Utrecht University

Master Thesis

Mathematical Sciences

Numerical analysis of domain walls in 2-dimensional square-lattice random bond Ising models using a new weighed-loop algorithm

Author:
Pepijn Overbeeke

Supervisors:
Gerard Barkema
Rob Bisseling
Rick Keesman

May 23, 2017

Universiteit Utrecht

Abstract

In this thesis we consider a 2-dimensional square-lattice random bond Ising model with random bond strength $J \pm \Delta J$ which is subject to thermal induced disorder and random bond induced disorder. We are interested in the properties of domain walls in order to achieve a better understanding of the behaviour of domain walls in real world magnetic materials.

Monte Carlo algorithms are often used to simulate the Ising models. When introducing random bond induced disorder most conventional algorithms tend to get stuck at low temperatures or systems with high disorder. To simulate random bond Ising models in these regimes while not getting stuck we introduce a new Monte Carlo algorithm; the weighed-loop algorithm. The new weighed-loop algorithm works by walking the graph induced by the lattice of the Ising model in order to form closed cycles in the graph. Flipping all the spins inside the closed cycle formed by a loop results in a difference in energy which is only determined by the bonds on the loop. By selecting more bonds which are energetically favourable to change the weighed-loop algorithm tends to get stuck less in the regimes where conventional algorithms do get stuck.

In this thesis we provide the theoretical foundations of the 1-dimensional domain walls in 2-dimensional Ising models and we provide the theory on Monte Carlo algorithms on Ising models. We give a detailed description of the new weighed-loop algorithm. We prove that the weighed-loop algorithm correctly simulates the random bond Ising models. To show that the weighed-loop algorithm tends to get stuck less than other algorithms we compare the autocorrelation of the weighed-loop algorithm and an algorithm for glassy spin systems, the Niedermayer's algorithm. We show that our simulations agree with the theoretical results of domain walls in the absence of random bond induced disorder. Furthermore, we simulate the domain walls in the presence of random bond induced disorder for different values in the parameter space and we deduce the Larkin length L_c which is the typical length scale for which a crossover takes place between random bond induced disorder and thermal induced disorder.

Contents

1 Introduction

In statistical physics the Ising model is one of the most well known and understood physical models [1]. For 1 and 2 dimensions the Ising model has been exactly solved [2, 3], but more complex models, like the random bond Ising model, have not yet been exactly solved [4]. For these kind of models numerical computer simulations are able to provide accurate results which can be used in practice. Since the research on magnetic domain walls is a tough theoretical research, numerical simulations can improve the knowledge of the properties of the domain walls in magnetic materials. The domain wall moves through a disorder configuration by the movement of segments between pinning sites. In this way the domain wall moves through the energy landscape from metastable state to metastable state and for a higher disorder the motion of the domain wall is slower.

Of the Monte Carlo type algorithms, Markov Chain Monte Carlo algorithms are the most used type of algorithm for simulating physics in statistical models and on the standard 2D Ising model. The first one of these algorithms is the Metropolis algorithm [5, 6]. Prior to the Metropolis algorithm, Monte Carlo algorithms consisted of repeatedly generating large numbers of configurations to compute the physical properties. The publication of the Metropolis algorithm [5] introduced the samplic method and it introduced periodic boundary conditions. Both have been a key part of Monte Carlo algorithms in statistical physics ever since. Due to the fact that the Metropolis algorithm is a single-spin flip algorithm, it is subject to critical slowing down, meaning that the simulations are less efficient in the neighbourhood of the critical temperature. To overcome the critical slowing down, Swendsen and Wang created the Swendsen-Wang algorithm [7] which was later generalized by Wolff in the Wolff algorithm [8]. The Swendsen-Wang algorithm and the Wolff algorithm are cluster type algorithms for the standard 2D Ising model contrary to the single-spin flip type algorithm like the Metropolis algorithm. Both the Swendsen-Wang and Wolff algorithms showed a smaller scaling of the correlation time with respect to the system size of the model. However, the Swendsen-Wang and Wolff algorithms are not able to simulate frustrated models. Niedermayer proposed an extension of the Swendsen-Wang and Wolff algorithm which is also applicable to glassy spin systems and random bond Ising models[9].

When introducing disorder by means of random bond strength in square-lattice Ising models the energy landscape of the system becomes rough and results in a degenerate ground state. Conventional algorithms tend to get stuck which results in the oversampling of parts of the phase space due to exponentially low probabilities of leaving meta-stable states in the energy landscape. To overcome these obstacles new algorithms have emerged with another update scheme [9, 10, 11, 12] as well as algorithms with a different simulation scheme [13, 14]. These algorithms also follow local updates but the local updates are not determined by the energy barrier between different meta-stable states. We propose a new algorithm, the weighed-loop algorithm, in which clusters of flippable spins are created by selecting bonds which are energetically favourable to change.

This thesis is outlined as follows. In section 2 the random bond Ising model is explained as well as the properties of domain walls. Section 3 describes the weighed-loop algorithm in detail and also provides a proof that the weighed-loop algorithm satisfies the detailed balance equation and ergodicity. The results of the weighed-loop algorithm and the application of the weighed-loop algorithm to domain walls is presented in section 4. Finally we present possible future works and a conclusion in the sections 5 and 6.

2 Theory

In this section we give a description of the random bond Ising model (RBIM) and list properties of domain walls in RBIM's. We present the general theory on Monte Carlo methods as well as the theory on the Metropolis algorithm, Wolff algorithm, and Niedermayer's algorithm. The last section of this chapter is devoted to the autocorrelation of algorithms.

2.1 Random Bond Ising Model

This section is devoted to the theory of random bond Ising models. The RBIM is based on the standard 2D Ising model, the key difference is that the bond strength between neighbouring spins in the model is randomized. We only consider the 2-dimensional square-lattice Ising models, but other models like triangular-lattices, hexagonal-lattices or multidimensional lattices are also valid.

The standard Ising model consists of spins-1/2 particles $\sigma_i \in \{-1, +1\}$ on a n-dimensional LxH lattice Λ. We call a certain configuration of all the different spins on the lattice a spin configuration σ. The Hamiltonian $\mathcal{H}(\sigma)$ of such a spin configuration σ of the system is given by

$$\mathcal{H}(\sigma) = -\sum_{i \in \Lambda} \sum_{j \in \mathcal{N}_i} J_{ij} \sigma_i \sigma_j \,,$$

where $\mathcal{N}_i$ is the set of indices of the all the neighbouring spins of spin i. J_{ij} is the strength of the bond between spins i and j. Bonds which contribute negatively to the Hamiltonian are satisfied bonds while bonds which contribute positively to the Hamiltonian are unsatisfied bonds. Since the Ising model is a thermodynamical model, it is subject to the Boltzmann distribution. That is, the probability $P(\sigma)$ of the system being in a state σ is equal to

$$P(\sigma) = \frac{e^{-\beta \mathcal{H}(\sigma)}}{Z} \,, \tag{1}$$

as shown in [15]. In the Boltzmann distribution $\beta = \frac{1}{k_b T}$ is the inverse temperature defined by the Boltzmann constant k_b and the temperature T of the system. Z is a normalizing constant, also known as the partition function.

The one dimensional case of the Ising model was solved by Ising in 1925 [2] in which Ising showed that the one dimensional case is not subject to a phase transition. The two dimensional case of the Ising model was solved by Onsager in 1944 [3]. Onsager showed that without an external magnetic field and equal bond strength J for all bonds in the system, the Ising model contains a continuous phase transition around the critical temperature where the critical temperature T_c occurs at

$$T_c = \frac{2J}{k_b \log\left(1 + \sqrt{2}\right)} \,,$$

so in other words $\frac{kT_c}{J} = 2.269...$ or equivalent $\beta J = 0.440....$

To define a RBIM we introduce the following disorder which is characterized in the bond strength $J_{ij} = J \pm \Delta J$. Here $0 \leq \Delta \leq 1$ and

$$\begin{aligned} \langle J_{ij} \rangle &= J \\ \langle J_{ij} J_{nm} \rangle &= J^2 + J^2 \Delta^2 \delta_{in} \delta_{jm} \,. \end{aligned}$$

The RBIM somewhat resembles perpendicularly magnetized materials where the domain walls are narrow. However, due to universality classes we can derive results for these experimental systems by using the RBIM. A universality class is a collection of thermodynamical models which behave the same under a scale invariant limit and each member of an universality class has identical critical behaviour. The shape of the domain wall does only depend on the dimension of the domain wall and the roughness, but since the roughness is universal and does not depend on the microscopic details of the system we expect to deduce properties of these experimental systems using simulations on the RBIM.

2.2 Domain Wall

In this subsection we define the domain wall for Ising models and we also present theoretical results for the behaviour of the domain walls in standard Ising models and random bond Ising models.

The domain wall is defined as a 1-dimensional separation between two regions of clusters of spins. Given a 2-dimensional Ising model on a LxH lattice we introduce anti-periodic boundary conditions along the L-boundary and periodic boundary conditions along the H-boundary. Using these boundary conditions the lattice is topological equivalent to a Klein bottle and thus at least one domain wall must always be present.

We describe the horizontal displacement of the domain wall by the function $\mathrm{dw}(l)$ where $1 \leq l \leq L$ is the vertical position. The function $\mathrm{dw}(l)$ is defined by

$$\mathrm{dw}(l) = -\frac{H}{2} + \frac{1}{2} \left(\sum_{h=0}^{H} \sigma_{lh} \pmod{H} \right) .$$

Since $\mathrm{dw}(l)$ is single-valued for each value of l and t it means that $\mathrm{dw}(l)$ ignores overhangs and local pockets of spins. This leads to $\mathrm{dw}(l)$ not representing the domain wall accurately for high temperatures since for high temperatures there are a relative large amount of overhangs and local pockets. However, for low temperature there are little to no overhangs and local pockets since both overhangs and local pockets of spins are energetic expensive. Below the critical temperature we also assume that the probability of the domain wall being as wide as it is long is negligible. A typical configuration of a part of a domain wall is in Figure 1.

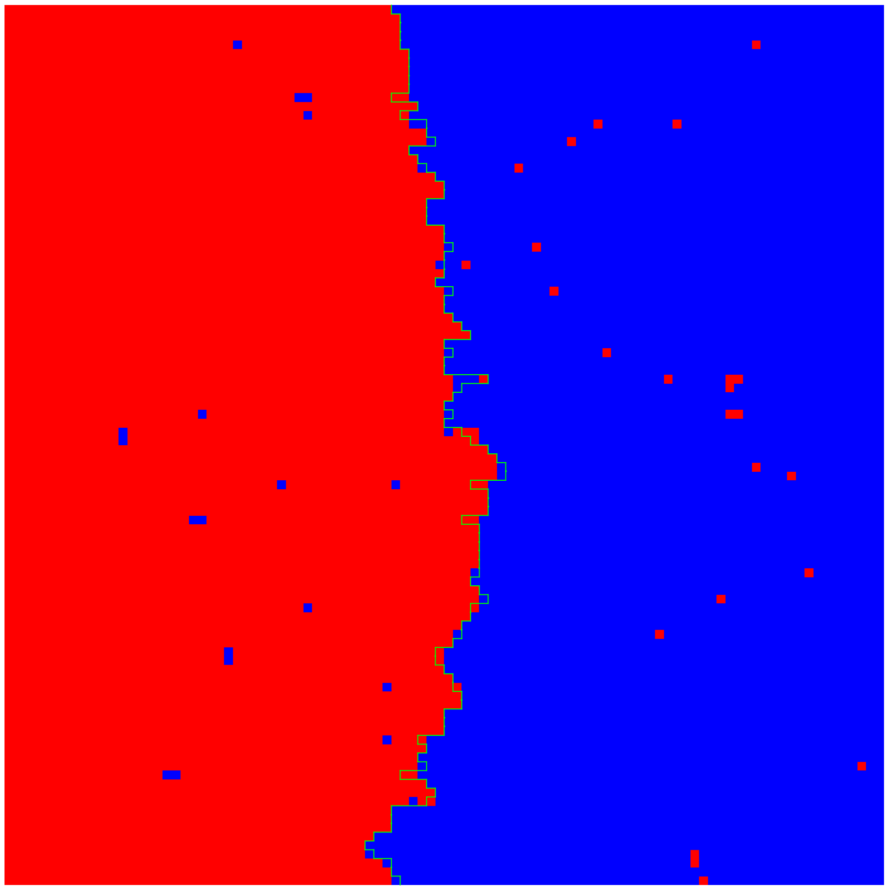

Figure 1: An example of a domain wall, displayed by the green line. This example was simulated using the values $L = 100, H = 100$, $\beta = 0.7$ and $\Delta = 0.0$ The vertical boundary has anti-periodic boundary conditions while the horizontal boundary has periodic boundary conditions. In this example the system has local pockets and the domain wall has overhangs so the definition of $\mathrm{dw}(l)$ is not an accurate description of the separation between the regions of the red spins and the blue spins.

The values of the horizontal displacement $\mathrm{dw}(l)$ can be transformed with the fourier transformation to get the fourier coefficients $\mathcal{S}(p)$, $\mathcal{C}(p)$ and $\mathcal{F}(p)$.

$$\begin{aligned}
\mathcal{S}(p) &= \frac{1}{L}\sum_{l=0}^{L}\sin\left(\frac{2\pi pl}{L}\right)\mathrm{dw}(l)\,,\\
\mathcal{C}(p) &= \frac{1}{L}\sum_{l=0}^{L}\cos\left(\frac{2\pi pl}{L}\right)\mathrm{dw}(l)\,,\\
\mathcal{F}(p) &= \frac{1}{L}\sum_{l=0}^{L}e^{\frac{2\pi ipl}{L}}\,\mathrm{dw}(l)\,,
\end{aligned}$$

We are interested in the equilibrium properties of the structure factor $A(p, L, \beta J) = \left\langle|\mathcal{F}(p)|^2\right\rangle$ of the fourier modes p of the domain wall as a disorder and thermal average over all possible disorder configurations. Since the position of the domain wall is real valued, we have that $\left\langle|\mathcal{F}(p)|^2\right\rangle = \left\langle|\mathcal{F}(-p)|^2\right\rangle$ the fourier coefficients are symmetric. When inducing a disorder Δ in the Ising model, $A(p, L, \beta J)$ also is dependent on Δ which is denoted by $A(p, L, \beta J, \Delta)$.

From dimensional analysis one can derive for the structure factor that $A(p, L, \beta J, \Delta) \sim p^{-(1+2\zeta)}$ for the low modes p of the fourier transformation. In a system without any disorder it is known that the shape of the domain wall are thermally induced and $\zeta_{\mathsf{T}} = \frac{1}{2}$. When the system is subject to a random field the disorder component is $\zeta_{\mathsf{RF}} = 1$ [16, 17]. Using a disorder Δ in the bond strength introduces a wandering exponent for the domain wall in equilibrium $\zeta_{\mathsf{RB}} = \frac{2}{3}$ [18, 19], leading to $A(p, L, \beta J, \Delta) \sim p^{-7/3}$ for the low modes p. This wandering exponent was also reproduced in experiments with thin films [20, 21, 22, 23]. When both the thermal fluctuations and the bond strength is randomized in the system a crossover point p_c exists which separates the regions dominated by thermal disorder and dominated by random bond induced disorder. The crossover point is also characterized as the Larkin length L_c [24] in terms of lattice spacing, meaning that locally (on a length scale below the Larkin length) thermal dynamics determine the shape of the domain wall while on a global scale random bond induced disorder determines the shape.

We will show that for systems without disorder $A(p, L, \beta J) = A(L, \beta J)\csc^2\left(\frac{\pi p}{L}\right)$ where $A(L, \beta J)$ is a scaling factor. For high temperatures it is known from capillary-wave theory [25] and surface tension of the 2-dimensional Ising model [3] that for systems without disorder the high temperature scaling factor A_{high} is given by

$$A_{\mathsf{high}}(L, \beta J) \sim \sinh^{-1}\left(2\beta J + \log\left(\tanh\left[\beta J\right]\right)\right).$$

For low temperatures we can use a low temperature expansion to determine the behaviour of $A(p, L, \beta J)$ in low temperatures which will yield a low temperature scaling factor $A_{\mathsf{low}}(L, \beta J)$. Using both the high temperature scaling factor $A_{\mathsf{high}}(L, \beta J)$ and the low temperature scaling factor $A_{\mathsf{low}}(L, \beta J)$ we can determine the scaling factor $A(L, \beta J)$ such that $A(L, \beta J)$ behaves correctly in both the high temperature regime and the low temperature regime. The low temperature expansion and the derivation of $A(L, \beta J)$ presented here are taken from [26].

