Fourth-order algorithms for solving the multivariable Langevin equation and the Kramers equation

Harald A. Forbert and Siu A. Chin
Center for Theoretical Physics, Department of Physics, Texas A&M University, College Station, Texas 77843
(Received 15 June 2000; published 27 December 2000)

We develop a fourth-order simulation algorithm for solving the stochastic Langevin equation. The method consists of identifying solvable operators in the Fokker-Planck equation, factorizing the evolution operator for small time steps to fourth order, and implementing the factorization process numerically. A key contribution of this paper is to show how certain double commutators in the factorization process can be simulated in practice. The method is general, applicable to the multivariable case, and systematic, with known procedures for doing fourth-order factorizations. The fourth-order convergence of the resulting algorithm allowed very large time steps to be used. In simulating the Brownian dynamics of 121 Yukawa particles in two dimensions, the converged result of a first-order algorithm can be obtained by using time steps 50 times as large. To further demonstrate the versatility of our method, we derive two new classes of fourth-order algorithms for solving the simpler Kramers equation without requiring the derivative of the force. The convergence of many fourth-order algorithms for solving this equation are compared.

DOI: 10.1103/PhysRevE.63.016703 PACS number(s): 02.70.Rr, 05.40.−a, 02.50.Ey

I. INTRODUCTION

A stochastic differential equation of the form

$$\dot{x}_i = G_i(\mathbf{x}) + h_{ij}\xi_j(t), \tag{1}$$

with Gaussian noise

$$\langle \xi_i(t)\xi_j(t')\rangle = \delta_{ij}\delta(t-t'), \tag{2}$$

is used to describe a variety of physical and chemical processes [1]. We will consider the case where $\mathbf{x}$ denotes an N-dimensional coordinate vector and study the equation in its equivalent Fokker-Planck form

$$\frac{\partial}{\partial t}P(\mathbf{x},t) = LP(\mathbf{x},t) \equiv \left[\frac{1}{2}D_{ij}\partial_i\partial_j - \partial_i G_i(\mathbf{x})\right]P(\mathbf{x},t). \tag{3}$$

The diffusion matrix is given by $D_{ij}=h_{ik}h_{kj}$ [1]. Even in the Langevin case, where the diffusion matrix D_{ij} is position independent, it is difficult to derive numerical algorithms for solving it beyond second order [2–6]. A direct Taylor expansion [2] approach is laborious, giving no insight into the overall structure of the algorithm and requires an eight term expansion to achieve fourth-order accuracy [7]. Heretofore, no fourth-order Langevin algorithm has been derived and applied to systems of more than one particle.

The Fokker-Planck equation (3) can be formally integrated to give

$$P(\mathbf{x},t) = \mathrm{e}^{tL}P(\mathbf{x},0) = [\mathrm{e}^{\epsilon L}]^N P(\mathbf{x},0). \tag{4}$$

This equation can be solved by factorizing the short-time Fokker-Planck evolution operator $\mathrm{e}^{\epsilon L}=\mathrm{e}^{\epsilon(T+D)}$ into exactly solvable parts. While our method has no difficulty in dealing with a general but constant diffusion matrix D_{ij}, to bring out as clearly as possible the character of our approach, we will consider the case $D_{ij}=\delta_{ij}$. The more general case can be easily restored by the interested reader. Thus L consists of two operators

$$T = \frac{1}{2}\partial_i\partial_i \quad \text{and} \quad D = -\partial_i G_i(\mathbf{x}), \tag{5}$$

with implied summations. This idea of operator factorization is not new, and has been used to derive a number of second-order Langevin algorithms [5,6]. We will briefly review the basic idea in Sec. II. However, it is only recently that one learns how to factorize operators of the form $\mathrm{e}^{\epsilon(T+D)}$ to fourth-order with positive coefficients [8,9]. All such fourth-order factorizations require the evaluation of the double commutator $[D,[T,D]]$, which is rather formidable at first sight. We will show in Sec. III, how this commutator can be implemented judiciously to yield a fourth-order Langevin algorithm. To demonstrate the high-order convergence of this algorithm, we use it to simulate the Brownian dynamics of 121 Yukawa particles in two dimensions, a system that has been studied extensively by Branka and Heyes [10] using second-order algorithms.

To further demonstrate the utility of the factorization method for solving stochastic equations, we derive systematically a number of fourth-order algorithms for solving the Kramers equation in Sec. IV. Drozdov and Brey [11] have used a similar factorization method to solve this equation in one dimension using grid points. Hershkovitz [7] has also derived a fourth-order algorithm by Taylor expansion. In both cases, it is not obvious how their respective approaches can be generalized to the multivariable case. We give a detailed comparison of all algorithms using Monte Carlo simulation, which can be easily generalized to any dimension. Finally, we summarize our findings and present some conclusions in Sec. V.

