Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 221 (2007) 1–8
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Quiet direct simulation Monte-Carlo with random timesteps

William Peter *

Johns Hopkins University, Applied Physics Laboratory, 11100 Johns Hopkins Road, Laurel, MD 20723-6099, United States

Received 10 August 2005; received in revised form 15 May 2006; accepted 1 June 2006
Available online 14 August 2006

Abstract

Use of a high-order deterministic sampling technique in direct simulation Monte-Carlo (DSMC) simulations eliminates statistical noise and improves computational performance by orders of magnitude. In this paper it is also shown that if a random timestep is used in place of a fixed timestep, there is an additional improvement in performance. This performance can be increased by using a timestep that samples a random variable with a high-kurtosis probability density function. As a simple example of the method, the one-dimensional diffusion equation with an exponentially-distributed timestep is simulated, and a performance gain of approximately two is obtained. Applications to numerical simulations of fluids and plasmas are indicated.

PACS: 02.50.Ey; 02.70.Tt; 47.11.+j; 52.65.−y

Keywords: Particle-in-cell methods; Direct simulation Monte-Carlo; Stochastic processess

1. Introduction

Finite-difference time domain (FDTD) fluid and particle-in-cell simulations use a fixed timestep $\mathrm{d}t$ to update the field quantities [1,2]. Here I will discuss the advantages of using an exponentially-distributed timestep [3]. Because the timestep $\mathrm{d}t$ is random with an exponential distribution, the exact time $t = t + \mathrm{d}t$ is not *formally* known. The relevant time in a random timestep code is just the expectation value $\langle t \rangle$, which is equivalent to the "sure" time "t" for a fixed timestep code. The motivation for using random timesteps in physics simulation codes is illustrated with a simple example based on the diffusion equation. In general, this random timestep method may be applicable to the wide range of simulation codes (such as fluid and plasma codes) described by the Fokker–Planck equation [4,5].

The direct simulation Monte-Carlo (DSMC) method is the principal computational method for fluids and rarefied flows involving gases and vapors having low-density regions and boundary layers [6–8]. It also has

* Tel.: +1 240 228 3694; fax: +1 240 228 5950.
E-mail addresses: bill.peter@jhuapl.edu, bill.peter@gmail.com.

0021-9991/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2006.06.008

application to the simulation of microelectromechanical systems (MEMS) such as micropumps, microvalves, and microturbines [9]. Although the DSMC method is extensible to any process described by the Fokker–Planck equation, it is deficient as a computational tool since its error is inversely proportional to the square root of the sample size [9–11]. The slow convergence of the method necessitates the use of a large number ($N > 10^3$) of computational particles per grid cell. Hence, DSMC may not be practical for a number of applications.

As a remedy, some recent publications have proposed using the Burnett equations [12], or variance-reduction techniques such as information preservation DSMC (also called DSMC-IP) [13,14] or molecular block DSMC (also called MB-DSMC) [15]. These hybrid techniques are themselves problematic. For example, at large Knudsen number the Burnett equations are not only quite complicated but also unstable to small wavelengths [9]. In addition, results for the MB-DSMC technique have been found to significantly disagree with those of classical DSMC [10].

The quickly converging simulation technique called quiet direct simulation Monte-Carlo (QDSMC) has been shown to have application to plasma and fluid flow [16,17] and other processes described by the Fokker–Planck equation. Because it is based on high-order deterministic sampling, instead of random sampling, no stochastic noise is generated. In this paper, the use of QDSMC with an exponential timestep is introduced. It is shown that even for the simple example of one-dimensional diffusion, simulation times can be improved over QDSMC by at least a factor of two.

2. Quiet direct simulation Monte-Carlo

The approach of quiet direct simulation Monte-Carlo (QDSMC) simulations is to replace the stochastic advance of a particle in phase space by a deterministic sampling chosen to preserve the low order moments of the normal random variable $N(0,1)$. For example, a normal (Weiner) update of a particle initially at a position $x(t)$ is defined by

$$x(t+\mathrm{d}t) = x(t) + \sqrt{2D\,\mathrm{d}t}\;\; N(0,1) \tag{1}$$

where $\mathrm{d}t$ is a fixed timestep, and $N(0,1)$ is a normal random variable with zero mean and unit variance [5]. Since the probability density defined by $N(0,1)$ is given by $p(x) = (2\pi)^{-1/2}\exp(-x^2/2)$, QDSMC uses weights w_j and abscissas q_j for which the Gaussian quadrature approximation [16–21]

$$\int_{-\infty}^{\infty} \frac{dx}{\sqrt{2\pi}} \mathrm{e}^{-x^2/2} f(x) = \sum_{j=1}^{J} w_j f(q_j) \tag{2}$$

becomes exact when the function $f(x)$ is a linear combination of the $2J-1$ polynomials $x^0, x^1, \ldots, x^{2J-1}$. The quantities q_j and w_j are known as Gauss–Hermite parameters [19–21].