Consider a square-lattice Ising model without disorder. Then the ground state of the domain wall is defined as a straight wall without any excitations contributing to the structure of the domain wall. We define a defect in the domain wall as an excitation of the domain wall of height 1 over a certain length b. A defect increases the length of the wall with 2 and thus increases the energy of the system with $4J$. Let $A_d(p, L, \beta J)$ be the average of $A(p, L, \beta J)$ given d defects. For the value of $A(p, L, \beta J)$ using $A_d(p, L, \beta J)$ we have

$$A(p, L, \beta J) = \sum_{d=0}^{\infty} A_d(p, L, \beta J) P(d, L, \beta J) \,.$$

where $P(d, L, \beta J)$ is the probability of finding exactly d defects. We are now considering the case where there is only one defect in the domain wall present. Since we are not interested in the phase of the fourier coefficients, we can assume without loss of generality that the defect starts at position $l = 1$. The horizontal displacement of the domain wall $\mathrm{dw}(l)$ is given by

$$\mathrm{dw}(l) = \begin{cases} 1 & \text{if } l \leq b \,, \\ 0 & \text{otherwise} \,. \end{cases}$$

Using these values of $\mathrm{dw}(l)$ we derive the following expressions for $\mathcal{S}(p)$ and $\mathcal{C}(p)$

$$\begin{aligned}
\mathcal{S}(p) &= \frac{1}{L} \sum_{l=1}^{b} \sin\left(\frac{2\pi l p}{L}\right) \\
&= \frac{1}{2L} \csc\left(\frac{\pi p}{L}\right) \left(\cos\left(\frac{\pi p}{L}\right) + \cos\left(\frac{2\pi b p + \pi(L-p)}{L}\right)\right) \,, \\
\mathcal{C}(p) &= \frac{1}{L} \sum_{l=1}^{b} \cos\left(\frac{2\pi l p}{L}\right) \\
&= \frac{1}{2L} \left(1 + \csc\left(\frac{\pi p}{L}\right) \sin\left(\frac{2\pi b p + \pi(L-p)}{L}\right)\right) \,.
\end{aligned}$$

Square both terms and adding them together yields

$$\mathcal{F}^2(p) = \mathcal{S}^2(p) + \mathcal{C}^2(p) = \frac{1}{L^2} \csc^2\left(\frac{\pi p}{L}\right) \sin^2\left(\frac{\pi b p}{L}\right) \,.$$

Averaging over all non-trivial values of b yields

$$\begin{aligned}
A_1(p, L, \beta J) &= \frac{1}{L-1} \sum_{b=1}^{L-1} \left(\mathcal{S}^2(p) + \mathcal{C}^2(p)\right) \\
&= \frac{1}{4L^2(L-1)} \csc^2\left(\frac{\pi p}{L}\right) \left(2L + 1 - \csc\left(\frac{\pi p}{L}\right) \sin\left(\frac{\pi p(2L-1)}{L}\right)\right) \\
&\approx \frac{1}{2L^2} \csc^2\left(\frac{\pi p}{L}\right) \,,
\end{aligned} \tag{2}$$

where in the last step we used that L is large and that for $1 \leq p \leq \lfloor \frac{L-1}{2} \rfloor$ we have

$$\csc\left(\frac{\pi p}{L}\right) \sin\left(\frac{\pi p(2L-1)}{L}\right) = 1 \,.$$

We now assume that for $d \ll L$ the different defects do not interact and thus behave independently. That is, $A_d(p, L, \beta J) \approx d A_1(p, L, \beta J)$. So

$$A(p, L, \beta J) = A_1(p, L, \beta J) \sum_{d=0}^{\infty} d P(d, L, \beta J) \,.$$

Since each defect changes the energy of the system, $P(d, L, \beta J)$ is determined by the Boltzmann distribution (1). Using the Boltzmann equation we get

$$P(d, L, \beta J) = \frac{1}{Z(L, \beta J)} g(d, L) e^{-4d\beta J} \,,$$

where $g(d, L)$ is the number of configurations in which d defects can occur given length L and $Z(L, \beta J)$ is the partition function. To estimate $g(d, L)$ we assume that we have to distribute d horizontal moves to the left and d horizontal moves to the right over L positions. For large d this estimation is an overestimation since we did not account for the fact that multiple moves of one kind can occur at the same position and that moves to the left and right cannot occur in the same position. As an estimation for $g(d, L)$ we now have

$$g(d, L) = \binom{L}{d}^2 \approx \left(\frac{L^d}{d!}\right)^2 \,.$$

We can use this estimation in the expression for $P(d, L, \beta J)$ together with (2) to get

$$\begin{aligned} P(d, L, \beta J) &= \frac{g(d, L) e^{-4d\beta J}}{Z(L, \beta J)} \\ &= \frac{\left(\frac{L^d}{d!}\right)^2 e^{-4d\beta J}}{\sum_{\delta=0}^{\infty} \left(\frac{L^\delta}{\delta!}\right)^2 e^{-4\delta\beta J}} \\ &= \frac{\left(\frac{L^d}{d!}\right)^2 e^{-4d\beta J}}{I_0\left(2Le^{-2\beta J}\right)} \,. \end{aligned}$$

Here I_n is the modified Bessel function of the first kind. We can plug the expression of $P(d, L, \beta J)$ in the expression of the structure factor $A(p, L, \beta J)$ to get

$$\begin{aligned} A(p, L, \beta J) &= A_1(p, L, \beta J) \sum_{d=0}^{\infty} \frac{d \left(\frac{L^d}{d!}\right)^2 e^{-4d\beta J}}{I_0\left(2Le^{-2\beta J}\right)} \\ &= \frac{e^{-2\beta J}}{2L} \frac{I_1\left(2Le^{-2\beta J}\right)}{I_0\left(2Le^{-2\beta J}\right)} \csc^2\left(\frac{\pi p}{L}\right) \\ &= A_{\text{low}}(L, \beta J) \csc^2\left(\frac{\pi p}{L}\right) \,. \end{aligned} \tag{3}$$

Here we have defined the scaling factor of the low temperature expansion A_{low} as

$$A_{\mathsf{low}} = \frac{e^{-2\beta J}}{2L} \frac{I_1\left(2Le^{-2\beta J}\right)}{I_0\left(2Le^{-2\beta J}\right)} \,.$$

We will now unify the expressions for $A_{\mathsf{low}}(L, \beta J)$ and $A_{\mathsf{high}}(L, \beta J)$ in a common expression for $A(L, \beta J)$ such that $A(L, \beta J)$ behaves as $A_{\mathsf{low}}(L, \beta J)$ for low temperatures and $A_{\mathsf{high}}(L, \beta J)$ for high temperatures.

For the unification we let

$$A_{\mathsf{high}}(L, \beta J) = a(L, \beta J) \sinh^{-1}\left(2\beta J + \log\left(\tanh\left[\beta J\right]\right)\right),$$

which results in

$$2\beta J + \log\left(\tanh\left[\beta J\right]\right) \approx 2\beta J \,,$$

so for $A_{\mathsf{high}}(L, \beta J)$ we have for $\beta_c J \ll \beta J$

$$\begin{aligned} A_{\mathsf{high}}(L, \beta J) &\approx a(L, \beta J) \sinh^{-1} 2\beta J \\ &= 2a(L, \beta J)\left(e^{2\beta J} - e^{-2\beta J}\right)^{-1} \\ &\approx 2a(L, \beta J) e^{-2\beta J} \,. \end{aligned}$$

For $A_{\mathsf{low}}(L, \beta J)$ we use the following approximation for the modified Bessel function for small z

$$I_n(z) \approx \frac{1}{\Gamma(n+1)} \left(\frac{z}{2}\right)^n \,.$$

Using this approximation results in

$$\begin{aligned} A_{\mathsf{low}}(L, \beta J) &= \frac{2Le^{-2\beta J}}{4L^2} \frac{2Le^{-2\beta J}}{2} \\ &= \frac{4L^2 e^{-4\beta J}}{8L^2} \\ &= \frac{e^{-4\beta J}}{2} \,. \end{aligned}$$

If $0 < \beta J - \beta_c J \ll 1$ we know that $A_{\mathsf{high}}(L, \beta J)$ diverges as $(\beta J - \beta_c J)^{-1}$ [25]. To achieve this divergence we want that $\sinh\left(2\beta J + \log\tanh\beta J\right) \to 0$, in other words we want $x = 2\beta J + \log\tanh\beta J \to 0$. This leads to

$$\begin{aligned} A_{\mathsf{high}}(L, \beta J) &\approx a(L, \beta J) \sinh^{-1} x \\ &\approx a\left(L, \beta J\right) \chi^{-1} \,, \end{aligned}$$

where $\chi = 4(\beta J - \beta_c J)$. Hence

$$A_{\mathsf{high}}(L, \beta J) \approx \frac{a(L, \beta J)}{4} (\beta J - \beta_c J)^{-1} \,.$$

For small z we have the following approximation for the Bessel function

$$I_n(z) \approx \frac{e^z}{\sqrt{2\pi z}} \,.$$

For $0 < \beta J - \beta_c J \ll 1$, this leads to

$$\begin{aligned} A_{\mathsf{low}}(L, \beta J) &\approx \frac{2Le^{-2\beta J}}{4L^2} \\ &= \frac{e^{-2\beta J}}{2L} \,. \end{aligned}$$

We determine $a(L, \beta J)$ such that $A_{\mathsf{high}}(L, \beta J)$ for low temperatures is equal to $A_{\mathsf{low}}(L, \beta J)$ for high temperatures. This means that

$$\begin{aligned} 2a(L, \beta J)e^{-2\beta J} &= \frac{e^{-2\beta J}}{2} \\ a(L, \beta J) &= \frac{1}{4L} \,. \end{aligned}$$

Using this value of $a(L, \beta J)$ we arrive at

$$A_{\mathsf{high}}(L, \beta J) = \frac{1}{4L} \sinh\left(2\beta J + \log \tanh \beta J\right) .$$

If we now let

$$A(L, \beta J) = A_{\mathsf{high}}(L, \beta J) \frac{I_1\left(4L^2 A_{\mathsf{high}}(L, \beta J)\right)}{I_0\left(4L^2 A_{\mathsf{high}}(L, \beta J)\right)} \,, \tag{4}$$

we have an expression for $A(L, \beta J)$ which best fits both $A_{\mathsf{high}}(L, \beta J)$ and $A_{\mathsf{low}}(L, \beta J)$ in the respective temperature regimes.

The idea of this thesis is to use the proposed weighed-loop algorithm to investigate the structure factor, the crossover point p_c and the Larkin length L_c of the different disorder regimes when changing the parameters L, β and Δ. By investigating the behaviour of both the thermal fluctuations induced disorder and the random bond induced disorder the properties of the shape of the domain wall can be compared with the known theory.

2.3 Monte Carlo methods

In this section we introduce Monte Carlo methods as well as Monte Carlo methods on the Ising model and the detailed balance condition. We will explain the Metropolis algorithm, the Wolff algorithm and Niedermayer's algorithm in more detail.

Monte Carlo methods (or Monte Carlo algorithms) are algorithms which rely on repeating random sampling to obtain results. The idea is that when using enough random samples the error in the sampling process decreases such that the averaged values approximate the real values.

Monte Carlo methods are used in a wide variety of scientific fields, ranging from mathematics, for the estimation the surface of an integral, to physics, for the simulation of interacting particle systems, and to economics, determining the price of an option of a share. Since there is such a wide variety on Monte Carlo methods there is no standard description of Monte Carlo methods.

As an example of a Monte Carlo algorithm we present a Monte Carlo algorithm for the calculation of an integral of a function. Consider a function $f : \mathbb{R} \to \mathbb{R}_{\geq 0}$. We now want to calculate the integral of f over the domain $[a, b]$. We can approximate this integral by taking random samples. Each iteration we pick a point in the plane $[a, b] \times [0, \max f]$. If this point is above f, do not count the point and if the point is below f, count the point. If there are enough random samples the fraction of counted points to the total points will approximate the fraction of the area below the function and thus the value of the integral. This algorithm is shown in algorithm 1. By the central limit theorem the error e in the approximation decreases as $1/\sqrt{n}$.

```
  Data: $n = 0$, $X = 0$
  Result: Approximation of $\int_a^b f(x)\,\mathrm{d}x$
1 while Stopping criteria not met do
2     Pick a point $(x, y)$ at random from $[a, b] \times [0, \max f]$.
3     Increment $n$.
4     If $f(x) > y$ increment $X$.
5 end
6 return $\frac{X}{n}(b - a)\max f$
```

Algorithm 1: Integral approximation using random sampling

Here the stopping criteria can be different criteria. For example, after a fixed amount of n steps are executed the algorithm is terminated. Or the algorithm is terminated if the squared error is below a certain tolerance.

As an example, suppose we want to integrate the function $\sin x$ on the interval $[0, \pi]$. Integration by hand yields the result of 2. If we use random sampling we get the results from Figure 2 and Figure 3. In Figure 2 we have the root square mean plotted against the number of random samples used. The true value of the integral is also plotted as a red line for comparison. We observe from the figure that the root square mean converges to the true value of the integral meaning that random sampling can be used to evaluate the integral. In Figure 3 the error of the average of the random sampling and the true value of the integral is plotted against the number of random samples used. The line $1/\sqrt{n}$ is also plotted as a red line to show that the error decreases as $1/\sqrt{n}$ as the theory predicted.

Monte Carlo integration is most often used for higher dimensional integrals which are too difficult to compute with known non-numerical methods. Dimensions up to 1000 are not uncommon with these kind of integrals.

The most used type of Monte Carlo methods used on the Ising model are Markov Chain Monte Carlo algorithms. The idea is that each step the algorithm selects a new state of the Ising model. Each step has a probability to transition from one

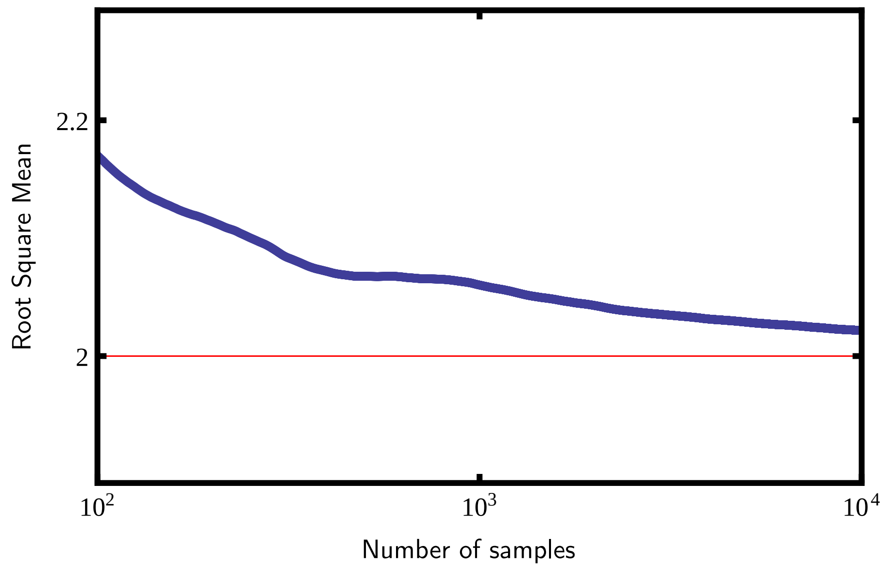

Figure 2: The root square mean of the sampling of $\int_0^\pi \sin x \, \mathrm{d}x$ (blue points) on a log-log scale. Each step a new random samples is added to the RSM. The red line corresponds to the true value of the integral. The data points converge to the true value of the integral when more random samples are used to computed the average. This shows that random sampling can be used to estimate the value of an integral.

state μ to another state ν. The resulting Markov chain has a limiting distribution and we force the limiting distribution to be the same as the Boltzmann distribution (1). Most Markov chain Monte Carlo algorithms follow a specific pattern, this is also shown in algorithm 2.

```
1 while Stopping criteria not met do
2     Select a transition from the current state to $\mu$ to a new state $\nu$.
3     Apply the transition with a certain probability.
4 end
```

Algorithm 2: A Markov Chain Monte Carlo algorithm

As with the Monte Carlo integration, the stopping criteria can be a variety of criteria, but most of the time the algorithm is stopped after a fixed amount of steps. The probability of applying the transition from μ to ν is defined such that the detailed balance condition is satisfied.