II. OPERATOR FACTORIZATION

When the operator $e^{\epsilon T}$ acts on $P(\mathbf{x},t)$, it evolves the latter forward in time according to the *diffusion* equation

$$\frac{\partial}{\partial t}P(\mathbf{x},t)=\frac{1}{2}\partial_i\partial_i P(\mathbf{x},t). \tag{6}$$

If $\{x_i\}$ is a set of points distributed according to $P(\mathbf{x},t)$, then the distribution ϵ time later can be exactly simulated by updating each point according to

$$x_i'=x_i+\sqrt{\epsilon}\xi_i, \tag{7}$$

where $\{\xi_i\}$ is a set of Gaussian distributed random numbers with zero-mean and unit variance. (For the general diffusion matrix case, the above generalizes to $x_i'=x_i+\sqrt{\epsilon D_{ij}}\xi_j$.)

When the operator $e^{\epsilon D}$ acts on $P(\mathbf{x},t)$, it evolves the latter forward in time according to the *continuity* equation

$$\frac{\partial}{\partial t}P(\mathbf{x},t)=-\partial_i[G_i(\mathbf{x})P(\mathbf{x},t)], \tag{8}$$

where $G_i(\mathbf{x})P(\mathbf{x},t)=J_i(\mathbf{x})$ is the probability current density with velocity field $G_i(\mathbf{x})$. The continuity equation can also be exactly simulated by setting

$$x_i'=x_i(\epsilon), \tag{9}$$

where $x_i(\epsilon)$ is the exact trajectory determined by

$$\frac{d\mathbf{x}}{dt}=\mathbf{G}(\mathbf{x}), \tag{10}$$

with initial condition $x_i(0)=x_i$.

Thus, if $e^{\epsilon(T+D)}$ can be factorized into products of operators $e^{\epsilon T}$ and $e^{\epsilon D}$, then each such factorization will give rise to an algorithm for evolving the system forward for time ϵ. For example, the second-order factorization,

$$e^{1/2\,\epsilon T}e^{\epsilon D}e^{1/2\,\epsilon T}=\exp[\epsilon(T+D)+O(\epsilon^3)\cdots], \tag{11}$$

leads to a second-order Langevin algorithm [5]

$$\begin{aligned} y_i&=x_i+\xi_i\sqrt{\epsilon/2},\\ x_i'&=y_i(\epsilon)+\xi_i'\sqrt{\epsilon/2}, \end{aligned} \tag{12}$$

where ξ_i and ξ_i' are independent sets of zero-mean unit variance Gaussian random numbers. For a second-order algorithm, it is sufficient to solve for the trajectory $y_i(\epsilon)$ correctly to second order in ϵ, e.g., via a second-order Runge-Kutta algorithm:

$$y_i(\epsilon)=y_i+\epsilon G_i\left(\mathbf{y}+\frac{1}{2}\epsilon\mathbf{G}(\mathbf{y})\right). \tag{13}$$

Alternatively, one has the factorization,

$$e^{1/2\,\epsilon D}e^{\epsilon T}e^{1/2\,\epsilon D}=\exp[\epsilon(T+D)+O(\epsilon^3)\cdots], \tag{14}$$

which yields the second-order algorithm

$$\begin{aligned} y_i&=x_i(\epsilon/2)+\xi_i\sqrt{\epsilon},\\ x_i'&=y_i(\epsilon/2). \end{aligned} \tag{15}$$

Again, it is sufficient to solve the trajectory equations $x_i(\epsilon/2)$ and $y_i(\epsilon/2)$ correctly to second-order via the Runge-Kutta algorithm. Despite the appearance that this algorithm requires solving the trajectory equation (10) twice, it can be shown [6] that by expanding the two trajectories to second-order and recollecting terms, one arrives at the second-order Runge-Kutta Langevin algorithm [2–4]. However, the canonical form of Eq. (15), with two evaluations of the trajectory, usually has a much smaller second-order error coefficient.

The method of operator factorization thus appears to provide a systematical way of generating higher-order algorithms. Unfortunately, Suzuki [12] proved in 1991 that, beyond second order, for any two operators, T and D, it is impossible to factorize the evolution operator as

$$\exp[\epsilon(T+D)]=\prod_{i=1}^{N}\exp[a_i\epsilon T]\exp[b_i\epsilon D] \tag{16}$$

for any finite N, without having some coefficients a_i and b_i being negative. In the present context, since $e^{a_i\epsilon T}$ is the diffusion kernel, a negative a_i would imply that one must simulate the diffusion process backward in time, which is impossible. Thus factorizations of the form (16) cannot be used to derive higher-order Langevin algorithms.