Unlike lower-order quadrature schemes like Simpson's method, which involve evaluating $f(x)$ at evenly spaced points x_j on the grid, Gaussian quadrature is a high-order integration scheme for which the sampling points and their corresponding weights are chosen to satisfy Eq. (2) [19,20]. In comparison, Monte-Carlo integration is a low-order scheme for which the sampling points are chosen randomly, and the weights $w_j = 1/N$ are equal.

The equivalent to Eq. (1) using Gaussian quadrature is

$$x(t+\mathrm{d}t) = x(t) + \sqrt{2D\,\mathrm{d}t}\; q_i \tag{3}$$

When calculating expectation values with Eq. (3), each abscissa q_i is weighted by its corresponding weight w_i.

3. Using random timesteps to push particles

A normal update $x(t) \to x(t+\mathrm{d}t)$ defined by Eq. (1) uses samples realized from a normal random variable. It might be advantageous in some cases to use a different probability density function. For example, consider the case where the magnitude of the timestep $\mathrm{d}t$ is not fixed, but distributed exponentially with a "rate" λ, so that the probability density is $p(t) = \lambda\exp(-\lambda t)$ [4,5]. The probability that $\mathrm{d}t > t$ is

$$\Pr[\mathrm{d}t > t] = \int_t^\infty p(t)\,\mathrm{d}t = \exp(-\lambda t) \tag{4}$$

Consider now a normal (Weiner) increment $W_{\mathrm{d}t}$ beginning at the origin at $t = 0$. Following Jansons and Lythe [3], we calculate the probability that after a timestep $\mathrm{d}t$, $W_{\mathrm{d}t} > x$. This is

$$\Pr[W_{\mathrm{d}t} > x] = \int_0^\infty \mathrm{d}t\; p(t) \int_0^\infty \mathrm{d}x\; G(x,t|0,0) = \int_0^\infty \mathrm{d}t\; p(t)\left[\frac{1}{2} - \frac{1}{2}\mathrm{erf}(x/\sqrt{4Dt})\right] \tag{5}$$

where

$$\mathrm{erf}(x) = \frac{2}{\pi}\int_0^x \exp(-y^2)\,\mathrm{d}y. \tag{6}$$

Performing the integration in Eq. (5), one finds that

$$\Pr[W_{\mathrm{d}t} > x] = \Pr[W_{\mathrm{d}t} < -x] = \frac{1}{2}\exp(-x\sqrt{\lambda/D}) \tag{7}$$

Hence, $W_{\mathrm{d}t}$ has a symmetric exponential, or *bi-exponential distribution*. Compared to a Gaussian, this distribution is highly peaked near the origin. Note that the exact value of $\mathrm{d}t$ is not *formally* known, although the mean timestep is

$$\langle \mathrm{d}t \rangle = 1/\lambda \tag{8}$$

The elapsed time after N steps is a random variable with mean N/λ. This time $\langle t \rangle = N/\lambda$ is analogous to the time $t = N\mathrm{d}t$ for a fixed-timestep simulation. An increment equivalent to the normal fixed-timestep case in Eq. (1) is now [3]

$$x(t + \mathrm{d}t) = x(t) + \sqrt{D/\lambda}\,\mathbf{s}\mathbf{p} \tag{9}$$

where $\mathbf{s}$ is a random variable that can take the values $+1$ or -1, and $\mathbf{p}$ is an exponentially-distributed random variable defined in Eq. (4).

Consider now the case of QDSMC for a bi-exponentially distributed random timestep. Because the probability density function is now symmetric exponential, Gaussian quadrature is determined by Gauss—Laguerre abscissas and weights [19,21]

$$\int_{-\infty}^{\infty} \frac{\mathrm{d}x}{\sqrt{2\pi}} \mathrm{e}^{-x\sqrt{\lambda/D}} f(x) = \sum_{j=1}^{J} w_j f(q_j) \tag{10}$$

Sample paths are then generated by

$$x(t + \mathrm{d}t) = x(t) + \sqrt{D/\lambda}\; q_i \tag{11}$$

where each abscissa q_i is understood to represent the symmetric pair $(q_i, -q_i)$.