2.3.1 Markov Chains and Detailed Balance

One can view most Monte Carlo methods on the Ising model as a discrete time Markov Chain. The state space $\mathcal{S}$ exists of all possible spin configurations σ of the

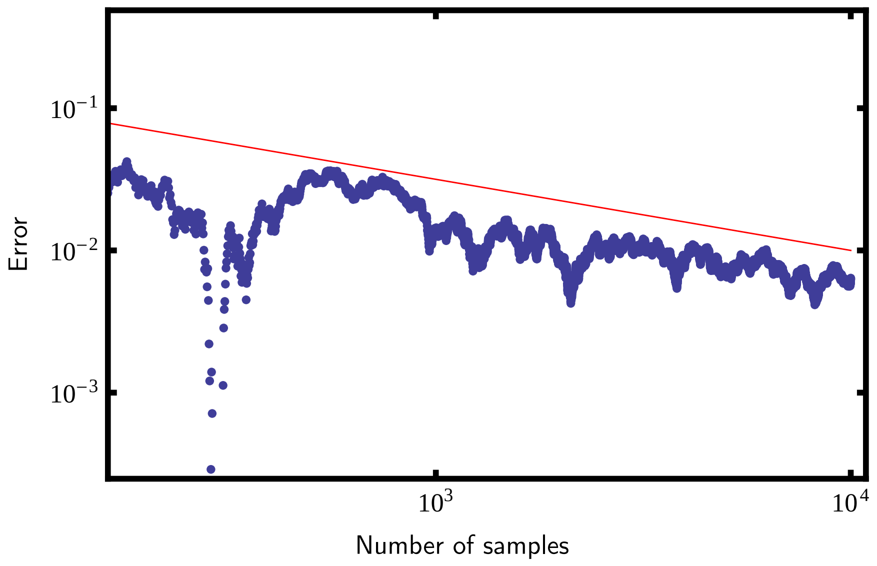

Figure 3: The error of the sampling of $\int_0^\pi \sin x \, \mathrm{d}x$ (blue points) on a log-log scale. The red line is the line $1/\sqrt{n}$. The plot shows that the error does indeed behave like $1/\sqrt{n}$ as predicted.

lattice of the Ising model and a Monte Carlo method transitions the system from one state μ to another state ν with some probability which only depends on ν and μ. We assume that for fixed temperature β and after a finite amount of time the system, and thus the Markov Chain, reaches its equilibrium. For the Markov Chain this means that there exists positive numbers $\pi(\nu)$ where $\nu \in \mathcal{S}$ is a possible state, summing to unity that satisfy the equilibrium equations

$$\pi(\nu) = \sum_{\mu \in \mathcal{S}} \pi(\mu) p(\mu, \nu) \, ,$$

Here $p(\nu, \mu)$ is the transition probability to transition from state ν to state μ. The distribution $\pi(\nu)$ is called the stationary distribution of the Markov chain.

To determine the stationary distribution of the Markov Chain for a given temperature β we use the following theorem:

Theorem 2.1. *Let C be a stationary Markov chain. Suppose there exists π with $\sum_{j \in \mathcal{S}} \pi(j) = 1$. Also suppose that for each pair $j, k \in \mathcal{S}$ the detailed balance equation*

$$\pi(j) p(j, k) = \pi(k) p(k, j) \, , \tag{5}$$

is satisfied. Then π is the stationary distribution of C.

Proof. We start by summing the detailed balance equation (5) over all $k \in \mathcal{S}$. We get

$$\pi(j) \sum_{k \in \mathcal{S}} p(j,k) = \sum_{k \in \mathcal{S}} \pi(k) p(k,j) \,.$$

Since the sum on the left hand side equals to unity (a state always transitions to a state in the state space), the above equation simply reduces to the equilibrium equations. Hence π is the stationary distribution of C. □

Using the above theorem we can now properly formulate the requirements for a Monte Carlo algorithm on the Ising model. Fristly, we want the algorithm to satisfy the detailed balance equations for the Ising model:

$$P(\mu)\Pi(\mu \to \nu) = P(\nu)\Pi(\nu \to \mu) \,,$$

where μ and ν are possible configurations of the system, $P(\sigma)$ is as defined in (1) and $\Pi(\mu \to \nu)$ is the transition probability to transition the system from μ to ν. From theorem 2.1 we know that the stationary distribution of the algorithm is equal to the Boltzmann distribution and thus in equilibrium the algorithm simulates according to the Boltzmann distribution.

Since in most Monte Carlo algorithms on the Ising model the algorithm first selects a possible transition before applying said transition, it is useful to split the transition probability $\Pi(\mu \to \nu) = g(\mu \to \nu)A(\mu \to \nu)$. Here $g(\mu \to \nu)$ is the selection probability, the probability that a transition from μ to ν us selected and $A(\mu \to \nu)$ is the acceptance probability, the probability that the transition from μ to ν is accepted.

We already know the distribution $P(\mu)$ so we can rearrange (5) as

$$\frac{g(\mu \to \nu)}{g(\nu \to \mu)} \frac{A(\mu \to \nu)}{A(\nu \to \mu)} = e^{-\beta(\mathcal{H}(\nu) - \mathcal{H}(\mu))} \,.$$

In the above equation we want the fraction $\frac{A(\mu \to \nu)}{A(\nu \to \mu)}$ to be as large as possible to increase the number of accepted transitions. If the number of accepted transitions is smaller more computational time is wasted calculating new possible transitions which are not accepted.

For the second requirement we want the algorithm to be ergodic so that we are certain that we can reach each state from any given other state in a finite amount of steps. For if the algorithm is not ergodic, we are not certain if there exists states the algorithm will not reach meaning that the algorithm violates the Boltzmann distribution.

2.3.2 Metropolis Algorithm

One of the most common and well known algorithms on Ising models is the Metropolis algorithm, also known as the Metropolis-Hastings algorithm [5]. The Metropolis algorithm is a single-spin flip algorithm where in each step a spin is selected and then said selected spin is flipped with a certain probability.

Suppose we have an Ising model with N spins and suppose that at the beginning of a Monte Carlo step the system is in configuration μ. In each step the algorithm first selects a random spin. Let us assume flipping this spin transitions the system from μ to ν. Each spin has an equal probability of being chosen, so $g(\mu \to \nu) = 1/N$. Note that this also means that $g(\nu \to \mu) = 1/N$. The detailed balance condition now tells us that

$$\begin{aligned}\frac{g(\mu \to \nu)}{g(\nu \to \mu)}\frac{A(\mu \to \nu)}{A(\nu \to \mu)} &= e^{-\beta(\mathcal{H}(\nu)-\mathcal{H}(\mu))}\,,\\ \frac{\frac{1}{N}}{\frac{1}{N}}\frac{A(\mu \to \nu)}{A(\nu \to \mu)} &= e^{-\beta(\mathcal{H}(\nu)-\mathcal{H}(\mu))}\,,\\ \frac{A(\mu \to \nu)}{A(\nu \to \mu)} &= e^{-\beta(\mathcal{H}(\nu)-\mathcal{H}(\mu))}\,.\end{aligned}$$

We want to maximize $\frac{A(\mu \to \nu)}{A(\nu \to \mu)}$, so if we choose

$$A(\mu \to \nu) = \begin{cases} e^{-\beta(\mathcal{H}(\nu)-\mathcal{H}(\mu))} & \text{if } H(\mu) > H(\nu) \\ 1 & \text{otherwise}\,, \end{cases}$$

we have satisfied all the constraints and maximized $\frac{A(\mu \to \nu)}{A(\nu \to \mu)}$. Note that this equation for $A(\mu \to \nu)$ is also the same as

$$A(\mu \to \nu) = \min\left(1, e^{-\beta(\mathcal{H}(\nu)-\mathcal{H}(\mu))}\right).$$

Since there are N spins in the system a step in the Metropolis algorithm conventionally consists of proposing a spin to flip N times. This is also due to the correlation length of the algorithm (section 2.4). The Metropolis algorithm is listed by algorithm 3.

```
1 while Stopping criteria not met do
2     repeat
3         Select a spin at random. The spin transitions $\mu$ to $\nu$.
4         Accept flipping of the spin with probability
          $A(\mu \to \nu) = \min\left(1, e^{-\beta(H(\nu)-H(\mu))}\right)$.
5     until N times;
6 end
```

Algorithm 3: Metropolis algorithm

We already showed that the Metropolis algorithm satisfies the detailed balance equation (5), but to show that the Metropolis correctly simulates the Ising model we must also show that the Metropolis algorithm is ergodic. To this end suppose we want to reach state ν from μ and μ and ν differ by k spins. Then doing k single-spin flips flipping these exact spins transitions the system from μ to ν. Since each of these spin flips has a non-zero probability, transitioning from μ to ν has a non-zero probability and thus we can reach state ν from state μ in a finite amount of steps.

The above argument shows that to prove that a Monte Carlo method on an Ising model is ergodic, it suffices to only prove that the algorithm can do any single-spin flip with a non-zero probability. Then by the above argument the algorithm is ergodic.

Although the Metropolis algorithm simulates the Ising model correctly, it has some major drawbacks. When the temperature is close to the critical temperature the Metropolis algorithm suffers from critical slowing down. That means that the time it takes for steps of the Metropolis algorithm to decorrelate is larger. This implies that around the critical temperature the Metropolis algorithm needs more time to simulate to achieve the same error of the average results as the error of the results with the temperature not in the neighbourhood of the critical temperature. Cluster algorithms are a way to partially avoid the problem of critical slowing down.

Note that the acceptance probability of the Metropolis algorithm is only determined by the bonds connected to the selected spin. This means that the Metropolis algorithm can simulate Ising models with random bond induced disorder. However, random bond Ising models tend to have meta-stable states. This means that for a low temperature the probability of the Metropolis algorithm to get out of a meta-stable state decreases exponentially.

2.3.3 Wolff Algorithm

One of the most well known cluster algorithm is the Wolff Algorithm [8], which is an improvement of the Swedsen-Wang algorithm [7]. Like the Metropolis algorithm, the Wolff algorithm is an algorithm on the standard Ising model without disorder. Instead of using single-spin flips, the Wolff algorithm flips clusters of spins. A cluster is created and grown by selecting each of the neighbours with the same spin of the cluster with some probability.

To find the acceptance probability and selection probability let us suppose we have an Ising model without disorder consisting of N spins. Also suppose that during one iteration a cluster is selected which will transition the system from μ to ν. The key observation is that flipping the cluster will only affect the bonds at the edge of the cluster. So let us assume that if the cluster is flipped is will satisfy n bonds and it will break k bonds. Each bond which is satisfied decreases the energy of the system by $2J$ while each bond which is broken increases the energy by $2J$. Thus the difference in the energy between μ and ν is $2J(k-n)$.

For the selection of the cluster which brings the system from μ to ν we allow the cluster to contain only spins with the same spin value. Since the cluster breaks k bonds, these k bonds correspond to k spins outside the cluster with the same spin value as the spins inside the cluster. Assuming all the spins are added to the cluster with an equal probability $P_{\textsf{add}}$, not selecting the k spins outside of the clusters has a probability of $(1-P_{\textsf{add}})^k$. To transition the system back from ν to μ we must use the same cluster. Since the cluster from μ to ν satisfies n bonds, the cluster from ν to μ breaks n bonds. Again, assuming all spins are added with equal probability $P_{\textsf{add}}$ the probability for not selecting these n spins in the cluster in ν is $(1-P_{\textsf{add}})^n$.

Using these values the selection probabilities reduce to $g(\mu \to \nu) = P(1-P_{\textsf{add}})^k$ and $g(\nu \to \mu) = P(1-P_{\textsf{add}})^n$. Here P is the probability of selecting the spins in the cluster. Crucial to note is that the clusters for both transitions are equivalent implying that the value of P is equal for both $g(\mu \to \nu)$ and $g(\nu \to \mu)$. We can now reduce the detailed balance equation (5)

$$\begin{aligned}\frac{g(\mu \to \nu)}{g(\nu \to \mu)}\frac{A(\mu \to \nu)}{A(\nu \to \mu)} &= e^{-\beta(\mathcal{H}(\nu)-\mathcal{H}(\mu))}\,,\\ (1-P_{\textsf{add}})^{k-n}\frac{A(\mu \to \nu)}{A(\nu \to \mu)} &= e^{-2\beta J(n-k)}\,,\\ \frac{A(\mu \to \nu)}{A(\nu \to \nu)} &= \left((1-P_{\textsf{add}})\,e^{2\beta J}\right)^{n-k}\,.\end{aligned}$$

We observe that if we choose $P_{\textsf{add}} = 1 - e^{-2\beta J}$ the right hand side of the equation becomes unity, regardless of temperature or spin configuration. If we now give both acceptance probabilities the value 1 a selected cluster of spins will always be flipped.

The full Wolff algorithm is listed in algorithm 4.

```
1 while Stopping criteria not met do
2     Select a spin at random.
3     Keep trying to add neighbours with the same spin value as the spins in
      the current cluster to the cluster with probability $P_{add}$. Spins which are
      already in the cluster are not added again and spins which have been
      rejected once can be still be added by another neighbour. This step is
      repeated as many times until all neighbours of the cluster are rejected.
4     Flip the cluster.
5 end
```

Algorithm 4: Wolff algorithm

We also have to prove that the Wolff algorithm is ergodic. Since not adding a spin to the cluster has a non-zero probability it is possible for the cluster to consist of only one spin. Thus the Wolff algorithm can do single-spin flips and by the argument we used at the Metropolis algorithm, the Wolff algorithm is ergodic.

The Wolff algorithm suffers less from critical slowing down as the Metropolis algorithm does due to the correlation length (section 2.4). However, the Wolff algorithm as presented above cannot be used to simulated RBIM's.

2.3.4 Niedermayer's Algorithm

The Wolff algorithm only works on standard Ising models without disored. To have a cluster algorithm which also works on various glassy spin systems and Ising models with non-homogeneous bonds a new algorithm was proposed by Niedermayer [9]. His proposed algorithm is an extension of the Wolff algorithm. In this section we explain how Niedermayer's algorithm works on a random bond Ising model where each bond has strength $J \pm \Delta J$ such that $\langle J_{ij} \rangle = J$.

The main difference between Niedermayer's algorithm and the Wolff algorithm is that in Niedermayer's algorithm the spins added to the cluster do not necessarily have the same spin value. Instead, we let $P_{\textsf{add}}$ depend on J_{ij}.

Suppose, by the same argument as with the Wolff algorithm, we have created a cluster transitioning from μ to ν. Like in the Wolff algorithm the selection probability from μ to ν only differs from the selection probability from ν to μ in the edges of the cluster. Since all bonds have a strength of either $J + \Delta J$ or $J - \Delta J$, we only have a view possibilities for each edge. Suppose when transitioning from μ to ν we satisfy n_+ amount of bonds with strength $J + \Delta J$ and n_- amount of bonds with strength $J - \Delta J$. Also suppose we break k_+ amount of bonds with strength $J + \Delta J$ and k_- amount of bonds with strength $J - \Delta J$. Then the probability of not selecting the bonds at the edges of the cluster in μ is equal to

$$\begin{aligned}(1 - P_{\mathsf{add}}(J + \Delta J))^{n_+}(1 - P_{\mathsf{add}}(J - \Delta J))^{n_-}\cdot\\ (1 - P_{\mathsf{add}}(-J - \Delta J))^{k_+}(1 - P_{\mathsf{add}}(-J + \Delta J))^{k_-}\,,\end{aligned}$$

and not selecting the bonds at the edges of the cluster in ν is

$$\begin{aligned}(1 - P_{\mathsf{add}}(-J - \Delta J))^{n_+}(1 - P_{\mathsf{add}}(-J + \Delta J))^{n_-}\cdot\\ (1 - P_{\mathsf{add}}(J + \Delta J))^{k_+}(1 - P_{\mathsf{add}}(J - \Delta J))^{k_-}\,.\end{aligned}$$

The difference in energy of μ and ν is

$$\mathcal{H}(\nu) - \mathcal{H}(\mu) = 2J(n_+ + \Delta n_+ + n_- - \Delta n_- - k_+ - \Delta k_+ - k_- + \Delta k_-)\,,$$

hence the detailed balance equation (5) reduces to

$$\begin{aligned}\frac{A(\mu\to\nu)}{A(\nu\to\mu)} = \left(e^{2\beta(J+\Delta J)}\frac{1 - P_{\mathsf{add}}(-J - \Delta J)}{1 - P_{\mathsf{add}}(J + \Delta J)}\right)^{n_+ - k_+}\,,\\ \left(e^{2\beta(J-\Delta J)}\frac{1 - P_{\mathsf{add}}(-J + \Delta J)}{1 - P_{\mathsf{add}}(J - \Delta J)}\right)^{n_- - k_-}\,.\end{aligned}$$

If we make sure that

$$\begin{aligned}\frac{1 - P_{\mathsf{add}}(-J - \Delta J)}{1 - P_{\mathsf{add}}(J + \Delta J)} = e^{-2\beta(J+\Delta J)}\,,\\ \frac{1 - P_{\mathsf{add}}(-J + \Delta J)}{1 - P_{\mathsf{add}}(J - \Delta J)} = e^{-2\beta(J-\Delta J)}\,,\end{aligned}$$

the acceptance ratio becomes unity. If we set $P_{\mathsf{add}}(J_{ij}) = 1 - e^{\beta(J_{ij} - E_0)}$ we indeed have satisfied both of the above equations. Here E_0 is a free parameter. $P_{\mathsf{add}}(J_{ij})$ must be a probability meaning it takes values between 0 and 1, so the best expression for $P_{\mathsf{add}}(J_{ij})$ is

$$P_{\mathsf{add}}(J_{ij}) = \begin{cases} 1 - e^{\beta(E - E_0)} & \text{if } J_{ij} \le E_0\,,\\ 0 & \text{otherwise}\,.\end{cases}$$

The only thing that remains is the behaviour of the algorithm for different E_0. There are few possibilities for E_0:

- $E_0 \geq J + \Delta J$: In this case the acceptance ratio will always be unity. Using different values for E_0 we can vary the size of the different clusters. If E_0 becomes larger P_{add} also increases and thus clusters grow in size.

