (blue) are shown as a function of temperature $T[J/k_B]$. Both are averages over $m = 2.5 \times 10^4$ samples for each T, obtained from MC simulations performed for various system sizes L. The inset shows the corresponding standard deviations.*

as n_{int} is the quantity that varies. The offset $-JL$ is added such that H_{Nw} is numerically (approximately) equal to the Ising energy. Irreverently said, the Ising energy is nothing but a counter for the number of nearest neighbour pairs of misaligned spins Z, with an offset given by the total number of nearest neighbour pairs $N_{n.n.}$. In 1D (with periodic boundary conditions), $N_{n.n.} = L$. Hence, $H_I = -JL + JZ$. Though a difference in energy by a constant should not alter the thermodynamcs, it is convenient to compare the energy and ultimately the free energy on the same scale. Finally, note that one could also use the number of edges e instead of n to define H_{Nw}. This amounts to having essentially the same function, as the network ensemble consists solely of trees, and for a tree graph the number of nodes is practically equal to the number of edges: $n = e + 1$.

Figure 3.7 shows the average network energy as defined by (3.23) and the Ising energy as functions of temperature. The same data used to create figure 3.6a was used to create this plot. As the two energy functions are practically equal, we are confident that our network model weighted by H_{Nw} will exhibit the same thermodynamics as that of the Ising model.

In order to show this explicitly, we performed a *separate* MC simulation where H_{Nw} was used instead of H_I, i.e. the networks are simulated with the target distribution being (3.4). The use of H_{Nw} prevented us from

using the Wolff algorithm (see appendix A.2) and the network simulation was performed using the more widely applicable Metropolis algorithm. In figure 3.8 the free energy of the network model is shown vs T for various system sizes. It was obtained through the method described in subsection 3.1.1, where we performed an upward temperature scan starting from $T_0 = 0.1$ and the groundstate degeneracy $g(E_g) = g(E_{T_0}) = 1$. At each temperature T_i an energy histogram was produced from $m = 2 \times 10^4$ samples. It is compared to the free energy of the 1D Ising model, obtained numerically from the Ising model MC simulation (using H_I), as well as from the exact solution. The Ising model free energy was calculated in similar fashion, using an upward temperature scan, but with notable differences: the Wolff algorithm was used, $g(E_g) = g(E_{T_0}) = 2$ and $m = 4 \times 10^4$. In the inset one finds the difference of the numerically computed free energies to the exact Ising model free energy [18], $\epsilon_{Nw/I} = |F_{Nw/I} - F_{Exact}|/L$. The linear increase of ϵ_I with T is in accordance with [39], where this behaviour was found for the 2D Ising model in the high temperature phase. For

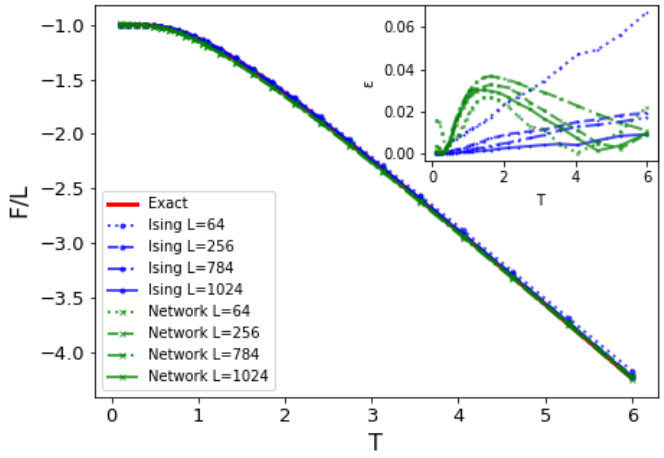


Figure 3.8: *The network free energy (green) along with the 1D Ising model free energy (blue) as a function of temperature $T[J/k_B]$, obtained from the network- and Ising model simulation respectively. The simulations were performed for various sizes L and the calculated free energies are compared to the exact value of the Ising model (red). With the network simulation, the Metropolis algorithm instead of the Wolff algorithm was used and an energy histogram of $m = 2 \times 10^4$ samples at each T was produced. For the Ising model simulation $m = 4 \times 10^4$ at each T was used. The inset shows the absolute difference ϵ of the calculated free energies to the exact value.*

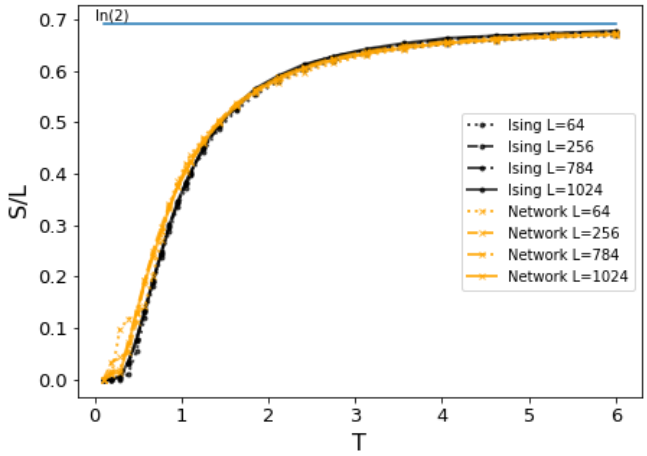


Figure 3.9: *The network (yellow) and 1D Ising (black) entropy as a function of temperature $T[J/k_B]$, obtained from the network- and Ising model simulation respectively. The simulations were carried out for various system sizes L and the entropies were calculated from the corresponding average energy $\langle E \rangle$ and free energy F. For the network simulation $m = 2 \times 10^4$ samples were used- and for the Ising model simulation $m = 4 \times 10^4$ samples were used to calculate each datapoint.*

larger scales, both ϵ_I and ϵ_{Nw} do not exceed 4×10^{-2} for any temperature that was probed. From the simulations used to calculate the free energy we can also compute the entropy. By calculating the average energy $\langle E \rangle$ in addition to F, we can obtain the entropy through (3.17). Figure 3.9 shows the resulting entropies for the network model as well as the Ising model.

We will further discuss the results in subsection 3.2.3. But from the free energy and entropy plot we can already conclude that our constructed network model shares the same thermodynamics as the 1D Ising model. In this sense the mapping is succesful, and it is an encouraging result as we move on to 2D.

3.2.2 AoSD in 2D

Construction Procedure

For the 2D case of our AoSD mapping scheme, we perform a MC simulation which generates configurations of N spins on a square lattice of size $L \times L$ (so $N = L \times L$). Just like in 1D we impose periodic boundary conditions, hence the topology of the boundary of the constructed networks

will be that of the torus. A more natural extension of our 1D mapping, would be to construct networks from spins defined on a spherical lattice, as the sphere is the higher-dimensional topological equivalent of the chain (circle). We have chosen to use the square lattice for a number of reasons, one of which is that it is easier to program. Also, the square lattice Ising model is well studied and the exact solution is known. Moreover, numerical studies of the Ising model on spherical lattices show that it belongs to the same universality class as that of the model defined on the square lattice [41–43]. Therefore, it is immaterial with respect to our goal of engineering a phase transition into our network model whether we use the spherical- or the square lattice.

The procedure by which a network G is constructed from a configuration of spins σ is much the same as it was in 1D. Step 1 consists of bundling all sites within each domain of equal spins to a new node. The resulting new nodes constitute the next to outer shell ($l = 1$) and spin values are assigned to them given by the sum of spin values in the domain they are connected to. Note that contrary to the 1D case, after performing step 1, the original 2D square lattice structure is lost. In 1D, step 1 produces an inner shell of sites which quite naturally can once again be regarded as a 1D lattice with periodic boundary conditions. The same cannot be said of the created inner shell in 2D, and the original 2D lattice. We regard nodes in the $l = 1$ shell to be neighbours if the domains in the $l = 0$ shell to which they are connected, are adjacent. And so in 2D, the $l = 1$ shell need not have the same lattice structure as the $l = 0$ shell. Also, in 1D, the number of domains is always even, which in 2D need not be the case. With these difficulties in mind, we have chosen to let step 2 consist of exactly the same coarse graining procedure that was used in 1D. Thus, the $l = 1$ shell is divided into pairs of neighbouring nodes and each pair is connected to a new node . A new node j, connected to a pair of nodes k_1 and k_2 of the $l = 1$ shell, then receives a spin value $\sigma_j^{(2)} = \text{sgn}\,(\sigma_{k_i}^{(1)})$ from whichever σ_{k_i} is larger in magnitude. If $|\sigma_{k_1}^{(1)}| = |\sigma_{k_2}^{(1)}|$, σ_j is assigned a value of 1 or -1 randomly. If the number of domains is uneven, one set of three nodes are coarse grained together with the same rules applied. In this way step 2 creates the $l = 2$ shell. Next the shells $l = 3, 4$ etc. are created by alternately applying step 1 and 2. The procedure ends when a shell with one node is created. This last node constitutes the root node of the created tree network.

The AoSD construction procedure is practically implemented in a MC simulation of the 2D Ising model via Python in a Jupyter Notebook. We refer to A.4.2 for the code.

Results

The generated networks are characterized by the two phases of the 2D Ising model, where the phase transition occurs at critical temperature $T_c \approx 2.26$ measured in reduced units $[J/k_B]$. In the low temperature phase one finds networks close to the star graph (the groundstate) and in the high temperature phase the networks become ever deeper rooted trees, see figure 3.10.

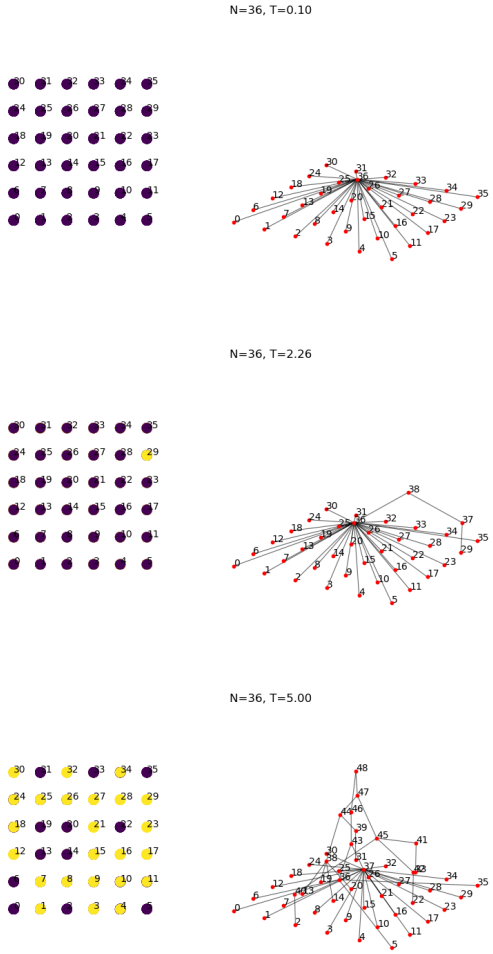
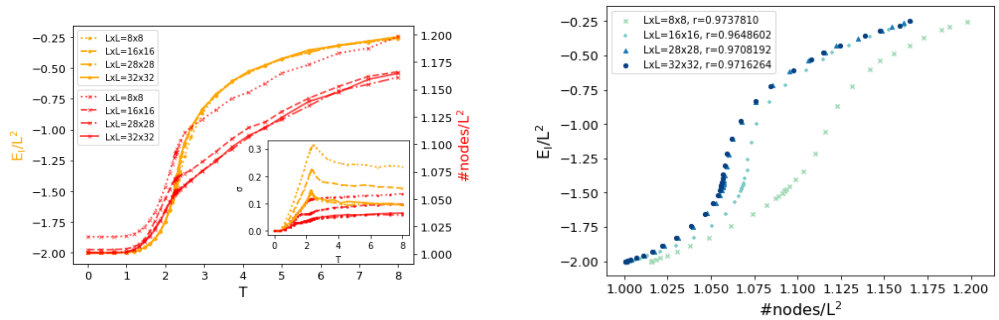


Figure 3.10: *Configurations of the 2D Ising model, along with the corresponding networks that were construced by the AoSD mapping procedure. The configurations/networks are representative for the ones generated by the simulation at a given temperature $T[J/k_b]$. The spins/nodes are labeled by numbers and the node colors (purple/yellow) indicate the spin value.*

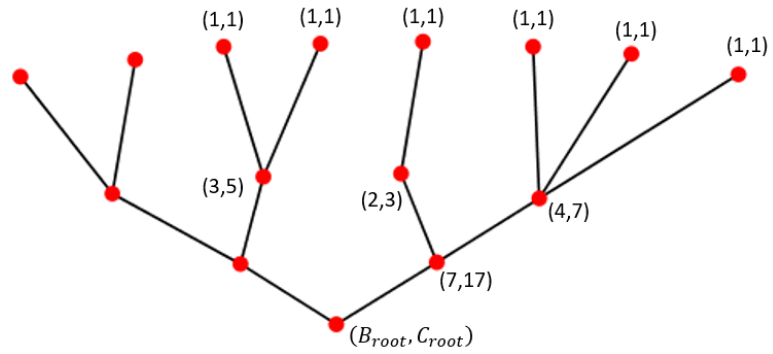
Satisfied with our choice in 1D, we once again consider the number of nodes n as a network property by which we can define an appropiate network hamiltonian that tracks the Ising energy E_I. The average (3.13) of both quantities as a function of T, obtained from simulation performed for different system sizes L are shown in the same plot of figure 3.11a. They were calculated using $m = 2.5 \times 10^4$ samples for each T. In contrast to the result found in 1D, the number of nodes function $n(T)$ clearly deviates from the Ising energy $E_I(T)$. Firstly, different from $E_I(T)$, $n(T)$ increases fairly linearly with T, where a kink is observed around T_c. Secondly, the inset shows the corresponding standard deviations (3.15), where we observe a peak around T_c of the fluctuation in E_I- indicative of the phase transition, while no such behaviour is seen for the fluctuation in n. Finally, the trend with increasing scale L indicates that $n(T)$ and $E_I(T)$ converge to different functions in the thermodynamic limit, as can also be seen from figure 3.11b, where $E_I(T)$ and $n(T)$ are plotted against each other. The legend of figure 3.11b includes the Pearson correlation coefficient $r \in [-1, 1]$ between the two quantities. Though the found values for r indicate a positive linear relationship, we need r to be closer to 1. We conclude that $n(T)$ is not an appropiate function to define the network hamiltonian. In subsection 3.2.3 we will discuss the reason for why, in contrast to 1D, we found such a disparative result in $2D$.



(a) *(b)*

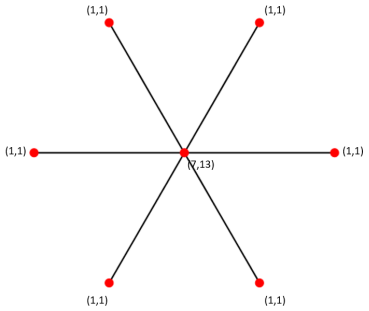
Figure 3.11: *Comparison of the Ising energy E_I with the number of nodes, obtained from MC simulation of the 2D Ising model and the correspondingly constructed networks. The simulation was done for different system sizes L and both quantities are averages over $m = 2.5 \times 10^4$ samples for each temperature $T[J/K_B]$. (a) Ising energy (yellow) and number of nodes (red) are shown as a function of T. The inset shows the corresponding standard deviation σ. (b) The Ising energy plotted against the number of nodes. In the legend the Pearson correlation coefficient r between the two quantities is shown.*

Next we make perhaps a more educated guess on which network function can serve as our network Hamiltonian. We consider a quantitiy used to study *allometric scaling relations* in transport networks, such as river-basin networks or nutrient transport networks found in biological systems [source?]. How this allometric number, N_{alm}, is calculated from a given network is best explained by an example. Consider the tree network shown in figure 3.12a. To each node two numbers (B_i, C_i) are assigned. All nodes on the boundary, i.e. the $l = 0$ shell, receive the values $(B_i^{(0)}, C_i^{(0)}) = (1, 1)$. For the nodes in the interior ($l = 1, 2$ etc.) these



(a)

(b)



(c)

Figure 3.12: *A few exemplary networks with allometric scale. (a) Tree network where some of the assigned numbers (B_i, C_i) are shown. The quantity of interest by which we aim to define the network Hamiltonian is $N_{alm} = C_{root}$. (b) Network topology for which C_{root} is large: the linear chain. Note that it does not belong to the set of networks generated by our mapping. (b) At the other extreme, with a small value for C_{root}, lies the star graph.*

numbers are calculated according to recursive equations:

$$B_i^{(l)} = 1 + \sum_{\langle i,j \rangle} B_j^{(l-1)} \tag{3.25}$$

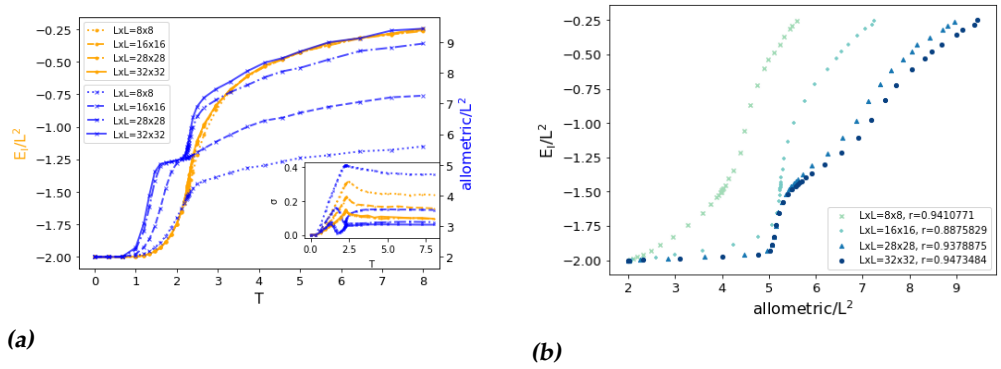
$$C_i^{(l)} = B_i^{(l)} + \sum_{\langle i,j \rangle} C_j^{(l-1)} \tag{3.26}$$

where $\langle i, j \rangle$ denotes that nodes i and j are connected by edges. So, for a node i in shell l, (B_i, C_i) is calculated from all (B_j, C_j) belonging to nodes that branch out off i (and are thus situated in shell $l-1$). The quantity of interest is then C_i of the root node, i.e. $N_{alm} = C_{root}$ will be our candidate for defining a network Hamiltonian.