Formally, Eq. (9) with a symmetric exponential random variable replaces Eq. (1) with a normal random variable. The interpretation of Eq. (9) is that it describes a process with a random time step (with a bi-exponential distribution), but it could also be interpreted as describing a constant time step and random displacement (generated with a bi-exponential random variable).

4. Summary of numerical results and conclusions

As a numerical example, assume an initial line mass $f(x) = 1$ situated at $-8 \leqslant x \leqslant 8$ on a one-dimensional grid defined in the region $-24 \leqslant x \leqslant 24$. The time dependence of the mass function $f(x)$ is described by the one-dimensional diffusion equation

$$\frac{\partial f}{\partial t} = D\frac{\partial^2 f}{\partial x^2} \tag{12}$$

with $D = 1$. The total number of grid cells is taken to be $N_c = 192$, giving a cell spacing $L = 48/192 = 0.25$. A plot of the mass function $f(x)$ at $t = 0$ is shown at the top of Fig. 1, and the analytic solution of $f(x)$ at $t = 19$ is shown at the bottom of Fig. 1.

Numerical integration of the diffusion equation is done by particle-in-cell (PIC) techniques [2], and proceeds as follows: at some initial time t_0, N particles are created at each grid point x_i of a spatial mesh where the mass function $f(x_i, t_0)$ is defined. Each particle created at x_i carries a fraction w_i of the total mass $f(x_i, t)$ at the grid point x_i. Then each particle is displaced from $x(t) = x_i$ to $x(t + \mathrm{d}t)$ according to a given Weiner increment. The particle's new position on the numerical grid is then used to weight its mass back to the grid, and the particle is destroyed. Linear weighting (also called area weighting) is used to assign the particle mass to the mesh [2]. A new set of N particles is created at the next timestep to further advance the mass.

For comparison of the different integration techniques, a PIC simulation code was written to handle each of the following four cases: (a) the classic DSMC method based on Eq. (1) with random samples of a normal variable, (b) the DSMC method based on Eq. (9) with random samples of a bi-exponential, (c) the QDSMC method based on Eq. (3) with deterministic samples of a normal variable, and (d) the QDSMC method with random timesteps based on Eq. (11) with deterministic samples of a bi-exponential. Each case was implemented as a distinct method in a C++ particle class. The simulations were conducted on a variety of GNU/Linux-based workstations. Average run times were obtained using the GNU compiler g++ and timing utility *time* [22].

Plots of the mass function $f(x)$ for the classic DSMC algorithm, Method (a), are shown in Fig. 2 for the two cases: 4 particles/cell (top), and 4000 particles/cell (bottom). The results are obtained by running the simulations up to the final time $t = 19$, and plotting the values of $f(x)$ calculated on the numerical grid. The timestep was $\mathrm{d}t = 0.05$. The normal random variable in Eq. (1) was sampled with the standard Box–Muller algorithm [21]. In comparing Fig. 2 with the analytic solution for $f(x)$ at the bottom of Fig. 1, it is seen that the significant statistical noise in the DSMC method requires a minimum of thousands of simulation particles for sufficiently accurate results.

Method (b), the exponential random timestep for DSMC described by Eq. (9), showed similar accuracy and runtime characteristics to standard DSMC. Both cases required thousands of particles/cell for reasonable fidelity, and had similar run times. Hence, there is no advantage in using a random timestep method in classical DSMC simulations.

The dynamic range and accuracy of the Quiet DSMC approach, Method (c), allows it to quickly converge to the correct solution with as little as 2–4 particles/cell. At the top of Fig. 3 is shown a plot of $f(x)$ at $t = 19$ for a Quiet DSMC simulation with $\mathrm{d}t = 0.05$ and with 4 particles/cell. It should be compared with the corre-

