The Loop Algorithm *

H.G. Evertz
Theor. Physik, Univ. Würzburg, 97074 Würzburg, Germany
evertz@physik.uni-wuerzburg.de

A review of the Loop Algorithm, its generalizations and its applications is given, including some new results. The loop algorithm is a Monte Carlo procedure which performs *nonlocal* changes of worldline configurations, determined by *local stochastic decisions*. It is based on a formulation of physical models in an extended ensemble of worldlines and graphs, and is related to Swendsen-Wang cluster algorithms. It overcomes many of the difficulties of traditional worldline simulations. Autocorrelations between successive Monte Carlo configurations are reduced by orders of magnitude. The grand-canonical ensemble (e.g. varying winding numbers) is naturally simulated. The continuous time limit can be taken directly. Improved Estimators exist which further reduce the errors of measured quantities. The algorithm remains unchanged in any dimension and for varying bond-strengths. It becomes less efficient in the presence of strong site disorder or strong magnetic fields. It applies directly to locally XYZ-like spin, fermion, and hard-core boson models. It has been extended to the Hubbard and the tJ model and generalized to higher spin representations. There have already been several large scale applications, especially for Heisenberg-like models, including a high statistics continuous time calculation of quantum critical exponents on a regularly depleted two-dimensional lattice of up to 20000 spatial sites at temperatures down to $T = 0.01J$.

1. INTRODUCTION AND SUMMARY

A pedagogical review of the Loop Algorithm, its generalizations, and its applications is given, including some new results. The loop algorithm [1–4] is a Monte Carlo procedure. It is applicable to models in worldline formulation [5]. It overcomes many of the difficulties of traditional worldline simulations. Its main feature is that as Monte-Carlo updates it performs *nonlocal* changes of worldline configurations. These nonlocal changes are determined by *local* stochastic decisions. The loop algorithm is based on a formulation of the worldline system in an extended ensemble which consists of both the original variables (spins or occupation numbers) and of graphs (sets of loops) [1–3,6]. It is related to Swendsen-Wang [7] cluster algorithms in classical statistical systems. Before we delve into the technical details, let us summarize the main features.

(a) Autocorrelations between successive Monte Carlo configurations are drastically reduced, thereby reducing the number of Monte Carlo sweeps required for a given system, often by orders of magnitude.

(b) The grand-canonical ensemble (e.g. varying magnetization, occupation number, winding numbers) is naturally simulated.

(c) The continuous time limit can be taken [8], completely eliminating the Trotter-approximation.

(d) So called Improved Estimators can be defined, which can strongly reduce the errors of measured quantities.

(e) Bond disorder and depleted lattices can be trivially included. The algorithm remains completely unchanged in any dimension.

Each of the points (a)-(d) can save orders of magnitude in computational effort over the traditional local worldline method. In addition, the algorithm is easier to program than traditional worldline updates. The method has some limitations:

(a) Site disorder as well as some other asymmetries in the Hamiltonian will make the simulations less efficient. This includes large magnetic field (or chemical potential) and other non "particle-hole-symmetric" terms like e.g. quadratic interactions of softcore bosons.

(b) A technical complication, called "freezing", occurs in some models and may reintroduce autocorrelations into the simulations. (This has not been tested for realistic cases).

*To be published in *"Numerical Methods for Lattice Quantum Many-Body Problems"*, ed. D.J. Scalapino, Addison Wesley Longman, Frontiers in Physics.

(c) Long range interactions make the algorithm more complicated and less effective.

Some of the usual limitations of worldline methods also remain in the loop algorithm, most notably the fermionic sign problem, which remains the most serious limitation of the applicability of worldline simulations. In addition, for all simulation methods, frustrated (spin) systems can cause a sign problem through negative matrix elements. Note that in the loop algorithm, both kinds of sign problems can be alleviated by a suitable improved estimator [9,10].

The loop algorithm has already been used for many physical models. The original formulation [1–4] of the algorithm (in vertex language) applies directly to general spin $\frac{1}{2}$ quantum spin systems in any dimensions, e.g. the 2D Heisenberg model [11], where improved estimators for this algorithm were first used [11]. At the root of the loop algorithm is a mapping of the physical model to an extended phase space which includes loops in addition to the original worldlines. In ref. [6] it was shown that this mapping is a Fortuin-Kasteleyn-like representation. A related mapping was later used in an axiomatic study of spin models [12]. The general anisotropic XYZ-model (including for example the XY-model quantized along the x-axis) maps to an eight-vertex-like situation for the shaded plaquettes of the worldline lattice. For the corresponding loop algorithm [3], explicit update probabilities were given in ref. [13]. The method has been adapted and extended to fermionic systems like the Hubbard model [14] and recently to the tJ model [15,9], and, in an extensive generalization, to quantum spin systems with higher spin representation [6,16], also for the XYZ-case [13]. The extension to more than (1+1) dimensions is immediate [1,2]: the algorithm remains completely unchanged, only the geometry of the plaquette lattice changes. In ref. [8] it has been shown that the continuous time limit can be taken. There have already been a number of very successful large scale applications, especially for Heisenberg-models, namely coupled chains [17–19] in two and three spatial dimensions, and two-dimensional XY-like [20], isotropic [11,8,21] and depleted spin-$\frac{1}{2}$ systems [22–24] including a high statistics calculation of quantum critical exponents on regularly disordered lattices [24] of up to 20000 spatial sites at temperatures down to $T = 0.01$. A recent exciting development along a somewhat different line is the "Worm" algorithm in continuous time [25], which allows calculation of single particle Greens functions in almost any model, by performing local moves of sources.

Section 2 describes the algorithm, with a brief review of the worldline representation, an intuitive outline of the loop algorithm, and a detailed and partially new step by step formal derivation of the algorithm, followed by a brief summary. We compute explicitly the update probabilities for the XXZ-model, and give a concise recipe for the Heisenberg antiferromagnet. Ergodicity is treated, and it is shown that in some important cases a mapping to a pure loop model can be done. In section 2.12 we introduce improved estimators, and in section 2.13 we describe the performance of the loop algorithm, its possibilities and limitations. Section 3 describes a number of generalizations, many of them immediate, and section 4 discusses some recent applications. Section 5 contains concluding remarks. The appendix covers some Monte Carlo formalism, describes the serious and sometimes overlooked problem posed by autocorrelations and critical slowing down, and finally provides recipes for proper handling of autocorrelations and error calculations.

2. ALGORITHM

The loop algorithm acts in the worldline representation, which was described in the previous chapter [5]. We will develop the formal procedure for the general anisotropic (XYZ-like) case. As an example we shall use the particularly simple but important case of the one-dimensional quantum XXZ model [5]. It includes the *Heisenberg model* and *hard core bosons* as special cases. We will see that the same calculation is valid for the loop algorithm in any spatial dimension and alreay covers most of the important applications.

2.1. Setup: Worldline representation and equivalent Vertex Model

Let us first recall the worldline representation for the example of the XXZ model on a one-dimensional chain of N sites [5]. The Hamiltonian is

$$\begin{aligned} \hat{H} \;= \sum_{\langle \mathbf{ij} \rangle} \hat{H}_{i,j} \;&= \sum_{\langle \mathbf{ij} \rangle} J_x (\hat{S}^x_{\mathbf{i}} \hat{S}^x_{\mathbf{j}} + \hat{S}^y_{\mathbf{i}} \hat{S}^y_{\mathbf{j}}) + J_z \hat{S}^z_{\mathbf{i}} \hat{S}^z_{\mathbf{j}} - B \sum_{\mathbf{i}} \hat{S}^z_{\mathbf{i}} \\ &= \sum_{\langle \mathbf{ij} \rangle} \tfrac{J_x}{2} (\hat{S}^+_{\mathbf{i}} \hat{S}^-_{\mathbf{j}} + \hat{S}^-_{\mathbf{i}} \hat{S}^+_{\mathbf{j}}) + J_z \hat{S}^z_{\mathbf{i}} \hat{S}^z_{\mathbf{j}} - B \sum_{\mathbf{i}} \hat{S}^z_{\mathbf{i}} \;, \end{aligned} \tag{1}$$

where $\vec{\hat{S}}_i$ are quantum spin $\frac{1}{2}$ operators at each site i, and $\hat{S}^+_i$, $\hat{S}^-_i$ are the associated raising and lowering operators. We use periodic boundary conditions.

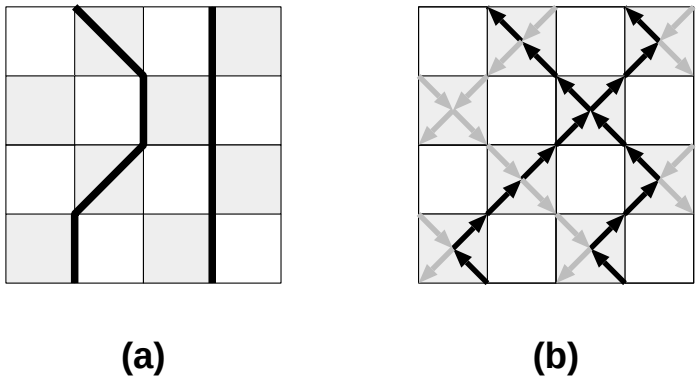

FIG. 1. Example of a worldline configuration on a checkerboard lattice of shaded plaquettes. (a) Worldline picture. (b) The *same* configuration as a vertex picture. Space direction (index i) is horizontal, imaginary time direction (index l) vertical. The variables $S^z_{i,l}$ are defined on each lattice site. Worldlines (arrows upwards in time) denote $S^z_{il} = +1$, empty sites (arrows downwards in time) denote $S^z_{il} = -1$. The Hamiltonian $\hat{H}_{i,i+1}$ acts on the shaded plaquettes.

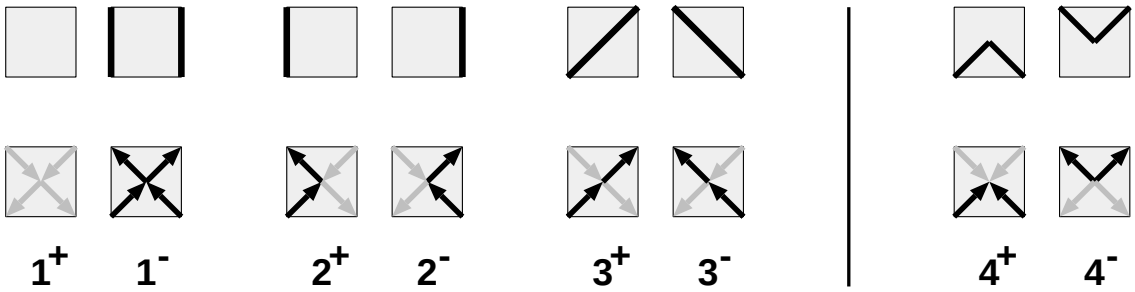

FIG. 2. Allowed plaquette configurations S_p. Top line: worldline picture. Bottom line: the same plaquettes in vertex picture. In the XXZ case (= six-vertex case), only the plaquettes with continuous worldlines, $S_p = 1^{\pm}, 2^{\pm}, 3^{\pm}$, have nonzero weight, eq. (6). In the anisotropic XYZ case (= eight-vertex case) the plaquettes $4^{\pm}$ also have nonzero weight.

After splitting the Hamiltonian into commuting pieces

$$\begin{aligned} \hat{H} &= \hat{H}_{even} + \hat{H}_{odd} \\ \hat{H}_{even,odd} &= \textstyle\sum_{i:\,\text{even,odd}} \hat{H}_{i,i+1}\,, \end{aligned} \tag{2}$$

performing a Trotter-Suzuki breakup [26]

$$Z^{XXZ} = tr\, e^{-\beta \hat{H}} = \lim_{L_t\to\infty} Z^{XXZ}_{tr} = \lim_{L_t\to\infty} tr\left(e^{-\frac{\beta}{L_t}\hat{H}_{even}}\, e^{-\frac{\beta}{L_t}\hat{H}_{odd}}\right)^{L_t}\,, \tag{3}$$

and inserting complete sets of $\hat{S}^z$ eigenstates, we arrive at the worldline representation

$$Z^{XXZ}_{tr} = \sum_{S^z_{il}} W(\{S^z_{il}\}) = \sum_{S^z_{il}} \prod_p W_p(\{S_p\})\,, \tag{4}$$

where the summation $\sum_{\{S^z_{il}\}}$ extends over all "configurations" $\mathcal{S} = \{S^z_{il}\}$ of "spins" $S^z_{il} = \pm 1$, which live on the sites $(i,l),\ i = 1..N,\ l = 1..2L_t$, of a (1+1)-dimensional checkerboard lattice. The index $l = 1, .., 2L_t$ corresponds to imaginary time. The product $\prod_p$ extends over all shaded plaquettes of that lattice (see fig. 1), and S_p stands for the 4-tupel of spins at the corners of a plaquette $p = (\,(i,l),(i+1,l),(i,l+1),(i+1,l+1)\,)$.
The weight W_p at each plaquette

$$W_p(S_p) = \langle S^z_{i,l} S^z_{i+1,l} | e^{-(\Delta\tau)\hat{H}_{i,i+1}} | S^z_{i,l+1} S^z_{i+1,l+1} \rangle\,, \tag{5}$$

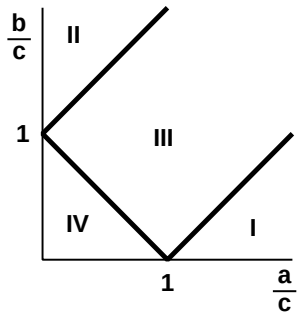

FIG. 3. Exact phase diagram of the classical six-vertex model [27] at $B = 0$. The weights a, b, c are defined in eq. (6). Phase III is massless (infinite correlation length). At $\frac{a}{c} + \frac{b}{c} = 1$ there is a Kosterlitz-Thouless phase transition [29] to the massive (finite correlation length) phase IV, and at $\frac{a}{c} - \frac{b}{c} = 1$ there is a first order KDP phase transition [30] to the ferroelectric phase I. The weights corresponding to the XY-model (or free fermions), i.e. $J_z = 0$, are located on the circle $a^2 + b^2 = c^2$.

where $\Delta\tau \equiv \beta/L_t$, is given by the matrix elements [1]

$$\begin{aligned}
W(1^+) \equiv a_+ \equiv \langle ++|e^{-(\Delta\tau)\hat{H}_{i,i+1}}|++\rangle &= e^{-\frac{\Delta\tau}{4}J_z} \quad e^{+\frac{\Delta\tau}{2}B} \\
W(1^-) \equiv a_- \equiv \langle --|e^{-(\Delta\tau)\hat{H}_{i,i+1}}|--\rangle &= e^{-\frac{\Delta\tau}{4}J_z} \quad e^{-\frac{\Delta\tau}{2}B} \\
W(2^\pm) \equiv c \equiv \langle +-|e^{-(\Delta\tau)\hat{H}_{i,i+1}}|+-\rangle &= \langle -+|e^{-(\Delta\tau)\hat{H}_{i,i+1}}|-+\rangle = e^{+\frac{\Delta\tau}{4}J_z} \quad \cosh\left(\tfrac{\Delta\tau}{2}|J_x|\right) \\
W(3^\pm) \equiv b \equiv \langle +-|e^{-(\Delta\tau)\hat{H}_{i,i+1}}|-+\rangle &= \langle -+|e^{-(\Delta\tau)\hat{H}_{i,i+1}}|+-\rangle = e^{+\frac{\Delta\tau}{4}J_z} \quad \sinh\left(\tfrac{\Delta\tau}{2}|J_x|\right) .
\end{aligned} \tag{6}$$

The matrix element b is positive for ferromagnetic XY couplings $J_x < 0$. For antiferromagnetic XY couplings $J_x > 0$, it can be made positive on a bipartite lattice by rotating $\hat{S}^{x,y} \to -\hat{S}^{x,y}$ on one of the two sublattices. We have already assumed such a rotation in eq. (6).