- $-J - \Delta J \leq E_0 \leq J + \Delta J$: In this case the acceptance ration is not unity and we cannot choose the acceptance probability equal to unity. As E_0 decreases the clusters become smaller but at the expense of an acceptance ratio which decreases exponentially for increases in the energy of the system.

- $E_0 \leq -J - \Delta J$: Now $P_{\text{add}} = 0$ so all clusters consist of only one spin and the acceptance ratio is equal to the acceptance ratio of the Metropolis algorithm.

Niedermayer's algorithm includes both the Wolff algorithm and the Metropolis algorithm but it is also an extension of both algorithms that work on RBIM's. Unfortunately there is not much known about the behaviour of the algorithm for different values of E_0 [27].

2.4 Autocorrelation

In this section we give the definition of the autocorrelation and we show why different algorithms have different autocorrelations.

Each Monte Carlo algorithm on the Ising model consists of repeating a certain number of steps. In each step an update of the system is proposed and then either accepted or declined. If one is interested in a certain observable O one could for example measure O after each Monte Carlo step. This might not always be the best strategy since two states can be correlated, meaning that the two states are not statistically independent. This results in a correlated value of the observable O since certain states are over-sampled in the algorithm. For example, when using the Metropolis algorithm on a large lattice, two consecutive states can only differ in one spin meaning that the magnetization and Hamiltonian of both states are nearly identical.

To measure the correlation of states, one can measure the autocorrelation $C_O(t)$ of observable O which is defined by

$$C_O(t) = \langle O(t)O(0)\rangle - \langle O(0)\rangle^2 .$$

Here t is the time after $t = 0$. $O(t)$ is the value of the observable O at time t. In the autocorrelation the time can be measured in real time or in Monte Carlo time steps. If the Monte Carlo steps separated by time t are uncorrelated $C_O(t)$ should be of order $\mathcal{O}\left(M^{-1/2}\right)$ where M is the number of steps used to measure O.

For an infinite-size standard Ising model without disorder the correlation time τ, the time it takes for the system to decorrelated, diverges by [28]

$$\tau \sim \xi^z \sim \frac{1}{|\beta - \beta_c|^z} .$$

Here ξ is the correlation length, the length for which spins have to be apart to not be correlated. The parameter z is called the dynamical critical exponent. The above equation for τ is called critical slowing down. Note that the critical slowing down is connected to the autocorrelation of O.

For the Metropolis algorithm we know that $z \approx 2.1$ [28]. Since for a finite system size ξ cannot be larger than the system size, we have that $\xi \sim L$. This means that $\tau \sim L^{2.1} \approx N$. Since the time of the Metropolis algorithm scales with N, the time to generate independent configurations scales like $N\tau \sim N^2 = L^4$.

A way to see why the Metropolis algorithm has $z \approx 2$ is to note that around the critical temperature the system has clusters of spins which are difficult to break. For the system it is most likely that a change occurs by moving the entire cluster. Since one sweep of a Metropolis step changes the location of the cluster by a maximum of one site and since the moving of the cluster is a random walk for which the distance scales as $\sqrt{t}$, moving a cluster a distance of ξ takes $\tau \sim \xi^2$ steps.

This argument also suggest that to decrease the correlation times and thus to increase the number of uncorrelated measurements is to use non-local updates. Swendsen and Wang showed that for the Swendsen-Wang algorithm the steps scale with at most N and $z \approx 0.35$ [7]. Hence $N\tau \sim N^{1.175}$. The Wolff algorithm has the same statistical properties as the Swendsen-Wang algorithm so for the Wolff algorithm we also have $N\tau \sim N^{1.175}$.

In this thesis we use the autocorrelation to compare different algorithms. When measuring the domain wall a faster decorrelation of $C(t)$ means the algorithm achieves a faster decorrelation. Since one would like to use as many uncorrelated measurements as possible one strives to a faster decorrelation. We only measure the autocorrelation of the first fourier mode of the fourier transformation of the domain wall. This is due to the fact that the first mode corresponds with the length of the lattice and it is the fourier mode which is the least prone to local changes in the domain wall.

3 Weighed-loop algorithm

In this section we present a new weighed-loop algorithm for the simulation of random bond Ising models. We give a motivation and an accurate description of the algorithm. We prove that the algorithm satisfies the detailed balance equation and that the algorithm is ergodic. Both conditions are needed to simulate the model correctly. At the end we list advantages and disadvantages of the algorithm.

When one introduces disorder in the Ising model the two-fold degeneracy of the energy explodes into a highly degenerate energy landscape. The energy landscape also has the tendency to be rough. At low temperatures traditional Monte Carlo algorithms tend to get stuck in a meta-stable state. This is due to the fact that the meta-stable state is surrounded by energy barriers for which the probability of the algorithms to overcome the barriers decreases exponentially as the temperature decreases. In both single-spin flip algorithms and cluster algorithms the transition probabilities are dependent on all bonds and spins inside the cluster and neighbouring the cluster.

Loop algorithms have an advantage, namely being the fact that they only depend on the bonds on the loop and not on the bonds inside the cluster defined by the loop. In Figure 4 is an example of such a loop. In the figure straight bonds are ferromagnetic bonds while wiggled bonds are anti-ferromagnetic bonds. The loop is indicated by the dashed line. Each spin inside the loop has three satisfied bonds and one unsatisfied bond. This means that flipping each spin inside the loop individually increases the energy of the system. However, if all spins inside the loop are flipped simultaneously the energy of the system decreases.

The main idea of the weighed-loop algorithm is that bonds are chained iteratively until a closed loop of bonds is formed. The energy changed by flipping all spins in the cluster enclosed by the closed loop is only dependent on the bonds on the loop. If the algorithm has a bias towards unsatisfied bonds which are energetically favourable to swap the algorithm is able to transition out of meta-stable states much faster.

3.1 Description

The weighed-loop algorithm walks over the lattice of the system by chaining bonds. In other words, the algorithm walks a graph G of the lattice Λ of the Ising model. Each spin of the lattice represents a face of the graph and each separation between two spins represents an edge. Each of the different edges thus correspond to a different bond. The points where the edges meet are nodes of G. The weighed-loop algorithm then chains edges of G until a cycle in G is created. We call the chained bonds a loop. A loop now defines a cluster of spins enclosed by the edges of the cycle in G. Note that for each loop there are two possible clusters of spins which are enclosed by the loop. An example of a loop is in Figure 4.

Using G, each step of the algorithm starts by selecting a starting node n_s of G at random with probability $1/N$. Given that the algorithm cannot walk back the algorithm has three possible edges to chain to the current loop. Each edge

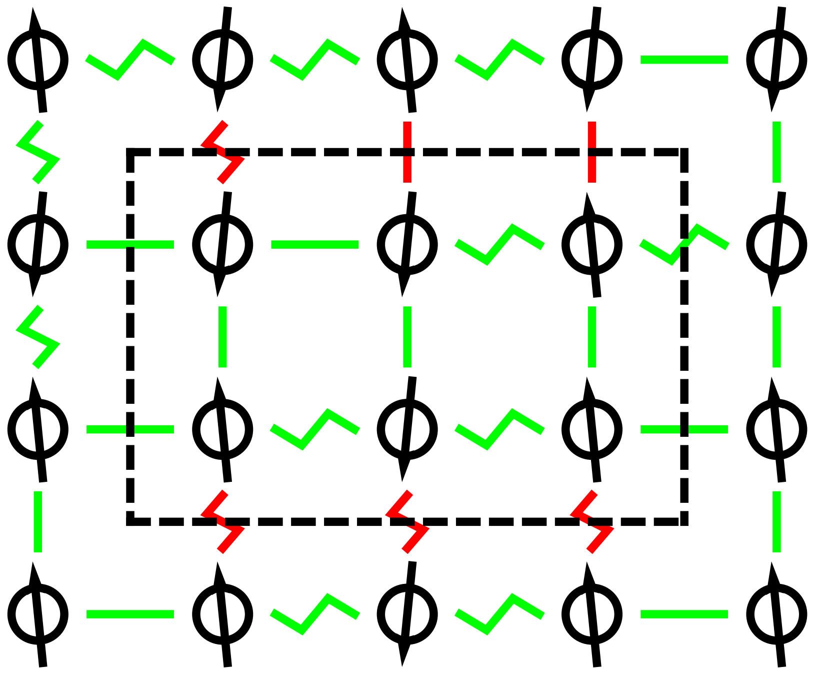

Figure 4: An example of a loop in a part of a typical spin configuration of a RBIM. The red bonds are unsatisfied bonds and the green bonds are satisfied. Squiggly lines represent anti-ferromagnetic bonds while straight lines represent ferromagnetic bonds. Flipping each spin independent in the cluster increases the energy of the system, but flipping the entire cluster decreases the energy of the system. Although we have restricted ourself to not include anti-ferromagnetic bonds ($\Delta \leq 1$) the same argument still applies. Note that flipping the spins inside the loop results in the same change in energy as flipping the spins outside the loop.

corresponds to a bond J_{ij} and the probability of choosing an edge has a weight $W(J_{ij})$. The algorithm now continues chaining edges. The chaining of edges continues until the algorithm visits an already visited node of G. Since there are a finite amount of edges in G, creating a loop takes a finite time. The constructed loop will consist of a cycle and possibly a tail section. For example, the example loop in Figure 4 does not have a tail section. The cluster of spins is now defined by the cycle while the tail section may be reused for a next loop.

The weight W of the selection step is modelled after the Boltzmann distribution (1) and is given by

$$W(J_{ij}) = e^{-\beta J_{ij}\sigma_i\sigma_j}$$

Using this weight the algorithm has a bias toward unsatisfied bonds over satisfied bonds. Note that if $\beta = 0$ the chaining of edges reduces to a random walk on G

while if $\beta \to \infty$ the algorithm always chains an unsatisfied bond if possible.

Since we simulate on an Ising model with periodic and anti-periodic boundary conditions, it is possible for the constructed loop to wrap around one or both of the edges. Since the lattice Λ is topological equivalent to a Klein bottle, the cluster enclosed by a loop which wraps an odd amount of times around one of the edges is the entire lattice. This results in the flipping of all the spins in the system which does not change the energy. This means that time is wasted on a loop which does change the domain wall. To solve this problem, the algorithm creates a second loop every time a loop wraps around one of edges an odd amount of times. If this second loop also wraps around the same edge an odd amount of times, the cluster of spins is defined as the cluster of spins between the two loops. If the second loop does not wrap an odd amount of times around the same edge as the first loop, the first loop is discarded and the second loop is used instead.

To prove that the algorithm correctly simulates the Ising model we have to prove the algorithm to be ergodic and prove that the algorithm satisfies the detailed balance condition. Showing that the algorithm is able to do single-spin flips proves ergodicity. Since it is possible for the loop to select a single-spin and since the probability of flipping this spin is also non-zero (see section 3.2), the algorithm can perform single-spin flips and thus is ergodic.

3.2 Acceptance probabilities

Given a cluster of spins enclosed by a loop λ that transitions the system from μ to ν, we denote the probability of selecting λ in μ as $g_\mu(\lambda)$ and the acceptance probability of accepting λ in μ as $A_\mu(\lambda)$. If the cluster is defined by two loops we denote λ as the intersection of both loops. For the loops we have that $g_\mu(\lambda)$ is simply the product of the stochastic choices the algorithm made during the chaining of bonds to create the loop. Note that λ is one of the many possible loops transitioning μ to ν. As an example, reversing the direction of the cycle in the loop or removing the tail section both transitions the system from μ to ν. If we call the set of all loops which transition the system from μ to ν as $\Theta_{\mu\to\nu}$ we have that

$$\Pi(\mu \to \nu) = \sum_{\lambda \in \Theta_{\mu\to\nu}} g_\mu(\lambda) A_\mu(\lambda)\,.$$

We also denote the set of all loops which transition the system from ν to μ as $\Theta_{\nu\to\mu}$, the probability of selection λ in ν as $g_\nu(\lambda)$ and the probability of accepting λ in ν as $A_\nu(\lambda)$ as the acceptance probability. Then we have equivalently

$$\Pi(\nu \to \mu) = \sum_{\lambda \in \Theta_{\nu\to\mu}} g_\nu(\lambda) A_\nu(\lambda)\,.$$

The key observation is that for every loop $\lambda \in \Theta_{\mu\to\nu}$ we have that $\lambda \in \Theta_{\nu\to\mu}$ and vice versa. This means that summing over all loops in $\Theta_{\nu\to\mu}$ yields the same result as summing over all loops in $\Theta_{\mu\to\nu}$. Using this property, the detailed balance condition now becomes

$$
\begin{aligned}
P(\mu)\Pi(\mu \to \nu) &= P(\nu)\Pi(\nu \to \mu) \\
P(\mu) \sum_{\lambda \in \Theta_{\mu\to\nu}} g_\mu(\lambda) A_\mu(\lambda) &= P(\nu) \sum_{\lambda \in \Theta_{\nu\to\mu}} g_\nu(\lambda) A_\nu(\lambda) \\
P(\mu) \sum_{\lambda \in \Theta_{\mu\to\nu}} g_\mu(\lambda) A_\mu(\lambda) &= P(\nu) \sum_{\lambda \in \Theta_{\mu\to\nu}} g_\nu(\lambda) A_\nu(\lambda)\,.
\end{aligned}
$$

If we define the acceptance probabilities as

$$
\begin{aligned}
A_\mu(\lambda) &= \min\left(1, e^{-\beta(\mathcal{H}(\mu)-\mathcal{H}(\nu))} \frac{g_\nu(\lambda)}{g_\mu(\lambda)}\right) \\
A_\nu(\lambda) &= \min\left(1, e^{-\beta(\mathcal{H}(\mu)-\mathcal{H}(\nu))} \frac{g_\mu(\lambda)}{g_\nu(\lambda)}\right),
\end{aligned}
$$

the detailed balance condition reduces to

$$
\begin{aligned}
P(\mu) \sum_{\lambda \in \Theta_{\mu\to\nu}} g_\mu(\lambda) A_\mu(\lambda) &= P(\nu) \sum_{\lambda \in \Theta_{\mu\to\nu}} g_\nu(\lambda) A_\nu(\lambda) \\
P(\mu) \sum_{\lambda \in \Theta_{\mu\to\nu}} g_\mu(\lambda) \min\left(1, e^{-\beta(\mathcal{H}(\mu)-\mathcal{H}(\nu))} \frac{g_\nu(\lambda)}{g_\mu(\lambda)}\right) &= \\
P(\nu) \sum_{\lambda \in \Theta_{\mu\to\nu}} g_\nu(\lambda) &\min\left(1, e^{-\beta(\mathcal{H}(\mu)-\mathcal{H}(\nu))} \frac{g_\mu(\lambda)}{g_\nu(\lambda)}\right) \\
\sum_{\lambda \in \Theta_{\mu\to\nu}} \min\left(P(\mu)g_\mu(\lambda), P(\nu)g_\nu(\lambda)\right) &= \sum_{\lambda \in \Theta_{\mu\to\nu}} \min\left(P(\nu)g_\nu(\lambda), P(\mu)g_\mu(\lambda)\right).
\end{aligned}
$$

Since the last equation is always valid we have satisfied the detailed balance condition using the acceptance probabilities as defined above.

Spins adjacent to the tail of a loop are not flipped when flipping the cluster of spins. So the acceptance probability for the tail section of a loop λ in μ is the same as the acceptance probability of the same loop in ν. Hence, in calculating the selection probability, one only needs to calculate the selection probability of the cycle. This also means that when reusing the tail section of a previous loop as the beginning of a next loop, the detailed balance is still satisfied. To explain this in another way, as stated above, the tail section does not contribute to the acceptance probability at all. Hence a closed cycle in μ always has the same acceptance probability, regardless of the tail which led to the construction of that particular cycle.