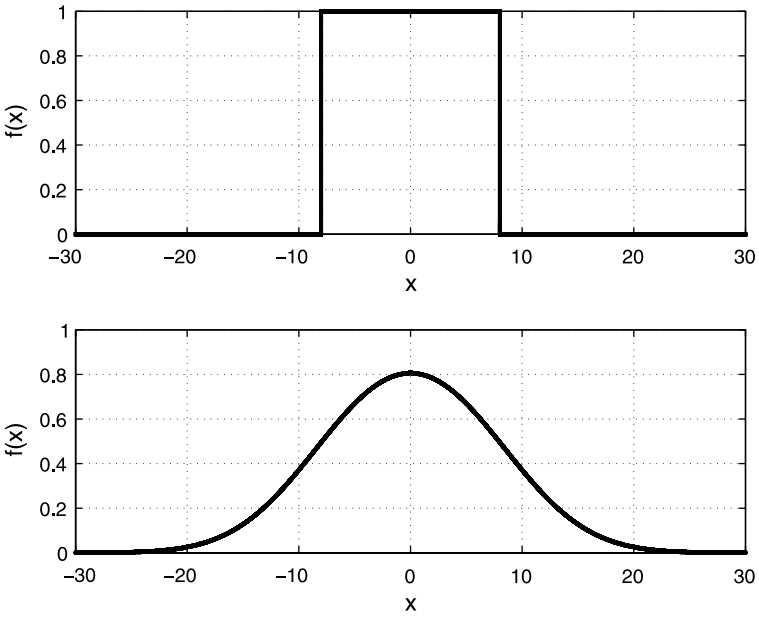

Fig. 1. Exact solution to the one-dimensional diffusion equation with an initial unit line mass at $-8 \leqslant x \leqslant 8$. The top plot corresponds to the initial mass distribution at $t = 0$ and the bottom plot corresponds to the calculated mass distribution at $t = 19$.

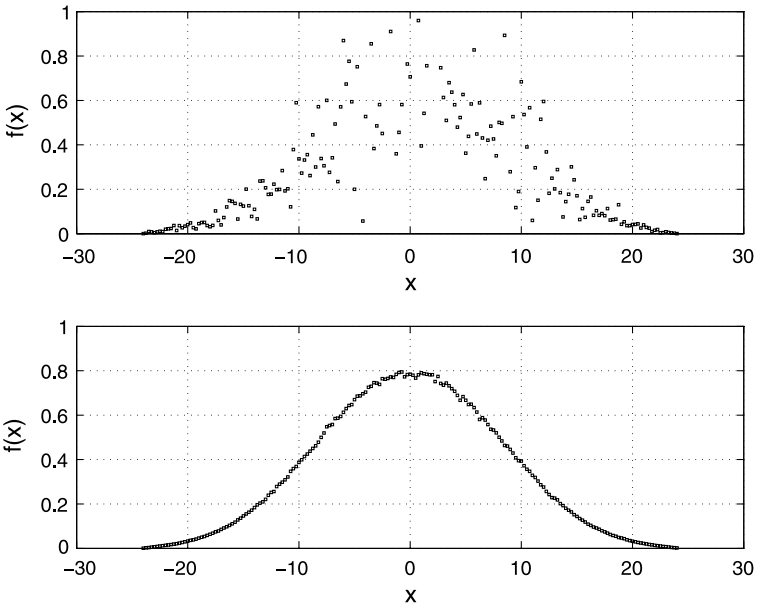

Fig. 2. Plots of the mass function $f(x)$ evaluated at the grid points of a DSMC simulation with $dt = 0.05$ and a final simulation time of $t = 19$. The upper plot used 4 particles/cell and is extremely noisy. The lower plot used 4000 particles/cell, is hundreds of times slower, and still exhibits some statistical noise.

sponding DSMC plot with 4000 particles/cell at the bottom of Fig. 2. With 4 particles/cell the deterministic sampling technique of Method (c) was many hundreds of times faster, and more accurate, than DSMC simulations with 4000 particles/cell. At the bottom of Fig. 3 is a plot of the mass function from a Quiet DSMC

