Fermion simulations in systems with negative weights

Steven R. White and John W. Wilkins
Laboratory of Atomic and Solid State Physics, Clark Hall, Cornell University, Ithaca, New York 14853-2501
(Received 19 October 1987)

We investigate the use of two methods—Langevin and molecular dynamics—for simulating fermion systems with real, nonpositive definite probability functions. The methods are tested on a simple one-dimensional model. Both methods have difficulties associated with the nodes of the probability function. While the Langevin equation can cross the nodes even for time steps as small as 10^{-10}, the crossing may be so violent that an equilibrium distribution is not readily achieved. The molecular-dynamics system is trapped within regions bounded by the nodes, not sampling all of phase space. A hybrid Monte Carlo–molecular-dynamics method is proposed which avoids these difficulties in the Langevin and molecular-dynamics techniques.

I. INTRODUCTION

Recent interest in the Langevin and molecular-dynamics techniques for simulating systems of strongly interacting fermions on a lattice has stemmed from the potential for much faster simulations than achieved with Monte Carlo techniques. The successes[1] with the Monte Carlo techniques have been limited to small- or medium-sized lattices, because of computation times which increased rapidly with the number of sites N. The new methods allow calculation times which are nearly proportional to N.

A serious problem with these simulations is that in most fermion systems the probability function is not always positive.[2] In this paper we consider this problem by testing the methods on a very simple model. We find that the negative probability poses real problems for either Langevin or molecular-dynamics simulations, although for different reasons in each case. We suggest a "hybrid" algorithm incorporating elements of the Monte Carlo, molecular dynamics, and Langevin techniques which work on our simple model, but, for realistic problems, may be no faster than pure Monte Carlo.

In Sec. II we describe the form of typical fermion effective actions and present a simple model with many of the same features to use in test simulations. In Sec. III we describe the Langevin method and apply it to our test model, and in Sec. IV we do the same for molecular-dynamics techniques (including the so-called hybrid molecular-dynamics–Langevin method[3]). In both cases we discuss the difficulties that are implied for fermion simulations by the problems encountered in our test calculations. A hybrid technique for dealing with these problems is discussed in Sec. V, and we summarize and conclude in Sec. VI.

II. THE MODEL

The partition function for a system of interacting fermions in the commonly used determinantal formulation can be written in the general form[4,5]

$$Z=\int_{-\infty}^{\infty} d[x_i]e^{-S_B(x)}\det M_+(x)\det M_-(x)\ . \tag{1}$$

The auxiliary variables $\{x\}$ are introduced in a Hubbard-Stratonovich transformation, and the determinants come from a trace over the fermions. To the best of our knowledge, there are no theorems that $\det M(x)$ is positive for all values of $\{x\}$. In fact, in the Hubbard model, for example, numerical evidence[6] has shown that the determinant does change sign. In some cases, such as the half-filled Hubbard model, $\det M_+(x)$ and $\det M_-(x)$ can be shown to always have the same sign,[6] and the integrand of (1) is always non-negative. But for the Hubbard model away from half-filling, and for most systems in general, the integrand can change sign.[7] (Even if the two determinants always have the same sign, the probability function can become zero, resulting in severe difficulties for simulations; see the end of this section.)

Operator expectation values have the form

$$\bar{G}=\frac{1}{Z}\int_{-\infty}^{\infty} d[x_i]e^{-S_{\rm eff}}G(x)\ , \tag{2}$$

where we have used the shorthand notation

$$e^{-S_{\rm eff}}\equiv P(x)\equiv e^{-S_B(x)}\det M_+(x)\det M_-(x)\ . \tag{3}$$

Monte Carlo methods rely on treating $Z^{-1}e^{-S_{\rm eff}}$ as a *probability*. In cases where the right-hand side of (3) is not positive definite, we set $e^{-S_{\rm eff}}=|P(x)|$, and define the modified partition function

$$\tilde{Z}=\int_{-\infty}^{\infty} d[x_i]\,|P(x)|\ . \tag{4}$$

We now calculate operator expectation values by rewriting (2) as

$$\bar{G}=\frac{\langle \mathrm{sgn}P(x)G(x)\rangle_{\tilde{Z}}}{\langle \mathrm{sgn}P(x)\rangle_{\tilde{Z}}}\ , \tag{5}$$

where $\langle\ \rangle_{\tilde{Z}}$ denotes a Monte Carlo average using $|P(x)|$ rather than $P(x)$. This way of calculating $\bar{G}$, while exact, is only feasible when $\langle s\rangle\equiv\langle \mathrm{sgn}P(x)\rangle_{\tilde{Z}}$ is not close to zero. If $\langle s\rangle$ is near zero, $\bar{G}$ will have huge statistical er-

rors. In this paper we consider the case where $\langle s \rangle$ is neither unity nor close to zero.

To test simulation techniques in as simple a model as possible with the same features as (3), we chose the single-variable effective action defined by

$$P(x) \equiv e^{-x^2}(1-3x^2+x^4) . \tag{6}$$

Figure 1 shows $P(x)$ and $S_{\rm eff}(x)=x^2-\ln|1-3x^2+x^4|$. Logarithmic singularities in $S_{\rm eff}$ correspond to nodes in $P(x)$. We chose e^{-x^2} because $S_B(x)$ is usually a quadratic form; we chose $1-3x^2+x^4$ as a simple polynomial which has positive and negative regions, but is mostly positive. For this model, $\langle s \rangle = 0.3329$ (doing the integrals analytically), which is close enough to zero to make simulation of the model significantly more difficult than the positive definite case.

