PHYSICAL REVIEW E **85**, 016709 (2012)

Optimizing the accuracy of lattice Monte Carlo algorithms for simulating diffusion

Mykyta V. Chubynsky[*] and Gary W. Slater[†]
Department of Physics, University of Ottawa, 150 Louis-Pasteur, Ottawa, Ontario, Canada K1N 6N5
(Received 20 August 2010; revised manuscript received 24 September 2011; published 12 January 2012)

The behavior of a lattice Monte Carlo (LMC) algorithm (if it is designed correctly) must approach that of the continuum system that it is designed to simulate as the time step and the mesh step tend to zero. However, we show for an algorithm for unbiased particle diffusion that if one of these two parameters remains fixed, the accuracy of the algorithm is optimal for a certain *finite* value of the other parameter. In one dimension, the optimal algorithm with moves to the two nearest neighbor sites reproduces the correct second and fourth moments (and minimizes the error for the higher moments at large times) of the particle distribution and preserves the first two moments of the first-passage time distributions. In two and three dimensions, the same level of accuracy requires simultaneous moves along two axes ("diagonal" moves). Such moves attempting to cross an impenetrable boundary should be projected along the boundary, rather than simply rejected. We also treat the case of absorbing boundaries. We discuss the relation between optimally accurate LMC algorithms and a particular case of lattice Boltzmann (LB) algorithms for simulating diffusion and compare the computational efficiency of optimal LMC and optimal LB algorithms.

DOI: 10.1103/PhysRevE.85.016709 PACS number(s): 02.70.Tt, 05.10.Ln, 05.40.Jc, 87.15.Vv

I. INTRODUCTION

Particles diffusing in a liquid medium perform random walks. In a computer simulation of this process, it is sometimes convenient to introduce a lattice, assume that the particles can only reside at lattice sites, and allow only certain types of moves between the sites [1–17]. Different variants of such *lattice Monte Carlo* (LMC) algorithms are distinguished by the sets of allowed moves and the corresponding probabilities. These are chosen depending on the purpose of the algorithm. For instance, Metropolis Monte Carlo [18] can be used to study equilibrium properties, but is, in general, inadequate for dynamics, especially in a strong external field [12]. For dynamical algorithms, another important (and often neglected) ingredient is the time step (the amount by which the time is incremented after each attempted LMC move) [19]. While, in principle, the time step can vary during a simulation, in this paper we restrict ourselves to algorithms where it remains constant. Moreover, we only consider the case of *unbiased* diffusion, where there is no external field or other external influence (such as a flow) that would drive the particles preferentially in a particular direction. Some of the approaches used in this paper are applicable to the case of biased diffusion as well, and this will be described in a separate publication. We also assume that the system is uniform, that is, the diffusion constant is the same everywhere.

As an example of a dynamical LMC algorithm, consider unbiased one-dimensional (1D) diffusion along an infinite line, with the 1D lattice sites placed equidistantly and numbered consecutively by integer numbers. In the simplest approach, at each step the particle moves left or right to a neighboring site with equal probabilities. If for a particle that is known to be at site i the probability that it moves to site j during a given time step is $p_{i\to j}$, then

$$p_{j\to j+1} = p_{j\to j-1} = 1/2, \tag{1}$$
$$p_{j\to l} = 0 \quad \text{for} \quad l \neq j \pm 1. \tag{2}$$

The time step τ can be determined, for example, using a mapping onto the mean first-passage time between sites [19] and is equal to

$$\tau = \frac{a^2}{2D}, \tag{3}$$

where a is the distance between the lattice sites (the lattice constant or the mesh step) and D is the diffusion constant of the particle in the medium.

Rather than working with individual particles, one can look at the evolution of particle distributions, described by the set of the mean particle numbers at each site, $\{n_i(t)\}$, where the subscript i refers to the site number. In general, the mean particle numbers after the move are given in terms of those before the move by

$$n_j(t+\tau) = \sum_l p_{l\to j} n_l(t), \tag{4}$$

where, in principle, $|l-j|$ can be allowed to be larger than unity. Equation (4) is known as the *master equation*. For the particular simple algorithm described by Eqs. (1)–(3),

$$n_j(t+\tau) = \tfrac{1}{2}[n_{j-1}(t) + n_{j+1}(t)]. \tag{5}$$

The set of equations (5) for all sites can be solved numerically for given initial and boundary conditions, and this provides, in a sense, a *numerically exact* solution of the LMC algorithm [20]. In effect, we have two variants of the LMC approach: the particle-based approach and the numerically exact master equation approach. Both can be used in practice and our results in this paper apply to both, with the former becoming equivalent to the latter in the limit when averaging over infinitely many realizations is done. Throughout this paper, we assume that the time step τ is the same for all sites

[*]chubynsky@gmail.com
[†]gslater@uottawa.ca

and moves and is history-independent. This is essential for the numerically exact variant; while not strictly necessary for the particle-based approach (e.g., in kinetic Monte Carlo [2,21–23] the time increment is different for different moves), keeping the time step constant simplifies both the analysis and the implementation of the LMC algorithm.

On the other hand, diffusion can be studied using *continuum equations* that describe the time evolution of the particle concentration. For example, for unbiased diffusion in 1D,

$$\frac{\partial n(x,t)}{\partial t} = D\frac{\partial^2 n(x,t)}{\partial x^2}. \tag{6}$$

To solve such an equation numerically, one can discretize it replacing the derivatives with differences:

$$\frac{n(x,t+\tau)-n(x,t)}{\tau} \approx D\times\frac{n(x+a,t)-2n(x,t)+n(x-a,t)}{a^2} \tag{7}$$

or

$$n(x,t+\tau) \approx n(x,t)+\frac{D\tau}{a^2}[n(x+a,t) - 2n(x,t)+n(x-a,t)]. \tag{8}$$

This is known as the forward time centered space finite-difference scheme [24]. If $x = ja$, where j is integer, then we get a set of equations involving only the values of $n(x,t)$ at discrete points:

$$n_j(t+\tau) \approx n_j(t)+\frac{D\tau}{a^2}[n_{j+1}(t)-2n_j(t)+n_{j-1}(t)], \tag{9}$$

where $n_j(t) \equiv n(ja,t)$. Note that this coincides with Eq. (5) when τ is given by Eq. (3). In other words, the master equation for a LMC algorithm can be viewed as a discretization of the continuum equation for the same diffusion problem.

Note, however, that other choices of the time step τ are possible in Eq. (9). For each such choice, one can define a new LMC algorithm with a new set of transition probabilities by comparing Eqs. (9) to (4) (with the caveat that these probabilities have to remain bounded between zero and one). Indeed, if we rewrite Eq. (9) as

$$n_j(t+\tau) \approx \frac{D\tau}{a^2}n_{j-1}(t)+\left(1-\frac{2D\tau}{a^2}\right)n_j(t)+\frac{D\tau}{a^2}n_{j+1}(t) \tag{10}$$

and compare this expression to Eq. (4), we obtain

$$p_{j-1\to j} = p_{j+1\to j} = \frac{D\tau}{a^2}, \tag{11}$$

$$p_{j\to j} = 1-\frac{2D\tau}{a^2}. \tag{12}$$

To ensure that these probabilities remain between zero and one, the time step must be in the range

$$0<\tau\leqslant\frac{a^2}{2D}. \tag{13}$$

Note that, in general, we now have a nonzero probability $p_{j\to j}$ of staying at the same site during a particular time step (Fig. 1). The only exception is the particular case of the algorithm given

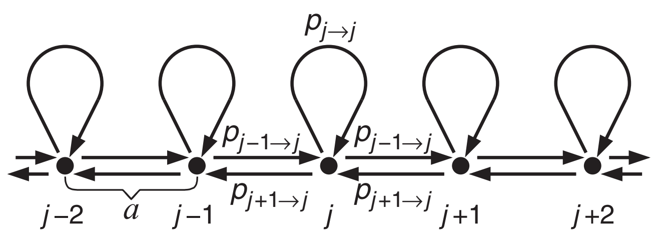

FIG. 1. A schematic of the 1D algorithm considered in this paper. At a particular step, in addition to the usual moves to nearest neighbors, the particle can also stay put (which can be considered as the move to the original site). The notation for the probabilities of the moves is given. These probabilities (along with the time step) can be adjusted to optimize the algorithm. For unbiased diffusion, we always have $p_{j-1\to j} = p_{j+1\to j}$.

by Eqs. (1)–(3), which corresponds to the largest possible time step,

$$\tau_{\max} = \frac{a^2}{2D}. \tag{14}$$

Such an algorithm (to which we refer as the *ordinary algorithm*) is a logical choice from the point of view of the computational efficiency of the simulation. However, choosing a shorter time step can increase the *accuracy* of the LMC algorithm.

The problem of optimizing the accuracy of the algorithm given by Eqs. (11) and (12) seems trivial at first. Indeed, the smaller the time step and the mesh step, the more accurately the discretization (9) represents the continuum equation (6) and thus the more accurate the corresponding LMC algorithm. In practice, however, computational resources are limited and one can only choose the smallest time step for which the simulation is still feasible. The practically relevant question then is as follows: Given that time step, what is the optimal mesh step? Based on Eq. (13), it cannot be infinitely small. The smallest possible mesh step (the finest space discretization), $a_{\min} = \sqrt{2D\tau}$, corresponds to the ordinary algorithm described by Eqs. (1)–(3). While at first glance it seems it should provide the best accuracy, we will see below that this is not the case. Conversely, in some cases one may wish to fix the mesh step (for example, choosing the largest value that ensures that spatial inhomogeneities are still reproduced accurately) and then optimize the time step. The largest possible time step, $\tau = a^2/2D$ [again, corresponding to Eqs. (1) and (2)], as mentioned, is optimal in terms of computational speed. As for the accuracy, at first glance it may seem that the smaller the τ the better, but again, this trivial answer turns out to be incorrect.

This problem of optimizing the accuracy of LMC algorithms for diffusion given a fixed time step or mesh step is the subject of this paper. While in our considerations we fix the mesh step and allow the time step and the transition probabilities to vary, it will be clear from our derivation that fixing the time step instead will lead to the same probabilities and the same relation between the mesh step and the time step at optimality, at least as long as all inhomogeneities, such as obstacles and their features, are much larger than the mesh step. In the next section, we solve the optimization problem for the 1D diffusion algorithm described above. We do this using three different approaches, with the same end result. In Secs. III and IV, we consider the 2D and 3D analogs of the same problem and show that the same accuracy as in 1D requires the introduction of simultaneous moves in at least two directions, or *diagonal moves*. In Secs. V and VI, we show how the LMC rules should be modified for sites next to reflecting or absorbing boundaries. In Sec. VII, we discuss the relation between the optimal LMC algorithms and lattice Boltzmann algorithms for simulating fluid dynamics and diffusion. We end the paper with a discussion of our results.

II. ONE DIMENSION

In this section, we optimize the 1D LMC algorithm for unbiased diffusion with the moves restricted to nearest neighbors. We do this in three different ways. The first approach is based on the comparison between the solutions of the continuum equation (6) and the master equation (4) (with only $p_{j\pm1\to j}$ and $p_{j\to j}$ nonzero); they should match particularly for long-wavelength, slowly decaying modes. The other two methods compare the exact moments of the particle distribution or the first-passage time (FPT) distribution with those of the respective distributions generated by the algorithm. Using these three different approaches reveals more properties of the optimal algorithm, but the resulting sets of parameters are the same. We also consider the *full* FPT distributions and show numerically that while for both the ordinary algorithm [Eqs. (1)–(3)] and the optimally accurate algorithm the distribution eventually approaches the correct one as the mesh gets finer, the latter algorithm achieves the same accuracy for a much coarser mesh.

A. Comparison to the continuum equation

Note first that the general solution of the continuum equation (6) on an infinite line can be written as a sum (or, rather, an integral) over modes with different wave numbers k, each of which decays exponentially in time:

$$n(x,t) = \int_{-\infty}^{\infty} C(k)\exp[ikx - \alpha_c(k)t]dk, \tag{15}$$

where $C(k)$ is an arbitrary complex function [except for the fact that we need $C(-k) = C^*(k)$ for the solution to be real] and the dispersion relation is

$$\alpha_c(k) = Dk^2. \tag{16}$$

The general form of the master equation for a LMC algorithm in 1D where only moves between neighboring sites are allowed is

$$n_j(t+\tau) = p_{j\to j}n_j(t) + p_{j-1\to j}n_{j-1}(t) + p_{j+1\to j}n_{j+1}(t). \tag{17}$$

The general solution of the system of master equations (17) can be written in a form similar to Eq. (15), except for the fact that the integration limits are finite due to space discretization:

$$n_j(t) = \int_{-\pi/a}^{\pi/a} C(k)\exp[ikaj - \alpha_d(k)t]dk. \tag{18}$$

The dispersion relation $\alpha_d(k)$ can be obtained by substituting Eq. (18) in Eq. (17):

$$\alpha_d(k) = -\frac{1}{\tau}\ln[p_{j\to j} + p_{j-1\to j}\exp(-ika) + p_{j+1\to j}\exp(ika)]. \tag{19}$$

Note that Eq. (19) is the *exact* dispersion relation for the discrete equation (17), even for finite values of a and τ.

The closer Eq. (19) approximates the dispersion relation (16) for the exact, continuum equation, the more accurate the discretization and the corresponding LMC scheme. One can argue that the behavior of the solution of the diffusion equation on the longest length scales (smallest k) should be reproduced in the first place, especially given that these longest-wavelength modes also take the most time to decay. For this reason, we expand the dispersion relations Eqs. (16) and (19) in powers of k and try to match as many terms as possible. Since Eq. (16) contains only a k^2 term, we need to find the conditions that would eliminate as many terms as possible in the series expansion of Eq. (19), starting with the lowest order, except for the k^2 term whose coefficient should equal D. The k^0 term disappears if

$$p_{j\to j} + p_{j-1\to j} + p_{j+1\to j} = 1. \tag{20}$$

In other words, the probabilities must be normalized. Given Eq. (20), we find that the k^1 term in Eq. (19) has a zero coefficient if

$$\frac{ia}{\tau}(p_{j-1\to j} - p_{j+1\to j}) = 0. \tag{21}$$

This means that the probabilities of moving to the left and right should be equal, which makes sense since our diffusion process is unbiased. Taking conditions (20) and (21) into account, Eq. (19) becomes

$$\alpha_d(k) = -\frac{1}{\tau}\ln[1 + p_{j-1\to j}(-k^2a^2 + k^4a^4/12 - k^6a^6/360 + \cdots)]. \tag{22}$$

After expanding the logarithm in a series, the k^2 term of the resulting expression matches Eq. (16) if

$$\frac{a^2 p_{j-1\to j}}{\tau} = D. \tag{23}$$

This is equivalent to Eq. (11) and ensures the correct diffusion rate. Requiring that the next (k^4) term be equal to zero gives the expression

$$-\frac{1}{\tau}\left(-\frac{p_{j-1\to j}^2 a^4}{2} + \frac{p_{j-1\to j}a^4}{12}\right) = 0. \tag{24}$$

The solution of this equation is simply

$$p_{j-1\to j} = \tfrac{1}{6}. \tag{25}$$

Note that $p_{j-1\to j} = 0$ is not a solution, since in the limit $p_{j-1\to j} \to 0$ not just the expression in the parentheses in Eq. (24), but also τ approaches zero and the ratio $p_{j-1\to j}/\tau$ remains finite, according to Eq. (23). This means that the k^4 term is not eliminated in the limit $\tau \to 0$ and so, perhaps surprisingly, the algorithm is not optimal in this limit. From Eqs. (20), (21), (23), and (25), we then get

$$p_{j\pm1\to j} = \frac{1}{6}, \tag{26}$$

$$p_{j\to j} = \frac{2}{3}, \tag{27}$$

$$\tau = \frac{a^2}{6D}. \tag{28}$$

Interestingly, the optimal algorithm (in terms of accuracy) requires a time step that is $1/3$ of the maximum value $\tau_{\max}$ allowed for a particular step length a [as given by Eq. (14)]; as a consequence, the particle must stay put $2/3$ of the time. While this is costly in terms of computing time, it is obviously better than the naive expectation that optimality would be achieved for $\tau \to 0$ even when a is finite.

Ideally, further terms in the series expansion of Eq. (19) should also be zero. All odd-order terms are automatically zero, but for even-order terms starting with $O(k^6)$ this is impossible to achieve since we have run out of adjustable LMC parameters. Indeed, with $p_{j-1\to j}$ and τ given by Eqs. (26) and (28), respectively, Eq. (22) becomes

$$\alpha_d(k) = D(k^2 - a^4k^6/540 + \cdots). \tag{29}$$

The sixth-order term (and all subsequent terms) do vanish in the limit $a \to 0$ (which also automatically means $\tau \to 0$). For a fixed and finite mesh step a, one can introduce more parameters by allowing longer-range moves, but this can cause problems since it would effectively increase the coarseness of the discretization of space (e.g., with jumps of size $2a$, a particle could move through an obstacle of size a).

In Fig. 2, we compare the exact dispersion relation $\alpha_c(k)$, as given by Eq. (16), with the dispersion relations that correspond to LMC algorithms built with different time steps τ. The latter dispersion relations are obtained using Eq. (19) with the probabilities given by Eqs. (11) and (12) and thus satisfying Eqs. (20), (21), and (23), which guarantees the correct expansion terms up to k^2, but the k^4 term only vanishes for $\tau = \frac{1}{3}\tau_{\max}$. For the maximum possible value $\tau = \tau_{\max} = a^2/2D$, which corresponds to the ordinary algorithm with zero probability of staying on the site, two peculiarities catch the eye. First, at $k = \pi/2a$, $\alpha_d(k)$ diverges; that is, the corresponding mode decays infinitely fast. Indeed, the distribution

$$\ldots,0,\tfrac{1}{2},1,\tfrac{1}{2},0,\tfrac{1}{2},1,\tfrac{1}{2},0,\ldots \tag{30}$$

decays to the uniform distribution $\ldots,\frac{1}{2},\frac{1}{2},\frac{1}{2},\ldots$ in a single time step and does not change afterward. Even more strikingly, for $k = \pi/a$, the real part of $\alpha_d(k)$ is zero; that is, this mode does not decay at all. At the same time, the imaginary part is $\mathrm{Im}\,\alpha_d(\pi/a) = \pi/\tau$. As a result, the initial distribution

$$\ldots,0,1,0,1,\ldots \tag{31}$$

oscillates indefinitely:

$$\ldots,0,1,0,1,\ldots \to \ldots,1,0,1,0,\ldots \to \ldots,0,1,0,1,\ldots \to \ldots. \tag{32}$$

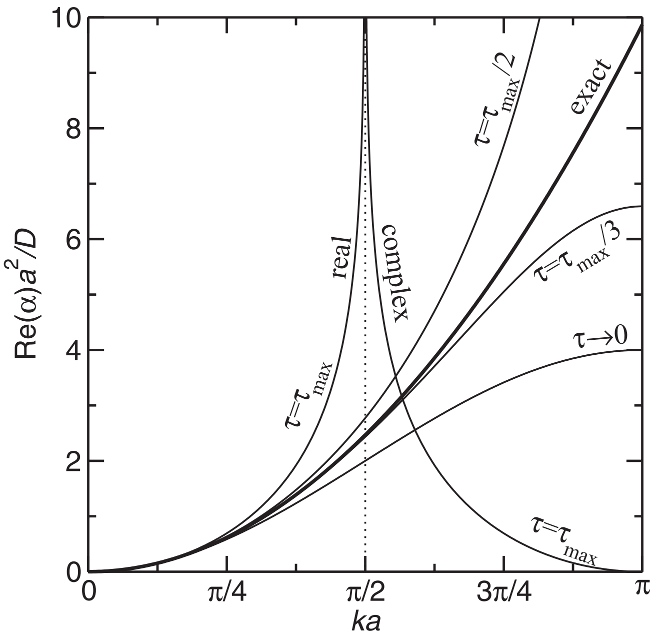

FIG. 2. For Fourier components of a 1D particle distribution, characterized by their wave number k, a comparison between the exact decay rate obtained by solving the continuum diffusion equation [given by Eq. (16), thick line] and the decay rate observed in a LMC algorithm with time step τ [given by Eq. (19)] for several different values of τ (thin lines). For the maximum possible time step, $\tau = \tau_{\max} = a^2/2D$, the decay rate diverges at $ka = \pi/2$, and for larger ka it is complex, becoming purely imaginary ($\mathrm{Re}\,\alpha = 0$) at $ka = \pi$. As τ decreases, the wave number at which the divergence occurs shifts to higher k, reaching $ka = \pi$ when $\tau = \tau_{\max}/2$. The closest matching between the exact and LMC decay rates at small k is observed when $\tau = \tau_{\max}/3$.

One practical consequence of this is that a particle starting at one of the even-numbered sites will always visit only odd-numbered sites at odd-numbered time steps, and only even-numbered sites at even-numbered time steps. When a smaller time step is used with the same lattice constant a, there is some probability of staying on the same site at each time step, and oscillations similar to Eq. (32) decay. Eventually, for $\tau < \frac{1}{2}\tau_{\max}$, the oscillations do not occur at all, as α_d is always real in that case. The exact dispersion curve is matched best for small k when $\tau = \frac{1}{3}\tau_{\max}$, as expected.

Instead of using the explicit solution of the master equations [Eq. (18)], another possible approach is to insert the partial solutions of the continuum problem [i.e., the integrand of Eq. (15)] in the master equation (17), replacing x with ja, and find for what values of the parameters the resulting equality is satisfied most accurately for small k. After the substitution, dividing both sides by $C(k)\exp(ikaj - Dk^2t)$, we obtain

$$\exp(-Dk^2\tau) \overset{?}{\simeq} p_{j\to j} + p_{j-1\to j}\exp(-ika) + p_{j+1\to j}\exp(ika). \tag{33}$$

Here we use the $\overset{?}{\simeq}$ sign as a reminder that this is a relation that needs to be verified (hence the question mark); moreover, it is not expected to hold exactly, since lattice diffusion is only an approximate representation of the continuum diffusion process (thus, "$\simeq$"). By expanding both sides in the Taylor series in k we can verify that this equality is satisfied to $O(k^4)$ solely when the probabilities and the time step are given by Eqs. (26)–(28).

An algorithm with a nonzero probability to stay put was proposed before [19] for *biased* diffusion in an external field. It was shown that in that case this nonzero probability was necessary even to reproduce the correct diffusion constant. Interestingly, the zero-field limit of that algorithm coincides with the optimal algorithm derived here, even though in zero field even the ordinary algorithm reproduces the diffusion constant correctly.