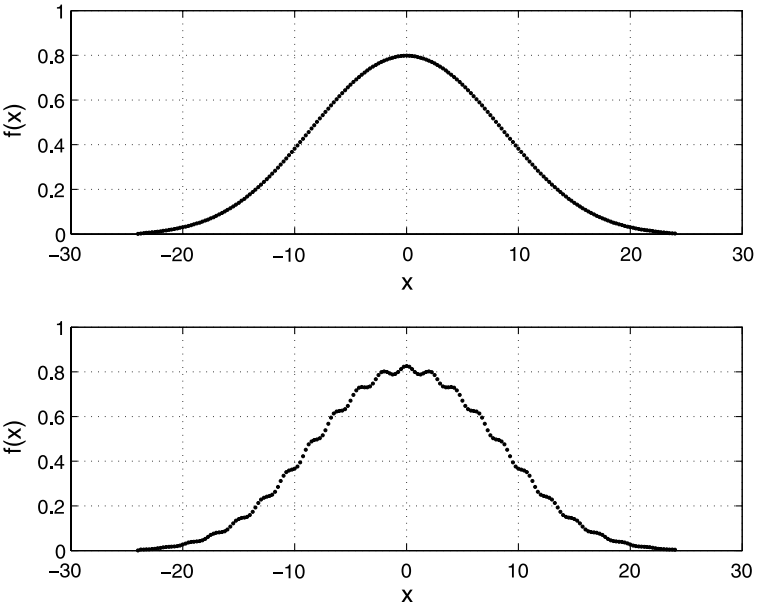

Fig. 3. Simulation results for the mass function $f(x)$ at $t = 19$ using the fixed-timestep Quiet DSMC method with 4 particles/cell. The top plot was run with a timestep $dt = 0.05$ and the bottom plot corresponds to $dt = 1.0$. Compare the upper plot to the equivalent DSMC case using 4000 particles/cell (the bottom of Fig. 2). At a larger timestep $dt = 1.0$, the simulation has lost fidelity, as shown in the lower plot.

simulation with a much larger timestep ($dt = 1.0$). In this case, the simulation has clearly lost accuracy, and will be discussed in more detail below.

In Method (d), the QDSMC algorithm with random timesteps, the update is given by Eq. (11) instead of the Gauss–Hermite form given by Eq. (3). Simulation times for Method (d) were not significantly faster than that of Method (c) with the same timestep. However, simulations using Method (d) followed the solution more accurately at larger timesteps than those using Method (c). Compare the simulation results of Method (c) shown in Fig. 3 with those of Method (d) in Fig. 4. For $dt = 0.05$ both methods give similar results for $f(x)$. For $dt = 1.0$, however, the solution for $f(x)$ using Method (c) becomes unphysical while the solution for Method (d) still retains considerable fidelity.

In the numerical runs for Quiet DSMC, the random timestep method consistently gave better results than the fixed-timestep case *even at larger timesteps*. For the present example, the fixed-timestep method began to noticeably lose fidelity at a timestep $dt \sim 0.5$, while the random timestep method lost similar accuracy at $dt \sim 1.5$. The capability of using larger timesteps in Method (d) renders this method more computationally efficient than Method (c).

Why does the random timestep method retain its fidelity at larger timesteps than the fixed timestep method? Timesteps are restricted in DSMC codes by a requirement that mass elements cannot move a distance greater than a cell length in one timestep. Requiring particles to remain long enough within a cell ensures that particle information is properly distributed within the computational grid, and the particles within a cell are able to interact. If L is the length of a grid cell, this means that a particle cannot travel a distance $\Delta x > L$ in a timestep Δt. This timestep restriction is not really a Courant condition, but is actually a requirement on the fidelity of the solution. That is, violation of the condition may yield solutions that are not physically realistic.

For DSMC simulations, $\Delta x \sim (2D\Delta t)^{1/2}$ from Eq. (1), so the maximum timestep must satisfy $\Delta t \sim L^2/2D$. This is just the Einstein-Smoluchowski equation for a Brownian particle with mean step distance L and mean time Δt between steps. In DSMC simulations the cell length L is typically identified with the particle mean free path. At a timestep of $dt = 0.05$, the step distance in the DSMC simulations of Method (a) with $D = 1$ is $\Delta x \sim (2D\,dt)^{1/2} N(0,1) \sim 0.316\ N(0,1)$. Since the expected value of the normal random variable $N(0,1)$ is zero, and since $L = 0.25$ in our example, the relation $\Delta x < L$ can be satisfied for a significant number of Monte-Carlo realizations.