To model a case where the determinants in (1) have the same sign, we set

$$P^{(2)}(x)=e^{-x^2}(1-3x^2+x^4)^2 . \tag{7}$$

In this case we have $S^{(2)}_{\rm eff}(x)=x^2-2\ln|1-3x^2+x^4|$, which only differs from $S_{\rm eff}(x)$ by a factor of 2 in front of the logarithm. Since the Langevin and molecular-dynamics equations of motion (see below) depend only on $S_{\rm eff}(x)$, not $P(x)$ directly, one might expect the qualitative behavior of the two models to be identical. There turn out to be important differences in their behavior in the Langevin equation, however, as we discuss in the next section.

III. THE LANGEVIN METHOD

In the Langevin method,[8,9] one solves a first-order equation of motion

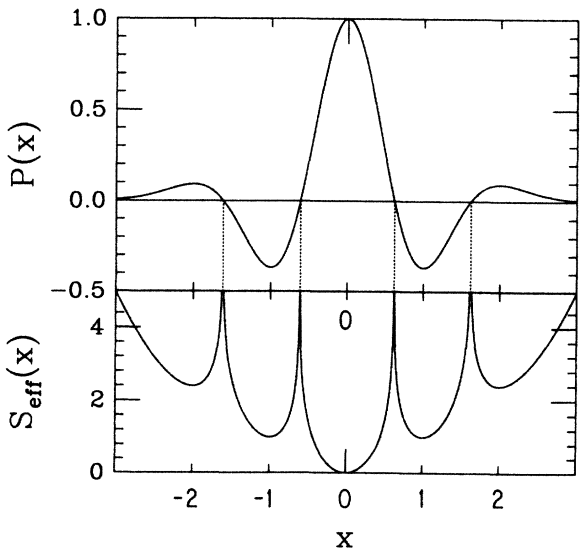

FIG. 1. Probability function $P(x)$ and effective action $S_{\rm eff}$ for the simple test model. The model is defined by (6). The upper figure shows $P(x)$; the lower shows the effective action corresponding to $|P(x)|$, $S_{\rm eff}(x)=-\ln|P(x)|$. Logarithmic singularities in $S_{\rm eff}$ correspond to nodes in $P(x)$ at $x=\pm\frac{1}{2}(\sqrt{5}\pm 1)$.

$$\frac{dx}{dt}=-\frac{dS_{\rm eff}}{dx}+\eta(t) , \tag{8}$$

for a particle subject to the drift term $-dS_{\rm eff}/dx$ and a white noise $\eta(t)$ satisfying

$$\langle \eta(t) \rangle = 0; \quad \langle \eta(t)\eta(t') \rangle = 2\delta(t-t') . \tag{9}$$

For well behaved $S_{\rm eff}$ and for large t the probability of $x(t)$ taking on the particular value $\bar{x}$ is proportional to $P(\bar{x})$; hence, expectation values can be calculated using

$$\langle G \rangle = \lim_{T\to\infty} \frac{1}{T}\int_0^T dt\, G(x(t)) . \tag{10}$$

The noise term in (8) serves to drive the system throughout phase space, while the drift term $-dS_{\rm eff}/dx$ makes the system spend most of the time where $S_{\rm eff}$ is smallest. Note that $-dS_{\rm eff}/dx=(1/P)(dP/dx)$ is independent of the sign of $P(x)$; hence, one does not need to know $\mathrm{sgn}P(x)$ to calculate the drift term. The drift term always drives the system in the direction of increasing $|P(x)|$. In cases where $P(x)$ changes sign, one might hope that averages of the form (10) could be used to compute averages of the form $\langle \ \rangle_{\tilde{Z}}$. As we shall see below, there are two problems with this.

To simulate the Langevin equation, we must discretize the time in the continuous equation (8). A simple discretization is[10]

$$x_{t+\Delta t}=x_t-\Delta t\left.\frac{dS_{\rm eff}}{dx}\right|_{x_t}+\sqrt{\Delta t}\,\eta_t , \tag{11}$$

where η_t is a random number obeying a Gaussian distribution with $\langle \eta_t\eta_{t'} \rangle = 2\delta_{tt'}$. The errors introduced by discretization in (11) are[10] $O(\Delta t)$. A discretization accurate to $O(\Delta t^2)$ is given by[10-12]

$$x_{t+\Delta t}=x_t-\frac{\Delta t}{2}\left[\left.\frac{dS_{\rm eff}}{dx}\right|_{x_t}+\left.\frac{dS_{\rm eff}}{dx}\right|_{\tilde{x}}\right]+\sqrt{\Delta t}\,\eta_t , \tag{12}$$

where $\tilde{x}$ is a tentative step using the first-order discretization (and the same random number η_t)

$$\tilde{x}=x_t-\Delta t\left.\frac{dS_{\rm eff}}{dx}\right|_{x_t}+\sqrt{\Delta t}\,\eta_t . \tag{13}$$

The test results we present here are based on the second-order discretization, but we have found the first-order equation to give the same qualitative behavior.

A. Node crossing

It is useful to discuss the Langevin dynamics in terms of a particle undergoing Brownian motion in the potential $S_{\rm eff}$. In that language, we ask the following question: Can the particle cross the infinite singularities in $S_{\rm eff}$ corresponding to the nodes of $P(x)$? The (perhaps) surprising answer is *yes*, it readily does so. Figure 2 shows a typical trajectory. While the particle avoids the regions near the nodes, it does cross them occasionally. The discrete Langevin equation (11) shows how this can happen: when observed on a small enough time scale, the noise term always dominates any finite drift force. The