B. Moments of the particle distribution

Another way of constructing and analyzing LMC algorithms is by looking at the moments of the particle distribution. In continuum space, for particles starting at the origin ($x = 0$) at $t = 0$, the distribution at time t is

$$n(x,t) = \frac{1}{2\sqrt{\pi Dt}} e^{-x^2/4Dt}. \tag{34}$$

The moments of this distribution are

$$\langle x^{2m}\rangle = (2m-1)!!(2Dt)^m = (2m-1)!! \times \langle x^2\rangle^m. \tag{35}$$

All odd moments are zero. The first three nonzero moments are

$$\langle x^2\rangle = 2Dt, \tag{36}$$
$$\langle x^4\rangle = 12D^2t^2, \tag{37}$$
$$\langle x^6\rangle = 120D^3t^3. \tag{38}$$

We now compute the moments of the particle distribution during a LMC random walk. For a lattice walk starting at site 0, the position after N steps is

$$x_N = a\sum_{i=1}^{N} \eta_i, \tag{39}$$

where η_i is $+1$ (-1) for a move to the right (left) and 0 when the particle does not move. We assume from the outset that $p_{j-1\to j} = p_{j+1\to j}$, which makes all odd moments zero automatically. Before proceeding, it is convenient to note that for any positive integer m,

$$\begin{aligned}\langle \eta^{2m}\rangle &= 0^{2m} \times p_{j\to j} + 1^{2m} \times p_{j-1\to j} + (-1)^{2m} \times p_{j+1\to j}\\ &= p_{j-1\to j} + p_{j+1\to j} = 2p_{j-1\to j},\end{aligned} \tag{40}$$

$$\begin{aligned}\langle \eta^{2m-1}\rangle &= 0^{2m-1} \times p_{j\to j} + 1^{2m-1} \times p_{j-1\to j}\\ &\quad + (-1)^{2m-1} \times p_{j+1\to j}\\ &= p_{j-1\to j} - p_{j+1\to j} = 0.\end{aligned} \tag{41}$$

The second moment, which is the average square of Eq. (39), is

$$\langle x_N^2\rangle = a^2 \sum_{i,k=1}^{N} \langle \eta_i\eta_k\rangle. \tag{42}$$

Since different steps are uncorrelated,

$$\langle \eta_i\eta_k\rangle = \delta_{ik}\langle \eta^2\rangle = 2\delta_{ik}p_{j-1\to j}. \tag{43}$$

Here and below $\delta_{i_1,\ldots,i_m}$ is (the generalization of) the Kronecker's δ, which is unity when all indices coincide and zero otherwise. We obtain

$$\langle x_N^2\rangle = 2a^2p_{j-1\to j}\sum_{i,k=1}^{N}\delta_{ik} = 2a^2p_{j-1\to j}N = \frac{2a^2p_{j-1\to j}}{\tau}t. \tag{44}$$

This coincides with the continuum result Eq. (36) when $a^2p_{j-1\to j}/\tau = D$. This is the same as Eq. (11) [or (23)] that is obtainable from matching the k^2 terms in the continuum and discrete dispersion relations. Thus, the second moment is correct *at all times* for both the ordinary and the optimal algorithms and, in general, whenever Eqs. (11) and (12) are valid. For the fourth moment,

$$\langle x_N^4\rangle = a^4\sum_{i,k,r,s=1}^{N}\langle \eta_i\eta_k\eta_r\eta_s\rangle. \tag{45}$$

The average $\langle \eta_i\eta_k\eta_r\eta_s\rangle$ equals: (1) $\langle\eta^2\rangle^2$ when there are two pairs of equal indices, but the indices in different pairs are different; (2) $\langle\eta^4\rangle$ when all four indices are equal; (3) 0 otherwise. As a single expression,

$$\begin{aligned}\langle \eta_i\eta_k\eta_r\eta_s\rangle &= (\delta_{ik}\delta_{rs} + \delta_{ir}\delta_{ks} + \delta_{is}\delta_{kr} - 3\delta_{ikrs})\langle\eta^2\rangle^2\\ &\quad + \delta_{ikrs}\langle\eta^4\rangle\\ &= 4(\delta_{ik}\delta_{rs} + \delta_{ir}\delta_{ks} + \delta_{is}\delta_{kr} - 3\delta_{ikrs})p_{j-1\to j}^2\\ &\quad + 2\delta_{ikrs}p_{j-1\to j},\end{aligned} \tag{46}$$

and we get

$$\begin{aligned}\langle x_N^4\rangle &= a^4\left[(12N^2 - 12N)p_{j-1\to j}^2 + 2Np_{j-1\to j}\right]\\ &= a^4\big[12(t^2/\tau^2)p_{j-1\to j}^2 + (t/\tau)\\ &\quad \times (2p_{j-1\to j} - 12p_{j-1\to j}^2)\big].\end{aligned} \tag{47}$$

When Eq. (11) is satisfied, so the second moment is correct, this becomes

$$\langle x_N^4\rangle = 12D^2t^2\left[1 + \frac{(p_{j-1\to j} - 6p_{j-1\to j}^2)a^4}{6D^2\tau t}\right]. \tag{48}$$

This approaches the continuum result [Eq. (37)] when $t\to\infty$, but is equal to it at *all* times only if $p_{j-1\to j} = 1/6$, which coincides with Eq. (25) and gives rise to the optimal algorithm Eqs. (26)–(28).

In general, even moments are given by

$$\langle x_N^{2m}\rangle = a^{2m}\sum_{i_j=1}^{N}\langle \eta_{i_1}\ldots\eta_{i_{2m}}\rangle. \tag{49}$$

For the sixth moment, we need $\langle \eta_{i_1}\ldots\eta_{i_6}\rangle$. This equals: (1) $\langle\eta^2\rangle^3$ when there are three pairs of equal indices, but no equality of indices between any of the pairs; (2) $\langle\eta^2\rangle\langle\eta^4\rangle$ when there are four equal indices and another pair of equal indices not equal to the first four; (3) $\langle\eta^6\rangle$ when all six indices are equal; (4) 0 otherwise. When the indices run from 1 to N, there are $15N(N-1)(N-2)$ combinations of the indices of the first type, $15N(N-1)$ combinations of the second type, and N combinations of the third type. Then

$$\begin{aligned}\langle x_N^6\rangle &= a^6[15N(N-1)(N-2)\langle\eta^2\rangle^3\\ &\quad +15N(N-1)\langle\eta^2\rangle\langle\eta^4\rangle + N\langle\eta^6\rangle]\\ &= a^6\big[120N(N-1)(N-2)p_{j-1\to j}^3\\ &\quad +60N(N-1)p_{j-1\to j}^2+2Np_{j-1\to j}\big]\\ &= a^6\big[120(t/\tau)^3p_{j-1\to j}^3+(t/\tau)^2\big(60p_{j-1\to j}^2\\ &\quad -360p_{j-1\to j}^3\big)+(t/\tau)\big(2p_{j-1\to j}-60p_{j-1\to j}^2\\ &\quad +240p_{j-1\to j}^3\big)\big]. \end{aligned} \tag{50}$$

It is easy to check that the coefficient of the t^3 term matches Eq. (38) whenever Eq. (11) is satisfied, that is, whenever the second moment (and the leading term in the fourth moment) are correct. Moreover, the t^2 term is zero if and only if $p_{j-1\to j}=1/6$, that is, for the optimal algorithm. However, even in this case the remaining t^1 term is incorrect. Yet, the error of the sixth moment at large t is optimized, since in this case the relative error is $O(t^{-2})$, whereas in all other cases the t^2 term is present and the error is $O(t^{-1})$.

In fact, this statement about the optimization of the error is true for *all* moments. In the case of the $(2m)$th moment, the two terms of the highest order in t are of order t^m and t^{m-1}. By analogy with the sixth moment, the only combinations of indices in $\langle\eta_{i_1}\ldots\eta_{i_{2m}}\rangle$ contributing to these terms are (1) those with m pairs of equal indices with no equality between pairs and (2) those with four equal indices and $m-2$ pairs of equal indices with no equality between these groups. For the first type, there are $(2m-1)!!$ ways to break the indices into pairs and $N!/(N-m)!$ ways to assign the values of the indices to these pairs, for a total of $(2m-1)!!N!/(N-m)!$ terms in the sum, each of which is $\langle\eta^2\rangle^m$. For the second type, there are $\binom{2m}{4}$ ways to choose the group of four indices, $(2m-5)!!$ ways to break the remaining indices into pairs, and $N!/(N-m+1)!$ ways to assign the values of the indices to all these groups, for a total of $\binom{2m}{4}(2m-5)!!N!/(N-m+1)!$ terms, each of which is $\langle\eta^4\rangle\langle\eta^2\rangle^{m-2}$. The $(2m)$th moment then is

$$\begin{aligned}\langle x_N^{2m}\rangle &= a^{2m}\left[\frac{(2m-1)!!N!}{(N-m)!}\langle\eta^2\rangle^m\right.\\ &\quad\left.+\frac{\binom{2m}{4}(2m-5)!!N!}{(N-m+1)!}\langle\eta^4\rangle\langle\eta^2\rangle^{m-2}+O(N^{m-2})\right]\\ &= a^{2m}\left[(2m-1)!!\left(N^m-\frac{m(m-1)}{2}N^{m-1}\right)\right.\\ &\quad\times(2p_{j-1\to j})^m+\binom{2m}{4}(2m-5)!!N^{m-1}(2p_{j-1\to j})^{m-1}\\ &\quad\left.+O(N^{m-2})\right]\\ &= a^{2m}\left[(2m-1)!!\left(\frac{2p_{j-1\to j}}{\tau}\right)^m t^m+(2m-1)!!m\right.\\ &\quad\times(m-1)2^{m-1}\frac{p_{j-1\to j}^{m-1}/6-p_{j-1\to j}^m}{\tau^{m-1}}t^{m-1}\\ &\quad\left.+O(t^{m-2})\right].\end{aligned} \tag{51}$$

We see again that the coefficient of the leading t^m term is correct whenever Eq. (11) is satisfied, and the subleading term vanishes when $p_{j-1\to j}=1/6$.

C. Moments of the FPT distribution

Consider again a particle diffusing in continuum space and starting at $x=0$ at time $t=0$. In the FPT problem [25], we consider two imaginary walls at $x=\pm b$ and find the first instance at which the particle reaches one of these walls. The corresponding time T is the FPT. The LMC analog of the FPT problem would be starting at site 0 at $t=0$ and determining the first time one of the sites $\pm N$ is reached. In a good LMC algorithm, the corresponding FPT distribution should match the continuum one for $b=Na$ as closely as possible.

Consider first the case $N=1$ (and thus $b=a$). In the LMC, the FPT then corresponds to the step at which the particle first moves out of site 0 (and into one of its neighboring sites ±1). This happens at mth step with probability

$$\pi_m = p_{j\to j}^{m-1}(1-p_{j\to j}). \tag{52}$$

Then the first moment of the FPT, or the mean FPT (MFPT), is

$$\langle T_1\rangle_d=\tau\sum_{m=1}^{\infty}m\pi_m=\frac{\tau}{1-p_{j\to j}}=\frac{a^2}{2D}, \tag{53}$$

where Eq. (12) was used, the subscript "1" denotes $N=1$, and the subscript "d" stands for "discrete." The second moment, or the mean-square FPT (MSFPT), is

$$\langle T_1^2\rangle_d=\tau^2\sum_{m=1}^{\infty}m^2\pi_m=\frac{\tau^2(1+p_{j\to j})}{(1-p_{j\to j})^2}=\frac{a^4}{4D^2}(1+p_{j\to j}). \tag{54}$$

On the other hand, the corresponding continuum results are [25]

$$\langle T_1\rangle_c=\frac{a^2}{2D}, \tag{55}$$

$$\langle T_1^2\rangle_c=\frac{5a^4}{12D^2}. \tag{56}$$

Note that while the results for the MFPT are always the same ($\langle T_1\rangle_d=\langle T_1\rangle_c$), those for the MSFPT only coincide for $p_{j\to j}=2/3$, which corresponds to the optimal algorithm. Note also that the MSFPT for the LMC [Eq. (54)] is the lowest for $p_{j\to j}=0$ (the "trivial" ordinary algorithm). In fact, in this case the MSFPT is simply the square of the MFPT; that is, the variance of the FPT is zero. The FPT is deterministic and is always equal to the time step of the algorithm. On the other hand, the optimal algorithm produces the correct variance of the FPT.

In fact, the matching of the first and second moments of the FPT for $N=1$ guarantees such matching for any other N and even for "asymmetric" FPT problems where the two walls are at different distances from the initial position. This can be seen from the following consideration. Consider a random walk in continuum space and map it onto a lattice walk by introducing sites at points $x=ja$ and making a jump of the lattice walk from site j to site $j\pm1$ when the continuum walk visits site

$j \pm 1$ for the first time after visiting site j. The probability of any particular sequence of visited sites in such a walk will be the same as in the LMC algorithms (either the ordinary or the optimal one), since the only requirement for obtaining the same probability is that the walk be unbiased. This means that the probability P_m to reach in m jumps a particular set of sites of interest starting from another specific site will always be the same for the walk obtained by direct mapping from the continuum walk and for any of the LMC approximations (where for the optimal LMC only the actual jumps are counted and not the steps where the particle stays put). On the other hand, the mean waiting times for a single jump are given by Eq. (53) for the LMC algorithms and by Eq. (55) for the walk obtained by mapping from the continuum walk; likewise, the mean-square waiting times are given by Eqs. (54) and (56), respectively. Therefore, if it is known that the first-passage number of jumps is m, then the MFPT is $m\langle T_1\rangle_{d,c}$ and the MSFPT is $m(m-1)\langle T_1\rangle_{d,c}^2 + m\langle T_1^2\rangle_{d,c}$, where the appropriate subscript d or c and the appropriate $p_{j\to j}$ are chosen depending on the walk. After averaging over all possible m we get for the MFPT

$$\langle T\rangle = \sum_m P_m m\langle T_1\rangle_{d,c} \tag{57}$$

and for the MSFPT we get

$$\langle T^2\rangle = \sum_m P_m\left[m(m-1)\langle T_1\rangle_{d,c}^2 + m\left\langle T_1^2\right\rangle_{d,c}\right]. \tag{58}$$

Given that P_m are the same for all three walks and since $\langle T_1\rangle_d = \langle T_1\rangle_c$ for all $p_{j\to j}$, the MFPT for both LMC algorithms is always the same as for the lattice walk obtained from the continuum walk and thus the same as for the continuum walk. This is also true for the MSFPT for the optimal LMC algorithm, as in that case $\langle T_1^2\rangle_d = \langle T_1^2\rangle_c$, but not for the ordinary LMC, as in that case this equality does not hold. Instead, for the ordinary LMC, using Eq. (58) and Eqs. (53)–(54) with $p_{j\to j} = 0$,

$$\begin{aligned}\langle T^2\rangle &= \sum_m P_m[m(m-1)(a^4/4D^2) + m(a^4/4D^2)]\\ &= \frac{a^4}{4D^2}\sum_m m^2 P_m = \frac{a^4}{4D^2}\langle m^2\rangle,\end{aligned} \tag{59}$$

where $\langle m^2\rangle$ is the mean-square first-passage number of jumps of the unbiased random walk. This is just the square of the (deterministic) interval between jumps (equal to τ) times $\langle m^2\rangle$. For the FPT problem where the particle starts at site 0 and the walls are at sites $\pm N$, $\langle m^2\rangle$ is [26]

$$\langle m^2\rangle = \frac{5N^4 - 2N^2}{3}, \tag{60}$$

so

$$\langle T^2\rangle = \frac{a^4}{12D^2}(5N^4 - 2N^2). \tag{61}$$

If $a = b/N$, then

$$\langle T^2\rangle = \frac{b^4}{12D^2}(5 - 2/N^2), \tag{62}$$

which coincides with the continuum result $5b^4/(12D^2)$ in the limit $N \to \infty$, but deviates for finite N. This is illustrated in Fig. 3(a), where the comparison to the optimal algorithm is made; for the latter, as mentioned, the exact result is recovered for all N. Note that the data in Fig. 3 are obtained for $b = 1$, which gives $a = 1/N$, and so $N = 1/a$ is simply the inverse mesh step.

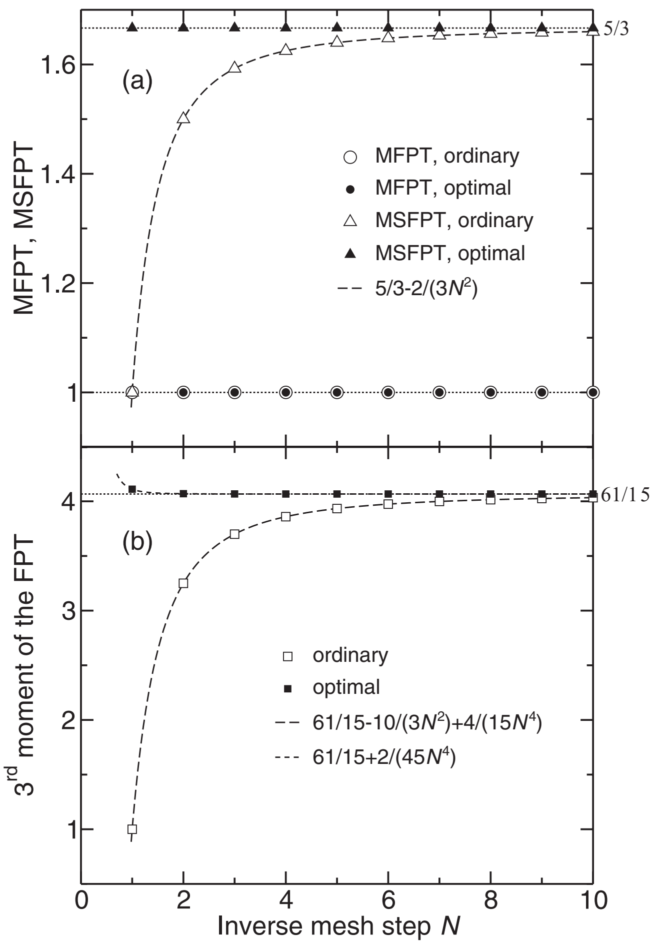

FIG. 3. For the ordinary Eqs. (1)–(3) and optimal Eqs. (26)–(28) LMC algorithms, the first two moments [MFPT and MSFPT; panel (a)] and the third moment [panel (b)] of the FPT that would be observed in the simulations of 1D diffusion with $D = 1/2$ starting at $x = 0$ at $t = 0$ and with the walls located at $x = \pm 1$ (i.e., $b = 1$), as a function of the inverse mesh step N. The data points are obtained by numerically iterating the corresponding master equations. For the optimal algorithm, the MFPT and MSFPT are independent of N and coincide with the continuum values (1 and 5/3, respectively, shown by the dotted lines). For the ordinary algorithm, the MFPT is likewise N-independent, but the MSFPT data lie on the curve given by Eq. (62) (dashed line). For the third moment, the dashed lines are perfect fits to the data, and the dotted line represents the continuum value.

In Fig. 3(b), we show the third moment of the FPT. These results have been obtained numerically, by iterating the corresponding master equations (the numerically exact method), which, of course, is much more accurate than would be achievable with particle-based LMC simulations. As expected, the results are now N-dependent in both cases, but the deviation from the asymptotic value is much smaller for the optimal algorithm and, in fact, is barely visible already for $N = 1$. While we have not derived the analytical results (this would be very tedious to do), the equations given in the figure provide essentially perfect fits to the data, and we believe that these are exact (with any deviations due entirely to numerical errors). It is seen that while the deviation from the continuum value is $O(1/N^2)$ for the ordinary algorithm, it is $O(1/N^4)$ for the optimal one, in full analogy with the higher moments of the spatial particle distribution (Sec. II B), where the leading term in the deviation from the continuum value likewise vanishes when the algorithm is optimized.

We can also obtain numerically, likewise by iterating the corresponding master equations, the whole FPT distributions for different algorithms and compare them to the continuum result. First of all, the immediate shortcoming of the ordinary algorithm is that for even N, the particle can only reach the boundaries at even steps, while for odd N, it can only reach the boundaries at odd steps. This means that the first-passage probability will vary wildly, alternating between zero and a nonzero value even for arbitrarily large N. To eliminate this, we plot only the nonzero values of the probabilities dividing them by two, in effect, averaging between the zero and nonzero values.

The continuum result for the probability density $r(t)$ of reaching the boundary at time t (or the first-passage *rate*) can be obtained by representing the solution of the continuum equation as a sum over Fourier components satisfying the absorbing boundary conditions, that is, equal to zero at $x = \pm b$, with the result (see, e.g., Ref. [27])

$$r(t) = \frac{\pi D}{b^2}\sum_{m=0}^{\infty}(-1)^m(2m+1)\exp[-(2m+1)^2\pi^2 Dt/4b^2]. \tag{63}$$

This series converges faster at large t. An alternative but equivalent series which converges more rapidly at small t results from representing the solution as a sum of Gaussians with alternating signs centered at $x = 2bm$ (where m are integer):

$$\begin{aligned} r(t) &= \frac{b}{2\sqrt{\pi D t^3}}\sum_{m=-\infty}^{\infty}(-1)^m(2m+1) \\ &\quad\times \exp[-(2m+1)^2 b^2/4Dt] \\ &= \frac{b}{\sqrt{\pi D t^3}}\sum_{m=0}^{\infty}(-1)^m(2m+1)\exp[-(2m+1)^2 b^2/4Dt]. \end{aligned} \tag{64}$$

The function $r(t)$ is thus exponentially small for small t, reaches a maximum at some intermediate time and decays exponentially for large t (the thick solid line in Fig. 4).