Since $[\hat{H}_{i,i+1}, \hat{S}^z_{tot}] = 0$, there are only the *six* nonvanishing matrix elements given in eq. (6), namely those that conserve $\hat{S}^z_i + \hat{S}^z_{i+1}$, as shown pictorially in fig. 2. Therefore, the locations of $S^z_{il} = 1$ in fig. 1(a) can be connected by continuous worldlines. The worldlines close in imaginary time-direction because of the trace in eq. (3).

For models with fermions or hard core bosons one inserts occupation number eigenstates instead of S^z_{ij}. Nearest neighbor hopping then again leads to the six-vertex case [5] of fig. 2. The term "worldline" derives from this case, since here they connect sites occupied by particles.

We will find it useful to also visualize worldline configurations in a slightly different way, namely as configurations of a *vertex model* [27]. To do this, we perform a one-to-one mapping of each worldline configuration to a vertex configuration. We stay on the same lattice of shaded plaquettes. We represent each spin S^z_{il} by an arrow between the centers of the two shaded plaquettes to which the site (i, l) belongs. The arrow points upwards (downwards) in time for $S^z_{il} = +1(-1)$. The worldline-configuration in fig. 1(a) is thus mapped to the vertex configuration of fig. 1(b). The one-to-one mapping of the worldline-plaquettes is shown in fig. 2. The conservation of $\hat{S}^z_{tot}$ on each shaded plaquette means in vertex language that for each vertex (center of shaded plaquette) two arrows point towards the vertex and two arrows point away from it. If one regards the arrows as a vector field, then this means a

$$\text{condition} \quad \text{"}\mathbf{divergence = zero}\text{"} \quad \text{for the arrows.} \tag{7}$$

Note again that vertex language and worldline language refer to the same configurations; they differ only in the pictures drawn.

We have now mapped the XXZ quantum spin chain to the *six-vertex model* of statistical mechanics [27], though with unusual boundary conditions, since the vertex lattice here is tilted by 45 degrees with respect to that of the

[1] The notation a, b, c is standard for vertex models [27], the notation $1, 3, 2$ (in different order) is that used in refs. [28,6,13].

standard six-vertex model. Let us look more closely at the case of vanishing magnetic field, $B = 0$. This model has been exactly solved in (1+1) dimensions [27]. The exact phase diagram is shown in fig. 3, in terms of the plaquette weights given in eq. (6) and in fig. 2.

It is interesting to note where the couplings of the Trotter-discretized XXZ-model at $B = 0$ are located in this phase diagram (see eq. (6)): For the Heisenberg antiferromagnet $J_x = J_z > 0$ we have $a + b = c$, i.e. we are *on* the Kosterlitz-Thouless line. As $\Delta\tau \to 0$, we approach the point $a/c = 1$, $b/c = 0$. For the Heisenberg ferromagnet $J_x = J_z < 0$ we have $a - b = c$, i.e. we are on the KDP transition line, approaching the same point $a/c = 1$, $b/c = 0$ as $\Delta\tau \to 0$. When $|J_z| < |J_x|$, the same point is approached from inside the massless (XY-like) region. When $|J_z| > |J_x|$, it is approached from below the respective transition line, i.e. from the massive (Ising-like) phase IV when $J_x > 0$ (AF) and from phase I when $J_x < 0$ (FM). Note that the local couplings a, b, c do not change in higher dimensions (see section 3.2).

2.2. Anisotropic case: XYZ model and eight-vertex weights

For generality later on, let us briefly describe the anisotropic case without magnetic field, in which $J_x \neq J_y$ in the Hamiltonian

$$\hat{H} \ = \sum_{\langle \mathbf{ij} \rangle} J_x \hat{S}^x_{\mathbf{i}} \hat{S}^x_{\mathbf{j}} \ + \ J_y \hat{S}^y_{\mathbf{i}} \hat{S}^y_{\mathbf{j}} \ + \ J_z \hat{S}^z_{\mathbf{i}} \hat{S}^z_{\mathbf{j}} \ . \tag{8}$$

We also get this case if we quantize the XXZ-model along an axis different from the z-axis. The treatment is the same as for the XXZ-model. Again we use $\hat{S}^z$ eigenstates to insert complete sets, and arrive at the following nonvanishing matrix elements on the (1+1)-dimensional checkerboard lattice,

$$\begin{aligned}
W(1^{\pm}) \ \equiv \ a \ &:= \ \langle ++|e^{-(\Delta\tau)\hat{H}_{i,i+1}}|++\rangle \ = \ \langle --|e^{-(\Delta\tau)\hat{H}_{i,i+1}}|--\rangle \ = \ e^{-\frac{\Delta\tau}{4}J_z} \ \cosh\left(\tfrac{\Delta\tau}{4}|J_x - J_y|\right) , \\
W(2^{\pm}) \ \equiv \ c \ &:= \ \langle +-|e^{-(\Delta\tau)\hat{H}_{i,i+1}}|+-\rangle \ = \ \langle -+|e^{-(\Delta\tau)\hat{H}_{i,i+1}}|-+\rangle \ = \ e^{+\frac{\Delta\tau}{4}J_z} \ \cosh\left(\tfrac{\Delta\tau}{4}|J_x + J_y|\right) , \\
W(3^{\pm}) \ \equiv \ b \ &:= \ \langle +-|e^{-(\Delta\tau)\hat{H}_{i,i+1}}|-+\rangle \ = \ \langle -+|e^{-(\Delta\tau)\hat{H}_{i,i+1}}|+-\rangle \ = \ e^{+\frac{\Delta\tau}{4}J_z} \ \sinh\left(\tfrac{\Delta\tau}{4}|J_x + J_y|\right) , \\
W(4^{\pm}) \ \equiv \ d \ &:= \ \langle ++|e^{-(\Delta\tau)\hat{H}_{i,i+1}}|--\rangle \ = \ \langle --|e^{-(\Delta\tau)\hat{H}_{i,i+1}}|++\rangle \ = \ e^{-\frac{\Delta\tau}{4}J_z} \ \sinh\left(\tfrac{\Delta\tau}{4}|J_x - J_y|\right) ,
\end{aligned} \tag{9}$$

which reduce to eq. (6) when $J_x = J_y$. We see that now there is an additional type of vertex with weight d, shown as type $4^{\pm}$ in fig. 2, in which all four arrows point either towards or away from the center. This vertex type may be thought of as a source (resp. sink) of arrows. Eq. (7) becomes the

$$\text{condition} \ \ \text{“}\textbf{divergence = zero mod 4}\text{”} \ \ \text{for the arrows.} \tag{10}$$

The vertices and their weights now correspond to the eight-vertex model [27]. We will see that very little changes for the loop algorithm in this case [3].

2.3. Outline of Loop Algorithm

Notation: From now on we will synonymously use “plaquette” or “vertex” to refer to the shaded plaquettes of the checkerboard lattice. We also use interchangeably the terms “spin direction”, “arrow direction”, and “occupation number” to refer to the 2 possible states S^z_{il} at each site (il) of the checkerboard lattice. We denote both probabilites and plaquettes by the letter p. S_p and W_p are the spin configuration at plaquette p and its weight, and G_p will be a breakup at p. “six-vertex-case” and “eight-vertex-case” will refer to the local plaquette constraints (i.e. nonzero weights), *not* to the respective models of statistical mechanics themselves.

The traditional way to perform Monte Carlo updates on a worldline configuration has been discussed in the previous chapter [5]. It consists of proposing *local deformations* of worldlines and accepting/rejecting them with suitable probability. In contrast, the updates for the loop algorithm are very nonlocal. We will first describe the basic idea for the example of the XXZ case and outline the resulting procedure. We postpone the formal discussion and the calculation of Monte Carlo probabilities to the next section.

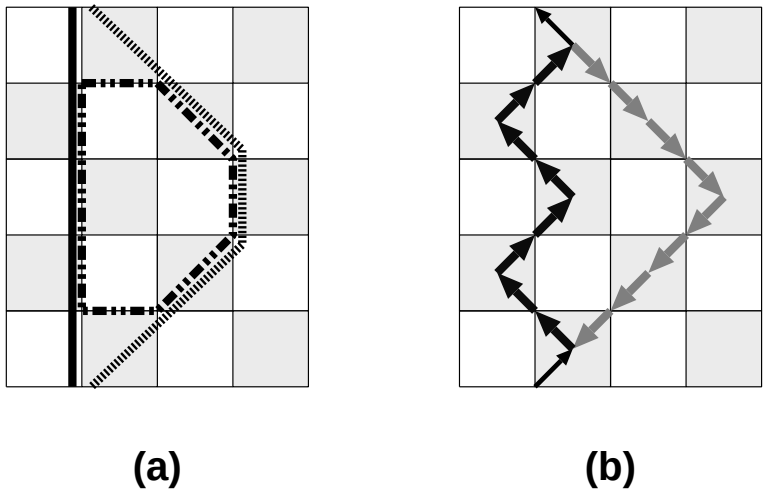

FIG. 4. Example of a loop update. (a) Worldline picture. The thick solid line denotes a single worldline, the dot-dashed line depicts a possible loop. By "flipping the loop", i.e. flipping the spin direction (arrow direction) on all sites along the loop, the worldline will be deformed into the dashed shape. Since the loop can potentially be very large, this deformation can be very nonlocal. (b) The same situation as a vertex picture. The loop is represented by the thick arrows. By construction, in the vertex picture each loop follows arrows of the spin-configuration. This means that the loop runs upwards in time-direction along sites with spin $S^z_{il} = +1$, i.e. along worldlines, and downwards in time-direction where there is no worldline.

Two observations lead to the loop algorithm:

(1) The Hamiltonian acts locally on individual plaquettes. Thus the detailed balance condition on Monte Carlo probabilities can be satisfied locally on those plaquettes.

(2) The allowed configurations of arrows have zero divergence, eq. (7). Therefore any two allowed configurations can be mapped into each other by changing the arrow-direction on any set of closed **loops**, where *along each of these loops, the arrows have to point in constant direction.* These are the loops that are constructed in our algorithm. The loops are also determined locally on each plaquette, by setting their direction there (see below).

An example is given in fig. 4. To emphasize the point: The loops are not worldlines; instead they consist of the locations of (proposed) *changes* in the worldline occupation number (=arrow directions = spin directions). Also, the loops are not prescribed, instead they will be determined stochastically, with probabilities that depend on the current worldline configuration. Since both the zero divergence condition and detailed balance can be satisfied locally at the plaquettes, we will be able to construct the loops by *local* stochastic decisions, yet achieve potentially very nonlocal worldline updates.

How does one specify a valid loop ? This is seen most easily in the vertex picture: the loop has to follow the arrows of the spin-configuration. At each vertex which the loop traverses, one has to specify the direction into which the loop continues. Since there can be at most one loop on each arrow, this specification also fixes the direction for a second loop entering the same vertex. I.e. what we specify is a "breakup" of the current arrow configuration at a vertex into two disconnected parts. The possible breakups of this kind are shown in fig. 5, and labeled G^{12},G^{23},G^{13} in the six-vertex case. Notice that for each of the 6 possible arrow configurations $i^{\pm}$, fig. 2, only 2 such breakups are compatible with the given arrow directions, namely those labelled $G^{ij}, (i \neq j)$.

There is another kind of "breakup" that maintains condition (7). Here all 4 spins on a plaquette are forced to be flipped together. We call this choice "freezing" (labeled G^{ii} in fig. 5), since for a flip-symmetric model like the six-vertex model at $B = 0$ it preserves the current weight $W(i^{\pm})$. If two different loops meet at a frozen vertex, they are glued together there, forming a cluster of more than one loop which must be flipped together. In general we shall denote by "**cluster**" a set of loops (often a single loop) that are mutually glued together.

We see that *by specifying a breakup for every shaded plaquette, the whole vertex configuration is subdivided into a set of clusters which consist of closed loops.* Each site of the checkerboard lattice is part of one such loop. We shall call such a division of the vertex configuration into loops a "graph" G. Flipping the direction of arrows (= spin or occupancy) on all sites of one or more clusters of a graph (a "cluster flip" which consists of "loop flips"), leads to a new allowed configuration. In the loop algorithm the loops are constructed by specifying breakups with suitable probabilities that depend on the current configuration (see below). In the vertex picture, the graph G resides on the

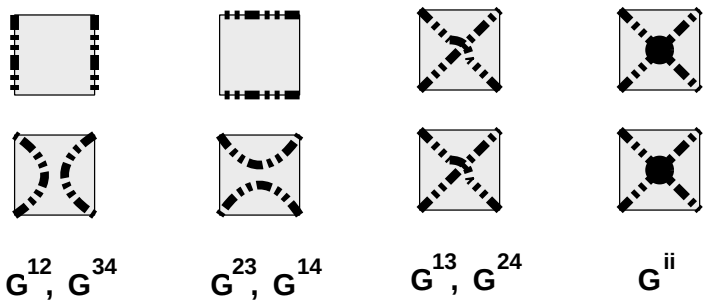

FIG. 5. Graphical representations of all possible breakups. Top row: worldline picture. Middle row: the same graphs in the vertex picture. Each graphical representation applies to two different breakups "G^{ij}", as denoted in the bottom row. Breakup "G^{ij}" is possible (i.e. compatible with the arrow directions) in plaquette configurations $S_p = i^{\pm}$ or $j^{\pm}$ (see fig. 2). Flip of the two spins on either one of the two lines in the graphical representation of G^{ij}, $i \neq j$, maps between configurations i and j. Breakup G^{ii}, called "freezing", forces all four spins to flip together, thus mapping between i^+ and i^-. The breakups G^{i4} occur only in the XYZ-like (eight-vertex-like) case. In the six-vertex-case, the three non-freezing breakups G^{ij}, $i \neq j$ are one-to-one equivalent to the three graphical representations.

same arrows as the spins. In the worldline picture, the elements of G look slightly differently, as seen in figures 4 and 5 . Note that by introducing loops, we have effectively extended the space of variables, from spins, to spins and breakups. This point will be formalized in the next section.

We see that the *basic procedure* for one Monte Carlo update consists of two *stochastic* mappings: First from spins to spins plus loops, and second from there to new spins. I.e., starting with the current configuration of worldlines:

(1) **Select a breakup** for each shaded plaquette with a probability that depends on the current spin configuration at that plaquette. These probabilities are discussed below. Identification of the clusters which are implicitely constructed by these breakups may then involve a search through the lattice.

(2) **Flip each cluster with suitable probability**, where "flipping a cluster" means to change the direction of all arrows along the loops in this cluster (or, equivalently, changing spin direction or occupation number, respectively). The combined cluster flips result in a new spin configuration. The flip probabilities depend in general on the Hamiltonian and on the current spin configuration. In the ideal case, for example the isotropic Heisenberg model in any dimension, each individual loop can be flipped independently with probability 1/2.

An example is given in fig. 6. Notice that in this example the flip of a loop which happened to wind around the lattice in spatial direction led to a change in spatial winding number of the worldline configuration, i.e. an update that *cannot* be done by local deformations of worldlines.

Little changes in the general XYZ-like (eight-vertex-like) case [3]. The loops now have to change direction [3] at every breakup of type $(i, 4)$. Alternatively, one can also omit assigning a direction to loops.

Let us now cast the general ideas into a valid procedure. Sections 2.4 to 2.11 are formal and comprehensive, with detailed explanations. A summary is given in section 2.8, and a recipe for the Heisenberg antiferromagnet in section 2.10. Previous formal expositions can be found in the original loop algorithm papers [2,3] (the best formal description there is that for the eight-vertex case in ref. [3]), as well as, in a more suitable language closer to the Fortuin-Kasteleyn mapping of statistical mechanics, in the papers by Kawashima and Gubernatis [6,13]. We shall use both the worldline picture and the vertex picture of ref. [2,3], in order to provide a bridge between the existing formulations and to make the simple geometry of the problem as obvious as possible.