In the context of transport networks, B_i can be interpreted as the flux (e.g. flow of water or blood per unit time) at node i that it receives from sites upstream (starting from the boundary) in a steady-state supply situation. C_i is then the total amount of fluid per unit time that streams through i and simultaneously through all nodes upstream that are connected to i directly or indirectly. It is a measure of how efficient a given network topology is able to maintain a certain amount of flux (or pressure) at a site downstream. For example, the topology for which C downstream at the root is found to be very large, is that of the linear chain (see figure 3.12b). A very high amount of total substance flowing through the chain is required in order to sustain a certain amount of flux at the root node, rendering it very inefficient. At the other extreme lies the star graph (see figure 3.12c), where one will find a relatively small value for C_{root}.

Note that B_i also equals the number of nodes upstream of i (including i). Hence, at the root it is nothing but the total number of nodes of the network: $B_{root} = n$. The networks generated by our mapping scheme will then obey the scaling relation: $N_{alm} \sim n^{\alpha}$. For networks near the star graph, the exponent α will be close to 1 (isometrically shaped networks), while for deeper rooted networks α will approach 2 (allometric networks).

Making the same comparison as we did for the number of nodes, figure 3.13 shows how the allometric scale N_{alm} compares to the Ising energy E_I. We observe that for the smallest scale used ($L = 8$), $N_{alm}(T)$ follows $E_I(T)$ closely in the low temperature phase, but deviates from it when T is raised above the critical temperature $T_c \approx 2.26$. For increasing scales it seems to be the other way round, e.g. for the largest scale used ($L = 32$), N_{alm} and $E_I(T)$ actually overlap quite nicely in the high temperature phase, while they differ significantly for $T < T_c$. In the range $T =$ 1-2, a curious plateau is found followed by a sudden increase near T_c. This behaviour of $N_{alm}(T)$ is, remarkably so, somewhat reminiscient of latent heat encountered in



(a) *(b)*

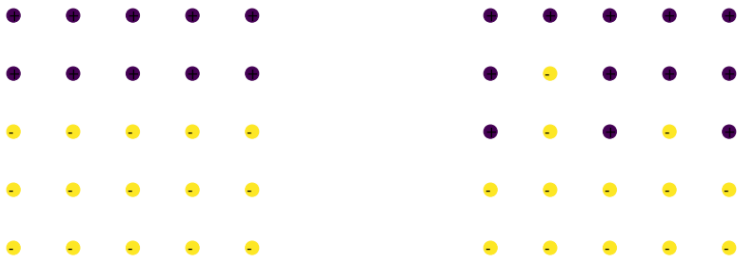
Figure 3.13: *Comparison of the Ising energy E_I with the allometric scale, obtained from MC simulation of the 2D Ising model and the correspondingly constructed networks. The simulation was done for increasing system size L and both quantities are averages over $m = 2.5 \times 10^4$ samples for each temperature $T[J/K_B]$. (a) Both are shown as a function of T. The inset shows the corresponding standard deviation σ. (b) The Ising energy plotted against the allometric scale. In the legend the Pearson correlation coefficient r between the two quantities is shown.*

first order phase transitions.

Though the trend with increasing scale indicates that in the thermodynamic limit $N_{alm}(T)$ will accurately track the Ising energy for high temperatures, we deem it unfit to be the network Hamilonian because of its behaviour for T below and near T_c. It is at this point that we abandoned our search for an appropiate network Hamiltonian for this particular mapping scheme and so also did not proceed in taking the next intended steps (perform separate simulation and compute network free energy). Instead, we decided to develop a whole new mapping scheme, which is the subject of section 3.3.

3.2.3 Discussion

The AoSD mapping in 1D is a showcase example of what we are trying to achieve in this chapter. A function of the whole network (boundary+bulk) was found, by which a network Hamiltonian could be defined that accurately tracks the energy of the boundary model (i.e. Ising model). Simulations done with the network Hamiltonian showed that the free energies of the two models are practically the same (figure 3.8). For some temperatures, the network model free energy was found to be closer to the exact value of the Ising model than the numerically calculated free energy of the latter model.



(a) *(b)*

Figure 3.14: *Two 2D spin configurations with the same number of domains (two) but with different energies (number of misaligned pairs of nearest neighbours).*

Repeating the same steps with a 2D boundary, the results were a little different. In hindsight, the fact that the number of nodes $n(T)$ tracks the Ising energy $E_I(T)$ so poorly in 2D when compared to 1D, can be easily explained by taking a closer look at our construction procedure. By nature of the constructed tree-networks, most nodes are found at the boundary. However, the number of boundary nodes $n^{(0)}$ is constant and so the largest contribution to the variation in $n(T)$ comes from the number of nodes in the next to outer shell, $n^{(1)}$. If one recalls how the $l = 1$ shell is constructed, one sees that $n^{(1)}$ is nothing but the *number of domains* of the spin configuration defined on the boundary. For the 1D Ising model, the number of domains equals, except for a constant factor, the nearest neighbour energy function. This is a property specific to the model in 1D and it does not extend to higher dimensions. See figure 3.14 for a simple example in 2D of two configurations with the same number domains but with different energies.

Finally, in exploring the possibility of using the allometric number N_{alm} to define an appropiate network Hamiltonian, we found a curious serendipitous result. For large enough scales, $N_{alm}(T)$ sharply increases for T well below the critical temperature T_c. The increase is followed by a levelling off when the temperature is raised, which in turn is followed by a sudden increase when T is raised even further and approaches T_c. This is somewhat similar to latent heat found in systems that exhibit a first order phase transition, which is remarkable as the (2D) Ising model is an example of a second order (continuous) phase transition. Further study would be needed in order to fully explain this finding.

3.3 Second Mapping Scheme: RG blocking

Compared to our first mapping, the networks generated by our second mapping scheme will be more rigid and thus less dynamical in nature. A tree network will serve as an underlying lattice, where spins are defined on its nodes by a procedure intricately related to the block spin renormalization group (RG) formalism. Hence, we will coin this second mapping scheme 'RG blocking' (RGb). This time we have paid special attention to designing the dynamics such that the Ising energy is reflected into the topology of the network. Once again, we will first treat the mapping of the 1D Ising model, before moving on to 2D.

3.3.1 RGb in 1D

Construction Procedure

Configurations of N spins $\sigma = \{.., \sigma_i, ..\}$ are generated by the MC simulation of the 1D Ising model. The lattice has size L (so $N = L$) and periodic boundary conditions are imposed. The input parameter of the simulation is the temperature $T[J/k_B]$, which controls the configuration probability.

A configuration σ is mapped onto a network G as follows. First of all, the spins σ_i of the Ising chain are identified as the boundary nodes (shell $l = 0$) of a full b-ary tree network, where the branching factor b equals 2. Then, the parent nodes of these, in shell $l = 1$, are identified with block-spins $\sigma_i^{(1)}$ that result from blocking together the spins defined on their offspring. So, the branching factor b is also the block size. See figure 3.15 for an accompanying picture. The block-spins receive a spin value of ± 1 by majority rule. In the case of a tie, they inherit the spin value of the most clockwise spin in their block. This whole identification procedure is repeated for all shells. So, the lth shell corresponds to the lth iteration of blocking spins. Note that there are $N^{(l)} = N/b^l$ number of spins in shell l. Next, dynamics are introduced by adding edges $e_{ij}^{(l)}$ across neighbouring spins i, j (of the same shell) that are misaligned. Letting these cross-edge variables be represented by numerical values as follows:

$$e_{ij}^{(l)} = \begin{cases} 1 & \text{if an edge is present between node } i \text{ and } j \\ 0 & \text{else} \end{cases} \tag{3.27}$$

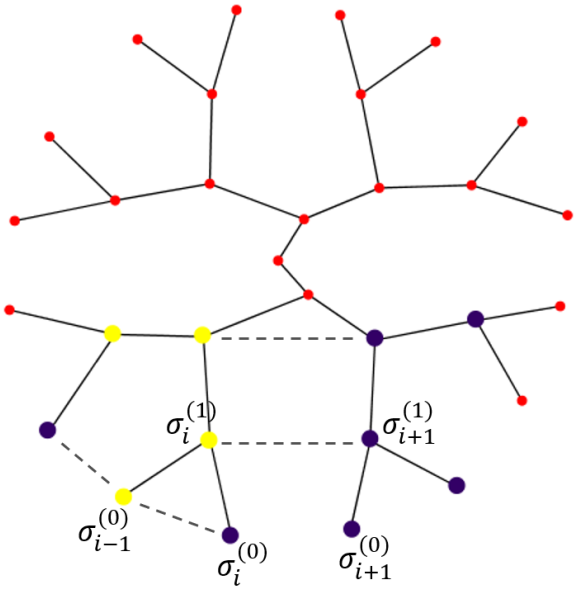


Figure 3.15: *RGb mapping procedure. The Ising spins $\sigma_i^{(0)}$ are defined on the boundary nodes (shell $l = 0$) of a b-ary tree network (with branching factor $b = 2$) and take on spin values ± 1 (yellow/purple). Nearest neighbour spins belonging to the same parent node are blocked together. The resulting block-spins $\sigma_i^{(1)}$ are associated with the parent nodes in shell $l = 1$. This is repeated for all shells untill all nodes have a designated spin value $\sigma_i^{(l)}$. The mapping is completed by adding edges e_{ij} between all nearest neighbour spins (within the same shell) that are misaligned (dashed edges).*

they are related to the spin variables by

$$e_{ij}^{(l)} = \frac{1 - \sigma_i^{(l)} \sigma_j^{(l)}}{2} \tag{3.28}$$

where the subscripts i, j index nearest neighbours. With the placement of these edges onto the network the mapping is completed.

The RGb construction procedure is practically implemented in a MC simulation of the 1D Ising model via Python in a Jupyter Notebook. We refer to A.4.3 for the code.

Results and Discussion

The corresponding network to the groundstate of the Ising model is clearly the tree network. With the increase of temperature $T[J/k_B]$, the networks

have more and more cycles that are found throughout all shells, as shown in figure 3.16.

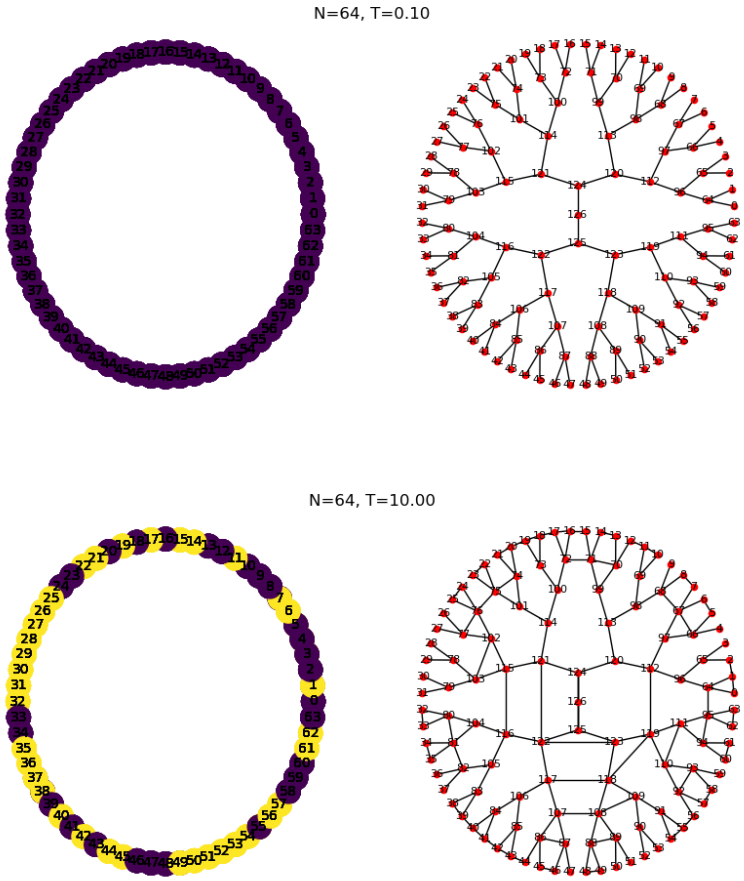


Figure 3.16: *Configurations of the 1D Ising model, along with the corresponding networks that were construced by the RGb mapping procedure. The configurations/networks are representative for the ones generated by the simulation at a given temperature $T[J/k_b]$. The spins/nodes are labeled by numbers and the node colors (purple/yellow) indicate the spin value.*

Our RGb mapping scheme is very reminiscient of the *Monte Carlo renormalization group* (MCRG) [44–47]- a method developed to study critical phenomena that combines MC simulations with real-space renormalization [48]. The strategy of MCRG is to use a MC simulation to generate a sequence of configurations characteristic of the system's Hamiltonian. Then an RG transformation (e.g. blocking spins) is applied directly on

these configurations. The result is a sequence of configurations characteristic of the (original) system with a *renormalized Hamiltonian*. One can iterate the RG transformation on the produced sequences of configurations-and one usually does so to find the RG flow near the critical point and ultimately the critical exponents. In this light, the generated networks can be seen as constructs that graphically represent the whole RG procedure. The lth shell of the network corresponds to a configuration of spins characteristic of the renormalized Hamiltonian for the lth iteration of the RG transformation.

By (3.28), the cross-edge variables $e_{ij}^{(l)}$ correspond to the nearest neighbour interactions of the (block-)spins. Therefore, we can write an approximation to the renormalized Hamiltonian in terms of these:

$$\begin{aligned} H^{(l)} =& \sum_{i,j} -J\sigma_i^{(l)}\sigma_j^{(l)} = -J\sum_{i,j} 1 - 2e_{ij}^{(l)} \\ =& J(-n^{(l)} + 2e^{(l)}) \end{aligned} \tag{3.29}$$

where $n^{(l)}$ is the maximum number of possible cross-edges in shell l (a constant) and $e^{(l)} = \sum_{i,j} e_{ij}^{(l)}$ is the actual number of cross-edges present in shell l. Note that as the shells are one-dimensional, $n^{(l)} = N^{(l)}$. For $l > 0$, $H^{(l)}$ is an approximation to the full renormalized Hamiltonian, where we have tacitly assumed it to be of the same form as the original Hamiltonian $H^{(0}$, i.e. incorporating only nearest neighbour interactions. This turns out to be exactly valid for the 1D Ising model, but usually the renormalized Hamiltonian may contain higher order interactions (next nearest neighbour etc.) or even an infinite number of coupling constants. Furthermore, note that $H^{(l)}$ is found explicitly from the block-spins. To complete the RG analysis, one would want to express $H^{(l)}$ in terms of the original spins $\sigma_i^{(0)} = \sigma_i$ (accompanied by an appropiate rescaling) and a renormalized coupling constant $J^{(l)}$ (or equivalently a renormalized temperature $T^{(l)}$).

Figure 3.17 shows the energies for some shells, as given by (3.29). They were obtained from the networks generated by the MC simulation and each data point is an average over $m = 2.5 \times 10^4$ samples for each temperature T. The Ising energy is also shown, which by construction corresponds exactly to the energy of the outer shell, i.e. $H_I = H^{(0)}$. For increasing l, $H^{(l)}$ is found to be increasingly higher relative to $H^{(0)}$ for the same T. This is in accordance with the flow of the RG transformation for the Ising model in 1D. The 1D Ising model has two fixed points, an attractive one at $K = \beta J = 0$ (high temperature $T \to \infty$ or equivalently weak coupling $J \to 0$) and a repulsive one at $K \to \infty$ (low temperature $T \to 0$ or

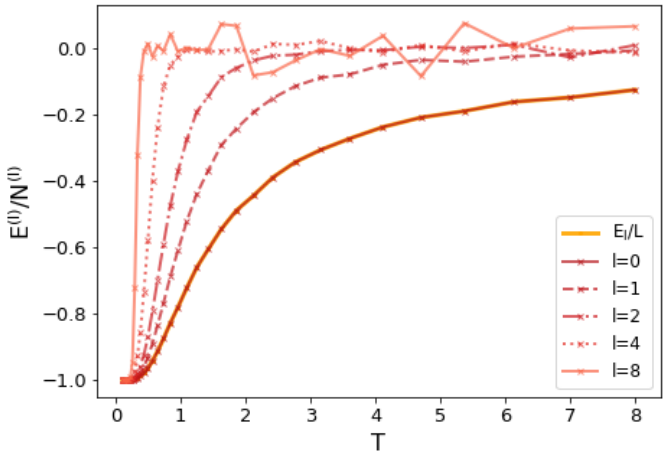


Figure 3.17: *Energies $E^{(l)}$ as given by (3.29) are shown as a function of temperature $T[J/k_B]$ for some shells. Also shown is the Ising energy E_I as a function of T, which by construction is exactly equal to $E^{(0)}$. Each datapoint is an average over $m = 2.5 \times 10^4$ samples, obtained from the MC simulation of the 1D Ising model and the correspondingly constructed networks. The Ising model was simulated on a lattice with size $L = 1024$. Note the values for $E^{(l)}$ at a given T are increasingly higher with increasing l, in accordance with the RG flow for the 1D Ising model.*

strong coupling $J \to \infty$). Thus, starting with the system at any finite T, the RG transformation is equivalent to making it more disorderd, i.e. finding higher energies in the system.

A natural candidate to consider then as the network Hamiltonian is the sum of all shell energies:

$$\begin{aligned} H_{Nw} &= \frac{1}{r_{geom}} \sum_{l=0}^{n-1} H^{(l)} \\ &= \frac{1}{r_{geom}} \sum_{l=0}^{n-1} \sum_{i,j} -J\sigma_i^{(l)}\sigma_j^{(l)} \\ &= \frac{J}{r_{geom}} \sum_{l=0}^{n-1} (-n^{(l)} + 2e^{(l)}) \end{aligned} \tag{3.30}$$

where n is the number of shells of the network. From the network perspective, H_{Nw} is nothing more than a function of the network that traces the total number of cross-edges $e = \sum_l e^{(l)}$. We have included a rescaling

factor r_{geom} to make the groundstate energy level per boundary spin of the Ising model and the network model coincide. Without rescaling (i.e. $r_{geom} = 1$), H_{Nw} would differ from the Ising energy H_I for two reasons: 1) as discussed above, the $H^{(l)}$'s for $l > 0$ differ qualitatively from H_I in accordance with the RG flow of the model, and thus so does H_{Nw}; 2) there is a quantitative difference, simply due to the fact that H_{Nw} is a sum over all shells. By rescaling- or equivalently setting the coupling strength of a cross-edge to $\bar{J} = J/r_{geom}$, we eliminate the difference stemming from the latter reason. The rescaling factor is determined by how much larger the total number of sites of the network is compared to the number of sites of its boundary. We can find this size difference from the bulk-boundary ratio, which is solely determined by the branching factor b. The number of sites found at the boundary is $N_{bd} = b^n$. The number of sites found in the bulk is given by a geometric series $N_{bulk} = \sum_{k=0}^{n-1} b^k$. In the thermodynamic limit $n \to \infty$, we find $N_{bulk}/N_{bd} = 1/1 - b$. As $b = 2$, the ratio bulk:boundary is $1 : 1$, so the network is twice the size of its boundary, and we set the rescaling factor to $r_{geom} = 2$.