Take for simplicity $b = 1$ and $D = 1/2$, which gives the MFPT $\langle T\rangle = 1$. In this case, $a = 1/N$ and the time step is $1/N^2$ for the ordinary algorithm and $1/(3N^2)$ for the optimal algorithm. The comparison of the ordinary and optimal algorithms (with different degrees of discretization N) with the exact result is made in Fig. 4. For $N = 1$, the FPT is deterministic and always equals 1 for the ordinary algorithm, as the particle always moves to site $+1$ or -1 during the first step. For the optimal algorithm, on the other hand, the particle can move to one of the sites ± 1 at any step with probability 1/3; therefore, the FPT distribution decays exponentially. While the maximum present in the continuum distribution is not reproduced, the continuum distribution is matched quite closely for large t. Already for $N = 2$ the optimal algorithm

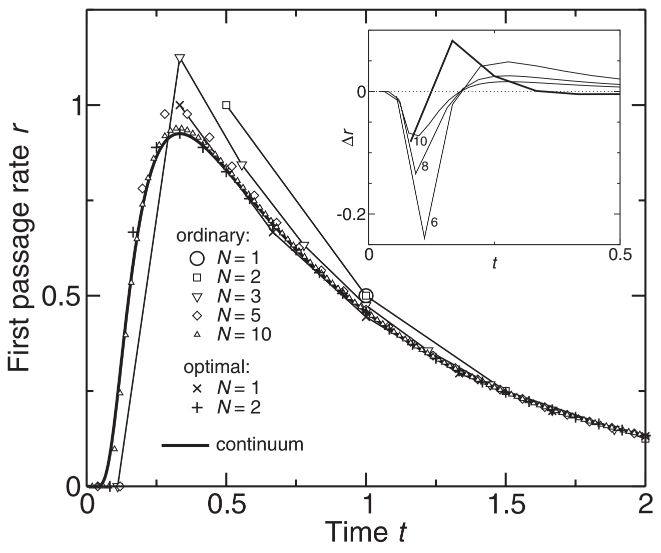

FIG. 4. The continuum first-passage rate for 1D diffusion with $D = 1/2$ starting at $x = 0$ at $t = 0$ and with the walls located at $x = \pm b = \pm 1$ (thick solid line), compared to an analogous quantity for the LMC algorithms (symbols, in some cases connected by thin lines for guidance to the eye). This first-passage rate analog is calculated as the probability of first passage at a particular step divided by the time step of the algorithm. For the ordinary algorithm, the even steps for odd N and the odd steps for even N, at which the passage probability is always zero, are discarded and the rates for remaining time steps are divided by two. The inset shows the difference between the LMC rate and the continuum rate, for the optimal algorithm with $N = 2$ (thick line) and for the ordinary algorithm with $N = 6$, 8, 10 (thin lines), as indicated by the labels next to the lines.

reproduces the continuum curve very accurately. With the ordinary algorithm, a similar accuracy is not achieved until $N \approx 8$–10, as the inset of Fig. 4 shows, where the difference between the LMC results and the continuum values is plotted. Very good results at large t obtained with the optimal algorithm would not be surprising, as it is designed specifically to be as accurate as possible in this case; however, we see that it also works very well for moderate t. In terms of the computational effort, the ordinary algorithm with $N = 8$ takes $N^2 = 64$ steps on average to reach the walls, whereas the optimal algorithm with $N = 2$ takes $3N^2 = 12$ steps on average; thus, the speedup is a factor of at least 5. The advantage is even larger when the numerically exact approach based on solving numerically the master equations is used, since decreasing the mesh step increases the number of such equations and not only the number of time steps.

As mentioned above, the accuracy of the optimal algorithm is expected to be particularly good at large times. In the limit of large t, the first-passage rate decays exponentially:

$$r(t) \simeq \gamma \exp(-\beta t). \tag{65}$$

It is therefore convenient to compare the algorithms by finding the rate of the exponential decay β and the prefactor γ and comparing to the continuum values. By looking at the first term of Eq. (63), for $D = 1/2$ and $b = 1$ we get the continuum decay rate of $\beta_c = \pi^2/8$ and the prefactor of $\gamma_c = \pi/2$. For the LMC algorithms, these quantities can be derived as well (see Appendix). For the ordinary algorithm, the decay rate is

$$\beta_{\text{ord}} = -N^2 \ln \cos\left(\frac{\pi}{2N}\right), \tag{66}$$

and the prefactor is

$$\gamma_{\text{ord}} = N \tan\left(\frac{\pi}{2N}\right). \tag{67}$$

For the optimal algorithm, they are, respectively,

$$\beta_{\text{opt}} = -3N^2 \ln\left[\frac{2}{3} + \frac{1}{3}\cos\left(\frac{\pi}{2N}\right)\right] \tag{68}$$

and

$$\gamma_{\text{opt}} = \frac{N \sin(\pi/2N)}{2/3 + (1/3)\cos(\pi/2N)}. \tag{69}$$

These quantities are plotted in Fig. 5. For the optimal algorithm, even for $N = 1$ the results are quite close to the continuum value ($\beta_{\text{opt}} = 1.216\ldots$ vs $\pi^2/8 = 1.233\ldots$ for the rate and $\gamma_{\text{opt}} = 1.5$ vs $\pi/2 = 1.570\ldots$ for the prefactor). For the ordinary algorithm, similar accuracy is achieved for $N =$ 4–5; note that for $N = 1$ the decay rate diverges, as the FPT is deterministic. For $N = 2$ the results for the optimal algorithm are nearly indistinguishable from the continuum values on the scale of the plot and are closer than those for the ordinary algorithm with $N = 10$. Note that the long-time approximation is applicable even for quite small t: For example, for $t = 1$ (equal to the MFPT), the full sum in Eq. (63) is $0.457\,365\ldots$ and its first term is $0.457\,436\ldots$, so the difference is only $\approx -7 \times 10^{-5}$.

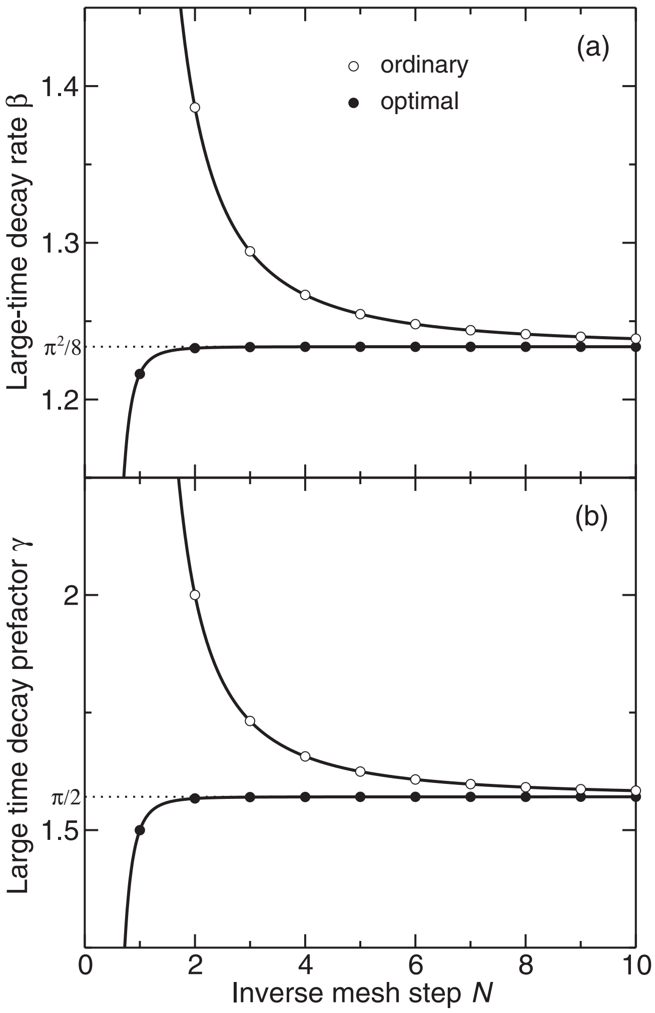

FIG. 5. The large-time rate of exponential decay (a) and the corresponding prefactor (b) for the first-passage rate to the walls located at $x = \pm 1$ starting at $x = 0$ in simulations of 1D diffusion using the ordinary (open circles) and the optimal (solid circles) LMC algorithms. The curves are the analytical expressions Eqs. (66)–(69). The continuum limit values are indicated by the dotted lines.

III. TWO DIMENSIONS

In 2D, the continuum unbiased diffusion equation is

$$\frac{\partial n(x,y,t)}{\partial t} = D\left(\frac{\partial^2 n(x,y,t)}{\partial x^2} + \frac{\partial^2 n(x,y,t)}{\partial y^2}\right). \tag{70}$$

Its general solution can be written in a form similar to Eq. (15):

$$n(x,y,t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} C(k_x,k_y)\exp[ik_x x + ik_y y - \alpha_c(k_x,k_y)t]\,dk_x dk_y. \tag{71}$$

The dispersion relation is analogous as well:

$$\alpha_c(k_x,k_y) = D\left(k_x^2 + k_y^2\right). \tag{72}$$

As for the design of the corresponding LMC algorithm, we first note that different types of lattices can be chosen. We consider the simplest choice, the square lattice, with the lattice constant still denoted a. The lattice sites can be denoted by pairs of integer numbers, (i,j). If only moves along the Cartesian axes are allowed, there are five different probabilities of moves: $p_{(i,j)\to(i+1,j)} \equiv p_{+x}$ and $p_{(i,j)\to(i-1,j)} \equiv p_{-x}$ for the moves along the x axis in the positive and negative directions, respectively; $p_{(i,j)\to(i,j+1)} \equiv p_{+y}$ and $p_{(i,j)\to(i,j-1)} \equiv p_{-y}$ for the moves along the y axis; and the probability of staying on the same site, $p_{(i,j)\to(i,j)} \equiv p_0$. The master equation for the mean particle number $n_{(j,l)}$ at a particular site (j,l) is

$$n_{(j,l)}(t+\tau) = p_0 n_{(j,l)}(t) + p_{+x} n_{(j-1,l)}(t) + p_{-x} n_{(j+1,l)}(t) + p_{+y} n_{(j,l-1)}(t) + p_{-y} n_{(j,l+1)}(t). \tag{73}$$

In the ordinary 2D LMC algorithm, the particle moves at every step, all moves are equiprobable, and since the particle only moves along a particular axis at every second step on average, the time step has to be equal to 1/2 that in the ordinary 1D algorithm:

$$p_0 = 0, \tag{74}$$

$$p_{\pm x} = p_{\pm y} = 1/4, \tag{75}$$

$$\tau = \frac{a^2}{4D}. \tag{76}$$

However, in general, as in 1D, we allow $p_0 \neq 0$.

The general solution of the system of master equations for all sites is (again, similarly to 1D)

$$n_{(j,l)}(t) = \int_{-\pi/a}^{\pi/a}\int_{-\pi/a}^{\pi/a} C(k_x,k_y)\exp[ik_x aj + ik_y al - \alpha_d(k_x,k_y)t]\,dk, \tag{77}$$

with

$$\alpha_d(k_x,k_y) = -\frac{1}{\tau}\ln[p_0 + p_{+x}\exp(-ik_xa) + p_{-x}\exp(ik_xa) + p_{+y}\exp(-ik_ya) + p_{-y}\exp(ik_ya)]. \quad (78)$$

As in 1D, the goal is to choose the probabilities and the time step τ so that α_c and α_d match as closely as possible for small k_x and k_y. While it is possible to match all coefficients of the Taylor expansion up to and including the quadratic terms (which, as in 1D, guarantees the correct diffusion rate), matching all quartic terms, like we did in 1D, is obviously impossible. This is because the form of Eq. (78) is such that there are always terms $\propto k_x^2$ and $\propto k_y^2$, but no quartic term $\propto k_x^2k_y^2$ *under the logarithm*. When the logarithm is expanded, this will necessarily produce such a quartic term. At the same time, no quartic terms are present in the expression for α_c [Eq. (72)] that we need to match. This reasoning is valid for *any* set of moves, even if moves to second neighbors and more distant sites are present, *as long as* all moves are along one of the axes. On the other hand, moves along both axes *simultaneously* will produce a term $\propto k_x^2k_y^2$ under the logarithm with a coefficient that can be chosen so that in the final expansion this term vanishes. The simplest moves of this type are the four "diagonal" moves whereby the particle moves by one lattice constant along the x axis and simultaneously along the y axis (Fig. 6). This introduces four additional probabilities: $p_{(i,j)\to(i+1,j+1)} \equiv p_{+x,+y}$, $p_{(i,j)\to(i+1,j-1)} \equiv p_{+x,-y}$, $p_{(i,j)\to(i-1,j+1)} \equiv p_{-x,+y}$, and $p_{(i,j)\to(i-1,j-1)} \equiv p_{-x,-y}$. The new master equation is

$$n_{(j,l)}(t+\tau) = p_0n_{(j,l)}(t) + p_{+x}n_{(j-1,l)}(t) + p_{-x}n_{(j+1,l)}(t) + p_{+y}n_{(j,l-1)}(t) + p_{-y}n_{(j,l+1)}(t)$$

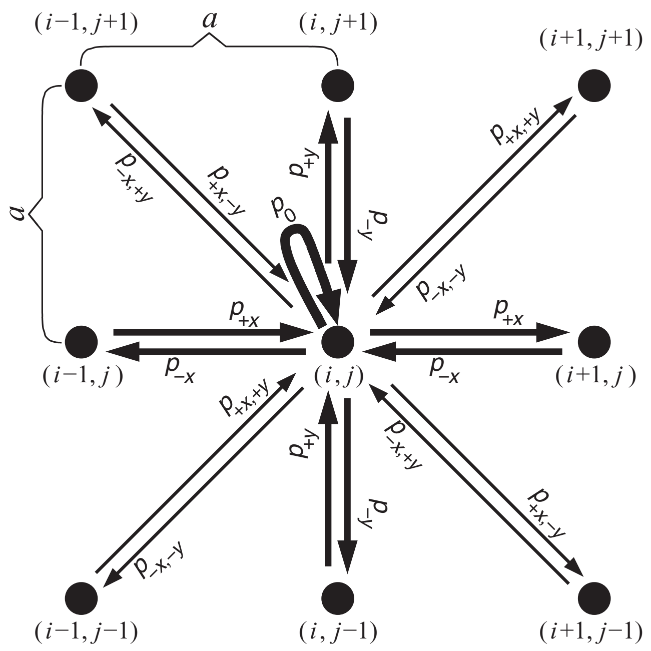

FIG. 6. A schematic of the general 2D LMC algorithm considered in this paper. Only moves to and from site (i,j) are shown. In addition to the traditional axial moves (arrows of medium thickness), diagonal moves (thin arrows) are introduced; also, there is a probability to stay put (represented by the thickest arrow). Notation for the probabilities of the moves is given.

$$+ p_{+x,+y}n_{(j-1,l-1)}(t) + p_{+x,-y}n_{(j-1,l+1)}(t) + p_{-x,+y}n_{(j+1,l-1)}(t) + p_{-x,-y}n_{(j+1,l+1)}(t). \quad (79)$$

The dispersion relation now becomes

$$\begin{aligned}\alpha_d(k_x,k_y) = -\frac{1}{\tau}\ln[&p_0 + p_{+x}\exp(-ik_xa) + p_{-x}\exp(ik_xa)\\ &+ p_{+y}\exp(-ik_ya) + p_{-y}\exp(ik_ya)\\ &+ p_{+x,+y}\exp(-ik_xa - ik_ya)\\ &+ p_{+x,-y}\exp(-ik_xa + ik_ya)\\ &+ p_{-x,+y}\exp(ik_xa - ik_ya)\\ &+ p_{-x,-y}\exp(ik_xa + ik_ya)]. \end{aligned} \quad (80)$$

Matching the k^0 term,

$$p_0 + p_{+x} + p_{-x} + p_{+y} + p_{-y} + p_{+x,+y} + p_{+x,-y} + p_{-x,+y} + p_{-x,-y} = 1, \quad (81)$$

which is again just the normalization statement. Matching the linear terms,

$$p_{+x} + p_{+x,+y} + p_{+x,-y} - p_{-x} - p_{-x,+y} - p_{-x,-y} = 0, \quad (82)$$

$$p_{+y} + p_{+x,+y} + p_{-x,+y} - p_{-y} - p_{+x,-y} - p_{-x,-y} = 0. \quad (83)$$

These ensure that the average velocities are zero in both directions (no bias). Taking Eqs. (81)–(83) into account, we can expand under the logarithm to obtain

$$\begin{aligned}\alpha_d(k_x,k_y) = -\frac{1}{\tau}\ln\big[&1 - (C_{xx}/2)a^2k_x^2 - (C_{yy}/2)a^2k_y^2\\ &+ C_{xy}a^2k_xk_y - i(C_{xxy}/2)a^3k_x^2k_y\\ &- i(C_{xyy}/2)a^3k_xk_y^2 + (C_{xx}/24)a^4k_x^4\\ &+ (C_{yy}/24)a^4k_y^4 - (C_{xy}/6)a^3k_x^3k_y\\ &- (C_{xy}/6)a^3k_xk_y^3 + (C_{xxyy}/4)a^4k_x^2k_y^2 + O(k^5)\big],\end{aligned} \quad (84)$$

where

$$\begin{aligned}C_{xx} &= p_{+x} + p_{-x} + p_{+x,+y} + p_{+x,-y} + p_{-x,+y} + p_{-x,-y}\\ &= 2(p_{+x} + p_{+x,+y} + p_{+x,-y}),\end{aligned} \quad (85)$$

$$\begin{aligned}C_{yy} &= p_{+y} + p_{-y} + p_{+x,+y} + p_{+x,-y} + p_{-x,+y} + p_{-x,-y}\\ &= 2(p_{+y} + p_{+x,+y} + p_{-x,+y}),\end{aligned} \quad (86)$$

$$C_{xy} = -p_{+x,+y} + p_{+x,-y} + p_{-x,+y} - p_{-x,-y}, \quad (87)$$

$$C_{xxy} = -p_{+x,+y} + p_{+x,-y} - p_{-x,+y} + p_{-x,-y}, \quad (88)$$

$$C_{xyy} = -p_{+x,+y} - p_{+x,-y} + p_{-x,+y} + p_{-x,-y}, \quad (89)$$

$$C_{xxyy} = p_{+x,+y} + p_{+x,-y} + p_{-x,+y} + p_{-x,-y}. \quad (90)$$

Expanding the logarithm, we now get

$$\begin{aligned}\alpha_d(k_x,k_y) = \frac{1}{\tau}\big[&(C_{xx}/2)a^2k_x^2 + (C_{yy}/2)a^2k_y^2 - C_{xy}a^2k_xk_y\\ &+ i(C_{xxy}/2)a^3k_x^2k_y + i(C_{xyy}/2)a^3k_xk_y^2\\ &+ \big(C_{xx}^2/8 - C_{xx}/24\big)a^4k_x^4\end{aligned}$$

$$+\left(C_{yy}^2/8 - C_{yy}/24\right)a^4k_y^4$$
$$+(C_{xy}/6 - C_{xx}C_{xy}/2)a^3k_x^3k_y$$
$$+(C_{xy}/6 - C_{yy}C_{xy}/2)a^3k_xk_y^3$$
$$+\left(C_{xx}C_{yy}/4 - C_{xxyy}/4 + C_{xy}^2/2\right)$$
$$\times a^4k_x^2k_y^2 + O(k^5)\big]. \tag{91}$$

Comparing to Eq. (72), we obtain the following equations:

$$C_{xx}a^2/2\tau = D, \tag{92}$$
$$C_{yy}a^2/2\tau = D, \tag{93}$$
$$C_{xy}/\tau = 0, \tag{94}$$
$$C_{xxy}/\tau = 0, \tag{95}$$
$$C_{xyy}/\tau = 0, \tag{96}$$
$$\left(C_{xx}^2/8 - C_{xx}/24\right)/\tau = 0, \tag{97}$$
$$\left(C_{yy}^2/8 - C_{yy}/24\right)/\tau = 0, \tag{98}$$
$$(C_{xy}/6 - C_{xx}C_{xy}/2)/\tau = 0, \tag{99}$$
$$(C_{xy}/6 - C_{yy}C_{xy}/2)/\tau = 0, \tag{100}$$
$$\left(C_{xx}C_{yy}/4 - C_{xxyy}/4 + C_{xy}^2/2\right)/\tau = 0. \tag{101}$$

From Eqs. (97) and (98), $C_{xx} = C_{yy} = \frac{1}{3}$ ($C_{xx} = 0$ and $C_{yy} = 0$ are not solutions, as in that case $\tau = 0$). From this, using Eq. (92) or (93), we can compute the time step:

$$\tau = \frac{a^2}{6D}. \tag{102}$$

Interestingly, this optimal time step is equal to the one we obtained previously for the 1D problem Eq. (28). From Eq. (94), $C_{xy} = 0$. By inserting the values of C_{xx}, C_{yy}, and C_{xy} in Eq. (101), we obtain $C_{xxyy} = 1/9$. From Eqs. (95) and (96), $C_{xxy} = C_{xyy} = 0$. Equations (99) and (100) are then satisfied automatically. Thus, all coefficients C are known. Using the expressions for C [Eqs. (85)–(90)], we obtain linear equations for the probabilities. Together with Eqs. (82) and (83), this forms a linear system of eight equations with eight unknowns that has a unique solution,

$$p_{+x} = p_{-x} = p_{+y} = p_{-y} = 1/9, \tag{103}$$
$$p_{+x,+y} = p_{+x,-y} = p_{-x,+y} = p_{-x,-y} = 1/36. \tag{104}$$

Finally, from Eq. (81),

$$p_0 = 4/9. \tag{105}$$

The set of parameters given by Eqs. (102)–(105) defines an unbiased 2D LMC algorithm that is "optimal" for the particular set of moves considered here, since it is the only set of parameters that reproduces the continuum dispersion relation up to $O(k^5)$, the same accuracy as for the optimal 1D algorithm considered in Sec. II. (In both cases, the k^5 terms were not required to vanish explicitly, but are zero automatically, as are all other odd-order terms.) The set of moves is itself optimal in the sense that it is the smallest set of shortest-ranged moves that ensures such accuracy. We note that the probability of staying put is reduced from 2/3 to 4/9, and that the diagonal moves have small but finite probabilities.

