5

Quantum Monte Carlo primer

In this chapter, we extend the ideas and techniques discussed in the previous chapter to introduce several basic methods for the Monte Carlo simulation of quantum many-body systems. For the sake of pedagogy, we consider only simple quantum spin-$\frac{1}{2}$ models at nonzero temperature. Some of the techniques we discuss, such as path-integral Monte Carlo methods with local updates and imaginary-time discretization, are introduced only to present key ideas and concepts in preparation for the discussion of the state-of-the-art continuous-time cluster and worm algorithms. In the remaining chapters, we discuss more advanced quantum spin algorithms and zero and finite-temperature algorithms for Fermion and Boson systems.

5.1 Classical representation

At present, all Monte Carlo simulations are performed on classical computers. With these computers we can directly manipulate only classical numbers whose products commute with each other. Quantum mechanics, however, deals with operators, most of whose products do not commute with each other. Therefore, we need to find a classical representation of our quantum system that reformulates the quantum problem in terms of classical numbers. We start by exploring a simple and obvious approach.

We are all familiar with the matrix formulation of quantum mechanics, in particular expressing the Hamiltonian operator as a Hermitian matrix. This representation is the most basic "classicization" of a quantum problem. While in general their products are noncommuting, matrices are just an ordered arrangement of classical numbers whose algebra classical computers handle easily. In particular, classical computers can diagonalize such matrices and give us all the eigenvalues and eigenvectors. This information "solves" the quantum problem.

In the quantum many-body problems of interest, the phase space is, as in the classical case, a direct product of many local phase spaces. If the size of the local space is n and we have N lattice sites, the phase space scales exponentially with the physical size of the system as n^N. The order of the Hamiltonian matrix is equal to the size of the phase space and hence exhibits an exponential scaling. Because of computer memory limitations, we can only obtain, and in fact ever expect to obtain, all the eigenvalues and eigenvectors for lattices of small size (Section 10.1). Solutions for small-sized lattices are, however, generally tainted by boundary effects and often are not indicative of the physics of the thermodynamic limit. We thus have to exploit the Monte Carlo method's ability to sample the important parts of very large phase spaces.

There are two other approaches to classicization: the high-temperature series expansion (Handscomb, 1962a) and the Feynman path-integral formulation (Suzuki, 1976a,b). The Feynman path integral converts a d-dimensional quantum problem into a $(d+1)$-dimensional classical problem. The original Hamiltonian in d dimensions acts in layers stacked along the new dimension, with a new set of interactions coupling the layers. In quantum statistical mechanics, this additional dimension is an imaginary-time axis.

There are discrete- and continuous-time versions of path-integral algorithms, and these algorithms are typically used to simulate the finite-temperature equilibrium properties of quantum many-body systems. The first generation of path-integral algorithms works with a discretized imaginary time. To eliminate discretization errors, which scale polynomially in the imaginary-time step, we have to extrapolate to the continuous imaginary-time limit. Often, we can do this at the level of the algorithm, so that the simulation results of modern algorithms are free of discretization errors.[1]

The high-temperature expansion leads to a classical representation that is formally similar to a discrete imaginary-time representation of the path-integral method. An important difference is that the high-temperature series method has an exponentially small discretization error that may be neglected for most practical purposes.

In Chapter 6, we detail both the quantum high-temperature series expansion and the path-integral formulation. In the present chapter, as part of our initial development of quantum Monte Carlo algorithms, we give a short introduction to the path-integral approach.

[1] In the past, imaginary-time steps were simply chosen small enough so that the statistical error masked the discretization error or else multiple Monte Carlo runs were made for different step sizes, and the results were fitted to a polynomial in the step size and then extrapolated to zero step size.

5.2 Quantum spins

We now develop our first quantum Monte Carlo algorithms. We begin with discrete-time path-integral representations of the spin-$\frac{1}{2}$ quantum Ising and XY models. By taking the continuous-time limit, we eventually replace the venerable discrete-time (world line) Monte Carlo methods with loop/cluster and worm algorithms. In doing so, we achieve an algorithmic evolution analogous to the one we presented for classical many-body problems in Chapter 4. Our objective is unveiling several basic features and techniques used in quantum Monte Carlo algorithms. While our discussion is in the context of specific simple finite-temperature examples, some of these features and techniques are also useful for zero-temperature problems.

5.2.1 Longitudinal-field Ising model

The quantum version of the classical one-dimensional Ising model (1.3) is given by the Hamiltonian

$$H = -\sum_{i=1}^{N} \left(J^z S_i^z S_{i+1}^z + H^z S_i^z \right). \tag{5.1}$$

Here, S_i^z is the z-component of the $S = \frac{1}{2}$ angular momentum operator at lattice site i, so its eigenvalue is $\frac{1}{2}$ or $-\frac{1}{2}$. If $H^z = 0$, this Hamiltonian is invariant under the transformation $S_i^z \to -S_i^z$ executed at each site i. If $H^z \neq 0$, it is invariant under this operation, provided the sign of H^z is reversed. Thus, this quantum model has the same symmetries as the classical model. As we will see below, this "quantum model" is actually equivalent to the classical Ising model.

In quantum statistical mechanics, the partition function is

$$Z = \mathrm{Tr}\left[\mathrm{e}^{-\beta H}\right], \tag{5.2}$$

with $\beta = 1/kT$ denoting the inverse temperature. (In the following, we set the Boltzmann constant $k = 1$.) The trace can be computed using a complete orthonormal set of basis states, which we denote as $\{|C\rangle\}$,

$$\langle C|C'\rangle = \delta(C, C'), \quad \sum_C |C\rangle\langle C| = I.$$

With such a basis, (5.2) becomes

$$Z = \sum_C \langle C| \, e^{-\beta H} \, |C\rangle \, .$$

If the $|C\rangle$ are also the eigenstates of H, that is, $H\,|C\rangle = E(C)\,|C\rangle$, it follows that $\langle C|\,e^{-\beta H}\,|C\rangle = e^{-\beta E(C)}$ and therefore that

$$Z = \sum_C e^{-\beta E(C)}. \tag{5.3}$$

Similarly, the thermal expectation value of some physical observable X becomes

$$\langle X\rangle = \frac{1}{Z}\mathrm{Tr}\Big[Xe^{-\beta H}\Big] = \frac{1}{Z}\sum_C\sum_{C'}\langle C|X|C'\rangle\langle C'|e^{-\beta H}|C\rangle. \tag{5.4}$$

In particular, when the $\{|C\rangle\}$ are orthonormal eigenstates of the Hamiltonian, we obtain

$$\langle X\rangle = \frac{1}{Z}\sum_C X(C)e^{-\beta E(C)},$$

with $X(C) = \langle C|X|C\rangle$, an expression reminiscent of an expectation value in classical statistical mechanics.

Because knowing the eigenenergies and eigenstates of H amounts to knowing the solution to the problem, the reduction of (5.2) to (5.3) is generally possible only formally. In the case of model (5.1), however, the eigenstates are

$$|C\rangle = |s_1\rangle \otimes |s_2\rangle \otimes \cdots \otimes |s_N\rangle = |s_1 s_2 \cdots s_N\rangle\,, \tag{5.5}$$

where $s_i = \pm\frac{1}{2}$ and $|s_i\rangle$ are the eigenpairs of S_i^z. Then, the eigenvalues $E(C)$ are just the energies of the classical model, apart from the classical spins s_i taking the values $\pm\frac{1}{2}$ instead of ± 1,[2] and the problem of sampling the partition function (5.3) is identical to the classical Ising problem.

5.2.2 Transverse-field Ising model

We now consider a less trivial quantum-spin model, a spin-$\frac{1}{2}$ chain in a transverse magnetic field, and explicitly derive the Feynman path integral by discretizing the imaginary-time interval. The Hamiltonian of the transverse-field Ising model is

$$H = -\sum_{i=1}^{N}\left(J^z S_i^z S_{i+1}^z + H^x S_i^x\right), \tag{5.6}$$

which we break into the two noncommuting pieces

$$H = H_0 + H_1 \tag{5.7}$$

[2] We could have made the correspondence exact by using the Pauli spin operators, instead of using the angular momentum operators, in the definition of the Hamiltonian.

defined by

$$H_0 = -J^z \sum_{i=1}^{N} S_i^z S_{i+1}^z, \quad H_1 = -H^x \sum_{i=1}^{N} S_i^x. \tag{5.8}$$

The challenge in constructing a finite-temperature quantum Monte Carlo algorithm is exponentiating the Hamiltonian. Since the current model has an exact solution (Sachdev, 1999), in principle we can do this exponentiation exactly, and we do not need Monte Carlo simulations. Our intent, however, is pedagogy, so for the time being we pretend we cannot do the exponentiation.

We begin by recasting the partition function $Z = \mathrm{Tr}\, e^{-\beta H}$ into a form with classical degrees of freedom. Noting that any operator commutes with itself, we write

$$e^{-\beta H} = \underbrace{e^{-\Delta\tau H} e^{-\Delta\tau H} \cdots e^{-\Delta\tau H}}_{M \text{ factors}}, \tag{5.9}$$

where $\Delta\tau = \beta/M$, and hence

$$Z = \sum_{C_1, C_2, \ldots, C_M} \langle C_1 | \, e^{-\Delta\tau H} \, | C_M \rangle \, \langle C_M | \, e^{-\Delta\tau H} \, | C_{M-1} \rangle \cdots \langle C_2 | \, e^{-\Delta\tau H} \, | C_1 \rangle \, . \tag{5.10}$$

This equation is the Feynman path-integral expression for the quantum mechanical partition function. The matrix products of the exponential of the Hamiltonian run through an ordered sequence of basis states. Each basis state is a configuration in phase space so we can interpret the summations in (5.10) as being over an imaginary-time ordered sequence of configurations. The sequence of configurations $C_1 \to C_2 \to \cdots \to C_M \to C_1$ defines a path through phase space. Thus, the partition function is a sum over all possible (closed) paths. In sampling the configurations, the Monte Carlo algorithm is selecting the most important of these paths.

As it stands, (5.10) is a trivial result. We can easily evaluate it if we know the matrix elements of $\exp(-\Delta\tau H)$ in the chosen basis. If we know them, the path-integral formulation is a step backward because we have to compute all the matrix products, which requires more work than simply tracing the matrix elements of $\exp(-\beta H)$. The representation (5.10) is useful in cases where we are unable to express the matrix elements of $\exp(-\Delta\tau H)$ exactly. By writing $\exp(-\beta H)$ as a product of many $\exp(-\Delta\tau H)$ factors, we introduce a small parameter $\Delta\tau$ into the problem, which enables us to obtain a good approximation for each matrix element of $\exp(-\Delta\tau H)$.

The difficulty in exponentiating H stems from the fact that S_i^x and S_j^z lack on-site ($i = j$) commutivity. The two parts of H are thus not simultaneously diagonalizable. Accordingly, the states defined in (5.5) are not eigenstates of the Hamiltonian (5.6);

they are eigenstates of H_0 only. However, in this basis (or in the eigenbasis of H_1) both $\langle C'|e^{-\Delta\tau H_0}|C\rangle$ and $\langle C'|e^{-\Delta\tau H_1}|C\rangle$ are known. Unfortunately, $e^{-\Delta\tau(H_0+H_1)} \neq e^{-\Delta\tau H_0}e^{-\Delta\tau H_1}$ if $[H_0, H_1] \neq 0$. To proceed, we need to develop a strategy for handling exponentials of the type $e^{-\Delta\tau(A+B)}$ when $[A, B] \neq 0$.