Figure 3.18 shows how the network energy H_{Nw} (as given by (3.30)) compares to the Ising/boundary energy H_I. The averages over $m = 2.5 \times 10^4$ samples of both quantities are plotted as a function of temperature. They were obtained from simulations done for various sizes L. In the inset

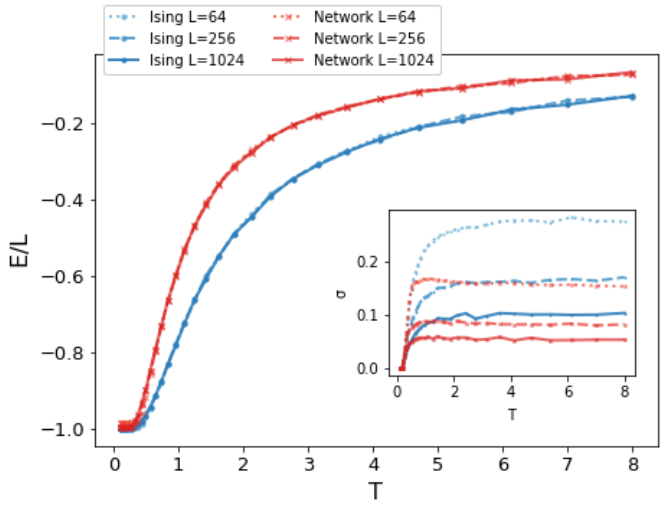


Figure 3.18: *Network energy (red), as defined by (3.30), and the 1D Ising energy (blue) are shown as a function of temperature $T[J/k_B]$. Both are averages over $m = 2.5 \times 10^4$ samples for each T, obtained from MC simulations performed for various system sizes L. The inset shows the corresponding standard deviations.*

we see that the corresponding standard deviations decreace with increasing L. The network energy is found to be higher than the Ising energy for virtually all T, which follows from the aggregate effect of having higher energies in the inner shells. Despite this difference, figure 3.19 shows that the network *free energy*- obtained from a simulation where the graphs/-configurations were weighted in accordance with H_{Nw} (i.e. the weights are given by (3.4)), is still found to be very close to the Ising free energy (both exact and numerical). With the largest scale used ($L = 1024$), the dif-

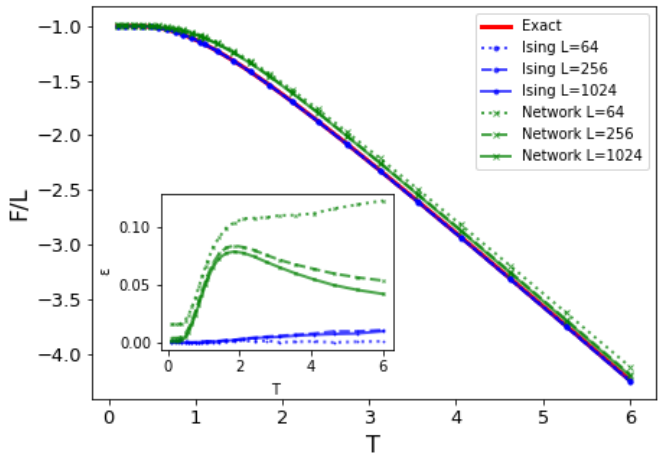


Figure 3.19: *Network (green) and 1D Ising model (blue) free energy as a function of temperature $T[J/k_B]$. They were obtained from the network- and Ising simulation respectively, which were performed for various sizes L. Also shown is the exact value (red) of the Ising model. With the network simulation, the Metropolis algorithm instead of the Wolff algorithm was used and an energy histogram of $m = 2 \times 10^4$ samples at each T was produced. For the Ising model simulation $m = 4 \times 10^4$ at each T was used. The inset shows the absolute difference ϵ of the calculated free energies to the exact value.*

ference to the exact value, $\epsilon_{Nw} = |F_{Nw} - F_{Exact}|/L$ is found to be smaller than 0.1 for any T that was used. The free energies were calculated through the histogram method described in subsection 3.1.1. For the upward temperature scans, an energy histogram of $m = 2 \times 10^4$ and $m = 4 \times 10^4$ samples at each T was used for the network- and Ising model simulation respectively. The scans were also used to compute the entropies (via 3.17), and the result is shown in figure 3.20.

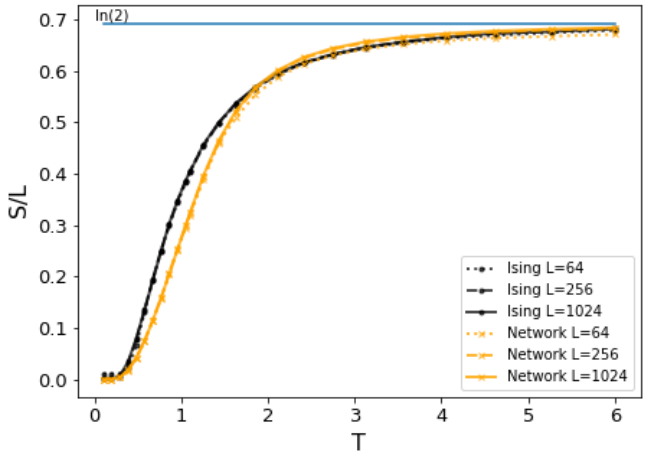


Figure 3.20: *Network (yellow) and 1D Ising (black) entropy as a function of temperature $T[J/k_B]$, obtained from the network- and Ising model simulation respectively. The simulations were carried out for various system sizes L and the entropies were calculated from the corresponding average energy $\langle E \rangle$ and free energy F. For the network simulation $m = 2 \times 10^4$ samples were used- and for the Ising model simulation $m = 4 \times 10^4$ samples were used at each T.*

3.3.2 RGb in 2D

Construction Procedure

We use the MC simulation of the Ising model to generate configurations of N spins $\sigma = \{.., \sigma_i, ..\}$ on a square lattice of size $L \times L$ (so $N = L \times L$). Periodic boundary conditions are imposed, which means that the topology of the lattice is that of the torus. The configuration probability is controlled by inputting the temperature $T[J/k_B]$.

A spin configuration σ is mapped onto a network G by essentially the same procedure used for the 1D case (see (3.3.1)). We once again start by identifying the Ising spins σ_i with the boundary nodes of a full b-ary tree. Ofcourse, as the spins are now defined on the 2D lattice, the boundary of the network is the square lattice instead of the linear chain. The lattice is divided into 2×2 squares to form block-spins $\sigma_i^{(1)}$, which are identified as the parent nodes of the spins in their corresponfing blocks. So, the branching factor/block size b is equal to 4. The block-spins receive their spin value ± 1 by majority rule. In the case of a tie, $\sigma_i^{(1)}$ receives the spin value of the spin in the top left corner of its block. In this way the $l = 1$ shell is formed. It is again a square lattice on which we impose periodic boundary

conditions. Next, it is in turn divided into 2×2 blocks to form the block-spins of the next shell. This process is repeated forming square lattices of decreasing size untill one arrives at the root node of the tree network. See figure 3.21 for an illustration. Note that a shell l counts $N^{(l)} = N/b^l$ spins defined on a lattice of size $L^{(l)} \times L^{(l)}$ (so $L^{(l)} = L/b^{l/2}$). The mapping is finalized by adding edges $e_{i,j}^{(l)}$ across nearest neighbouring spins of the same shell that are misaligned. In other words, all nearest neighbour interactions of spins within the same shell are graphically represented by these cross-edges. They are related to the spins by (3.28).

The RGb construction procedure is practically implemented in a MC simulation of the 2D Ising model via Python in a Jupyter Notebook. We refer to A.4.4 for the code.

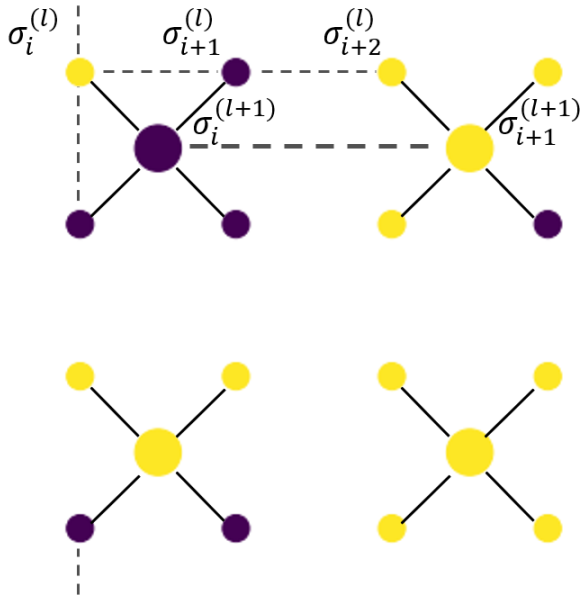


Figure 3.21: *Top view of two consecutive layers that are part of the network constructed by the RGb mapping procedure of the 2D Ising model. The square lattice of spins $\sigma_i^{(l)}$ (small circles) that constitute the lth shell of the network, is divided into 2×2 blocks. The blocks are used to define new spins $\sigma_i^{(l+1)}$ (large circles) by a coarse-graining procedure (majority rule). They are connected to all the spins in their corresponding blocks by edges (solid lines) and make up the $l+1$ shell of the network. All spins can take on values ± 1 (yellow/purple) and edges (dashed lines) are drawn between nearest neighbour spins of the same shell whenever they are misaligned.*

Results and Discussion

As already discussed in the 1D case (3.3.1), the networks resulting from the RGb mapping (see figure 3.22 for two examples) can be regarded as constructs that embody the whole RG procedure. On the boundary shell ($l = 0$) of the network one has the original system, i.e. the Ising model. In the inner shells ($l > 0$) one finds spins characteristic of the original model

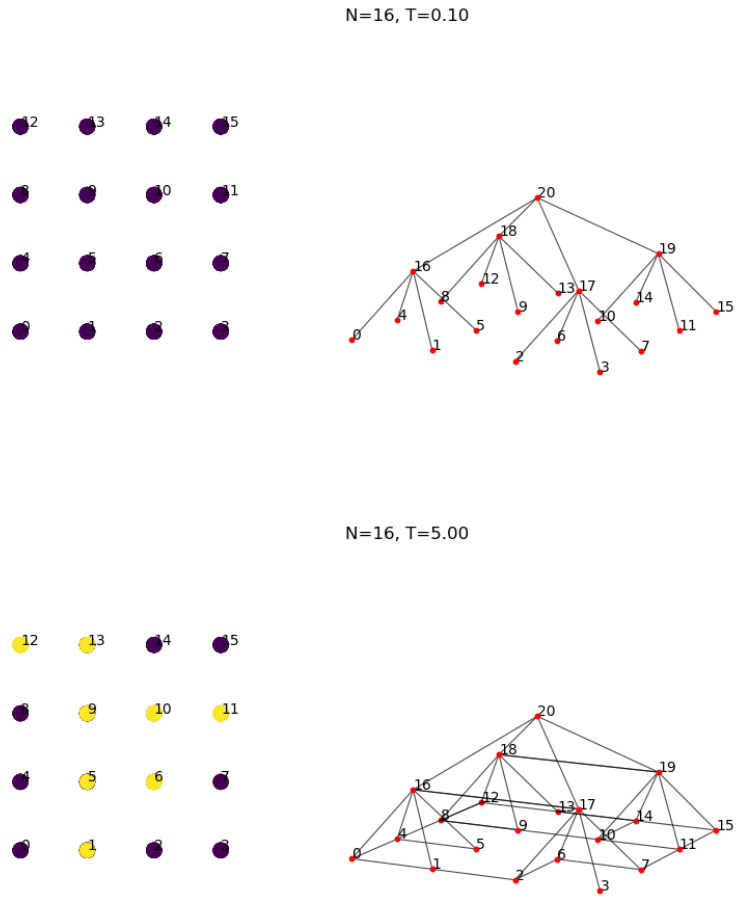


Figure 3.22: *Configurations of the 2D Ising model, along with the corresponding networks that were construced by the RGb mapping procedure. The configurations/networks are representative for the ones generated by the simulation at a given temperature $T[J/k_b]$. The spins/nodes are labeled by numbers and the node colors (purple/yellow) indicate the spin value.*

with a renormalized Hamiltonian, where going radially inward into the network corresponds to the RG flow of the model's parameter $K = \beta J$. The 2D Ising model has in addition to the trivial fixed points $K = 0$ and $K = \infty$, a critical fixed point at $K_c \approx 0.441$ ($T_c \approx 2.26$) where the phase transition occurs. The shell energies $H^{(l)}$, composed of nearest neighbour interactions between the (block-)spins (see 3.29), give for $l > 0$ a (crude) approximation to the renormalized Hamiltonians. From the network perspective these are (apart from a constant) equal to the number of cross-edges $e^{(l)}$ found in the shell. Note that the number of maximum possible cross-edges in a shell is now equal to twice the number of spins found in the shell: $n^{(l)} = 2N^{(l)}$.

Figure 3.23 shows shell energies as a function of T, obtained from the MC simulation using $m = 2.5 \times 10^4$ samples. The Ising energy, which by construction corresponds to the boundary shell energy $H_I = H^{(0)}$, is also shown. For $l > 0$, we observe a distinct difference in the behaviour of the $H^{(l)}$'s depending on which phase the system is in. In the high temperature phase ($T > T_c$), we find $H^{(l)}$ to be increasingly higher relative to $H^{(0)}$ with increasing l. This corresponds to the RG flow towards the

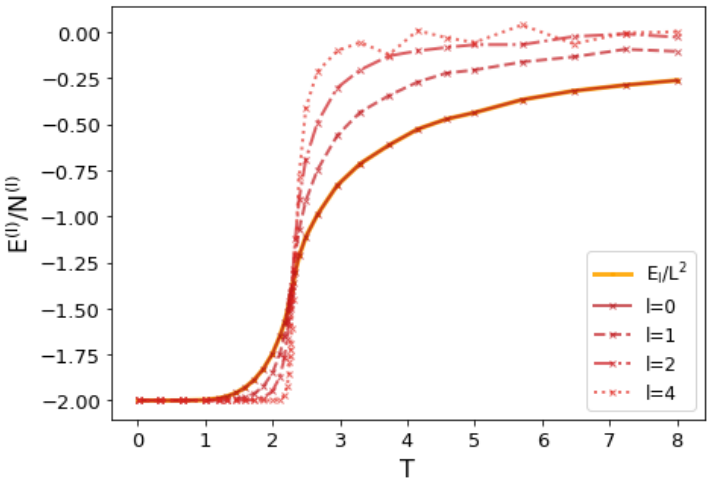


Figure 3.23: *Energies $E^{(l)}$, defined by (3.29), are shown as a function of temperature $T[J/k_B]$ for some shells. Also shown is the Ising energy E_I as a function of T, which by construction is exactly equal to $E^{(0)}$. The data is obtained from the MC simulation of the 2D Ising model and the correspondingly constructed networks, where $m = 2.5 \times 10^4$ samples and a boundary lattice of size $L \times L = 32 \times 32$ was used. We observe that the $E^{(l)}$'s compare to $E^{(0)}$ in accordance with the RG flow of the 2D Ising model.*

high temperature fixed point $K = 0$, which was also observed for the 1D mapping. Different from the 1D case however, we now have a low temperature phase and the low temperature fixed point $K = \infty$ is attractive. Thus for $T < T_c$, we see that $H^{(l)}$ is found to be increasingly lower relative to $H^{(0)}$ with increasing l. The intersection point of all shell energies around the critical temperature $T_c \approx 2.26$ corresponds to the non-trivial fixed point where the system is expected to display universal behaviour. In other words, the scale invariance of the Ising model at the fixed points is explicitly reflected into the network, as there the network appears the same throughout all shells, i.e. the fraction of cross-edges $e^{(l)}/n^{(l)}$ is the same.

We define the network energy H_{Nw} in the same way as for the 1D case, namely as the sum over all shell energies $\frac{1}{r_{geom}} \sum_{l=0} H^{(l)}$ rescaled by the factor r_{geom}, see (3.30). As the branching factor $b = 4$, the bulk-boundary ratio in the thermodynamic limit is now $1 : 3$ and so we rescale by a factor $r_{geom} = 4/3$. Figure 3.24 shows how the network energy H_{Nw} compares to the Ising/boundary energy H_I as a function of T, both obtained from the simulation using different sizes L and $m = 2.5 \times 10^4$ samples. We see that H_{Nw} tracks H_I reasonably well, especially in the range $T < T_c$. For temperatures above T_c, the network energy is found to be slightly higher

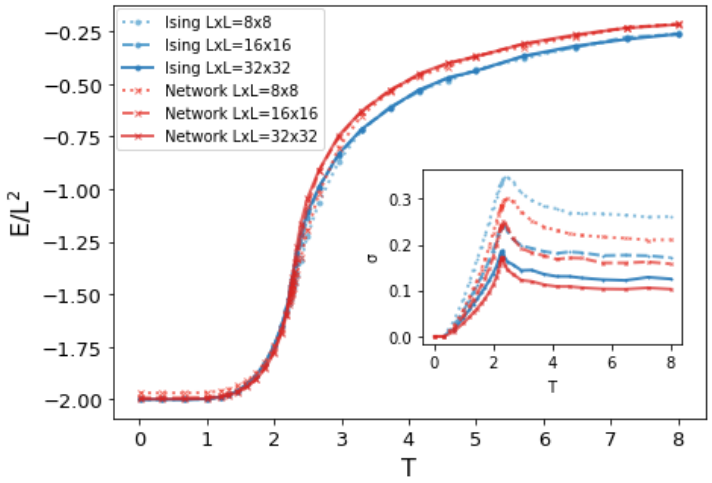


Figure 3.24: *Network energy (red), defined by (3.30), and the 2D Ising energy (blue) are shown as a function of temperature $T[J/k_B]$. Both are averages over $m = 2.5 \times 10^4$ samples for each T, obtained from MC simulations performed for various system sizes L. In the inset the corresponding standard deviations are shown.*

than the Ising/boundary energy, similar to what was found in 1D. The discrepancy is much less though here in 2D, which is to be expected from the higher boundary-to-bulk ratio. In the inset the corresponding standard deviations show a peak around T_c, highlighting the increase of fluctuations near criticality.