III. A FOURTH-ORDER LANGEVIN ALGORITHM

The essence of Suzuki's proof is to note that in order to obtain a fourth-order algorithm, one must eliminate third-order error terms involving double commutators $[T,[D,T]]$ and $[D,[T,D]]$. With purely positive coefficients a_i and b_i, one can eliminate either one or the other, but not both. Thus to obtain a fourth-order factorization with all positive coefficients, one must retain one of the two double commutators. Recently, Chin [9] has derived three such factorization schemes, two of which were also found previously by Suzuki [8].

The form of the operators T and D, as given in Eq. (5), dictates that one should keep only the commutator $[D,[T,D]]$, which is at most a second-order differential operator. Since the velocity (or force) field $\mathbf{G}$ is usually given in terms of a potential function $V(\mathbf{x})$,

$$G_i(\mathbf{x})=-\partial_i V(\mathbf{x}), \tag{17}$$

the double commutator has the form

$$[D,[T,D]]=\partial_i\partial_j f_{i,j}+\partial_i v_i, \tag{18}$$

where

$$f_{i,j}\equiv V_{i,j,k}V_k-2V_{i,k}V_{j,k},$$

$$v_i \equiv \frac{1}{2}(2V_{i,j,k}V_{j,k} + V_{i,j}V_{j,k,k} - V_{i,j,k,k}V_j). \tag{19}$$

The indices on V indicate corresponding partial derivatives. Since the operator D requires solving for the particle's trajectory, we must minimize its occurrence. This dictates that we use a variant of Chin's scheme B [9] to factorize

$$\exp[\epsilon(T+D)] = \exp\left[\frac{\epsilon}{2}\left(1-\frac{1}{\sqrt{3}}\right)T\right]\exp\left(\frac{\epsilon}{2}D\right)\exp\left(\frac{\epsilon}{\sqrt{3}}\tilde{T}\right) \times \exp\left(\frac{\epsilon}{2}D\right)\exp\left[\frac{\epsilon}{2}\left(1-\frac{1}{\sqrt{3}}\right)T\right] + O(\epsilon^5), \tag{20}$$

where we have included the double commutator in $\tilde{T}$

$$\tilde{T} = T + \frac{\epsilon^2}{24}(2\sqrt{3}-3)[D,[T,D]]. \tag{21}$$

To obtain a fourth-order algorithm, we must simulate this new term

$$\exp\left(\frac{\epsilon}{\sqrt{3}}\tilde{T}\right) = \exp\left[\frac{\epsilon}{\sqrt{3}}T + \frac{\epsilon^3}{24}(2-\sqrt{3})(\partial_i\partial_j f_{i,j} + \partial_i v_i)\right] \tag{22}$$

correctly to fourth order. If we simply took all x dependent terms in this operator as fixed, evaluated at the starting point, this operator would describe a nonuniform Gaussian random walk. However, this normal ordering would be correct only to third order. To implement it to fourth order, we first decompose it as

$$\exp\left(\frac{\epsilon}{\sqrt{3}}\tilde{T}\right) = \exp\left(\frac{\epsilon}{2\sqrt{3}}T\right) \times \exp\left[\frac{\epsilon^3}{24}(2-\sqrt{3})(\partial_i\partial_j f_{i,j} + \partial_i v_i)\right] \times \exp\left(\frac{\epsilon}{2\sqrt{3}}T\right) + O(\epsilon^5). \tag{23}$$

If $f_{i,j}$ is positive definite, normal ordering the middle operator above, i.e., interpreting it as a nonuniform Gaussian random walk with $f_{i,j}$ evaluated at the starting point, would be correct to fourth order (actually to fifth order). However, if some eigenvalues of $f_{i,j}$ were negative, we would not be able to sample the operator as a Gaussian walk. To avoid this possibility, we implement the normal order process as follows:

$$\begin{aligned}\exp\left(\frac{\epsilon}{\sqrt{3}}\tilde{T}\right) &= \exp\left(\frac{\epsilon}{2\sqrt{3}}T\right) \times \mathcal{N}\left\{\exp\left[\frac{\epsilon^3}{24}(2-\sqrt{3})(\partial_i\partial_j f_{i,j} + \partial_i v_i)\right]\right\} \times \exp\left(\frac{\epsilon}{2\sqrt{3}}T\right) \\ &= \mathcal{N}\left\{\exp\left[\frac{\epsilon}{2\sqrt{3}}\left(\frac{1}{2}\partial_i\partial_j\delta_{i,j}\right) + \frac{\epsilon^3}{24}(2-\sqrt{3}) \times (\partial_i\partial_j f_{i,j} + \partial_i v_i)\right]\right\}\exp\left(\frac{\epsilon}{2\sqrt{3}}T\right),\end{aligned} \tag{24}$$