Let us consider the error incurred if we make the approximation

$$e^{-\Delta\tau(A+B)} \approx e^{-\Delta\tau A}e^{-\Delta\tau B} \approx e^{-\Delta\tau B}e^{-\Delta\tau A}. \tag{5.11}$$

For operators (and matrices), $\exp(A) = I + A + \frac{1}{2!}A^2 + \frac{1}{3!}A^3 + \cdots$ is defined by its power series. Accordingly,

$$e^{-\Delta\tau(A+B)} = e^{-\Delta\tau A}e^{-\Delta\tau B} + \mathcal{O}((\Delta\tau)^2), \tag{5.12}$$

which means that the error scales as $(\Delta\tau)^2$. A higher order (symmetric) approximation is

$$e^{-\Delta\tau(A+B)} = e^{-\Delta\tau B/2}e^{-\Delta\tau A}e^{-\Delta\tau B/2} + \mathcal{O}((\Delta\tau)^3). \tag{5.13}$$

Approximations such as (5.12) and (5.13) are called *Suzuki-Trotter approximations* (Hatano and Suzuki, 2005), or just *Trotter approximations* for short.

It is natural to choose a basis in which we can easily find the matrix elements of the exponentials of H_0 and H_1. The basis (5.5) allows a trivial exponentiation of H_0. It also permits an easy calculation of the matrix elements of $\exp(-\Delta\tau H_1)$. Using the lower-order Trotter approximation (5.12), we write $e^{-\Delta\tau H} \approx e^{-\Delta\tau H_0}e^{-\Delta\tau H_1}$ and then

$$\langle C_{k+1}|e^{-\Delta\tau H}|C_k\rangle \approx e^{-\Delta\tau E_0(C_{k+1})}\langle C_{k+1}|e^{-\Delta\tau H_1}|C_k\rangle,$$

where $E_0(C_k)$ is the classical Ising energy in zero magnetic field corresponding to the state $|C_k\rangle$ (again apart from the s_i taking the values $\pm\frac{1}{2}$ instead of ± 1).

We have to evaluate the energy $E_0(C_k)$ for each time step $\tau_k = k\Delta\tau$, and we record which spin eigenvalue belongs to which time step by attaching to them an additional subscript and writing

$$E_0(C_k) = -J^z \sum_i s_{i,k}s_{i+1,k}.$$

Consequently,

$$e^{-\Delta\tau \sum_k E_0(C_k)} = e^{\Delta\tau J^z \sum_{ik} s_{i,k}s_{i+1,k}},$$

which corresponds to the Boltzmann weight for M uncoupled classical Ising-like chains.

The remaining task is to specify the matrix elements of the exponential of the transverse-field part of the Hamiltonian. First, we note that

$$\left\langle C|e^{-\Delta\tau H_1}|C'\right\rangle = \prod_{i=1}^{N}\left\langle s_i|e^{\Delta\tau H^x S_i^x}|s_i'\right\rangle. \tag{5.14}$$

Next,

$$\begin{aligned} e^{\Delta\tau H^x S_i^x} &= I + \Delta\tau S_i^x H^x + \tfrac{1}{2!}\left(\Delta\tau H^x S_i^x\right)^2 + \tfrac{1}{3!}\left(\Delta\tau H^x S_i^x\right)^3 + \cdots \\ &= I + (\Delta\tau H^x)\, S_i^x + \tfrac{1}{4\cdot 2!}(\Delta\tau H^x)^2 I + \tfrac{1}{4\cdot 3!}(\Delta\tau H^x)^3 S_i^x + \cdots, \end{aligned}$$

where we used the fact that $(S^x)^2 = \frac{1}{4}$. Summing the power series yields

$$\langle s|e^{\Delta\tau S^x H^x}|s'\rangle = \langle s|s'\rangle \cosh \tfrac{1}{2}\Delta\tau H^x + 2\langle s|S^x|s'\rangle \sinh \tfrac{1}{2}\Delta\tau H^x.$$

Noting that the only nonzero matrix elements in the second term are

$$\langle\downarrow|S^x|\uparrow\rangle = \tfrac{1}{2}, \quad \langle\uparrow|S^x|\downarrow\rangle = \tfrac{1}{2},$$

we reexpress the matrix elements of the exponential as

$$\langle s|e^{\Delta\tau S^x H^x}|s'\rangle = Qe^{K^\tau ss'}, \tag{5.15}$$

where $s, s' = \pm\frac{1}{2}$ and

$$Q = \sqrt{\sinh \tfrac{1}{2}\Delta\tau H^x \cosh \tfrac{1}{2}\Delta\tau H^x}, \quad K^\tau = 2\ln\coth \tfrac{1}{2}\Delta\tau H^x. \tag{5.16}$$

Ultimately,

$$Z \propto \sum_{C_1,C_2,\ldots,C_M} \exp\left(\Delta\tau J^z \sum_{i,k} s_{i,k}s_{i+1,k} + K^\tau \sum_{i,k} s_{i,k}s_{i,k+1}\right). \tag{5.17}$$

This is just the partition function for a zero-field, anisotropic, two-dimensional classical Ising-like model (apart from the factor of $\frac{1}{2}$ in the fields). If we set the inverse temperature $\beta_{\text{classical}}$ of this classical model to one, we have a coupling $K^x = \Delta\tau J^z$ in the x-direction, and a field-dependent exchange K^τ in the τ-direction:

$$H_{\text{classical}} = -K^x \sum_{i,k} s_{i,k}s_{i+1,k} - K^\tau \sum_{i,k} s_{i,k}s_{i,k+1}. \tag{5.18}$$

The size of the lattice is L in the spatial dimension and $M = \beta/\Delta\tau$ in the temporal dimension. To leading order in $\Delta\tau$, the coupling constants in the spatial and temporal directions are

$$K^x = \Delta\tau J^z, \qquad K^\tau = -2\ln \tfrac{1}{2}\Delta\tau H^x. \tag{5.19}$$

See (5.16).

Let us take stock of what just happened. By expressing the matrix elements of $\exp(-\beta H)$ as a product of matrices representing $\exp(-\Delta\tau H)$, invoking the Trotter approximation, and using the eigenstates of H_0 as the basis, we obtained a two-dimensional lattice of Ising-like variables $s_{i,k}$. The matrices representing $\exp(-\Delta\tau H_0)$ produced the Boltzmann-like factors of independent, zero-field Ising-like chains. The spin variables in these chains interact only in the x-direction. The matrices representing $\exp(-\Delta\tau H_1)$ generated an interaction between these chains in the τ-direction. From the path-integral representation, it follows that this τ-direction is actually an imaginary-time direction. Expressing $\exp(-\beta H)$ as a product of factors $\exp(-\Delta\tau H)$ led to a discrete imaginary-time expression of the path integral. According to Feynman (Feynman and Hibbs, 1965), transforming field theory into quantum statistical mechanics is done by analytically continuing the path-integral's dynamical factor $\exp(-itH)$ by $it \to \tau$ and requiring periodicity in the imaginary-time direction. The trace expresses this periodicity. In mapping the d-dimensional quantum system onto a (d+1)-dimensional classical model, the Hamiltonian is not simply boosted to the extra dimension, but rather new interactions are introduced along this extra dimension to account for the imaginary-time dynamics.

In our derivation, based on the Trotter decomposition, the strategy was to separate the Hamiltonian into pieces whose exponentials we easily obtain in some basis, often the eigenbasis of one piece of the Hamiltonian. The simplest consequence is the introduction of a discretization error, which, however, as discussed below, we can completely avoid by reformulating the algorithm. A more problematic consequence (which remains even in a continuous-time formulation) is a possible *sign problem*. Because the Hamiltonian is Hermitian, its exponential is always a *positive-definite operator*, meaning that its eigenvalues are all greater than zero. This property of an operator is independent of the basis we choose to express its matrix elements. This positivity is manifest, for example, in the matrix representing the exponential of the Hamiltonian for the longitudinal-field Ising model. It is diagonal in the chosen basis, and because its diagonal elements are exponentials of some real number, they are all positive.

A positive-definite matrix differs from a *positive matrix*, that is, one having all elements greater than zero, and from a *non negative matrix*, that is, one that has no negative elements. Positivity and non negativity depend on the basis we choose to represent the operator. For Monte Carlo simulations, we want positive or non negative matrices. Quantum imaginary-time propagators, however, are only guaranteed to be positive definite. Only in a diagonal representation, in which we almost never work, are we assured that the matrix representing $\exp(-\beta H)$ is non negative. If we are unable to find a workable representation that removes the negative matrix elements, a *sign problem* may appear. This problem affects our ability to validate our results and can be so severe that the quantum Monte Carlo algorithm is unable to produce reliable estimates of the statistical error associated with computed averages. We say more about this problem in subsequent sections and chapters.

For the transverse-field model, a potential sign problem occurred, but was let pass without comment. In (5.15), the off-diagonal matrix elements are negative if H^x is negative. If so, some configurations give a negative contribution to the partition function. For the case at hand, we can avoid the sign problem by requiring, as we implicitly assumed, that $H^x > 0$. We recover the physics for $H^x < 0$ by applying a unitary transformation $(S_i^x, S_i^y, S_i^z) \to (-S_i^x, S_i^y, -S_i^z)$ to our Hamiltonian, which results in the inversion of the magnetic field $H^x \to -H^x$. This invariance is manifested by the fact that there is always an even number of spin flips caused by S_i^x along the imaginary-time direction for every spin, which allows us to simply neglect the sign associated with individual spin-flip events.

5.2.3 Continuous-time limit

The transformation of the one-dimensional quantum transverse-field Ising model to a (1+1)-dimensional classical Ising model means that we can use whatever is our favorite Monte Carlo algorithm for the classical Ising model as a quantum Monte Carlo algorithm for the transverse-field model. However, unless we take the limit $M \to \infty$ ($\Delta\tau \to 0$), (5.17) is not exact, and hence the results of our simulation have a systematic error due to the time discretization. We need to eliminate this error. A straightforward solution is to perform different Monte Carlo simulations with different values of M and extrapolate to $M \to \infty$. While this procedure works fine and was used in practice, it is no longer needed for most problems. In this section and Section 5.2.5, we extend the basic idea of the cluster algorithm discussed in Section 4.4 to two simple quantum spin models. Then, in Chapter 6, we discuss applications to various other systems.