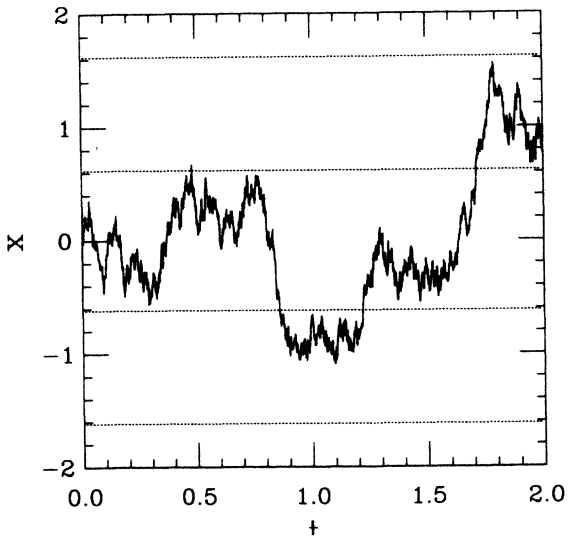

FIG. 2. Typical trajectory in a Langevin simulation for the test model. The dashed lines show the location of the nodes of $P(x)$. The "particle" avoids the nodes, but occasionally crosses them. The crossings take place regardless of the time step, which in this case is 10^{-3}.

noise term can cause the particle to hop over the singularity.

Are the crossings simply the result of too large a time step? To test whether reducing the time step eliminated crossings, we performed a series of runs with decreasing Δt. In Fig. 3 we compare the average number of times

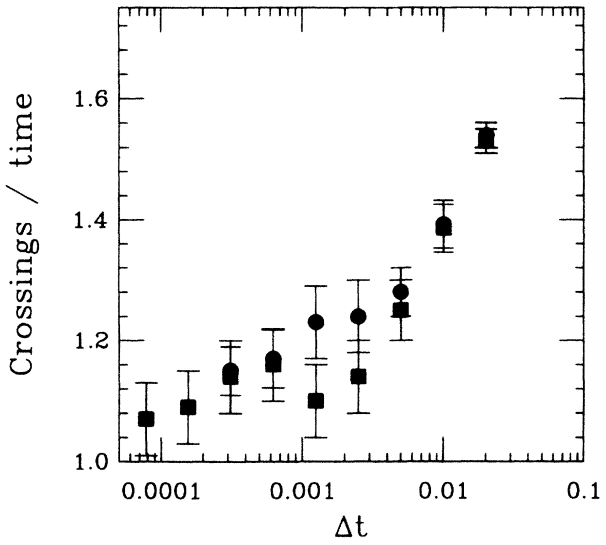

FIG. 3. Rate of nodal crossings for various time steps for the model $P(x)$. The solid circles are for the unmodified Langevin algorithm, which occasionally undergoes "jumps" caused by the singularities in $S_{\text{eff}}(x)$ (see Fig. 4). The solid squares are for a modified algorithm which limits the size of the jumps. The rate of crossings appears to tend to a finite constant as $\Delta t \to 0$ for both algorithms. The calculations used the second-order discretization of the Langevin equation (12). The algorithm was unstable for $\Delta t = 4 \times 10^{-2}$, a factor of 2 larger than the largest Δt shown. The errors (twice the standard deviation is shown) were estimated using the square root of the number of crossings observed.

nodes are crossed per unit of Langevin time for different Δt. As Δt is decreased, initially we see a reduction in the number of crossings, but the crossing *rate* (the average number of crossings per unit Langevin time) appears to approach a finite limit. The figure shows time steps varying by a factor of 32, which, if S_{eff} were smooth, would imply a reduction in the error by a factor of 32^2. In calculations with the initial value of x close to one of the nodes (so the computations could be carried out in a reasonable amount of time) we have observed crossings with $\Delta t = 10^{-10}$.

We can understand the constant rate of crossing as $\Delta t \to 0$ as follows: assume that the probability of the particle being at x is proportional to $|P(x)|$. For simplicity we will consider the first-order discretization (11). In considering a single step, to first order the drift term, which varies as Δt, is negligible compared to the noise term, which varies as $\sqrt{\Delta t}$. Given that there is a node at $\bar{x}$, if at a particular Langevin step the particle is at $x = \bar{x} + \delta x$ (assume $\delta x > 0$) then the probability that the particle will cross $\bar{x}$ is the probability that $-\eta_t \sqrt{\Delta t}$ is greater than δx. With η_t obeying a Gaussian distribution of width $\sqrt{2}$, this probability is roughly proportional to $\exp[-(\delta x)^2/(4\Delta t)]$. Hence, the total probability of crossing at that Langevin time step is proportional to the integral over all x greater than $\bar{x}$ of the product of the probability of being at x and the probability that $-\eta_t \sqrt{\Delta t}$ is greater than $x - \bar{x}$:

$$\int_{\bar{x}}^{\infty} dx \, |P(x)| \, e^{-(x-\bar{x})^2/(4\Delta t)} . \tag{14}$$

In the vicinity of the node $|P(x)| \propto \delta x$, thus (14) is proportional to Δt. Since the number of time steps per unit Langevin time is $1/(\Delta t)$, the crossing *rate* tends to a constant as $\Delta t \to 0$.

What happens when the probability function is $P^{(2)}(x)$, which has second-order zeroes in the place of nodes? In that case, for small δx, $P^{(2)}(x) \propto (\delta x)^2$, making (14) proportional to $(\Delta t)^{3/2}$, and the rate of crossing proportional to $(\Delta t)^{1/2}$. Figure 4 shows the crossing rate as a function of $(\Delta t)^{1/2}$ for $P^{(2)}(x)$. We see that the crossing rate roughly behaves as expected.

B. Large displacements

Figure 5 shows a severe difficulty with the Langevin calculations. Occasionally as the particle is crossing a node, the particle will land almost exactly on the singularity. The particle then experiences a very large drift force, driving it in a single time step far from the origin (e.g., at $t \sim 1.3$, $x \sim -0.6$). In a calculation similar to the one shown in Fig. 5, we observed a single jump of length $x_{t+\Delta t} - x_t \approx 1500$. Jumps of more moderate size, such as the ones shown in Fig. 5, were fairly common. The presence of these jumps makes this version of the Langevin method unusable for practical calculations, since the probability distribution of $x(t)$ is no longer $|P(x)|$. For example, the probability of the particle being at $x = 1500$ is roughly $e^{-1500^2} \sim 10^{-10^6}$.