where $\mathcal{N}$ denotes the normal ordering of all derivative operators to the left. Since the left (and only the left) operator $\exp(\epsilon/2\sqrt{3}T)$ is already normal ordered with respect to the position-dependent operators in the middle term, the two normal ordered exponentials can be combined to remove the restriction of a positive definite $f_{i,j}$. Now, only the full covariance matrix C needs to be positive definite, which will always be the case for ϵ sufficiently small. The final normal ordered exponential describes a nonuniform Gaussian random walk with mean μ_i and covariance matrix $C_{i,j}$:

$$\mu_i = -\frac{\epsilon^3}{24}(2-\sqrt{3})v_i, \tag{25}$$

$$C_{i,j} = \frac{\epsilon}{2\sqrt{3}}\left[\delta_{i,j} + \left(\frac{1}{\sqrt{3}} - \frac{1}{2}\right)\epsilon^2 f_{i,j}\right]. \tag{26}$$

To sample this random distribution we need $\sqrt{C}$, which we can approximate correctly to fourth order as

$$(\sqrt{C})_{i,j} = \sqrt{\frac{\epsilon}{2\sqrt{3}}}\left[\delta_{i,j} + \frac{1}{2}\left(\frac{1}{\sqrt{3}} - \frac{1}{2}\right)\epsilon^2 f_{i,j}\right]. \tag{27}$$

Thus the entire factorization (20) can be simulated by setting

$$\begin{aligned} w_i &= x_i + \xi_i\sqrt{\frac{\epsilon}{2}\left(1-\frac{1}{\sqrt{3}}\right)}, \\ y_i &= w_i(\epsilon/2) + \xi_i'\sqrt{\frac{\epsilon}{2\sqrt{3}}}, \\ z_i &= y_i - \frac{\epsilon^3}{24}(2-\sqrt{3})v_i(\mathbf{y}) + \sqrt{\frac{\epsilon}{2\sqrt{3}}} \times \left[\delta_{i,j} + \frac{1}{2}\left(\frac{1}{\sqrt{3}} - \frac{1}{2}\right)\epsilon^2 f_{i,j}(\mathbf{y})\right]\xi_j'', \\ x_i' &= z_i(\epsilon/2) + \xi_i'''\sqrt{\frac{\epsilon}{2}\left(1-\frac{1}{\sqrt{3}}\right)}, \end{aligned} \tag{28}$$

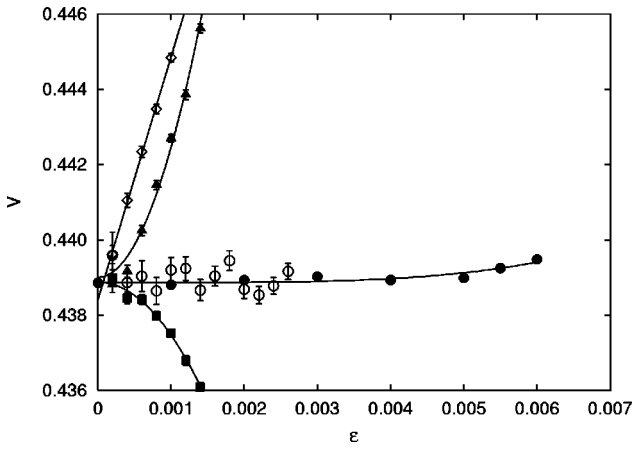

FIG. 1. The convergence of Langevin algorithms for simulating the Brownian dynamics of 121 interacting colloidal particles in two dimensions. The equilibrium potential energy per particle is plotted as a function of the time step size ϵ used. Open diamonds are results using the first-order Langevin algorithm. Solid triangles and solid squares denote results of the two second order algorithms LGV2a and LGV2b, respectively, as described in the text. Open circles give results of our fourth-order Langevin algorithm using the standard fourth-order Runge-Kutta algorithm for determining the particle trajectory. The solid circles give results with improved trajectory determination as discussed in the text.

where ξ_i to ξ_i''' are four sets of independent Gaussian random numbers with zero-mean and unit variance.

As a severe test of the fourth-order convergence of this algorithm, we use it to simulate the Brownian dynamics of 121 colloidal particles in two dimensions, with dimensionless surface density $N/A=0.5$, interacting via a pairwise strongly repulsive Yukawa potential

$$V(r)=\frac{V_0}{r}\exp[-\lambda(r-1)], \tag{29}$$

with $\lambda=8$. This system has been described and simulated extensively via second-order algorithms by Branka and Heyes [10]. We will refer readers to this work for a detailed description of the system and their algorithms. In Fig. 1 we show the convergence of the potential energy at one parameter setting as a function of the time step size used. (Compare this figure to that of Fig. 6 of Branka and Heyes [10].) The linear and quadratic convergences are clearly evident. The two second-order algorithms used are as described by Eqs. (12) and (15). These are referred to in Ref. [6] as algorithms LGV2b and LGV2a, respectively.