We now explain how to take the continuous-time limit for the transverse-field Ising model (5.6). The discussion in the previous subsection showed that the discrete imaginary-time representation of this model is nothing but the two-dimensional *classical* Ising model with anisotropic interactions given by $K^x = \Delta\tau J^x$ and $K^\tau = -2\ln(\Delta\tau H^x/2)$, and with "temperature" $1/\beta_{\text{classical}} = 1$. We observe that the effective coupling constant in the time direction increases as we make $\Delta\tau$ smaller. This increase is a generic feature of the quantum-to-classical mapping that ensures that the configurations (which in this case are collections of time intervals for spin up and down) have a well-defined limit $\Delta\tau \to 0$. We can understand this by noting that the spin configurations in the discrete-time formulation are simply "low-resolution" images of the "true" continuous Feynman paths. Suppose that in the continuous limit, the average distance between spin-flips (kinks) is τ_{kink}. Then, in discrete time, we have on average one kink in every $\tau_{\text{kink}}/\Delta\tau$ time steps. To obtain such a long characteristic length (in units of $\Delta\tau$) in the temporal direction, the effective coupling in this direction must be large.

We have introduced the Swendsen-Wang algorithm as an efficient algorithm for simulating the Ising model (Section 4.4). To apply this method to the discrete-time representation of the transverse-field Ising model (5.17), we switch to Ising variables $\tilde{s}_i = \pm 1$, and couplings $\tilde{K}^x = \frac{1}{4}K^x = \frac{1}{4}\Delta\tau J^z$ and $\tilde{K}^\tau = \frac{1}{4}K^\tau = -\frac{1}{2}\ln(\frac{1}{2}\Delta\tau H^x)$. In the discrete-time cluster simulation, we then

1. Place a horizontal bond that binds two neighboring parallel spins in the space direction with probability $1 - e^{-2\tilde{K}^x} \approx \frac{1}{2}\Delta\tau J^z$.
2. Connect neighboring parallel spins in the time direction with probability $1 - e^{-2\tilde{K}^\tau} \approx 1 - \frac{1}{2}\Delta\tau H^x$.

We can replace these procedures by different ones that do not involve $\Delta\tau$ but are statistically equivalent in the limit $\Delta\tau \to 0$.

To this end, we change the "encoding system" of the configurations. In the discretized imaginary-time representation, we defined a spin on every space-time lattice point. Doing this is obviously impossible when the imaginary time is continuous, so instead of specifying the spin value on every space-time point, we specify for every lattice site only the imaginary-time locations of the kinks (the points at which the local spin value changes) and the spin values just after these kinks. Because the probability of having a kink at a given discrete-time point is proportional to $1/M$ and the number of time points is M, the total number of kinks should not strongly depend on M and should converge to some finite value in the limit $M \to \infty$. As a result, we can express a configuration C in the new Markov process as

$$\begin{aligned} C = ((\tau_1(1), \tilde{s}_1(1)), (\tau_1(2), \tilde{s}_1(2)), \cdots, (\tau_1(n_1), \tilde{s}_1(n_1)), \\ (\tau_2(1), \tilde{s}_2(1)), \cdots (\tau_N(n_N), \tilde{s}_N(n_N))), \end{aligned}$$

where $\tau_i(k)$ is the imaginary-time value of the k-th kink (that is, the k-th spin-flip) on the site i, $\tilde{s}_i(k)$ is the value of the Ising spin just after the kink, and n_i is the number of kinks on the site i. (In the case where there is no kink on site i, we just store the spin state.)

The second step in the Swendsen-Wang procedure is equivalent to placing *cuts* between neighboring pairs along the time direction with probability $e^{-2\tilde{K}^\tau} = \frac{1}{2}\Delta\tau H^x$. In the limit of $\Delta\tau \to 0$, this amounts to cutting the segments of constant spins into shorter ones by inserting cuts randomly distributed on the imaginary-time axis according to the uniform probability density $\frac{1}{2}H^x$ (Fig. 5.1).

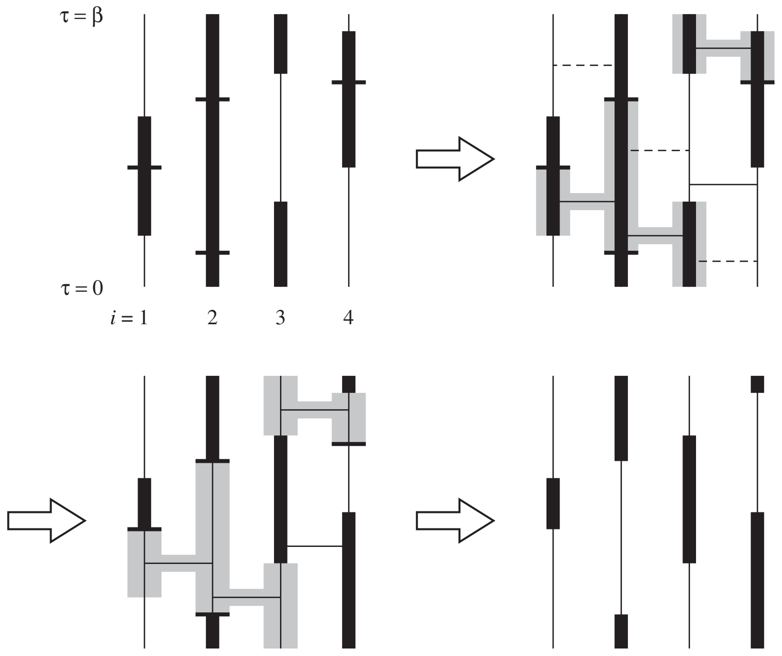

Figure 5.1 Illustration of the continuous-time cluster algorithm for the transverse-field Ising model. The vertical lines represent the imaginary-time intervals of four sites. Spin-up segments are marked by thick lines. The four steps are: (i) insertion of cuts on the imaginary-time intervals, (ii) insertion of nearest neighbor bonds (a bond, represented by a solid horizontal line, is possible only if the spin states on both sides are identical; rejected bonds are marked by dashed lines), (iii) random flipping of the clusters connected by the nearest neighbor bonds, and (iv) elimination of cuts and definition of the new segments. In the second and third panels, we highlight one of the clusters with gray shading.

In the spatial directions, on the other hand, the first step in the discrete-time Swendsen-Wang procedure has us placing *bonds* between two neighboring parallel spins with probability $1-e^{-2\tilde{K}^x} \sim \frac{1}{2}\Delta\tau J^z$. In the continuous-time limit, these bonds bind two overlapping neighboring segments with density $\frac{1}{2}J^z$ if they have the same spin value. In practice, we propose potential bond insertion times with the uniform probability density $\frac{1}{2}J^z$ (irrespective of the spin states), and then check if the spin states on the two segments connected by these bonds are identical. Only if the spin states at the location of the bond are the same is the bond inserted (solid horizontal line in Fig. 5.1); otherwise, the proposed bond is rejected (dashed horizontal line in Fig. 5.1).

The next step in the Swendsen-Wang algorithm is to identify and randomly flip the clusters of segments bound together by the spatial bonds. To do this, we use a union-find algorithm, such as the Hoshen-Kopelman algorithm (Hoshen and

Algorithm 11 Continuous imaginary-time cluster algorithm for the transverse-field Ising model.

```
Input: A configuration consisting of spin-up and spin-down segments.
  for every site do
    Remove all existing cuts at which the spin state does not change ;
    Generate new cuts by selecting the cut times $\tau \in [0, \beta)$ uniformly and
    randomly with density $\frac{1}{2}H^x$ ;
  end for
  for every pair of interacting sites do
    Select potential bond times $\tau \in [0, \beta)$ uniformly and randomly with density
    $\frac{1}{2}J^z$ ;
    for every selected $\tau$ do
      if the two sites have the same spin orientation at $\tau$ then
        Insert a horizontal bond connecting the two segments at $\tau$ ;
      end if
    end for
  end for
  Identify clusters of connected segments ;
  for every cluster do
    With probability $\frac{1}{2}$, flip the spin on all the segments in the cluster ;
  end for
  return the updated configuration.
```

Kopelman, 1976) discussed in Section 13.4. In Fig. 5.1, one example of a cluster is indicated by the gray shading. Finally, we define new segments by removing obsolete cuts. Algorithm 11 gives the pseudocode for these steps (Rieger and Kawashima, 1999).

Since we record only the positions of the segment end points and the spin state associated with each segment, the memory requirement for this algorithm (at low temperature and for $H^x \neq 0$) is proportional to the number of cuts, which is $\mathcal{O}(\beta H^x N)$. Therefore, its memory requirement is reduced compared with the discrete-time representation by at least a factor of $M/(\beta H^x)$. Although the discrete-time simulation produces qualitatively correct results as long as this factor is greater than unity, we usually need to make it much larger than unity to reduce the systematic error and obtain quantitatively correct results.

We now discuss how to generate the uniformly and randomly distributed times in Algorithm 11. It is important to note that these times and their number n are random variables, and the times are ultimately placed in ascending order, that is, $0 < \tau_1 < \tau_2 < \cdots < \tau_n < \beta$. A stochastic process that generates discrete time-ordered events with a uniform probability density λ is called a homogenous *Poisson process* (Karlin and Taylor, 1975). Not surprisingly, then, we will need to work with the Poisson distribution (2.11),

$$P(N(\beta) = n) = \frac{(\lambda\beta)^n}{n!} e^{-\lambda\beta}, \tag{5.20}$$

where $N(\beta)$ is the number of events occurring in the interval $[0, \beta)$.

We can motivate the appropriateness of this distribution in the following way: As noted in Section 2.3.1, the average number of events occurring in the interval $[0, \beta)$ is $\lambda\beta$ and hence they occur with a uniform density λ. From the functional form, it is easy to show that the Poisson distribution has the following short-time behaviors:

1. When the interval size $\Delta\tau$ is small, the probability that only one event occurs in the time interval $[\tau, \tau + \Delta\tau)$ is $\lambda\Delta\tau$; that is, it is proportional to the length of the interval.
2. The probability that no events occur in this interval is $1 - \lambda\Delta\tau$.
3. The probability that two events occur is zero to $\mathcal{O}((\Delta\tau)^2)$.

The important observation is that the short-time behavior is consistent with our limiting probabilities. The conditions on the events with respect to the number that can occur in a time $\Delta\tau$ are the same as what we implicitly assumed. Hence, to develop our sampling algorithm for the times, instead of building up to a finite interval from short-time properties, we can simply use the Poisson distribution over the targeted finite time interval $[0, \beta)$.

The functional form of the Poisson distribution leads to many interesting properties that provide the rigorous basis for the sampling algorithm. It follows directly from the definition that $P(N(\beta) = 1) = \lambda\beta \exp(-\lambda\beta)$, the exponential distribution, and $P(N(\beta) = 0) = \exp(-\lambda\beta)$. Stated without proof (Karlin and Taylor, 1975) is the important relation that

$$P(N(\tau_1 + \tau_2) - N(\tau_2) = n) = P(N(\tau_1) = n) = \frac{(\lambda\tau_1)^n}{n!} e^{-\lambda\tau_1}, \quad \tau_1, \tau_2 > 0.$$

This relation says that we can break a Poisson process into intervals, and the distribution in each interval is a Poisson distribution. Implicit in this result is the fact that nonoverlapping intervals are statistically independent.