To obtain more accurate data on the frequency of crossings as Δt varies, we modified the discrete Langevin equation to put an arbitrary limit on how far the particle

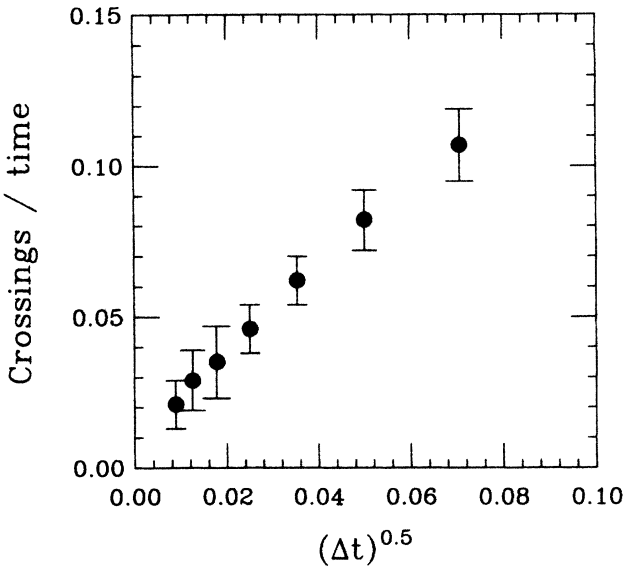

FIG. 4. Rate of nodal crossings for the model $P^{(2)}(x)$ as a function of the square root of the time step Δt. The rate is expected to vary as $(\Delta t)^{0.5}$ for small Δt. The discretization and error estimates were the same as in Fig. 3.

could move in one step. Since for small Δt the noise term in (12) should be larger than the drift term, we imposed a limit of $10\sqrt{2\Delta t}$, i.e., 10 times the average noise term, on the magnitude of the drift term. In other words, if the drift term was larger than this limit it was replaced by the limit. This choice of the limit restricts only large jumps; it serves to prevent the particle from being driven

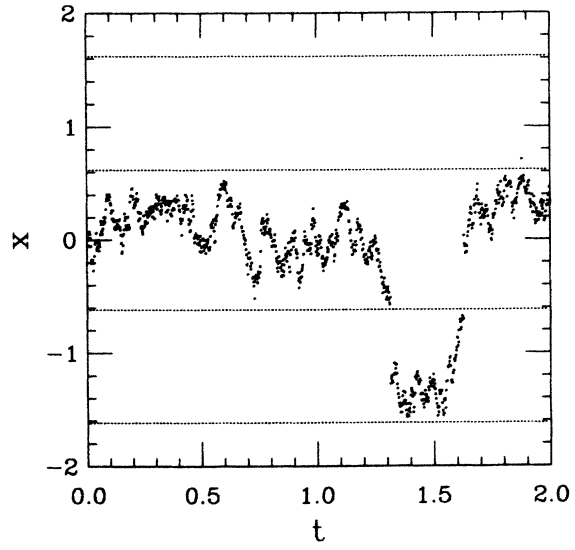

FIG. 5. Langevin trajectory showing a typical large "jump" caused by the singularities in S_{eff}. The individual dots show the position of the "particle" at each time step. The jumps occur as the particle, while crossing a singularity, lands almost exactly on it. In the next time step (or half-time step in the second-order discretization shown here) the particle travels a large distance because of the divergence of the force at the singularity. The jumps can be quite large and are a severe difficulty of the Langevin method. The time step was 2×10^{-3}.

away from the region near the origin. Figure 3 shows the rate of crossings for the modified algorithm as well as the unmodified. The modified algorithm, while decreasing the rate slightly, also appears to yield a finite rate of crossings as $\Delta t \to 0$.

The Langevin method in real fermion simulations has an additional difficulty: it cannot readily tell when a node is crossed. One needs to know when nodes have been crossed because $\text{sgn}P(x)$ is needed at all times in order to calculate observables using (10) and (5). Of course, in our one-dimensional model, where the location of the nodes is known, it is easy to tell when the particle has crossed; in addition, one could always evaluate $P(x)$ at each step to find its sign. In real fermion simulations, however, one does not know the location of the nodes of $P(x)$, and explicit evaluation of the determinants in (1) is extremely time consuming (more so than evaluating the drift term, which can be put in an easily evaluated form).[13,10] Until a fast way of determining $\text{sgn}P(x)$ is found, use of the Langevin equation method for systems with negative weights is probably feasible only for small systems.

IV. THE MOLECULAR-DYNAMICS METHOD

In the molecular-dynamics method, one solves the classical equation of motion

$$\frac{d^2x}{dt^2} = -\frac{dS_{\text{eff}}}{dx} \tag{15}$$

for a particle in the potential S_{eff}. The mass m is set to unity, since a change of mass simply renormalizes t. The many-variable version of (15) can be used to calculate averages weighted by $e^{-\beta S_{\text{eff}}}$ under certain conditions: (1) the system is in the thermodynamic limit (i.e., an infinite number of variables x), so that the microcanonical ensemble corresponds to the canonical ensemble; and (2) the system is ergodic. The effective temperature β depends on the initial conditions. Therefore, for properly chosen initial conditions, and if conditions (1) and (2) are both satisfied, then a solution $x(t)$ of (15) can be used to calculate averages via (10), just as can be done with Langevin trajectories. Of course, the simple model (6) is nowhere near the thermodynamic limit. However, experience has shown that, even in simulations with large numbers of fermions, lack of ergodicity remains.[3]