In the following, we first introduce the procedure of Kandel and Domany for extending the space of variables. Suitably applying this procedure to worldline models, we arrive at the loop algorithm. We shall calculate explicit probabilities for the case of the XXZ model. The most important practical application of the loop algorithm at present is for variations of the Heisenberg model. This special case of the six-vertex model turns out to be particularly simple, and we give a short recipe. We also point out that for some models (like XXZ) it is possible to sum over all spin variables to obtain a pure loop model. Finally, we introduce improved estimators and describe the performance of the loop algorithm.

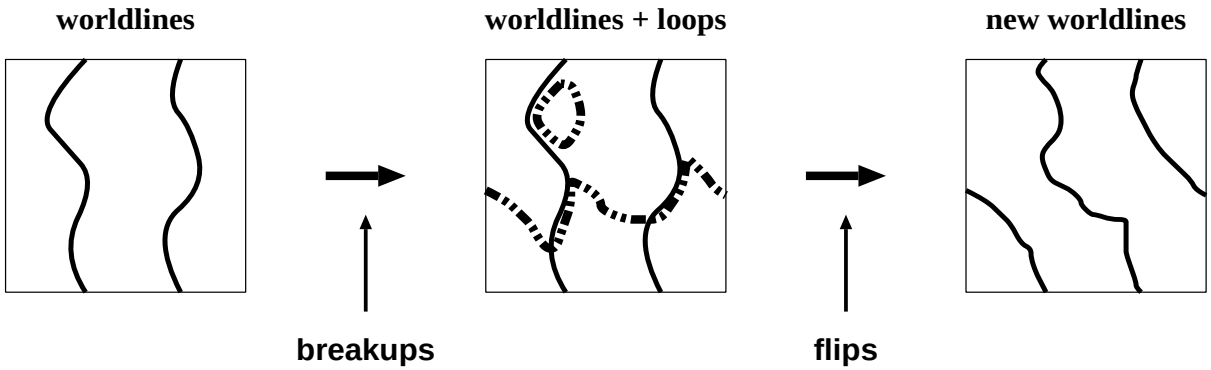

FIG. 6. Example of a worldline update with the loop algorithm. For clarity, we show a situation with only two worldlines. We start with the worldline configuration $\mathcal{S}$ in the left picture. The stochastic breakup decision on each plaquette, with probabilities depending on this worldline configuration, defines loops (in general clusters of loops), only two of which are drawn. In this example we then flip both loops, i.e. flip the spin direction (= worldline occupancy) along the loops. This results in the new worldline configuration $\mathcal{S}'$ in the right picture, which, as in this example, can be very different from the original one. Since one of the loops happened to wind around the lattice in spatial direction, its flip produced a worldline configuration with nonzero winding number. Note that for the next worldline update, the current loop configuration is discarded, and a completely new set of breakups will be determined with probabilities depending on the new worldline configuration.

2.4. Kandel-Domany framework

A brief overview of the basics of Monte Carlo algorithms is given in Appendix A. The derivation [2,3] of the loop algorithm is similar to that for the Swendsen Wang cluster algorithm in statistical mechanics [7] which uses the Fortuin-Kasteleyn mapping of the Ising model to an extended phase space. (For an excellent review see e.g. ref. [31]). A general formalism for such a mapping was given by Kandel and Domany [32]. Here we use the more suitable language similar of Kawashima and Gubernatis [6], who made the Fortuin-Kasteleyn-like nature of the mapping obvious.

For future reference, we first write down the general scheme, without yet making reference to individual spins, loops, or plaquettes. We start with a set $\{\mathcal{S}\}$ of configurations $\mathcal{S}$ and a set $\{G\}$ of graphs G, which together constitute the extended phase space. The partition function

$$Z = \sum_{\mathcal{S}} W(\mathcal{S}) \tag{11}$$

depends only on $\mathcal{S}$. In addition we now *choose* a new weight function $W(\mathcal{S}, G)$ which must satisfy

$$\begin{aligned} \textstyle\sum_G W(\mathcal{S}, G) &= W(\mathcal{S})\,, \\ W(\mathcal{S}, G) &\geq 0\,. \end{aligned} \tag{12}$$

Thus we have a Fortuin-Kasteleyn-like [33] representation:

$$Z = \sum_{\mathcal{S}} \sum_{G} W(\mathcal{S}, G) \tag{13}$$

A Monte Carlo update now consists of 2 steps:

i) Given a configuration $\mathcal{S}$ (which implies $W(\mathcal{S}) \neq 0$), choose a graph G with probability

$$p(\,\mathcal{S} \to (\mathcal{S}, G)\,) \;=\; \frac{W(\mathcal{S}, G)}{W(\mathcal{S})}\;. \tag{14}$$

ii) Given $\mathcal{S}$ and G (this implies $W(\mathcal{S}, G) \neq 0$), choose a new configuration $(\mathcal{S}', G')$ with a probability $p\,(\,(\mathcal{S}, G) \to (\mathcal{S}', G')\,)$ that satisfies detailed balance with respect to $W(\mathcal{S}, G)$:

$$W(\mathcal{S}, G) \times p\,(\,(\mathcal{S}, G) \to (\mathcal{S}', G')\,) \;=\; W(\mathcal{S}', G') \times p\,(\,(\mathcal{S}', G') \to (\mathcal{S}, G)\,)\;, \tag{15}$$

for example the heat bath probability

$$p\,(\,(\mathcal{S},G)\to(\mathcal{S}',G')\,)\;=\;\frac{W(\mathcal{S}',G')}{W(\mathcal{S},G)+W(\mathcal{S}',G')}\;. \tag{16}$$

Then the mapping $\mathcal{S}\to\mathcal{S}'$ also satisfies detailed balance with respect to the original weight $W(\mathcal{S})$. Proof:

$$\begin{aligned} W(\mathcal{S})\,p(\mathcal{S}\to\mathcal{S}') &= W(\mathcal{S}) \quad \textstyle\sum_{G,G'} p(\,\mathcal{S}\to(\mathcal{S},G)\,)\;\; p\,(\,(\mathcal{S},G)\to(\mathcal{S}',G')\,) \\ &= W(\mathcal{S}) \quad \textstyle\sum_{G,G'} \frac{W(\mathcal{S},G)}{W(\mathcal{S})}\;\; p\,(\,(\mathcal{S}',G')\to(\mathcal{S},G)\,)\;\; \frac{W(\mathcal{S}',G')}{W(\mathcal{S},G)} \\ &= W(\mathcal{S}') \quad \textstyle\sum_{G,G'} \frac{W(\mathcal{S}',G')}{W(\mathcal{S}')}\;\; p\,(\,(\mathcal{S}',G')\to(\mathcal{S},G)\,) \\ &= W(\mathcal{S}') \quad \textstyle\sum_{G,G'} p(\,\mathcal{S}'\to(\mathcal{S}',G')\,)\;\; p\,(\,(\mathcal{S}',G')\to(\mathcal{S},G)\,) \\ &= W(\mathcal{S}') \quad p(\mathcal{S}'\to\mathcal{S})\;. \end{aligned} \tag{17}$$

(Within a Monte Carlo the denominators in eq. (17) cannot vanish.)

2.5. Application to loop clusters

Now we apply this formalism to a model defined on plaquettes, with

$$\begin{aligned} W(\mathcal{S}) &= A_{global}(\mathcal{S})\;\times\; W^{plaq}(\mathcal{S})\;, \\ &= A_{global}(\mathcal{S})\;\times\; \textstyle\prod_p W_p(S_p)\;. \end{aligned} \tag{18}$$

To cover the general case, we have split off a global weight factor A_{global}. This split is not unique. Now we devise an algorithm for $W^{plaq}(\mathcal{S})\equiv\prod_p W_p(S_p)$. Because of its product structure, we will be able to do the decomposition into graphs separately on every plaquette. Thus in analogy with eq. (12) we look for a set of "breakups" G_p and new weights $W_p(S_p,G_p)$ on every plaquette p which satisfy

$$\begin{aligned} \textstyle\sum_{G_p} W_p(S_p,G_p) &= W_p(S_p)\,, \\ W_p(S_p,G_p) &\geq 0\,, \end{aligned} \tag{19}$$

which implies

$$W^{plaq}(\mathcal{S})=\prod_p\sum_{G_p} W_p(S_p,G_p)=\sum_{\bigcup_p G_p}\prod_p W_p(S_p,G_p)\equiv\sum_G W^{plaq}(\mathcal{S},G) \tag{20}$$

with $G\equiv\bigcup_p G_p$, and $W^{plaq}(\mathcal{S},G)\equiv\prod_p W_p(S_p,G_p)$. Thus

$$\begin{aligned} W(\mathcal{S},G) &= A_{global}(\mathcal{S})\times W^{plaq}(\mathcal{S},G) \\ &= A_{global}(\mathcal{S})\times\textstyle\prod_p W_p(S_p,G_p)\;. \end{aligned} \tag{21}$$

We can use this form of $W(\mathcal{S},G)$ in eq. (12), and apply the Kandel-Domany procedure.

Restricting ourselves to $G'=G$, the 2 steps i), ii) in the previous section now become the procedure for the loop algorithm. Starting with a configuration $\mathcal{S}$:

(1) Breakup: For each plaquette, choose G_p with probability

$$p(\,S_p\to(S_p,G_p)\,)\;=\;\frac{W_p(S_p,G_p)}{W_p(S_p)}\;. \tag{22}$$

This satisfies eq. (14).

(2) Flip: Choose a new configuration $(\mathcal{S}',G)$ with a probability $p\,(\,(\mathcal{S},G)\to(\mathcal{S}',G)\,)$ that satisfies detailed balance with respect to $W(\mathcal{S},G)$.

In the next section we shall explicitely find a suitable set of breakups G_p and weights $W_p(S_p,G_p)$.

2.6. Graph weights

Detailed balance with respect to $W^{plaq}(\mathcal{S},G) \equiv \prod_p W_p(S_p,G_p)$ is particularly simple if $W^{plaq}(\mathcal{S},G)$ does not change upon any of the possible updates, i.e.

$$W^{plaq}(\mathcal{S},G) = W^{plaq}(\mathcal{S}',G) \tag{23}$$

for all configurations $\mathcal{S}'$ for which $W^{plaq}(\mathcal{S}',G) \neq 0$. Then $W^{plaq}(\mathcal{S},G)$ cancels in eq. (16), which becomes

$$p_{flip}(\mathcal{S}\to\mathcal{S}') = \frac{A_{global}(\mathcal{S}')}{A_{global}(\mathcal{S}) + A_{global}(\mathcal{S}')} \ . \tag{24}$$

We see that, given the breakups G, all compatible spin configurations (i.e. all configurations $\mathcal{S}'$ for which $W(\mathcal{S}',G) \neq 0$) can be reached with a probability that depends only on the global weight A_{global}. Thus, up to A_{global}, all cluster flips are independent from each other.

Eq. (23) is equivalent to

$$W^{plaq}(\mathcal{S},G) = \Delta(\mathcal{S},G)\ V(G)\ , \quad \Delta(\mathcal{S},G) := \begin{cases} 1, & W^{plaq}(\mathcal{S},G) \neq 0, \\ 0, & \text{otherwise} \end{cases} \ , \tag{25}$$

which is the form used in ref. [6]. We shall achieve eq. (23) by choosing breakups G_p and weights $W_p(S_p,G_p)$ on every plaquette such that

$$W_p(S_p,G_p) = W_p(\mathcal{S}'_p,G_p) \tag{26}$$

is enforced by construction. The nontrivial part in this point of view is that all allowed plaquette updates $S_p \to \mathcal{S}'_p$ match for different plaquettes, to give an overall allowed update $\mathcal{S}\to\mathcal{S}'$. As we have seen in section 2.3, it is the six- (or eight-) vertex constraint, stemming from local conservation of S^z (or S^z *mod* 2) in the Hamiltonian, that makes these plaquette updates match in the form of loops. In other words, by enforcing eq. (26), we will achieve that all clusters (sets of loops that are glued together at frozen plaquettes) constructed during the breakup-step can be flipped independently, up to acceptance with p_{flip}, eq. (24). (In ref. [34] we will outline a more general construction, without independent cluster flips, suitable for models with strong asymmetries.)

Let us now find weights satisfying eq. (26). Independent cluster flips require that eq. (26) at least include the case $\mathcal{S}'_p = \overline{S}_p$, where all four spins at a plaquette are flipped:

$$W_p(S_p,G_p) = W_p(\overline{S}_p,G_p)\ , \tag{27}$$

which implies the requirement

$$W_p(S_p) \equiv \sum_{G_p} W_p(S_p,G_p) = \sum_{G_p} W_p(\overline{S}_p,G_p) = W_p(\overline{S}_p) \tag{28}$$

on the plaquette weights $W_p(S_p)$. The first step in our construction is therefore to

$$\textit{Choose}\quad A_{global}\quad \textit{such that}\quad W_p(S_p) = W_p(\overline{S}_p)\ . \tag{29}$$

Such an A_{global} always exists. It is not unique. The ideal case is $A_{global} = const$, since then $p_{flip} \equiv \frac{1}{2}$. We will comment on different choices of A_{global} in section 3.5.

For worldline models, there are a total of eight allowed spin configurations $S_p = i^{\pm} = 1^{\pm}, 2^{\pm}, 3^{\pm}, 4^{\pm}$, as shown in fig. 2. With eq. (28), the plaquette weight $W_p(S_p)$ depends only on i. Following ref. [3], let us

$$\text{Define a different "}\textbf{breakup}\text{"}\quad G_p := G^{ij} \equiv G^{ji} \text{ for every transition } \ i \leftrightarrow j\ , \tag{30}$$

such that the breakup G^{ij} allows exactly the transitions $i \leftrightarrow j$. Thus we define

$$W_p(S_p,G^{ij}) := \begin{cases} w^{ij}, & \text{if } S_p = i^{\pm} \text{ or } S_p = j^{\pm}\ , \\ 0 & \text{otherwise} \end{cases} \tag{31}$$

with suitable constants $w^{ij} \equiv w^{ji}$. We have satisfied eq. (26) by construction. By inspection of fig. 2 we see that every transition $i \leftrightarrow j$, $i \neq j$, corresponds to the flip of 2 spins on a plaquette (all four spins for $i^+ \leftrightarrow i^-$).

We also see by inspection of fig. 5 that, *given* the current worldline configuration $S_p = i^{\pm}$, we can identify each of the 4 breakups G^{ij}, $j = 1, 2, 3, 4$, with one of the *graphs* in fig. 5. Namely, flipping 2 of the spins connected in the graph for G^{ij}, $i \neq j$, leads to one of the two plaquette configurations $j^{\pm}$, flipping the other two spins leads to the other configuration, flipping all four spins maps from $i^{\pm}$ to $i^{\mp}$. Therefore, *given* a worldline configuration, the combined breakup $G = \bigcup_p G_p$ can be represented as as a graph consisting of the plaquette-graphs in fig. 5. It is however important that we distinguish between the breakups G^{ij} (there are 6 (10) different breakups in the six (eight) -vertex case), and their graphical representation in fig. 5. [2]

Since G_p connects pairs of sites, the breakups of all plaquettes will combine to give a set of clusters consisting of loops, as already described in section 2.3. By construction, the weight $W^{plaq}(\mathcal{S}, G) = \prod_p W_p(S_p, G_p)$ remains unchanged when all spins are flipped in a cluster. Thus we have achieved independent cluster flips, up to acceptance with p_{flip}. When there is no freezing, i.e. no breakups G^{ii} occur, then all clusters consist of single loops.