Having performed a separate MC simulation, where H_{Nw} was used to weight the generated networks (i.e. the target distribution is (3.4)), we calculated the free energy of the network ensemble using the histogram method described in subsection 3.1.1. Figure 3.25 shows the network free energy as a function of T, where it is compared to the Ising free energy, given exactly as well as numerically from simulation. Both the numerical Ising and network free energy were obtained by an upward temperature scan, where an energy histogram of $m = 4 \times 10^4$ and $m = 2 \times 10^4$ samples was used for each T_i respectively. We observe that for the most part the network free energy agrees with its Ising counterpart; only around T_c we find a slight difference. In fact, as shown in the inset, the difference of the network free energy to the exact value of the Ising model

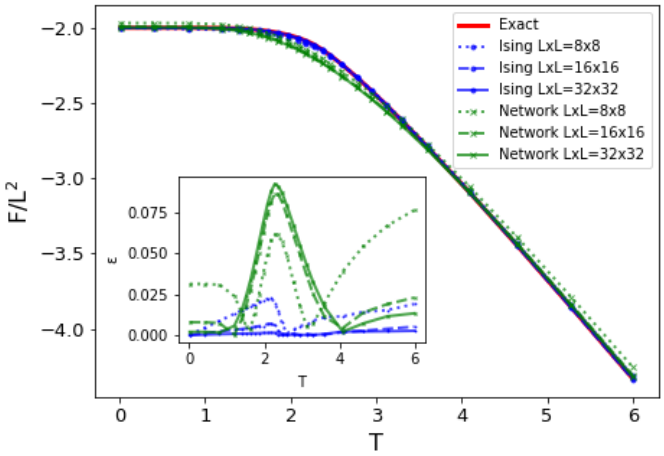


Figure 3.25: *Network (green) and 2D Ising model (blue) free energy as a function of temperature $T[J/k_B]$. They were obtained from the network- and Ising simulation respectively, which were performed for various sizes L. With the network simulation, the Metropolis algorithm instead of the Wolff algorithm was used and an energy histogram of $m = 2 \times 10^4$ samples at each T was produced. For the Ising model simulation $m = 4 \times 10^4$ at each T was used. Also shown is the exact value (red) of the Ising model. The inset shows the absolute difference ϵ of the calculated free energies to the exact value.*

$\epsilon_{Nw} = |F_{Nw} - F_{Exact}|/L^2$ shows a clear peak at $T_c \approx 2.26$. As H_{Nw} was expected and found to be close to H_I at T_c (fixed point), this difference is somewhat unexpected. The peak in ϵ_{Nw} may result from the histogram method being less accurate near the order-disorder transition rather than that it reflects a true difference in the free energies. It would be good to get an indication of the error in the histogram method near T_c, perhaps by calculating the free energy numerous times from independent data (i.e. running the simulation multiple times) and then considering the corresponding standard deviation.

Comparison of the network entropy with that of the Ising model (see figure 3.26), both calculated from the free energy data through (3.17), shows a different picture. The network entropy intersects the Ising entropy at T_c

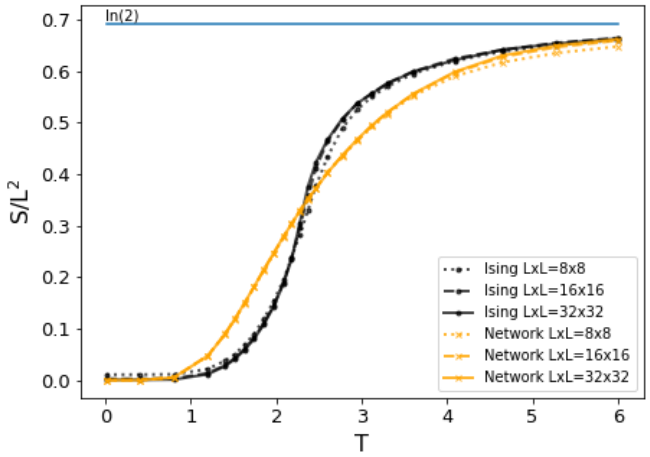


Figure 3.26: *Network (yellow) and 2D Ising (black) entropy as a function of temperature $T[J/k_B]$, obtained from the network- and Ising model simulation respectively. The simulations were carried out for various system sizes L and the entropies were calculated from the corresponding average energy $\langle E \rangle$ and free energy F. For the network simulation $m = 2 \times 10^4$ samples were used- and for the Ising model simulation $m = 4 \times 10^4$ samples were used at each T.*

and deviates from it below and above T_c. As $T \to 0$ or $T \to \infty$ the entropies become equal again. This is in line with the RG flow and the three fixed points of the Ising model. For example, for $T > T_c$ the energies in the inner shells are found to be higher than the Ising/boundary energy (thus H_{Nw} is higher than H_I), and so disorder is penalized more in our network model resulting in a lower entropy.

In summary, we have mapped each configuration of the 2D Ising model

onto a network through the RGb procedure. Weighted with the Ising measure (3.2), the resulting network ensemble trivially inherits the thermodynamics of the Ising model. We then studied by means of a MC simulation, a network ensemble consisting of the same networks (i.e. the same state space), but weighted with a different measure (3.4) using a network Hamiltonian H_{Nw} instead of the Ising energy H_I. The network Hamiltonian was defined to be a function of the whole network, consisting of a 'boundary term' ($H^{(0)}$) that corresponds exactly to H_I and an additional term ($\sum_{l>0} H^{(l)}$) that is a function of the 'bulk'. Our result (figure 3.25) shows that despite the added contribution from the bulk, the free energy of this network model remains more or less equal to the free energy of the Ising/boundary model. With this, we argue that we have demonstrated the relative ease of obtaining a model with holographic properties when hyperbolic structure is imposed. The argument is not without a few disclaimers. In defining H_{Nw} we added a rescaling factor which takes care of the quantitative difference with H_I. Also, as the bulk of the networks were constructed by an RG procedure starting from the boundary, the effective interactions between the block-spins in the inner shells *near the boundary* is still similar to that of original Ising spins. As one moves inward in the network towards the root, the interactions between the block-spins differ more and more from that of the original system, but by hyperbolicity contribute less and less to the network's Hamiltonian. Thus, it really is a combination of the tree-like structure *and* the RG flow that results in H_{Nw} remaining close to H_I. So, the effect of hyperbolicity demonstrated here is a subtle one.

Chapter 4

Creating an Independent Holographic Network Model by using the Maximum-Entropy Method and RG formalism

In the previous chapter, we discussed our attempts of creating a network model whose dynamics is equivalently descibed by a statistical mechanical model defined on its boundary. Our attempts consisted of mapping each configuration of the (1D or 2D) Ising model onto a network. Weighted with the Ising measure, the resulting network ensemble is trivially identical to the boundary model. We tried to extend the network model beyond the Ising model by weighting the networks with a different measure, using a Hamiltonian given by a network property. This represents a model that is only semi-independent of the Ising model, as the networks require the Ising spins for construction. In order to realize a full duality, the network model should be able to be constructed independently from the Ising model. In this chapter, we provide ways of contructing a network model that roughly resembles the one resulting from our last mapping scheme, the RGb mapping, but without making reference to the Ising spins.

4.1 RGb mapping model

Before discussing ways to create a network model that is established independently from the Ising model, it is useful to highlight certain aspects of the networks resulting from the RGb mapping. Recall from section 3.3

that a RGb network is a b-ary tree enhanced with cross-edges $e_{ij}^{(l)}$, linking nearest neighbour nodes i and j within shell l. The cross-edges graphically represent the nearest neighbour interactions of the Ising (block)-spins $\sigma_i^{(l)}$, i.e. they take on values 0 or 1 as given by (3.28). As each network is constructed from a configuration of Ising spins defined on the boundary, the networks occur with the probability given by the Ising measure (3.2). So, the model's Hamiltonian is given by the nearest neighbour interactions of the Ising spins:

$$\begin{aligned} \beta H_I &= \sum_{\langle i,j \rangle} -\beta J \sigma_i^{(0)} \sigma_j^{(0)} = \sum_{\langle i,j \rangle} -\beta J (1 - 2e_{ij}^{(0)}) \\ &= \beta J (-n^{(0)} + 2e^{(0)}) \end{aligned} \tag{4.1}$$

where in the last line it is written in terms of the number of links $e^{(0)}$, and the maximum number of possible links (number of nearest neighbour interactions) $n^{(0)}$ of the boundary shell. The parameter of the model is $K = \beta J$, (β the inverse temperature and $J = J^{(0)}$ the coupling constant).

As alluded to in subsection 3.3.1, the sequence of configurations of *block-spins* $\sigma_i^{(l)}$ in shell $l > 0$, can be characterized as a system of the *original spins*, $\sigma_i^{(0)} = \sigma_i$, with a renormalized hamiltonian $H_I^{(l)}$. We assume for simplicity's sake that the hamiltonians will be of the same form, i.e. only incorporating nearest neighbour interaction. Also, let's keep the temperature $T[J/k_B] = \beta^{-1}$ the same for all shells, so all of the RG-flow goes into $J^{(l)}$. Then

$$H_I^{(l)} = \sum_{\langle i,j \rangle} -J^{(l)} \sigma_i^{(0)} \sigma_j^{(0)} \tag{4.2}$$

and it is found from the block-spins explicitly:

$$\left\langle H_I^{(l)} \right\rangle = \left\langle \sum_{\langle i,j \rangle} -J^{(0)} \sigma_i^{(l)} \sigma_j^{(l)} \right\rangle = J^{(0)} \left(-n^{(l)} + 2 \left\langle e^{(l)} \right\rangle \right) \tag{4.3}$$

The ratios $J^{(l)}/J^{(0)}$ can then be obtained from simulation using:

$$\frac{J^{(l)}}{J^{(0)}} = \frac{\left\langle H_I^{(l)} \right\rangle / N^{(l)}}{\left\langle H_I^{(0)} \right\rangle / N^{(0)}} \tag{4.4}$$

where $N^{(l)}$ is the number of spins in shell l. In figure 4.1, these ratios-obtained from simulation of the RGb mapping of the 1D Ising model on

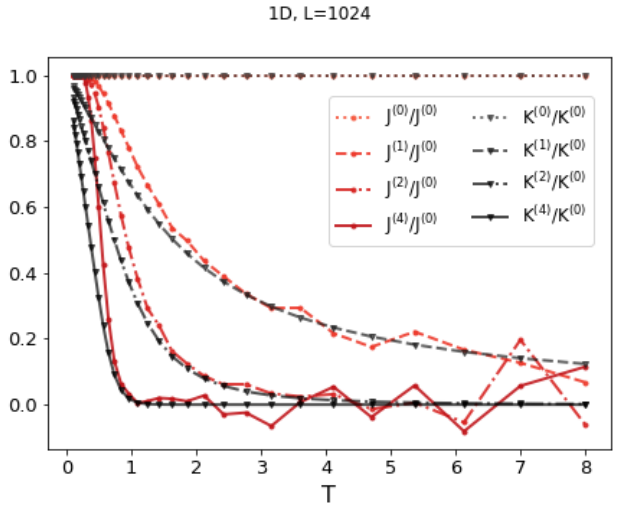


Figure 4.1: *The ratio of $J^{(l)}$ to the original coupling constant $J = J^{(0)}$ (red) as a function of T are shown for some shells. They are calculated from the average shell energies by (4.4), which were obtained from simulation of the RGb mapping of the1D Ising model on a $L = 1024$ lattice. For each T, $m = 2.5 \times 10^4$ samples were used. Also shown are theoretical counterparts (black), obtained from a decimation scheme [49].*

a $L = 1024$ lattice, are plotted as a function of T. So, for a given T, they show the RG flow along the shells. (Note that the plotted $J^{(l)}$'s essentially display the same information as was shown in figure 3.17.) As a reference, we have also included these ratios as found from theory [49]. The theoretical results we consider come from a decimation scheme. Starting from the input parameter $K^{(0)} = \beta J^{(0)}$, $K^{(l)}$ is found recursively by:

$$K^{(l)} = \frac{1}{2} \ln \left(\cosh 2K^{(l-1)} \right) \tag{4.5}$$

We find $J^{(l)}$'s to be in reasonable agreement with the theoretical values. For low values of T, they start to deviate slightly, which is probably a finite size effect. For finite lattices, it becomes harder to sample the true statistics of the model at low temperatures (strong coupling).

4.2 (Exponential) Random Graph Model

As a starting point in creating an independent network model, we propose to use the data gathered from the simulated RGb networks, for the creation

of a *random graph model* in their image. We start again with the b-ary tree, but now place each cross-edge $e_{ij}^{(l)}$ with a probability $p_{ij}^{(l)}$. The connection probabilities $p_{ij}^{(l)}$ are chosen in such a way that the links $e_{ij}^{(l)}$ are on average, occupied the same number of times as encountered in the RGb mapping model. Note that as $e_{ij}^{(l)}$ is a binary variable, one has:

$$\left\langle e_{ij}^{(l)} \right\rangle = \sum_{e_{ij}^{(l)}=0}^{1} p_{ij}^{(l)} e_{ij}^{(l)} = p_{ij}^{(l)} \tag{4.6}$$

Also, due to symmetry, the connection probabilities are only l-dependent, i.e. $p_{ij}^{(l)} = p^{(l)}$; and they are equal to the average fraction of cross-edges encountered in the shell:

$$p^{(l)} = \left\langle e_{ij}^{(l)} \right\rangle = \frac{\left\langle e^{(l)} \right\rangle}{n^{(l)}} \tag{4.7}$$

with $e^{(l)}$ denoting the number of cross-edges and $n^{(l)}$ the maximum number of possible cross-edges in shell l. So, using the average $e^{(l)}$'s obtained from the simulation (which essentially are identical to the average shell energies (4.3) and are also shown in figure 3.17) to calibrate the $p^{(l)}$'s, we create the random graph model.

Considering a graph G in this model, let us find what the probability for it to occur is. First note that the cross-edges $e_{ij}^{(l)}$ have become Bernoulli random variables (analogous to a sequence of independent coin flips with a possibly unfair coin). Being independent and identically distributed within a shell l, they take on values 1 (link present) with probability $p^{(l)}$ and 0 (no link present) with probability $1 - p^{(l)}$. The probability for the subgraph $G^{(l)}$, that constitutes the shell l of G, is then given by a Bernoulli product measure:

$$P(G^{(l)}) = \prod_{\langle i,j \rangle} p_{ij}^{(l)} (1 - p_{ij}^{(l)}) = (p^{(l)})^{e^{(l)}} (1 - p^{(l)})^{n^{(l)} - e^{(l)}} \tag{4.8}$$

The probability for the total graph is then:

$$P(G) = \prod_{l} P(G^{(l)}), \tag{4.9}$$

From the product above, it may seem that we have a collection of independent subgraphs $G^{(l)}$, and as we extract the whole array of connection

probabilities for each shell from simulation, this is indeed the case. If we take a closer look however, the $p^{(l)}$'s are not independent and one should be able to find them all from $p^{(0)}$. We will come back to this point in section 4.3.

For the reason of comparing the model to the Ising model, we would like to calculate the corresponding entropy and free energy. Knowing the occurence probability, we can calculate the Gibbs entropy:

$$S(G) = -\sum_{G} P(G) \ln P(G) \tag{4.10}$$

which by (4.9) is equal to

$$S(G) = \sum_{l} S^{(l)} \tag{4.11}$$

with $S^{(l)} = -\sum_{G^{(l)}} P(G^{(l)}) \ln P(G^{(l)})$ the entropy for shell $G^{(l)}$. The latter can be quite easily enumerated, as each graph with the same number of links has the same weight, and the number of graphs with the same number of links is a binomial coefficient,

$$S^{(l)} = -\sum_{e^{(l)}=0}^{n^{(l)}} \binom{n^{(l)}}{e^{(l)}} (p^{(l)})^{e^{(l)}} (1-p^{(l)})^{n^{(l)}-e^{(l)}} \left[e^{(l)} \ln p^{(l)} + (n^{(l)} - e^{(l)}) \ln (1-p^{(l)}) \right] \tag{4.12}$$

The graph model we have introduced is a variant of what is known as an Erdös-Rényi random graph [50, 51], named after its authors and independently introduced by Gilbert [52]. When such a model is described by the connection probability and the measure of the form (4.8), there is no notion of a free energy. There is however, a completely equivalent description of a classical random graph model, which is more akin to statistical mechanics. That is, our random graph model can be recovered from a so-called *exponential random graph model* [53]. The exponential random graph can be derived by the *maximum entropy principle*, a method developed in information theory–more specifically statistical inference–to uncover the 'least biased' probability distribution for a system, given partial information in the form of averages. Let us go over these two concepts by using our random graph model as an example in what follows.

Observe that in contrast to the RGb model, all possible graphs that one can think of on the tree enhanced with cross-edges, occur with nonzero

probability. For instance, the tree with no cross-edge found at its boundary, but *with* cross-edges in its interior, is now (though with low probability) part of the ensemble. This is a reflection of the fact that the model consists of all possible trees enhanced with cross-edges, where the only condition we impose, is that the *number of cross-edges* in each layer will on average be a given value. Apart from this constraint, the probability distribution $P(G)$ is chosen to be 'as uniform as possible'. The above statement is made mathematically rigorous by the maximum entropy approach. It states that such a probability distribution is found by maximizing the entropy (4.11) subject to the constraints

$$\sum_G P(G)e^{(l)}(G) = \left\langle e^{(l)} \right\rangle \quad \forall l \tag{4.13}$$

and the normalization condition

$$\sum_G P(G) = 1 \tag{4.14}$$

Maximizing a function (or in this case a functional) subject to constraints can be done by the method of Lagrange multipliers:

$$\frac{\partial}{\partial P(G)} \left[S + \alpha \left(1 - \sum_G P(G) \right) + \sum_l \theta^{(l)} \left(\left\langle e^{(l)} \right\rangle - \sum_G P(G)e^{(l)}(G) \right) \right] = 0 \tag{4.15}$$

for all graphs G. The parameters $\theta^{(l)}$ and α are the Lagrange multipliers tuning the sensitivity to the constraints (4.13) and (4.14) respectively. Equation (4.15) leads to

$$1 + \ln P(G) + \alpha + \sum_l \theta^{(l)} e^{(l)}(G) = 0 \tag{4.16}$$

And after rearranging, the solution is

$$P(G) = \frac{1}{Z} e^{-H(G)} \tag{4.17}$$

with

$$H(G) = \sum_l \theta^{(l)} e^{(l)}(G) \tag{4.18}$$

and

$$Z = e^{1+\alpha} = \sum_G e^{-H(G)} \tag{4.19}$$

One immediately recognizes these as analogs of statistical mechanics, i.e. $H(G)$ is the graph Hamiltonian and Z is the partition function. The probability distribution (4.17) defines the exponential random graph and is analogous to the Boltzmann distribution for the microstates of a physical system. In fact, the Boltzmann distribution can be derived by the same means–maximizing the entropy with respect to contraints that one lays on the observable canonical averages (e.g. total energy). Hence, there is an information-theoretic viewpoint, as especially promoted by Jaynes [54, 55], purporting that statistical mechanics is just a particular example of a more general inference problem within information theory.