When our fourth-order Langevin algorithm is implemented by using the standard fourth-order Runge-Kutta algorithm to solve the trajectory equation (10), we obtained results as shown by open circles in Fig. 1. The variance of the potential energy increases abruptly at around $\epsilon=0.0028$ and the algorithm becomes unstable at larger ϵ's. The problem can be traced to the instability of the Runge-Kutta algorithm itself in solving for the many-body dynamics.While the trajectory evolution $\exp(\epsilon D)$ should *always* decrease the potential energy,

$$\frac{dV}{dt}=\frac{\partial V}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial t}=-|\nabla V|^2, \tag{30}$$

this is no longer respected by the Runge-Kutta algorithm at larger time steps. The failure is due to the fact that Gaussian random walks can deposit particles so close together that the velocity field is changing too steeply for the Runge-Kutta algorithm to integrate accurately. Each of these particles then gets placed chaotically somewhere in the periodic box, often again too near others, thus multiplying the number of particles that will be moved erratically in the next iteration. At time steps below but near $\epsilon=0.0028$, the system can recover the regular behavior after several to hundreds of iterations, but only at the cost of increased variances and larger errors. Thus the inaccuracy in the trajectory determination causes the Langevin algorithm to fail prematurely.

To improve on this situation, we monitor the difference between the results of the standard fourth-order Runge-Kutta and the embedded second-order algorithm (13). We use the absolute value squared of this difference as a gauge of the fourth-order method, even though it is strictly only an error estimate for the embedded second-order algorithm. If the value of this difference is larger than some tolerance (0.01 in our case), we reject the result of the Runge-Kutta and recompute the trajectory more accurately by applying our trajectory algorithm twice at half the time step size. At small time steps, this incurs only a very small overhead. Even at a time step of 0.004, only 3% of the trajectories have to be reevaluated. With this improvement, our fourth-order Langevin algorithm gives results as shown by solid circles in Fig. 1. (We also applied similar monitoring processes to LGV2a and LGV2b by comparing the results of their first- and second-order Runge-Kutta algorithms.) The step-size dependence of the fourth-order algorithm is remarkably flat, and yielded the converged results of the lower algorithms at step sizes nearly 50 times as large.

IV. SOLVING THE KRAMERS EQUATION

While we are not aware of other multivariable fourth-order Langevin algorithms, there are two fourth-order algorithms in the literature for solving the Kramers equation in one dimension [7,11]. Despite its more complicated appearance, the Kramers equation is actually simpler to solve than the Langevin equation. To illustrate the versatility of our operator approach, we will derive systematically a number of fourth-order algorithms for solving this equation. Following Hershkovitz [7], we write the Kramers equations in the form

$$\ddot{q}_i=F_i(\mathbf{q})-\gamma\dot{q}_i+\zeta_i, \tag{31}$$

where the force is derivable from a potential, $F_i(\mathbf{q})=-\partial_i V(\mathbf{q})$. A key simplification follows from the Hamilton form of the equation

$$\dot{q}_i=p_i,$$

$$\dot{p}_i=F_i(\mathbf{q})-\gamma p_i+\zeta_i, \tag{32}$$

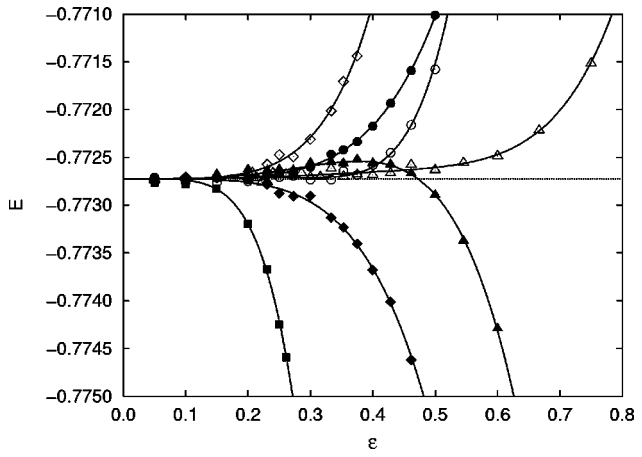

FIG. 2. The convergence of various fourth-order algorithms for solving the Kramers equation in one dimension. The energy calculated is at a finite time of $t=6$ with system parameters $\beta=5$ and $\gamma=1$. Solid squares: Hershkovitz's algorithm. Solid and open circles: Drozdov and Brey's algorithm and K4a. Solid and open diamonds: two variants of algorithm K4b. Solid and open triangles: two variants of algorithm K4c. See text for algorithm descriptions. The fitted lines all have leading term ϵ^4 or higher. Error bars are comparable or smaller than the size of plotting symbols.

where ζ_i is the zero-mean Gaussian random noise vector with variance

$$\langle \zeta_i(t)\zeta_j(t')\rangle = \frac{2}{\beta}\gamma\delta_{ij}\delta(t-t'). \tag{33}$$

The advantage here is that the noise only affects the momentum, and classically, the momentum commutes with the position-dependent force term. We will study the case of the bistable potential

$$V(q)=q^4-2q^2, \tag{34}$$

at parameter value $\gamma=1$ and $\beta=5$. For each algorithm considered below, starting with $q(0)=0$ and $p(0)=0$, we evolve the system to a finite time of $t=6$. For comparison, we note that the total energy approaches the equilibrium limit of $E=-0.8$ at infinite time.