We now use the above properties to generate the sequence of times needed for Algorithm 11. We generate one event at a time. The probability of having one event in any finite interval of time is the exponential probability whose rate constant is λ. In Section 2.3.2, we showed how to sample from this continuous probability density by generating a uniform random number ζ and then calculating

$$\tau_1 = -\ln \zeta / \lambda$$

Algorithm 12 Sequential generation of events $\tau_k < \tau_{k+1} \in [0, \beta)$ with probability density λ.

```
Input: $\lambda$, $\beta$.
  $\tau = 0$ ;
  $k = 0$ ;
  repeat
    Generate a uniform random number $\zeta \in (0, 1]$ ;
    $\tau \leftarrow \tau - \ln \zeta / \lambda$ ;
    $k \leftarrow k + 1$ ;
    $\tau_k \leftarrow \tau$ ;
  until $\tau \geq \beta$.
  $n = k - 1$ ;
  return $n$ and $\{\tau_1, \tau_2, \ldots, \tau_n\}$.
```

to obtain the first event time τ_1. Invoking the statistical independence of Poisson time intervals, we simply repeat the process to obtain $\tau_2 - \tau_1$, the time from the current event time τ_1 to the next one. Then, by the same procedures, we obtain $\tau_3 - \tau_2$, and so on, until $\sum_i \tau_i$ becomes greater than β. This strategy is summarized in Algorithm 12.

While this algorithm is effective, an alternative exists that is generally more efficient because it eliminates the calculation of logarithms. The alternative is also a simple algorithm, but its justification is not as obvious. To develop the idea, we start by considering the case when just one event occurs in the interval $[0, \beta)$ and ask how its location is distributed. We answer this question in two steps. First, we consider a Poisson process with a rate λ that we condition on $N(\beta) = 1$, that is, on only one event occurring in $[0, \beta)$. Then, we consider the arrival time τ_1 relative to any other random time ξ uniformly distributed in the interval $[0, \beta)$. From the definition of a conditional probability and the independence of events in nonoverlapping intervals

$$
\begin{aligned}
P\left(\tau_1 \leq \xi \,|N(\beta) = 1\right) &= \frac{P(\tau_1 \leq \xi, N(\beta) = 1)}{P(N(\beta) = 1)} \\
&= \frac{P(N(\xi) = 1, N(\beta) - N(\xi) = 0)}{P(N(\beta) = 1)} \\
&= \frac{\lambda \xi e^{-\lambda \xi} e^{-\lambda(\beta - \xi)}}{\lambda \beta e^{-\lambda \beta}} \\
&= \frac{\xi}{\beta}.
\end{aligned}
$$

This result says that the arrival time of a single event in the interval $[0, \beta)$ is uniformly distributed over this interval. The Poisson point is located at ξ.

If we were to generate a set of n random numbers $\{\xi_i\}$ uniformly over $[0, \beta)$, we would generate n times $\{t_{\xi_n}\}$ where each was uniformly distributed over the interval $[0, \beta)$. Of the $n!$ arrangements of these times, only one has ascending order. The set could always be placed into this order by sorting. The joint distribution (probability density) of the ordered sequence $0 < t_1 < t_2 < \cdots < t_n < \beta$ is

$$P(\tau_1 = t_1, \ldots, \tau = t_n | N(\beta) = n) = \frac{n!}{\beta^n}.$$

If we generated the needed sequence the way just implied, would this distribution be consistent with an ordered sample of n events from a Poisson distribution?

We can show that it is by again exploiting conditional probabilities and the statistical independences of a Poisson distribution:

$$\begin{aligned} P(\tau_1 = t_1, \ldots, \tau_n = t_n | N(\beta) = n) &= \frac{P(t_1, t_2, \ldots, t_n, N(\beta) = n)}{P(N(\beta) = n)} \\ &= \frac{P(t_1, t_2 - t_1, \ldots, t_n - t_{n-1}, t_{n+1} > \beta - t_n)}{P(N(\beta) = n)} \\ &= \frac{\lambda^n e^{-\lambda\beta}}{P(N(\beta) = n)} \\ &= \frac{n!}{\beta^n}. \end{aligned}$$

In going from the first to the second equation, we invoked the Poisson nature of times and intervals of time. In going from the second to the third equation, we noted that in each interval the distribution is Poisson and, except of the last interval, that only one event occurs in each interval. We went from the third to fourth equation by using the definition of the Poisson distribution. Algorithm 13 expresses this result. For a given β and λ, the first part computes $N(\beta) = n$, the second part generates the unordered set of τ_i, and the last step produces the desired result by sorting the values in this set.

5.2.4 Zero-field *XY* model

In this section and the following three, we discuss various basic concepts using the zero-field, one-dimensional, spin-$\frac{1}{2}$ quantum XY model

$$H = -\sum_{i=1}^{N} J^{xy}\left(S_i^x S_{i+1}^x + S_i^y S_{i+1}^y\right) \tag{5.21}$$

as the example. The algorithms that we discuss for this model, the loop algorithm (Section 5.2.5) and the worm algorithm (Section 5.2.6), are representative

Algorithm 13 Faster generation of uniformly distributed random events $\tau_k \in [0, \beta)$ with density λ.

```
Input: $\lambda, \beta$.
  $n = 0$ ;
  $d = e^{-\beta\lambda}$ ;
  $p = d$ ;
  Generate a uniform random number $\zeta \in [0, 1]$ ;
  while $\zeta > p$ do
    $n \leftarrow n + 1$ ;
    $d \leftarrow d\beta\lambda/n$ ;
    $p \leftarrow p + d$ ;
  end while
  for $k = 1$ to $n$ do
    Generate a uniform random number $\tau_k \in [0, \beta)$ ;
  end for
  Sort the set $\{\tau_k\}$ in ascending order ;
  return $\{\tau_k\}$.
```

of important classes of quantum Monte Carlo algorithms that have a broad range of application, as we will see in Chapter 6.

Again, we start by expressing the partition function in terms of classical degrees of freedom. While no term in this model is diagonal in the S^z basis (5.5), we continue to use this basis because of its convenience. Noting that spin operators on different sites commute, and hence all pairs of spin operators starting on odd (even) sites commute, we split H into two parts by writing

$$H = H_{\text{odd}} + H_{\text{even}},$$

where

$$H_{\text{odd}} = -\sum_{i=1}^{N/2} J^{xy}\left(S^x_{2i-1}S^x_{2i} + S^y_{2i-1}S^y_{2i}\right),$$

$$H_{\text{even}} = -\sum_{i=1}^{N/2} J^{xy}\left(S^x_{2i}S^x_{2i+1} + S^y_{2i}S^y_{2i+1}\right).$$

Making the Trotter approximation for the partition function, we find that

$$Z = \text{Tr}\left[\left(e^{-\Delta\tau H_{\text{odd}}} e^{-\Delta\tau H_{\text{even}}}\right)^M\right],$$

and then inserting a complete set of states between each factor, we find that the basic matrix element we need to evaluate is

$$w\begin{pmatrix} s_{i,k+1} & s_{i+1,k+1} \\ s_{i,k} & s_{i+1,k} \end{pmatrix} \equiv \left\langle s_{i,k+1}s_{i+1,k+1}\left|e^{\Delta\tau J^{xy}\left(S_i^xS_{i+1}^x+S_i^yS_{i+1}^y\right)}\right|s_{i,k}s_{i+1,k}\right\rangle.$$

Evaluating it is relatively straightforward. First, for $i \neq j$, and with $S^+ = (S_x + iS_y)$, $S^- = (S_x - iS_y)$,

$$S_i^xS_j^x + S_i^yS_j^y = \tfrac{1}{2}\left(S_i^+S_j^- + S_i^-S_j^+\right).$$

Using this in the argument of the exponential and expressing the exponential as a power series, we then explicitly consider term by term the possible matrix elements, finding that

$$\begin{pmatrix} w\binom{\uparrow\uparrow}{\uparrow\uparrow} & w\binom{\uparrow\downarrow}{\uparrow\uparrow} & w\binom{\downarrow\uparrow}{\uparrow\uparrow} & w\binom{\downarrow\downarrow}{\uparrow\uparrow} \\ w\binom{\uparrow\uparrow}{\uparrow\downarrow} & w\binom{\uparrow\downarrow}{\uparrow\downarrow} & w\binom{\downarrow\uparrow}{\uparrow\downarrow} & w\binom{\downarrow\downarrow}{\uparrow\downarrow} \\ w\binom{\uparrow\uparrow}{\downarrow\uparrow} & w\binom{\uparrow\downarrow}{\downarrow\uparrow} & w\binom{\downarrow\uparrow}{\downarrow\uparrow} & w\binom{\downarrow\downarrow}{\downarrow\uparrow} \\ w\binom{\uparrow\uparrow}{\downarrow\downarrow} & w\binom{\uparrow\downarrow}{\downarrow\downarrow} & w\binom{\downarrow\uparrow}{\downarrow\downarrow} & w\binom{\downarrow\downarrow}{\downarrow\downarrow} \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cosh\frac{1}{2}\Delta\tau J^{xy} & \sinh\frac{1}{2}\Delta\tau J^{xy} & 0 \\ 0 & \sinh\frac{1}{2}\Delta\tau J^{xy} & \cosh\frac{1}{2}\Delta\tau J^{xy} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{5.22}$$

A number of things have happened. The odd-even site breakup of the Hamiltonian and the Trotter decomposition created a structure consisting of $2M$ layers in the imaginary-time direction. Pairs of spin operators starting at even sites act on layers $k = 1, 3, 5, \ldots$, while pairs starting at odd sites act on layers $k = 2, 4, 6, \ldots$ Two subsequent layers correspond to the propagation of the full Hamiltonian by one time-step $\Delta\tau$.

The local transition matrix depends only on the values of two adjacent spin variables on one layer and those of the same pair of sites on the next layer. These four spins are located at the positions (i,k), $(i+1,k)$, $(i+1,k+1)$, and $(i,k+1)$, which define a *plaquette*. We notice that these plaquettes naturally decompose space-time into a checkerboard structure (Fig. 5.2) and that we need to sample only the light or the dark plaquettes.

Associated with each plaquette are 16 possible spin configurations. However, (5.22) says that only six of them are allowed. The nonzero matrix elements express a *conservation law*,

$$s_{i,k} + s_{i+1,k} = s_{i,k+1} + s_{i+1,k+1}, \tag{5.23}$$

for the z-component of total angular momentum at each plaquette. Accordingly, the total z-component of angular momentum $S^z \equiv \sum_i S_i^z$ is conserved at each imaginary

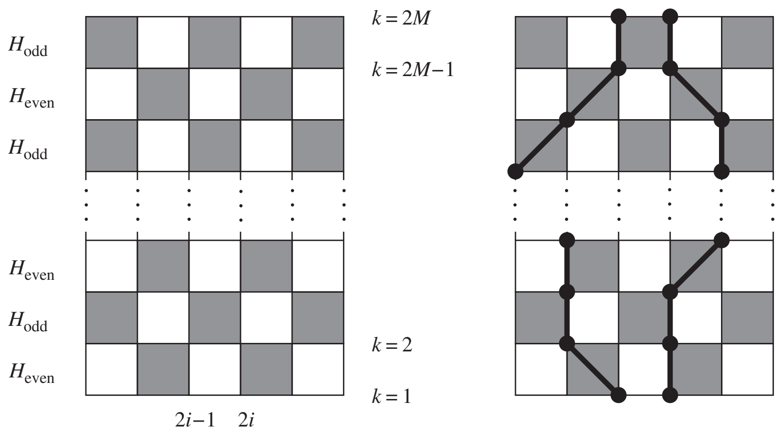

Figure 5.2 The checkerboarding of space and imaginary time in one dimension (left). World lines on the checkerboard (right).

time. In other words, the number of up-spins on a spatial plane is independent of the imaginary time. This is an example of a reduction of phase space due to symmetry. While the quantum Ising model was invariant under spin-reversal, the *XY* model is invariant under arbitrary rotations of all spins about the z-axis. This invariance conserves S^z. This conservation law demands that if we change the number of up-spins on a spatial plane during the Monte Carlo simulation, we must change it in the same way on all the other planes as well. A peculiar implication of the invariance and conservation law is that the value of S^z that results from our initialization of the simulation is the value that remains throughout, unless we introduce some "global flip" procedure.