The free energy for the exponential random graph model is defined by:

$$F = -\ln Z \tag{4.20}$$

Note that as $e^{(l)}(G)$ is in reality only a function of $G^{(l)}$, the Hamiltonian is actually

$$H(G) = \sum_l H^{(l)} \text{ with } H^{(l)} = \theta^{(l)} e^{(l)} \tag{4.21}$$

and the partition function factors out:

$$Z = \prod_l Z^{(l)} \tag{4.22}$$

with

$$Z^{(l)} = \sum_{G^{(l)}} e^{-\theta^{(l)} e^{(l)}} = \prod_{\langle i,j \rangle} \sum_{e_{ij}^{(l)}=0}^{1} e^{-\theta^{(l)} e_{ij}^{(l)}} = (1 + e^{-\theta^{(l)}})^{n^{(l)}} \tag{4.23}$$

Hence:

$$F = \sum_l F^{(l)} \tag{4.24}$$

where

$$F^{(l)} = -\ln Z^{(l)} = -n^{(l)} \ln (1 + e^{-\theta^{(l)}}) \tag{4.25}$$

Finally, note that from $\left\langle e^{(l)} \right\rangle = -\frac{\partial \ln Z}{\partial \theta^{(l)}}$ and $\left\langle e^{(l)} \right\rangle = n^{(l)} p^{(l)}$ one can extract the connection probability from θ or vice versa:

$$p^{(l)} = \frac{1}{1 + e^{\theta^{(l)}}} \tag{4.26}$$

One can check that by this relation, (4.9) is recovered from (4.17), showing the equivalence between the two descriptions presented here.

4.3 Comparison to the 1D Ising model

In order to compare our (exponential) random graph model with the Ising model, it is important to identify which parameters and functions correspond with each other. First of all, note that the inverse temperature is incorporated in the definition of the network Hamiltonian (4.18). Hence, $H(G)$ and F should be compared to the Ising functions βH_I and βF_I respectively. Looking at (4.1) and (4.21), one sees that the change in energy due to having a link (misaligment of nearest neighbour spins) for the Ising model is $\beta \Delta H_I = 2\beta J$, while it is $\Delta H = \theta$ for the random graph model. This means that the parameters are related like so $\theta^{(l)} \sim 2\beta J^{(l)}$. In particular, working in units where $J = 1$ and $k_B = 1$, we have $\theta^{(0)} \sim (2\beta J^0 = 2\beta)$, thus the equivalent parameter for the temperature $T = \beta^{-1}$ in the random graph model is identified as $T = 2(\theta^{(0)})^{-1}$. Finally, there is a difference in offset (given by the number of nearest neighbour pairs) with the energy and consequently with the free energy. Putting all of this together, the network energies and free energies correspond to their (renormalized) Ising counterparts as:

$$\begin{aligned} H_I^{(l)} &\sim -n^{(l)} + \frac{2}{\theta^{(0)}} H^{(l)} \\ F_I^{(l)} &\sim -n^{(l)} + \frac{2}{\theta^{(0)}} F^{(l)} \end{aligned} \tag{4.27}$$

Besides the differences in definitions, note that there is also a difference in size. As we are interested in a purely thermodynamic comparison, we will consider the thermodynamic functions per degree of freedom. For the Ising model, the number of degrees of freedom is the number of spins N, which is equal to the lattice size L, as we are in 1D. We will use L to characterize the size of both models. With the random graph model, the number of degrees of freedom in each shell l is the number of cross-edge variables (number of bonds), which is found from L by $n^{(l)} = L/2^l$. So, the total number of degrees of freedom for the random graph model is $n = \sum_l n^{(l)} = L + L/2 + L/4 + ...$

Having extracted the $p^{(l)}$'s–or equivalently the $\theta^{(l)}$'s–from simulation of the 1D Ising model and the subsequently generated RGb networks via (4.7), we used (4.12) to calculate the entropy $S^{(l)}$ and (4.25) to calculate the free energy $F^{(l)}$. As shown in figure 4.2 and 4.3, the entropy and free energy of the boundary shell $l = 0$ correspond pretty much exactly to the Ising entropy and free energy respectively. It is a reminder that the 1D Ising model can be described as a system of independent biased coin

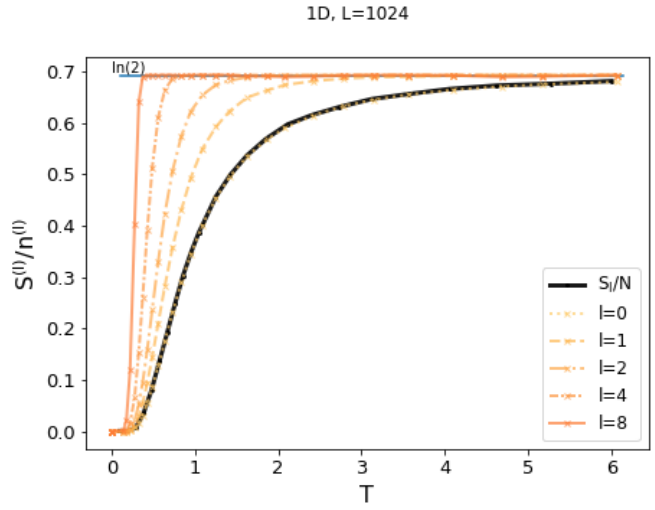


Figure 4.2: *Entropies $S^{(l)}$ as given by (4.12), are shown as a function of temperature $T[J/k_B]$ for some shells. Also shown is the Ising entropy S_I as a function of T. With the MC simulation of the 1D Ising model and the correspondingly constructed networks, a lattice size of $L = 1024$ and $m = 2.5 \times 10^4$ samples at each T was used.*

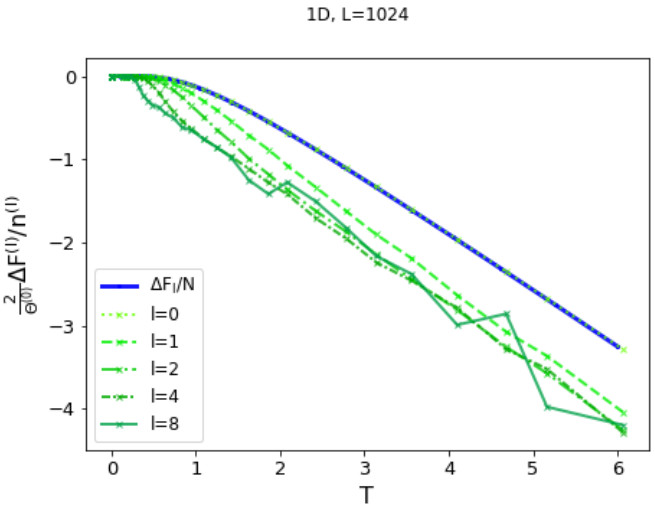


Figure 4.3: *Change in free energies, where $F^{(l)}$ is given by (4.25), are shown as a function of $T[J/k_B]$ for some shells. Also shown is the change in the Ising free energy ΔF_I as a function of T. With the MC simulation of the 1D Ising model and the correspondingly constructed networks, a lattice size of $L = 1024$ and $m = 2.5 \times 10^4$ samples at each T was used.*

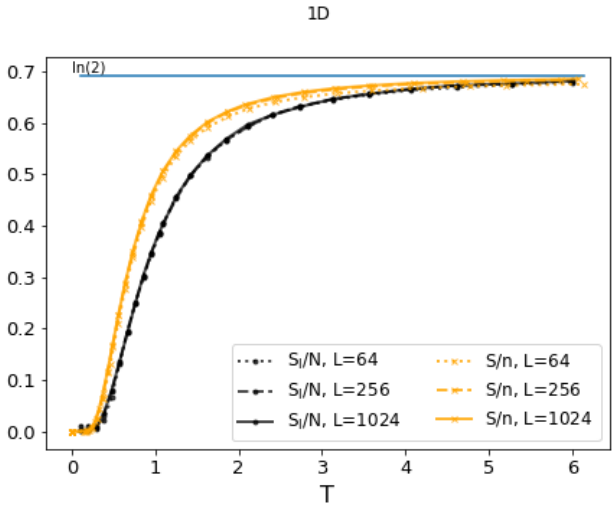


Figure 4.4: *Random graph (yellow) and 1D Ising (black) entropy as a function of temperature $T[J/k_B]$. The simulation of the Ising model and the subsequent RGb mapping, was carried out with various system sizes L. At each T, $m = 2.5 \times 10^4$ samples were used.*

tosses, i.e. one can replace the coupled spin variables and work with 'bond variables' $\sigma_i\sigma_{i+1}$, which are independent in 1D. The bond variables are essentially the cross-edge variables of our network model. In the free energy plot, the change in free energy ΔF is shown, so that the results are not obscured by the difference in offset–irrelevant to the thermodynamics. For $l > 0$, $S^{(l)}$ and $F^{(l)}$ start to deviate as functions of T, in accordance with the higher fraction of cross-edges that are found as one moves along the inner shells towards the root, which in turn can be traced back to the RG flow of the 1D Ising model. Whether or not this contribution from the interior is enough to let the entropy (4.11) and free energy (4.24) of the *whole* graph deviate significantly from the original Ising model, can be found in figure 4.4 and figure 4.5. We observe that they differ from the corresponding Ising functions. As discussed in subsection 3.3.1, the bulk-to-boundary ratio is $1:1$. Thus, the contribution to the free energy coming from the inner shells is still one half of the whole, and apparently enough to make it differ slightly from the Ising free energy.

As mentioned earlier, the parameters $\theta^{(l)}$ are not independent. We have now seen that they correspond to $2\beta J^{(l)}$. Hence, all $\theta^{(l)}$ for $l > 0$ are ultimately functions of $\theta^{(0)}$, which is the true and only parameter of the random graph model, and their relation is given by the RG flow. In figure 4.1 we saw how $J^{(l)}$, obtained explicitly from the block-spins of the RGb

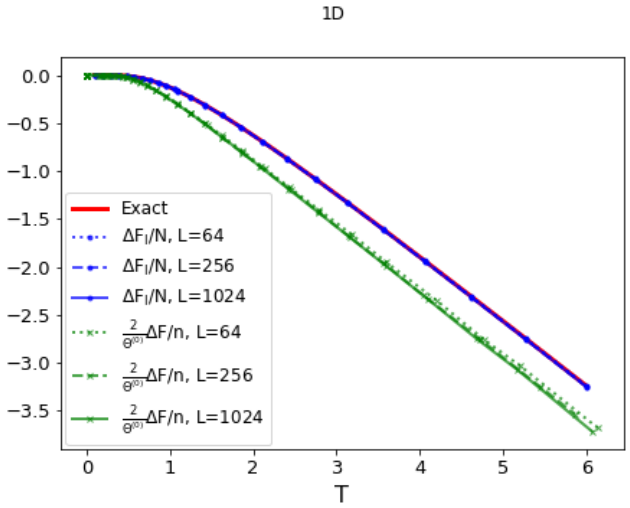


Figure 4.5: *Random graph (green) and 1D Ising model (blue) change in free energy as a function of temperature $T[J/k_B]$. Also shown is the exact value (red) for the Ising model. The Ising model simulation and subsequent RGb mapping was done for various sizes L, and $m = 2.5 \times 10^4$ samples were used at each T.*

networks , agrees reasonably well with the RG trajectory as given by a decimation scheme [49]. Making the same comparison in figure 4.6 for the ratio's $\theta^{(l)}/\theta^{(0)}$, we find that they are more in agreement with the theoretical result than the $J^{(l)}$'s. This came a bit as a surprise to the present author. Though both $J^{(l)}$ and $\theta^{(l)}$ are obtained from the average energy/number of cross-edges, it seems that $\theta^{(l)}$ does not suffer from finite size effects of the simulation. Anyhow, it is convenient, as it shows that $\theta^{(l)}$'s can be given by the recursive relation (4.5), which from the network perspective is then seen as a radial equation. This would make the random graph model complete and truly independent, as one does not need the Ising-RGb simulation anymore to extract the parameters.

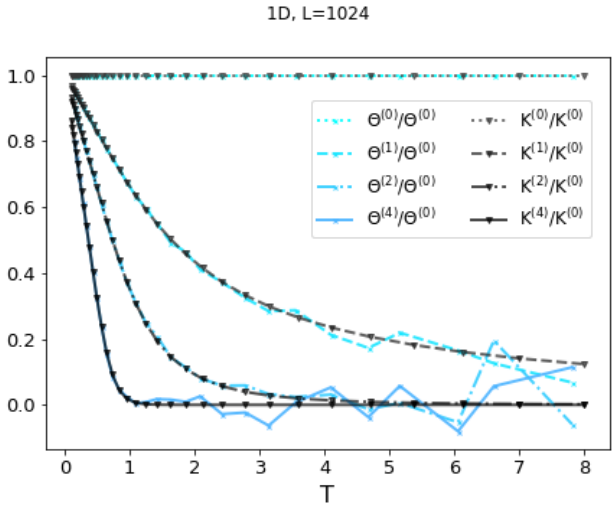


Figure 4.6: *The ratio $\theta^{(l)}/\theta^{(0)}$ as a function of T are shown for some shells. They are compared to the equivalent ratio, obtained from a decimation scheme of the 1D Ising model [49]. The $\theta^{(l)}$'s were extracted from the average number of cross-edges in the shell, as found from the RGb simulation. The lattice had size $L = 1024$ and for each T, $m = 2.5 \times 10^4$ samples were used.*

4.4 Extension to 2D

We have straightforwardly extended the random graph model to 2D (dimension of the boundary). Thus, from simulation of the Ising model, now on a $N = L \times L$ lattice of spins, and the subsequently generated RGb networks, we have extracted the connection probabilities $p^{(l)}$ (and thus also $\theta^{(l)}$) using (4.7). Note that there are now two cross-edge variables for each spin. Hence the number of degrees of freedom in a shell for the random graph model is $n^{(l)} = 2N^{(l)}$, where $N^{(l)} = L^2/4^l$ (branching factor is 4) is the number of spins found in shell l. The number of degrees of freedom for the whole network is then $n = \sum_l n^{(l)} = 2(L^2 + L^2/4 + L^2/16 + ...)$.

Figure 4.7 shows the entropy of each shell (4.12) and figure 4.8 shows the free energy of each shell (**??**). Notice that the entropy and the free energy of shell $l = 0$, i.e. the boundary, already does not correspond with that of the Ising model. It then comes as no surprise that the total entropy (4.11) and free energy (4.24) differs significantly from the 2D Ising model, as shown in figure (4.9) and (4.10) respectively.

So why is it that the boundary of our random graph model is practically identical to the Ising model in 1D, but not in 2D? As alluded to earlier, the 1D Ising model can actually be rewritten as a system of independent

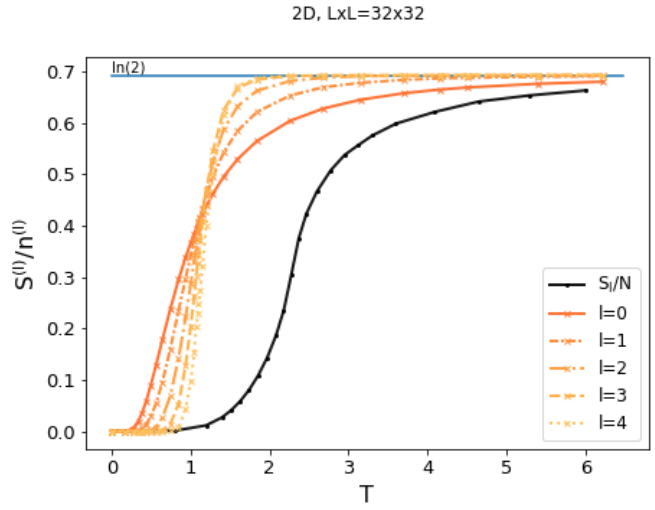


Figure 4.7: *Entropies $S^{(l)}$ as given by (4.12), are shown as a function of temperature $T[J/k_B]$ for some shells along with the Ising entropy S_I. With the MC simulation of the 2D Ising model and the correspondingly constructed networks, a lattice size of $L \times L = 32 \times 32$ and $m = 2.5 \times 10^4$ samples at each T was used.*

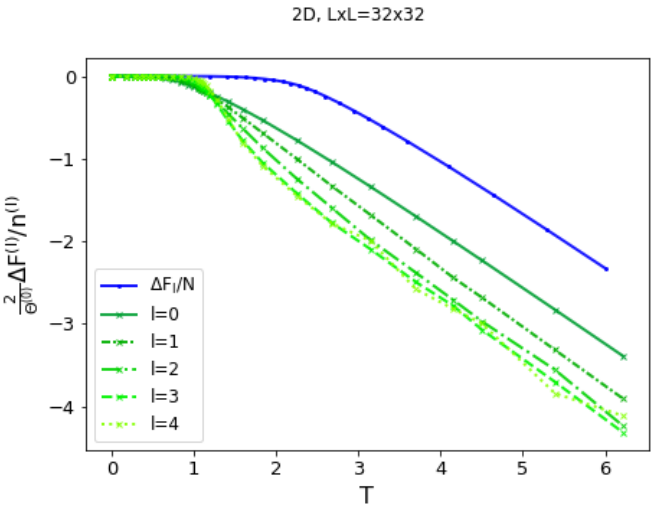


Figure 4.8: *Change in free energies, where $F^{(l)}$ is given by (4.25), are shown as a function of $T[J/k_B]$ for some shells along with the change in the Ising free energy ΔF_I. With the MC simulation of the 2D Ising model and the correspondingly constructed networks, a lattice size of $L \times L = 32 \times 32$ and $m = 2.5 \times 10^4$ samples at each T was used.*

bond variables. *While the spins $\{\sigma_i\}$ are obviously coupled* by the nearest