3.3 Advantages and disadvantages

The main advantage of the weighed-loop algorithm is that it is a loop algorithm. This implies that each step the probability of flipping a cluster of spins only depends on the bonds at the edges of the cluster. In contrast, other cluster type algorithms or single-spin flip algorithms depend on all the bonds inside the cluster. The latter results in a exponential decrease probability of flipping energetically unfavourable clusters as the temperature decreases.

The weighed-loop algorithm is flexible in the function for determining the weights of the stochastic choices for the chaining of the bonds. There are no requirements for the function for the weights except that the algorithm must retain ergodicity. This means that one could use almost any possible function and still satisfy the detailed balance condition. The weights function used in this thesis is only dependent on the contribution to the energy of bonds. However, one can add preferred directions to the weights function by adjusting the weights of optional bonds in different directions. For example, if one increases the odds for straight paths the sizes of the clusters enclosed by loops would increase while if one increases the odds for right turns the clusters enclosed by loops would decrease.

Another advantage of the weighed-loop algorithm is with respect to simulating domain walls. With the weights function presented in this thesis the algorithm favours unsatisfied bonds over satisfied bonds hence the loop is inclined to walk over the domain wall if given the opportunity. This means that a relative large fraction of the spin flips will happen close to the domain wall and thus less time is wasted on loops not updating the domain wall. Adding a preferred direction when the loop diverges from the domain wall to increase the likelihood of the loop returning to the domain wall will increase the flips close to the domain wall.

The main disadvantage of the algorithm is the complexity in the code and the running time. While most Monte Carlo algorithms are fairly simple to implement, the weighed-loop algorithm is more complex to implement. This also means that the algorithm is more prone to errors. When a loop is created one cannot know a priori which of the two clusters created by the loop is the smallest cluster. This means that to flip the cluster one must the algorithm must consider all spins in the entire lattice. This also means that this part is subject to scaling effects of the lattice size. The construction of the loop is also subject to scaling effects. Since for low temperatures the loops will walk along the domain wall a large amount of the time, a relative large amount of computing time is wasted on constructing loops which do not change the physics of the system.

4 Results

In this section we present several results obtained using the weighed-loop algorithm. All simulations were run on a single core of a Intel(R) Xeon(R) CPU E5-1620 and 32G RAM. For all the simulations we can set, without loss of generality, the bond strength $J = 1$. The latter is valid since in all the usages of the weighed-loop algorithm β is multiplied by J and thus we can view βJ as a single variable. In other words we can set $J = 1$ and vary β instead of varying both β and J. All simulations were run on a L by L random bond Ising model with anti-periodic boundary conditions along one border and periodic boundary conditions along the other border, unless specified otherwise. We start each simulation with exactly one domain wall. Each domain wall costs a macroscopic amount of energy so by initiating one domain wall we always have exactly one domain wall in the system below the critical temperature.

First, we used simulations to investigate if it is more efficient to use wrapped loops instead of discarding these constructed loops. In the next simulations we looked at the effect of re-using the tails from the previous loops on the autocorrelation of the first mode of the fourier transformation. Next we recorded the average loop sizes and the acceptance probabilities for different system sizes L and different temperatures β and we compared the results. Next we induce disorder in the Ising model and using simulations of both Niedermayer's algorithm and the weighed-loop algorithm we can compare the performance of both algorithms given different parameters in the parameter space consisting of L, β and Δ to derive a conclusion on which algorithm is the best choice for different regions. Without disorder we measured the thermal averaging of the structure factor of the domain wall for different temperatures and different lattice sizes and we show that these results agree with the theory. Varying the different parameters we measured the typical length scale L_c at which the system switches from a thermal dominated roughness to higher length scales at which roughness is determined by disorder.

4.1 Using wrapped loops

When a loop wraps around one of the edges of the lattice an odd amount of times, flipping the cluster of spins enclosed by the loop simply results in the entire lattice being flipped. To solve this problem, a second loop can be constructed which also wraps around the same edge as the first loop. However, constructing this second loop takes computational time and it might be more efficient to not use loops which wrap around of the edges whenever they occur.

To test what is the most efficient in terms of computational time, we ran simulations with using the wrapped loops and with not using the wrapped loops and we measured the autocorrelation $C(t)$ of the first mode of the fourier transformation. For both algorithms we ran 10 simulations, both algorithms were run on the same 10 disorder configurations. We used the parameters $L = 64$, $\Delta = 0.15$ and $\beta = 2$. For small t we assume an exponential decay in the autocorrelation, that is $C(t) \sim \exp(-at)$ with equal prefactors for both algorithms. An example of the autocorrelation of the simulation for one specific disorder configuration is shown in

Figure 5.

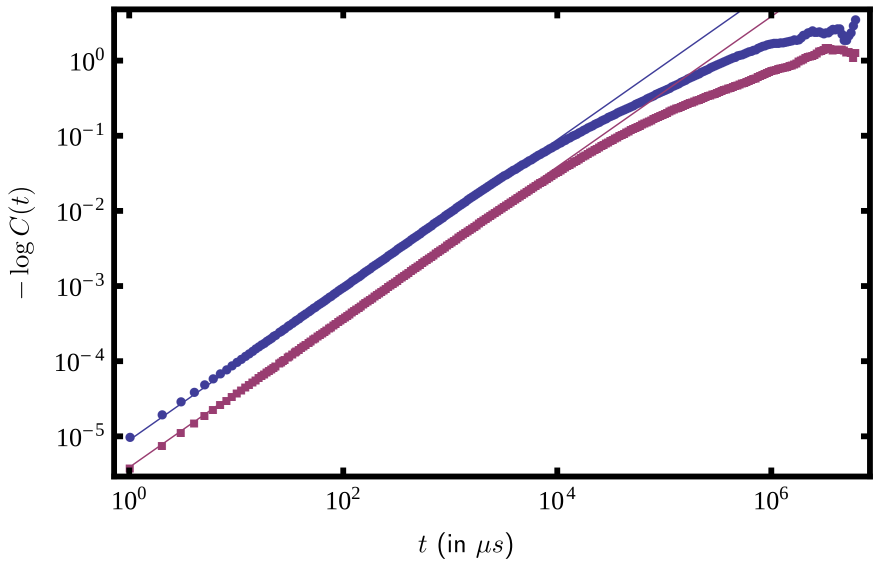

Figure 5: The negative $\log$ of the autocorrelation of the first mode of the fourier coefficients $C(t)$ for both using the wrapped loops (blue disks) and not using the wrapped loops (purple squares) on a log-log plot for the values $L = 64$, $\Delta = 0.15$ and $\beta = 2$ as a function of real computing time t. The decorrelation occurs exponentially at first after which a slowing down occurs. The best unweighed exponential fits up to $t = 10^3$ are shown as corresponding straight lines. In this particular plot the result from using the wrapped loops are faster decorrelated that the results from not using the wrapped loops.

To compare the results we fitted an exponential function to the data up to $t = 10^3$. From the fits we extracted the constant in the exponent a for the different simulations and averaged the result over the different disorder configurations. A larger value for a leads to a faster decorrelation. We found that $a = 3.6(6) \cdot 10^{-6}$ for the simulations without using wrapped loops and $a = 1.5(2) \cdot 10^{-5}$ for the simulations with using wrapped loops. From this we conclude that using wrapped loops leads to a faster decorrelation compared with without using wrapped loops. Hence, it is computational more efficient to use wrapped loops.

4.2 Reusing the tails

Re-using the tails in the weighed-loop algorithm for the construction of the next loop is a method that decrease the time wasted on the construction of loops by making sure that every selected bond will be used at some time in the algorithm. One expects that the different simulated systems are now more correlated than without reusing the tails so the question is whether the gain in computational time is beneficial compared to the increase in correlation.

To compare the two different algorithms we created 10 runs with both reusing the tails and not reusing the tails. Each simulation was run on a disorder configuration with $L = 64$, $\Delta = 0.15$ and $\beta = 2$. Both algorithms were simulated on the same 10 disorder configurations. From these simulations we measured the autocorrelation $C(t)$ in the first mode of the fourier transformation. As with the results on wrapping the loops, we assume an exponential decay in the autocorrelation with equal prefactors for both algorithms, i.e. $C(t) \sim \exp(-at)$. An example of the results of one particular simulation of one specific disorder configuration is in Figure 6.

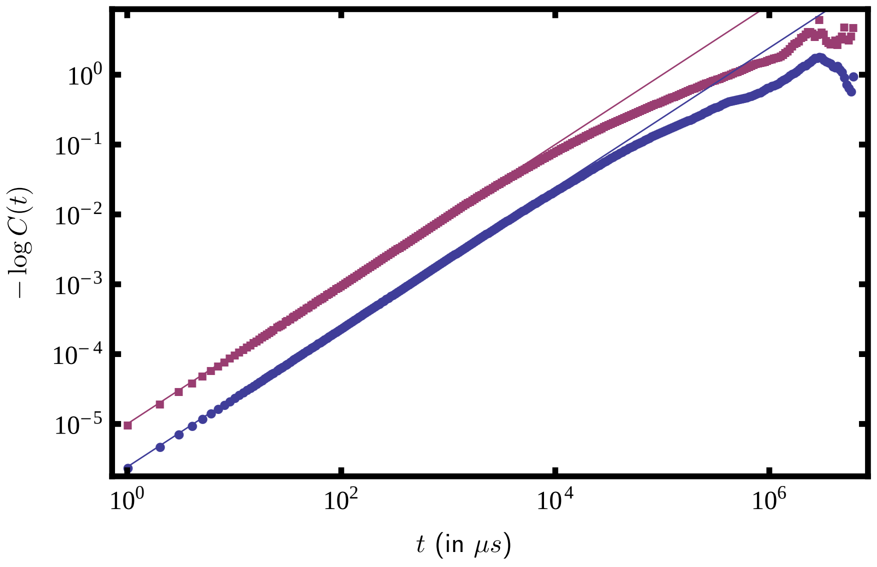

Figure 6: The negative $\log$ of the autocorrelation of the first mode of the fourier coefficients $C(t)$ for both not reusing the tail (blue disks) and reusing the tail (purple squares) is shown in a log-log plot for the values $L = 64$, $\Delta = 0.15$ and $\beta = 2$ as a function of real computing time t. The decorrelation occurs exponentially at first after which a slowing down occurs. The best unweighed exponential fits up to $t = 10^3$ are shown as corresponding straight lines. In this figure the simulations with reusing the tail are faster uncorrelated that the simulations without reusing the tail.

All other simulations for both reusing the tail and not reusing the tail show the same shape for the autocorrelation as in Figure 6, first an exponential decorrelation occurs after which a slowing down occurs. To compare the two algorithms we fitted an unweighed exponential function to the data up to $t = 10^3$ from which we deduced the constant in the exponent a for both algorithms and we averaged the results over the disorder configurations. Using this we obtained $a = 1.2(2) \cdot 10^{-5}$ for not reusing the tail and $a = 4.1(7) \cdot 10^{-6}$. The value of a is larger without reusing the tail. This implies that on average not reusing the tail results in faster decorrelation compared to using the tail. From this we conclude that it is computational more efficient to discard the tail sections from loops.

4.3 Acceptance probability and loop size

It is interesting to look at the acceptance probabilities and loop sizes of the loops created by the weighed-loop algorithm. To this end we simulated the Ising model for different lattice sizes L and for different temperatures β with a disorder $\Delta = 0.15$. We measured the average length of the cycles of the accepted loops and the fraction of accepted loops to the total constructed loops. When a cluster is selected using two loops, the length is the sum of the lengths of the two individual loops. The data of the average length and the fraction of accepted loops are plotted in figure Figure 7 and Figure 8 respectively.

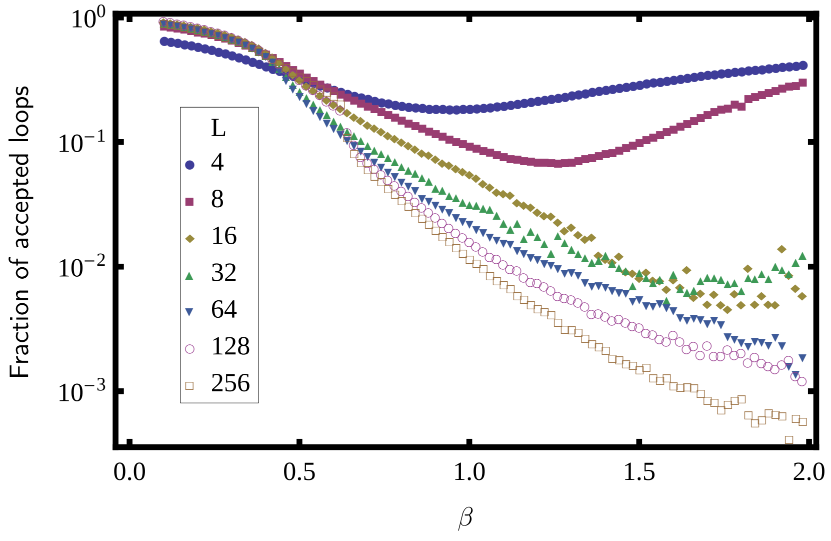

Figure 7: The fraction of accepted loops to the total constructed loops for different lattice sizes L and different temperatures β using a disorder of $\Delta = 0.15$ on a log scale. From the plot one can see that the fraction of accepted loops decreases for lower temperatures, however for smaller lattice sizes the fraction of accepted loops increases after the decrease for lower temperatures. For large lattice sizes the fraction shows a 'jump' around $\beta = 0.64$.

From Figure 7 we observe that the fraction of accepted loops decreases as the temperature decreases. This can be attributed to the construction of the loops. During the chaining of the bonds, the algorithm is presented with three

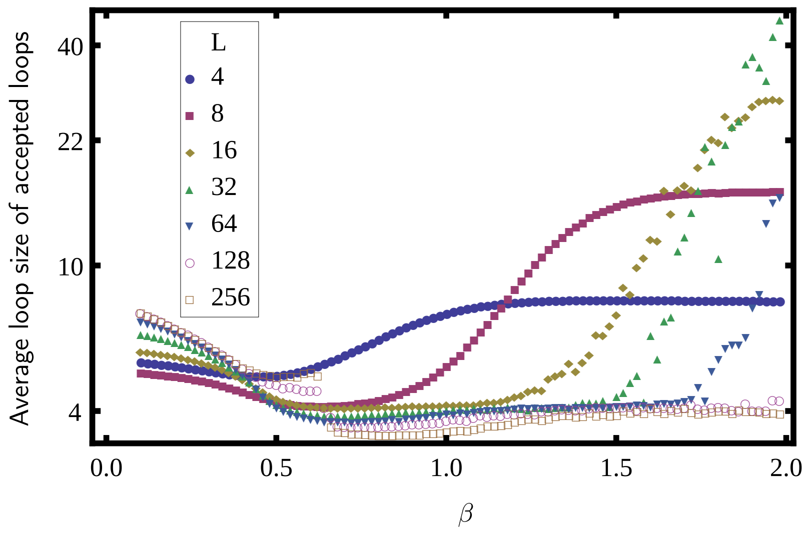

Figure 8: The average size of the accepted loops for different lattice sizes L and different temperatures β using a disorder of $\Delta = 0.15$ on a log scale. Lowering the temperature decreases the average size of the loops but further decreasing the temperature increases the average size of the loops. Large lattice sizes show a 'jump' around $\beta = 0.64$. The sizes for $L = 4$ and $L = 8$ converge to 8 and 16 respectively, meaning that for low temperature two loops are used each time, each having a total length of $2L$.

bonds to choose from, each bond having a weight equal to the Boltzmann factor $W(J_{ij}) = e^{-\beta J_{ij}\sigma_i\sigma_j}$. This means that for decreasing temperatures the likelihood of choosing satisfied bonds over unsatisfied bonds decreases exponentially. So, when transitioning from μ to ν, the selection probability in μ increases while the selection probability in ν decreases and thus the acceptance probability decreases. From the results in Figure 7 we observe that it is indeed the case that for lower temperature the fraction of accepted loops decreases. For small lattice sizes however, the fraction of accepted loops increases when the temperature decreases. This can be explained due to the fact that the size of the loops also increases since the loop is more likely to walk along the domain wall. With a larger loop size it is possible to have a large negative contribution to the energy and thus a higher acceptance probability. It is also possible that two loops are constructed which are exactly the same and both lie on the domain wall, meaning that the acceptance probability is unity. We can also observe this behaviour from Figure 8.