Let us emphasize that (5.23) not only implies the conservation of the total S^z but expresses a local conservation law, involving only the four spin variables of a plaquette. This restriction of the phase space leads to a challenge in defining a sampling strategy. Going site to site and proposing local spin reversals via the Metropolis algorithm is not consistent with the conservation law. An ergodic sampling must involve multiple spins simultaneously.

If we mark the corners of the shaded plaquettes that have an up-spin with a dot and connect the dots with a line, with the rule that on the all-up plaquette the lines run up the left and the right edges (so the lines do not intersect), we obtain a collection of continuous paths, looping in the imaginary-time direction (see right panel of Fig. 5.2). Each such path is called a *world line*. Monte Carlo sampling then becomes a matter of deforming the world lines. There is a particularly convenient way of doing this while maintaining spin conservation: We visit each *unshaded* plaquette in turn, and if there is a segment running along one edge and not the

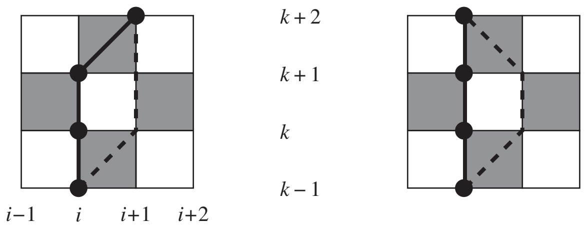

Figure 5.3 Basic world-line update. The world line may move from the solid line to the dashed one and vice versa.

other, we attempt to move it from one edge to the other. We illustrate these moves in Fig. 5.3, where we assume that the unshaded plaquette is located at

$$\begin{pmatrix} s_{i,k+1} & s_{i+1,k+1} \\ s_{i,k} & s_{i+1,k} \end{pmatrix}.$$

The move then is accepted or rejected via a Metropolis decision based on

$$P_{\text{accept}} \equiv \min[1, R_{\text{north}} R_{\text{south}} R_{\text{east}} R_{\text{west}}],$$

where

$$R_{\text{north}} = \frac{w\begin{pmatrix} s_{i,k+2} & s_{i+1,k+2} \\ -s_{i,k+1} & -s_{i+1,k+1} \end{pmatrix}}{w\begin{pmatrix} s_{i,k+2} & s_{i+1,k+2} \\ s_{i,k+1} & s_{i+1,k+1} \end{pmatrix}}, \quad R_{\text{south}} = \frac{w\begin{pmatrix} -s_{i,k} & -s_{i+1,k} \\ s_{i,k-1} & s_{i+1,k-1} \end{pmatrix}}{w\begin{pmatrix} s_{i,k} & s_{i+1,k} \\ s_{i,k-1} & s_{i+1,k-1} \end{pmatrix}},$$

$$R_{\text{east}} = \frac{w\begin{pmatrix} -s_{i+1,k+1} & s_{i+2,k+1} \\ -s_{i+1,k} & s_{i+2,k} \end{pmatrix}}{w\begin{pmatrix} s_{i+1,k+1} & s_{i+2,k+1} \\ s_{i+1,k} & s_{i+2,k} \end{pmatrix}}, \quad R_{\text{west}} = \frac{w\begin{pmatrix} s_{i-1,k+1} & -s_{i,k+1} \\ s_{i-1,k} & -s_{i,k} \end{pmatrix}}{w\begin{pmatrix} s_{i-1,k+1} & s_{i,k+1} \\ s_{i-1,k} & s_{i,k} \end{pmatrix}} \tag{5.24}$$

are the ratios of the local weights before and after the proposed change in the world-line configuration, for plaquettes to the north, south, east, and west of the unshaded plaquette. The proposal is reversing the spins on the border of the unshaded plaquette, provided the spin states on the two sides are different (Algorithm 14).

We notice from (5.22) that we have a sign problem if $J^{xy} < 0$. If our lattice is *bipartite*, that is, the sites separate into an A and B sublattice so that only B sites are nearest neighbors to A sites and vice versa, then we can remove the problem by performing a canonical transformation that reverses the spins on one sublattice (Marshall, 1955). Square lattices are bipartite, while triangular lattices

Algorithm 14 World-line Monte Carlo method for the one-dimensional spin-$\frac{1}{2}$ *XY* model with local updates. (5.22) defines the weights w in (5.24).

```
Input: A spin configuration $\{s_{i,k}\}$.
  for every unshaded plaquette $(s_{i,k}, s_{i+1,k}, s_{i+1,k+1}, s_{i,k+1})$ do
    if $s_{i,k} = s_{i,k+1}$ and $s_{i+1,k} = s_{i+1,k+1}$ and $s_{i,k} = -s_{i+1,k}$ then
      Compute $R_{\text{north}}, R_{\text{south}}, R_{\text{east}}, R_{\text{west}}$ by (5.24) ;
      Generate a uniform random number $\zeta \in [0, 1]$ ;
      if $\zeta < R_{\text{north}} R_{\text{south}} R_{\text{east}} R_{\text{west}}$ then
        $s_{i,k} \leftarrow -s_{i,k}$ ;
        $s_{i+1,k} \leftarrow -s_{i+1,k}$ ;
        $s_{i+1,k+1} \leftarrow -s_{i+1,k+1}$ ;
        $s_{i,k+1} \leftarrow -s_{i,k+1}$ ;
          ▷ Need to account for boundary conditions.
      end if
    end if
  end for
  return the updated spin configuration.
```

are not. Nonbipartite lattices often frustrate local spin configurations. For frustrated systems, the sign problem generally persists.

As in the case of the discrete-time algorithm for the transverse-field Ising model, we have to remove the systematic error caused by the Trotter approximation by extrapolating to the limit $M \to \infty$. Fortunately, we can again replace the discrete-time algorithm by a continuous-time one. Since our restructuring of the partition function of the *XY* model did not produce one analogous to the classical model, we cannot directly apply the Swendsen-Wang algorithm and proceed from there. However, as discussed in the next subsection, we can do something analogous.

5.2.5 Simulation with loops

We now generalize the cluster scheme, which was first introduced in Section 4.4 and reformulated for the transverse-field Ising model in Section 5.2.3, to the path-integral representation of the partition function of quantum systems. Here we use the *XY* model as our example. The generalization to the transverse-field Ising model discussed in Section 5.2.3 was rather straight forward through the mapping of the path integral with discretized imaginary time to the $(d + 1)$-dimensional classical Ising model. The generalization to the *XY* model in the present section involves something more, namely, the introduction of various types of graph elements

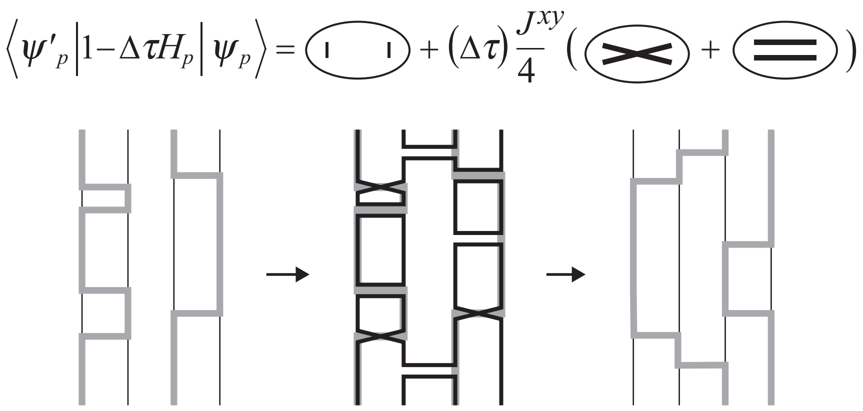
$\langle \psi'_p | 1 - \Delta\tau H_p | \psi_p \rangle = $ + $(\Delta\tau)\frac{J^{xy}}{4}$ (+)

Figure 5.4 Graph elements of the loop algorithm for the spin-$\frac{1}{2}$ XY model, and one Monte Carlo step.

according to the nature of the interaction terms. In the present section, we focus on "how" rather than "why" and postpone a detailed derivation and justification of the proposed cluster-type algorithm to Chapter 6.

As in the case of the transverse-field Ising model, we obtain the continuous-time version of this algorithm simply by reinterpreting stochastic assignments by distributing cuts and bonds to imaginary-time intervals with a certain density, instead of assigning them to individual plaquettes. While we focus on the one-dimensional spin-$\frac{1}{2}$ XY model, the generalization of the cluster updating scheme to higher dimensions and to other types of Hamiltonians is straightforward in many cases (Chapter 6).

The first step is to express the local imaginary-time evolution operator in terms of Kronecker deltas. We start with essentially the same matrix elements of the imaginary-time evolution operator as (5.22) but correct only up to the first order in $\Delta\tau$.[3] For convenience, we add a constant $-\frac{1}{4}J^{xy}$ to the pair Hamiltonian and then express the matrix elements as

$$\begin{aligned}
&\langle s_i' s_j' | 1 - \Delta\tau H_{ij} | s_i s_j \rangle \\
&\quad = \langle s_i' s_j' | s_i s_j \rangle + \tfrac{1}{2}\Delta\tau J^{xy} \langle s_i' s_j' | (S_i^+ S_j^- + S_i^- S_j^+) + \tfrac{1}{2} | s_i s_j \rangle \\
&\quad = \delta_{s_i', s_i}\delta_{s_j', s_j} + \tfrac{1}{4}\Delta\tau J^{xy} (\delta_{s_i', s_j}\delta_{s_j', s_i} + \delta_{s_i, -s_j}\delta_{s_i', -s_j'}). \qquad (5.25)
\end{aligned}$$

There are the three terms, which we identify with the three different graph elements, encircled in Fig. 5.4. To convince ourselves that this graph assignment

[3] This "approximation" does not yield any systematic error in the end because we take the continuous imaginary-time limit.

is reasonable, let us examine the contributions of the different plaquette configurations. The second $\delta_{s'_i,s_j}\delta_{s'_j,s_i}$ term contributes $\frac{1}{4}\Delta\tau J^{xy}$ in the plaquette configurations

$$\begin{pmatrix} \uparrow & \uparrow \\ \uparrow & \uparrow \end{pmatrix}, \quad \begin{pmatrix} \downarrow & \downarrow \\ \downarrow & \downarrow \end{pmatrix}, \quad \begin{pmatrix} \uparrow & \downarrow \\ \downarrow & \uparrow \end{pmatrix}, \quad \begin{pmatrix} \downarrow & \uparrow \\ \uparrow & \downarrow \end{pmatrix},$$

while the third $\delta_{s_i,-s_j}\delta_{s'_i,-s'_j}$ term contributes $\frac{1}{4}\Delta\tau J^{xy}$ in the configurations

$$\begin{pmatrix} \uparrow & \downarrow \\ \uparrow & \downarrow \end{pmatrix}, \quad \begin{pmatrix} \downarrow & \uparrow \\ \downarrow & \uparrow \end{pmatrix}, \quad \begin{pmatrix} \uparrow & \downarrow \\ \downarrow & \uparrow \end{pmatrix}, \quad \begin{pmatrix} \downarrow & \uparrow \\ \uparrow & \downarrow \end{pmatrix}.$$

In all the configurations contributing to the second term, the spin variables located on the diagonals are parallel, whereas in those contributing to the third term, the spin variables in the same row are antiparallel.