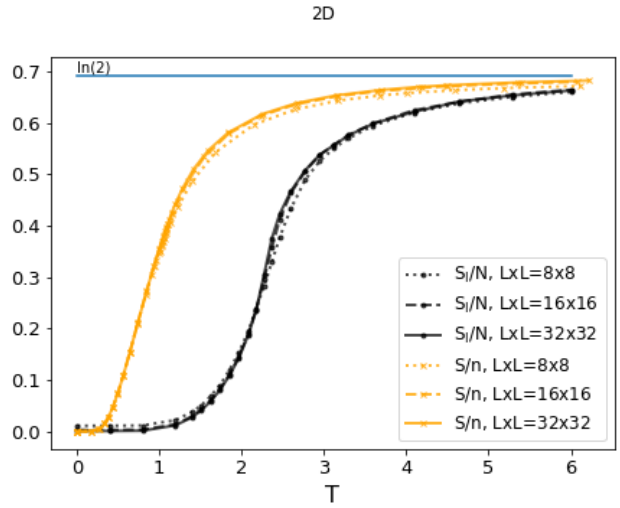


Figure 4.9: *Random graph (yellow) and 2D Ising (black) entropy as a function of temperature $T[J/k_B]$. The simulation of the Ising model and the subsequent RGb mapping, was carried out with various system sizes L. At each T, $m = 2.5 \times 10^4$ samples were used.*

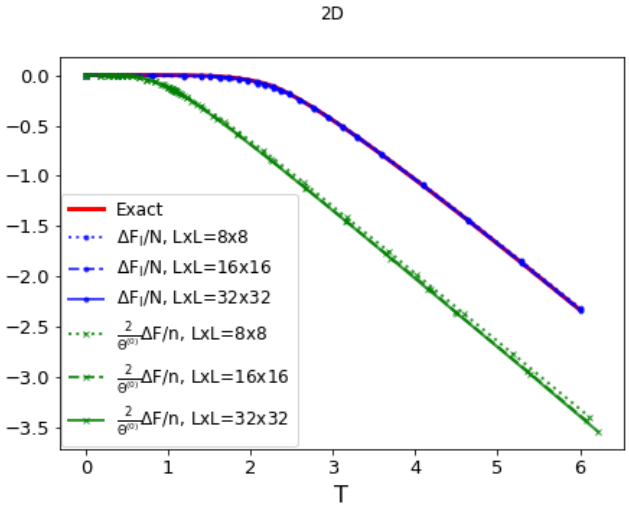


Figure 4.10: *Random graph (green) and 2D Ising model (blue) change in free energy as a function of temperature $T[J/k_B]$. Also shown is the exact value (red) for the Ising model. The Ising model simulation and subsequent RGb mapping was done for various sizes L, and $m = 2.5 \times 10^4$ samples were used at each T.*

neighbour Hamiltonian, *the bond variables $\{\sigma_i\sigma_j\}$ are independent in 1D*. In 2D this is not the case, as due to the increased number of neighbours per spin, the occurence of a cross-edge (anti-aligned bond) fixes the presence

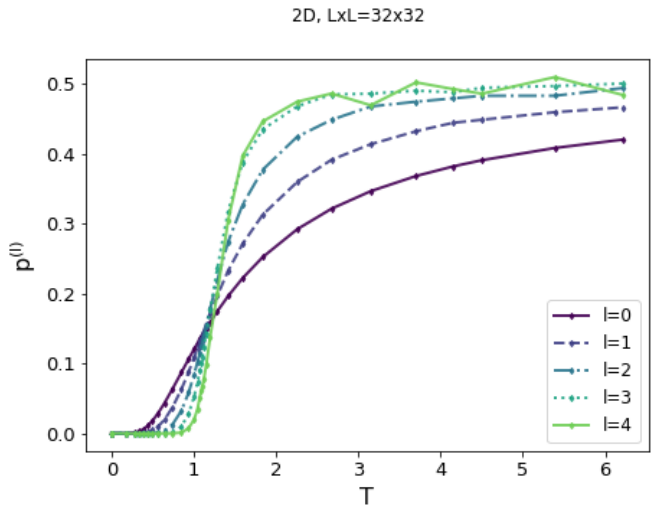


Figure 4.11: *Connection probabilities of the random graph model, as given by (4.7), are shown as a function of temperature $T[J/k_B]$ for some shells. With the MC simulation of the 2D Ising model and the correspondingly constructed RGb networks, a lattice size of $L \times L = 32 \times 32$ and $m = 2.5 \times 10^4$ samples at each T was used.*

of neighbouring cross-edges. Thus, a graph model with independent bond variables cannot accurately describe the Ising model. We can also see this by simply considering the number of degrees of freedom. The number of cross-edges is twice the number of spins for the 2D case. As a model with independent d.o.f. cannot describe a system with half the number of d.o.f., it is natural that the boundary of our random graph model corresponds poorly to the 2D Ising model.

Though it is clear that the random graph model does not inherit the Ising phase transition, we do expect it to be very sensitive to which phase the Ising model is in. The inner shells of the RGb networks, in a way, 'exaggerate' the phase in which the Ising model on the boundary is in. That is, in the low temperature phase, the block-spins in the interior are characteristic of the boundary Ising model with an even lower temperature (or stronger coupling) and it is the opposite for the high temperature phase. We do see, especially from the entropy of each shell in figure 4.7, that this effect is reflected onto the random graph model. One would think however, that the point where the system looks the same in all shells is at the critical temperature of the Ising model $T_c \approx 2.26$, as then the fraction of cross-edges in the RGb networks should be the same for all shells. We observe that the shell entropies coincide already for lower temperature

$T \approx 1.2$. This can also be clearly seen from the connection probabilities, displayed in figure 4.11. It would be interesting to know why the self-similarity point of our random graph model lies where it does.

4.4.1 Outlook

Suppose we could recreate the situation found in 1D for the random graph model in 2D. In particular having the boundary correspond exactly to the Ising model. Then we might find that the thermodynamic functions of the whole network are much closer to their Ising counterparts than was found in 1D. We expect this because of the smaller deviating contribution coming from the inner shells, due to the larger branching factor ($b = 4$ as opposed to $b = 2$).

Finding a graph model that is equivalent to the Ising model for all dimensions (even generalized for the Potts model) has been done in the early 70's by Fortuin and Kasteleyn [56, 57]. The probability for a planar graph in the Fortuin-Kasteleyn model, which also goes by the name random cluster (RC) model, is similar to that of the Bernoulli product measure we saw in (4.8):

$$P(G^{(l)}) = \frac{1}{Z_{RC}} q^{k(G^{(l)})} \prod_{\langle i,j \rangle} p^{(l)} (1 - p^{(l)}) \tag{4.28}$$

where $p^{(l)}$ denotes again the connection probability and is the parameter of the model*. Notice that there is an extra factor $q^{k(G^{(l)})}$. For our purposes $q = 2$, as then it corresponds to the Ising model (in general q is a positive real that determines which model the RC model is equivalent to, e.g. for $q \in \{2, 3, ..\}$ it corresponds to the q-state Potts model). The exponent $k(G^{(l)})$ denotes the number of connected components (clusters) in $G^{(l)}$. Therefore, the factor $q^{k(G^{(l)})}$ couples the cross-edge variables $e_{ij}^{(l)}$. The normalizing factor Z_{RC} is the model's partition function:

$$Z_{RC} = \sum_{G^{(l)}} \left\{ q^{k(G^{(l)})} \prod_{\langle i,j \rangle} p_{ij}^{(l)} (1 - p_{ij}^{(l)}) \right\} \tag{4.29}$$

Carrying on as before, we would create the random graph model by

*We have kept using the notation $\langle i, j \rangle$, which refers to spin i and j to denote the corresponding bond variable. Ofcourse, as the spins are not needed to define the RC model, it is perhaps more appropiate to start indexing the bonds themselves.

setting the connection probability for each shell equal to the value found from simulation of the RGb networks. Or alternatively, it is known that the connection probability is related to the parameter of the Ising model as: $p^{(l)} = e^{-2\beta J^{(l)}}$. However, due to the inclusion of the factor $q^{(k(G^{(l)})}$, graphs with the same number of cross-edges $e^{(l)} = \sum_{\langle i,j \rangle} e_{ij}$, are no longer measured with the same weight. Thus, simple numerical enumeration using binomial factors is no longer possible. If one would want to study the resulting random graph model numerically, one probably would need to use a MC simulation of the RC model for each shell.

In light of the RC model being the equivalent graph representation of the Ising model in any dimension, one might wonder how it is that our random graph model, with independent cross-edge variables, was found to correspond to the 1D Ising model already so well. Should not the RC model then be the same as our random graph model when the planar graph is a one-dimensional chain? In fact, this is indeed the case. In 1D, the number of clusters $k(G^{(l)})$ is equal to the number of cross-edges (pairs of misaligned spins), as we have emphatically learned from our AoSD mapping scheme (section 3.2). Hence the factor q^k becomes $q^{e^{(l)}}$, and (4.28) reduces to a Bernoulli product measure equivalent to the one in (4.8), where the cross-edges are independent variables.

Instead of focussing on the thermodynamics of the network itself, we could also take a different approach. Our aim is to find a network model that encaptures *a boundary dimensional phase transition*, not nessecarily *the Ising phase transition*, in its higher dimensional structure. Thus, we could also for example study percolation on the networks of our random graph model. It would be interesting to find a percolation phase transition specific to 2D, in the seemingly higher-dimensional networks. Care should be taken that one is not finding trivial results, as in its present form, the edges only vary in 2D within the shells, i.e. the edges in the radial direction are fixed.

Chapter 5

Conclusion

Inspired by the AdS/CFT correspondence, we have incorporated some key ideas, such as the geometrization of the RG formalism and having a hyperbolic background into the construction of a network model. The goal was to see if by merely adopting these concepts one can create a statistical mechanical-network system that exhibits boundary specific thermodynamics. In particular, we aimed at encapturing the 2D Ising phase transition in a higher dimensional network model.

After offering a review of the Ising model on the Cayley tree–a network equivalent of hyperbolic space–in chapter 2, where by self-similarity arguments the model can be transformed into a Bethe lattice, we dicussed mappings of the 1D and 2D Ising model onto a network model in chapter 3. With our AoSD mapping, we found in 1D that although weighted with a Hamiltonian that was equal to the boundary (Ising) hamiltonian plus an added contribution from the bulk, the thermodynamics remained virtually the same (the same as when it was weighted with only the boundary Hamiltonian). In 2D, the variation of the Ising energy was found not to be reflected into the topology of the network. Therefore, we developed another construction procedure, the RGb mapping. The same holographic effect as was shown for the 1D AoSD mapping was demonstrated, though to a lesser extent and with a caveat (the quantitative difference between network and boundary Hamiltonian was ignored).

In chapter 4 we formulated a(n) (exponential) random graph model to create an ensemble of networks that resembles the one seeded of the Ising model by the RGb mapping independently. In 1D (dimension of the Ising model/boundary), the boundary of the random graph model was found to correspond exactly to the Ising model. The entropy per degree of freedom was found to be bit higher than its Ising counterpart, due to the con-

tribution of the interior. We expected the entropies to correspond better in 2D due to the higher boundary-to-bulk ratio, but the boundary graph description already did not correspond to the Ising model. The 1D Ising model, can be descibed by independent 'bond variables', which is not true for the model in any dimension higher. If one wants to create the same situation one has in 1D, we suggest to use the random cluster model to descibe each shell of the RGb network, as it is a planar graph model that is equivalent to the Ising model in any dimension. An alternative move forward is to study percolation, on the random graph model with the 2D boundary. Perhaps it displays a 2D specific percolation phase transition in the higher dimensional network model.

Appendices

Appendix A

Simulation Routines and Code

A.1 Metropolis Algorithm

The Metropolis algorithm [23] is an algorithm that can be used to implement transitions in the Monte Carlo (MC) simulation of the Ising model. It proceeds in two stages: 1) given a state X, a trial state X' is proposed with trial probability $W(X \to X')$, 2) the trial state X is accepted to be the new state with acceptance probabiltiy $A(X \to X')$. The transition probability then reads

$$T(X \to X') = W(X \to X')A(X \to X') \tag{A.1}$$

and the trial probabilities are required to meet the following conditions

$$W(X \to X') > 0, \tag{A.2}$$
$$W(X \to X') = W(X' \to X), \tag{A.3}$$
$$\sum_{X'} W(X \to X') = 1, \tag{A.4}$$
$$\text{for all } X, X'$$

Note that the symmetry requirement A.3 turns the detailed balance condition 3.11 into an equation for the acceptance probabilities:

$$\frac{A(X \to X')}{A(X' \to X)} = \frac{P(X')}{P(X)} \tag{A.5}$$

The Metropolis solution is then defined as:

$$A(X \to X') = \min\left(1, P(X')/P(X)\right) \tag{A.6}$$

So, if the trial state has a larger weight than the present state $P(X') > P(X)$, it is always accepted to be the new state; if the trial state has a smaller weight than the present one $P(X') < P(X)$, it is accepted with probability $P(X')/P(X)$.

Using the Metropolis method to simulate the Ising model, the first stage of the move is realized by selecting a spin at random from the current configuration σ and taking the configuration with this selected spin flipped as the trial configuration σ'. Therefore the trial probabilities are chosen to be:

$$W(\sigma \to \sigma') = \begin{cases} 1/N, & \text{if } \sigma \text{ and } \sigma' \text{ differ by a single spin flip} \\ 0, & \text{otherwise} \end{cases} \tag{A.7}$$

As the weights are given by: $P(\sigma) \propto e^{-\beta E(\sigma)}$, A.6 boils down to always accepting the trial configuration if the energy is lowered or remains the same, $\Delta E(\sigma \to \sigma') = E(\sigma') - E(\sigma) \leq 0$. If the proposed transition increases the energy $\Delta E(\sigma \to \sigma') > 0$, then the trial configuration is accepted with probability $e^{-\beta \Delta E(\sigma \to \sigma')}$. This last step is realized in practice by generating a random number $r \in Uniform[0,1]$, and accepting the trial state if $r < e^{-\beta \Delta E(\sigma \to \sigma')}$.

Besides obviously satisfying the detailed balance condition, the Metropolis algorithm aslo satisfies the ergodicity requirement. Note that in principal each configuration can be obtained from any other configuration by an appropriate succession of single spin flips. As the Metropolis algorithm applied to the Ising model consists of single spin flips and there is no periodicty in the generated configurations, one can conclude that it is ergodic.

A.2 Wolff Algorithm

The Wolff algorithm [24] is a cluster algorithm that can be used to implement transititions in the MC simulation of the Ising model. A cluster of spins is constructed and updated at each move. The move is defined as follows:

1. From the current configuration of spins σ, select a spin σ_0 at random to be the first spin of the cluster C.

2. Next consider its nearest neighbours. Only if they have the same spin value, add to the cluster with a probability P_{add} (and thus exclude equal spins with probability $1 - P_{add}$).

3. Repeat 2 for each spin added to the cluster, visiting all nearest neighbours that are not already in the cluster.

4. The construction process stops automatically when no more spins are added. All spins in the cluster $\sigma_i \in C$ are flipped, which yields a new spin configuration σ' that is always accepted.

Setting

$$P_{add} = 1 - e^{-2\beta J} \tag{A.8}$$

ensures that the transition $\sigma \to \sigma'$ as defined above satisfies the detailed balance condition 3.12:

$$\frac{T(\sigma' \to \sigma)}{T(\sigma \to \sigma')} = e^{-\beta(E(\sigma)-E(\sigma'))} \tag{A.9}$$

To see this, first note that the configurations σ and σ' differ from each other only by the flip of a single cluster, see figure A.1 for an example. Let m be

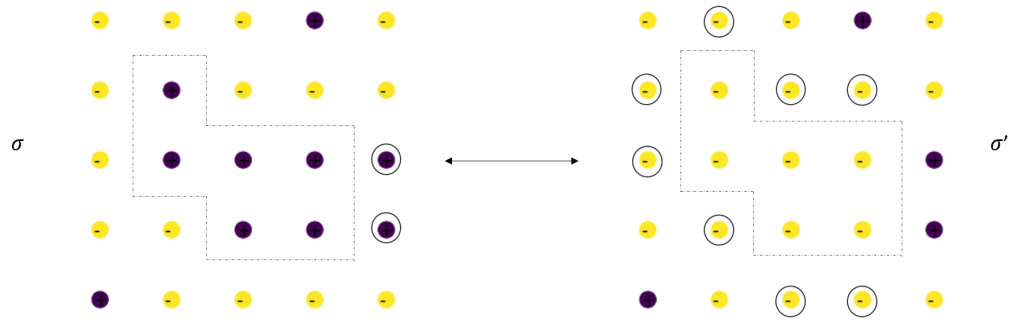


Figure A.1: *Example of two configurations of spins σ and σ' on the 2D lattice that differ by one cluster flip of the Wolff algorithm. The dashed line indicates the boundary of the cluster that is comprised of equal spins. The encircled spins- though eligible to be part of the cluster, they are not.*

the number of nearest neighbour pairs $\langle i, j \rangle$ (bonds) of a spin in the cluster $\sigma_i \in C$ and an *equal* spin outside the cluster $\sigma_j \notin C$ for configuration σ. Let n be the same quantity for configuration σ' (in the example of figure A.1 $m = 2$ and $n = 10$). Then observe that the transistion probability resulting from the above procedure for the move $\sigma \to \sigma'$ has a lot of factors in common with the transition probability for the reverse move $\sigma' \to \sigma$, e.g. selection probability for the seed spin ($1/N$ for a system with N spins)

and the probability to have the same spins in the cluster($P_{add}^{|C|-1}$). Actually, they only differ by factors coming from excluding spins: $T(\sigma \to \sigma') \propto (1 - P_{add})^m$ and $T(\sigma' \to \sigma) \propto (1 - P_{add})^n$. So, the ratio in the detailed balance equation becomes:

$$\frac{T(\sigma' \to \sigma)}{T(\sigma \to \sigma')} = (1 - P_{add})^{n-m} = e^{-\beta(E(\sigma)-E(\sigma'))} \tag{A.10}$$

Now, for the Ising model the energy increases $+2J$ for each aligned nearest neighbour pair that becomes misaligned, while vice verse it decreases by $-2J$. The change in energy is then precicely given by $E(\sigma) - E(\sigma') = (n - m)2J$. And so one has:

$$(1 - P_{add})^{n-m} = e^{-2\beta J(n-m)} \tag{A.11}$$

from which A.8 follows.