Hershkovitz [7] has formally derived a fourth-order algorithm for solving Eq. (32) using Taylor expansion, but he has given an explicit implementation only for one dimension. In one dimension, each update of his algorithm requires one determination of the particle trajectory to fourth order, four Gaussian random variables, and one evaluation of the derivative of the force. The results of using his algorithm to evolve the system energy as a function of the time step size ϵ is shown as solid squares in Fig. 2. The standard fourth-order Runge-Kutta algorithm, which requires four evaluations of the force, is used to solve for the particle's trajectory.

To derive factorization algorithms in any dimension, we note that the probability density function evolves according to

$$\dot{P}(\mathbf{q},\mathbf{p},t)=LP(\mathbf{q},\mathbf{p},t), \tag{35}$$

where

$$L=\frac{\gamma}{\beta}\nabla_{\mathbf{p}}^2+\gamma\nabla_{\mathbf{p}}\cdot\mathbf{p}-\mathbf{p}\cdot\nabla_{\mathbf{q}}-\mathbf{F}(\mathbf{q})\cdot\nabla_{\mathbf{p}}\equiv L_1+L_2+L_3+L_4. \tag{36}$$

To factorize the evolution operator $\exp(\epsilon L)$ for small ϵ, we decompose L into exactly solvable parts T plus D and apply known fourth-order factorization schemes [8,9]. Drozdov and Brey [11] have recently initiated such a study of the Kramers equation. In this paper, we have done an exhaustive search of all possible choices of solvable T and D such that $[D,[T,D]]$ or $[T,[D,T]]$ is also solvable. We use the word ''solvable'' here loosely to denote either analytical result or trajectory determination. For example, the effect of $\exp[\epsilon(L_2+L_3+L_4)]$ on the distribution function $P(\mathbf{q},\mathbf{p},t)$ corresponds to evolving the particle *trajectory* forward in time with a linear friction. Since this can be computed using any trajectory integration algorithm, we consider $L_2+L_3+L_4$ to be solvable. While there are many solvable choices for T and D, such as the sum of any two L_i, few resulting double commutators are simple. The possible choices for T and D are dramatically reduced if we insist that one of their double commutators is also structurally similar to the original T or D. There are then only three possibilities.

The first possibility is to take

$$\begin{aligned} T&=L_1+L_2+L_3,\\ D&=L_4, \end{aligned} \tag{37}$$

which is the choice originally made by Drozdov and Brey [11]. The Green's function corresponding to $\exp(\epsilon T)$ is known analytically [11], and can be sampled via

$$\begin{aligned} p_i'&=p_i e^{-\gamma\epsilon}+\mu_i,\\ q_i'&=q_i+p_i(1-e^{-\gamma\epsilon})/\gamma+\nu_i, \end{aligned} \tag{38}$$

where corresponding to each pair of (p_i,q_i), (μ_i,ν_i) is a pair of correlated Gaussian random numbers given by

$$\begin{aligned} \mu_i&=\xi_i\sqrt{\frac{1}{\beta}(1-e^{-2\gamma\epsilon})},\\ \nu_i&=\frac{1}{\gamma}\left(\frac{1-e^{-\gamma\epsilon}}{1+e^{-\gamma\epsilon}}\right)\mu_i+\xi_i'\sqrt{\frac{1}{\beta\gamma^2}\left[2\gamma\epsilon-4\left(\frac{1-e^{-\gamma\epsilon}}{1+e^{-\gamma\epsilon}}\right)\right]}. \end{aligned} \tag{39}$$