Expression (5.25) leads us to the continuous-time cluster algorithm for the *XY* model in question. Specifically, we identify the first term with a "vertical" graph element, the second term with a "diagonal" graph element, and the third term with a "horizontal" graph element. Let us first consider the positions where the world lines have no kinks. The vertical graph element corresponds to the identity operator. As for the diagonal graph element, because its weight is $\frac{1}{4}J^{xy}\Delta\tau$ when $s_i = s_j$, we generate this graph element with density $\frac{1}{4}J^{xy}$ on the imaginary-time interval if the two spins are parallel to each other. This graph element imposes the restriction on the local spin configuration that two spins located at diagonally opposite corners must be parallel to each other, and we thus represent it by a pair of (disconnected) diagonal lines, as in Fig. 5.4. Similarly, the third term in (5.25) gives the weight $\frac{1}{4}J^{xy}\Delta\tau$ for $s_i = -s_j$. Therefore, we insert the corresponding graph element with density $\frac{1}{4}J^{xy}$ to intervals with antiparallel spins. The Kronecker delta functions representing this term indicate that we bind the antiparallel spins horizontally, again as illustrated in Fig. 5.4. At the positions where the world lines have kinks, only diagonal or horizontal elements can match. As the weight of these two are equal, we choose one of these elements with equal probability for each kink.

Because all three graph elements connect the four points on the plaquette pairwise, the resulting clusters are loops, as illustrated in the middle panel of Fig. 5.4 (hence the name *loop algorithm*). Whether a cluster algorithm becomes a loop algorithm or not depends on the nature of the Hamiltonian.

Taking the continuous-time limit $\Delta\tau \to 0$, similar to the case of the transverse-field Ising model, just becomes a matter of switching from the lattice point representation to the segment representation and placing the graph elements with appropriate density, except at the locations of the kinks. Algorithm 15 summarizes the loop algorithm in its continuous-time version.

Algorithm 15 Loop algorithm for the spin-$\frac{1}{2}$ XY model.

```
Input: A world-line configuration ;
  for all kinks do
    Select a diagonal or horizontal graph element with probability $\frac{1}{2}$, and assign it to the kink ;
  end for
  for all pairs of neighboring sites do
    Generate graph positions $\tau \in [0, \beta)$ with density $\frac{1}{4}J^{xy}$ ;
    Place diagonal (horizontal) graph elements if the spins are parallel (antiparallel) ;
  end for
  Identify loops ;
  for all loops do
    Flip the loop with probability $\frac{1}{2}$ ;
  end for
  return the updated world-line configuration.
```

5.2.6 Simulation with worms

Instead of decomposing the entire lattice into loops and computing a new world-line configuration by random flipping of these loops, we can also update the world-line configuration by creating worms and letting them move, as in the worm update for the Ising model discussed in Section 4.5. Here the essential idea is the same: Extend the configuration space by introducing points of discontinuity, generalize the weight accordingly, and construct the Markov transition matrix such that it satisfies the detailed balance with the generalized weight. In some simple cases, including the one for the $S = \frac{1}{2}$ XY model shown in the present section, the resulting algorithm is very close to the loop algorithm discussed in the previous subsection, with the only major difference that loops are created and flipped one by one sequentially, in contrast to the loop algorithm in which all loops are constructed simultaneously. Because of this the algorithm is also referred to as the *directed loop algorithm*.[4] In general, however, there is no direct correspondence between the worm algorithm and the loop algorithm. We look into the details of the worm and directed loop algorithms in Chapter 6. The following simple example is a "sneak preview" without a rigorous derivation of the procedure.

The first step of the algorithm is defining the locations of potential kink positions. But instead of inserting specific graph elements, we insert "vertices." A vertex is

[4] The word *directed* comes from the fact that the worm head has a direction.

similar to a graph element in the loop algorithm, but the type of the graph element on a vertex is not predetermined. It is more like a scatterer at which the worm head changes its course in a nondeterministic way.

Specifically, after removing the vertices from the previous cycle, we place a vertex on every existing kink, and with density $\frac{1}{4}J^{xy}$ place additional vertices between each pair of neighboring sites. Then we put the head and tail of the worm at a randomly chosen point in space-time and randomly choose the initial direction (forward or backward in time) of the head's motion. Suppose the selected direction is "forward." In this case, the head moves forward in time until it hits the next vertex.[5] If the vertex is on a kink, then with probability $\frac{1}{2}$ we assign a graph element (diagonal or horizontal graph) to the vertex. If the vertex is not on a kink, then the graph element is determined by the world-line configuration: If the vertex is located between two parallel spins, the graph element is a diagonal graph; otherwise, it is a horizontal graph. In either case, the worm head scatters at the vertex. In the case of a diagonal graph, the scattering means that the head continues to move forward on the neighboring site, while a horizontal graph element implies that the head reverses direction and moves backward on the neighboring site. Following these rules, the worm head eventually returns to its original position and annihilates with its tail. At this point the world-line configuration has been updated by a single loop, and we place another worm head and tail at some random position and repeat the procedure. This cycle is illustrated in Fig. 5.5.

A noteworthy subtlety is that the assignment of graph elements to the vertices may change during the worm's excursion through space-time and that the worm head may scatter multiple times at the same vertex in different ways. For example, in Fig. 5.5, there is only one vertex that is actually visited by the head, and it is visited twice. At the first encounter, the head is scattered diagonally, whereas the second time, it is scattered horizontally. The graph element assigned to this vertex is not fixed but stochastically determined every time the head visits the vertex. In this sense, the algorithm is not exactly identical to a sequential execution of the loop algorithm. While we used the phrase "assign graph elements to vertices" in the earlier discussion to emphasize the connection to the loop algorithm, "scattering of the head at the vertices" is a more appropriate description of the process.

After N_{cycle} worm update cycles, we remove the vertices and start over, that is, place a new set of vertices, perform N_{cycle} worm updates for this set of vertices, and so on. A pseudocode for this algorithm is given in Algorithm 16.

In the original worm algorithm, as proposed in Prokov'ev et al. (1998), the kink positions are inserted on the fly during the worm's trajectory, using special updates

[5] If the worm is placed on a world line, then the head erases this world line segment; otherwise, it draws a new world line.

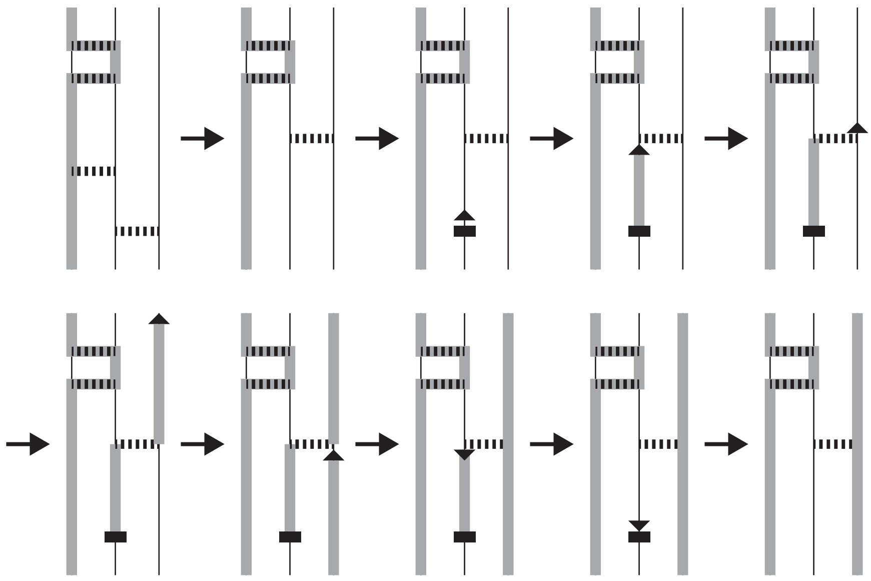

Figure 5.5 Illustration of the directed-loop algorithm with worm update for the spin-$\frac{1}{2}$ XY model.

that insert or remove kinks. Whether or not we separate the vertex generation from the worm motion usually does not make much difference in terms of efficiency. However, the parallelized implementation of the worm algorithm (Masaki et al., 2013) is based on prefixed vertex positions. On the other hand, when the system is very dilute, most of the vertices are never visited by the head, as the head motion is always accompanied by a particle creation/annihilation, which can seriously reduce the efficiency. In such cases, the vertex generation on the fly may be very advantageous because vertices in the deserted area are not even created. In Section 6.4.6, we describe a method that is essentially the same as the original worm algorithm but more similar to the algorithm depicted in Fig. 5.5.

Let us finally note that although we took the same spin-$\frac{1}{2}$ XY model as an example to make the correspondence clearer between the loop and the worm algorithm, introducing worms is not actually necessary for this model, as the loop algorithm works fine in this case. The loop update overcomes the (computational) critical slowing down from which the discrete-time world-line algorithm with local updates suffers. In addition, it is relatively easy to split the computational task of the loop update across many processors simply by splitting the whole space-time domain into small blocks.[6] For other models, however, the worm algorithm can be much

[6] While identifying large loops that do not fit into a single block is technically challenging, a good solution to this problem exists (Todo et al., 2012).