We still need to find constants $wij \equiv w^{ji}$, $i, j = 1, 2, 3, 4$, such that the constraint eq. (19) is satisfied, which now reads [3]

$$\begin{aligned} \textstyle\sum_j w^{ij} &= W(j)\,, \\ w^{ij} &\geq 0\,, \end{aligned} \tag{32}$$

(with $W(4) = 0$ in the six-vertex-case). This constraint underdetermines the w^{ij}. There are 3 equations for 6 unknowns in the six-vertex case, and 4 equations for 10 unknows in the eight-vertex case. It can always be solved. One explicit solution is the following: Let $W(k)$ be the smallest of the n weights $W(j)$, $j = 1, .., n$ (n is 3 or 4). Eq. (32) is satisfied by

$$\begin{aligned} w^{ij} &= W(k)/n && \text{for } i \neq j\ , \\ w^{ii} &= W(i) - \textstyle\sum_{j \neq i} w^{ij} && \text{for } i = 1, .., n\ . \end{aligned} \tag{33}$$

Experience tells us that for an efficient algorithm, one should keep the loops as independent as possible. Thus we should minimize the weights w^{ii} which cause loops to be glued together. Let $W(l)$ be the largest of the n weights $W(j)$. Given a solution w^{ij} we can always find another one in which no diagonal element w^{ii} except at most w^{ll} is nonzero [13]. For example, to remove $w^{jj}, j > 1$, define

$$\begin{aligned} w'^{\,j,j} &&&= 0 \\ w'^{\,j,j-1} &= w'^{\,j-1,j} &&= w^{j-1,j} &+\ w^{jj} \\ w'^{\,j-1,j-1} &&&= w^{j-1,j-1} &-\ w^{jj}\ . \end{aligned} \tag{34}$$

Iterating this transformation leads to the one surviving diagonal element

$$w'^{\,ll} = w^{ll} - \sum_{i \neq k} w^{ii}\ . \tag{35}$$

2.7. Ergodicity

We would like to show ergodicity, including global configuration changes, of the overall algorithm. Ergodicity is obvious when all $w^{ij} > 0$ for $i \neq j$, and when p_{flip} is always nonzero (which is the case when we use eq. (24) for p_{flip}). Any two allowed configurations (i.e. $W(\mathcal{S}) \neq 0$) are, as always, mapped into each other by a unique set of spin-flips (loop-flips), which are compatible with a set of breakups G^{ij}, $i \neq j$. With $w^{ij} > 0$, this set of breakups has a finite probability to occur, and with $p_{flip} > 0$, the two configurations will be mapped into each other in a single Monte Carlo step with finite probability. Note that the case $w^{ij} > 0$ can always be constructed, as seen in eq. (33); this may not be an efficient algorithm, however.

When some of the w^{ij} vanish, ergodicity has to be shown case by case. One example is the Heisenberg model, for which we shall show ergodicity in section 2.9. On the other hand, one can always construct weights w^{ij} such that ergodicity is not achieved, for example by choosing $w^{ij} = \delta_{ij} W(i)$, i.e. only freezing.

---

[2] It is also possible to give a common graphical representation of G^{ij} for all (ij) [13], which we shall not need here. This representation requires more than one loop-element per site.

[3] Eqs. (31),(32) are eqs. (15),(16) in ref. [3].

2.8. Summary of the loop algorithm

Since the detailed derivation of the general formalism was a bit tedious, we summarize the actual procedure here. Start with a model in worldline representation with $Z = \sum_{\mathcal{S}} W(\mathcal{S})$, eq. (11).

(1) Choose a split $W(\mathcal{S}) = A_{global}(\mathcal{S}) \times \prod_p W_p(S_p)$, eq. (18), such that $W_p(S_p) = W_p(\overline{S}_p)$, eq. (29).

(2) Find new weights $w^{ij} = w^{ji} \geq 0$ such that $\sum_j w^{ij} = W(j)$, eq. (32), while preferrably minimizing the "freezing" weights w^{ii}, see eq. (34).

Each Monte Carlo update from a worldline configuration $\mathcal{S}$ to a new configuration $\mathcal{S}'$ then involves the following steps:

(i) (Breakup) For each shaded plaquettes, with current spin configuration $i^{\pm}$, choose a breakup G^{ij} with probability $p = w^{ij}/W(i)$, eq. (22).

(ii) (Cluster identification) All plaquette breakups together subdivide the vertex lattice into a set of clusters, which consist of closed loops. Loops which have a frozen vertex ("G^{ii}") in common belong to the same cluster. Identify which sites belong to which clusters. (This can be the most time consuming task).

(iii) (Flip) Flip each cluster separately with probability p_{flip}, eq. (24), where flipping means to change the sign of S^z_{il} on all sites in the cluster. This gives the new configuration $\mathcal{S}'$. (If desired, one can artificially restrict the simulation to some sector of phase space here, e.g. to constant magnetization, by prohibiting updates that leave this sector.)

In section 2.9 we will give explicit solutions for w^{ij} for our example, the XXZ-model, and in section 2.10 we give a recipe for this case.

2.9. Graph weights for the XXZ and Heisenberg model

We shall now come back to our example and compute [2,3] the weights $w^{ij} \equiv w^{ji}$, and thus the breakup and flip probabilities, for the spin-flip symmetric six-vertex case, with weights a,b,c, eq. (6). This includes the Heisenberg model and the XXZ-model at $B = 0$ (eq. (6)) in any dimension (see section 3.2). Explicit solutions for the XYZ (eight-vertex) case have been given in ref. [13]. We need to find a solution to eq. (32). Here it reads

$$\begin{aligned} W(1) \;\equiv\; a \;&=\; w^{11} + w^{12} + w^{13} \\ W(2) \;\equiv\; c \;&=\; w^{22} + w^{12} + w^{23} \\ W(3) \;\equiv\; b \;&=\; w^{33} + w^{13} + w^{23} \; . \end{aligned} \tag{36}$$

From fig. 5 we see that w^{12}, w^{23}, and w^{13} correspond to vertical, horizontal, and diagonal breakups, respectively. The weights w^{ii} correspond to transitions $i^{\pm} \to i^{\pm}$, i.e. to flipping zero or four spins on a plaquette. They freeze the value of the weight $W(i)$. As mentioned above, experience tells us that we should minimize freezing in order to get an efficient algorithm, in which then loops are as independent as possible.

Eq. (36) has different types of solutions in different regions of the parameter space (a, b, c). Remarkably, these regions are exactly the same [2,3] as the phases of the two-dimensional classical six-vertex model [30,27], shown in fig. 3.

Let us first look at region IV, the Ising-like region of the XXZ-model, where $c > a + b$. To minimize the freezing of weight c, we have to minimize w^{22}. From eq. (36), $w^{22} = c - a - b + w^{11} + w^{33} + 2w^{13}$. With $w^{ij} \geq 0$ this implies $w^{22,\min} = c - a - b$. This minimal value of w^{22} is achieved for $w^{11} = w^{33} = 0$, i.e. when we minimize all freezing. The optimized parameters for region IV are then:

$$\begin{aligned} &w^{11} = 0, \;\; w^{33} = 0, \;\; w^{22} = c - a - b, \\ &w^{13} = 0, \;\; w^{12} = a, \;\; w^{23} = b \; . \end{aligned} \tag{37}$$

In region I the situation is technically similar. Here $a > b + c$, and the solution with minimal freezing is given by eq. (37) with indices 1 and 2 interchanged. Similarly for region II, $b > a + c$, where we obtain minimal freezing with eq. (37) with indices 2 and 3 interchanged. (Region II does not occur in the XXZ model).

Region III (XY-like, the massless phase of the six-vertex model) is characterized by $a, b, c \leq \frac{1}{2}(a + b + c)$. This includes the Heisenberg model. Here we have the ideal case that we can set all freezing probabilities to zero, obtaining

$$
\begin{aligned}
w^{11} = 0,\ \ 2w^{12} &= a + c - b\ , \\
w^{22} = 0,\ \ 2w^{13} &= b + a - c\ , \\
w^{33} = 0,\ \ 2w^{23} &= c + b - a\ .
\end{aligned}
\tag{38}
$$

We now show *ergodicity:* In each of the regions I, II, IV, one of the w^{ij} vanishes, including the case of the Heisenberg antiferromagnet and the ferromagnet. We therefore have to show ergodicity explicitely there.

Region I (including the Heisenberg FM): $w^{23} = 0$, i.e. there are only vertical and diagonal breakups (see fig. 5). These breakups permit a loop configuration which is identical to any given worldline configuration. That loop configuration will occur with finite probability. Flipping all loops in this configuration leads to the empty worldline configuration. Conversely, any worldline configuration can be generated from the empty one in a single (!) update by such a choice of loops. Therefore the algorithm is ergodic, mapping any two worldline configurations into each other in only two steps.

Region IV (including the Heisenberg AF): $w^{13} = 0$, i.e. there are only vertical and horizontal breakups. On a *bipartite* lattice with open or periodic spatial boundary conditions, ergodicity can be shown easily.[4] Start with any worldline configuration $\mathcal{S} = \{S_{xl}\}$. Our reference configuration this time is not the "empty" configuration $S'_{xl} = -1$, but instead the staggered configuration $S'_{xl} = (-1)^x S_{xl}$, i.e. the configuration with straight worldlines on one of the two sublattices. As always, there is a unique set of loops whose flips will map $\mathcal{S}$ into $\mathcal{S}'$. By inspection we see that these loops contain only vertical and horizontal breakups (horizontal where $\mathcal{S}$ has diagonal worldline parts, vertical elsewhere). Since these breakups have finite probability to occur, the whole set of loops will be constructed with finite probability. Thus, again, any worldline configuration will be mapped to the reference configuration with finite probability, and vice versa, so that on a bipartite lattice the algorithm is ergodic. Furthermore, on any lattice, the loop algorithm is at least as ergodic as the algorithm with the conventional local updates. The latter consist of spin-changes around non-shaded plaquettes, equivalent to the flip of a small loop with two vertical and two horizontal breakups, which will occur with finite probability in the loop algorithm.

For completeness, we mention region II (which does not occur in worldline models). Here there is no vertical breakup. In case of periodic spatial boundary conditions, interchange of "space" and "time" leads us to the situation of region I, for which we have shown ergodicity.

In the limit $J_x/J_z \to 0$ the XXZ model becomes the classical Ising model, since then $b \to 0$, so that there is no more hopping and all worldlines become straight. Remarkably, in this limit the loop algorithm becomes [28] the Swendsen-Wang cluster algorithm [7] ! Frozen plaquettes now connect the sites of clusters in the Ising model, i.e. they correspond to the "freezing" operation [32] of the Swendsen-Wang case !

2.10. Recipe for the spin $\frac{1}{2}$ Heisenberg antiferromagnet

In order to make the loop algorithm as clear as possible, and as an example of great practical importance, we restate the procedure for the simple case of the isotropic spin $\frac{1}{2}$ Heisenberg antiferromagnet.

A Monte Carlo update leads from a worldline configuration $\mathcal{S}$ of spin variables $S^z_{il} = \pm 1$ to a new configuration $\mathcal{S}'$. On each shaded plaquette p, the local spin configuration S_p takes one of the six possibilities shown in the left part of fig. 2, with weights $W_p(S_p)$ given in eq. (6), satisfying $a + b = c$ in the isotropic antiferromagnetic case. The weights w^{ij} in eq. (37), eq. (38), are then all zero except for

$$
w^{12} = a\ ,\ \ w^{23} = b\ . \tag{39}
$$

Therefore we get only vertical (G^{12}) and horizontal (G^{23}) breakups.

The update consists of the following steps:

(i) For each shaded plaquettes, choose the horizontal breakup with probability (see eqs. (6,22,31,37))

$$
p(S_p, G^{23}) \ = \ \delta_{S_p,2^\pm}\delta_{S_p,3^\pm}\ \ \tfrac{w^{23}}{W_p(S_p)} \ = \ \begin{cases} 0, & S_p = 1^\pm\ , \\ 1, & S_p = 2^\pm\ , \\ \tanh(\frac{\Delta\tau}{2}J) & S_p = 3^\pm\ , \end{cases} \tag{40}
$$

[4] This proof has apparently not appeared before.

otherwise choose the vertical breakup.

(ii) Identify the clusters constructed in step (i). Since there is no freezing here, all clusters consist of single loops.

(iii) Flip each loop with probability $\frac{1}{2}$, where flipping means to change the sign of S^z_{il} on all sites along the loop. This gives the new configuration $\mathcal{S}'$.

This procedure is even simpler than local worldline updates. Moreover, it remains completely unchanged in arbitrary dimensions (see section 3.2).

2.11. Reduction to a pure loop model

We can explicitely sum over the spin degrees of freedom in eq. (13) when there is no freezing and when $A_{global} \equiv 1$ (as well as in some other cases not covered here [10]). Using eqs. (11,13,21,25) we see that

$$Z = \sum_{\{\mathcal{S}\}} W(\mathcal{S}) = \sum_{\{\mathcal{S}\}} \prod_p \sum_{G_p} \Delta(S_p, G_p) V(G_p) \; . \tag{41}$$

The condition $\Delta(S_p, G_p)$ restricts the graph G to consist of clusters, i.e. divergence-free components. Without freezing, each cluster consists of a single loop. For a given G, the same condition restricts the spins on each loop to just 2 configurations: the original one, and that obtained by a loop flip. Since there is no freezing, the loops can be flipped independently. Therefore the sum over spin configurations just contributes a factor 2 for each loop, i.e.

$$Z = \sum_{G=\bigcup_p G^{ij}} \prod_p w^{ij} \, 2^{N_l(G)} \equiv \sum_G W(G) \; , \tag{42}$$

where $N_l(G)$ is the number of loops in G. We have thus mapped the original model to a *graph* model, in complete analogy with the Fortuin-Kasteleyn mapping [33] of statistical mechanics. In the six-vertex case, the graphs are those given in fig. 5. In the eight-vertex case without freezing, the number of breakup choices doubles, though they are graphically still the same, with interesting consequences for the mapping [10].

The mapping to a pure loop model is useful for analytical purposes. One could also perform a Monte Carlo simulation purely in the loop model. This has not yet been tried. It might be inefficient, since it is apparently the back-and-forth between the spin- and graph-representations which makes the loop cluster algorithm move through phase space very fast. Note that occurence of freezing might prevent the formulation of a loop *model* like eq. (42), whereas the loop-*algorithm*, which maps back and forth between spin-configurations and (spin + loop)-configurations, will still work.

The cases in which there are only vertical and horizontal, or vertical and diagonal breakups (e.g. regions I and IV of the six-vertex case) are very interesting theoretically [10]. For example, the loop model for the Heisenberg antiferromagnet consists of selfavoiding polygons, and the loop model for the Heisenberg ferromagnet has the same graphical representations as the worldlines themselves.

2.12. Improved Estimators

In addition to the reduction of autocorrelations, the combined representation eq. (13) allows a potentially drastic reduction of statistical errors by using socalled improved estimators [35,36,11,9,37]. The Monte Carlo procedure provides us with a series of configurations $\mathcal{S}_i$. For each such configuration, we decompose the checker board lattice into a set of n_i clusters by the breakup procedure. Given the worldline configuration $\mathcal{S}$ and the breakup G, we can reach any state in a set "$\mathcal{F}_i$" of 2^{n_i} worldline configurations by flipping some subset of the clusters. The probability $p(\mathcal{S})$ for each of these configurations is determined by the cluster flip probabilities p_{flip}. In the loop algorithm one configuration $\mathcal{S}_{i+1}$ will be chosen randomly according to these probabilities as the next Monte Carlo configuration.