Note that the probabilities (103)–(105) can be obtained as products of 1D optimal probabilities Eqs. (26) and (27): The probability of every 2D move is the product of the probabilities of the 1D moves that it "consists of." For example, the 2D diagonal move $(+x, +y)$ can be thought of as consisting of two 1D moves, $j-1 \to j$, one in the x direction and one in the y direction, and indeed, $p_{+x,+y} = p_{j-1\to j} \times p_{j-1\to j}$. Likewise, a move along one axis, say, $-x$, in 2D LMC can be represented as a move by one site in that direction and staying put in the orthogonal direction, which gives the correct equality $p_{-x} = p_{j+1\to j} \times p_{j\to j}$. For the probability of staying put, by similar reasoning $p_0 = p_{j\to j} \times p_{j\to j}$, which is also correct. Thus, the 2D optimal algorithm can be considered as the *direct product* of the 1D optimal algorithm with itself, in the sense that if the 2D probabilities are arranged in a matrix, this matrix is the direct product of the vectors of the 1D probabilities:

$$\begin{pmatrix} p_{-x,-y} & p_{-x} & p_{-x,+y} \\ p_{-y} & p_0 & p_{+y} \\ p_{+x,-y} & p_{+x} & p_{+x,+y} \end{pmatrix} = \begin{pmatrix} p_{j+1\to j}\times p_{j+1\to j} & p_{j+1\to j}\times p_{j\to j} & p_{j+1\to j}\times p_{j-1\to j} \\ p_{j\to j}\times p_{j+1\to j} & p_{j\to j}\times p_{j\to j} & p_{j\to j}\times p_{j-1\to j} \\ p_{j-1\to j}\times p_{j+1\to j} & p_{j-1\to j}\times p_{j\to j} & p_{j-1\to j}\times p_{j-1\to j} \end{pmatrix}$$
$$= \begin{pmatrix} p_{j+1\to j} \\ p_{j\to j} \\ p_{j-1\to j} \end{pmatrix} \otimes (p_{j+1\to j} \quad p_{j\to j} \quad p_{j-1\to j}). \tag{106}$$

The fact that the direct product algorithm should work as well in 2D as its "factors" do in 1D can be seen directly from Eq. (80): If the probabilities entering that equation are the products of the 1D probabilities, then the expression under the logarithm can be represented as a product of two expressions, $f(k_x)f(k_y)$, where $f(k) = p_{j\to j} + p_{j-1\to j}\exp(-ika) + p_{j+1\to j}\exp(ika)$ is the expression under the logarithm in the 1D analog of Eqs. (80) and (19). On the other hand, if the probabilities in $f(k)$ are chosen optimally, then, according to Eq. (19) and the considerations following it, $\ln[f(k)] = -\tau Dk^2 + O(k^6)$; therefore, in Eq. (80)

$$\alpha_d(k_x,k_y) = -\frac{1}{\tau}\ln[f(k_x)f(k_y)]$$
$$= \left[Dk_x^2 + O\left(k_x^6\right)\right] + \left[Dk_y^2 + O\left(k_y^6\right)\right], \tag{107}$$

which matches the continuum dispersion relation Eq. (72) up to and including the fifth-order terms. (Note, by the way, that there are no cross terms containing both k_x and k_y in *any* order and also, as in 1D, no odd-order terms.) Note that this consideration assumes that the time step in 2D is the same as in 1D, which is indeed the case.

Moreover, the following general statement can be made. Suppose the general solution of some continuum equation in 2D can be represented as

$$n(x,y,t) = \int C(k_x,k_y) g_{k_x}(x,t) g_{k_y}(y,t) dk_x dk_y, \quad (108)$$

where $g_k(x,t)$ are solutions (depending on the parameter k) of some 1D equation for which there exists a LMC algorithm. Then a 2D algorithm that is the direct product of the 1D LMC algorithm with itself is as good for the 2D equation as the 1D algorithm is for the 1D equation. This can be checked by substituting $g_{k_x}(x,t) g_{k_y}(y,t)$ into the master equation (79) (or its generalization, if, e.g., more distant sites are involved) [much like we did in Eq. (33)], which makes both sides of that equation products of the corresponding sides of the 1D master equation (17) or its generalization, in which g_{k_x} and g_{k_y} are likewise substituted. In view of this, we could have immediately "guessed" the correct 2D algorithm by looking at the solution of the 2D diffusion equation [Eq. (71), from which $g_k(x,t) = \exp(ikx - Dk^2t)$], without the lengthy derivation that followed. That derivation was still useful, however, as it proved both the uniqueness of the solution given the set of moves and the fact that the diagonal moves were necessary.

Note that from the fact that the 2D optimal algorithm is the direct product of the 1D optimal algorithm with itself, it follows that projecting all moves of the 2D optimal algorithm along the x axis and keeping the time step the same produces the 1D optimal algorithm. Then, if we take a solution of the master equation (79) and "project" it onto the x axis by summing up over columns of sites (i.e., $n_i = \sum_j n_{(i,j)}$), the result will be a solution of the master equation (17) for the 1D optimal algorithm. Similarly, in the continuum case a 2D solution can be projected onto the x axis by integrating over y and the same statement can be made. It is easy to see then that the second- and fourth-order moments $\langle x^2 \rangle$ and $\langle x^4 \rangle$ [where all particles start at site (0,0)] are still correct at all times in the 2D optimal algorithm, since these moments do not change when the projecting is done. Arguing similarly for the y axis, we can prove the same for $\langle y^2 \rangle$ and $\langle y^4 \rangle$. Note, though, that there is also one fourth-order moment in 2D that involves both the x and the y axes, namely, $\langle x^2y^2 \rangle$. To easily calculate this moment, note that if at time $t = 0$ the 2D particle distribution can be represented as a product of two 1D distributions and this 2D distribution evolves according to the optimal algorithm, then it is also representable as a similar product at later times, with the two factors evolving according to the 1D optimal algorithm. That is,

$$n_{(i,j)}(0) = n_i(0)n_j(0) \Rightarrow n_{(i,j)}(t) = n_i(t)n_j(t). \quad (109)$$

This can be proved by induction, by assuming that the 2D distribution is representable as a product at time t and proving this for time $t + \tau$. [Note, by the way, that Eq. (109) does not hold for the 2D and 1D ordinary algorithms.] Similarly, in the continuum case

$$n(x,y,0) = n(x,0)n(y,0) \Rightarrow n(x,y,t) = n(x,t)n(y,t). \quad (110)$$

Then, since the initial distribution where all particles are concentrated at the origin is representable in this way, it is easy to see that in both the continuum case and for the optimal algorithm,

$$\langle x^2y^2 \rangle = \langle x^2 \rangle \langle y^2 \rangle, \quad (111)$$

and thus this mixed moment is the same in both cases. All other moments involving two axes are likewise products of 1D moments and therefore some sixth-order ($\langle x^4y^2 \rangle$ and $\langle x^2y^4 \rangle$) and even eighth-order ($\langle x^4y^4 \rangle$) moments are correct as well, but, of course, other sixth-order moments ($\langle x^6 \rangle$ and $\langle y^6 \rangle$) are incorrect.

For the ordinary algorithm given by Eqs. (74)–(76), projecting along the x axis gives a 1D LMC algorithm with the probability of staying put equal to 1/2 [Eqs. (11) and (12) with $\tau = a^2/4D$]; that is, it is intermediate between the ordinary and the optimal 1D algorithms. Since Eqs. (11) and (12) are still obeyed, the second moment $\langle x^2 \rangle$ is still correct for this 1D algorithm and thus $\langle x^2 \rangle$ and $\langle y^2 \rangle$ are correct for the 2D ordinary algorithm. However, the fourth-order moments $\langle x^4 \rangle$, $\langle y^4 \rangle$ and $\langle x^2y^2 \rangle$ are incorrect, the latter because the considerations above for the optimal algorithm do not apply and so $\langle x^2y^2 \rangle \neq \langle x^2 \rangle \langle y^2 \rangle$. In fact, it is obvious that $\langle x^2y^2 \rangle$ cannot be correct after the first step, because in the ordinary algorithm the particle only moves along x or along y at any step, but not in both directions at the same time, so after the first step either $x = 0$, or $y = 0$, and $\langle x^2y^2 \rangle = 0$. At large t, we have checked numerically that the relative error is $O(1/t)$, just as for $\langle x^4 \rangle$ and $\langle y^4 \rangle$; on the other hand, for the optimal algorithm even for higher-order moments that are not exact, the error is always $O(1/t^2)$.

Unfortunately, the statements about the moments of the FPT distribution (Sec. II C) do not generalize in the same way for the optimal algorithm. First of all, it is not immediately clear how to establish the correspondence between the discrete and continuum first-passage problems. In the discrete case, one is given a set of sites, the first passage to which is investigated. In the continuum case, one needs a closed curve, but there are multiple ways in which a curve can be drawn between a given set of sites. Apparently, the most straightforward case is that of a rectangular region with all four segments of the boundary lying along rows and columns of sites. In particular, one can consider a square region with the boundaries at $x = \pm 1$ and $y = \pm 1$ and compare to a set of discrete problems with $a = 1/N$ for all positive integer N, which is the closest analog of the situation we considered in 1D. It can be checked numerically for the optimal algorithm that not only the MSFPT, but even the MFPT is not reproduced correctly in the discrete case for finite N. However, for all N both moments are much more accurate than those obtained with the ordinary algorithm (results not shown). In fact, for large N, the error is $O(1/N^4)$ for the optimal algorithm and $O(1/N^2)$ for the ordinary algorithm. For small and moderate N, the difference between the algorithms is particularly striking for the MSFPT. The full FPT distribution is also very accurate for the optimal algorithm, with the results similar to those shown in Fig. 4 for 1D, although the ordinary algorithm works better than it did in the 1D case, in particular, the first-passage probability no longer alternates between zero and a nonzero value (but still, this probability goes up and down, being higher at odd steps for odd N and at even steps for even N).

IV. THREE AND MORE DIMENSIONS

In 3D, first of all, we can use the same arguments as in 2D to prove that the 3D algorithm constructed from the "direct product" of the 1D algorithms will reproduce the dispersion relation with the same precision, that is, up to $O(k^5)$. This 3D algorithm involves four types of moves: in addition to moves along one of the Cartesian directions, diagonal moves along two directions, and staying put, the new type of move in 3D is a move along all three directions simultaneously. The probabilities of the moves are

$$p_0 = (2/3)^3 = 8/27, \tag{112}$$

$$p_{\pm x} = p_{\pm y} = p_{\pm z} = (2/3)^2 \times (1/6) = 2/27, \tag{113}$$

$$p_{\pm x,\pm y} = p_{\pm y,\pm z} = p_{\pm x,\pm z} = (2/3) \times (1/6)^2 = 1/54, \tag{114}$$

$$p_{\pm x,\pm y,\pm z} = (1/6)^3 = 1/216, \tag{115}$$

and the time step is the same as in 1D and 2D:

$$\tau = \frac{a^2}{6D}. \tag{116}$$

However, unlike in 2D, this *direct product algorithm* is not the only way to satisfy the dispersion relation up to $O(k^5)$. For instance, there is no need to introduce moves simultaneously along all three directions. Just like the moves along two directions are used to adjust the term $\propto k_x^2 k_y^2$, the moves along three directions would be useful to adjust the term $\propto k_x^2 k_y^2 k_z^2$, but this term is $O(k^6)$, and the terms of this order cannot all be made to vanish simultaneously anyway, since there are not enough different moves (i.e., not enough free parameters). Therefore, we can retain just the moves that we used to optimize the 2D algorithm, that is, those along either one or two directions, and the resulting equations are very similar. In particular, the analogs of Eqs. (80), (84), and (91) for α_d are obtained simply by adding the analogous terms involving the z axis. This is also true for the analogs of Eqs. (81), (82), and (83), to which a similar equation obtained from equating the coefficient of k_z to zero is added. Expressions for the coefficients C in terms of the probabilities p [the analogs of Eqs. (85)–(90)], as well as those for the new coefficients involving the z axis (C_{zz}, C_{xz}, C_{yz}, C_{xxz}, C_{yyz}, C_{xzz}, C_{yzz}, C_{xxzz}, C_{yyzz}), are also built by analogy. The equations for C [Eqs. (92)–(101)] are still valid, and the analogous equations involving the z axis are added to them. From these equations, $C_{xx} = C_{yy} = C_{zz} = 1/3$, $C_{xxyy} = C_{xxzz} = C_{yyzz} = 1/9$, and all other C are zero; the time step is again the same as in 1D and 2D, as well as in the direct product algorithm in 3D; that is, it is given by Eq. (116). Finally, for the probabilities we obtain

$$p_{+x} = p_{-x} = p_{+y} = p_{-y} = p_{+z} = p_{-z} = 1/18, \tag{117}$$

$$\begin{aligned} p_{+x,+y} &= p_{+x,-y} = p_{-x,+y} = p_{-x,-y} = p_{+x,+z} = p_{+x,-z} \\ &= p_{-x,+z} = p_{-x,-z} = p_{+y,+z} = p_{+y,-z} = p_{-y,+z} \\ &= p_{-y,-z} = 1/36, \end{aligned} \tag{118}$$

$$p_0 = 1/3. \tag{119}$$

Projecting the moves of either this algorithm or the direct product algorithm Eqs. (112)–(116) onto one of the axes produces the 1D optimal unbiased algorithm and projecting onto one of the planes (xy, yz, or xz) produces the 2D optimal unbiased algorithm. Also, as in 2D, we can "project" the *solution* onto one of the axes by summing over the remaining two indices and the resulting 1D distribution will be a solution of the 1D optimal algorithm. Moreover, we can also "project" the solution onto one of the planes and obtain a solution of the 2D optimal algorithm. Noting that all nonzero moments of the particle distribution of order four and lower [where all particles start at site (0,0,0)] involve only one ($\langle x^2\rangle$, $\langle y^2\rangle$, $\langle z^2\rangle$, $\langle x^4\rangle$, $\langle y^4\rangle$, $\langle z^4\rangle$) or two ($\langle x^2y^2\rangle$, $\langle y^2z^2\rangle$, $\langle x^2z^2\rangle$) axes, all of these moments will be the same as in 2D and will therefore remain correct. As in 2D, some of the higher-order moments involving two axes are also correct. In addition, for the direct product algorithm, the moments involving three axes in which none of the coordinates enter with the power higher than four will be correct as well (the lowest-order moment of this type is the sixth-order $\langle x^2y^2z^2\rangle$). Note that such moments cannot be correct in the other algorithm, as there are no simultaneous moves in three directions, so after the first step these moments are zero.

We now have two different "optimal" LMC algorithms for 3D. The second algorithm is the only possible one without moves in three directions; however, if these moves are allowed, the first algorithm is but one example. This direct product algorithm, in a way, has an advantage that one additional sixth-order ($k_x^2k_y^2k_z^2$) and some additional higher-order terms in the dispersion relation vanish; in fact, it follows from a consideration analogous to that of the previous section that *all* cross terms containing at least two of the three variables k_x, k_y, and k_z are zero *in all orders* [28]. However, since other sixth-order terms (k_x^6, k_y^6, and k_z^6) do not vanish and given that, generally speaking, all sixth-order terms are expected to be of about the same magnitude, the advantage of the direct product algorithm is perhaps not very significant. Note that fifth-order terms (as well as all other odd-order terms, in fact, even those odd in any of the variables) vanish in both algorithms. On the other hand, the advantage of the algorithm without moves in three directions is that it is somewhat more short-ranged and slightly easier to program, as there are fewer moves. There may also be CPU time savings, in particular, for the numerically exact variant of the algorithm, as the master equation that needs to be solved iteratively will contain fewer terms, as well as, perhaps to a lesser extent, for the particle-based version, as one needs to choose between fewer moves.

We note that in the 3D case, the time step of either of the two "optimal" algorithms we have considered, $\tau = a^2/6D$, is also the time step of the nonoptimal but straightforward "ordinary" algorithm where the particle moves at each step along one of the six Cartesian directions. This is unlike in 1D and 2D, where the analogous straightforward algorithms have a larger time step and thus are more computationally efficient (as we discussed in detail for 1D). Note, however, that compared to the straightforward algorithm, we now use an extended set of moves including longer-range diagonal moves that allow an even larger time step when optimal accuracy is not required. That a larger time step is possible with the expanded set of moves is particularly obvious in 2D, where the algorithm with diagonal moves *only* is equivalent to the straightforward algorithm with the mesh step of $\sqrt{2}$ times the original mesh step (and the axes rotated by 45°) and thus the corresponding time step is twice that of the original straightforward algorithm.

It is also interesting to note that the projection of the ordinary 3D algorithm onto one of the axes coincides "accidentally" with the *optimal* 1D algorithm. For this reason, some fourth-order moments of the particle distribution, namely, $\langle x^4\rangle$, $\langle y^4\rangle$, and $\langle z^4\rangle$, are correct for the ordinary 3D algorithm; moreover, higher-order moments involving a single axis are the same as for the 1D optimal algorithm and thus the relative error is $O(1/t^2)$ at large t. However, the "mixed" fourth-order moments, $\langle x^2y^2\rangle$, $\langle x^2z^2\rangle$, and $\langle y^2z^2\rangle$, are incorrect; in particular, they are zero after the first step, since simultaneous moves along more than one axis do not exist in this algorithm. We have checked numerically that for large t the relative error for these moments is $O(1/t)$. On the other hand, for both optimal algorithms, even for those moments that are not exact the error is always $O(1/t^2)$. This is obvious for the direct product algorithm, as in this case all moments are products of 1D moments; for the other optimal algorithm, we have checked this numerically for a few moments of order six and higher. Moments involving three axes (e.g., $\langle x^2y^2z^2\rangle$) are zero in the ordinary algorithm after both the first and the second steps.

Extending these results to a space of arbitrary dimensionality using a hypercubic lattice is straightforward. First of all, direct product algorithms can be constructed, with the same properties as in 2D and 3D. On the other hand, simultaneous moves in more than two directions *may* still be unnecessary. To construct an algorithm with moves in one direction (axial moves) and two directions (diagonal moves) only, equations analogous to Eqs. (92)–(101) can be written for any dimensionality, and the solutions are always the same: All coefficients C with two equal indices (e.g., C_{xx}) are always equal to 1/3, all such coefficients with two pairs of equal indices (e.g., C_{xxyy}) are equal to 1/9, and all other C are zero. In d dimensions,

$$C_{xx} = 2p_1 + 4(d-1)p_2, \tag{120}$$

$$C_{xxyy} = 4p_2, \tag{121}$$

where p_1 is the probability of any particular axial move and p_2 is the probability of any particular diagonal move. This gives

$$p_1 = \frac{4-d}{18}, \tag{122}$$

$$p_2 = \frac{1}{36}. \tag{123}$$

It is easy to check that this works for $d = 1$, 2, and 3 (except for $d = 1$, there are no diagonal moves and so p_2 is meaningless). For $d > 3$, though, p_1 is non-negative only for $d = 4$. In this case, $p_1 = 0$, so the optimal algorithm contains no axial moves, only diagonal ones. Since there are $\binom{4}{2} = 6$ pairs of axes and thus $2\times 2\times 6 = 24$ different diagonal moves, the total probability of a diagonal move is $24/36 = 2/3$, so the probability to stay put is $p_0 = 1/3$, as in 3D. The time step is still $a^2/6D$. Note that this is now *larger* than in the straightforward algorithm with only axial moves and zero probability to stay put; but again, a larger time step is only possible because we use longer-range diagonal moves. For $d > 4$, it is impossible to make all k^4 terms in the dispersion relation zero with the set of moves considered. This may be possible using even longer-ranged moves (e.g., moves to second neighbors along axes) and/or simultaneous moves in three or more directions. In particular, the direct product algorithm is always an option in space of any dimensionality.

As in 3D, projecting the moves onto the axes gives the 1D optimal algorithm and projecting onto the coordinate planes gives the 2D optimal algorithm. In fact, projecting the moves of the direct product algorithm onto a coordinate hyperplane always produces the moves of the optimal direct product algorithm in the space of corresponding dimensionality, and projecting the moves of the 4D optimal algorithm with moves in one and two directions only onto a 3D hyperplane produces the moves of the 3D optimal algorithm with moves in one and two directions. The analogous statements about "projecting" the solutions follow immediately, as is the correctness of the second and fourth moments of the particle distribution.

V. TREATMENT OF IMPENETRABLE (REFLECTING) BOUNDARIES

So far, we have treated diffusion in a homogeneous medium without any boundaries. Of course, this is a trivial situation and the really interesting cases are those where some spatial inhomogeneities (e.g., in the form of obstacles) are present. In this section, we examine how to take impenetrable "reflecting" boundaries into account in LMC algorithms. Such boundaries are introduced into the continuum diffusion problems via boundary conditions, which in the unbiased case reduce to the expression

$$\hat{b}\cdot\vec{\nabla} n = 0, \tag{124}$$

where $\hat{b}$ is the unit vector normal to the boundary. Since the particle flux is proportional to the concentration gradient $\vec{\nabla} n$, Eq. (124) ensures that there is no particle flow across the boundary.

A. One dimension

We consider the case of a single boundary (or wall), which means that particles cannot go beyond a certain point $x = x_b$ on the line; without the loss of generality, we choose $x_b = 0$ and assume that the region $x > 0$ is accessible (Fig. 7). Let the site nearest to the wall have index 0 (the solid square in Fig. 7). Since this site has a neighbor only on the right, moves from the left to that site are impossible, and the master equation for the mean particle number at that site has to be modified:

$$n_0(t+\tau) = p_{0\to 0}n_0(t) + p_{1\to 0}n_1(t). \tag{125}$$

All other sites (which we refer to as *bulk sites*) have two neighbors and we assume that the corresponding master equations, including all the probabilities that are involved and

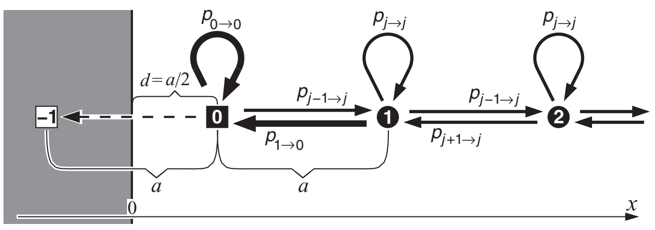

FIG. 7. Treatment of a reflecting boundary in the 1D optimal LMC algorithm. The inaccessible region is shaded. The site closest to the boundary is the boundary site 0 (solid square); all other sites are bulk sites (solid circles). The open square is a fictitious site in the inaccessible region useful in the analysis of the algorithm, as described in the text. The optimal placement of the boundary site is at the distance $d = a/2$ from the wall. Only the probabilities of the two moves leading to the boundary site (thick arrows), including the move from the boundary site to itself, are allowed to change (although $p_{1\to0}$ does not in the end); the probabilities of all other moves (thin arrows) are assumed to be the same as in the bulk from the outset. Notation for the probabilities of the moves used in the text is given. The move to the left from the boundary site (dashed arrow) is rejected and contributes to the probability of staying put.

the time step, remain as derived previously for free space (Fig. 7). Given that assumption, the probabilities entering Eq. (125), $p_{0\to0}$ and $p_{1\to0}$, can be determined uniquely from the condition that the probabilities of all moves leaving a given site (including the move to the same site, $j \to j$) should sum up to 1 (the normalization condition). In particular, for site 0 we have

$$p_{0\to0} + p_{0\to1} = 1. \tag{126}$$

However, $p_{0\to1}$ enters the master equation for site 1, which, by our assumption, remains the same as in free space, so $p_{0\to1}$ is the same as $p_{j-1\to j}$ in free space, which determines

$$p_{0\to0} = 1 - p_{j-1\to j} = p_{j+1\to j} + p_{j\to j}. \tag{127}$$

Likewise, for site 1

$$p_{1\to0} + p_{1\to1} + p_{1\to2} = 1. \tag{128}$$

Here $p_{1\to1}$ is involved in the equation for site 1 (and thus equals the "bulk" $p_{j\to j}$) and $p_{1\to2}$ is involved in the equation for site 2 (thus equals the "bulk" $p_{j-1\to j}$), and so $p_{1\to0}$ is determined uniquely as

$$p_{1\to0} = 1 - p_{j\to j} - p_{j-1\to j} = p_{j+1\to j}. \tag{129}$$

Since we assume throughout the paper that the time step is the same for all moves, the time step for "boundary" moves should be the same as for "bulk" moves and is thus uniquely determined.