As a sidenote, the fact that the quantities m and n drop out of the detailed balance condition makes P_{add} independent of the configuration of spins. Herein lies the difficulty in applying the Wolff algorithm to a model with a different energy function than that of nearest neighbour interaction. For a general energy function m and n does not drop out and one would need to determine them to calculate P_{add}. The Wolff algorithm would effectively then reduce to a local update algorithm, losing its advantageous features.

Besides the detailed balance condition, the Wolff algorithm also satisfies the ergodicity criterion. Observe that in principal, each spin configuration can be obtained from any other spin configuration by a finite number of successive single spin flips. As there is a non-zero chance that the cluster in the Wolff move only consists of a single spin, there is a finite probability that given a configuration any other configuration can be obtained by a particular succession of Wolff moves. Also, there are no periodicities in the sequence of configurations generated by the Wolff algorithm.

It is natural to program the Wolff algorithm by means of recursion. We have adopted a recursive routine from [25]. Let $\vec{x} = (x_1, x_2, .., x_d)$ be the location of a spin $\sigma_{\vec{x}}$ in the d-dimensional lattice. Then after randomly selecting a seed spin, its spin value σ_0 is stored and the rest of the routine for the MC step is given in pseudocode as follows:

Function `GrowCluster(`$\vec{x}$, σ_0`)`**:**
Flip $\sigma_{\vec{x}}$
Mark $\sigma_{\vec{x}}$ as added to Cluster

Consider neighbouring spin $\sigma_{\vec{y}}$ at $y = (x_1 + 1, x_2, ..., x_d)$
if $\sigma_{\vec{y}}$ *at is not already part of the Cluster* **then**
`TryAdd(`$\vec{y}$, σ_0`)`
end

...do the same for all other neighbours $(x_1, x_2 + 1, ..., x_d)$..etc...
end

Function `TryAdd(`$\vec{x}$, σ_0`)`**:**
if $\sigma_{\vec{x}}$ *equal to* σ_0 **then**
Generate random number $r \in Uniform[0, 1]$
if $r < 1 - e^{-2\beta J}$ **then**
`GrowCluster(`$\vec{x}$, σ_0`)`
end
end
end

A.3 Spin Lattice Domain Decomposition Routine

In order to decompose the lattice of spins into its domains for our AoSD mapping procedure, we have adopted a routine presented in [25], where it is given to implement the Swendsen-Wang algorithm (another algorithm that can be used to simulate e.g. the Ising model). It identifies all the clusters of a d-dimensional lattice by 'backtracking' in a recursive way as follows. Considering a spin site $\vec{x} = (x_1, x_2, .., x_d)$, it is marked as visited and added to the cluster. Then we consider one of its neighbour and check if it has been visited already. If not, we check if the neighbour has identical spin value. If so, then we mark the neighbour as visited, add it to the cluster and consider *its* neighbour similarly. By repeating these steps recursively for all the neighbours and neighbours of neighbours etc., the cluster of the starting spin is identified. The routine to identify a cluster of spins is given in pseudocode below. Here, `VisitedMap(`$\vec{x}$`)` is a boolean map of the lattice which keeps track of which sites have already been visited. `FrozenMap(`$\vec{x}$, $\vec{y}$`)` is a boolean map of all nearest neighbour bonds, i.e. it returns $TRUE$ if the spins at $\vec{x}$ and $\vec{y}$ have equal spin and $FALSE$ if

```
Function BackTrack(x⃗):
    if VisitedMap(x⃗) is FALSE then
        Set VisitedMap(x⃗) to TRUE

        Consider neighbouring spin at y = (x_1 + 1, x_2, ..., x_d)
        if FrozenMap(x⃗, y⃗) then
            BackTrack(y⃗)
        end

        ...do same for all other neighbours (x_1, x_2 + 1, ..., x_d)..etc...
    end
end
```

otherwise.

Letting the backtrack routine start with every site in the lattice, all clusters are scanned.

A.4 Source Code

All of the code for the simulations in this thesis were programmed in Jupyter Notebooks with Python. Below, one can find the links to the full source code of the various simulations, which are stored on github. Apart from these, for some simulations (e.g. 1D AoSD mapping), we list some function definitions here and also provide a link to an exhibition notebook. These serve as entry-material, for the reader who wants to study the simulation in detail.

A.4.1 AoSD 1D

The full code (collection of jupyter notebooks) used for the AoSD mapping in 1D can be found here:

https://github.com/JosephSalaris/AoSD1D/tree/master/AoSD1DfullCode

An exhibition jupyter notebook of the 1D AoSD simulation, which also visualizes the generated networks, can be found here:

https://github.com/JosephSalaris/AoSD1D/tree/master/AoSD1Dexhibition

Lastly, below one finds the function definitions needed to contruct a graph from a configuration of spins in 1D by the AoSD procedure. These are at the heart of the AoSD simulation and are used in every MC step.

```
import numpy as np
import matplotlib.pyplot as plt
import networkx as nx

def create_frozen_map(lat):
    """ Creates a boolean map of (freezes) all bonds (nn interactions) in the lattice

        Input:
            lat: dtype numpy array
                [N] array containing only 1 and -1 values representing the spin configuration

        Returns:
            f_map: dtype numpy array
                [N] boolean array containing True if bond is present or False if not. For example spin lattice [1, -1, 1, 1]
                corresponds to f_map [True, True, False, False]
    """
    lat = np.array(lat)
    a = lat[1:]-lat[0:len(lat)-1]; b = lat[0]-lat[len(lat)-1]
    c = np.append(a,b)
    f_map = (c == 0)
    return(f_map)

def backtrack(x,vis,fm,cl):
    """ Part of the cluster decomposition. It recursively checks whether neighbouring spin should be added to the cluster
    """
    if not vis[x]:
        vis[x] = True
        cl.append(x)
        if fm[x]:
            vis, cl = backtrack((x+1)%len(fm),vis,fm,cl)
        if fm[(x-1)%len(fm)]:
            vis, cl = backtrack((x-1)%len(fm),vis,fm,cl)
    return(vis,cl)

def cluster_decomp(lat):
    """ Decomposes the spin lattice into clusters, i.e. creates list of cluster arrays. A cluster array contains the indices of
        the spins in a cluster

        Input:
            lat: dtype numpy array
                [N] array containing only 1 and -1 values representing the spin configuration

```

```
43          Returns:
44              cls: dtype list
45                  list of cluster arrays. Each cluster array contains
    the indices of the spins in that cluster
46      """
47      visited = np.full((len(lat)), False)
48      frozen_map = create_frozen_map(lat)
49      #zeros = np.where(np.array(lat)==0)[0]
50      #tried = dict.fromkeys(zeros,False)
51      cls = []
52      for i in range(len(lat)):
53          clu = []
54          visited, clu = backtrack(i,visited,frozen_map,clu)
55          if len(clu) > 0:
56              cls.append(clu)
57      return(cls)
58
59  def create_boundary_graph(lat):
60      """ Create boundary graph (collection of nodes) corresponding to
        the spin lattice
61
62          Input:
63              lat: dtype numpy array
64                  [N] array containing only 1 and -1 values
    representing the spin configuration
65
66          Returns:
67              bdGr: dtype networkx graph
68                  boundary graph
69      """
70      n_nodes = len(lat)
71      bdGr = nx.Graph()
72      for i in range(N):
73          bdGr.add_node(i)
74      return(bdGr)
75
76  def construct_graph(lat,bdGr):
77      """ Construct graph from spin lattice according to the AoSD
    procedure
78
79          Input:
80              lat: dtype numpy array
81                  [N] array containing only 1 and -1 values
    representing the spin configuration
82
83              bdGr: dtype networkx graph
84                  graph containing the boundary nodes (i.e. as
    constructed by the function 'create_boundary_graph')
85
86          Returns:
87              Gr: dtype networkx graph
88                  AoSD graph
89      """
```

```
90      lats = []; lats.append(lat)
91      shell = 0; n_nodes_shell = len(np.array(lats[shell])); add = 0
92      Gr = bdGr.copy()
93      CGP_step1 = True
94      while(n_nodes_shell>1):
95          new_lat = []
96          if CGP_step1:
97              # perform step 1 of the coarse grain procedure (create
        domain branches)
98              clusters = cluster_decomp(lats[shell])
99              # go through clusters
100             for i in range(len(clusters)):
101                 value = 0
102                 n_Gr = nx.number_of_nodes(Gr); Gr.add_node(n_Gr)
103                 for j in clusters[i]:
104                     value += lats[shell][j]
105                     Gr.add_edge(n_Gr,j+add)
106                 new_lat.append(value)
107             CGP_step1 = False
108         else:
109             # peform step 2 of the coarse grain procedure (average
        over domains)
110             for k in np.arange(0,n_nodes_shell,2):
111                 value = 0
112                 n_Gr = nx.number_of_nodes(Gr); Gr.add_node(n_Gr)
113                 #print('n_Gr', n_Gr)
114                 #print('k',k+add)
115                 value = lats[shell][k]+lats[shell][k+1]
116                 if value == 0:
117                     value = np.random.choice([-1,1])
118                 else:
119                     value = value/abs(value)
120                 Gr.add_edge(n_Gr,k+add); Gr.add_edge(n_Gr,k+1+add)
121                 new_lat.append(value)
122             CGP_step1 = True
123         lats.append(new_lat)
124         shell += 1
125         add += n_nodes_shell; n_nodes_shell = len(np.array(lats[
        shell]))
126     return(Gr)
127
128 N = 50 # Global parameter denoting the number of spins (boundary
        nodes)
129 lattice = np.ones(N)
130 bdG = create_boundary_graph(lattice)
131 G = construct_graph(lattice, bdG)
132 nx.draw(G)
```

A.4.2 AoSD 2D

The full code (collection of jupyter notebooks) used for the AoSD mapping in 2D can be found here:

https://github.com/JosephSalaris/AoSD2D/tree/master/AoSD2DfullCode

An exhibition jupyter notebook of the 1D AoSD simulation, which also visualizes the generated networks, can be found here:

https://github.com/JosephSalaris/AoSD2D/tree/master/AoSD2Dexhibition

A.4.3 RGb 1D

The full code (collection of jupyter notebooks) used for the RGb mapping in 1D can be found here:

https://github.com/JosephSalaris/RGb1D/tree/master/RGb1DfullCode

An exhibition jupyter notebook of the 1D RGb simulation, which also visualizes the generated networks, can be found here:

https://github.com/JosephSalaris/RGb1D/tree/master/RGb1Dexhibition

Below one finds the function definitions needed to contruct a graph from a configuration of spins in 1D by the RGb procedure. These are at the heart of the RGb simulation and are used in every MC step.

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3  import networkx as nx
4
5  def RGgraph_skeleton():
6      """ Creates boundary graph (collection of nodes) corresponding
       to the spin lattice, and the 'bare' tree graph
7
8          Returns:
9              bdG: dtype networkx multigraph
10                 boundary graph
11
12             bulkG: dtype networkx multigraph
13                 'bare' tree graph, i.e. the boundary + bulk tree
       graph that serves as a skeleton for the RGb graph
14     """
15     # Boundary graph
```

```
16      bdG = nx.MultiGraph()
17      for i in range(N):
18          bdG.add_node(i)
19      # Bulk graph
20      n_nodes_shell = N; add = 0
21      bulkG = bdG.copy()
22      while n_nodes_shell > 1:
23          for k in np.arange(0,n_nodes_shell,2):
24              n_Gr = nx.number_of_nodes(bulkG); bulkG.add_node(n_Gr)
25              bulkG.add_edge(n_Gr,k+add); bulkG.add_edge(n_Gr,k+1+add)
26          add += n_nodes_shell
27          n_nodes_shell //= 2
28      return(bdG, bulkG)
29
30  def RGlats_skeleton():
31      """ Creates the 'bare' shells of the RGb graph. That is, a list
        of spin lattices of decreasing size in accordance with the
32          branching factor 2, that serve as 'skeleton lattices' that
        can be modified by the RGb procedure
33
34          Returns:
35              lats: dtype list
36                  list of lattice arrays. They serve as the initial
        spin lattices that constitute the shells of the RGb graph
37      """
38      lats = []; lats.append(np.zeros(N))
39      n_nodes_shell = N
40      while n_nodes_shell > 1:
41          n_nodes_shell //= 2
42          new_lat = np.zeros(n_nodes_shell)
43          lats.append(new_lat)
44      return(lats)
45
46  def create_frozen_map(lat):
47      """ Creates a boolean map of (freezes) all bonds (nn
        interactions) in the lattice
48
49          Input:
50              lat: dtype numpy array
51                  [N] array containing only 1 and -1 values
        representing the spin configuration
52
53          Returns:
54              f_map: dtype numpy array
55                  [N] boolean array containing True if bond is present
         or False if not. For example spin lattice [1, -1, 1, 1]
56                  corresponds to f_map [True, True, False, False]
57      """
58      lat = np.array(lat)
59      a = lat[1:]-lat[0:len(lat)-1]; b = lat[0]-lat[len(lat)-1]
60      c = np.append(a,b)
61      f_map = (c == 0)
62      return(f_map)
```

```
63
64 def construct_graph(lat, lats, Gr):
65     """ Construct graph from spin lattice according to the RGb
       procedure
66
67         Input:
68             lat: dtype numpy array
69                 [N] array containing only 1 and -1 values
       representing the spin configuration
70
71             lats: dtype list
72                 list containing the initial spin lattices that
       constitutes the shells of the graph (i.e. as constructed by the
73                 function 'RGlats_skeleton')
74
75             Gr: dtype networkx multigraph
76                 'bare' tree graph that serves as the 'skeleton' for
       the RGb graph (i.e. as constructed by the function
77                 'RGgraph_skeleton')
78
79         Returns:
80             Gr: dtype networkx multigraph
81                 RGb graph
82     """
83     n_nodes_shell = N; shell = 0
84     add = 0
85     Gr = Gr.copy()
86     while shell < len(lats):
87         lats[shell] = lat
88         if shell == (len(lats)-1):
89             break
90
91         # Update RG graph accordingly:
92         frm = create_frozen_map(lat)
93         idx = np.where(frm==False)
94         for i in idx[0]:
95             Gr.add_edge(i+add, (i+1)%n_nodes_shell+add)
96         add += n_nodes_shell
97
98         # Do RG step:
99         old_lat = lat
100        n_nodes_shell //= 2
101        lat = lat.reshape(n_nodes_shell, 2)
102        lat = np.sum(lat, axis=1)
103        # Deal with ties:
104        idx_zeros = np.where(lat==0)
105        old_idx_zeros = tuple([2*i for i in idx_zeros])
106        lat[idx_zeros] = old_lat[old_idx_zeros] #np.random.choice
       ([1,-1], size=len(idx_zeros[0]))
107        lat = lat/abs(lat)
108
109        shell += 1
110    return(Gr)
```

```
111
112 N = 16 # Global parameter denoting the number of spins (boundary
        nodes)
113 lattice = np.ones(N)
114 bdG, initG = RGgraph_skeleton()
115 lattices = RGlats_skeleton()
116 G = construct_graph(lattice, lattices, initG)
117 nx.draw(G)
```

A.4.4 RGb 2D

The full code (collection of jupyter notebooks) used for the RGb mapping in 1D can be found here:

https://github.com/JosephSalaris/RGb2D/tree/master/RGb2DfullCode

An exhibition jupyter notebook of the 1D RGb simulation, which also visualizes the generated networks, can be found here:

https://github.com/JosephSalaris/RGb2D/tree/master/RGb2Dexhibition

Appendix B

Autocorrelation Time and Error

Correlations between successive samples generated by the Markov Chain Monte Carlo simulation can be analyzed from the autocorrelation function. Given an observable A, the autocorrelation function reads:

$$C(t) = \int dt' \left(A\left(t'\right) - \langle A \rangle\right)\left(A\left(t+t'\right) - \langle A \rangle\right) \tag{B.1}$$

As $C(t)$ measures the correlation between observed values for A and a delayed copy of A separated by a time t, it is also called the 'time correlation function'. For obvious reasons (time is discrete with the MC simulation), we use the discretized version:

$$C(t) = \frac{1}{n_{\max} - t}\left(\sum_{n=0}^{n_{\max}-t} A(n)A(n+t) - \sum_{n=0}^{n_{\max}-t} A(n) \times \sum_{n=0}^{n_{\max}-t} A(n+t)\right) \tag{B.2}$$

The 'time' is measured in the number of moves of whichever algorithm one is using. As an example, the autocorrelation function of the number of cross-edges e–encountered in the 2D RGb mapping–is shown for a few temperatures in figure B.1.

We make the reasonable assumption that the autocorrelation function decays exponentially, $C_A(t)/C_A(0) \approx e^{-t/\tau}$. The correlation time τ can be used to calculate the standard deviation σ for correlated data. Because correlated samples are not statistically independent, the usual estimate for the standard deviation that one obtains from simulation, $\bar{\sigma} = \sqrt{\bar{A^2} - \bar{A}^2}$ (see 3.14), is not correct. Instead, for the true standard deviation σ, one has an additional term:

$$\sigma^2 = \bar{\sigma}^2 + 2\sum_{n<m=0}^{n_{max}} \bar{\sigma}_{nm} \tag{B.3}$$

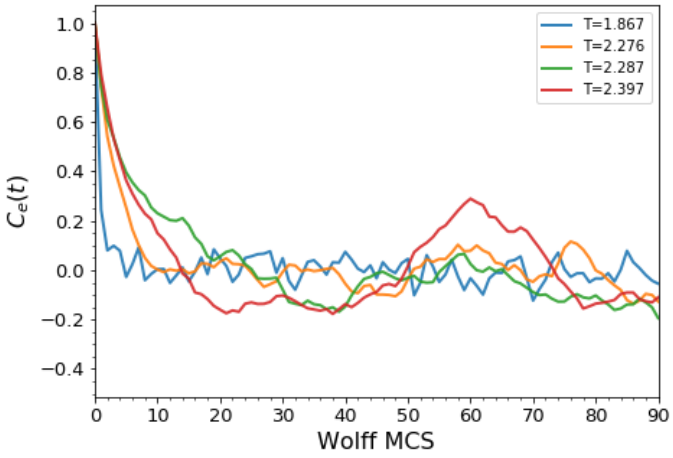


Figure B.1: *Autocorrelation function of the number of cross-edges e, which is an observable encounterd in the RGb mapping of the 2D Ising model, is shown for a few temperatures.*

where $\bar{\sigma}_{nm}$ are corresponding covariances $\overline{A(n)A(m)} - \overline{A(n)}\,\overline{A(m)}$. Noting that

$$\sum_{t=0}^{n_{max}} C(t)/C(0) = \sum_{t=0}^{n_{max}} e^{-t/\tau} \approx \tau \tag{B.4}$$

and that in fact $C(0) = \bar{\sigma}^2$, one can show [27, 28] that for large n_{max} the second term in (B.3) is equal to $2\tau\bar{\sigma}$. Thus, the proper standard error in calculated averages $\bar{A}$ is given by:

$$\sigma = \bar{\sigma}\sqrt{1+2\tau} \tag{B.5}$$

We have computed the correlation time by using a linear fit to the correlation function on a semi-log plot, see the code B and figure B.2 below. The correlation time is then determined from the slope. We have automated the process of selecting an interval for the fit, by selecting increasingly smaller intervals untill a certain accuracy (correlation coefficient corresponding to the r-value of the linear regression) is obtained. In practice, this meant starting with the interval $[0, t_{end}]$, where t_{end} is initially set to the time for which $C(t) < 0$, and from there decreasing t_{end}. In the case of finding an initial value of $t_{end} < 3$, we automatically set $\tau = 0$. The drop-off in $C(t)$ is then so fast that time correlations are virtually nonexistent.

Listing B.1: *Code to calculate correlation time*