Algorithm 16 Directed-loop algorithm with worm update for the spin-$\frac{1}{2}$ XY model.

```
Input:  A world-line configuration.
  Place vertices on all kinks ;
  for all pairs of neighboring sites do
    Generate new vertex positions $\tau \in [0, \beta)$ with density $\frac{1}{4}J^{xy}$, and place vertices
    there ;
  end for
  for $k = 1$ to $N_{\mathrm{cycle}}$ do
    Pick a point uniform-randomly in the whole space-time domain ;
    Place the head and the tail there ;
    Choose the initial direction of the head with probability $\frac{1}{2}$ ;
    loop
      Move the head to the next vertex ;
      if the head hits the tail then
        Let the head and the tail annihilate and exit the loop ;
      else if the head hits a vertex on a kink then
        Scatter the head diagonally or horizontally with probability $\frac{1}{2}$ ;
      else
        Scatter the head according to the world-line configuration ;
          ▷ There is only one way to satisfy the local conservation law.
      end if
    end loop
  end for
  return  the updated world-line configuration.
```

more efficient. The performance of the loop update is generally very poor when the Hamiltonian contains mutually competing terms. A typical example is the antiferromagnetic Heisenberg model with a uniform external magnetic field, in which the exchange coupling favors an antiparallel alignment of neighboring spins while the external field favors the opposite. If the temperature is low, the loop algorithm experiences a *bottleneck*: The path between antiferromagnetic and ferromagnetic alignments is blocked by a high free-energy barrier, even when both give a non negligible contribution to the thermodynamic distribution. Thus, even though the notorious negative-sign problem does not exist for this example, the loop update has an equilibration problem at low temperature. A fix is possible by exploiting the aforementioned "subtlety" that the worm update does not involve "fixed" graph elements, in contrast to the loop algorithm. Namely, in the worm algorithm, all the matrix elements competing with each other can be taken into account in the scattering probability of the worms. (Hence cancellation of the frustration.) The worm algorithm is more flexible than the loop algorithm when it comes to decomposing the interactions into graph elements.

5.2.7 Ergodicity and winding numbers

Strictly speaking, the world-line method with local updates is not ergodic. It conserves the *winding numbers*. As we have discussed, the loop or worm algorithm replaces local plaquette sampling with a global sampling, a change that allows winding numbers to fluctuate and makes measurements of unequal time correlation functions and simulations in the grand canonical ensemble more accessible. While the efficiency of the algorithm may not depend strongly on the difference between the ensembles, by working in the grand canonical ensemble we obtain direct access to some interesting physical quantities.

If spatial periodic boundary conditions are used (the imaginary-time direction is always periodic), the space-time lattice is on a $(d+1)$-dimensional torus. The winding number is the number of times the path wraps around the torus before it comes back to its starting point. The winding number in the spatial direction is closely related to the superfluid density in Bose systems and the helicity modulus in spin systems (Pollock and Ceperley, 1987). Specifically, we obtain the superfluid density ρ_s from the winding number W_μ in the μ $(= x, y, \ldots)$ direction as

$$\rho_s = \frac{L_\mu^2}{\Omega}\langle W_\mu^2\rangle, \tag{5.26}$$

where $\Omega \equiv L_\tau L_x L_y \cdots$ with $L_\tau \equiv 2t\beta$ in Boson models with hopping constant t. In the XY model, $L_\tau = J^{xy}\beta$ is the dimensionless length in the temporal direction. Likewise, we obtain the dimensionless compressibility $\tilde{\kappa} \equiv 2t\partial\langle n\rangle/\partial\mu$ as

$$\tilde{\kappa} = \frac{L_\tau^2}{\Omega}\langle W_\tau^2\rangle, \tag{5.27}$$

where W_τ is the temporal winding number. These quantities are exponentially small in the normal state (ρ_s) and in the incompressible state ($\tilde{\kappa}$). Therefore, for large lattices, nonzero winding number configurations are highly improbable in such states, so the seemingly disastrous nonergodicity of the local update is actually not a serious problem. Even in superfluid or compressible states, thermodynamic quantities are usually insensitive to the difference between the canonical and the grand canonical ensembles. However, working in the grand canonical ensemble is often advantageous, as ρ_s and κ are often the relevant response functions that characterize the state of the quantum system.

5.3 Bosons and Fermions

Quantum Monte Carlo simulations are not restricted to quantum spin systems at finite temperature. Systems of Bosons and Fermions are of considerable interest, as are the ground-state properties of quantum many-body systems. Not all finite-temperature algorithms are easily formulated in continuous time via loop/cluster or worm algorithms. Even if so, the efficiency of these algorithms is not guaranteed. Most quantum Monte Carlo algorithms, whether for quantum spins, Bosons, or Fermions, are potential victims of a sign problem.

Algorithms for Fermionic systems have characteristic features that distinguish them from quantum spin and Boson algorithms. Some Boson systems, however, are simulated with algorithms that are very similar to those for quantum spins. One-dimensional systems with different quantum statistics map from one to another. In some cases this equivalence might allow algorithmic adoption as opposed to algorithmic creation. In the following subsections, we briefly address these topics.

5.3.1 Bosons

Today, the loop and worm algorithms are the standard methods for simulating quantum spin models not only in one dimension, but also in two and three. They mesh nicely with the short-range exchange interactions typical of these models. Loop and worm algorithms are similarly useful for the simulation of *hard-core Boson* models.[7] The derivations of world-line algorithms for this quantum statistics proceed in a manner analogous to that for quantum spins, with the basis states being the eigenstates of the Boson number operators.

In one dimension, a shortcut exists. Instead of deriving the transition probabilities for a Bosonic world-line algorithm anew, we use the *Matsubara-Matsuda transformation* (Matsubara and Masuda, 1956) to map a hard-core Boson model to a spin-$\frac{1}{2}$ model and then adopt the spin algorithms.

Specifically, the Matsubara-Masuda transformation

$$S_i^+ \leftrightarrow a_i^\dagger, \quad S_i^- \leftrightarrow a_i, \quad S_i^z + \tfrac{1}{2} \leftrightarrow a_i^\dagger a_i = n_i$$

maps a hard-core Boson onto a spin-$\frac{1}{2}$ and vice versa. For example, if we apply this transformation to the longitudinal-field *XXZ* model

$$H = -\sum_{i=1}^{N} \left[J^{xy} \left(S_i^x S_{i+1}^x + S_i^y S_{i+1}^y \right) + J^z S_i^z S_{i+1}^z + H^z S_i^z \right]$$

[7] A hard-core-Boson model is one where only a finite number of Bosons are allowed to occupy a given state.

we obtain

$$H = -t \sum_{i=1}^{N} \left(a_i^\dagger a_{i+1} + a_{i+1}^\dagger a_i \right) - \mu \sum_{i=1}^{N} n_i + V \sum_{i=1}^{N} n_i n_{i+1} + \text{ const.},$$

where $t = \frac{1}{2} J^{xy}$, $V = J^z$, and $\mu = H^z - J^z$. The resulting model is called the extended hard-core *Bose-Hubbard model*. The word *extended* comes from the fact that we have a nearest neighbor coupling, whereas the word *hard-core* indicates that the on-site repulsion is infinite. In Section 6.6, we will discuss the standard Bose-Hubbard model in which we have only an on-site repulsion, which is finite.

We see that the mapping is quite direct. Not surprisingly, techniques used to eliminate potential sign problems from quantum spin algorithms are also useful here. For these two classes of quantum statistics, the sign of the matrix elements of the Hamiltonian is usually dictated by the signs of the model parameters. This distinguishes the construction of spin and Boson algorithms from the challenges of constructing them for Fermions.

5.3.2 Fermions

The derivations of world-line algorithms for Fermions proceed in a manner analogous to that for quantum spins. In exponentiating the pieces of the Hamiltonian, slight differences occur to account for the different quantum statistics. The basis states are typically the eigenstates of the Fermion number operators. It is not that algorithms cannot be constructed, but rather it is difficult to formulate ones that work well.

To better understand the source of the difficulty, we consider a special case in which the *Jordan-Wigner transformation* (Jordan and Wigner, 1928) maps one-dimensional Fermion models onto quantum spin-$\frac{1}{2}$ models. Let us apply this transformation, defined by

$$c_i = K_i S_i^+, \quad c_i^\dagger = S_i^- K_i^\dagger, \quad K_i \equiv e^{i\pi \sum_{j=1}^{i-1} S_j^+ S_j^-},$$

to the one-dimensional, noninteracting, *spinless Fermion model*

$$H = -t \sum_{i=1}^{N} \left(c_i^\dagger c_{i+1} + c_{i+1}^\dagger c_i \right).$$

The operator K_i, called the *kink operator*, counts the number of down-to-up spin flips that appear to the left of i. This automatically accounts for the Fermion sign coming from the permutation of two particles. The kink operator is unitary and self-conjugate,

$$K_i^\dagger = K_i, \quad K_i^2 = I,$$

and commutes with $S_j^\pm$ for $j \geq i$. Using

$$c_i^\dagger c_{i+1} = S_i^+ S_{i+1}^-, \quad c_{i+1}^\dagger c_i = S_{i+1}^+ S_i^-, \quad c_i^\dagger c_i = S_i^+ S_i^-, \tag{5.28}$$

we find

$$H = -2t \sum_{i=1}^{N} \left(S_i^x S_{i+1}^x + S_i^y S_{i+1}^y \right),$$

the XY model with $J^{xy} = 2t$. Hence, we can simulate spinless Fermions by using the algorithms discussed for this spin model.[8] When we add Fermion spin into the problem, we have to introduce world lines for the up-spins and for the down-spins. The simulation then requires an additional Monte Carlo step to accept or reject world-line movements on the basis of the interactions between up- and down-spins (Kawashima et al., 1994).

Let us emphasize the kink operator in the mapping between the spin raising and lowering operators and the creation and destruction operators. A similar operator did not exist in the Matsubara-Matsuda transformation. We recall that our basis states are direct product states listing the lattice sites in a specific order. As long as the lattice sites are sequentially numbered, the Jordan-Wigner transformation applies to any dimension. What makes it difficult to generalize the above one-dimensional solution to higher dimensions is that it is impossible to number the sites in such a way that all interacting pairs appear next to each other in the list. (If it is possible, it means that the lattice is one-dimensional.) To be more specific, the general form of (5.28) is

$$c_i^\dagger c_j = e^{i\pi \sum_{k=i+1}^{j-1} S_k^+ S_k^-} S_i^+ S_j^-,$$

with an ugly phase factor that does not vanish if $|j - i| > 1$ and makes the problem unsolvable in general. This sign problem for all practical purposes restricts the use of the world-line method to one-dimensional systems.

In short, the source of the difficulty in simulating Fermion problems is the sign arising from the permutation among Fermions that is represented by the phase factor $e^{i\pi \sum_{k=i+1}^{j-1} n_k} = \pm 1$. If we forget about this phase factor for the time being, the Fermion Hamiltonian is identical to that of hard-core Bosons or $S = \frac{1}{2}$ spin models. We can therefore apply the loop/cluster (or any other) algorithms for spins or Bosons to Fermions as well. However, to take into account the phase

[8] Historically, the inverse transformation

$$S_i^+ = c_i K_i, \quad S_i^- = c_i^\dagger K_i,$$

mapping the XY model to the easily solvable spinless Fermion model, was used to solve the former exactly.

factor in a way that does not lead to severe cancellations between different Monte Carlo configurations is very difficult, and it is called "Fermion sign problem" or "negative-sign problem." There is no general solution to this problem to this day.