A remarkable observation is that in the simulations for $L = 128$ and $L = 256$ both the fraction of accepted loops and the size of the loops show a jump in the data around temperature $\beta = 0.64$. To further investigate this jump we also simulated the lattice sizes $L = 90$, $L = 100$, $L = 110$ and $L = 120$. These results are in

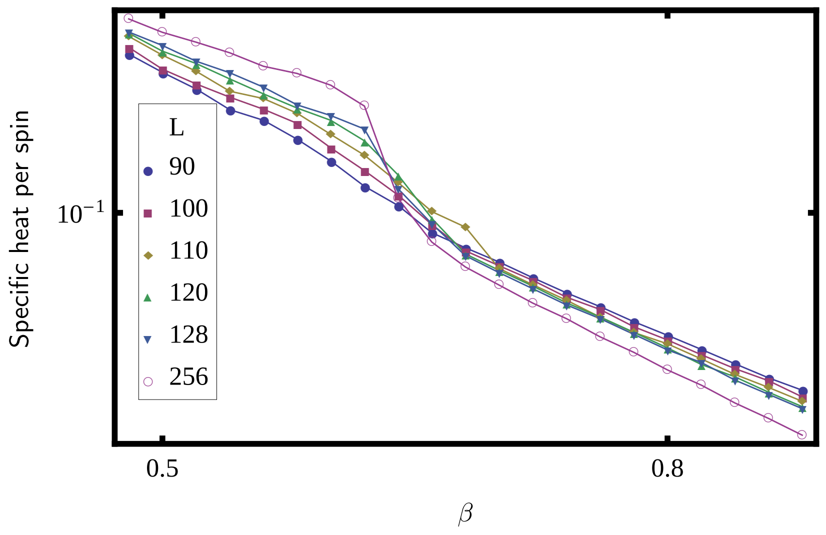

Figure 9: A close-up of the jump in the fraction of accepted loops to the total constructed loops on a log scale for different lattice sizes L and for different temperatures β and with a disorder $\Delta = 0.15$. We added lines between data points for visibility. This plot shows that the size of the jump increases when the lattice size increases.

Figure 9. From the figure we observe that the jump becomes more explicit for larger lattice sizes. This can be explained by the specific heat.

In Figure 10 the specific heat per spin for different lattice sizes L is plotted for different temperatures β. From the figure it is clear that for larger lattice sizes the specific heat drops sharper around the $\beta = 0.64$ mark, since the phase transition of the system occures at this temperature. By the definition of the acceptance probability of the loops, a larger specific heat amounts to a larger acceptance rate and due to the definition of the chaining of bonds in the loop a larger specific heat amounts to larger loop sizes. Thus a sudden drop in the specific heat results in a sudden drop in the loop sizes and in the fraction of accepted loops.

4.4 Comparison to Niedermayer's algorithm

We are interested in how the weighed-loop algorithms compares to other already existing algorithms. To this end we compared the weighed-loop algorithm with a known algorithm for disordered Ising models, Niedermayer's algorithm. For both the weighed-loop algorithm and Niedermayer's algorithm we ran multiple simulations for a fixed amount of time of 10 minutes each on the same 24 different disorder configurations for different values of L, β and Δ to find the regime in the parameter space where the weighed-loop algorithm outperforms Niedermayer's algorithm. We

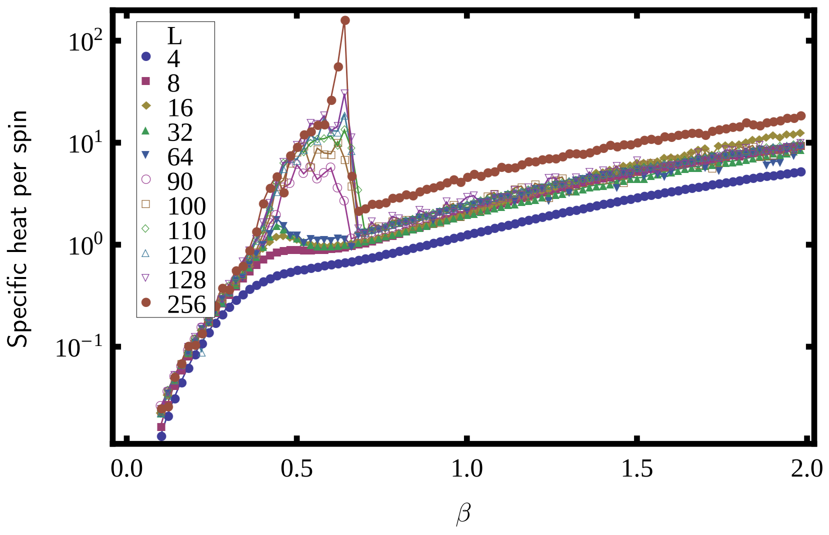

Figure 10: A plot of the specific heat per spin for different lattice sizes L and different temperatures β using a disorder of $\Delta = 0.15$ on a log-scale. We joined the data points for visibility. From the plot it is clear that the phase transition occurs around $\beta = 0.6$ and the specific heat per spin shows a sharp drop after the phase transition.

measured the autocorrelation of the first mode of the fourier transformation $C(t)$ since the first mode takes the most time to reach the equilibrium.

For small time t we can assume an exponential decay in the autocorrelation with equal prefactors such that $C(t) \sim \exp(-at)$ for both Niedermayer's algorithm and the weighed-loop algorithm. To compare the different algorithms we fitted the exponential function to the data up to $t = 10^3$ for both algorithms. From the fit we extracted the constant in the exponent a and averaging the values of a over the disorder configurations. Figure 11 shows typical autocorrelations for both algorithms on the same disorder configuration. The averaged values of a are listen in Table 1.

From the data we can conclude that for the temperature $\beta = 2$ there is a larger regime where the weighed-loop algorithm achieves faster decorrelation than Niedermayer's algorithm. However, for larger lattice sizes the value of the disorder for which the weighed-loop algorithm decorrelates faster than Niedermayer's algorithm also increases. This means that the weighed-loop algorithm is the superior algorithm in the relative small lattice sizes as well as a larger disorder on larger lattice sizes. In short, the weighed-loop algorithm has a faster decreasing autocorrelation in the regions where conventional algorithms get stuck.

From the data we can also observe that the weighed-loop algorithm decorrelates exponentially on longer time scales. This can be an indication that the weighed-loop algorithm simulates more different local minimums which are far apart, meaning

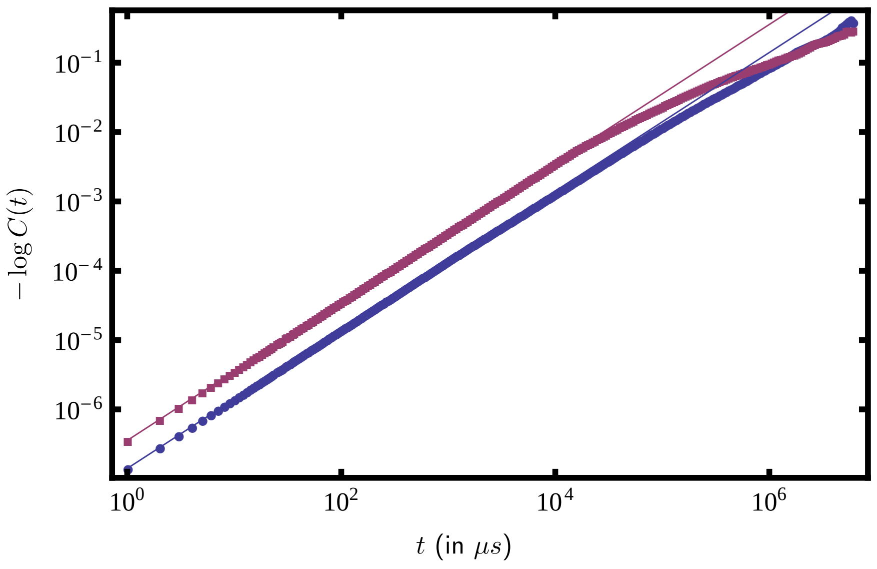

Figure 11: The negative $\log$ of the autocorrelation of the first mode of the fourier coefficients $C(t)$ for the weighed-loop algorithm (blue disks) and Niedermayer's algorithm (purple disks) on a log-log plot for the values $L = 128$, $\Delta = 0.50$ and $\beta = 1$ as a function of real computing time t. The decorrelation occurs exponentially at first after which a slowing down occurs. The best unweighed exponential fits up to $t = 10^3$ are shown as corresponding straight lines. In this particular plot the result from Niedermayer's algorithm are faster decorrelated that the results from the weighed-loop algorithm.

that the weighed-loop algorithm is more efficient in changing local clusters to hop to different minimums in the energy spectrum.

4.5 Domain walls

The weighed-loop algorithm is mainly used to simulate domain walls in RBIM's. First we simulated a standard Ising model without random bond induced disorder and we measured the structure factors for different temperatures on different lattice size and we compare these simulated results to the theory from section 2.2. A 265x256 lattice and a 64x64 lattice were simulated using the weighed-loop algorithm with a disorder $\Delta = 0$ and the temperature β ranges from 0.5 to 3.5 with a step size of 0.1. For each different temperature β we get a set of $L/2$ fourier coefficients and we fit the fourier modes to the expression for the structure factor (3). From the fit we get the scaling factor $A(L, \beta J)$. We expect the values of the scaling factor to fit the theory, except in the high temperature region, since for high temperatures the approximation of the domain wall does not hold. In Figure 12 the values of $LA(L, \beta J)$ are plotted for

$\beta = 1,\ H = 32$	Niedermayer	Loop
$\Delta = 0.1$	$a = 6.5(2) \cdot 10^{-4}$	$a = 1.35(4) \cdot 10^{-5}$
$\Delta = 0.3$	$a = 1.8(6) \cdot 10^{-5}$	$a = 1.42(8) \cdot 10^{-5}$
$\Delta = 0.5$	$a = 2.1(3) \cdot 10^{-5}$	$a = 1.3(2) \cdot 10^{-5}$
$\beta = 1,\ H = 64$	Niedermayer	Loop
$\Delta = 0.1$	$a = 9.4(9) \cdot 10^{-7}$	$a = 8.7(6) \cdot 10^{-7}$
$\Delta = 0.3$	$a = 1.4(1) \cdot 10^{-6}$	$a = 8.9(8) \cdot 10^{-7}$
$\Delta = 0.5$	$a = 1.0(1) \cdot 10^{-6}$	$a = 1.2(2) \cdot 10^{-6}$
$\beta = 1,\ H = 128$	Niedermayer	Loop
$\Delta = 0.1$	$a = 2.0(2) \cdot 10^{-7}$	$a = 4.5(4) \cdot 10^{-8}$
$\Delta = 0.3$	$a = 5.3(6) \cdot 10^{-7}$	$a = 8.1(7) \cdot 10^{-8}$
$\Delta = 0.5$	$a = 6.4(5) \cdot 10^{-7}$	$a = 1.1(2) \cdot 10^{-7}$
$\beta = 2,\ H = 34$	Niedermayer	Loop
$\Delta = 0.1$	$a = 7.9(4) \cdot 10^{-5}$	$a = 9.4(6) \cdot 10^{-6}$
$\Delta = 0.3$	$a = 1.1(2) \cdot 10^{-6}$	$a = 1.1(1) \cdot 10^{-5}$
$\Delta = 0.5$	$a = 0.5(1) \cdot 10^{-6}$	$a = 1.3(3) \cdot 10^{-5}$
$\beta = 2,\ H = 64$	Niedermayer	Loop
$\Delta = 0.1$	$a = 2.6(3) \cdot 10^{-7}$	$a = 8.4(9) \cdot 10^{-7}$
$\Delta = 0.3$	$a = 4.7(5) \cdot 10^{-7}$	$a = 1.2(2) \cdot 10^{-6}$
$\Delta = 0.5$	$a = 1.9(2) \cdot 10^{-7}$	$a = 1.8(5) \cdot 10^{-6}$
$\beta = 2,\ H = 128$	Niedermayer	Loop
$\Delta = 0.1$	$a = 0.4(2) \cdot 10^{-6}$	$a = 6.1(8) \cdot 10^{-8}$
$\Delta = 0.3$	$a = 1.9(3) \cdot 10^{-7}$	$a = 1.3(2) \cdot 10^{-7}$
$\Delta = 0.5$	$a = 8.7(9) \cdot 10^{-8}$	$a = 2.5(3) \cdot 10^{-7}$

Table 1: The values of a for both Niedermayer's algorithm and the weighed loop algorithm for different values in the parameter space of L, β and Δ. For each set of parameter points the algorithms with the fastest decorrelation is coloured blue. The values of a are obtained by averaging the best fits for the first 120 data points for each data set.

different temperatures β with the theory plotted as reference in the corresponding colours.

From Figure 12 it is clear that the simulated data agrees with the theory as predicted. For the low and high temperature regimes of Figure 12 the data is less accurate. In the high temperature regime there exist more thermal fluctuations as well as more local pockets of spins and overhangs in the domain wall, which means that the approximation of the domain wall does not hold. For low temperatures the probability of accepting moves is lower which results in the domain wall changing

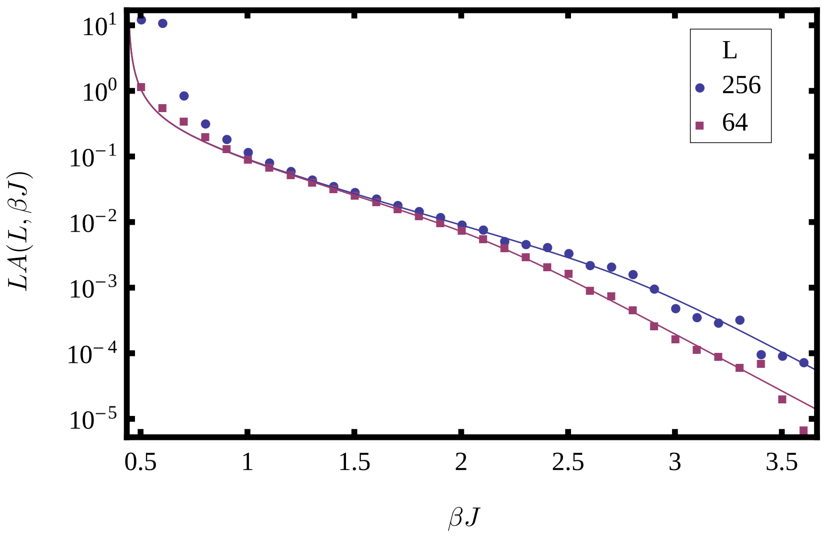

Figure 12: The simulated values of $LA(L, \beta J)$ are plotted against the temperature β for the values $L = 256$ (blue circles) and $L = 64$ (purple squares) on a log scale. The values of $A(L, \beta J)$ are calculated using the fit (6) with $\Delta = 0$. The plot also shows the theory (4) as a reference with the corresponding colours. The simulated values agree with the theory. For both the high and low temperature the simulated values are less accurate since for high temperatures the model of the domain wall is not accurate enough and for low temperatures the domain wall collapses to an equilibrium making it harder for the algorithm to change the domain wall.

less between different simulated steps in the algorithm.

Introducing disorder Δ in the system creates a parameter space consisting of the three parameters L, β and Δ. As described in section 2.2 the behaviour of the low modes is dominated by the random bond induced disordered while the behaviour of the high modes is dominated by the thermal fluctuations. We are interested in the typical length L_c scale for which the separation between the two dominated regions occurs. This Larkin length L_c and the crossover point relate to each other by $L_c = \frac{1}{2p_c}$.

To investigate the Larkin length for different regimes in the parameter space we performed three sets of simulations. In each set of simulations we fixed two of the three parameters L, β and Δ and varied the third parameter. The default values of the parameters we used are $L = 64$, $\beta = 2$ and $\Delta = 0.15$. We fit each set of fourier modes to

$$A(p, L, \beta J, \Delta) = A(L, \beta J, \Delta) \left(\left[c\left(L, \beta J, \Delta\right) p^{-7/3} \right]^b + \csc^{2b} \left[\frac{\pi p}{L} \right] \right)^{1/b}, \quad (6)$$

where $c(L, \beta J, \Delta)$ is a prefactor and b is a smoothing factor such that the function transition smoothly at the crossover point. Using $A(L, \beta J, \Delta)$ and $c(L, \beta J, \Delta)$ one is able to find the crossover point p_c and Larkin length L_c.

When introducing randomness Δ in the system simulating becomes more time inefficient. First of all the simulations need to be averaged across different disorder configurations, next to the standard thermal averaging. This means that more simulations are needed in order to achieve an accurate result. Also with disorder, there are more regions of local energy in the energy landscape due to an increase of roughness and meta-stable states in the energy landscape. This implies that more simulations are needed in order to simulate all different pockets to achieve an accurate averaged result. When simulating, one must therefore choose: either small systems with higher disorder or lower temperature, or larger systems with lower disorder or higher temperature.