Then, for instance, for the algorithm for unbiased diffusion optimized for accuracy, Eqs. (26)–(28) lead to

$$p_{0\to0} = 5/6, \tag{130}$$

$$p_{1\to0} = 1/6, \tag{131}$$

$$p_{0\to1} = 1/6, \tag{132}$$

$$\tau = a^2/6D. \tag{133}$$

The probability $p_{0\to1}$ is a bulk probability and is given here for completeness. Note that the probability of moving to the left from site 1 is the same as from any site with index $j+1 > 1$ ($p_{1\to0} = p_{j+1\to j}$). However, the probability of staying put increases by 1/6. This quantity 1/6 can be interpreted as the probability of the move to the left from site 0 that is now forbidden [in fact, this interpretation follows directly from Eq. (127)]. In other words, the algorithm is as follows: For the particle in any site, including the boundary site, select the next move according to the bulk probabilities, but if the boundary is crossed, reject the move and stay on the same site.

We can check explicitly that using Eqs. (130)–(133) in the master equation (125) indeed provides the best approximation of the continuum equation. In 1D, the boundary condition Eq. (124) becomes

$$\left.\frac{\partial n(x,t)}{\partial x}\right|_{x=0} = 0. \tag{134}$$

The general solution of the diffusion equation (6) with this boundary condition can be written as

$$n(x,t) = \int_0^\infty C(k)\cos(kx)\exp(-Dk^2t)dk, \tag{135}$$

which is just the general solution in free space Eq. (15) with the restriction $C(-k) = C(k)$ arising from the reflecting boundary condition (134) [which, together with the original condition $C(-k) = C^*(k)$, means that $C(k)$ is real]. Note that because of broken translational symmetry, *generally speaking*, the solution of the discrete master equation can no longer be written in the same form we used in the bulk Eq. (18). Therefore, the dispersion relation cannot be introduced and we have to use the alternative method based on inserting the integrand of the solution of the continuum equation (135) into the master equation and requiring that it turns into an identity as accurately as possible for small k [see Eq. (33) and the accompanying discussion].

Even though the master equations for the bulk sites involve only the bulk probabilities that we have already determined in Sec. II, it is still instructive to sketch the consideration for these sites using the new form of the solution Eq. (135), as there are some subtleties compared to Eq. (33). Inserting the integrand of Eq. (135) into Eq. (17) and dividing both sides by $C(k)\exp(-Dk^2t)$, we get

$$\begin{aligned}\cos(kx_j)\exp(-Dk^2\tau) \overset{?}{\simeq} {} & p_{j\to j}\cos(kx_j)\\ & + p_{j-1\to j}\cos[k(x_j-a)]\\ & + p_{j+1\to j}\cos[k(x_j+a)],\end{aligned} \tag{136}$$

where x_j is the position of site j. Note, however, that expanding straightforwardly in a series in k would not be correct, since x_j can be arbitrarily large. We did not face this problem in Sec. II, since dividing by $\exp(ikaj)$ eliminated all dependence on j. However, ka can still be assumed to be small, so the correct approach is expanding $\cos[k(x_j \pm a)]$ in series in ka:

$$\begin{aligned}\cos[k(x_j \pm a)] = {} & \cos(kx_j)(1 - k^2a^2/2 + k^4a^4/24 + \cdots)\\ & \mp \sin(kx_j)(ka - k^3a^3/6 + \cdots).\end{aligned} \tag{137}$$

The exponential on the left-hand side of Eq. (136) can be expanded as normal. This gives

$$\begin{aligned}&\cos(kx_j)(1 - Dk^2\tau + D^2k^4\tau^2/2 + \cdots)\\ &\overset{?}{\simeq} p_{j\to j}\cos(kx_j) + (p_{j-1\to j} + p_{j+1\to j})\\ &\quad\times \cos(kx_j)(1 - k^2a^2/2 + k^4a^4/24 + \cdots)\\ &\quad + (p_{j-1\to j} - p_{j+1\to j})\\ &\quad\times \sin(kx_j)(ka - k^3a^3/6 + \cdots). \qquad (138)\end{aligned}$$

This should be satisfied for any j, and thus for any x_j, which immediately gives $p_{j-1\to j} = p_{j+1\to j}$. Then after dividing both parts by $\cos(kx_j)$ the dependence on x_j is eliminated, and we end up with a series in k on both sides. Matching the coefficients gives the equations for the parameters whose only solution is given by Eqs. (26)–(28), as expected.

To obtain the probabilities for the boundary site 0, we need to repeat the same procedure for that site. Note, however, that in addition to the probabilities, there is one other free parameter: the distance d from the boundary to site 0. In free space without boundaries the problem is translationally invariant, so displacement of all sites by the same amount does not influence the results, and we assumed, without any loss of generality, that site 0 had coordinate $x_0 = 0$. However, now that the translational symmetry of the problem is broken by the presence of the boundary, the coordinate $x_0 = d$ of site 0 (Fig. 7) is an important parameter. The coordinate of an arbitrary site j is thus $x_j = d + aj$. Then, inserting the integrand of Eq. (135) in Eq. (125) and dividing both sides by $C(k)\exp(-Dk^2t)$, we get

$$\cos(kd)\exp(-Dk^2\tau) \overset{?}{\simeq} p_{0\to 0}\cos(kd) + p_{1\to 0}\cos[k(d+a)]. \qquad (139)$$

Dividing by $\cos(kd)$ and using $\tau = a^2/6D$, as in the bulk,

$$\exp(-a^2k^2/6) \overset{?}{\simeq} p_{0\to 0} + p_{1\to 0}\frac{\cos[k(d+a)]}{\cos(kd)}. \qquad (140)$$

Note that $d \simeq a$, so kd is small, and the Taylor expansion can be done normally, without any complications. Expanding up to $O(k^4)$,

$$\begin{aligned}&1 - a^2k^2/6 + a^4k^4/72\\ &\overset{?}{=} (p_{0\to 0} + p_{1\to 0}) - p_{1\to 0}\{a^2k^2(1/2 + d/a)\\ &\quad - (a^4k^4/12)[1/2 + 2d/a - 4(d/a)^3]\}. \qquad (141)\end{aligned}$$

This equality is only satisfied when the probabilities are given by Eqs. (130)–(131) and

$$d = a/2. \qquad (142)$$

The meaning of the result for d and the algorithm as a whole can be understood as follows. On the one hand, any solution of the continuum diffusion problem with a boundary is also a solution of the problem in free space, without a boundary, if it is continued symmetrically beyond the boundary. On the other hand, there is a similar correspondence between the discretizations of these problems as well. Indeed, looking at the corresponding master equations, the master equations for the bulk sites Eqs. (17) and (26)–(28) are, of course, identical for these problems, and the equation for the boundary site Eqs. (125) and (130)–(133) in the problem with the boundary is also equivalent to the equation for a bulk site, if we introduce a fictitious site (numbered -1) to the left of the boundary (the open square in Fig. 7) and assume that the mean particle number at that site is always equal to that at the boundary site ($n_{-1} = n_0$). This matches the correspondence between the continuum problems, if the sites 0 and -1 are located symmetrically with respect to the boundary, which immediately gives $d = a/2$. Note also that because of the equivalence between the master equations for the boundary and bulk sites, the solution of the system of all master equations can, in fact, be represented as

$$n_j(t) = \int_0^{\pi/a} C(k)\cos(kaj)\exp[-\alpha_d(k)t]\,dk \qquad (143)$$

[same as Eq. (18) with $C(k)$ real], even though, as mentioned, this is not possible in general for an arbitrary set of parameters p and d. Here j runs from 0 to ∞; that is, it includes both the boundary site and the bulk sites. The dispersion relation $\alpha_d(k)$ is, of course, exactly the same as in infinite space.

Another situation of interest is diffusion in a finite interval bounded by two reflecting walls. Since in the consideration above for a single wall the effect of that wall on the algorithm is local, only modifying the probabilities for a single boundary site, it should be possible to consider the two walls independently whenever there are at least two sites between them, with the modifications of the probabilities at each of the two boundary sites the same as in the case of a single wall. Indeed, even when there are only two sites, so the incoming transition for one of the sites is the outgoing transition for the other site and vice versa, both sites can still be treated just as regular boundary sites, since the incoming and outgoing probabilities for boundary sites are equal [see Eqs. (131) and (132)], so the algorithm remains internally consistent. Introducing the fictitious sites behind the walls and repeating the consideration in the previous paragraph leads to the same result. Of course, this assumes that the distance between the walls is a multiple of the mesh step a, so that it is possible to place both boundary sites at distance $a/2$ from the respective wall. This restricts the possible choices of a given the distance between the walls.

B. Two dimensions

In 2D, the situation is more complicated, especially since diagonal moves (some of which would cross the boundary) are involved. It is no longer possible to deduce the probabilities of the moves into the boundary sites based on the normalization conditions for the probabilities alone. We restrict our consideration to the case of a perfectly flat infinite boundary parallel to one of the Cartesian axes. By analogy with 1D, the case of two parallel infinite boundaries, with the motion of the particle confined to the space between these boundaries, is easy, as each boundary can be considered independently. A more complicated and interesting case of finite and/or curved obstacles and boundaries is touched upon briefly in Sec. VIII.

Consider specifically the case of a planar boundary located at $x = 0$ so that the half-space $x > 0$ is accessible (Fig. 8). The boundary sites form a column at $x = d$ (the solid squares in Fig. 8) and have indices $(0,l)$. For a site (j,l), the coordinates

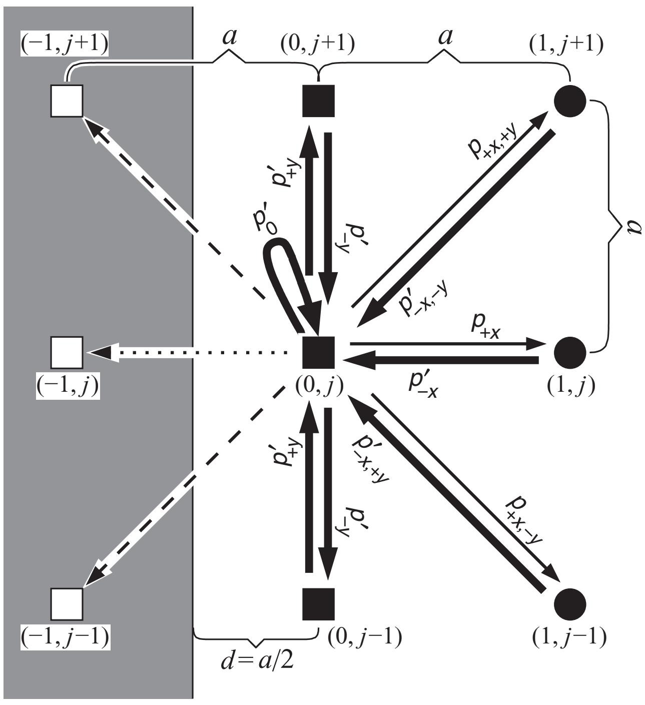

FIG. 8. Treatment of a reflecting boundary in the 2D optimal LMC algorithm. The diagram only shows the moves to and from one of the boundary sites $[(0,j)]$. The inaccessible region is shaded. The sites closest to the boundary are the boundary sites (solid squares); all other sites are bulk sites (solid circles). Open squares are fictitious sites in the inaccessible region useful in the analysis of the algorithm, as described in the text. The optimal placement of the line of boundary sites is at the distance $d = a/2$ from the wall. Only the probabilities of the moves to the boundary sites (thick arrows) are allowed to change compared to the bulk. Notation for the probabilities of the moves used in the text is given. The move to the left from the boundary site (dotted arrow) is rejected and increases the probability of staying put, but the diagonal moves across the boundary (dashed arrows) are projected along the boundary increasing $p'_{\pm y}$ compared to $p_{\pm y}$.

are $x_j = d + aj$, $y_l = al$ (there is still translational symmetry in the y direction, so we can choose arbitrarily $y_0 = 0$). The master equations for the boundary sites are

$$\begin{aligned} n_{(0,l)}(t+\tau) = {} & p'_0 n_{(0,l)}(t) + p'_{+y} n_{(0,l-1)}(t) + p'_{-y} n_{(0,l+1)}(t) \\ & + p'_{-x} n_{(1,l)}(t) + p'_{-x,+y} n_{(1,l-1)}(t) \\ & + p'_{-x,-y} n_{(1,l+1)}(t), \end{aligned} \tag{144}$$

where the primed probabilities correspond to moves ending on boundary sites and may differ from the bulk (unprimed) probabilities, whose values remain as determined in Sec. III.

The boundary condition (134) is still true (except for the fact that the particle concentration n now depends on y as well), and the general solution of the diffusion equation with this boundary condition is

$$\begin{aligned} n(x,y,t) = {} & \int_{k_x=0}^{\infty} \int_{k_y=-\infty}^{\infty} C(k_x,k_y) \cos(k_x x) \\ & \times \exp\left[ik_y y - D\left(k_x^2 + k_y^2\right)t\right] dk_x dk_y. \end{aligned} \tag{145}$$

Inserting the integrand in Eq. (144) and remembering that Eq. (102) for τ should remain valid, we obtain after dividing by $C(k_x,k_y)\cos(k_x d)\exp[ik_y al - D(k_x^2 + k_y^2)t]$

$$\begin{aligned} & \exp\left[-a^2\left(k_x^2 + k_y^2\right)/6\right] \\ & \overset{?}{\simeq} p'_0 + p'_{+y} e^{-ik_y a} + p'_{-y} e^{ik_y a} + (p'_{-x} + p'_{-x,+y} e^{-ik_y a} \\ & \quad + p'_{-x,-y} e^{ik_y a}) \frac{\cos[k_x(d+a)]}{\cos(k_x d)}. \end{aligned} \tag{146}$$

After expanding up to $O(k^4)$ and equating the corresponding coefficients, we get

for k^0,

$$p'_0 + p'_{+y} + p'_{-y} + p'_{-x} + p'_{-x,+y} + p'_{-x,-y} = 1; \tag{147}$$

for k_y^1,

$$-ia(p'_{+y} - p'_{-y} + p'_{-x,+y} - p'_{-x,-y}) = 0; \tag{148}$$

for k_y^2,

$$(-a^2/2)(p'_{+y} + p'_{-y} + p'_{-x,+y} + p'_{-x,-y}) = -a^2/6; \tag{149}$$

for k_y^3, equivalent to Eq. (148);
for k_y^4, equivalent to Eq. (149);
for k_x^2,

$$-a^2(1/2 + d/a)(p'_{-x} + p'_{-x,+y} + p'_{-x,-y}) = -a^2/6; \tag{150}$$

for $k_x^2 k_y$,

$$ia^3(1/2 + d/a)(p'_{-x,+y} - p'_{-x,-y}) = 0; \tag{151}$$

for $k_x^2 k_y^2$,

$$(a^4/2)(1/2 + d/a)(p'_{-x,+y} + p'_{-x,-y}) = a^4/36; \tag{152}$$

for k_x^4,

$$\begin{aligned} & (a^4/12)[1/2 + 2d/a - 4(d/a)^3](p'_{-x} + p'_{-x,+y} + p'_{-x,-y}) \\ & = a^4/72. \end{aligned} \tag{153}$$

It is convenient (but not necessary) to use the normalization conditions for the probabilities. For the boundary sites,

$$\begin{aligned} & p'_0 + p'_{+y} + p'_{-y} + p_{+x,+y} + p_{+x} + p_{+x,-y} \\ & = p'_0 + p'_{+y} + p'_{-y} + 1/6 = 1; \end{aligned} \tag{154}$$

for their neighbors [sites $(1,l)$],

$$\begin{aligned} & p'_{-x} + p'_{-x,+y} + p'_{-x,-y} + p_0 + p_{+x} \\ & \quad + p_{+y} + p_{-y} + p_{+x,+y} + p_{+x,-y} \\ & = p'_{-x} + p'_{-x,+y} + p'_{-x,-y} + 5/6 = 1. \end{aligned} \tag{155}$$

Comparing Eqs. (155) and (150), we get $d = a/2$, as in 1D, which makes Eq. (153) an identity. Then from Eqs. (151) and (152),

$$p'_{-x,+y} = p'_{-x,-y} = 1/36, \tag{156}$$

and from Eq. (155),

$$p'_{-x} = 1/9. \tag{157}$$

Also using Eq. (156), from Eqs. (148) and (149),

$$p'_{+y} = p'_{-y} = 5/36, \tag{158}$$

and finally, from either Eq. (147) or Eq. (154),

$$p'_0 = 5/9. \tag{159}$$

While

$$p'_{-x} = p_{-x}, \tag{160}$$

$$p'_{-x,+y} = p_{-x,+y}, \tag{161}$$

$$p'_{-x,-y} = p_{-x,-y}, \tag{162}$$

the other three probabilities are different from the bulk values:

$$p'_{+y} = p_{+y} + 1/36, \tag{163}$$

$$p'_{-y} = p_{-y} + 1/36, \tag{164}$$

$$p'_0 = p_0 + 1/9. \tag{165}$$

Note that not just the probability of staying put changes, so unlike in 1D, moves into the boundary are not always simply rejected. In fact, Eqs. (163)–(165) can be rewritten as

$$p'_{+y} = p_{+y} + p_{-x,+y}, \tag{166}$$

$$p'_{-y} = p_{-y} + p_{-x,-y}, \tag{167}$$

$$p'_0 = p_0 + p_{-x}. \tag{168}$$

Therefore, the correct changes of the probabilities can be obtained if the moves to the left ($-x$) from the boundary sites are rejected, but the moves in the ($-x, +y$) and ($-x, -y$) directions are replaced by the moves in the $+y$ and $-y$ directions, respectively. In other words, it is as if the moves into the wall are replaced by their projections along the wall. We also note that, similarly to 1D, the master equations for boundary sites become equivalent to those for bulk sites when a column of fictitious sites behind the boundary is introduced (open squares in Fig. 8) and the values of the mean particle numbers at these sites are made equal to those of the real boundary sites across the boundary (i.e., $n_{(-1,l)} = n_{(0,l)}$). All the probabilities could have been obtained from this consideration alone. It is also possible to obtain the 2D algorithm with a boundary as the direct product of the 1D algorithm with a boundary and the 1D algorithm in free space.

C. Three dimensions

We now consider the 3D case. Since we have considered two different 3D LMC algorithms in free space, the direct product algorithm with simultaneous moves in three directions and the unique optimal algorithm without such moves, we show how to treat a reflective boundary for each of these two algorithms in turn. We assume that the $x = 0$ plane serves as the boundary.

1. The direct product algorithm

The extension of the direct product algorithm to the case when a boundary is present is straightforward and can be obtained as the direct product of the 1D algorithm with a boundary and two 1D algorithms (or one 2D algorithm) in free space. This leaves the probabilities of moves into bulk sites unchanged compared to the free space algorithm, but the "primed" probabilities for the moves into boundary sites are now

$$p'_{-x} = 2/27, \tag{169}$$

$$p'_{\pm y} = p'_{\pm z} = 5/54, \tag{170}$$

$$p'_{-x,\pm y} = p'_{-x,\pm z} = 1/54, \tag{171}$$

$$p'_{\pm y,\pm z} = 5/216, \tag{172}$$

$$p'_{-x,\pm y,\pm z} = 1/216, \tag{173}$$

$$p'_0 = 10/27. \tag{174}$$

2. The algorithm without simultaneous moves in three directions

In this case, the consideration is similar to 2D. Compared to the 2D case, the master equation has additional terms involving $p'_{\pm z}$, $p'_{-x,\pm z}$, and $p'_{\pm y,\pm z}$. The solution of the continuum equation is obtained from Eq. (145) simply by introducing the same dependence of the integrand on k_z as its dependence on k_y and integrating over k_z from $-\infty$ to ∞. The matching equation (146) becomes

$$\begin{aligned}\exp\left[-a^2\left(k_x^2+k_y^2+k_z^2\right)/6\right] \overset{?}{\simeq}\, & p'_0 + p'_{+y}e^{-ik_ya} + p'_{-y}e^{ik_ya} + p'_{+z}e^{-ik_za} + p'_{-z}e^{ik_za} + p'_{+y,+z}e^{-ik_ya-ik_za} + p'_{+y,-z}e^{-ik_ya+ik_za} \\ & + p'_{-y,+z}e^{ik_ya-ik_za} + p'_{-y,-z}e^{ik_ya+ik_za} + (p'_{-x} + p'_{-x,+y}e^{-ik_ya} + p'_{-x,-y}e^{ik_ya} \\ & + p'_{-x,+z}e^{-ik_za} + p'_{-x,-z}e^{ik_za})\frac{\cos[k_x(d+a)]}{\cos(k_xd)}.\end{aligned} \tag{175}$$

First of all, there are new types of terms in the expansion with no analog in 2D, namely, those involving y and z (k_yk_z, $k_y^2k_z$, $k_yk_z^2$, $k_y^2k_z^2$, $k_y^3k_z$, and $k_yk_z^3$). Only four of the resulting equations are independent and they involve only four probabilities (those corresponding to simultaneous moves along y and z axes). Moreover, these equations are the same as they would be for a site in the bulk, since only moves parallel to the boundary are involved. Thus the four probabilities involved are the same as for bulk sites:

$$p'_{+y,+z} = p'_{+y,-z} = p'_{-y,+z} = p'_{-y,-z} = 1/36. \tag{176}$$

Equations (150) and (153) will have the probabilities of the ($-x, \pm z$) moves added to the sums of the probabilities entering these equations. The normalization condition

Eq. (155) becomes

$$\begin{aligned} p'_{-x} &+ p'_{-x,+y} + p'_{-x,-y} + p'_{-x,+z} + p'_{-x,-z} \\ &+ p_0 + p_{+x} + p_{+y} + p_{-y} + p_{+z} + p_{-z} \\ &+ p_{+x,+y} + p_{+x,-y} + p_{+x,+z} + p_{+x,-z} \\ &+ p_{+y,+z} + p_{+y,-z} + p_{-y,+z} + p_{-y,-z} \\ &= p'_{-x} + p'_{-x,+y} + p'_{-x,-y} + p'_{-x,+z} \\ &+ p'_{-x,-z} + 5/6 = 1. \end{aligned} \tag{177}$$

Just as in 2D, all three equations [the modified Eqs. (150) and (153) and Eq. (177)] involve the same sums of "primed" probabilities, which can thus easily be eliminated, and $d = a/2$ follows naturally. Equations (151) and (152) remain unchanged, and there will also be analogous equations involving the z axis instead of the y axis, so

$$p'_{-x,+y} = p'_{-x,-y} = p'_{-x,+z} = p'_{-x,-z} = 1/36. \tag{178}$$

Then from Eq. (177),

$$p'_{-x} = 1/18. \tag{179}$$

Equations (148) and (149) will have the probabilities of the $(\pm y, \pm z)$ moves added to them in the parentheses [or subtracted in the case of the $(-y, \pm z)$ moves in the first of these equations]. From these equations,

$$p'_{+y} = p'_{-y} = 1/12. \tag{180}$$

Equations obtained by matching k_z and k_z^2 terms are analogous to Eqs. (148) and (149), and from them similarly

$$p'_{+z} = p'_{-z} = 1/12. \tag{181}$$

Finally, Eq. (147) is modified to include the sum of all "primed" probabilities, and from it,

$$p'_0 = 7/18. \tag{182}$$

All other equations (modified as appropriate) and their analogs involving the z axis become identities once the above values are inserted in them.