A simple way to (a) make molecular dynamics ergodic for *positive* $P(x)$ and (b) remove the necessity for being in the thermodynamic limit is to randomize the velocity at random times, taking the new velocity from a Boltzmann distribution (at temperature $\beta=1$, so that the distribution is $e^{-S_{\text{eff}}}$). This simple idea, long used[14] in molecular-dynamics simulations of chemical systems, has recently been applied to fermion simulations in the form of hybrid Langevin–molecular-dynamics algorithms.[3]

A particle obeying the classical equations of motion (15) should not be able to cross the singularities in S_{eff}, even with velocity randomization. In a discretized form, however, the particle might cross because of too large a step size. Although this might prevent the particle from being trapped in a finite interval, it would also subject the system to the same spurious jumps seen in the Langevin

system. Unfortunately, since the logarithmic singularities in S_{eff} are very sharp, a very small time step is necessary to integrate (15) correctly near them. Since the particle will sometimes come quite close to the singularities, a constant time step is not practical: a step size suitable for the rare cases when the particle approaches the singularity would be much too small for the majority of the time when the particle was in the smooth regions. We have found that to integrate (15) accurately in a reasonable amount of time, a variable time-step integrator is needed.

Figure 6 shows a typical molecular-dynamics trajectory (with velocity randomization) using a Bulirsch-Stoer variable time-step integrator.[15] The discontinuities in the slope of the trajectory correspond to velocity randomizations, which occur at random intervals. In a few instances (e.g., at $t \approx 12$, $x \approx -0.6$) the particle comes quite close to a node and the change in direction is so sharp as to resemble a velocity randomization, but this is actually the particle bouncing off the "wall" of the singularity. It is in these instances that the variable-step integrator is crucial.

This procedure, while fairly practical for sampling one of the nodally-bounded regions, provides no way to sample other regions. In the next section we discuss a procedure for adding discrete Monte Carlo steps to correct this deficiency.

V. HYBRID MOLECULAR-DYNAMICS–MONTE CARLO METHOD

For a fermion simulation algorithm to be successful in problems with negative weights, it must have (at least) three characteristics: (1) It must be able to avoid numerical difficulties at the nodes of $P(x)$ such as the "jumps" seen in the Langevin simulations. (2) In order to explore all of phase space, it must be able to cross the nodes. (3) The method should either (a) keep track of each node crossing so $\text{sgn}P(x)$ is known at all times, or (b) provide a method for efficiently calculating $\text{sgn}P(x)$. Monte Carlo algorithms have the features (1), (2), and (3a), but have the disadvantage of calculation times which rise rapidly with the number of sites N. In this section we propose an algorithm which combines molecular dynamics and Monte Carlo in a way that satisfies the above criteria [(1), (2), and (3a)] for a successful algorithm.

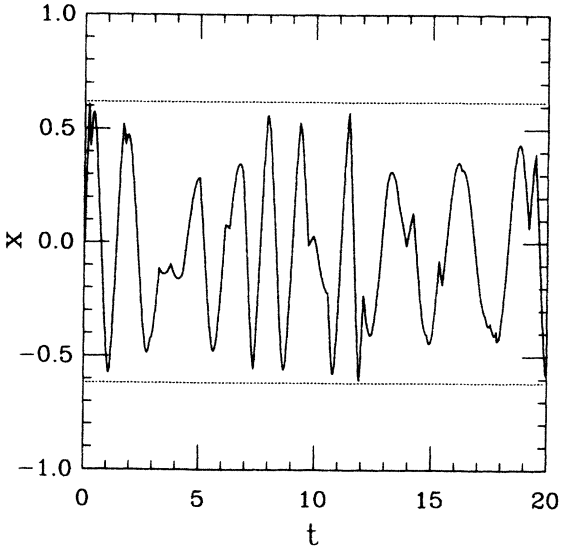

FIG. 6. Typical molecular dynamics with velocity randomization trajectory. The particle is trapped within a single region bounded by the nodes of $P(x)$ (the dashed lines). Because of the sharpness of the singularities in S_{eff}, a variable time-step integrator is used to accurately solve the equations of motion. The run shown consists of 1000 steps, each consisting of an integration of the equations of motions for a length of time $\epsilon = 0.02$. After each step the there was a probability $p = 0.05$ of the velocity being randomized.

The idea is simple: most of the time is spent solving the molecular-dynamics equations of motion, but every once in a while (at random times) a Monte Carlo step is attempted. After the Monte Carlo step, the velocity is randomized and molecular dynamics resumes. A variable time-step integrator is used to avoid numerical difficulties at the nodes. A Monte Carlo step explicitly calculates $P(x+\delta x)/P(x)$, enabling the algorithm to keep track of $\text{sgn}P(x)$.

The specific details of the algorithm are as follows: a random velocity v is chosen from the distribution $e^{-(1/2)v^2}$, and the equations of motion are integrated [with the integrator used in the previous section, and from a position x with known $\text{sgn}P(x)$] for a fixed length of time ϵ. At the end of that time, a Monte Carlo step is taken with probability p, and the integration is continued with probability $1-p$. To do the Monte Carlo step, a step δx is chosen from a Gaussian distribution (with width w). The Metropolis algorithm is used to accept or reject the step: if the ratio $|P(x+\delta x)/P(x)|$ is greater than a random number between 0 and 1, the step is accepted. Regardless of whether the step is accepted or rejected, a new velocity is chosen at random, and the procedure starts over. The sign of $P(x)$ is recorded and used for the calculation of observables. Input parameters are the molecular dynamics run time ϵ, the probability of a Monte Carlo step p, and the average Monte Carlo step size w.