Here, ξ_i and ξ_i' are again two independent Gaussian random numbers with zero-mean and unit variance. Note that at a given step size ϵ, all the above functions involving $e^{-\gamma\epsilon}$, etc., only need to be evaluated once at the beginning of the simulation. The operator $\exp(\epsilon D)$ can be exactly simulated by

$$p_i'=p_i+\epsilon F_i(\mathbf{q}). \tag{40}$$

As we will see, this choice is clever because there is no trajectory equation to solve. The double commutator required for a fourth-order factorization is

$$[D,[T,D]]=[L_4,[L_3,L_4]]=-\nabla_{\mathbf{q}}|\mathbf{F}|^2\cdot\nabla_{\mathbf{p}} \tag{41}$$

which is just D but with a force $\nabla_{\mathbf{q}}|\mathbf{F}|^2$. For each choice of T and D, there are three generic schemes [9] for factorizing the decomposed operator $\exp[\epsilon(T+D)]$ to fourth order with purely positive coefficients. For this choice of T and D, we found that schemes A and B of Ref. [9] give rather similar results, so we will only present results for schemes A and C. Scheme A and C are, respectively,

$$e^{\epsilon(T+D)}=e^{1/6\epsilon D}e^{1/2\,\epsilon T}e^{2/3\epsilon\tilde{D}}e^{1/2\epsilon T}e^{1/6\epsilon D}+O(\epsilon^5), \tag{42}$$

and

$$e^{\epsilon(T+D)}=e^{1/6\epsilon T}e^{3/8\epsilon D}e^{1/3\epsilon T}e^{1/4\epsilon\tilde{D}}e^{1/3\epsilon T}e^{3/8\epsilon D}e^{1/6\epsilon T}+O(\epsilon^5), \tag{43}$$

where

$$\tilde{D}=D+\frac{\epsilon^2}{48}[D,[T,D]]. \tag{44}$$

The results of these two algorithms are shown as solid and open circles in Fig. 2. We will refer to these two as algorithms, Drozdov and Brey and K4a, respectively. Each algorithm evaluates the force three times and the derivative of the force once. Drozdov and Brey's algorithm uses 4 Gaussian random numbers and K4a uses eight. For the extra effort, algorithm K4a has a much flatter convergence curve. Drozdov and Brey solved their one-dimensional problem on a grid. We used Monte Carlo simulation, which can be generalized to any dimension.

The second possibility is to take

$$T=L_1+L_2, \tag{45}$$

$$D=L_3+L_4. \tag{46}$$

The operator $\exp(\epsilon T)$ now corresponds to an Ornstein-Uhlenbeck process in p_i,

$$p_i'=p_i e^{-\gamma\epsilon}+\xi_i\sqrt{\frac{1}{\beta}(1-e^{-2\gamma\epsilon})}, \tag{47}$$

and $\exp(\epsilon D)$ evolves the particle trajectory forward in time without friction,

$$\begin{aligned} p_i'&=p_i(\epsilon),\\ q_i'&=q_i(\epsilon). \end{aligned} \tag{48}$$

In this case, the simpler double commutator is

$$[T,[D,T]]=[L_2,[D,L_2]]=-\gamma^2 D, \tag{49}$$

which *does not* require the derivative of the force. For this choice, we need to switch $T\leftrightarrow D$ in scheme A and slightly modify it as follows:

$$e^{\epsilon(T+D)}=e^{1/6\epsilon T}e^{1/2\epsilon[1-\epsilon^2\gamma^2/72]D}e^{2/3\epsilon T}e^{1/2\epsilon[1-\epsilon^2\gamma^2/72]D}e^{1/6\epsilon T}+O(\epsilon^5). \tag{50}$$

The effect of the double commutator simply reduces the time of the trajectory evolution. This algorithm, which will be referred to as K4b, requires two trajectory determinations but no derivative of the force and only three Gaussian random numbers. The trajectory can be computed using the standard fourth-order Runge-Kutta algorithm with four force evaluations, or the fourth-order Forest-Ruth symplectic algorithm [13] with three force evaluations. The results from these two cases are plotted as solid and open diamonds, respectively, in Fig. 2. For this choice of D, we did not bother with factorization schemes B or C, since either would have required more than two trajectory determinations.

The third possibility is to take

$$T=L_1, \tag{51}$$

$$D=L_2+L_3+L_4, \tag{52}$$

where now $\exp(\epsilon T)$ is just a Gaussian process in p_i,

$$p_i'=p_i+\zeta_i\sqrt{\epsilon}, \tag{53}$$

and $\exp(\epsilon D)$ evolves the particle trajectory forward in time with friction. For this case, we have the simplest result,

$$[T,[D,T]]=0, \tag{54}$$

and a simplified fourth-order factorization

$$e^{\epsilon(T+D)}=e^{1/6\epsilon T}e^{1/2\epsilon D}e^{2/3\epsilon T}e^{1/2\epsilon D}e^{1/6\epsilon T}+O(\epsilon^5). \tag{55}$$

We shall refer to this as algorithm K4c. This algorithm is similar to K4b, with no force derivative necessary. If we solve the trajectory equation by the fourth-order Runge-Kutta algorithm, we obtain results as shown by solid triangles in Fig. 2. Note that in contrast to previous algorithms, this algorithm does not converge monotonically. It overshoots and converges from the top.