3. *Relations for the probabilities*

In analogy to 2D, the probabilities for the boundary sites can be written (for both algorithms) as

$$p'_{+y} = p_{+y} + p_{-x,+y}, \tag{183}$$

$$p'_{-y} = p_{-y} + p_{-x,-y}, \tag{184}$$

$$p'_{+z} = p_{+z} + p_{-x,+z}, \tag{185}$$

$$p'_{-z} = p_{-z} + p_{-x,-z}, \tag{186}$$

$$p'_0 = p_0 + p_{-x}. \tag{187}$$

In addition, for the direct product algorithm

$$p'_{+y,+z} = p_{+y,+z} + p_{-x,+y,+z}, \tag{188}$$

$$p'_{+y,-z} = p_{+y,-z} + p_{-x,+y,-z}, \tag{189}$$

$$p'_{-y,+z} = p_{-y,+z} + p_{-x,-y,+z}, \tag{190}$$

$$p'_{-y,-z} = p_{-y,-z} + p_{-x,-y,-z}. \tag{191}$$

All other probabilities remain unchanged compared to the bulk. Just as in 2D, this can be interpreted as the diagonal moves across the boundary being projected along the boundary, rather than rejected. Also, as in 1D and 2D, the equivalence of the master equations for boundary and bulk sites once fictitious sites behind the boundary are introduced still holds, again, for both algorithms.

VI. ABSORBING BOUNDARIES

Another case of practical interest is that of *absorbing boundaries*. In this case, particles reaching the boundary are "absorbed" and disappear from the system. One reason for the importance of this problem is that the FPT problem can be reduced to it: The cumulative FPT distribution is equal to the fraction of absorbed particles as a function of time. In the continuum diffusion problem, the absorbing boundary condition is simply

$$n = 0 \tag{192}$$

at the boundary. In LMC, since the particle number is no longer conserved, there will inevitably be a new type of "move" for particles in sites adjacent to the boundary, during which the particle simply disappears. The probability of this occurring does not enter the master equations explicitly (the form of these equations remains unchanged, although the values of the probabilities change), but the probabilities of all other moves will no longer sum up to unity, so the normalization conditions for boundary sites cannot be used (but those for bulk sites are still valid).

A. One dimension

The general solution of the continuum diffusion equation with an absorbing boundary at $x = 0$ is

$$n(x,t) = \int_0^\infty C(k)\sin(kx)\exp(-Dk^2t)dk, \tag{193}$$

that is, the same as for the reflecting boundary Eq. (135), except the cosine is replaced by the sine. In the discrete case, the master equation for the boundary site is still given by Eq. (125). Plugging the integrand of Eq. (193) into this master equation, replacing τ with $a^2/6D$ and dividing both parts by $C(k)\sin(kd)\exp(-Dk^2t)$, we get

$$\exp(-k^2a^2/6) \overset{?}{\simeq} p_{0\to0} + p_{1\to0}\frac{\sin[k(d+a)]}{\sin(kd)}. \tag{194}$$

Expanding this up to $O(k^4)$, we obtain

$$1 - \frac{k^2a^2}{6} + \frac{k^4a^4}{72} \overset{?}{=} p_{0\to0} + p_{1\to0}\frac{d+a}{d}\left(1 - \frac{k^2}{6}a(2d+a) + \frac{k^4}{360}(3a^4+12a^3d+8a^2d^2-8ad^3)\right). \tag{195}$$

It is also convenient (but not necessary) to use the condition that the sum of the probabilities of all moves leaving site 1 is unity (since this site is not adjacent to the boundary). Given that two of these moves ($1 \to 1$ and $1 \to 2$) lead to a bulk site, their probabilities should coincide with the bulk probabilities ($p_{j\to j} = 2/3$ and $p_{j-1\to j} = 1/6$, respectively), and therefore the remaining probability, $p_{1\to0}$, should equal $1/6$. Using this in Eq. (195) and equating the k^2 terms, we get a quadratic equation for d that has two solutions:

$$d^{(1)} = a; \quad d^{(2)} = a/2. \tag{196}$$

Equating the k^0 terms and using Eq. (196) and $p_{1\to 0} = 1/6$, we get

$$p^{(1)}_{0\to 0} = 2/3; \quad p^{(2)}_{0\to 0} = 1/2. \tag{197}$$

As mentioned, in both cases

$$p^{(1)}_{1\to 0} = p^{(2)}_{1\to 0} = 1/6. \tag{198}$$

Both of these solutions turn the equation obtained from equating the k^4 terms into an identity; in fact, using that equation instead of the normalization condition would have given the same two solutions. Probabilities of moves leading to sites other than site 0 (i.e., to bulk sites) are the same as the corresponding free space probabilities Eqs. (26) and (27); in particular,

$$p^{(1)}_{0\to 1} = p^{(2)}_{0\to 1} = 1/6. \tag{199}$$

The time step likewise remains the same Eq. (28), which we have already used in the above derivation. Note that for the boundary site 0 the normalization condition is not satisfied: the sum of the outgoing probabilities is 5/6 for the first solution and 2/3 for the second one. The complements of these sums to unity,

$$p^{(1)}_d = 1/6 \quad \text{and} \quad p^{(2)}_d = 1/3, \tag{200}$$

respectively, can be interpreted as the probabilities of particle disappearance (or absorption by the boundary).

Let us now consider the physical interpretation of the two solutions. Note first of all that the first solution ($d = a$) corresponds to the absorption probability of 1/6, which is the same as the probability of the move to the left from a bulk site, and the probabilities of all allowed moves are unchanged compared to the bulk. Therefore, when using the first algorithm, we can treat the boundary site as a bulk site, except whenever a move to the left is attempted, the particle is deleted instead. There is no such simple interpretation for the second solution ($d = a/2$). Just as for the reflecting boundary, we can also analyze the algorithms by looking at the corresponding master equation. If the first set of parameters is used in the master equation (125), the resulting equation coincides with that for bulk sites Eq. (17), if a fictitious site (numbered -1) is introduced and the mean particle number at that site is made identically equal to zero. The physical interpretation of this is clear: If $d = a$, then site -1 is placed right at the absorbing boundary (Fig. 9), where the particle concentration in the continuum problem is indeed exactly zero. On the other hand, if the second set of parameters is used in the master equation, the resulting equation will coincide with that for bulk sites, if $n_{-1} = -n_0$. Since for $d = a/2$ the wall is halfway between sites -1 and 0, we get $n = 0$ at the wall simply by interpolating between n_0 and n_{-1}.

Of course, similar interpolation can be done for *any* d; for the interpolated value at $x = 0$ to be zero,

$$n_{-1} = -\frac{a-d}{d} n_0. \tag{201}$$

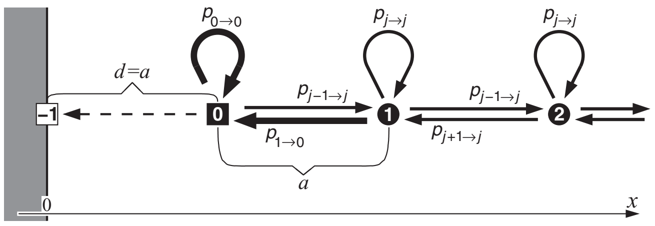

FIG. 9. One of the two variants of treatment of an absorbing boundary in the 1D optimal algorithm. The inaccessible region is shaded. The site closest to the boundary is the boundary site (solid square); all other sites are bulk sites (solid circles). In this variant, the placement of the boundary site is at the distance $d = a$ from the wall. The open square is a fictitious site at the wall useful in the analysis of the algorithm, as described in the text. The move left from the boundary site (dashed arrow) leads to the disappearance of the particle. Only the probabilities of the moves to the boundary site (thick arrows) are allowed to change in the analysis, but turn out to be the same as in the bulk for the optimal algorithm. Choosing $d = a/2$, as in Fig. 7, is also possible, but leads to different probabilities for the moves; note that in that case, the simple interpretation of the move to site -1 as leading to particle disappearance is no longer valid, since moves *from* that site to site 0 are also possible, as explained in the text.

We can use Eq. (201) in Eq. (17) with $j = 0$ and by comparison with Eq. (125) obtain

$$p_{0\to 0} = p_{j\to j} - p_{j-1\to j}\frac{a-d}{d}. \tag{202}$$

All other probabilities remain the same as in the bulk. Note that this approach is equally justified for all algorithms given by Eqs. (11)–(13), including both the ordinary and the optimal algorithm. However, for the ordinary algorithm, $p_{j\to j} = 0$ and so $p_{0\to 0}$ is non-negative and thus physically meaningful only for $d = a$ (assuming $d \leqslant a$, of course). Thus, there is only one way of implementing the absorbing boundary condition within the framework of the ordinary algorithm. Introducing a nonzero probability of staying put broadens the set of allowed values of d and thus gives more freedom. In particular, for $p_{j\to j} = 2/3$ and $p_{j-1\to j} = 1/6$, as in the optimal algorithm, any $d \geqslant a/5$ is possible. This is yet another advantage of using the optimal algorithm, or nonzero $p_{j\to j}$ in general. Even when optimality is required, we still have two choices, $d = a/2$ and $d = a$, more than for the ordinary algorithm. Note that for $d = a$, $p_{0\to 0} = p_{j\to j}$, so the probability that a particle stays put, which is the only probability of a move that can be different for the boundary site in this approach, is actually the same as in the bulk. For all other d, including $d = a/2$, $p_{0\to 0} \neq p_{j\to j}$. This is because, whenever $n_{-1} \neq 0$, moves *from* the fictitious site -1 to the boundary site 0 are possible in the auxiliary problem with the fictitious site, and these moves correspond to changing $p_{0\to 0}$ in the original boundary problem without the fictitious site. Another consequence of these moves from site -1 is that the probability of a move *to* site -1 from site 0, which in the auxiliary problem is always equal to the bulk probability $p_{j+1\to j}$, is not equal to the probability of particle disappearance in the original problem,

$$p_d = 1 - [p_{0\to 0} + p_{0\to 1}] = 1 - [p_{0\to 0} + p_{j-1\to j}], \tag{203}$$

since moves from site -1 lead to the "reappearance" of the particles from behind the wall. In fact, moves from site -1 are "negative", since $n_{-1} < 0$, so this "reappearance" actually *increases* the disappearance probability.

Note that the algorithm for arbitrary $d \geqslant a/5$ described above can be obtained by retaining just the zeroth-order terms in Eq. (195) *and* assuming that $p_{1\to 0}$ is the same as in the bulk, which maintains the normalization condition for the probabilities of moves leaving site 1. This condition makes physical sense, since site 1 is not adjacent to the wall and so particles leaving this site should not disappear [and, in fact, we have used this condition when obtaining Eqs. (196)–(198), although it was not necessary]. Matching k^2 terms in Eq. (195) as well requires dropping the normalization condition for site 1, which can lead to the unphysical disappearance of particles from or their creation at that site, unless $d = a/2$ or $d = a$.

Given the choice between the two fifth-order algorithms (with $d = a$ and $d = a/2$), which one should be preferred? In principle, the first choice has certain advantages. First, it follows from the previous discussion that the boundary condition $n = 0$ is explicit in the first algorithm, but is obtained indirectly via interpolation in the second. It is not obvious, however, that this improves the accuracy of the algorithm overall, as in both cases the continuum solution satisfies the respective master equations to the same order [$O(k^5)$]. Second, recalling the correspondence between diffusion with absorbing boundaries and the FPT problem, we note that it is the variant with $d = a$ that translates directly into the discrete FPT problem as considered in Sec. II C: In this case the rate of reaching a site is equal to the absorption rate when that site is replaced by an absorbing wall. This means that this variant of the algorithm will produce two correct moments of the absorption time distribution; it can be checked that this is not the case for the other variant, for which even the first moment (MFPT) is incorrect. On the other hand, one advantage of the $d = a/2$ variant is that if we consider bins of width a centered on the sites, then the bins associated with the boundary sites extend exactly to the wall; thus, if we assign to the sites the numbers of particles equal to the particle concentration at the sites in the continuum case, the total number of particles will be approximated more accurately than in the $d = a$ case.

B. Two dimensions

As in the case of a reflecting boundary, we assume that $x = 0$ is the boundary and $x > 0$ is the accessible region. We can derive the parameters of the optimal algorithm using the same approach as in the reflecting boundary case, by writing down the continuum solution, which in this case is

$$n(x,y,t) = \int_{k_x=0}^{\infty}\int_{k_y=-\infty}^{\infty} C(k_x,k_y)\sin(k_x x) \times \exp\left[ik_y y - D\left(k_x^2+k_y^2\right)t\right]dk_x dk_y, \qquad (204)$$

plugging the integrand into the master equation (144) and demanding that the resulting equality be satisfied as accurately as possible. A much shorter route is obtaining the optimal algorithm as the direct product of the 1D algorithm with an absorbing boundary and another one in free space. Since there are two variants of the optimal 1D algorithm (with $d = a$ and with $d = a/2$), we end up with two variants of the 2D algorithm: variant 1,

$$d^{(1)} = a, \qquad (205)$$
$$p_0'^{(1)} = 4/9, \qquad (206)$$
$$p_{-x}'^{(1)} = p_{\pm y}'^{(1)} = 1/9, \qquad (207)$$
$$p_{-x,\pm y}'^{(1)} = 1/36, \qquad (208)$$

and variant 2,

$$d^{(2)} = a/2, \qquad (209)$$
$$p_0'^{(2)} = 1/3, \qquad (210)$$
$$p_{-x}'^{(2)} = 1/9, \qquad (211)$$
$$p_{\pm y}'^{(2)} = 1/12, \qquad (212)$$
$$p_{-x,\pm y}'^{(2)} = 1/36. \qquad (213)$$

The particle disappearance probabilities are 1/6 and 1/3, respectively, as in 1D. Note that in the first variant all probabilities are the same as the bulk probabilities. Therefore, as in 1D, the boundary sites can be treated as the bulk sites, except moves into the walls are replaced by particle disappearance. This is not the case for the second variant. Also, it is still true, as in 1D, that the master equations for the boundary sites coincide with those for the bulk sites with $n_{-1,l} = 0$ for the first variant and $n_{-1,l} = -n_{0,l}$ for the second variant.

The disadvantage of the direct product approach is that, strictly speaking, we do not prove that the resulting sets of parameters are the only possible ones. A much longer consideration using the first method described above shows that the two variants of the algorithm we have obtained are indeed the only sets of parameters turning the master equation into an identity up to $O(k^4)$ when the continuum solution Eq. (204) is inserted into it.

C. Three dimensions

In 3D, both in free space and with a reflecting boundary, we considered two algorithms: the direct product algorithm and the algorithm with moves in one and two directions only. We now extend these considerations to the case of an absorbing boundary at $x = 0$.

1. The direct product algorithm

The direct product algorithm is obtained as the direct product of the 2D algorithm with an absorbing boundary and the 1D algorithm in free space. Since there are two variants of the former, we end up with two variants of the 3D algorithm. In the first variant,

$$d^{(1)} = a, \qquad (214)$$
$$p_0'^{(1)} = 8/27, \qquad (215)$$
$$p_{-x}'^{(1)} = p_{\pm y}'^{(1)} = p_{\pm z}'^{(1)} = 2/27, \qquad (216)$$
$$p_{-x,\pm y}'^{(1)} = p_{-x,\pm z}'^{(1)} = p_{\pm y,\pm z}'^{(1)} = 1/54, \qquad (217)$$
$$p_{-x,\pm y,\pm z}'^{(1)} = 1/216; \qquad (218)$$

in the second variant,

$$d^{(2)} = a/2, \tag{219}$$

$$p_0^{\prime(2)} = 2/9, \tag{220}$$

$$p_{-x}^{\prime(2)} = 2/27, \tag{221}$$

$$p_{\pm y}^{\prime(2)} = p_{\pm z}^{\prime(2)} = 1/18, \tag{222}$$

$$p_{-x,\pm y}^{\prime(2)} = p_{-x,\pm z}^{\prime(2)} = 1/54, \tag{223}$$

$$p_{\pm y,\pm z}^{\prime(2)} = 1/72, \tag{224}$$

$$p_{-x,\pm y,\pm z}^{\prime(2)} = 1/216. \tag{225}$$

2. *The algorithm without simultaneous moves in three directions*

To quickly derive this algorithm, we use an approach based on the correspondence between the master equations for the boundary and bulk sites, as described below.

First, we have mentioned when discussing the 1D algorithms in Sec. VI A that for the variant with $d = a$ these master equations coincide when n_{-1} is put equal to zero in the bulk equation. Extending this to 3D, we replace $n_{-1,l,m}$ with zero for all l and m in the 3D bulk master equation for the evolution of n_{0,l_0,m_0} and compare to the master equation for a boundary site. The result is that all probabilities remain as in the bulk [i.e., are given by Eqs. (117)–(119)], except that moves involving the $+x$ direction are, of course, impossible; also, the complement of the sum of the probabilities to unity (which equals 1/6, as in 1D and 2D) is the probability of the particle disappearance. The same interpretation as in 1D and 2D is still valid: the boundary sites can be treated as bulk sites, but all attempts to move into the boundary are replaced by particle disappearance.

On the other hand, for the variant with $d = a/2$ the 1D master equations for the boundary and bulk sites coincide when $n_{-1} = -n_0$ in the latter. Again, extending this to 3D, we obtain the following probabilities:

$$p'_0 = p_0 - p_{+x} = 5/18, \tag{226}$$

$$p'_{-x} = p_{-x} = 1/18, \tag{227}$$

$$p'_{\pm y} = p'_{\pm z} = p_{\pm y} - p_{+x,\pm y} = p_{\pm z} - p_{+x,\pm z} = 1/36, \tag{228}$$

$$p'_{-x,\pm y} = p'_{-x,\pm z} = p'_{\pm y,\pm z} = p_{-x,\pm y} = p_{-x,\pm z} = p_{\pm y,\pm z} = 1/36. \tag{229}$$

The sum of all these "primed" probabilities is 2/3, and so the disappearance probability is 1/3, as in 1D and 2D.

VII. RELATION TO LATTICE BOLTZMANN ALGORITHMS

One fact worth mentioning is a correspondence between the optimal LMC algorithms and *lattice Boltzmann* (LB) algorithms. The most well-known use of the LB approach is for mesoscopic fluid simulations [29,30]. LB equations can be thought of as master equations for particles residing at the sites of a lattice and moving at each time step to a predetermined set of nearby sites, as in LMC. However, a crucial difference is that the mean numbers of particles with each particular *velocity* are considered separately and evolve individually at each site according to the LB equations. The velocity of a particle determines how that particle moves at the next time step, and since the set of moves is discrete, the set of possible velocities in the algorithm is discrete as well. In different varieties of the LB approach different sets of discrete velocities (and thus of possible moves) are used. The standard notation used to distinguish these varieties is DmQs, where m is the dimensionality of the space and s is the number of possible velocity vectors.

The most popular LB equation has the form

$$n_{\vec{r}+\vec{e}_i}^{(i)}(t+\tau) = n_{\vec{r}}^{(i)}(t) - \omega\big(n_{\vec{r}}^{(i)}(t) - n_{\vec{r}}(t)\tilde{f}^{(i)}[\vec{V}_{\vec{r}}(t)]\big). \tag{230}$$

Here $n_{\vec{r}}^{(i)}(t)$ is the mean number of particles with velocity $\vec{v}_i$ at the site given by the vector $\vec{r}$, at time t; $\vec{e}_i$ is one of the lattice vectors and is related to $\vec{v}_i$ by $\vec{v}_i = \vec{e}_i/\tau$. The total mean number of particles at site $\vec{r}$ at time t, $n_{\vec{r}}(t)$, is given by

$$n_{\vec{r}}(t) = \sum_{i=1}^{s} n_{\vec{r}}^{(i)}(t). \tag{231}$$

The average velocity at site $\vec{r}$, $\vec{V}_{\vec{r}}(t)$, is

$$\vec{V}_{\vec{r}}(t) = \frac{\sum_{i=1}^{s} n_{\vec{r}}^{(i)}(t)\vec{v}_i}{n_{\vec{r}}(t)}. \tag{232}$$

The set of parameters $\tilde{f}^{(i)}[\vec{V}]$ is an approximation of the Maxwell velocity distribution shifted by velocity $\vec{V}$ and is discussed in more detail below. Equation (230) is a discretization of the Boltzmann kinetic equation for the particle distribution function in the phase space with the single-relaxation-time or Bhatnagar-Gross-Krook (BGK) approximation for the collision operator [31] and is often referred to by the abbreviation LBGK [29]. Finally, ω is a constant that determines how fast the relaxation toward the Maxwell distribution occurs and is called the *relaxation parameter*. Changing ω changes the resulting viscosity of the fluid, but by varying the mesh step and/or the time step simultaneously it is possible to keep the viscosity the same, so ω can serve as an adjustable parameter of the algorithm.