5.4 Negative-sign problem

The negative-sign problem is the biggest bottleneck in numerical simulations of quantum models. For the algorithms considered here, it originates from the negative matrix elements of the Hamiltonian. When some of the off-diagonal matrix elements of the local Hamiltonian H are negative, the weight $w(C)$ generally is negative for some of the configurations C. In such a case, we perform a Monte Carlo simulation for which the target distribution is $|w(C)|$, not $w(C)$. Then, we compute the expectation value of an observable Q using the identity

$$\begin{aligned}\langle Q\rangle &= \frac{\sum_C |w(C)|\,\mathrm{sign}(C)\; Q(C)}{\sum_C |w(C)|\,\mathrm{sign}(C)} \\ &= \frac{\sum_C |w(C)|\,\mathrm{sign}(C)\; Q(C)/\sum_C |w(C)|}{\sum_C |w(C)|\,\mathrm{sign}(C)/\sum_C |w(C)|} \\ &= \frac{\langle \mathrm{sign}(C)\; Q(C)\rangle_{\mathrm{MC}}}{\langle \mathrm{sign}(C)\rangle_{\mathrm{MC}}}, \end{aligned} \tag{5.29}$$

where $\mathrm{sign}(C) = \pm 1$ denotes the sign of $w(C)$ and $\langle\cdots\rangle_{\mathrm{MC}}$ is the Monte Carlo average with the weight $|w(C)|$.

The negative contribution Z_- to the partition function cancels part of the positive contribution Z_+, while the total $Z = Z_+ - Z_-$ must always be positive. In fact, in many cases, the negative contribution *almost completely* cancels the positive one. We can understand this (Loh et al., 1990; Hatano and Suzuki, 1992) by considering a fictitious Hamiltonian H', whose matrix elements are the absolute value of the corresponding matrix elements of the original Hamiltonian H: $\langle C'|H'|C\rangle \equiv |\langle C'|H|C\rangle|$. The difference in the free energy per site $\Delta f \equiv (F' - F)/N$ is of $\mathcal{O}(1)$ when the difference between the two Hamiltonians is extensive. Because $\beta F = \ln Z$ and $\beta F' = \ln Z'$, where Z' is the partition function of the fictitious system, it follows that the denominator of (5.29) can be expressed as

$$\frac{Z_+ - Z_-}{Z_+ + Z_-} = \frac{Z}{Z'} = e^{-\beta N \Delta f}.$$

It is clear from this expression that the positive and the negative parts of Z cancel almost completely at low temperature and in systems of large size, and it becomes practically impossible to calculate the expectation value $\langle Q\rangle$ from (5.29), because in both the numerator and denominator, the statistical error will be much larger than the mean.

5.5 Dynamics

Equilibrium quantum Monte Carlo results possess physically meaningful dynamical information, while classical Monte Carlo results do not. The access to *real* dynamical information is the reward for having to deal with the extra imaginary-time dimension in the simulation. To access this information, specific types of measurements must be made. By definition, the expectation value of a quantum mechanical operator X is

$$\langle X \rangle = \frac{\mathrm{Tr}\, e^{-\beta H} X}{\mathrm{Tr}\, e^{-\beta H}}.$$

The operator at imaginary time τ being $X(\tau) = e^{\tau H} X e^{-\tau H}$ we find

$$\langle X(\tau) \rangle = \frac{\mathrm{Tr}\, e^{-(\beta-\tau)H} X e^{-\tau H}}{\mathrm{Tr}\, e^{-\beta H}}.$$

As a result of the time-translation invariance of equilibrium states, $\langle X \rangle = \frac{1}{\beta}\int_0^\beta d\tau \langle X(\tau)\rangle$. This class of expectation values is called *equal-time expectation values*.

Dynamical probes involve the product of two (or more) operators at different times

$$\langle A(\tau) B(0) \rangle = \frac{\mathrm{Tr}\, e^{-(\beta-\tau)H} A e^{-\tau H} B}{\mathrm{Tr}\, e^{-\beta H}}.$$

These expectation values also enjoy time-translation invariance: $\langle A(\tau)B(0)\rangle = \langle A(\tau+\tau')B(\tau')\rangle$.

The importance of unequal-time measurements is their relation to spectral densities,

$$\langle A(\tau) B(0) \rangle = \int_{-\infty}^{\infty} d\omega \frac{e^{-\tau\omega} D(\omega)}{1 \pm e^{-\beta\omega}},$$

where the $\pm$ depends on whether A and B anticommute or commute. The spectral density $D(\omega)$ connects imaginary-time dynamics with real dynamical information. More specifically, the spectral density contains measurable information about the elementary excitations in the system. For example, measuring the imaginary-time dependence of the spatial Fourier transformation of $\langle S_i^z S_j^z \rangle$ and inverting the integral equation yields $D(k,\omega)$ (Kawashima et al., 1996), which is often measured in inelastic neutron scattering experiments. Peaks in this function are indicators of elementary z-component spin excitations, and the locations of the peaks as a function of k yield the excitation's dispersion relation $\omega(k)$. Inverting the integral equation is a difficult task even with data that lack the statistical errors associated with a Monte Carlo estimate. We discuss techniques for this inversion in Chapter 12. More accessible are static susceptibilities, found by integrating over imaginary time. For example, the *static longitudinal and transverse spin susceptibilities* are

$$\chi_{ij}^{\parallel} = \int_0^{\beta} d\tau \left\langle S_i^z(\tau) S_j^z(0) \right\rangle,$$

$$\chi_{ij}^{\perp} = \int_0^{\beta} d\tau \left\langle S_i^x(\tau) S_j^y(0) + S_i^y(\tau) S_j^x(0) \right\rangle.$$

For the measurement of diagonal correlation functions, we use the identity

$$\langle S_i^z(\tau) S_j^z(0) \rangle \equiv \langle s_i(\tau) s_j(0) \rangle_{\mathrm{MC}},$$

where $\langle \cdots \rangle_{\mathrm{MC}}$ denotes the simple average over the world-line configurations generated by the Monte Carlo simulation. Although the measurement of off-diagonal correlation functions is not as straightforward as this, when we use the loop/cluster update or the worm update, the formula is almost as simple. For example, we can estimate the correlation function between S^x operators in the spin-$\frac{1}{2}$ XY model using

$$\langle S_i^x(\tau) S_j^x(0) \rangle \equiv \frac{1}{4} \langle \theta((i,\tau),(j,0)) \rangle_{\mathrm{MC}},$$

where $\theta(X,Y)$ is 1 if the space-time points X and Y are on the same loop, and 0 otherwise. By integrating over X and Y, we obtain the uniform, static transverse susceptibility,

$$\begin{aligned}
\chi^{\perp} &= \frac{1}{N\beta} \sum_{i,j} \int_0^{\beta} d\tau_1 d\tau_2 \langle \mathcal{T} S_i^x(\tau_2) S_i^x(\tau_1) \rangle \\
&= \frac{1}{4} \frac{1}{N\beta} \sum_{i,j} \int_0^{\beta} d\tau_1 d\tau_2 \, \langle \theta((i,\tau_2),(j,\tau_1)) \rangle_{\mathrm{MC}} \\
&= \frac{1}{4} \frac{1}{N\beta} \left\langle \sum_l \Lambda_l^2 \right\rangle_{\mathrm{MC}},
\end{aligned}$$

where $\mathcal{T}$ denotes the time-ordered product and Λ_l is the length of the loop specified by the loop index l. Similarly, we estimate the correlation function in the worm algorithm by simply measuring how frequently the head visits a specific position relative to the tail. The simple and efficient measurement of off-diagonal correlation functions is an important advantage of the loop/cluster and worm algorithms. We discuss these points in more detail in Chapter 6.

Suggested reading

M. Creutz and B. Freedman, "A statistical approach to quantum mechanics," *Ann. Phys.* **132**, 427 (1981).

R. T. Scalettar, "World-line quantum Monte Carlo," in *Quantum Monte Carlo Methods in Physics and Chemistry*, series C, volume 525, NATO Science Series, ed. M. P. Nightingale and C. J. Umrigar (Dordrecht: Kluwer Academic, 1999).

H. G. Evertz, "The loop algorithm," *Adv. Phys.* **52**, 1 (2003).

N. Prokof'ev and B. Svistunov, "Worm algorithm for problems in quantum and classical systems," in *Understanding Phase Transitions*, ed. L. D. Carr (Boca Raton, FL: Taylor and Francis, 2010).

A. W. Sandvik, "Computational studies of quantum spin systems," in *Lectures on the Physics of Strongly Correlated Systems XIV*, ed. A. Avella and F. Mancini, AIP Conference Proceedings, **1297**, 135 (2010).

Exercises

5.1 Conservation Rule. Prove that if a quantum spin Hamiltonian is invariant under the $U(1)$ rotation,

$$S_i^+ \to e^{i\phi} S_i^+,\ S_i^- \to e^{-i\phi} S_i^- \text{ (for all } i \in V),$$

the Hamiltonian has zero matrix elements between the eigenstates of $\sum_{i\in V} S_i^z$ with different eigenvalues, where V is the area on which the Hamiltonian is defined.

5.2 The most familiar example of a Poisson process is the decay of a radioactive atom with decay rate λ. For this decay,

1. Show that the probability of the atom surviving longer than some time τ is

$$P_{\text{survive}}(\tau) = (1 - \lambda\Delta\tau)^{\tau/\Delta\tau} \to e^{-\lambda\tau} \quad \text{as } \Delta\tau \to 0.$$

2. Show that the probability of the first decay event occurring in the short-time interval $[\tau, \tau + \Delta\tau)$ is

$$P_{\text{survive}}(\tau)\lambda\Delta\tau = \lambda\Delta\tau e^{-\lambda\tau}.$$

3. Obtain the same result by using the product of two Poisson distributions: the first to calculate the probability of no events in the interval $[0, \tau)$ and the second to calculate the probability of one event in the short-time interval $[\tau, \tau + \Delta\tau)$.

5.3 Suppose we have two independent Poisson processes over a time interval τ that are characterized by (N_1, λ_1) and (N_2, λ_2) for their number of events and density of events. Let $N = N_1 + N_2$ and $\lambda = \lambda_1 + \lambda_2$.

1. Show that $P(N = 0) = e^{-\lambda\tau}$ and $P(N = 1) = \lambda\tau e^{-\lambda\tau}$.

2. Generalize the above to show that two independent Poisson processes have a joint distribution of density λ.
3. Show that conditional on N, the distribution of N_1 is a binomial distribution (2.34) of index N and parameter $p = \lambda_1/(\lambda_1 + \lambda_2)$.

5.4 Suppose x has a Poisson distribution parameterized by a density λ, and for a given x, y has a binomial distribution of index k and parameter p (Exercise 2.2). Show that the unconditional distribution of y is a Poisson distribution with density $p\lambda = E[E[y|x]]$.

5.5 Verify (5.12) and (5.13).

5.6 Show that

$$e^A C e^{-A} = C + [A, C] + \frac{1}{2!}[A, [A, C]] + \frac{1}{3!}[A\,[A\,[A, C]]] + \cdots \tag{5.30}$$

5.7 Relate the steps in Algorithm 15 with the panels in Fig. 5.4, keeping in mind some graph assignments are done probabilistically. In the middle panel of Fig. 5.4, identify the loops. There are four. Which were flipped to obtain the right figure?

5.8 What are the matrices representing $\exp(-\tau S^z)$ and $\exp(-\tau S^x)$?

5.9 To be canonical, the Matsubara-Matsuda and Jordan-Wigner transformations must conserve the operator commutation relations. Verify this for both transformations and their inverse. Using the two transformations, establish the one that transforms directly between Fermions and Bosons.