Figure 7 shows a typical trajectory. Many of the features shown are similar to the features of the molecular-dynamics trajectory shown in Fig. 6. The discontinuities in the trajectory are the discrete Monte Carlo steps, which allow the system to cross the nodes. After each Monte Carlo step the slope is different because a velocity randomization has taken place. The discontinuities in the slope when there is no Monte Carlo step visible indicate that the Monte Carlo step was rejected. Note that, on average, the time between node crossings is considerably larger than the typical time to move from one side of a region to the other.

For fermion systems, efficient procedures for solving the molecular-dynamics equations are known, but the standard method[4] for taking Monte Carlo steps is unsuitable in the context of a hybrid algorithm. The standard procedure requires knowledge of a certain matrix which changes each time a variable x is changed. The matrix must be calculated initially at considerable expanse, but thereafter can be updated in considerably less time each

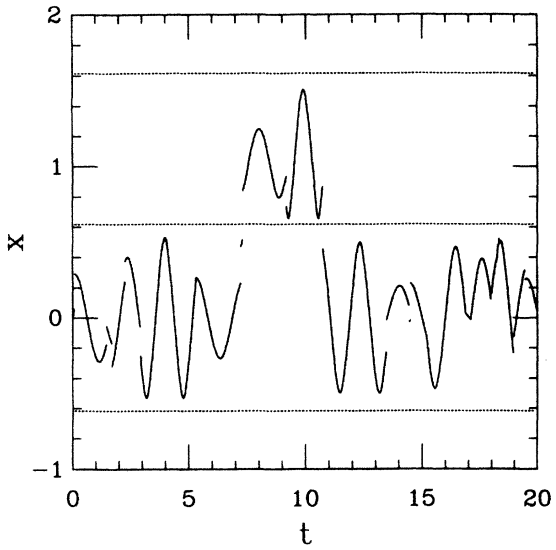

FIG. 7. Typical hybrid molecular-dynamics–Monte Carlo trajectory. The hybrid algorithm includes discrete Monte Carlo steps which allow the particle to hop over the nodes in $P(x)$, allowing the system to sample all of phase space. The parameters for this run (see text) were $\epsilon=0.02$, $p=0.02$, and $w=0.2$.

time an x is changed. In a hybrid algorithm, where most of the steps are not Monte Carlo, the matrix would have to be calculated from scratch before each Monte Carlo step. In the Appendix we give a more efficient way of taking Monte Carlo steps in the context of the hybrid algorithm.

Some features of our model relating to the hybrid algorithm are not representative of typical fermion systems. The many-dimensional nature of the fermion systems is the main difference. Since Monte Carlo methods typically vary only one of the variables at a time, it is impossible to move a very large "distance" in phase space (in terms of correlation time to reach an independent configuration) in one Monte Carlo step. Molecular-dynamics steps in fermion systems change all the variables at once, and thus are able to move a larger "distance." In contrast, we are free to choose w as large as we wish, much larger than the typical distance traveled in a molecular-dynamics step. To make our tests more realistic, we should probably make w small, but the difference in dimensionality and the unknown nature of the nodal surfaces in the fermion systems make any quantitative comparisons difficult.

Unfortunately, using a small w in our model makes the average time between node crossings very large. We can estimate how the rate of crossing varies with w and p for small w as we estimated its dependence on Δt in Sec. III for the Langevin method. For the model $P(x)$, we find that the probability of crossing at a particular Monte Carlo step is roughly proportional to w^2. The average number of Monte Carlo steps is *strictly* proportional to p, and we expect the average number of crossings per hybrid step to be strictly proportional to p (even when w is not small). Hence, for small w we expect the average number of crossing per hybrid step (i.e., per molecular-dynamics step plus possible Monte Carlo step) to be proportional to pw^2. The rate of crossing for the model $P^{(2)}(x)$ is similarly expected to be proportional to pw^3. The rate of crossing is independent of the molecular-dynamics integration time ϵ because molecular dynamics cannot cross the nodes, and on average the probability of being an any point at the start of a Monte Carlo step is $|P(x)|$, independent of the parameters ϵ, w, or p.

The molecular-dynamics steps *do* affect correlation times because they allow the system to move quickly within the nodally-bounded regions. The relative importance of molecular dynamics versus the Monte Carlo steps in a fermion system depends on the relative importance of moving within regions compared to crossing nodes for that system. For large systems where most of the important areas of phase space are contained within one nodally-bounded region, the efficient movement within the region possible with molecular dynamics will probably favor the hybrid algorithm over pure Monte Carlo.

Table I shows the results of calculations using the hybrid algorithm. The first group shows the variation of the crossing rate with w. For very large w the crossing rate is reduced because most attempted jumps take the system away from the region near the origin. For small w the crossing rate is roughly proportional to w^2. The second group shows the proportionality of the rate of crossing with p. The last group is for the model $P^{(2)}(x)$, where the crossing rate for small w is roughly proportional to w^3. In all cases the crossing rate determines the statistical accuracy (for a given number of steps). Accurate results require the number of crossings to be very high, since in order to properly weigh the probability of being in each of the nodally-bounded regions, the nodes must be crossed many times. Thus obtaining a small statistical error with small w and p requires very long running times. In this model efficient movement through phase space within each of the nodally-bounded regions is less important that movement across the nodes.

VI. CONCLUSIONS

It is difficult to use *any* simulation technique when the probability function is not positive definite (as is frequently the case in fermion systems) because of the increased statistical fluctuations in observables calculated using (5). In the Langevin and molecular-dynamics techniques additional difficulties arise due to singularities in the effective action corresponding to nodes in the probability function. These difficulties persist even when the probability function is proportional to the square of the fermion determinant (as in the half-filled Hubbard model), since the same sign changes in the fermion determinant produce second-order "nodes" in the probability function and hence singularities in the effective action.