An important question in the design of LB algorithms is finding the best approximation of the equilibrium Maxwell velocity distribution, $\tilde{f}^{(i)}[\vec{V}]$, using a particular set of allowed discrete velocity vectors $\vec{v}_i$. It turns out that for $\vec{V} = 0$, in the best approximation the fractions of particles with particular velocities are equal to the probabilities of the corresponding moves in the optimal LMC algorithm with the same set of moves. For the D2Q9 LB algorithm [32], this will be our 2D optimal LMC; for the D3Q19 algorithm [30], our 3D LMC with moves along one and two directions; for the D3Q27 algorithm [32], our 3D direct product algorithm. This correspondence is not coincidental. The Maxwell distribution is a Gaussian distribution, as is the distribution of positions of particles diffusing in continuum space at a given time after they started at the same point. Given this, it is easy to see that the problem of finding the distribution of discrete velocities approximating the Maxwell distribution is mathematically equivalent to finding the set of probabilities of moves such that after a single step starting from the same site the resulting particle distribution is as close to Gaussian as possible. There is also a $\vec{V}$-dependent factor in $\tilde{f}^{(i)}[\vec{V}]$ that takes into account the shift of the velocity distribution; the only thing important for us for the remainder of this section is that it is unity at $\vec{V}=0$.

It is less well-known that the LB approach can also be used for diffusion problems [33–37]. In particular, LBGK algorithms for unbiased diffusion can be obtained from Eq. (230) by simply setting $\vec{V}\equiv 0$ regardless of the actual mean velocity at a particular site [35]. From now on, we will omit the velocity argument of $\tilde{f}^{(i)}$, since it is always zero. In analogy to fluid dynamics, varying the relaxation parameter ω changes the diffusion constant, but by changing the mesh step and/or the time step at the same time D can be kept constant [see, e.g., Eq. (239) below].

It is easy to see that for $\omega=1$ the resulting algorithm with a particular set of moves is equivalent to the optimal LMC algorithm with the same set of moves. Indeed, in this case Eq. (230) becomes

$$n^{(i)}_{\vec{r}+\vec{e}_i}(t+\tau)=n_{\vec{r}}(t)\tilde{f}^{(i)}, \tag{233}$$

which can be rewritten as

$$n^{(i)}_{\vec{r}}(t+\tau)=n_{\vec{r}-\vec{e}_i}(t)\tilde{f}^{(i)}. \tag{234}$$

As mentioned above, $\tilde{f}^{(i)}$ are equal to the probabilities of LMC moves; that is,

$$\tilde{f}^{(i)}=p_{\vec{e}_i}, \tag{235}$$

where $p_{\vec{e}_i}$ is the probability of the LMC move by vector $\vec{e}_i$. Summing up both sides of Eq. (234) over i and using Eq. (235), we get

$$n_{\vec{r}}(t+\tau)=\sum_{i=1}^{s}n_{\vec{r}-\vec{e}_i}(t)p_{\vec{e}_i}, \tag{236}$$

which is exactly the LMC master equation in its most general form. In fact, the equivalence between LBGK with $\omega=1$ and the forward time centered space finite-difference scheme (which is what the LMC master equation is) has been mentioned in the literature [35]. The fact that LB algorithms with $\omega=1$ are fifth-order-accurate can be inferred from Ref. [37]. Nevertheless, our work answers the following question: *If* we are restricted to using LMC algorithms, what parameters of these algorithms are optimal?

However, this begs the following question: Since LB algorithms with $\omega=1$ are as accurate as (in fact, entirely equivalent to) optimal LMC algorithms, can we do even better by using LB algorithms with $\omega\neq 1$? We note first that generalizing to $\omega\neq 1$ is not entirely trivial: If the equilibrium distribution $\tilde{f}^{(i)}$ is kept the same, the resulting algorithm is not optimally accurate (in fact, it is generally only third-order-accurate). This is perhaps somewhat paradoxical, given that this distribution is supposed to be the best discrete representation of the Maxwell distribution and this fact does not depend on ω. In 1D, the problem of finding the optimal values of $\tilde{f}^{(i)}$ for arbitrary ω has been solved very recently by Suga [38]. These optimal values are

$$\tilde{f}^{(\pm)}=\frac{\omega^2-12\omega+12}{6(\omega^2-8\omega+8)}, \tag{237}$$

$$\tilde{f}^{(0)}=1-2\tilde{f}^{(\pm)}, \tag{238}$$

where the first equation corresponds to the moves left and right and the second one to staying put. Note that there are a number of typos in equations in Ref. [38]; in particular, the equation for $\tilde{f}^{(\pm)}$ has a wrong sign in one of the terms that we are correcting here. The corresponding time step is

$$\tau=\frac{a^2(2-\omega)\tilde{f}^{(\pm)}}{D\omega}. \tag{239}$$

The resulting algorithms are indeed fifth-order-accurate for all allowed values of ω ($0<\omega<6-2\sqrt{6}\approx 1.101$). Above the upper limit, $\tilde{f}^{(\pm)}$ becomes negative. At $\omega=1$, these values become $\tilde{f}^{(\pm)}=1/6$, $\tilde{f}^{(0)}=2/3$, $\tau=a^2/6D$, corresponding to the optimal LMC algorithm, as expected.

The first reasonable question to ask is whether the sixth-order error can be eliminated as well by varying ω. Let us first obtain the dispersion relation. In 1D, the LB equation (230) turns into the following set of equations:

$$n^{(0)}_j(t+\tau)=(1-\omega)n^{(0)}_j(t)+\omega\tilde{f}^{(0)}\big[n^{(0)}_j(t)+n^{(+)}_j(t)+n^{(-)}_j(t)\big], \tag{240}$$

$$n^{(+)}_j(t+\tau)=(1-\omega)n^{(+)}_{j-1}(t)+\omega\tilde{f}^{(+)}\big[n^{(0)}_{j-1}(t)+n^{(+)}_{j-1}(t)+n^{(-)}_{j-1}(t)\big], \tag{241}$$

$$n^{(-)}_j(t+\tau)=(1-\omega)n^{(-)}_{j+1}(t)+\omega\tilde{f}^{(-)}\big[n^{(0)}_{j+1}(t)+n^{(+)}_{j+1}(t)+n^{(-)}_{j+1}(t)\big]. \tag{242}$$

Seeking the solution in the form

$$n^{(0,+,-)}_j(t)=C^{(0,+,-)}\exp[ikaj-\alpha(k;\omega)t], \tag{243}$$

we obtain the following characteristic equation for $\lambda=\exp(-\alpha\tau)$:

$$\det\begin{vmatrix}(1-\omega)+\tilde{f}^{(0)}\omega-\lambda & \tilde{f}^{(0)}\omega & \tilde{f}^{(0)}\omega\\ \tilde{f}^{(+)}\omega e^{-ika} & [(1-\omega)+\tilde{f}^{(+)}\omega]e^{-ika}-\lambda & \tilde{f}^{(+)}\omega e^{-ika}\\ \tilde{f}^{(-)}\omega e^{ika} & \tilde{f}^{(-)}\omega e^{ika} & [(1-\omega)+\tilde{f}^{(-)}\omega]e^{ika}-\lambda\end{vmatrix}=0. \tag{244}$$

This is a cubic equation, so it has three solutions for every value of k. Two of the values of $\alpha(k;\omega)$ remain nonzero in the limit $k \to 0$ and so decay fast and are irrelevant at long times (but are relevant at short times, as discussed below). The third solution should have the form

$$\alpha(k;\omega) = D[k^2 - A(\omega)a^4k^6 + O(k^8)], \tag{245}$$

when the optimal parameters given by Eqs. (237)–(239) are used in Eq. (244). By comparison to Eq. (29), $A(1) = 1/540$. For other values of ω, while it is possible to solve Eq. (244) analytically and expand in k, this is too cumbersome, and it is easier to put $D = 1$ and $a = 1$, obtain the numerical solutions for several small values of k, and extrapolate the deviation of α from k^2 to $k \to 0$ for each ω of interest. The result is plotted in Fig. 10(a). Note that $A(\omega)$ never reaches zero, so it is *impossible* to make the algorithm seventh-order-accurate.

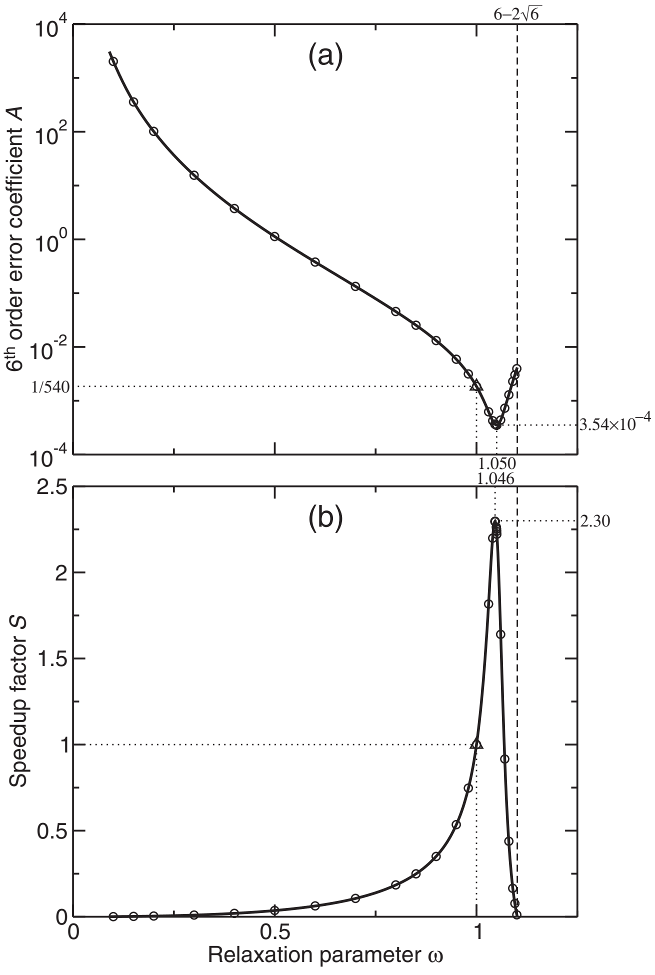

FIG. 10. For the fifth-order-accurate 1D LBGK algorithm, as derived by Suga [38]: (a) the numerical prefactor A in the sixth-order term in the dispersion relation [see Eq. (245)] as a function of the relaxation parameter ω; (b) the speedup factor S [Eq. (248)] for the algorithm with the relaxation parameter ω compared to that with the relaxation parameter equal to unity, assuming that both have the same accuracy. In both plots, the circles mark the values calculated numerically and the curves are cubic spline interpolations between these points. The triangles correspond to the $\omega = 1$ case equivalent to the optimal LMC.

The lowest value of $A(\omega)$ is about 3.54×10^{-4}, which is about a factor of 5.2 lower than $A(1)$, and is reached at $\omega \approx 1.050$. Does it make sense to use the LB algorithm with this value of ω instead of the LMC algorithm? We note that according to Eq. (245), the algorithms with $\omega = 1$ and with an arbitrary $\omega = \omega_0$ are equally accurate at small k when the mesh step in the latter is a factor of $[A(1)/A(\omega_0)]^{1/4} = \{1/[540A(\omega_0)]\}^{1/4}$ larger than in the former, that is,

$$\frac{a_{\omega=\omega_0}}{a_{\omega=1}} = \left(\frac{1}{540A(\omega_0)}\right)^{1/4}. \tag{246}$$

Since the computational effort per LB iteration is inversely proportional to the mesh size, we have the speedup factor of $\{1/[540A(\omega_0)]\}^{1/4}$, which at $\omega_0 = 1.050$ is about 1.51. Note that this is the speedup *per time step*. However, the time step τ itself depends on ω, both explicitly and through a [see Eq. (239)], and the total number of iterations is inversely proportional to τ, so there is an additional speedup factor of

$$\frac{\tau_{\omega=\omega_0}}{\tau_{\omega=1}} = \frac{a^2_{\omega=\omega_0}}{a^2_{\omega=1}} \cdot \frac{(2-\omega_0)(\omega_0^2 - 12\omega_0 + 12)}{\omega_0(\omega_0^2 - 8\omega_0 + 8)}$$
$$= \left(\frac{1}{540A(\omega_0)}\right)^{1/2} \frac{(2-\omega_0)(\omega_0^2 - 12\omega_0 + 12)}{\omega_0(\omega_0^2 - 8\omega_0 + 8)}. \tag{247}$$

The total speedup factor $S(\omega_0)$ is

$$S(\omega_0) = \left(\frac{1}{540A(\omega_0)}\right)^{3/4} \frac{(2-\omega_0)(\omega_0^2 - 12\omega_0 + 12)}{\omega_0(\omega_0^2 - 8\omega_0 + 8)}. \tag{248}$$

At $\omega_0 = 1.050$, this is about 2.24. The function $S(\omega_0)$ is plotted in Fig. 10(b); its maximum is reached at $\omega_0 \approx 1.046$, with the value not much different from that at 1.050 ($\approx$2.30, to be precise).

Note, however, that the above calculation refers to the speedup factor of the LB algorithm at a particular ω compared to the *LB algorithm* at $\omega = 1$ (assuming that both calculations use the same code written for an arbitrary ω). The latter is equivalent to the optimal LMC algorithm, but much more computationally costly, both because three equations per site (instead of just one in the numerically exact LMC) need to be solved and because of the additional relaxation step. The expected speedup of at least a factor of three when moving from LB with $\omega = 1$ to the numerically exact version of LMC will, of course, more than compensate the speedup factor of 2.30 achievable by moving to the optimal value of ω. Therefore, it appears that switching from LMC to LB does not improve the efficiency if the goal is achieving a particular precision. (Note also that in terms of memory requirements, the increase in the mesh step does not compensate the need to store three values of particle numbers per site instead of one.) Using LB does give more flexibility, in particular, since optimal LMC imposes a relation between the time step and the mesh step Eq. (28), which can be relaxed in LB by varying ω [see Eq. (239)]. Note, however, that only values of ω below 1.101 are allowed and at the same time, the error increases extremely rapidly when ω is decreased below unity [note the logarithmic scale of the y axis in Fig. 10(a)], so the usable range of ω is probably rather narrow. (This rapid increase of the error can also be seen in the actual simulation data presented in Table I in Ref. [38].) In particular, even though it may seem at first glance from Eq. (239) that the time step can be increased indefinitely by decreasing ω, this is not true, since, unless one is content with an enormous loss of accuracy, the mesh step a has to be decreased as well, which actually leads to a net *decrease* of the time step and the resulting loss of efficiency, as Fig. 10(b) shows. Another problem with choosing small ω is that the sensitivity to initial conditions (see below) extends to longer times.

We have also shown that the optimal LMC algorithms are not only best for modes with small k that are the most important ones at long times, but also reproduce the fourth moment of the particle distribution exactly at *all* times. How do the optimal LB algorithms with $\omega \neq 1$ fare in this respect? An immediate problem when answering this question is that the initial condition for the LB algorithm is *underdetermined*: To solve the diffusion problem, only the *total* average number of particles at each site at $t = 0$ is required, but the LB algorithm also requires the knowledge of the number of particles with each velocity component separately. In effect, instead of one mode at each value of k, there are as many modes as there are different velocities [indeed, Eq. (244) for the 1D LB algorithm has three solutions]. The decay rate of only one of these modes approaches zero as $k \to 0$, and it is the dispersion relation of this mode that we have required to reproduce the dispersion relation of the diffusion equation as accurately as possible. All other modes decay much faster and thus do not matter in this limit and, as a consequence, how the particles are distributed between different velocity components should not matter at long times. However, it certainly matters at short times. The question of choosing the initial conditions of the LB algorithms in an optimal way is a complex one, both for fluid dynamics [39] and for diffusion problems [40]. Nevertheless, one can ask: Are there *any* initial conditions for which the first four moments of the particle distribution are correct at all times? It is easy to see that the answer, at least in 1D, is *no*, unless $\omega = 1$. Indeed, if all particles start at site 0, then, no matter how these particles are distributed between the three possible velocities, after one step they can end up only on sites -1, 0, and 1, and therefore at this time all moments are necessarily the same (apart from the factors a^m), so the ratio of the fourth and second moments is a^2. This ratio is only achieved in the continuum problem at $t = a^2/6D$ [see Eqs. (36) and (37)], and this fixes the time step of the algorithm to that value. For fifth-order-accurate algorithms, based on Eqs. (239) and (237), the time step is only equal to $a^2/6D$ when ω satisfies the following equation:

$$\frac{(2-\omega)(\omega^2-12\omega+12)}{\omega(\omega^2-8\omega+8)} = 1. \tag{249}$$

Besides $\omega = 1$, the only other solutions are $\omega = 5 \pm \sqrt{13}$, which are outside the allowed range $0 < \omega < 6 - 2\sqrt{6}$.

Of course, the above analysis deals with the 1D case and the shortest-ranged algorithms (D1Q3). It does not rule out the possibility that in higher dimensions seventh-order-accurate algorithms exist or the usable range of ω would turn out to be wider. However, constructing even fifth-order-accurate algorithms in higher dimensions is most likely very tedious (unlike LMC and standard LB using approximations to the Maxwell distribution as $\tilde{f}$, there is no reason to expect that the optimal $\tilde{f}$ can be obtained simply as the direct product of the optimal 1D values). The same applies to algorithms with longer-ranged moves [41–43]. Also, we have only considered free space diffusion; however, we suspect that, if anything, the presence of boundaries and obstacles can make the case for LMC stronger: For instance, while in free space it is possible to use a larger mesh step in LB than in LMC while keeping the same precision, this may not be possible with boundaries since the coarser mesh may not reproduce them as accurately. All in all, while we cannot rule out that there may be cases where use of LB for simulating diffusion is justified, LMC algorithms certainly have many advantages and, given this, the question of optimizing their accuracy arises, which is precisely the question this paper addresses.

VIII. DISCUSSION

In this paper, we have obtained the sets of parameters (the transition probabilities and the time step) of LMC algorithms for particle diffusion problems that optimize their accuracy. The problem was solved by demanding that the solutions of the master equation of the LMC algorithm approximate those of the continuum diffusion equation as accurately as possible. The matching between the continuum equations and the discrete master equations was done using the general solution of the continuum equation written as a Fourier expansion in space where each component decays exponentially in time. The general solution of the master equation written in a similar form can also be used, in which case the goal is to make the decay rates of the Fourier modes as close as possible in the continuum and discrete equations. However, having the solution of the master equation is not necessary: For example, we did not use it in Secs. V and VI, where we considered the treatment of boundaries. In that case, the solution of the continuum equation was simply inserted into the master equation and the parameters of the latter adjusted to satisfy the resulting equality as accurately as possible. Having the solution of the master equation, while not necessary, allowed the illustrative comparison between the dispersion relations presented in Fig. 2.

Since LMC simulations are generally carried out in order to study long length and long time scale diffusion processes, all expressions are expanded in the wave number k and terms of the lowest order are matched. Matching the dispersion relations up to $O(k^2)$ (producing third-order-accurate algorithms, since third-order terms are automatically zero, as are all other odd-order terms) guarantees that the average velocity (equal to zero in the unbiased case) and the diffusion constant are reproduced correctly by the algorithm. In other words, both the mean displacement (the first moment of the distribution for particles starting at a particular site; again, equal to zero) and the mean-square displacement (the second moment) are correct not only in the long-time limit, but, in fact, *at all times*. On the other hand, for all higher-order moments, while the leading term at large times has the correct coefficient, these higher moments are, in general, incorrect for shorter times. The algorithms can also suffer from some artifacts, as discussed in Sec. II for the 1D case. Going to $O(k^4)$ (fifth-order-accurate algorithms), which is possible with only first neighbor moves in 1D and when both first neighbor and second neighbor (diagonal) moves are included in 2D and 3D, removes the artifacts and makes the fourth moment, as well as the *subleading* terms of all higher moments, correct, too. We refer to the resulting algorithms as *optimal*, in the sense that they achieve the best accuracy given the chosen set of moves, although they are not optimal in terms of the simulation speed if accuracy is not a concern. The latter fact is due, in particular, to a distinctive feature of the optimal algorithms: a nonzero probability for a particle not to move during a particular time step, or a *waiting time*.

We have also shown for the 1D case that for a particle starting at a particular point between two boundaries, the optimal algorithm reproduces correctly the first two moments of the distribution of the times of first passage to the boundaries if both the initial point and the boundaries coincide with lattice sites. In fact, the optimal algorithm can also be obtained based on the requirement that the first two moments of the FPT are reproduced correctly *or* on the requirement that the first four moments of the particle distribution are correct, instead of matching the dispersion relations. We have also studied the full distributions of the FPTs comparing them to the exact continuum result. We have shown that the optimal algorithm converges much faster to the exact result than the ordinary algorithm without a waiting time as the mesh step decreases, so if a particular accuracy is required, a much coarser mesh can be used with the optimal algorithm, which provides a speedup that in the end more than compensates for the loss of computational speed due to the waiting time. In higher dimensions, the moments of the FPT distribution are no longer reproduced perfectly (and, in fact, they depend on how exactly the boundary in the continuum problem is chosen), but, at least in the case of a square boundary going through rows and columns of sites in 2D, both the moments and the full distributions are much more accurate and converge much faster as the mesh is made finer than for the ordinary algorithm.