The thermal expectation value of an observable $\mathcal{O}$ is calculated by averaging over the value of the observable in the configurations $\mathcal{S}_i$:

$$\langle \mathcal{O} \rangle = \sum_i \mathcal{O}(\mathcal{S}_i). \tag{43}$$

An improved estimator can be formally defined as the average of $\mathcal{O}(\mathcal{S})$ over the 2^{n_i} states $\mathcal{S} \in \mathcal{F}_i$ that *can* be reached from the state $\mathcal{S}_i$ via some given breakup G, instead of measuring only the value in the single state $\mathcal{S}_i$:

$$\langle \mathcal{O} \rangle \;=\; \langle \mathcal{O}_{impr} \rangle ; \;\; \mathcal{O}_{impr} \;=\; \sum_{\mathcal{S} \in \mathcal{F}_i} \mathcal{O}(\mathcal{S}) \; p(\mathcal{S}) \; , \tag{44}$$

where the probability $p(\mathcal{S})$ of the configuration $\mathcal{S}$ can be calculated as a product of the cluster flip probabilities. (Actually, $p(\mathcal{S})$ here can be different from the flip probabilities, it just needs to satisfy detailed balance with respect to $W(S, G)$, eq. (15).) To really gain an improvement we need to calculate this average over 2^{n_i} states in a time comparable to the time needed for a single measurement. Fortunately that is possible. Simpler improved estimators can often be found in the case that $p_{\text{flip}} = \frac{1}{2}$ for all clusters. In that case eq. (44) simplifies to

$$\mathcal{O}_{impr} \;=\; 2^{-n_i} \sum_{\mathcal{S} \in \mathcal{F}_i} \mathcal{O}(\mathcal{S}), \tag{45}$$

as all of the states in $\mathcal{F}_i$ now have the same probability 2^{-n_i}.

Even if the cluster flip probabilities are not all equal we can still map the Monte Carlo procedure onto one that has all probabilities fixed to $p_{\text{flip}} = \frac{1}{2}$, by fixing some of the clusters in a certain state [9]. There are many possibilities to do that. One can for example fix the state of a cluster with a probability of $p_{\text{fix}} = |2p_{\text{flip}} - 1|$. If $p_{\text{flip}} < \frac{1}{2}$ it is fixed in the old state and if $p_{\text{flip}} > \frac{1}{2}$ in the flipped state. The remaining set of clusters can then be flipped with $p_{\text{flip}} = \frac{1}{2}$. Another option is to fix a cluster in its *current* state, with suitable probability.

Let us now calculate some useful improved estimators. Consider as an example the spin correlation function $\langle S^z_{\mathbf{r},\tau} S^z_{\mathbf{r}',\tau'} \rangle$ between two spins at sites $\mathbf{r}$ and $\mathbf{r}'$ and at imaginary times τ and τ' respectively. The improved estimator is

$$\mathcal{O}_{impr} = \sum_{\mathcal{S} \in \mathcal{F}_i} S^z_{\mathbf{r},\tau}(\mathcal{S}) \; S^z_{\mathbf{r}',\tau'}(\mathcal{S}) \; p(\mathcal{S}) \tag{46}$$

As each spin can be in one cluster only, this sum can be simplified substantially to

$$\mathcal{O}_{impr} = \begin{cases} (1 - 2p_{\text{flip}}) \, (1 - 2p'_{\text{flip}}) \; \sigma\sigma', & \text{if the spins are in different clusters,} \\ \sigma\sigma', & \text{if the spins are in the same cluster,} \end{cases} \tag{47}$$

where $\sigma = \pm 1$ is the value of the worldline variable $S^z_{\mathbf{r},\tau}$ in the original state $\mathcal{S}_i$ (and thus the flipped value is $-\sigma$), and the flipping probability of this cluster is p_{flip}. Similarly the primed symbols refer to the other spin. In the case $p_{\text{flip}} = \frac{1}{2}$, the improved estimator is extremely simple:

$$\mathcal{O}_{impr} = \begin{cases} 0, & \text{if the sites } (\mathbf{r}, \tau), \, (\mathbf{r}', \tau') \text{ are in different clusters,} \\ \sigma\sigma', & \text{otherwise.} \end{cases} \tag{48}$$

The equation for the cases where one or both spins are in a cluster that has been fixed is straight forward. We see that the calculation of improved estimators of correlation functions can be performed with similar effort as the non-improved estimators. Remarkably then, the size distribution of the clusters corresponds to the spin-spin correlation function, and in general there are close relations between the loop clusters and n-point Greens functions [10].

The potential gain from using improved estimators is easy to see in the Heisenberg case. For the Heisenberg FM at momentum $q = 0$ and for the Heisenberg AF at momentum $q = \pi$, we always have $\sigma\sigma' = 1$ in eq. (48), so that the improved estimator is never negative. Yet it has the same expectation value as the unimproved estimator $\mathcal{O} = S^z_{\mathbf{r},\tau} S^z_{\mathbf{r}',\tau'} = \pm 1$. When $\langle \mathcal{O} \rangle$ is small (e.g. $\langle \mathcal{O} \rangle \sim \exp(-r/\xi)$ at large r), then the variance of $\mathcal{O}$ is

$$\langle \mathcal{O}^2 \rangle - \langle \mathcal{O} \rangle^2 \;=\; 1 - \langle \mathcal{O} \rangle^2 \;\approx\; 1 \; , \tag{49}$$

whereas the variance of $\mathcal{O}_{impr}$ is

$$\langle \mathcal{O}^2_{impr} \rangle - \langle \mathcal{O}_{impr} \rangle^2 \;=\; \langle \mathcal{O}_{impr} \rangle - \langle \mathcal{O}_{impr} \rangle^2 \;\approx\; \langle \mathcal{O}_{impr} \rangle \equiv \langle \mathcal{O} \rangle \ll 1 \; . \tag{50}$$

For a given distance r, the gain from using the improved estimator appears largest at small correlation length ξ, whereas the gain from reducing autocorrelations with the loop algorithm is largest at large ξ. Using the improved estimator can therefore reduce the variance, and thus the computer time required for a given accuracy, by a large

factor. The non-improved estimator may, however, have a sizeable amount of self-averaging, which can cancel part of this gain.

Especially simple estimators can also be derived for the uniform magnetic susceptibility, which can be expressed as the sum over all correlation functions:

$$\langle \chi \rangle = \frac{\beta}{4V} \left\langle \left(\sum_{\mathbf{r}} \frac{1}{L_t} \sum_{\tau} S^z_{\mathbf{r},\tau} \right)^2 \right\rangle, \tag{51}$$

where V is the spatial volume (number of spins) and L_t the number of time slices at which measurements are performed, and we use again $S^z_{\mathbf{r},\tau} = \pm 1$. This simplifies in the XXZ case, by using

$$\sum_{\tau} \frac{1}{L_t} \sum_{\mathbf{r}} S^z_{\mathbf{r},\tau} = \sum_{(\text{clusters c})} \sum_{((\mathbf{r},\tau) \text{in } c)} \frac{1}{L_t} S^z_{\mathbf{r},\tau} = \sum_{\text{clusters c}} w_t(c) \tag{52}$$

to the sum of the square of the temporal winding numbers $w_t(c)$ of the clusters c:

$$\langle \chi \rangle = \frac{\beta}{4V} \left\langle \sum_{\text{clusters c}} w_t(c)^2 \right\rangle \tag{53}$$

In the single-cluster variant (see section 3.1), the sum over the clusters in Eq. 53 is also calculated stochastically. Since we pick a single cluster with a probability $\frac{|c|}{V_t}$ where $|c|$ is the cluster size and V_t the number of sites in the space-time lattice, we have to compensate for this extra factor and obtain:

$$\langle \chi \rangle = \frac{\beta}{4V} \left\langle \frac{V_t}{|c|} w_t(c)^2 \right\rangle. \tag{54}$$

Further improved estimators can be constructed, including cases with a sign problem [9].

2.13. Performance

The most important advantages and limitations of the loop algorithm have already been summarized in the introduction. Let us be more explicit here. Further aspects of the performance are mentioned in the following sections.

Autocorrelations: The biggest obstacle which the loop algorithm addresses are the long autocorrelation times of worldline algorithms with local updates, as discussed in the appendices (see eq. (75)). They require a proportional increase in computer time, so that simulations for large systems and/or low temperatures quickly become impossible. The loop algorithm appears to remove these autocorrelations completely in many cases, like the spin $\frac{1}{2}$ Heisenberg AF in any dimension, the two-dimensional spin $\frac{1}{2}$ XY-model, and the spin 1 Heisenberg chain. For large systems and low temperatures this can save many orders of magnitude in computer time. As one striking example, we will show the gain in autocorrelation time for the one-dimensional Hubbard model in fig. 7 in section 4.3. Autocorrelations and critical slowing down have been carefully determined in the original loop algorithm paper [1] for the nonquantum six-vertex model, with the single-cluster variant of the loop algorithm. In [38], a related study was done in which spatial winding was allowed to vary, with similar results for autocorrelations. In the massless phase (infinite correlation length) at $\frac{a}{c} = \frac{b}{c} = \sqrt{2}$, the loop algorithm completely eliminates critical slowing down, i.e. the autocorrelation times are small and constant, with $z^{MC}_{int} \approx 0$ for all measured quantities, and $z^{MC}_{exp} = 0.19(2)$. On the KT transition line, the exponential autocorrelation times are slightly larger (up to 20 on a 256^2 lattice), with $z^{MC}_{exp} = 0.71(5)$, yet for the integrated autocorrelation times, which are relevant for MC errors, we saw barely any autocorrelations in either case, up to the largest lattices of size 256^2. In contrast, local updates indeed showed very long autocorrelation times, and $z^{MC} = 2.2(2)$, as expected.

Continuous time: In section 3.7 we shall explain how the time continuum limit $\Delta\tau \to 0$ can be taken immediately in the loop algorithm, eliminating the Trotter approximation, and further extending the accessible temperature range.

Improved Estimators: The use of improved estimators (section 2.12) provides additinional potential gains of orders of magnitudes. For example, in ref [18] it has been possible to calculate the spin-spin correlation function (which in standard updates has large variance) down to values of 10^{-5}.

Change of global quantities: Since the loops are determined locally by the breakup decisions, they can easily, "by chance", wind around the lattice in temporal or in spatial direction. An example is given in fig. 6. The flip of

such a loop then changes a global quantity (magnetization, particle number, spatial winding number). (Of course one can also choose to artificially restrict the simulation to part of the total phase space by not allowing such flips). This kind of configuration change is virtually impossible with local methods.

Note, however, that with finite magnetic field (resp. chemical potential) the global weight A_{global} changes upon such an update, resulting in potentially small flip probabilities p_{flip}. This causes autocorrelations to reappear, and does limit the system sizes and parameter regions accessible with the loop algorithm.

Asymmetries: Because of eq. (29), asymmetric weights (stemming for example from site disorder) should be taken into the global weight. See also section 3.5. In general, if there are sizeably fluctuating global weights, the performance of the loop algorithm will be impaired.

Freezing: For the loop algorithm itself, apart from effects of global weights, models which require finite freezing weights w^{ii} can be (but need not be) difficult. The intuitive argument can easily be understood. If two different loops meet at a "frozen" plaquette (i.e. one for which the breakup G^{ii} was chosen), they are glued together. If this happens at overly many plaquettes, then the cluster of glued loops which must be flipped together can occupy most of the lattice. The flip of such a cluster is not an effective move in phase space. It is basically equivalent to flipping all of the (few !) spins outside of that cluster. As an example, in ref. [1] we also investigated versions of the loop algorithm in which w^{ii} was (unnecessarily !) chosen finite. Sizeable autocorrelations were the result. Note however that it has *not* been tested whether such an impairment also occurs in cases where the *minimal* freezing is finite. As an example that freezing need not always be bad, note that, as mentioned in section 2.9 [28], the limiting case $J_{x,y} \to 0$ of the loop algorithm is the classical Swendsen-Wang cluster algorithm, in which "freezing" is the only operation. Yet this cluster algorithm can also completely eliminate critical slowing down in the corresponding classical models.

Implementation: Implementation of the loop algorithm is actually considerably easier than for local updates, which, especially in more than one dimension, require rather complicated local updates [39].

The loop algorithm can be vectorized and parallelized similarly to the Swendsen Wang cluster algorithm (see e.g. [40,41]). A vectorized version was used in ref. [20]. Vectorization or parallelization of the breakup process is trivial. The computationally dominant part is to identify the resulting clusters. This is equivalent to the well know problem of connected component labeling. For a brief discussion and references, see ref. [42]. The optimal strategies are likely to be different from the Swendsen Wang case, because loops are linear objects.

In some special cases, the loop algorithm is related to cluster algorithms for classical spin systems. When $J_x/J_z \to 0$, it becomes identical [28] to the Swendsen-Wang algorithm for the Ising model (see section 2.9). When $a = b = \frac{1}{2}c$, the loop algorithm constructs [1] the *boundaries* of the clusters which the VMR-cluster algorithm [43,44] for the (1+1)-dimensional BCSOS model produces, i.e. it constructs these clusters more efficiently.

3. GENERALIZATIONS

So far we have purposely restricted ourselves to XYZ-like (1+1)-dimensional models in order to simplify the presentation. This covered both the case of spin $\frac{1}{2}$ quantum spin models, where we have inserted eigenstates $|S^z_{il}\rangle = |\pm 1\rangle$ (or eigenstates along a different quantization axis) [5] and models of fermions or hard core bosons, where we have inserted occupation number eigenstates [5]. We have developed the formalism for the general anisotropic XYZ-like (eight-vertex-like) case. We have computed explicit update probabilities for the important XXZ-like case.

Let us now describe further generalizations, several of which are immediate. For all generalizations here, with a slight modification for those to continuous time (sections 3.7, 3.8), it remains true that *locally* on the vertices we have a situation like in the six- (or eight-) vertex model, so that the loop formulation described above can be applied directly.

3.1. Single-Cluster Variant

As in Swendsen-Wang Cluster updates, there are several ways to perform an update of the "spin" configuration $\mathcal{S}$ with the required detailed balance with respect to $W(\mathcal{S}, G)$, eq. (15). There are two important approaches:

(i) *Multi-Cluster Variant:* Determine the whole graph (set of loops) G and flip each cluster (set of mutually glued loops) in G with suitable probability p_{flip}, eq. (24).

(ii) *Single-Cluster Variant* [35,1] Pick a site (i_0, l_0) at random, and construct only the cluster that includes that site. This can be done iteratively, by following the course of the loop through (i_0, l_0) until it closes, while determining the breakups (and thus the route of the loop(s)) only on the plaquettes which are traversed. At each plaquette

at which a frozen breakup G^{ii} is chosen, the current loop is glued to the other loop traversing this plaquette. That other loop (and any loops glued to it) then also has to be constructed completely. Flip the complete cluster with probability p_{flip} to get to a new spin configuration. (Otherwise, as usual, take the old configuration unchanged as the new one.) Note that when $A_{global} = 1$, we can choose $p_{flip} = 1$ instead of $\frac{1}{2}$.

Both approaches satisfy detailed balance in eq. (15). One may think of the single-cluster variant as if all clusters had actually been constructed first, and then one of them chosen at random, by picking a site, to make an update proposal.

The advantage of the single-cluster variant [35,31] is that by picking a random site, on is likely to pick a *large* cluster, whose flip will produce a big change in the configuration and thus a large step in phase space. This can reduce critical slowing down still further. The effort (computer time) to construct the single-cluster is proportional to its length. Normalized to constant effort, one finds indeed that the single-cluster variant (and the corresponding single-cluster update for Swendsen-Wang-like algorithms) usually have even smaller dynamical critical exponents (see appendix B) than the multi-cluster variant. Note that improved estimators get a different normalization in the single-cluster variant. In some circumstances, the multi-cluster variant can still be advantageous overall, for example when employing parallel [40] or vectorized [41] computers.

3.2. Arbitrary spatial dimension

There is vitually no change algorithmically in going to higher dimensions [1], if one chooses to stay on a vertex-lattice. Let us look at two spatial dimensions as an example. The even/odd split of the Hamiltonian in eq. (2) can be generalized to

$$\hat{H} = \sum_{\nu} \hat{H}_{\nu} = \sum_{\nu} \sum_{i} \hat{H}_{i,i+\hat{\nu}} \tag{55}$$

with a separate $\hat{H}_{\nu}$ for each direction of hopping (resp. spin coupling) in the Hamiltonian $\hat{H}$. For a two-dimensional square lattice with nearest neighbor hopping we thus get 4 parts $\hat{H}_{\nu}$, each the sum of commuting pieces living on single bonds, in complete analogy with the one-dimensional case.