In Sec. II we discussed the typical form of a fermion effective action and devised a simple one-dimensional model $P(x)$ for testing simulation methods, as well as a model $P^{(2)}(x)$ for the case where the probability function is proportional to square of a determinant. In Sec. III we showed that in the Langevin method, the system always crosses the nodes (for time steps as small as 10^{-10}). We

TABLE I. Hybrid molecular-dynamics–Monte Carlo calculations for the simple model. In each step, the molecular-dynamics equations are integrated with a variable time-step integrator for a total time of ϵ, after which there is a probability p that a Monte Carlo move (of average distance w) will be attempted. In the runs shown, $\epsilon=5\times10^{-2}$, and a total of 10^6 steps were taken. After each Monte Carlo attempt (whether it is accepted or rejected) a new velocity is chosen from a Boltzmann distribution. The last five runs are for the model $P^{(2)}(x)$, the rest are for $P(x)$. For low p and w the nodes are crossed infrequently, and the simulation cannot accurately weigh the probability of being in the different regions. The exact results for $P(x)$ are $\langle s\rangle=0.3329$, $\langle x\rangle=0$, and $\langle x^2\rangle=0.5$; for $P^{(2)}(x)$ the exact results are $\langle s\rangle=1$, $\langle x\rangle=0$, and $\langle x^2\rangle=217/50=4.34$. The numbers in parentheses are the statistical uncertainty in the last digit (one standard deviation), estimated by dividing the run into 10 parts, treating the averages of the parts as independent, and calculating their variance. (This procedure will underestimate the error if there is correlation between the parts.)

p	w	Crossings/step	$\langle \mathrm{sgn}P(x)\rangle_{\tilde{Z}}$	$\langle x\rangle$	$\langle x^2\rangle$
1.0	3.0	0.13	0.332(8)	0.005(6)	0.482(14)
1.0	2.0	0.17	0.331(8)	0.006(6)	0.500(8)
1.0	1.5	0.19	0.336(2)	0.005(8)	0.490(6)
1.0	1.0	0.21	0.333(2)	0.004(6)	0.501(11)[a]
1.0	0.7	0.19	0.332(2)	−0.003(7)	0.524(16)
1.0	0.5	0.15	0.339(2)	0.007(9)	0.513(11)
1.0	0.3	0.071	0.333(2)	−0.014(13)	0.503(32)
1.0	0.2	0.034	0.339(4)	0.051(30)	0.553(57)
1.0	0.1	0.0090	0.337(7)	0.023(39)	0.668(102)
0.2	1.0	0.042	0.333(4)	0.003(22)	0.508(22)
0.05	1.0	0.011	0.332(13)	−0.026(32)	0.493(80)
0.01	1.0	0.0021	0.356(19)	0.065(40)	0.511(166)
1.0	1.0	0.13	1.000(0)	−0.002(12)	4.353(13)
1.0	0.5	0.063	1.000(0)	−0.011(27)	4.334(20)
1.0	0.3	0.020	1.000(0)	−0.004(67)	4.258(52)
1.0	0.2	0.0065	1.000(0)	0.081(131)	4.395(84)
1.0	0.1	0.00082	1.000(0)	0.233(270)	4.710(158)

[a]Additional runs with $p=1$, $w=1$, and ϵ ranging from 10^{-5} to 0.2 showed no change (within statistical errors, which were also unchanged) of any of the results as ϵ was varied.

found that the rate of crossing tends to a constant as the time step $\Delta t\to0$ for $P(x)$, and varies as $\sqrt{\Delta t}$ for $P^{(2)}(x)$. We also found that the system could be driven completely out of equilibrium when, on a particular time step, the system landed almost exactly on a node. In addition, we pointed out that the Langevin method does not indicate when a node has been crossed, which makes keeping track of the sign of the probability function, needed for calculating observables, very difficult. In Sec. IV we showed that in the molecular-dynamics method, the system is trapped in one region bounded by nodes and cannot explore all of phase space. In addition, a variable time-step integrator is needed to avoid numerical problems in integrating the equations near the nodes.

In Sec. V, we proposed a hybrid molecular-dynamics–Monte Carlo algorithm which we used to correctly simulate our simple model. In the model the key to simulations with low statistical error was to make the system cross the nodes very frequently, so that the proper probability weight could be assigned to each of the nodally-bounded regions. In this regard the Monte Carlo part of the algorithm was much more useful than the molecular-dynamics part, and the algorithm performed best in the pure Monte Carlo limit. In real fermion systems, however, molecular dynamics can be done much more quickly than Monte Carlo, with the difference increasing with the size of the system. Thus, in the case of very large fermion systems, the hybrid method may be faster.

ACKNOWLEDGMENTS

S.R.W. would like to thank Richard Scalettar, Jorge Hirsch, and Ken Wilson for helpful discussions. This work was supported in part by the Department of Energy-Basic Energy Sciences, Division of Materials Research. This research was conducted using the Cornell National Supercomputer Facility, funded in part by the National Science Foundation, New York State, and IBM.

APPENDIX

We present a fermion Monte Carlo method suitable for use in the hybrid algorithm. In particular, we consider the evaluation of $P(x')/P(x)$ when the effective action is of the form (1). The most time-consuming part of the calculation of this ratio is evaluation of the determinants; so for the discussion we can set $P(x)=\det M(x)$.