4.5.1 Varying L

We are interested in the behaviour of the Larkin length when varying the lattice size L. Due to finite size effects we expect the Larkin length to increase for larger lattice sizes and converge to a certain typical length scale for infinite sized systems. To investigate the length scale we simulated different lattices sizes where $8 \leq L \leq 128$ with a step size of 8. We combined the different fourier modes and we fitted the modes unweighed to (6). However, for small lattice sizes, more simulations do not yield any results above the typical length scale and thus for small systems the fit is not possible since the higher-mode portion does not collapse. Hence we did not used any fourier modes for which $L \leq 40$ in the fit. The results of the simulations are in Figure 13. In Figure 13 we also fitted the best unweighed fit (6) as a solid black line as well as the power-laws in both the low and high mode regimes as dashed lines.

From Figure 13 we find the crossover point $p_c \approx 0.37$ which is plotted in the figure as a red dashed line, from which we can derive that $L_c \approx 13.4$. The fit also shows that the fourier modes align to the same curve, for large enough lattice sizes L. For the modes with small L this is not the case as we described earlier.

The fit was made using all fourier modes with $L \geq 48$ from which a Larkin length of $L_c \approx 13.4$ was obtained. However, if one would drop more fourier modes from the data sets of increasing lattice sizes L, a different length scale can be obtained. In Figure 14 is a plot of the Larkin length as a function of the lowest value of L used in the fit showing an increase in length scale when less fourier modes are used. This implies that the different sets of fourier modes collapse less onto each other. Since there is no clear limit to the different values of the Larkin length, we are unable to derive the Larkin length for infinite system sizes.

4.5.2 Varying Δ

Another parameter we are interested in for the behaviour of the Larkin length is the disorder parameter Δ. We expect an increase in the crossover point (or decrease in the Larkin length) for larger disorder since for a system with more disorder the

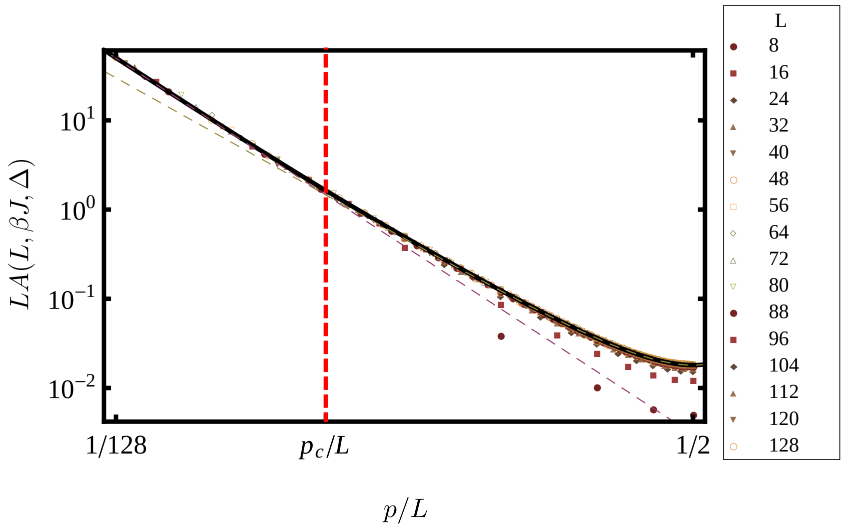

Figure 13: The thermal and disorder averaged squared of the fourier modes $A(p, L, \beta J, \Delta)$ as a function of p/L for $\beta = 2$ and $\Delta = 0.15$ for different lattice sizes $8 \leq L \leq 128$ on a log-log scale. The best unweighed fit from (6) is also plotted as a black line for reference, as are the power-law behaviours for the low-mode and high-mode regime as dashed lines. For the best fit the fourier modes from the lowest lattice sizes with $L \leq 40$ were not used since the higher-mode portion did not collapse onto the power-law behaviour induced by the random bonds. In this plot the value of the crossover point happens at $p_c/L \approx 0.037$ and is show as a dashed red line. From the crossover point we can conclude that $L_c \approx 13.4$.

random bond induced disorder dominates a larger part of the high modes. Also, we expect the prefactor $A(L, \beta J, \Delta)$ to increase for larger Δ. This means that a natural data collapse and the fit (6) are not possible for the fourier modes, as is possible with the fourier modes when varying the lattice size L.

For the simulations we fixed the size $L = 64$ and the temperature $\beta = 2.0$ and we let the disorder Δ vary such that $0.00 \leq \Delta \leq 0.30$ with step sizes of 0.02. For each value of Δ we fitted (6) to the fourier modes to get the values of the prefactor $A(L, \beta J, \Delta)$ and $c(L, \beta J, \Delta)$. The fourier modes and the best fit for each value of Δ are shown in Figure 15.

From each set of fourier modes we can determine the crossover point and Larkin length. In Figure 17 the Larkin lengths for different values of Δ are plotted. We fitted the Larkin lengths to

$$L_c(\delta) = x + \frac{y}{\delta^z}\,, \tag{7}$$

where δ is a parameter, in this case Δ.

In Figure 17 we have that, as expected, for higher values of Δ the crossover point

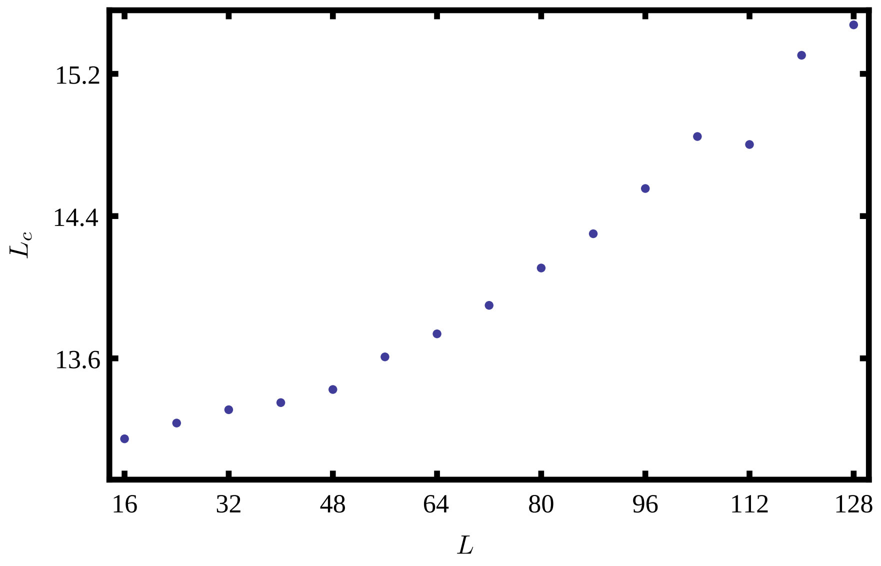

Figure 14: The Larkin length L_c calculated by using the fit (6) to the fourier modes of all data sets with the lattice size L larger than a certain value. Dropping more of fourier modes for the lower lattice sizes results in a higher Larkin length L_c. There is no clear limit to the value of the Larkin length so we are unable to derive the Larkin length for infinite system sizes.

increases and thus the Larkin length decreases. From the fit we derive the value $x \approx 3.55$ which means that in the limit $\Delta \to \infty$ the Larkin length converges to the value $L_c \approx 3.55$. Note that we restricted ourselves to not use anti-ferromagnetic bonds ($\Delta \leq 1$) but the same value holds for values of $\Delta > 1$. Also, using an infinite Δ results in a rigid domain wall hardly subject to thermal fluctuations since each breaking of a satisfied bond has zero probability of being chosen in the loop.

We plotted the prefactor $A(L, \beta J, \Delta)$ for different values of Δ in Figure 17. We fitted the different prefactors to

$$A(L, \beta J, \Delta) = x + y\Delta^z \,, \tag{8}$$

and we also plotted the fit in Figure 17. From the fit we find that $z \approx 1.66$ which suggests that the prefactors increase polynomial as disorder increases.

4.5.3 Varying β

As the temperature decreases one expects an increasing of the crossover point and a decreasing of the Larkin length. We expected this result since for lower temperatures there are less fluctuations in the domain wall due to thermal disorder, resulting in more random bond induced disorder. We also expect there to be a limiting value for

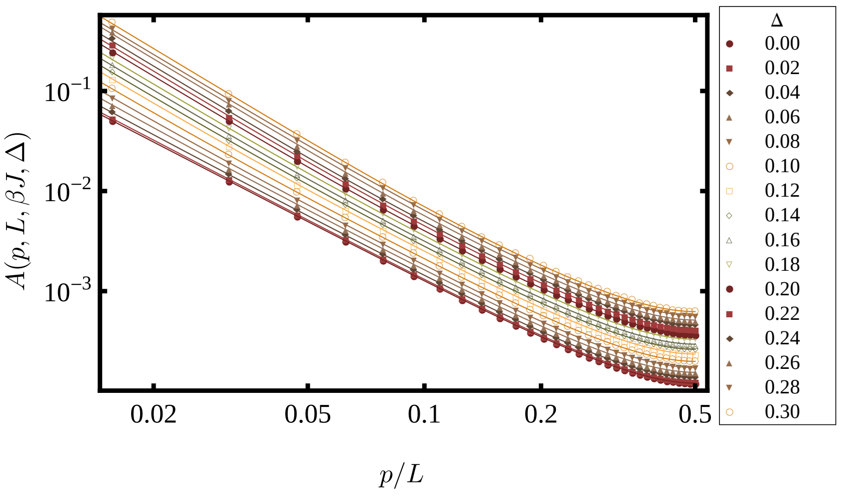

Figure 15: The thermal and disorder averaged squared of the fourier modes $A(p, L, \beta J, \Delta)$ as a function of p/L for $L = 64$ and $\beta = 2.0$ for different values of Δ with $0.00 \leq \Delta \leq 0.30$ on a log-log scale. The best fit (6) for each value of Δ is also plotted.

the crossover point and the Larkin length in the case of zero temperature. This due to the fact that lowering the temperature in an already cold system hardly affects the dynamics of the system.

For the simulations we fixed $L = 64$ and $\Delta = 0.15$ and we let $0.5 \leq \beta \leq 3.0$ vary with a step size of 0.1. For each different value of β we fit the fourier modes to (6) to get the prefactors $A(L, \beta J, \Delta)$ and $c(L, \beta J, \Delta)$. These fourier modes for different values of β and their respective fits (6) are shown in Figure 18.

For the different values of β with β sufficiently large we derived the Larkin length. This derivation is not possible for high temperatures since the fourier modes cannot be fitted to (6), as can be observed from Figure 18. We fitted the different Larkin lengths to (7) and we plotted the different Larkin lengths and the fit in Figure 19. From Figure 19 we can observe that the Larkin length decreases when the temperature decreases as expected. From the fit (7) we can derive the Larkin length in the limit $\beta \to \infty$ to be $L_c \approx 8.85$.

From the fit (6) we derive the prefactors $A(L, \beta J, \Delta)$. We can compare these prefactors with the prefactors $A(L, \beta J)$ in the absence of random bond disorder. In Figure 20 we have plotted the prefactors $A(L, \beta J, \Delta)$ with disorder and $A(L, \beta J)$ without disorder. The figure also shows the theory (4). At high temperatures the thermal fluctuations in the domain wall dominate the disorder and thus no difference in the prefactors. When the temperature is lower the random bond induced disorder begins to affects the prefactor resulting in different values for the prefactors with disorder relative to the prefactors without disorder.

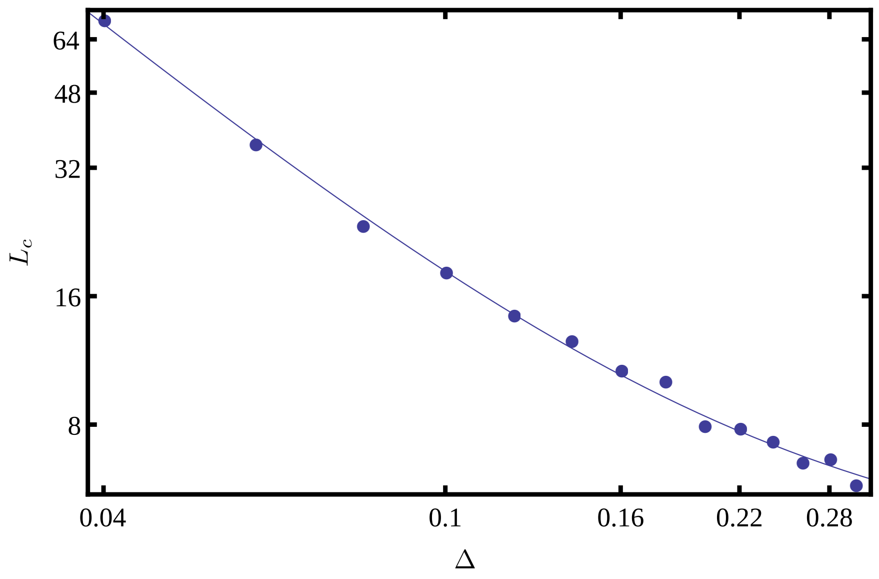

Figure 16: The Larkin length for different Δ's with $\beta = 2.0$ and $L = 64$ on a log-log scale. The fit (7) is also plotted as aline. As expected, the Larkin length decreases for larger values of Δ. From the fit (7) we can derive that in the limit $\Delta \to \infty$ we have that $L_c \approx 3.55$.

5 Future work

The weighed-loop algorithm as presented in this thesis is an algorithm for the simulation of domain walls in random bond Ising models. There are possible improvements on the weighed-loop algorithm for special use cases.

A first possible improvement is in the creation of the loops. A next bond is added to the loop with a weight equal to the Boltzmann distribution and only dependent on the bonds the algorithm can choose from. From section 3 it is clear that this selection probability results in a low acceptance probability of loops on large lattice sizes and high β. Other weight functions result in other selection probabilities and these weight functions can increase the fraction of accepted loops in these regimes while not decreasing the acceptance probability of the loops in other regimes. Further investigation can lead to a better weight function.

As already discussed, the weighed-loop algorithm also presents the possibility to incorporate a preferred direction. When for example the loop diverges from the domain wall, a preferred direction can be added when chaining the bonds to increase the likelihood of the loop returning to the domain wall. In this way the loop will return to the domain wall more often and thus there are less updates away from the domain wall which do not influence the domain wall and there are more updates in the neighbourhood of the wall decreasing the correlation in the modes

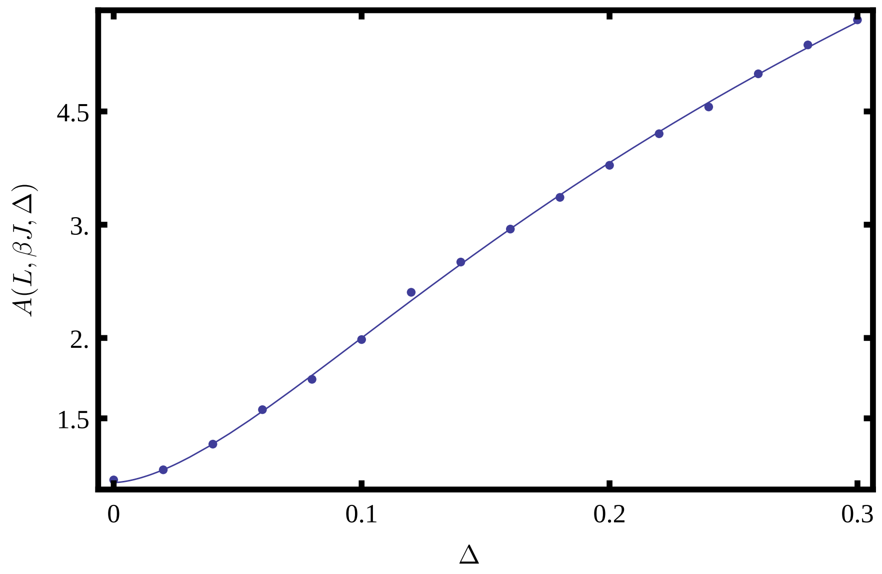

Figure 17: The prefactors $A(L, \beta J, \Delta)$ as a function of Δ with $L = 64$ and $\beta = 2.0$. The fit (8) is also plotted as reference showing that the prefactors increases polynomial as the disorder increases.

of the domain wall between consecutive measurements. However, such a preferred direction increases the complexity of the weighed-loop algorithm.

In each step of the weighed-loop algorithm only a single loop is created. The algorithm can be extended to construct multiple loops simultaneously. This increases the running time of the algorithm in each step, but since there are on average more updates each step, it lead to less correlation between consecutive measurements. However, there is also the possibility that constructing more loops leads to an allocation of more computational time to areas of the system which are not close to the domain wall and which are not relevant to update.

Finally, the parameter space for the properties of the domain wall in RBIM's consists of the three parameters L, β and Δ. We only investigated along straight lines in the parameter space parallel to the axes of the space. This does not cover the entire parameters space and there can exists possible regions in the parameter space for which unexpected behaviour of the domain wall occurrs. A more diverse study using varying more than one of the parameters at the same time leads to a more accurate understanding of the behaviour of the crossover point in the entire space.