```
import matplotlib.pyplot as plt
import matplotlib.cm as cm
import numpy as np
import pickle
import scipy.stats as sc

# Read correlation function data:
CorrData = pickle.load(open('2DcorrData.p', 'rb'))
possibleScales = CorrData['possibleScales']; temperatures = CorrData
    ['temperatures'];
LcorrNEGdata = CorrData['LcorrNEGdata'] # The observables used for
    this example: Number of Edges in Graph

i = 2 # scale index
j = 30 # temperature index
L = possibleScales[i]
T = temperatures[j]
t = np.arange(len(LcorrNEGdata[0][0]))

def corr_time(corr, acc):
    ''' Automatically fits the correlation function to an
    exponention function and finds the decay length

        Parameters:
            corr: dtype numpy array
                The correlation function of a data set

        Returns: dtype float
            The decay length of the exponential fit'''

    t = np.arange(len(corr))
    l = 0; k = np.argmax(corr<0)-1
    corr_decay = np.log(corr[l:k])

    slope, cep, r_val, _, err = sc.linregress(t[l:k], corr_decay)

    while r_val**2<acc:
        k -= 1
        slope, cep, r_val, _, err = sc.linregress(t[l:k], corr_decay
    [:k])

    plt.plot(t[l:k],corr_decay[:k],label='$\mathrm{ln(C_e(t))}$')
    plt.plot(t[l:k],cep+slope*t[l:k],ls='dashed',c='r',label='fit')
    plt.xlabel('t')
    plt.legend()
    plt.savefig('RGb2DCorrTimeFit')
    plt.show()
    tau = -1/slope
    return(tau)

print('L = %d, T = %.2f'%(L,T))
tau = corr_time(LcorrNEGdata[i][j], 0.98)
```

```
49 print('tau = %d'%(tau))
```

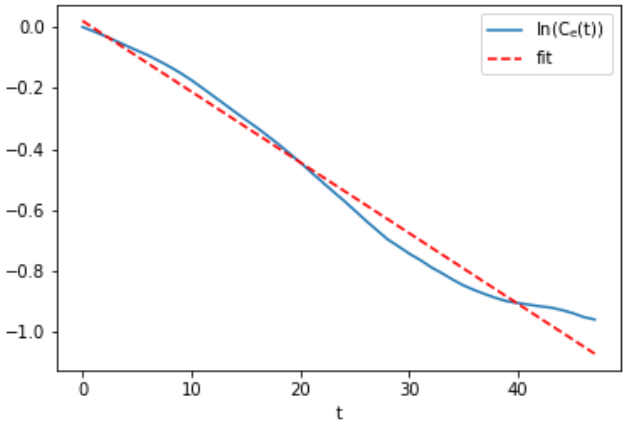


Figure B.2: *Semi-log plot of the autocorrelation function C_e and corresponding linear fit.*

In computing the correlation time of an observable using a temperature mesh $\{.., T_i, ..\}$, we found that the variation of τ with T to be somewhat unreliable. Near criticality $\tau(T)$ was usually found to be largest, however using temperatures successively closer to the critical temperature T_c did not necessary result in successively larger correlation times. One could probably solve this issue by using larger temperature intervals, but we wanted to simulate the system with a temperature mesh that was dense around T_c. In the end we decided to use the largest correlation time found over the different temperatures: $\tau_{max} = \max\{.., \tau(T_i), ..\}$, to compute the error *for all* temperatures. Hence, the standard error we used is:

$$\sigma = \bar{\sigma}\sqrt{1 + 2\tau_{max}} \tag{B.6}$$

and so for some temperatures (especially for T away from T_c) the error is overestimated.

Appendix C

Effect of Random Tie Breaker in AoSD Mapping

In our AoSD mapping, we used a random tie breaker in assigning spin values to the spins/nodes created by step 2 of the network construction procedure. Here, we explore if this 'added randomness' has any significant effect on the thermodynamics of the network model.

In step 1 of the procedure, domains of equal spins are bundled together to form new nodes/spins. The spin value of these nodes are given by the sum of the spins in the respective domains they are connected to. Next, in step 2 of the procedure, these nodes are grouped together in pairs to once again form new nodes. The nodes created by step 2, receive a spin value ± 1, where the sign is determined by whichever of the two spins they are connected to is larger in magnitude. Hence, ties only occur if the spins grouped together in step 2, are connected to domains of equal size in step 1. One might think that this does not happen very often and that the likelyhood of such an event becomes negligible when moving on to larger scales. In order to find out if this is the case, we kept track of the relative frequency in which ties occur (number of ties normalized by the number of times step 2 was performed) in our 1D AoSD simulation. See figure C.1. We observe that the frequency of ties is significant, especially for higher temperatures, where ties occur around 30% of the time. Also, increasing the scale does not lead to a diminished frequency.

Though the frequency of ties–and thus the number of times a part of the network is constructed randomly–is not negligible, the effect it has on the model may well be insignificant. To check if this is so, we performed a simulation of our 1D AoSD model where instead of the random tie breaker the sign of the spin in step 2 was chosen specifically (the spin inherited the

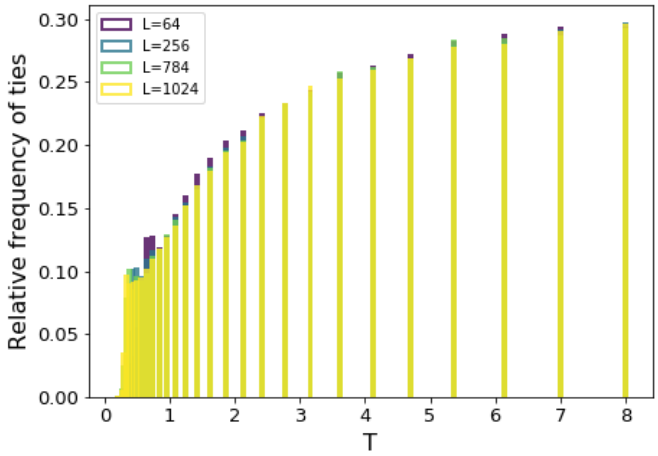


Figure C.1: *Relative frequency of ties occuring in step 2 of the 1D AoSD construction procedure. They are obtained from (the 1D Ising) simulation, performed with different scales and using* $m = 2.5 \times 10^4$ *samples for each temperature.*

sign of the most clockwise spin). Figure C.2 shows how the entropies compare. We observe that the entropies of the model with and without the use of the random tiebreaker practically overlap for all scales. The difference $\Delta = |S_{specific} - S_{random}|$, as shown in the inset, is not larger than 3×10^{-2} for the largest scale used. We conclude that the effect of the use the random

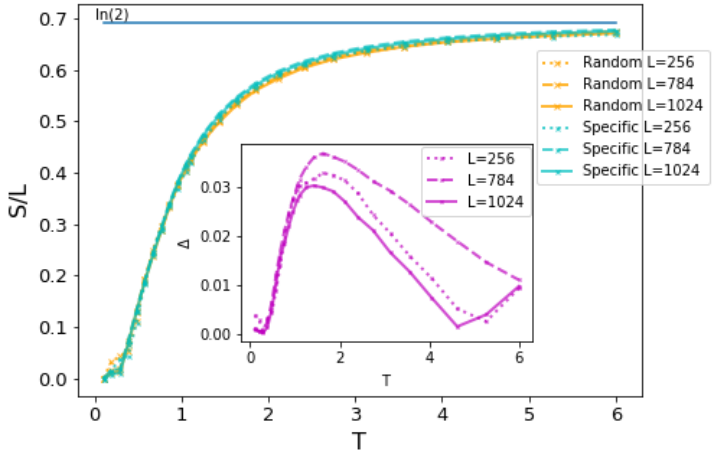


Figure C.2: *They are obtained from (the 1D Ising) simulation, performed with different scales and using* $m = 2.5 \times 10^4$ *samples for each temperature.*

tiebreaker to the model's entropy is subleading and does not significantly alter our results.

Acknowledgements

First and foremost I would like to thank my supervisors Koenraad Schalm and Diego Garlaschelli, for granting me the opportunity to do a–in my opinion–very original and multifaceted project. I would like to thank them for their supervision and the many discussions we had during our meetings. I would also like to thank the researchers and staff at the Lorentz Institute, and my fellow master students with whom I discussed my project, in particular Ahmad Jamalzada for suggesting parallel computing which literally saved hours of simulation time.

Bibliography

[1] Event Horizon Telescope Collaboration et al. First m87 event horizon telescope results. i. the shadow of the supermassive black hole. *Astophysical Journal Letters*, 875(1), 2019.

[2] J. D. Bekenstein. Black holes and entropy. *Phys. Rev. D*, 7:2333–2346, Apr 1973.

[3] S. W. Hawking. Particle creation by black holes. *Comm. Math. Phys.*, 43(3):199–220, 1975.

[4] G. 't Hooft. Dimensional reduction in quantum gravity. *Conf. Proc. C*, 930308:284–296, 1993.

[5] L. Susskind. The world as a hologram. *Journal of Mathematical Physics*, 36(11):6377–6396, 1995.

[6] J. M. Maldacena. The Large N limit of superconformal field theories and supergravity. *Int. J. Theor. Phys.*, 38:1113–1133, 1999.

[7] H. Yan. Hyperbolic fracton model, subsystem symmetry, and holography. *Physical Review B*, 99(15), Apr 2019.

[8] M. Gromov. *Hyperbolic Groups*, pages 75–263. Springer New York, New York, NY, 1987.

[9] S. W. Hawking and D. N. Page. Thermodynamics of black holes in anti-de sitter space. *Comm. Math. Phys.*, 87(4):577–588, 1982.

[10] A. Cayley. Xxviii. on the theory of the analytical forms called trees. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 13(85):172–176, 1857.

[11] H. Matsuda. Infinite susceptibility without spontaneous magnetizationexact properties of the ising model on the cayley tree. *Progress of Theoretical Physics*, 51(4):1053–1063, 1974.

[12] T. P. Eggarter. Cayley trees, the ising problem, and the thermodynamic limit. *Phys. Rev. B*, 9:2989–2992, Apr 1974.

[13] J. von Heimburg and H. Thomas. Phase transition of the cayley tree with ising interaction. *Journal of Physics C: Solid State Physics*, 7(19):3433, 1974.

[14] E. Müller-Hartmann and J. Zittartz. New type of phase transition. *Phys. Rev. Lett.*, 33:893–897, Oct 1974.

[15] M. Ostilli. Cayley trees and bethe lattices: A concise analysis for mathematicians and physicists. *Physica A: Statistical Mechanics and its Applications*, 391(12):3417 – 3423, 2012.

[16] M. Kurata, R. Kikuchi, and T. Watari. A theory of cooperative phenomena. iii. detailed discussions of the cluster variation method. *The Journal of Chemical Physics*, 21(3):434–448, 1953.

[17] C. Domb. On the theory of cooperative phenomena in crystals. *Advances in Physics*, 9(34):149–244, 1960.

[18] E. Ising. Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik*, 31(1):253–258, Feb 1925.

[19] R. J. Baxter. *Exactly solved models in statistical mechanics*. Academic Press Limited, London, 1982.

[20] H. A. Bethe. Statistical theory of superlattices. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 150(871):552–575, 1935.

[21] R. Peierls. On ising's model of ferromagnetism. *Mathematical Proceedings of the Cambridge Philosophical Society*, 32(3):477–481, 1936.

[22] S. R. A. Salinas. *The Ising Model*, pages 257–276. Springer New York, New York, NY, 2001.

[23] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[24] U. Wolff. Collective monte carlo updating for spin systems. *Phys. Rev. Lett.*, 62:361–364, Jan 1989.

[25] J. Thijssen. *Computational Physics*. Cambridge University Press, 2 edition, 2007.

[26] M. E. J. Newman and G. T. Barkema. *Monte Carlo Methods in Statistical Physics.* Clarendon Press, 1999.

[27] V. Ambegaokar and M. Troyer. Estimating errors reliably in monte carlo simulations of the ehrenfest model. *American Journal of Physics*, 78(2):150–157, 2010.

[28] A. K. Hartmann and H. Riegr. *Optimization Algorithms in Physics*. John Wiley & Sons, Ltd, 2003.

[29] D. Frenkel. Free-energy computation and first-order phase transitions. *Molecular-dynamics simulation of statistical-mechanical systems: proceedings of the 97th international school of physics "Enrico Fermi"*, pages 151–188, 1986.

[30] I. R. McDonald and K. Singer. Machine calculation of thermodynamic properties of a simple fluid at supercritical temperatures. *The Journal of Chemical Physics*, 47(11):4766–4772, 1967.

[31] C. H. Bennett. Efficient estimation of free energy differences from monte carlo data. *Journal of Computational Physics*, 22(2):245 – 268, 1976.

[32] G.M. Torrie and J.P. Valleau. Nonphysical sampling distributions in monte carlo free-energy estimation: Umbrella sampling. *Journal of Computational Physics*, 23(2):187 – 199, 1977.

[33] A. M. Ferrenberg and R. H. Swendsen. New monte carlo technique for studying phase transitions. *Phys. Rev. Lett.*, 61:2635–2638, Dec 1988.

[34] A. M. Ferrenberg and R. H. Swendsen. Optimized monte carlo data analysis. *Phys. Rev. Lett.*, 63:1195–1198, Sep 1989.

[35] S. Kumar, J. M. Rosenberg, D. Bouzida, R. H. Swendsen, and P. A. Kollman. The weighted histogram analysis method for free-energy calculations on biomolecules. i. the method. *Journal of Computational Chemistry*, 13(8):1011–1021, 1992.

[36] J. Wang and R. H. Swendsen. Transition matrix monte carlo method. *Journal of Statistical Physics*, 106(1):245–285, Jan 2002.

[37] J. Lee. New monte carlo algorithm: Entropic sampling. *Phys. Rev. Lett.*, 71:211–214, Jul 1993.

[38] F. Wang and D. P. Landau. Efficient, multiple-range random walk algorithm to calculate the density of states. *Phys. Rev. Lett.*, 86:2050–2053, Mar 2001.

[39] S. Bi and N. Tong. Monte carlo algorithm for free energy calculation. *Phys. Rev. E*, 92:013310, Jul 2015.

[40] R. H. Swendsen and J. Wang. Nonuniversal critical dynamics in monte carlo simulations. *Phys. Rev. Lett.*, 58:86–88, Jan 1987.

[41] O. Diego, J. Gonzalez, and J. Salas. The ising model on tetrahedron-like lattices: a finite-size analysis. *Journal of Physics A: Mathematical and General*, 27(9):2965–2983, may 1994.

[42] Ch. Hoelbling, A. Jakovac, J. JersÃ¡k, C.B. Lang, and T. Neuhaus. Spin and gauge systems on spherical lattices. *Nuclear Physics B - Proceedings Supplements*, 47(1):815 – 818, 1996.

[43] Ch. Hoelbling and C. B. Lang. Universality of the ising model on spherelike lattices. *Phys. Rev. B*, 54:3434–3441, Aug 1996.

[44] R. H. Swendsen. Monte carlo renormalization group. *Phys. Rev. Lett.*, 42:859–861, Apr 1979.

[45] R. H. Swendsen. Monte carlo renormalization-group studies of critical phenomena. *Journal of Applied Physics*, 53(3):1920–1924, 1982.

[46] R. H. Swendsen. Monte carlo renormalization-group studies of two-dimensional models. *Surface Science*, 125(1):104 – 115, 1983.

[47] A. Brown, A. Edelman, J. Rocks, A. Coniglio, and R. H. Swendsen. Monte carlo renormalization-group analysis of percolation. *Phys. Rev. E*, 88:043307, Oct 2013.

[48] T. W. Burkhardt and J.M.J. van Leeuwen. *Real-space renormalization*. Topics in current physics ; vol. 30. 862127785. Springer, Berlin [etc.], 1982.

[49] H. J. Maris and L. P. Kadanoff. Teaching the renormalization group. *American Journal of Physics*, 46(6):652–657, 1978.

[50] P. Erdös and A. Rényi. On random graphs i. *Publicationes Mathematicae Debrecen*, 6:290, 1959.

[51] P. Erdos and A. Renyi. On the evolution of random graphs. *Publ. Math. Inst. Hungary. Acad. Sci.*, 5:17–61, 1960.

[52] E. N. Gilbert. Random graphs. *Ann. Math. Statist.*, 30(4):1141–1144, 12 1959.

[53] J. Park and M. E. J. Newman. Statistical mechanics of networks. *Phys. Rev. E*, 70:066117, Dec 2004.

[54] E. T. Jaynes. Information theory and statistical mechanics. *Phys. Rev.*, 106:620–630, May 1957.

[55] E. T. Jaynes. *E. T. Jaynes: Papers on Probability, Statistics and Statistical Physics*. Springer Netherlands, Dordrecht, 1989.

[56] C.M. Fortuin and P.W. Kasteleyn. On the random-cluster model: I. introduction and relation to other models. *Physica*, 57(4):536 – 564, 1972.

[57] G. Grimmett. *The Random-Cluster Model*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.