After the Trotter-Suzuki breakup, eq. (3), these single bonds again develop into shaded plaquettes. Each Trotter timeslice now has 4 subslices. Locally on each shaded plaquette we have the *identical* situation as in (1+1) dimension. Thus the loop algorithm can be applied *unchanged*. The only thing that changes is the way that different plaquettes connect. (Thus it is easy to write a loop-cluster program for general dimension. This contrasts with the traditional local worldline updates, where a number of different, rather complicated updates are necessary [39] to achieve acceptable performance).

The same construction can be applied as long as the lattice and the Hamiltonian admit a worldline representation in which commuting pieces of the Hamiltonian live on bonds.

3.3. Long range couplings

Hopping or spin-spin interactions beyond nearest neighbor can be handled by the same approach as higher dimensions, namely by introducing extra parts $\hat{H}_{\nu}$ in the split of the Hamiltonian, i.e. extra "bonds" on the lattice, with a corresponding set of shaded plaquettes living on separate Trotter time subslices.

This approach becomes impractical when the number of additional couplings is large. In that case, all or part of these couplings need to be moved into the global weight A_{global}, eq. (18).

3.4. Bond disorder, diluted lattices, and frustration

Bond disorder refers to spatial variations in the spin couplings J_{ij} (resp. hopping strengths t_{ij} and/or density-density couplings V_{ij}). This modifies the loop-construction probabilities *locally*, making them plaquette-dependent. Otherwise nothing changes ! (One does need to check whether ergodicity is still achieved.)

The same is true for diluted lattices, which can be viewed as a case of bond disorder in which the coupling vanishes completely on some bonds.

The situation is different for frustrated couplings whence some matrix elements become negative (and cannot be transformed to a positive representation). Whereas the loop algorithm itself is unaffected, this produces a second type of sign problem which needs to be handled in the same way as the fermion sign problem [5]. If the strength and/or frequency of frustrated matrix elements is not too large, this sign problem can remain manageable. It is possible to find improved estimators for frustrated couplings [9], to further alleviate this sign problem.

3.5. Asymmetric Hamiltonians

We have split the worldline weight in such a way into $A_{global}(\mathcal{S}) \times W^{plaq}(\mathcal{S})$, eq. (29), that W^{plaq} is symmetric with respect to a flip of all four spins on each shaded plaquette. This choice is not unique. We did this in order to keep clusters independent with respect to W^{plaq}. With this choice, any magnetic field (chemical potential) in the Hamiltonian, as well as other asymmetries, has to be taken into A_{global}, and contributes to the flip probability p_{flip} of a cluster.

Note that a magnetic field B (resp. chemical potential μ) affects only clusters which change the number of worldlines, i.e. which wind around the lattice in temporal direction. At large values of βB the acceptance rate for the flip of such clusters can become very small, slowing down the simulation considerably. (For an alternative approach, see section 3.8). To minimize this acceptance problem, one should normally choose A_{global} such that its fluctuations are minimized.

On the other hand, the couplings in W^{plaq} may cause sizeable freezing, i.e. glueing of loops, again with increased autocorrelations. It may therefore be advantageous in some cases, especially for strongly asymmetric Hamiltonians, to use a different choice of A_{global}, even with an asymmetric plaquette weight W^{plaq} [34].

3.6. Higher Spin representations

The loop algorithm for spin models has so far been formulated for the spin-$\frac{1}{2}$ case. One way to extend it to higher spin representations would be [1] to use the corresponding vertex representation (19-vertex model for spin-1) and to try the same formalism as for spin-$\frac{1}{2}$.

Kawashima and Gubernatis have successfully employed a different approach [28,6], which is an extensive generalization of the loop algorithm. They write higher spin representations as a product of spin-$\frac{1}{2}$ representations, with a projection operator onto the proper total spin. They arrive at new "shaded plaquettes", between the different spin-$\frac{1}{2}$ representations at each space-time site. Locally on each plaquette, the situation looks again like a six- or eight- vertex model. By the same approach, Kawashima also treated the anisotropic XYZ case for general spins [13]. (Note that the number of different graphs quickly proliferates, e.g. to 105 in the spin 1 case.) This generalization was successfully tested on an antiferromagnetic Heisenberg chain with $S = 1$, finding complete removal of autocorrelations [16].

3.7. Continuous time

As one of the most important generalizations, Beard and Wiese [8] have shown that within the loop formulation, one can directly take the time continuum limit $\Delta\tau \to 0$ in the Trotter-Suzuki decomposition, eq. (3).

In continuous time it is appropriate to describe worldlines by specifying the times t_i at which a worldline jumps to a different site [45] (sorted either by site or by worldline). In continuous time this jump is instantaneous.

Let us now describe the iterative construction of a single loop (see section 3.1) for the six-vertex case in a suitable language. We begin with discrete time. Let the loop be moving upwards in time along a worldline at site j. At each plaquette which it traverses, it will either continue to move vertically or, with the prescribed breakup probability, jump to the neighboring site. Depending on the worldline configuration at the current plaquette, a jump can either only be diagonal, so that the loop continues to move upwards along a worldline, or only horizontal, so that the loop changes direction and begins moving downwards in time on an empty site.

While the loop moves upwards in time at site j, and as long as the configuration of the neighboring worldlines does not change, i.e. in a "constant neighborhood" $t_1 < t < t_2$, the breakup probabilities are the same on all traversed plaquettes. In each such region we can take the continuum limit $\Delta\tau \to 0$. The probability of the loop to jump to a neighboring site on a single plaquette then transforms to a constant probability per unit of time for such a jump. The situation is the same as in radioactive decay, with a decay constant $\lambda = \lim_{\Delta\tau \to 0}(p_{breakup}/\Delta\tau)$. For example, in the Heisenberg antiferromagnet we get from eq. (40) the decay constant $\lambda = J/2$ for moving to a specific unoccupied

neighbor of site j. In places where the neighborhood changes, the jump-probability changes, too. Note that the jump-probability can also be 1, e.g. for the Heisenberg AF in those places where the worldline itself jumps, see eq. (40). Eventually the single loop closes and can be flipped as usual, with the continuum limit of the flip probability p_{flip} [9]. Cases with freezing can be handled in the same way. In addition to constructing a single-cluster, one can of course also construct all clusters, as in the multi-cluster variant.

The continuous time limit has several important advantages over the discrete time case. It removes completely the systematic error from the Trotter breakup, thus also removing the cumbersome need for calculations at several values of $\Delta\tau$ in order to extrapolate to $\Delta\tau = 0$. In addition, worldlines are now specified much more economically by just specifying their transition times. This helps especially at low temperatures, where worldlines will be mostly straight in time, by strongly reducing the storage requirements for a simulation.

The advantages of the loop algorithm are preserved, most notably ergodicity, removal or reduction of autocorrelations, and availability of improved estimators.

Another approach, the stochastic series expansion (SSE) [46], works directly in contionuous time. It is not a cluster algorithm (with the corresponding advantages for autocorrelations, improved estimators, etc.), but this algorithm is directly applicable to many models for which the loop algorithm is not well suited.

3.8. Worms

In the standard worldline formulation it is almost impossible to compute single particle Greens functions like e.g. $\langle a(x_0,t_0)\, a^\dagger(x_1,t_1)\rangle$ (where $a^\dagger, a$ are creation and annihilation operators). In order to do so, one would have to introduce sources at (x_0,t_0) and (x_1,t_1) explicitely, with a partial worldline between these two points, and to perform a separate simulation for each such pair of coordinates.

A very elegant solution to this problem was recently provided by Prokof'ev, Svistunov, and Tupitsyn [25]. Their method can be viewed from the perspective of single loop construction. *During* that construction, there is a partial loop with two open ends. Flipping this partial loop would result in a partial worldline, i.e. a propagator between two sources, just as desired. Thus every step in a single-loop construction can be taken to provide a configuration for the measurement of Greens functions [10].

Prokof'ev et al. turn this observation around and explicitely construct a single propagator with two ends ("Worms") in continuous time, in the same way that a single loop would be constructed. The Monte-Carlo-moves are thus *local* in space and in time (within a constant neighborhood, see previous section). Each local step provides a new configuration to the measurement of Greens functions. When the sources meet and annihilate (equivalent to the closing of a single loop), contact is made to the sourceless partition function, thus providing the correct normalization. (This contact can also be provided by matching the Greens function at distance 1 with measurements of the corresponding energy expectation value in the sourceless case). Prokof'ev et al. supplement these moves by additional moves corresponding to the flip of small closed loops, in order to make the simulation faster.

An additional very important advantage of the *local* updates is that *all* interactions in the Hamiltonian, like e.g. magnetic fields, can be taken into account in each step, without encountering prohibitively small acceptance rates. This is in contrast to the loop algorithm itself, which has to put unsuitable interactions into the global weight A_{global}, and which for some models (like soft core bosons) can apparently not be formulated in an efficient way. The Worm-algorithm is available for any spin-magnitude. It is ergodic in the same way that the loop algorithm is. By avoiding global acceptance probabilities, it can moreover change global quantities like the number of worldlines or the spatial winding number more easily. Note that the autocorrelation times in this algorithm, and the (Monte Carlo) dynamical critical exponent z^{MC} have not yet been measured. Because of the similarity with the loop algorithm it can be expected that they are again strongly suppressed. (They might however be larger than in the loop algorithm since the movement of worms allows backtracking.) Improved estimators are not available in the Worm-algorithm.

Prokof'ev et al. also provide a general framework for their continuous time worldline method (CTWL) for Hamiltonians with a discrete representation. They apply the method very successfully to the 1D Bose Hubbard model with soft core bosons, for which no other good algorithm exists.

4. APPLICATIONS

The loop algorithm has already been used to great advantage in a number of applications, some of which we will discuss now. Some other applications have already been mentioned in previous sections.

4.1. Spin $\frac{1}{2}$ Heisenberg Antiferromagnet

For variations of this model, the loop algorithm has been particularly valuable. It has for example allowed high precision calculations of the critical exponents of a quantum critical point [23,24]. No sign of critical slowing down has been reported in the calculations of this model.

Coupled quantum chains, also called ladder systems [47,48] have been studied by Frischmuth et al. [17] and by Greven et al. [18]. They employed up to $n_c = 6$ coupled chains of lengths up to several 100, temperatures down to about $J/50$, and various ratios of interladder to intraladder coupling $J_\perp/J$. They were able to show that even in the isotropic case $J_\perp = J$ the ladders behave in the same way as expected for $J_\perp \gg J$: For odd n_c, the systems are gapless and resemble a single chain, whereas for even n_c they show a gap, which rapidly decreases with larger n_c. Therefore the crossover from one to two dimensions in the Heisenberg AF is far from smooth. The even/odd difference is analoguous to the Haldane conjecture for single spin chains: chains of half-integer spin are gapless, whereas those of integer spin have a gap, which decreases exponentially with the spin magnitude. In ref. [18] the spin correlation function was determined and shown to be in excellent agreement with results from conformal field theory.

A three-dimensional system of coupled ladders corresponding to $LaCuO_{2.5}$ was studied by Troyer et al. [19]. At a critical ratio of the interladder to intraladder coupling, they find a quantum phase transition between a Néel ordered and a disordered state. At intermediate temperatures there is a crossover to a regime similar to uncoupled two-chain ladders with pseudogap behaviour. The simulation results for the uniform susceptibility match well with the experimental results.

The two-dimensional model on a square lattice has been investigated in several studies [11,21,8], including the paper by Beard and Wiese introducing the continuous time formulation, in which the simulation goes up to system sizes of 20×20 at $\beta = 100$. It was found that chiral perturbation theory [49] describes the model well, and the free parameters of that theory (ground state energy, staggered magnetization, and spin stiffness) were determined.

A depleted two-dimensional lattice describes the structure of the material CaV_4O_9, which exhibits a spin gap. On this lattice every fifth site is missing. There are two inequivalent coupling constants J_0 and J_1, with a quantum phase transition as a function of J_0/J_1. Troyer et al. [22–24] have performed very precise continuous time simulations of this model, with up to 20000 spins and $\beta = 100$, much larger than had been possible before with other methods. They were able to determine the static and dynamic quantum critical exponents of this transition, by way of a thorough finite size scaling analysis. The critical exponents match those expected from a mapping of the Heisenberg model to the nonlinear σ-model.

4.2. Other Spin Models

Harada and Kawashima [20] have investigated the universal jump in the helicity modulus of the 2D Quantum XY model, on lattices of up to 64^2 sites at $\beta \leq 5$, and have successfully matched it to a scaling form inferred from the Kosterlitz Thouless renormalization group equations. The helicity modulus can be measured through the expectation value $\langle W^2 \rangle$ in the worldline formulation, where W is the spatial winding number of a worldline configuration. (Since W cannot be changed efficiently in standard local updates, this observable used to be inaccessible).

Zhang et al. [50] studied 2D hard core bosons with random onsite disorder, as a model for the superfluid to insulator transition. Without the disorder this model also maps onto the 2D quantum XY model. Disorder changes its behavior, and also causes the loop algorithm to show large autocorrelation times at low temperature. Zhang et al. found a physical critical exponent $z \approx 2$ consistent with the prediction of Fisher et al. [51].

4.3. Fermionic Models

Fermionic models can be treated in the same way as hard core bosons [5], with the addition of a fermionic sign for each winding (permutation) of worldlines. In practice, this restricts the simulations to unfrustrated models on single chains, which do not have a serious sign problem, and to relatively small systems on coupled chains or with frustrated couplings. The same sign problem remains in the loop algorithm. It appears possible, however, to provide an "improved estimator" (see section 2.12) for both types of sign problem [9,10] thus extending the accessible parameter region. Still, the fermion sign problem remains the most serious limitation for these models.

The Hubbard Model can be viewed as consisting of two systems of tight binding fermions, each mapping to an XXZ-model (plus fermion sign), coupled by the Hubbard interaction $U \sum_i n_i^\uparrow n_i^\downarrow$. It can therefore immediately be simulated by employing a loop algorithm for each of the XXZ models, and taking the Hubbard interaction as well as

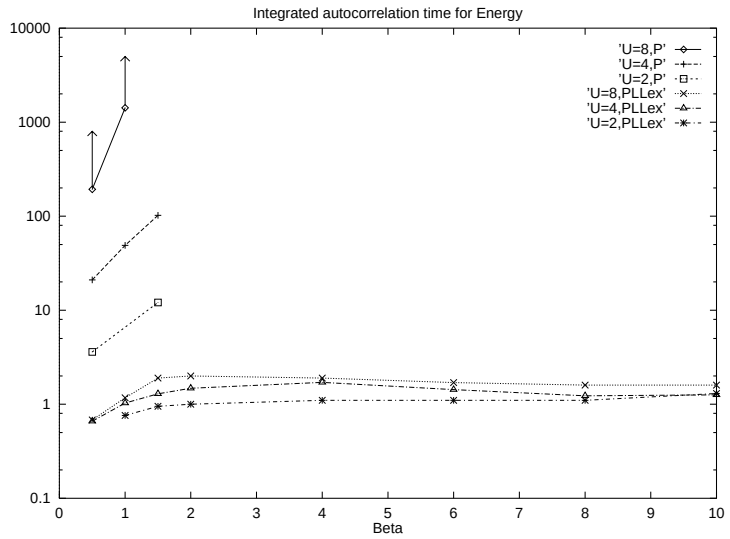

FIG. 7. Comparison of the loop algorithm and local updates for the 1D Hubbard model (32 sites), adapted from ref. [14]. The figure shows the integrated autocorrelation time τ_{int}^{E} (which is proportional to the computer time required for a given accuracy) for the total energy E, on a logarithmic scale. The autocorrelation times for other quantities behave similarly. The upper three curves are for local updates, the others for loop updates including loop exchange. In both cases, $U = 2$, 4, and 8, with autocorrelation times increasing with growing U. With *local updates* the autocorrelation times are very large already at small β, and grow rapidly when β or U are increased (roughly consistent with $\tau \sim \beta^z U^x$ with $z \lesssim 2$ and $x \gtrsim 3$). For $U = 8$ these simulations did not converge, and only lower bounds for τ are shown. At $\beta \gtrsim 1.5$ none of the simulations with local updates converged. With *loop updates* the situation improves drastically. All autocorrelation times are less than about 2. Thus at large β and U several (likely many) orders of magnitude in computer time are saved.

the chemical potential $\mu \sum_i (n_i^{\uparrow} + n_i^{\downarrow})$ into the global weight A_{global}, eq. (18). However, for large values of $|U|$ this procedure is not very efficient, since the global weight will fluctuate too strongly, resulting in small acceptance rates, especially for the flips of large loops.