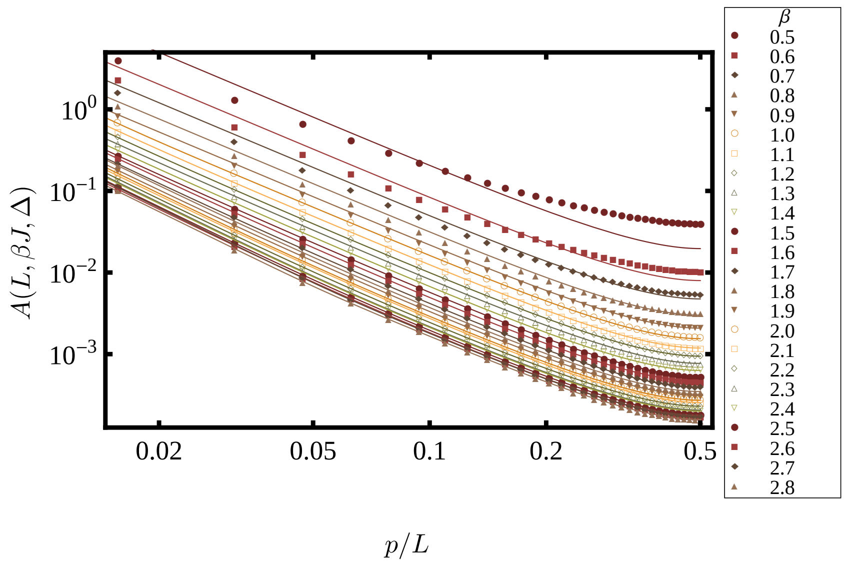

Figure 18: The thermal and disorder averaged squared of the fourier modes $A(p, L, \beta J, \Delta)$ as a function of p/L for $L = 64$ and $\Delta = 0.15$ for different temperatures with $0.5 \leq \beta \leq 2.8$ on a log-log scale. The best fit (6) for each value of β is also plotted. For low values of β the fourier modes cannot be fitted to (6) since the model of the domain wall dies not hold due to overhangs and local pockets of spins.

6 Conclusion

Introducing more efficient methods for the simulation of domain walls in random bond Ising models can improve the knowledge of real world magnetic domain walls in magnetic materials. This in turn will lead to a better understanding of magnetic memory devices. The goal of this thesis was to introduce a new weighed-loop algorithm for more efficient simulations of domain walls in random bond Ising models and using this new weighed-loop algorithm to simulate properties of the typical length scale of the domain wall at which a crossover occurs for different types of induced disorder.

In section 3 we proposed a new weighed-loop algorithm. This weighed-loop algorithm selects clusters of spins by chaining bonds of the Ising model to form closed loops in the spin lattice. The different options on each lattice point on the graph G induced by the Ising model are biased such that is more favourable to choose unsatisfied bonds over satisfied bonds, which in turn increases the acceptance probability of the loop and thus decreases the autocorrelation of the algorithm, especially at low temperatures or with large disorder in the bond strength of the system. Extra care should be taken when the constructed chains loop around the edges of the lattice since in these cases it is possible for the loop to enclose the

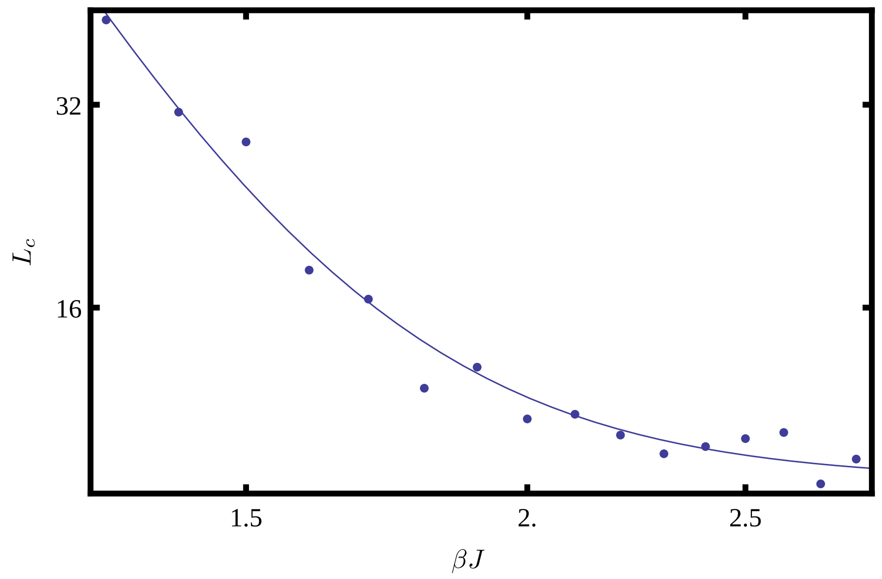

Figure 19: The Larkin length L_c for different values of $\beta > 1.2$ with $L = 64$ and $\Delta = 0.15$ on a log-log scale. For high temperatures the model of the domain wall does not hold. This implies that the Larkin length deduced from the fit (6), if possible, is not an accurate value of the Larkin length. The Larkin lengths for $\beta > 1.2$ are used to fit (7) from which the Larkin length $L_c \approx 8.85$ in the infinte β limit can be derived.

entire lattice. The algorithm can be implemented for any graph-type model in which the energy of the system is defined by the bond strength between spins. We have proven that the weighed-loop algorithm satisfies the detailed balance equation and we have proven that the algorithm is ergodic. With both these conditions it is proven that the algorithm simulates the Ising model correctly. The new algorithm as presented in section 3 does have clear advantages for simulating domain walls, but the algorithm also has disadvantages.

The weighed-loop algorithm can use multiple loops prevent wrapped loops from flipping all the spins in the lattice. However, the construction of more loops needs more computational time. We compared the autocorrelation of the weighed-loop algorithm using these wrapped loops and thus constructing more loops against the autocorrelation of the algorithm which discards the wrapped loops. We observed that it is more favourable for the autocorrelation to use the wrapped loops in contrast to discarding the wrapped loops.

Each constructed loop in the weighed-loop algorithm consists of a closed loop and a tail section. The tail section does not contribute to a change in the dynamics of the system and can be seen as wasted computational time. The tail section can be used as the start of the next loop in the next step of the algorithm. This decreases

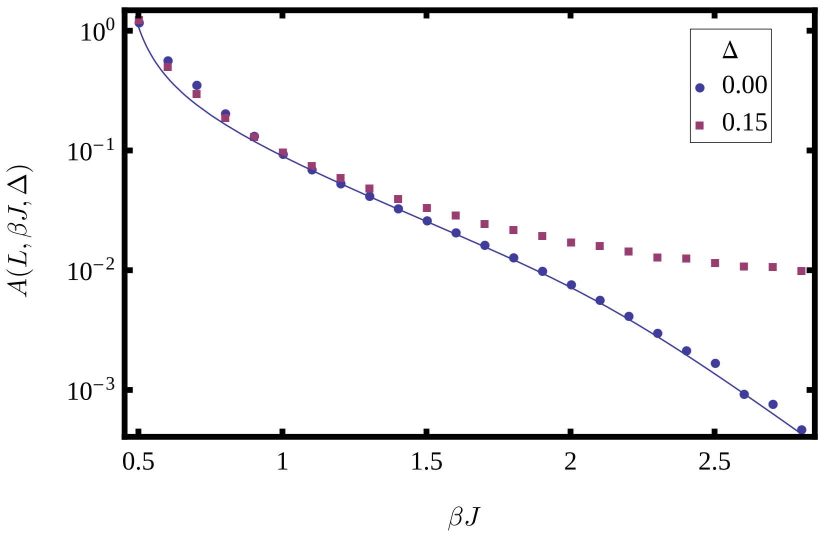

Figure 20: The prefactors $A(L, \beta J, \Delta)$ as a function of β with $L = 64$ for $\Delta = 0.00$ (blue disks) and $\Delta = 0.15$ (purple squares). The theory for the behaviour of the domain wall in the absence of disorder (4) is plotted as a blue line. From the plots it is clear that a minimal disorder-induced roughness is present for both with and without disorder.

the wasted computational time but it increases the correlation between consecutive steps in the algorithm. We compared the autocorrelation of the weighed-loop algorithm which reuses the tails for the next loops against the algorithm which discards the tails. We observed that it is more favourable for the autocorrelation to discard the tail sections of each loop.

Since the new weighed-loop algorithm is designed for simulating domain walls in the random bond Ising model we are interested in the regimes of the parameter space consisting of the parameters L, β and Δ for which the weighed-loop algorithm simulates more efficient than current known algorithms. Niedermayer's algorithm is one of such algorithms for simulations on glassy spin systems and random bond Ising models and in section 4.4 we compared the weighed-loop algorithm against Niedermayer's algorithm. As a measure of effectiveness we measured the autocorrelation in the first mode of the fourier transform of the domain wall for both algorithms since the first mode is the least dependent on local changes and more dependent on global changes and takes the longest time to reach its equilibrium. This implies that the first mode takes the longest time to decorrelate. A smaller autocorrelation implies more updates on the domain wall, or larger updates on the domain wall. For both Niedermayer's algorithm and the weighed-loop algorithm we measured the autocorrelation for different regimes in the L,β,Δ-space. From the simulations it followed that the weighed-loop algorithm is the more efficient algorithm in the regime at which conventional algorithms on the Ising model tend to get stuck, the regime with low temperatures and high disorder. These are the regimes in the parameters space for which many issues arise due to a rough energy landscape and exponentially high energy barriers.

From theory, properties of the behaviour of the domain walls are known and we verified these properties experimentally. For Ising models without disorder we have (4). We used the weighed-loop algorithm to simulate the domain walls in Ising models without disorder. For different lattice sizes the simulated results agree with the theory. For high and low temperatures the simulated values are less accurate since for high temperatures the domain wall contains a significant amount of overhangs and the system contains more local pockets of spins. For high temperatures a relative larger number of simulations is needed since for large temperatures less loops are accepted.

When disorder is introduced in the system the system shows a minimal disorder-induced roughness, even when the temperature goes to zero. We found that when increasing the system size, both disorder and temperature increases the roughness of the domain wall. It is known from theory that for Ising models with disorder the fourier modes can be divided into two regimes separated by the crossover point p_c/L. Low modes below p_c/L behave like the power law $p^{-(1+2\zeta)}$ and high modes above p_c/L behave like $\csc^2\left(\frac{p\pi}{2}\right)$. Without disorder the domain wall is only subject to thermal induced disorder and $\zeta_{\mathsf{T}} = 1/2$ in contrast with $\zeta_{\mathsf{RB}} = 2/3$ when the domain wall is also subject to random bond induced disorder. When simulating different regimes in the parameter space we confirmed that the modes follow the power laws in both regimes. Also, decreasing the temperature or increasing the disorder increases the crossover point and thus decreases the typical length scale L_c. Increasing β or Δ to infinite, one expects the crossover point to reach a limiting value, since for low temperatures decreasing the temperature hardly affects the dynamics of the system. The same holds for increasing the disorder when the system is already subject to a large disorder. We found the limiting values $L_c \approx 8.85$ in the limit $\beta \to \infty$ limit and $L_c \approx 3.55$ in the limit $\Delta \to \infty$. The methods we used to determine the limiting values of the Larkin length have no theoretical foundation so we cannot accurately known whether these limiting values are the true physical limiting values. Longer and more simulations may provide a clearer convergence for the Larkin length L_c.

To conclude, in this thesis we introduced a new weighed-loop algorithm and we have proven that the algorithm simulates the Ising model correctly. We also showed that there exists a regime in the L,β,Δ-space for which the algorithm simulates the domain wall more efficient than Niedermayer's algorithm. We used the weighed-loop algorithm for the simulation of the domain wall in the random bond Ising model and we showed that the simulated results agree with the theoretical results.

References

[1] K. Binder, "Ising model," *Hazewinkel, Michiel, Encyclopedia of Mathematics. Springer*, 2001.

[2] E. Ising, "Beitrag zur theorie des ferromagnetismus," *Zeitschrift für Physik*, vol. 31, no. 1, pp. 253–258, 1925.

[3] L. Onsager, "Crystal statistics. i. a two-dimensional model with an order-disorder transition," *Phys. Rev.*, vol. 65, pp. 117–149, Feb 1944.

[4] B. A. Cipra, "The ising model is np-complete," *SIAM News*, vol. 33, no. 6, pp. 1–3, 2000.

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

[6] W. K. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.

[7] R. H. Swendsen and J.-S. Wang, "Nonuniversal critical dynamics in monte carlo simulations," *Phys. Rev. Lett.*, vol. 58, pp. 86–88, Jan 1987.

[8] U. Wolff, "Collective monte carlo updating for spin systems," *Phys. Rev. Lett.*, vol. 62, pp. 361–364, Jan 1989.

[9] F. Niedermayer, "General cluster updating method for monte carlo simulations," *Phys. Rev. Lett.*, vol. 61, pp. 2026–2029, Oct 1988.

[10] Z. Zhu, A. J. Ochoa, and H. G. Katzgraber, "Efficient cluster algorithm for spin glasses in any space dimension," *Phys. Rev. Lett.*, vol. 115, p. 077201, Aug 2015.

[11] J. Houdayer, "A cluster monte carlo algorithm for 2-dimensional spin glasses," *The European Physical Journal B - Condensed Matter and Complex Systems*, vol. 22, no. 4, pp. 479–484, 2001.

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

[13] R. H. Swendsen and J.-S. Wang, "Replica monte carlo simulation of spin-glasses," *Phys. Rev. Lett.*, vol. 57, pp. 2607–2609, Nov 1986.

[14] G. Torlai and R. G. Melko, "Learning thermodynamics with boltzmann machines," *Phys. Rev. B*, vol. 94, p. 165134, Oct 2016.

[15] J. W. Gibbs, *Elementary principles in statistical mechanics.* Courier Corporation, 2014.

[16] A. P. Young, *Spin glasses and random fields*, vol. 12. World Scientific, 1998.

[17] R. H. Dong, B. Zheng, and N. J. Zhou, "Creep motion of a domain wall in the two-dimensional random-field ising model with a driving field," *Europhys. Lett.)*, vol. 98, no. 3, p. 36002, 2012.

[18] D. A. Huse and C. L. Henley, "Pinning and roughening of domain walls in ising systems due to random impurities," *Phys. Rev. Lett.*, vol. 54, pp. 2708–2711, Jun 1985.

[19] G. Blatter, M. V. Feigel'man, V. B. Geshkenbein, A. I. Larkin, and V. M. Vinokur, "Vortices in high-temperature superconductors," *Rev. Mod. Phys.*, vol. 66, pp. 1125–1388, Oct 1994.

[20] S. Lemerle, J. Ferré, C. Chappert, V. Mathet, T. Giamarchi, and P. Le Doussal, "Domain wall creep in an ising ultrathin magnetic film," *Phys. Rev. Lett.*, vol. 80, pp. 849–852, Jan 1998.

[21] L. S. E. Alvarez, K.-Y. Wang, and C. Marrows, "Field-driven creep motion of a composite domain wall in a pt/co/pt/co/pt multilayer wire," *Journal of Magnetism and Magnetic Materials*, vol. 322, no. 17, pp. 2529 – 2532, 2010.

[22] G. Rodríguez-Rodríguez, J. L. Menéndez, A. Hierro-Rodriguez, A. Pérez-Junquera, N. Montenegro, D. Ravelosona, J. M. Alameda, and M. Vélez, "Interplay between collective pinning and artificial defects on domain wall propagation in co/pt multilayers," *Journal of Physics D: Applied Physics*, vol. 43, no. 30, p. 305002, 2010.

[23] F. Cayssol, D. Ravelosona, C. Chappert, J. Ferré, and J. P. Jamet, "Domain wall creep in magnetic wires," *Phys. Rev. Lett.*, vol. 92, p. 107202, Mar 2004.

[24] M. V. Feigel'man, V. B. Geshkenbein, A. I. Larkin, and V. M. Vinokur, "Theory of collective flux creep," *Phys. Rev. Lett.*, vol. 63, pp. 2303–2306, Nov 1989.

[25] M. P. A. Fisher, D. S. Fisher, and J. D. Weeks, "Agreement of capillary-wave theory with exact results for the interface profile of the two-dimensional ising model," *Phys. Rev. Lett.*, vol. 48, pp. 368–368, Feb 1982.

[26] R. Keesman, "Domain-wall structure," *unpublished*.

[27] M. Newman and G. Barkema, *Monte Carlo Methods in Statistical Physics*, pp. 107–110. Oxford University Press: New York, USA, 1999.

[28] D. W. Heermann and K. Binder, *Monte Carlo Simulation in Statistical Physics*. Springer-Verlag Berlin Heidelberg, 2010.