In Sec. VII, we discussed the relation between optimal LMC algorithms and LB algorithms for diffusion. It turns out that the optimal LMC algorithms are equivalent to a particular class of LB algorithms, the LBGK algorithms with the relaxation parameter $\omega = 1$. A natural question is whether replacing LMC with LB and varying ω can make further improvements possible within the LMC framework without extending the range of the moves. We analyzed this issue in 1D and found that, first of all, varying ω cannot eliminate the sixth-order term in the dispersion relation. While this term can be decreased, thus allowing one to increase the mesh step needed to achieve the same accuracy, this does not compensate the significant increase in the CPU time per site in the LB algorithm due to having to keep track of three kinds of particles with different velocities instead of one. Moreover, we have also shown that the LBGK algorithm in 1D cannot reproduce both the second and the fourth moments of the particle distribution simultaneously at all times, unlike the optimal LMC. On the other hand, an advantage of LB is that varying ω allows one to vary the mesh step and the time step independently within some range, although we also showed that the usable range of ω is relatively narrow.

We have also considered the treatment of impenetrable reflecting boundaries (Sec. V). First, for the algorithms to remain fifth-order-accurate, the sites nearest to the boundaries need to be located at distance $d = a/2$ from them, where a is the mesh step. Second, often in MC algorithms, if an attempted move is forbidden, for example, because a particle would overlap with an obstacle, it is simply rejected. This is a correct choice in algorithms that only intend to reproduce equilibrium properties, as it preserves detailed balance, which is all that matters in that case. However, as we have shown, this is not the best choice in a *dynamical* algorithm in a two- or higher-dimensional space. In that case, the best accuracy is achieved by replacing forbidden moves across the boundary with their projections along that boundary. In LB literature, this is referred to as *specular reflection* [44]. This result makes physical sense: If, for example, in 2D all boundaries are orthogonal to the x direction and are infinitely long in the y direction, the diffusion in the y direction should not be affected, which is only possible if the simultaneous moves in the x and y directions (present in the optimal LMC algorithm) are projected along the y direction.

Next, we have considered the case of absorbing boundaries (Sec. VI). It is interesting to note that for each of the optimal free space algorithms, there are two ways of treating the boundaries parallel to one of the lattice axes while keeping the same order of accuracy of the algorithm. The more obvious one is to destroy the particle whenever it moves into the wall, but leave all other moves unmodified. This is the best approximation for a wall that is at the distance equal to the mesh step from the last row of sites next to the boundary ($d = a$ using the notation of the paper). However, there is also a second variant, with rules that do not have a simple interpretation, which corresponds to the last row of sites being at the distance from the boundary of half the mesh step ($d = a/2$). This second variant (as well as intermediate, nonoptimal choices) is only possible in algorithms with a waiting time (even if optimality is not required).

Note that we have only treated the case of infinite, flat boundaries. In real situations of interest, one deals with finite and/or curved boundaries (finite obstacles, tortuous pores, etc.). In the simplest case, the walls consist of flat pieces, each of which is orthogonal to one of the axes, that are joined at corners, like those in Fig. 11. An example would be a rectangular obstacle in 2D with the sides along the axes. Sites adjacent to the walls that are far from the corners can be treated as boundary sites next to infinite walls (this assumes that it is possible to choose the lattice so that the distance d from the boundary sites to the boundary is as required by the algorithm everywhere, or else the accuracy will be sacrificed). However, we still need a way to treat sites next to the corners. We note, though, that we have assumed throughout that the particle concentration varies smoothly on the length scale of the mesh step a, and this implies that any features of the obstacles and the walls are likewise smooth on this length scale, so corners should be rare. It does not matter much then how exactly these rare situations are treated; we will consider these complications in a separate paper in the future. However, we can offer some intuitive guidance in some of these situations. For example, it is intuitively clear how to deal with absorbing boundaries in the algorithms with $d = a$

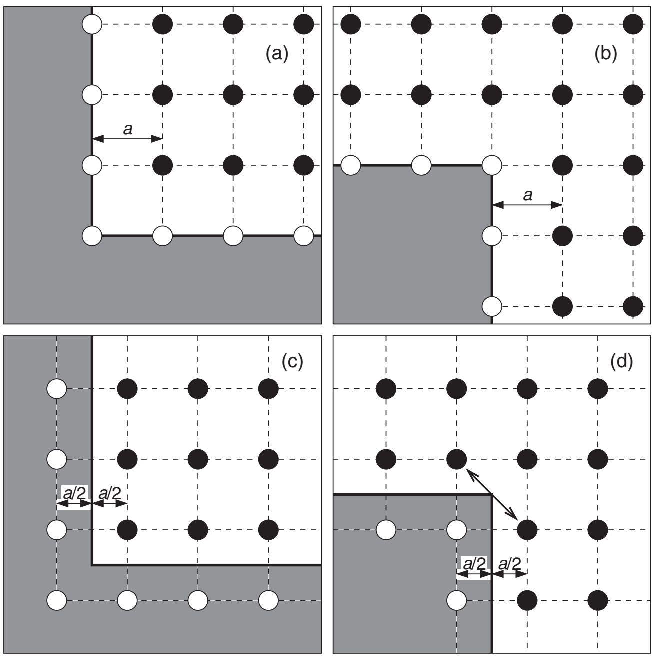

FIG. 11. 2D examples of different configurations of walls forming corners and different variants of meshes for optimal LMC algorithms. Forbidden areas are shaded; real sites are black circles and fictitious sites introduced to facilitate the analysis of the algorithms are white circles. Cases (a), (b), and (c) can be analyzed, as described in the text, but in the case (d) there are complications that physically correspond to the uncertainty about the probabilities of the diagonal transitions "across the corner" indicated by the arrow.

[Figs. 11(a) and 11(b)]: In these algorithms, all moves into the wall lead to absorption and all other moves are unchanged, which generalizes straightforwardly to corners. It is also clear how to handle the situation with $d = a/2$ (either absorbing or reflecting walls), when, for example, in 2D the particle is allowed to move in one quadrant and the other three are blocked [Fig. 11(c)]. In this case, fictitious sites are introduced inside the forbidden area symmetrically with respect to the boundary to the row of sites nearest to it [Fig. 11(c)], and the mean particle numbers at these sites are assumed equal to those at sites nearest to them across the boundary (with the minus sign in the case of absorption). The generalization to the case of a corner is straightforward, by assuming that the fictitious site closest to the corner takes on the value of the real site closest to the corner (with the plus sign even in the case of absorption, to ensure antisymmetry with respect to both axes in that case). However, in the otherwise analogous case when three quadrants are free and one is blocked [Fig. 11(d)], there is some ambiguity, as it is not clear which of the three real sites closest to the corner should give its value to the fictitious site nearest the corner. This translates into an uncertainty about the probabilities of the diagonal moves "passing through the corner" shown in the figure. Of course, in some cases it may be necessary or convenient to allow values of d not equal to either a or $a/2$, in which case further generalization is needed.

Curved boundaries or those not orthogonal to the axes are even more complicated. Approximating such boundaries by piecewise flat boundaries with each piece orthogonal to one of the axes, in such a way that those sites inside the original boundary are still inside and those outside are still outside, would in effect produce "jagged" boundaries with many corners adjacent to each other that need special treatment. A more accurate approach would be approximating such boundaries with flat pieces oriented *arbitrarily*. However, this also requires a separate consideration. The rules we have derived cannot be extended straightforwardly to such "tilted" boundaries, even when they are perfectly flat: for instance, projecting forbidden moves along the boundary is impossible, as such a projection would not coincide with any of the allowed moves. Solving these problems will be a subject of future work. Considering finite-size hard particles with arbitrary shapes is a more complicated problem since particle rotation needs to be taken into account (spherical particles are an exception). An even more complicated problem is that of several diffusing hard-core particles that can collide with each other. Such particles would move independently between collisions (according to the rules derived in this paper, if rotation can be ignored), but rules for treating collisions still need to be worked out. It is worth noting that, given the close relationship between the LMC and LB algorithms, extensive literature on boundary conditions for LB algorithms (e.g., [44–49]) can perhaps be useful.

The preceding discussion may sound as if there is still some way to go before the algorithms described here become practically useful. This is not entirely true, however. As mentioned in Sec. II A, our 1D optimal algorithm is the zero-field limit of a previously derived algorithm for biased diffusion [19]. We have recently applied the latter algorithm to a 1D model of polymer translocation through a nanopore [50] and showed that this algorithm gives very different widths of the translocation time distributions compared to the ordinary algorithm without waiting times, thus demonstrating the importance of introducing waiting times [51]. It is true, though, that the utility of the new algorithms will be further enhanced when the issues described above are taken care of, as well as when the approach is extended to biased diffusion in 2D and 3D as well. In general, we expect the new algorithms to be useful in those problems where the transient behavior of the system at short times is important. This includes various first-passage problems (including the polymer translocation problem mentioned above), as well as adsorption, aggregation [1], anomalous diffusion [17], and reaction-diffusion [3] problems. One example of a problem dealing with transient effects where our approach may be useful is studying drug release profiles from disordered porous matrices [14,15]. On the other hand, if only the long-time behavior is of interest and short-time transient dynamics does not affect it, then the ordinary algorithms with jumps at every time step may be just as good and may offer computational time savings compared to the optimal algorithms. Tests of the algorithms developed in this paper and their extensions in various situations of practical interest, as well as comparisons to other algorithms, will be a subject of future work.

In those cases where discrete-time LMC can be used, there is also an option to use MC algorithms with continuous space (off-lattice MC), continuous time (often referred to as kinetic MC [2,21–23]), or both. Of course, we do not claim that LMC is always the best option; rather, the goal of this paper is to suggest that in those cases where the choice in favor of LMC is made, one could improve the accuracy of the simulations by using the LMC modifications described here. We do note, however, that since the choice between LMC and continuous MC is often driven by the tradeoff between higher speed of the former and higher accuracy of the latter, improving the accuracy of LMC may actually make LMC preferable in some situations where this would not be the case otherwise.

ACKNOWLEDGMENTS

We are grateful to the anonymous referee for pointing out the existence of LB algorithms for simulating diffusion and the connection between these algorithms and our optimal LMC algorithms (see Sec. VII). This research was supported by grants from NSERC and the NIH (Grant No. 2 R01 HG001970-07 through Stanford University).

APPENDIX: A DERIVATION OF THE LONG-TIME DEPENDENCE OF THE FIRST-PASSAGE RATE FOR THE LMC ALGORITHMS

The 1D first-passage problem for LMC algorithms, as formulated in Sec. II C, can be solved by considering the master equations (17) for sites from $-N+1$ to $N-1$ and fixing

$$n_{\pm N} = 0. \tag{A1}$$

The solution of this system will give the mean numbers of particles at every site that have not yet reached the walls at sites $\pm N$. This solution can be obtained from the general solution for the infinite lattice Eq. (18) by picking those modes that satisfy the boundary conditions Eq. (A1), which gives $k_m = \pi(2m+1)/2Na$ with $m = 0,\ldots,N-1$ and $C(-k) = C(k)$; thus, the solution is

$$n_j(t) = \sum_{m=0}^{N-1} C_m \cos[\pi(2m+1)j/2N] \times \exp\{-\alpha_d[k = \pi(2m+1)/2Na]t\}. \tag{A2}$$

The mean number of particles reaching the walls at a time step occurring at time t, $\nu(t)$, is proportional to the number of particles at adjacent sites at the previous time step, that is,

$$\nu(t) = p_{j-1\to j} n_{N-1}(t-\tau) + p_{j+1\to j} n_{-N+1}(t-\tau). \tag{A3}$$

This number per unit time (i.e., the rate) is

$$r(t) = \frac{\nu(t)}{\tau} = \frac{2D}{a^2} n_{N-1}(t-\tau), \tag{A4}$$

where we have used Eq. (11) and the fact that, according to Eq. (A2) or simply from symmetry considerations, $n_{-N+1} = n_{N-1}$.

For the optimal algorithm, $\alpha_d(k)$ is a monotonically increasing function (line $\tau = \tau_{\max}/3$ of Fig. 2); thus, in the large t limit, only the first term in the sum survives, which gives

$$n_j(t) \simeq C_0 \cos(\pi j/2N) \exp[-\alpha_d(k = \pi/2Na)t]. \tag{A5}$$

Then from Eq. (A4),

$$r(t) \simeq \frac{2DC_0}{a^2} \sin(\pi/2N) \exp[\alpha_d(k = \pi/2Na)\tau] \times \exp[-\alpha_d(k = \pi/2Na)t]. \tag{A6}$$

Therefore, the decay rate is

$$\beta_{\text{opt}} = \alpha_d(k = \pi/2Na) = -\frac{6DN^2}{b^2} \ln[2/3 + (1/3)\cos(\pi/2N)], \tag{A7}$$

where we have used $a = b/N$ and Eq. (19) with p and τ given by Eqs. (26)–(28). The prefactor is

$$\gamma_{\text{opt}} = \frac{2DC_0}{a^2} \sin(\pi/2N) \exp[\alpha_d(k = \pi/2Na)\tau]. \tag{A8}$$

To find C_0, we need to expand the initial condition, $n_j(0) = \delta_{j,0}$, in the Fourier series,

$$\delta_{j,0} = \sum_{m=0}^{N-1} C_m \cos[\pi(2m+1)j/2N]. \tag{A9}$$

Since

$$\sum_{j=-N}^{N} \cos[\pi(2m_1+1)j/2N] \times \cos[\pi(2m_2+1)j/2N] = N\delta_{m_1,m_2}, \tag{A10}$$

we get

$$C_m = \frac{1}{N} \sum_{j=-N}^{N} \delta_{j,0} \cos[\pi(2m+1)j/2N] = \frac{1}{N} \tag{A11}$$

for all m, including 0, and therefore

$$\gamma_{\text{opt}} = \frac{2D}{Na^2} \sin(\pi/2N) \exp[\alpha_d(k = \pi/2Na)\tau] = \frac{2DN \sin(\pi/2N)}{b^2[2/3 + (1/3)\cos(\pi/2N)]}. \tag{A12}$$

For the ordinary algorithm, the situation is slightly more complicated, because $\text{Re}[\alpha_d(k)]$ is no longer monotonic; as a result, $\text{Re}[\alpha_d(k_{N-1})] = \text{Re}[\alpha_d(k_0)]$ and two terms of equal magnitude survive at large time:

$$n_j(t) \simeq C_0 \cos\left(\frac{\pi j}{2N}\right) \exp\left[-\alpha_d\left(k = \frac{\pi}{2Na}\right)t\right] + C_{N-1} \cos\left(\frac{\pi(2N-1)j}{2N}\right) \times \exp\left[-\alpha_d\left(k = \frac{\pi(2N-1)}{2Na}\right)t\right]. \tag{A13}$$

The first-passage rate is

$$r(t) \simeq \frac{2D}{Na^2} \sin\left(\frac{\pi}{2N}\right) \left\{ \exp\left[-\alpha_d\left(k = \frac{\pi}{2Na}\right)(t-\tau)\right] + (-1)^{N+1} \exp\left[-\alpha_d\left(k = \frac{\pi(2N-1)}{2Na}\right)(t-\tau)\right] \right\}, \tag{A14}$$

where we have used Eq. (A11) for C_m (that is still valid for the ordinary algorithm). According to Eq. (19) with p and τ

given by Eqs. (1)–(3),

$$\tau\alpha_d\left(k=\frac{\pi}{2Na}\right)=-\ln\cos\left(\frac{\pi}{2N}\right), \tag{A15}$$

$$\tau\alpha_d\left(k=\frac{\pi(2N-1)}{2Na}\right)=-\ln\left[-\cos\left(\frac{\pi}{2N}\right)\right]$$
$$=\tau\alpha_d\left(k=\frac{\pi}{2Na}\right)-i\pi, \tag{A16}$$

so Eq. (A14) becomes

$$r(t)\simeq\frac{2D}{Na^2}\sin\left(\frac{\pi}{2N}\right)\exp\left[-\alpha_d\left(k=\frac{\pi}{2Na}\right)(t-\tau)\right]$$
$$\times[1+(-1)^{N+M}], \tag{A17}$$

where M is the number of the time step ($t=M\tau$). The expression in the rightmost square brackets alternates between 0 and 2, which represents a familiar feature of the ordinary algorithm: No particles reach the boundaries at even steps for odd N and at odd steps for even N. Since we have decided to ignore these oscillations, we simply replace this expression with its average value of unity. Then we immediately obtain for the decay rate and the prefactor

$$\beta_{\text{ord}}=\alpha_d\left(k=\frac{\pi}{2Na}\right)=-\frac{2DN^2}{b^2}\ln\cos\left(\frac{\pi}{2N}\right), \tag{A18}$$

$$\gamma_{\text{ord}}=\frac{2DN}{b^2}\tan\left(\frac{\pi}{2N}\right). \tag{A19}$$

[1] M. Q. López-Salvans, J. Casademunt, G. Iori, and F. Sagués, Physica D **164**, 127 (2002).
[2] M. H. Flamm, S. L. Diamond, and T. Sinno, J. Chem. Phys. **130**, 094904 (2009).
[3] D. Bernstein, Phys. Rev. E **71**, 041103 (2005).
[4] M. J. Saxton, Biophys. J. **70**, 1250 (1996).
[5] Y. Aghababaie, G. I. Menon, and M. Plischke, Phys. Rev. E **59**, 2578 (1999).
[6] C. Keller, F. Marquardt, and C. Bruder, Phys. Rev. E **65**, 041927 (2002).
[7] M. Saadatfar and M. Sahimi, Phys. Rev. E **65**, 036116 (2002).
[8] G. W. Slater and H. L. Guo, Electrophoresis **17**, 977 (1996).
[9] J.-F. Mercier and G. W. Slater, J. Chem. Phys. **113**, 9109 (2000).
[10] J.-F. Mercier and G. W. Slater, Macromolecules **34**, 3437 (2001).
[11] J.-F. Mercier, F. Tessier, and G. W. Slater, Electrophoresis **22**, 2631 (2001).
[12] M. G. Gauthier and G. W. Slater, J. Chem. Phys. **117**, 6745 (2002).
[13] M. G. Gauthier and G. W. Slater, Electrophoresis **24**, 441 (2003).
[14] S. Casault, M. Kenward, and G. W. Slater, Int. J. Pharm. **339**, 91 (2007).
[15] K. Kosmidis, P. Argyrakis, and P. Macheras, Pharm. Res. **20**, 988 (2003).
[16] F. A. Torres, M. G. Gauthier, and G. W. Slater, Phys. Rev. E **78**, 065701(R) (2008).
[17] C. C. Fritsch and J. Langowski, J. Chem. Phys. **133**, 025101 (2010).
[18] M. E. J. Newman and G. T. Barkema, *Monte Carlo Methods in Statistical Physics*, Chap. 3 (Clarendon, Oxford, 1999).
[19] M. G. Gauthier and G. W. Slater, Phys. Rev. E **70**, 015103(R) (2004).
[20] J.-F. Mercier, G. W. Slater, and H. L. Guo, J. Chem. Phys. **110**, 6050 (1999).
[21] C. C. Battaile and D. J. Srolovitz, Annu. Rev. Mater. Res. **32**, 297 (2002).
[22] A. F. Voter, in *Radiation Effects in Solids*, NATO Science Series II: Mathematics, Physics and Chemistry, Vol. 235 (Springer, Dordrecht, The Netherlands, 2007), p. 1.
[23] A. Chatterjee and D. G. Vlachos, J. Comput.-Aided Mater. Des. **14**, 253 (2007).
[24] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in Fortran 77: The Art of Scientific Computing*, Chap. 19 (Cambridge University Press, Cambridge, 1992).
[25] S. Redner, *A Guide to First-Passage Processes* (Cambridge University Press, Cambridge, 2001).
[26] E. W. Montroll and G. H. Weiss, J. Math. Phys. **6**, 167 (1965).
[27] A. Nagar and P. Pradhan, Physica A **320**, 141 (2003).
[28] Also, as noted above, some additional moments of the particle distribution are correct for the direct product algorithm.
[29] C. K. Aidun and J. R. Clausen, Annu. Rev. Fluid Mech. **42**, 439 (2010).
[30] B. Dünweg and A. J. C. Ladd, *Advanced Computer Simulation Approaches for Soft Matter Sciences III*, Advances in Polymer Science, Vol. 221 (Springer-Verlag, Berlin, Heidelberg, 2009), p. 89.
[31] P. Bhatnagar, E. P. Gross, and M. K. Krook, Phys. Rev. **94**, 511 (1954).
[32] X. He and L.-S. Luo, Phys. Rev. E **56**, 6811 (1997).
[33] S. Ponce Dawson, S. Chen, and G. D. Doolen, J. Chem. Phys. **98**, 1514 (1993).
[34] D. Wolf-Gladrow, J. Stat. Phys. **79**, 1023 (1995).
[35] R. G. M. van der Sman and M. H. Ernst, J. Comput. Phys. **160**, 766 (2000).
[36] X. Zhang, A. G. Bengough, L. K. Deeks, J. W. Crawford, and I. M. Young, Water Resour. Res. **38**, 1167 (2002).
[37] I. Ginzburg, Adv. Water Resour. **28**, 1171 (2005).
[38] S. Suga, J. Stat. Phys. **140**, 494 (2010).
[39] R. Mei, L.-S. Luo, P. Lallemand, and D. d'Humières, Comput. Fluids **35**, 855 (2006).
[40] P. van Leemput, W. Vanroose, and D. Roose, Multiscale Model. Simul. **6**, 1234 (2008).
[41] P. C. Philippi, L. A. Hegele Jr., L. O. E. dos Santos, and R. Surmas, Phys. Rev. E **73**, 056702 (2006).
[42] S. S. Chikatamarla and I. V. Karlin, Phys. Rev. E **79**, 046701 (2009).
[43] X. Shan, Phys. Rev. E **81**, 036702 (2010).
[44] R. Cornubert, D. d'Humières, and D. Levermore, Physica D **47**, 241 (1991).

[45] Z. Guo, C. Zheng, and B. Shi, Phys. Fluids **14**, 2007 (2002).

[46] X. Zhang, J. W. Crawford, A. G. Bengough, and I. M. Young, Adv. Water Resour. **25**, 601 (2002).

[47] I. Ginzburg, Adv. Water Resour. **28**, 1196 (2005).

[48] C. Pan, L.-S. Luo, and C. T. Miller, Comput. Fluids **35**, 898 (2006).

[49] J. Latt, B. Chopard, O. Malaspinas, M. Deville, and A. Michler, Phys. Rev. E **77**, 056703 (2008).

[50] M. G. Gauthier and G. W. Slater, J. Chem. Phys. **128**, 065103 (2008).

[51] H. W. de Haan, M. G. Gauthier, M. V. Chubynsky, and G. W. Slater, Comput. Phys. Commun. **182**, 29 (2011).