Kawashima, Gubernatis, and Evertz [14] therefore added an additional new type of loop-update, called loop-exchange, which flips between spin-up and spin-down, leaving unoccupied sites in the worldline lattice unchanged. These loops move upwards in time on spin-up sites, and downwards on spin-down sites. The breakup probabilites can be constructed with the formalism of section 2. The loops were chosen to change direction (i.e. use a horizontal breakup) when spin-up and spin-down worldlines meet. Flips of these loops are not affected by the Hubbard interaction nor by the chemical potential, and can therefore always be accepted.

In the 1D Hubbard model this additional type of loop updates eliminated all remaining autocorrelations in the Hubbard simulations. An example is shown in fig. 7. For the loop updates, the autocorrelation times remain smaller than 2 at all temperatures. No slowing down is visible at all. For local updates, on the other hand, the autocorrelation times in fig. 7 show the expected rapid increase (see appendix B), consistent with $t \sim \beta^z$. They are orders of magnitude larger than the loop-autocorrelation times already at small β, and even for the energy as an observable, which as a locally defined quantity is expected to converge relatively fast in a local algorithm. Beyond $\beta \gtrsim 1.5$, the local Monte Carlo did not converge anymore. The autocorrelation times are expected to continue to grow. Note that the autocorrelation times for the local algorithm will additionally grow like $1/(\Delta\tau)^2$ for improved Trotter discretization, whereas the loop algorithm does not suffer from this effect, and can moreover be implemented directly in continuous time.

For the tJ model, a generalization of the loop algorithm has recently been developed [15,9]. Here we have three kinds of site-occupation: spin-up, spin-down, and empty. The model can be simulated by a divide and conquer strategy, using three different types of loop updates. In the first update, the empty sites in the worldline configuration are left untouched. The remaining sites with spin-up or spin-down can be updated with a loop algorithm very similar to that for the spin-$\frac{1}{2}$ Heisenberg antiferromagnet. In the second update, sites with spin-up are left untouched, and updates between spin-down and empty sites are done with a loop algorithm (which now looks similar to that for hard core particles). In the third update, loops on spin-up and empty sites are constructed. All three kinds of loops fall within the XXZ case discussed in section 2.9. The combined algorithm is indeed a working cluster algorithm for

the tJ-model, ergodic, removing autocorrelations, and providing improved estimators which further reduce statistical errors. It is successfully used in ref. [9] on single tJ-chains without and with frustrated couplings, and on two coupled tJ chains.

5. CONCLUSIONS

The loop algorithm and its generalizations have opened up exciting new opportunities. Many of them, e.g. for the two-dimensional Bose Hubbard model, remain to be investigated. A summary of advantages and limitations of the loop approach has been given in the introduction. For models in which the loop algorithm can be applied without sign problem and without overly big global weights, it offers large benefits. An example was given in fig. 7 in the previous section. An attractive alternative for less suitable models is the stochastic series expansion [46]. The recent development of the "worm" algorithm opens up further exciting possibilities. Last, but not least, the mapping to a combined spin and loop model that is the basis of the loop algorithm promises further advances on the theoretical side.

ACKNOWLEDGMENTS

I am indebted to Mihai Marcu for his long standing friendship and collaboration. Without him the loop algorithm would not have seen the light of day. I am grateful to W. Hanke and D.J. Scalapino for their support and for their ongoing interest in the loop algorithm and its applications.

APPENDIX

We review some aspects of Monte Carlo simulations, beginning with the requirements of detailed balance and ergodicity. Appendix B covers autocorrelations and their increase in critical slowing down, which can drastically increase the necessary simulation times. The loop algorithm was designed to overcome this problem. Overlooked autocorrelations can be a serious problem in practice. They can easily cause wrong results in MC simulations. Appendix C describes how to properly ensure convergence and how to calculate correct error estimates.

A. Detailed Balance and Ergodicity

There are several excellent reviews of the Monte Carlo approach, e.g. in refs. [31,52]. Here we briefly summarize some properties which we need elsewhere. The Monte Carlo procedure in classical statistical physics allows stochastic evaluation of expectation values

$$\langle O \rangle = \frac{1}{Z} \sum_{\mathcal{S} \in \{\mathcal{S}\}} O(\mathcal{S})\, W(\mathcal{S}) \tag{56}$$

with respect to the partition function $Z = \sum_{\mathcal{S} \in \{\mathcal{S}\}} W(\mathcal{S})$ and the phase space $\{\mathcal{S}\}$, by generating a Markov chain of configurations $\mathcal{S}$:

$$\mathcal{S}_{(1)},\, \mathcal{S}_{(2)},\, \mathcal{S}_{(3)},\, ... \tag{57}$$

which is distributed like $W(\mathcal{S})$. Therefore one can compute $\langle O \rangle$ from a sample of configurations

$$\langle O \rangle = \lim_{n \to \infty} \frac{1}{n} \sum_{i=1}^{n} O(\mathcal{S}_{(k+i)})\ . \tag{58}$$

(In practice, the first $k > \tau_{exp}$ configurations should be discarded to allow "thermalization" into the Boltzmann distribution. See appendix C.) A set of sufficient conditions to achieve this distribution is

(1) *Detailed Balance:* The transition probability $0 \leq\ p(\mathcal{S}_{(i)} \to \mathcal{S}_{(i+1)})\ \leq 1$ of the Markov chain satisfies

$$W(\mathcal{S})\ p(\mathcal{S} \to \mathcal{S}')\ =\ W(\mathcal{S}')\ p(\mathcal{S}' \to \mathcal{S}) \tag{59}$$

together with

(2) *Ergodicity:* Every configuration $\mathcal{S} \in \{\mathcal{S}\}$ can be reached from every other configuration with finite probability in a finite number of steps.

Solutions for detailed balance are for example the Metropolis probability

$$p(\mathcal{S} \to \mathcal{S}') = \max(1, \frac{W(\mathcal{S}')}{W(\mathcal{S})}) \tag{60}$$

and the heat bath like probability

$$p(\mathcal{S} \to \mathcal{S}') = \frac{W(\mathcal{S}')}{W(\mathcal{S}) + W(\mathcal{S}') + const} \; . \tag{61}$$

It is often advantageous, as it is for the loop algorithm, to split the weight $W(\mathcal{S})$ into two parts:

$$W(\mathcal{S}) = W_1(\mathcal{S}) \; \cdot \; W_2(\mathcal{S}) \; . \tag{62}$$

Let $p_1(\mathcal{S} \to \mathcal{S}')$ be a transition probability that satisfies detailed balance with respect to W_1. We get a Monte Carlo procedure for W by using W_2 as a "filter" to accept or reject $\mathcal{S}'$. More precisely: First apply p_1 to propose a Markov step $\mathcal{S} \to \mathcal{S}'$. Then decide with a probability $p_{accept}(\mathcal{S} \to \mathcal{S}')$ whether to take $\mathcal{S}'$ as the next configuration in the Markov chain. Otherwise keep $\mathcal{S}$. Here p_{accept} only needs to satisfy detailed balance between $\mathcal{S}$ and $\mathcal{S}'$ with respect to W_2,

$$W_2(\mathcal{S}) \; p_{accept}(\mathcal{S} \to \mathcal{S}') = W_2(\mathcal{S}') \; p_{accept}(\mathcal{S}' \to \mathcal{S}) \; . \tag{63}$$

One can easily see that the overall update satisfies detailed balance with respect to W. For p_{accept} we can for example choose the heatbath probability

$$p_{accept}(\mathcal{S} \to \mathcal{S}') = \frac{W_2(\mathcal{S}')}{W_2(\mathcal{S}) + W_2(\mathcal{S}')} \; . \tag{64}$$

Ergodicity has to be shown separately for the overall procedure.

B. Autocorrelations and Critical Slowing Down

Successive configurations $\mathcal{S}_{(1)}, \mathcal{S}_{(2)}, \mathcal{S}_{(3)}, ...$ in the Markov chain of a Monte Carlo configuration are correlated. Here we discuss the corresponding autocorrelation times, which can be extremely large. For other treatments of this topic, see e.g. references [53,31,52]. We follow refs. [31] and [14] in slighlty simplified form. Define a Monte Carlo average from n measurements

$$\overline{\mathcal{O}} := \frac{1}{n} \sum_{i=1}^{n} \mathcal{O}_{(i)} \tag{65}$$

and define the autocorrelation function for the observable $\mathcal{O}$

$$\begin{aligned} C_{\mathcal{O}\mathcal{O}}(t) \; &:= \; \langle \mathcal{O}_{(i)} \mathcal{O}_{(i+t)} \rangle - \langle \mathcal{O}_{(i)} \rangle \langle \mathcal{O}_{(i+t)} \rangle \\ \approx \; \overline{C}_{\mathcal{O}\mathcal{O}}(t) \; &:= \; \tfrac{1}{n} \textstyle\sum_{i=1}^{n} \left\{ \left(\mathcal{O}_{(i)} - \tfrac{1}{n} \sum_{i=1}^{n} \mathcal{O}_{(i)} \right) \; \left(\mathcal{O}_{(i+t)} - \tfrac{1}{n} \sum_{i=1}^{n} \mathcal{O}_{(i+t)} \right) \right\} \; , \end{aligned} \tag{66}$$

with the normalized version

$$\Gamma_{\mathcal{O}\mathcal{O}}(t) := \frac{C_{\mathcal{O}\mathcal{O}}(t)}{C_{\mathcal{O}\mathcal{O}}(0)} \; . \tag{67}$$

Typically, $\Gamma_{\mathcal{O}\mathcal{O}}(t)$ is convex and will decay exponentially at large t like $e^{-|t|/\tau}$. Define the *exponential autocorrelation time* for the observable $\mathcal{O}$ by this asymptotic decay

$$\tau^{\mathcal{O}}_{exp} := \limsup_{t \to \infty} \frac{1}{-\log |\Gamma_{\mathcal{O}\mathcal{O}}(t)|} \; . \tag{68}$$

This is the relaxation time of the slowest mode in the Monte Carlo updates which couples to $\mathcal{O}$. The slowest overall mode is

$$\tau_{exp} := \sup_{\mathcal{O}} \left(\tau_{exp}^{\mathcal{O}}\right) \tag{69}$$

and corresponds to the second largest eigenvalue of the Markov transition matrix. (The largest eigenvalue is one and has the Boltzmann distribution as eigenvector).

If the $\mathcal{O}_{(i)}$ were statistically independent, then the error estimate of $\overline{\mathcal{O}}$ would be $\sigma/\sqrt{n}$ with

$$\sigma^2 = \frac{n}{n-1} \overline{C}_{\mathcal{O}\mathcal{O}}(0) \; . \tag{70}$$

Instead, the statistical error of $\overline{\mathcal{O}}$ is controlled by the *integrated autocorrelation time*

$$\tau_{int}^{\mathcal{O}} := \frac{1}{2} + \sum_{t=1}^{\infty} \Gamma_{\mathcal{O}\mathcal{O}}(t) \tag{71}$$

and becomes $\sqrt{\sigma_{int}}/n$ with

$$\sigma_{int}^2 \; \simeq \; 2\tau_{int}^{\mathcal{O}} \; \overline{C}_{\mathcal{O}\mathcal{O}}(0) \quad \text{for } n \gg \tau_{int}^{\mathcal{O}} \; . \tag{72}$$

Therefore a Monte Carlo run of n measurements effectively contains only $n/(2\tau_{int}^{\mathcal{O}})$ independent samples for measuring $\langle\mathcal{O}\rangle$. If $\Gamma_{\mathcal{O}\mathcal{O}}(t)$ is a single exponential $e^{-t/\tau_{exp}^{\mathcal{O}}}$, i.e. if only a single mode of the Markov transition matrix couples to $\mathcal{O}$, then $\tau_{int}^{\mathcal{O}} = \tau_{exp}^{\mathcal{O}}$, otherwise $\tau_{int}^{\mathcal{O}} < \tau_{exp}^{\mathcal{O}}$.

In simulations of classical statistical systems, autocorrelation times typically grow like

$$\tau_{int,exp}^{\mathcal{O}} \sim \min(L,\xi)^{z_{int,exp}^{MC}(\mathcal{O})} \; , \tag{73}$$

where L is the linear size of the system, ξ is the physical correlation length in the infinite volume limit at the same couplings, and z^{MC} is called the (Monte-Carlo) dynamical critical exponent. In general, z^{MC} depends on the observable $\mathcal{O}$, and $z_{int}^{MC}(\mathcal{O}) \neq z_{exp}^{MC}(\mathcal{O})$. (Note that $\tau_{exp}^{\mathcal{O}}$ is a correlation time, whereas $\tau_{int}^{\mathcal{O}}$ resembles the corresponding "susceptibility"; they will in general have different critical behavior.) For local updates, one has so far always found

$$z^{MC,local} \gtrsim 2 \; . \tag{74}$$

The intuitive reason is that changes in a configuration have to spread over a distance $\min(L,\xi)$ in order to provide a statistically independent configuration. With local updates, this spread resembles a random walk with step size one [54], which needs r^2 steps to travel a distance r. (This includes the case of local updates in the determinental formalism.)

For nonrelativistic quantum simulations, space and imaginary time are asymmetric. With local Monte Carlo updates one can expect

$$\tau \; \sim \; \left\{ \max\left(\; \min(L,\xi) \, , \; \frac{1}{\Delta\tau} \min(\beta, \frac{1}{\Delta}) \; \right) \right\}^{z^{MC}} \; , \tag{75}$$

where now L and ξ are spatial lengths, $\beta/\Delta\tau = L_t$ is the temporal extent of the lattice, Δ is the energy gap, and $\Delta/\Delta\tau$ is the temporal correlation length. Again we need to distinguish $\tau_{int}^{\mathcal{O}}$ from $\tau_{exp}^{\mathcal{O}}$, and again $z^{MC} \gtrsim 2$ for local updates.

Close to phase transitions ($\xi \to \infty$) or at low temperatures and small gaps ($\Delta \to 0$), the autocorrelation times of local algorithms, with $z^{MC} \gtrsim 2$, will grow very fast. In addition, one needs to take the limit $\Delta\tau \to 0$. For local algorithms, this results in another large factor $1/(\Delta\tau)^{(z+1)}$ in required computer time. (We get "$(z+1)$" since each MC sweep has to update $\beta/\Delta\tau$ timeslices). With the loop algorithm, on the other hand, critical slowing down often disappears: $z^{MC} \approx 0$. In continuous time, the factor $1/\Delta\tau$ also disappears entirely, so that the autocorrelation times remain small even for large values of L, ξ, β, or $\frac{1}{\Delta}$.

C. Convergence and Error Calculation

Since MC measurements are correlated (see appendix B), it is not at all trivial to calculate correct statistical errors, or even to ensure convergence of a MC simulation. For error calculations, there are two good strategies in practice, *binning* and *Jackknife*, which can also be combined. To ensure convergence, it is indispensable to begin simulations on extremely small systems and at unproblematic parameters (e.g. high temperature), and to slowly increase system size, while monitoring autocorrelations through binning and/or a thorough analysis of the time series and its autocorrelation function Γ for all measured observables.