In the course of our calculations, we find that for each algorithm, a more accurately determined particle trajectory will yield a flatter convergence curve. If we now further decompose $D=D_1+D_2$ in algorithm K4c, with

$$\begin{aligned} D_1&=L_2,\\ D_2&=L_3+L_4, \end{aligned} \tag{56}$$

the double commutator $[D_1,[D_2,D_1]]=-\gamma^2 D_2$ is just a restatement of Eq. (49). We can again factorize,

$$e^{\epsilon D}=e^{1/6\epsilon D_1}e^{1/2\epsilon[1-\epsilon^2\gamma^2/72]D_2}e^{2/3\epsilon D_1}e^{1/2\epsilon[1-\epsilon^2\gamma^2/72]D_2}\times e^{1/6\epsilon D_1}+O(\epsilon^5). \tag{57}$$

The friction evolution $e^{\epsilon D_1}$ rescales the momentum,

$$p_i' = p_i e^{-\gamma\epsilon}, \tag{58}$$

and $e^{\epsilon D_2}$ again evolves the trajectory forward for time ϵ. This way of solving the trajectory with friction doubles the number of trajectory calculations, but also further flattens the convergence curve. To minimize the number of force evaluations, we use the Forest-Ruth symplectic algorithm to calculate the trajectory. The results are shown as open triangles in Fig. 2.

Of the algorithms studied, Drozdov and Brey's algorithm makes maximum use of analytical knowledge and is very efficient. The improvement we suggested, algorithm K4a, with twice the number of Gaussian random numbers, seemed to double the range of the convergence. Our new algorithms K4b and K4c, while requiring two trajectory determinations, have no need of evaluating the force derivative. All these algorithms serve to illustrate the power of the factorization method. While the diligence of Hershkovitz is rewarded with just a single fourth-order algorithm, we can survey the form of the evolution operator and derive many fourth-order algorithms.

V. SUMMARY AND CONCLUSIONS

In this paper, we have shown how the method of operator factorization can be applied to the Langevin equation to derive a practical fourth-order algorithm. This method of factorizing an evolution operator of the form $e^{\epsilon(A+B)}$ leads to *unitary* algorithms for solving the Schrödinger equation in quantum mechanics, *symplectic* algorithms for solving Hamilton's equations in classical mechanics, and *norm-preserving* algorithms for solving the Langevin equation in stochastical mechanics. A key step in deriving a fourth-order Langevin algorithm is our treatment of the double commutator term through successive use of normal ordering. The resulting algorithm (28) is computationally demanding, but one is rewarded by a very flat convergence curve, virtually eliminating the step-size dependent error. Future use of this algorithm in other applications may lead to further simplifications and enhancements of its utility.

We also derived a number of fourth-order algorithms for solving the Kramers equation. The freedom in decomposing the kernel operator and choosing a particular factorization scheme illustrates the power of this approach. It is difficult to see these global structures from just doing Taylor expansions. One advantage of our simulation approach is that we are not restricted to solving the Kramers equation in one dimension. We can solve it in any dimension. Our use of the Kramers equation is also only illustrative, one can apply this method of operator factorization to other stochastic equations of one's own interest.

It is observed in solving both equations that the step-size error is reduced by solving the trajectory more exactly. Different fourth-order algorithms for solving the trajectory equation can yield different convergence curves. One should therefore explore the effect of using fourth-order algorithms other than Runge-Kutta in implementing any of the above stochastic algorithms.

ACKNOWLEDGMENTS

This research was funded, in part, by the U. S. National Science Foundation Grant Nos. PHY-9512428, PHY-9870054, and DMR-9509743.

[1] H. Risken, *The Fokker-Planck Equation, Methods of Solution and Applications*, 2nd ed. (Springer, New York, 1989).
[2] E. Helfand, Bell Syst. Tech. J. **58**, 2289 (1979).
[3] I. Drummond *et al.*, Nucl. Phys. B **220**, 119 (1983).
[4] A. Ukawa and M. Fukugita, Phys. Rev. Lett. **55**, 1854 (1985).
[5] S. A. Chin, Nucl. Phys. B **9**, 498 (1989).
[6] S. A. Chin, Phys. Rev. A **42**, 6991 (1990).
[7] E. Hershkovitz, J. Chem. Phys. **108**, 9253 (1998).
[8] M. Suzuki, *Computer Simulation Studies in Condensed Matter Physics VIII*, edited by D. Landau, K. Mon, and H. Shuttler (Springer-Verlag, Berlin, 1996).
[9] S. A. Chin, Phys. Lett. A **226**, 344 (1997).
[10] A. Branka and D. Heyes, Phys. Rev. E **60**, 2381 (1999).
[11] A. N. Drozdov and J. J. Brey, Phys. Rev. E **57**, 1284 (1998).
[12] M. Suzuki, J. Math. Phys. **32**, 400 (1991).
[13] E. Forest and R. D. Ruth, Physica D **43**, 105 (1990).