In a fermion system with N sites and the inverse temperature β divided into L time slices, $M(x)$ can be written as the $2L\times2L$ matrix with $N\times N$ elements[4,5]

$$M(x)=\begin{bmatrix} 1 & -V(1) & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & -T & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & -V(2) & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & -T & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ +T & 0 & 0 & 0 & 0 & \cdots & 1 \end{bmatrix} . \tag{A1}$$

The $N\times N$ matrices T are independent of x, while $V(1),\ldots,V(L)$, depend on x.[16] We have assumed $M(x)$ comes from a Trotter approximation which breaks the Hamiltonian into kinetic (T) and potential $V(l)$ pieces, followed by a Hubbard-Stratonovich transformation. The form for $M(x)$ in (16) is large but sparse; alternatively, we can write the determinant in terms of a smaller ($N\times N$), dense matrix

$$\det M(x)=\det[1+TV(1)\cdots TV(L)] . \tag{A2}$$

We first discuss the usual Monte Carlo procedure,[4] following the discussion of Scalettar.[5] We assume there are NL variables x_{nl}, one for each site and time slice. Consider changing one of the x_{nL} for time slice L. The only change in $M(x)$ will occur in $V(L)$, which we can write as $V(L)\rightarrow V(L)\Delta$, where Δ is also an $N\times N$ matrix. Then we can write

$$\frac{\det M(x')}{\det M(x)}=\det[1+(1-G)(\delta-1)] , \tag{A3}$$

where

$$G=[1+TV(1)\cdots TV(L)]^{-1} . \tag{A4}$$

One finds that $\delta-1$ has only a few nonzero entries, so this ratio can be evaluated very quickly if one knows G. Computation of G from scratch requires $O(N^3)$ operations, but this need only be done once: after a single x_{nl} is changed, it is possible to update G with only N^2 operations.[4] Since there are NL variables x_{nl}, a sweep through the entire lattice requires $O(N^3L)$ operations. Unfortunately, if only a single Monte Carlo step of a single variable x_{nl} is required (as in the hybrid algorithm), G must be calculated from scratch, and the cost is N^3 (or N^4L for the entire lattice).

We now propose an alternative procedure more suitable for use in the hybrid algorithm. We note that it is possible to arrange the Trotter breakup so that $V(l)$ is diagonal,[16] with diagonal element $[V(l)]_{nn}$ depending only on x_{nl}. In that case, a change in x_{nl} only affects a single element of (16); i.e., $M(x')=M(x)+\bar{\Delta}$, where $\bar{\Delta}$ has only one nonzero element. The ratio of determinants in this case can be written as

$$\frac{\det M(x')}{\det M(x)}=\det[1+M^{-1}\bar{\Delta}] , \tag{A5}$$

Since $\bar{\Delta}$ has only one nonzero element, say $\bar{\Delta}_{ij}$, the expression for the determinant ratio is drastically simplified to

$$\frac{\det M(x')}{\det M(x)}=1+M^{-1}_{ji}\bar{\Delta}_{ij} , \tag{A6}$$

where there is no sum on i or j. Only a single element of M^{-1} is needed.

A single row m of M^{-1} can be obtained by solving the system of equations $Mm=e$, where e is a unit vector. The solution of systems of equations of this form is required for efficient fermion molecular dynamics and Langevin calculations, and recently considerable progress has been made using preconditioned conjugate-gradient techniques.[13] The calculation time for solving this system is roughly proportional to the dimension of the matrix, in this case $2NL$. Hence, the calculation time for a single Monte Carlo step is reduced from $O(N^3)$ to $O(NL)$. [If this procedure is used in a pure Monte Carlo algorithm the calculation time for a sweep is $O(N^2L^2)$ instead of $O(N^3L)$. This may result in a savings in some cases.]

---

[1]For example, J. Hirsch has studied the Hubbard model in two and three dimensions; see J. E. Hirsch, Phys. Rev. Lett. **51**, 1900 (1983); Phys. Rev. B **31**, 4403 (1985); **35**, 1851 (1987). Gubernatis *et al.* have studied two-dimensional fermion gases; see J. E. Gubernatis, D. J. Scalapino, R. L. Sugar, and W. D. Toussaint, *ibid.* **32**, 103 (1985).

[2]A special case where the probability function is never *negative*, but can be zero, is the half-filled Hubbard model (see Sec. II).

[3]S. Duane, Nucl. Phys. **B257[FS14]**, 652 (1985); J. B. Kogut, *ibid.* **B275[FS17]**, 1 (1986); E. Dagotto and J. B. Kogut, Phys. Rev. Lett. **58**, 299 (1987).

[4]R. Blankenbecler, D. J. Scalapino, and R. L. Sugar, Phys. Rev. D **24**, 2278 (1981); D. J. Scalapino and R. L. Sugar, Phys. Rev. B **24**, 4295 (1981).

[5]R. T. Scalettar, Ph. D. thesis, University of California, 1986.

[6]J. E. Hirsch, Phys. Rev. B **31**, 4403 (1985).

[7]For additional discussion on this point see Ref. 13.

[8]G. Parisi and Wu Yongshi, Sci. Sin. **24**, 483 (1981).

[9]Recently there has been interest in the use of the Langevin method for *complex* weights; see J. R. Klauder, Phys. Rev. A **29**, 2036 (1984). For a comparison of Langevin and Monte Carlo methods for complex weights, see H. Q. Lin and J. E. Hirsch, Phys. Rev. B **34**, 1964 (1986).

[10]G. G. Batrouni, G. R. Katz, A. S. Kronfeld, G. P. Lepage, B. Svetitsky, and K. G. Wilson, Phys. Rev. D **32**, 2736 (1985).

[11]G. G. Batrouni, Phys. Rev. D **33**, 1815 (1986).

[12]E. Helfand, Bell Syst. Tech. J. **58**, 2289 (1979); H. S. Greenside and E. Helfand, *ibid.* **60**, 1927 (1981).

[13]R. T. Scalettar, D. J. Scalapino, R. L. Sugar, and D. Toussaint, Phys. Rev. B **36**, 8632 (1987).

[14]H. C. Andersen, J. Chem. Phys. **72**, 2384 (1980); P. H. Berens, S. R. White, and K. R. Wilson, *ibid.* **75**, 515 (1981).

[15]W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes: The Art of Scientific Computing* (Cambridge University Press, Cambridge, 1986), p. 563.

[16]See the second paper (by Hirsch) listed in Ref. 1.