Binning: (We follow ref. [14]). Group the n measurements $\mathcal{O}(i) \equiv \mathcal{O}_{(i)}$ into k bins of length $l = n/k$ with (e.g.) $l = 2, 4, 8, ...$. Compute k bin averages

$$\overline{\mathcal{O}}_b(l) := \frac{1}{l} \sum_{i=(b-1)l+1}^{bl} \mathcal{O}(i) \quad , b = 1, .., k \tag{76}$$

and the variance of these averages

$$\sigma^2(l) := \frac{1}{n-1} \sum_{b=1}^{n} \left(\overline{\mathcal{O}}_b(l) - \overline{\mathcal{O}}\right)^2 \ . \tag{77}$$

This variance should become inversely proportional to l as the bin size l becomes large enough, whence the $\overline{\mathcal{O}}_b(l)$ as a function of b become statistically independent [53]. The expectation value of the quantity

$$\tau_{int}^{\mathcal{O}}(l) := \frac{l\sigma^2(l)}{2\sigma^2} \ , \tag{78}$$

where σ is given by eq. (70), grows monotonically in l. When statistical independence is approached, $\tau_{int}^{\mathcal{O}}(l)$ approaches the integrated autocorrelation time $\tau_{int}^{\mathcal{O}}$ from below. The converged asymptotic value of $\tau_{int}^{\mathcal{O}}(l)$ (or rather its expectation value) can therefore be used in eq. (72) to compute the actual statistical error of $\overline{\mathcal{O}}$. Note that $\tau_{int}^{\mathcal{O}}(l)$ will start to fluctuate at large l, since for finite number of measurements the number of bins k becomes small.

Convergence: If $\tau_{int}^{\mathcal{O}}(l)$ does not converge, then its expectation value at the largest l is a *lower bound* for $\tau_{int}^{\mathcal{O}}$, giving a *lower* bound for the error of $\overline{\mathcal{O}}$. In that case the MC run has not converged, and the data cannot be used to deduce physical results for $\langle\mathcal{O}\rangle$. Convergence of $\tau_{int}^{\mathcal{O}}(l)$ is a *prerequisite* for using the MC results. Since $\tau_{int}^{\mathcal{O}}$ varies for different observables $\mathcal{O}$, $\tau_{int}^{\mathcal{O}}(l)$ may have converged for some $\mathcal{O}$, and not for others. This is a dangerous situation, since the very slow modes visible in the nonconverged observables may be relevant for the apparently converged observables, too. Moreover, before starting measurements, the Monte Carlo configuration must be allowed to *thermalize*, i.e. to approach the Boltzmann distribution. It can be shown that the thermalization (from an arbitrary starting configuration) is governed by the overall exponential autocorrelation time τ_{exp}, i.e. the very largest time scale in the simulation. The thermalization time needs to be a reasonably large multiple of τ_{exp}. Therefore it is necessary to have at least an upper bound on τ_{exp} available. If an insufficient time is spent on thermalization, then the MC averages $\overline{\mathcal{O}}$ contain a systematic bias, and will converge more slowly.

An unfortunate problem in practice is that simulations may be started on an overly big system, for which – unbeknownst to the simulator – there are huge autocorrelation times. Then it may happen that within any feasible MC run, these large time scales remain invisible, so that the MC run appears to have converged, whereas in reality it has barely moved in phase space and the results may be completely wrong. (Take for example the simulation of a simple Ising model with a local algorithm at low temperature. The total magnetization takes an exponentially large time to change sign. It may never do so during the simulation, and may appear converged at a large finite value, whereas the true average magnetization is zero.) Apparently the only way to avoid this problem is to begin simulations on extremely small systems and away from problematic parameter regions, for which convergence is guaranteed by brute force. Slowly increasing system size, while measuring autocorrelation times, one can ensure that autocorrelations do not get out of hand. This appears to be the only reasonably safe procedure to ensure that a simulation produces correct converged results. Note that this approach does not require much additional computer time, since simulations on small systems will be fast.

One rather sensitive and simple instrument to detect some autocorrelations long before they are visible in a binning analysis is to simply plot the MC evolution $\mathcal{O}(i)$, $i = 1, ..., n$ graphically and to look for long correlations by eye.

Autocorrelation function: A quantitative analysis of autocorrelations beyond $\tau_{int}^{\mathcal{O}}$, e.g. in order to calculate $\tau_{exp}^{\mathcal{O}}$, requires calculation of the autocorrelation function $\Gamma_{\mathcal{O}\mathcal{O}}(t)$. This is feasible only when $\overline{\mathcal{O}}$ has converged. Contrary to

claims in the literature, one *cannot* reliably extract the integrated autocorrelation time $\tau_{int}^{\mathcal{O}}$ from $\Gamma_{\mathcal{O}\mathcal{O}}(t)$ by neglecting it beyond a "window" $t < W$ selfconsistently determined from the slope of $\Gamma_{\mathcal{O}\mathcal{O}}(W)$. Typically (but simplified [31]), we have

$$\Gamma_{\mathcal{O}\mathcal{O}}(t) = \sum_j c_j^{\mathcal{O}}\, e^{-t/\tau_j} \ , \tag{79}$$

with contributions from all eigenmodes of the Markov transition matrix, unless they are orthogonal to $\mathcal{O}$ (whence $c_j^{\mathcal{O}} = 0$). Therefore $\tau_{int}^{\mathcal{O}} \approx \sum_j c_j^{\mathcal{O}} \tau_j$ can get sizeable contributions from very large time scales τ_j, even when they couple only with small matrix elements $c_j^{\mathcal{O}}$ and are therefore not visible at small times t. This does indeed commonly happen in practice. A more reliable procedure is the following: Ensure convergence of $\overline{\mathcal{O}}$. Calculate $\Gamma_{\mathcal{O}\mathcal{O}}(t)$ for $t < t_{max}$, where t_{max} is chosen as large as possible while $\Gamma_{\mathcal{O}\mathcal{O}}(t)$ remains well above zero within error bars for all $t < t_{max}$. Compute estimates for $\tau_{exp}^{\mathcal{O}}$ and its matrix element $c_{exp}^{\mathcal{O}}$ from the (hopefully) asymptotic decay of $\Gamma_{\mathcal{O}\mathcal{O}}(t)$. Calculate $\tau_{int}^{\mathcal{O}}$ from eq. (71) by summing t up to the order of $\tau_{exp}^{\mathcal{O}}$ and computing the remainder of $\tau_{int}^{\mathcal{O}}$ from the asymptotic form of eq. (79), $\Gamma_{\mathcal{O}\mathcal{O}}(t) \sim c_{exp}^{\mathcal{O}} e^{-t/\tau_{exp}^{\mathcal{O}}}$. Of course, even this procedure will fail if the MC run is too short to show the largest autocorrelation times.

Jackknife: A binning analysis is a prerequisite for checking convergence. It also produces values for the autocorrelation times $\tau_{int}^{\mathcal{O}}$. However, it becomes rather cumbersome for quantities that are nonlinear combinations of simple observables and that require extensive error propagation (like, e.g., correlation functions in simulations with a sign-problem). In these cases, and if a safe upper bound for the autocorrelation times is known, it is much easier to analyze the data with the jackknife procedure [55]. We give a brief recipe. It starts in a similar way as the binning analysis. Split the measured values $\mathcal{O}(i)$ into k groups of length $l = n/k$, with l significantly larger than the relevant autocorrelation time $\tau_{int}^{\mathcal{O}}$. Compute the bin averages $\overline{\mathcal{O}}_b$, $b = 1, .., k$, eq. (76). Do this for all required observables $\mathcal{O}$.

Now perform the complete analysis of the MC-data, possibly highly nonlinear, a total of $k+1$ times: first with all data, leading to a result "$R^{(0)}$", then, for $j = 1, .., k$, with all data except those in bin j (i.e. pretend that bin j was never measured), leading to values "$R^{(j)}$". Then the overall result R is

$$\begin{aligned} R &= R^{(0)} - \text{Bias}, \quad \text{where} \\ \text{Bias} &= (k-1)\,(R^{av} - R^{(0)}), \\ R^{av} &= \tfrac{1}{k} \textstyle\sum_{j=1}^{k} R^{(j)} \ , \end{aligned} \tag{80}$$

with statistical error

$$\delta(R) = (k-1)^{1/2} \left(\frac{1}{k} \sum_{j=1}^{k} (R^{(j)})^2 - (R^{av})^2 \right)^{1/2} . \tag{81}$$

In this procedure, error propagation is automatic. In each of the $k+1$ analyses, almost the full set of data is used, avoiding problems in the usual analysis like potential systematic errors or instabilities in fits. It is also possible to combine Jackknife and binning by repeating the Jackknife procedure for different bin lengths and checking that the results converge when the bin lengths become long enough.

---

[1] H.G. Evertz, G. Lana, and M. Marcu, Phys. Rev. Lett. **70** (1993) 875.
[2] H.G. Evertz and M. Marcu, in *Lattice 92*, Amsterdam 1992, ed. J. Smit et al., Nucl. Phys. B (Proc. Suppl.) **30** (1993) 277.
[3] H.G. Evertz and M. Marcu, in "Quantum Monte Carlo Methods in Condensed Matter Physics", ed. M. Suzuki, World Scientific, 1994, p. 65.
[4] The loop algorithm was first presented in a talk by H.G. Evertz at *Lattice 91*, Tsukuba, Japan.
[5] R. Scalettar, this volume.
[6] N. Kawashima and J.E. Gubernatis, J. Stat. Phys. **80**, 169 (1995).
[7] R. H. Swendsen and J. S. Wang, Phys. Rev. Lett. **58** (1987) 86.
[8] B.B. Beard and U.J. Wiese, Phys. Rev. Lett. **77**, 5130 (1996).
[9] B. Ammon, H.G. Evertz, N. Kawashima, and M. Troyer, in preparation.

[10] H.G. Evertz, in preparation.
[11] U.J. Wiese and H.-P. Ying, Z. Phys. B**93** (1994) 147; Phys. Lett. A **168**, 143 (1992).
[12] M. Aizenmann and B. Nachtergaele, Comm. Math. Phys. **164**,17 (1994).
[13] N. Kawashima, J. Stat. Phys. **82**, 131 (1996).
[14] N. Kawashima, J.E. Gubernatis, and H.G. Evertz, Phys. Rev. B**50** (1994) 136.
[15] N. Kawashima, in "Computer Simulations in Condensed Matter Physics IX", ed. D.P. Landau et al., Springer Proceedings in Physics, 1996.
[16] N. Kawashima and J.E. Gubernatis, Phys. Rev. Lett. **73**, 1295 (1994).
[17] B. Frischmuth, B. Ammon, and M. Troyer, Phys. Rev. B**54**, R3714 (1996).
[18] M. Greven, R.J. Birgeneau, and U.J. Wiese, Phys. Rev. Lett. **77**, 1865 (1996).
[19] M. Troyer, M.E. Zhitomirsky, and K. Ueda, cond-mat/9606089.
[20] K. Harada and N. Kawashima, cond-mat/9702081.
[21] H.-P. Ying, U.J. Wiese, and D.-R. Ji, Phys. Lett. **A183**, 441 (1993).
[22] M. Troyer, H. Kontani, and K. Ueda, Phys. Rev. Lett. **76**, 3822 (1996).
[23] M. Troyer, M. Imada, and K. Ueda, cond-mat/9702077.
[24] M. Troyer and M. Imada, in "Computer Simulations in Condensed Matter Physics X", ed. D.P. Landau et al., Springer Proceedings in Physics, 1997, to appear.
[25] N.V. Prokof'ev, B.V. Svistunov, and I.S. Tupitsyn, cond-mat/9703200.
[26] H.F. Trotter, Proc. Am. Math. Soc. **10**, 545 (1959); M. Suzuki, Prog. of Theor. Phys. **56** 1454 (1976).
[27] R. J. Baxter, *Exactly Solved Models in Statistical Mechanics*, (Academic, New York 1989).
[28] N. Kawashima and J.E. Gubernatis, Phys. Rev. E**51**, 1547 (1995).
[29] J. M. Kosterlitz and D. J. Thouless, J. Phys. C **6**, 1181 (1973).
[30] E. H. Lieb, Phys. Rev. Lett. **18** (1967) 1046; E. H. Lieb and F. Y. Wu, in *Phase Transitions and Critical Phenomena* Vol. **1**, C. Domb and M. S. Green, editors, (Academic, 1972) p. 331.
[31] A.D. Sokal, *Bosonic Algorithms*, in "Quantum Fields on the Computer", ed. M. Creutz, 1992.
[32] D. Kandel and E. Domany, Phys. Rev. B**43** (1991) 8539.
[33] P.W. Kasteleyn and C.M. Fortuin, J. Phys. Soc. Jpn. **26**(Suppl.), 11 (1969); C.M. Fortuin and P.W. Kasteleyn, Physica **57**, 536 (1972).
[34] H.B. Schüttler and H.G. Evertz, in preparation.
[35] U. Wolff, Phys. Rev. Lett. **62** (1989) 361.
[36] U. Wolff, Nucl. Phys. B**334** (1990) 581.
[37] M. Sweeny, Phys. Rev. B **27** (1983) 4445.
[38] M. A. Novotny and H. G. Evertz, in *Quantum Monte Carlo Methods in Condensed Matter Physics*, ed. M. Suzuki (World Scientific 1993).
[39] M.S. Makivic and H.-Q. Ding, Phys. Rev. B **43**, 3562 (1991).
[40] H. Mino, Computer Phys. Comm. **66**, 25 (1991).
[41] H.G. Evertz, J. Stat. Phys. **70**, 1075 (1993).
[42] H.G. Evertz and M. Marcu, Int. J. Mod. Phys. **C4** (1993) 1147.
[43] H. G. Evertz, M. Hasenbusch, M. Marcu, K. Pinn and S. Solomon, Phys. Lett. **254B** (1991) 185, and Int. J. Mod. Phys. **C3** (1992) 235.
[44] M. Hasenbusch, G. Lana, M. Marcu and K. Pinn, Phys. Rev. B**46** (1992) 10472.
[45] E. Farhi and S. Gutmann, Ann. Phys. (N.Y.) **213**, 182 (1992).
[46] A. Sandvik, this volume.
[47] E. Dagotto and M. Rice, Science **271**, G18 (1996).
[48] H.G. Evertz, in "Computer Simulations in Condensed Matter Physics IX", ed. D.P. Landau et al., Springer Proceedings in Physics, 1996.
[49] P. Hasenfratz and F. Niedermeyer, Z. Phys. B**92**, 91 (1993).
[50] S. Zhang, N. Kawashima, J. Carlson, and J.E. Gubernatis, Phys. Rev. Lett. **74**, 1500 (1995).
[51] M.P.A. Fisher et al., Phys. Rev. B**40**, 546 (1989).
[52] See e.g. *Monte Carlo Methods in Statistical Physics*, 2nd ed., ed. by K. Binder (Springer Verlag Berlin New York 1986); *Applications of the Monte Carlo Method in Statistical Physics* 2nd ed., ed. by K. Binder (Springer Verlag Berlin New York 1987); K. Binder and D.W. Heermann, *Monte Carlo Simulation in Statistical Physics* (Springer Verlag Berlin New York 1986).
[53] M.P. Allen and D.J. Tildesley, *Computer Simulations of Liquids* (Oxford University Press, Oxford, 1987), Chap. 6.
[54] P. C. Hohenberg and B. I. Halperin, Rev. Mod. Phys. **49** (1977) 435.
[55] See e.g. C.K. Yang and D.H. Robinson, *Understanding and Learning Statistics by Computer*, World Scientific Series in Computer Sciences, Vol. 4, 1986.