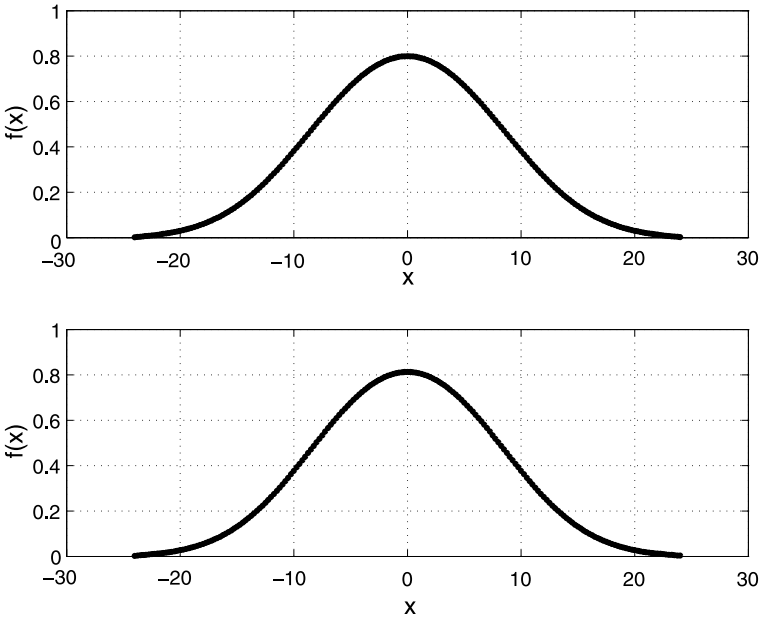

Fig. 4. Simulation results at $\langle t \rangle = 19$ using the random-timestep Quiet DSMC method. The top figure corresponds to a mean timestep $\langle dt \rangle = 0.05$, and the bottom figure to $\langle dt \rangle = 1.0$. Compare the lower plot to the corresponding fixed-timestep plot with $dt = 1.0$ (bottom of Fig. 3).

For Quiet DSMC simulations with fixed timesteps and 4 particles/cell, the Gauss–Hermite abscissas are $q_i = \pm 0.742$ and $q_i = \pm 2.33$ (see Ref. [19]). Hence, from Eq. (3), for a timestep $\mathrm{d}t = 0.05$, there are a pair of steps with $\Delta x = \pm 0.235$ and a pair of steps with $\Delta x = \pm 0.738$. Although the two particles with steps $|\Delta x| = 0.738 > L$ traverse a distance greater than a cell length in one timestep, the other two particles with $|\Delta x| = 0.235 < L$ seem to be able to "regulate" the fidelity of the simulation. For the larger timestep $\mathrm{d}t = 1.0$, the step distances are $\Delta x = \pm 1.05$ and $\Delta x = \pm 3.30$. These step distances are considerably larger than the cell size $L = 0.25$ and the simulation (although mathematically stable) becomes inaccurate and unphysical, as shown at the bottom of Fig. 3.

For Quiet DSMC simulations with exponentially-distributed random timesteps, the position updates are obtained from Eq. (11) using Gauss—Laguerre abscissas. In the case of 4 particles/cell, the symmetric $\pm$ pair of the first two Gauss—Laguerre abscissas (0.586 and 3.41) are used. For an average timestep $\langle \mathrm{d}t \rangle = 0.05$, the step sizes are then $\Delta x = \pm 0.131$ and ± 0.763, so that the two particles with $|\Delta x| = 0.131 < L$ are able to "regulate" the accuracy of the simulation. For the timestep $\langle \mathrm{d}t \rangle = 1.0$, the step sizes are $\Delta x = \pm 0.586$ and ± 3.41. Although these step distances are larger than the cell size $L = 0.25$, there was only a minor (and unnoticeable) loss in accuracy, as seen at the bottom of Fig. 4. Note that the step size $\Delta x = 0.586$ for $\mathrm{d}t = 1.0$ is one-half the value ($\Delta x = 1.05$) of the step size for the fixed-timestep case. In fact, because of the relative magnitudes of the step sizes for the two cases, the random timestep method seemed to have similar accuracy to the fixed-timestep method with timesteps a factor ~2–4 times larger.

Although both the fixed timestep and random timestep method use Gaussian quadrature, the random timestep method takes "deterministic samples" of a distribution that is more strongly peaked near the origin. That is, particles that are sampled from the exponential distribution will have some step distances that are less (and some that are more) than the corresponding particles from a normal distribution. Those particles that have smaller step distances $\Delta x < L$ are able to control the accuracy of the solution. One can compare the magnitudes of the smallest QDSMC step size $\Delta x_{\mathrm{fixed}}$ for a fixed timestep to the smallest step size $\Delta x_{\mathrm{random}}$ using a random timestep. For the case above ($\mathrm{d}t = 1.0$), $\Delta x_{\mathrm{random}} \sim 0.586$, while $\Delta x_{\mathrm{fixed}} \sim 1.05$. The ratio of these steps is essentially the allowable increase in timestep for the random-timestep simulation over the fixed-timestep case, or a factor $\Delta x_{\mathrm{fixed}}/\Delta x_{\mathrm{random}} \sim 2$.

The reason for the performance gain is that the distribution function of an exponentially-distributed random variable is more sharply peaked near the origin than that of a normally-distributed random variable. In other words, the symmetric exponential distribution (sometimes called the bi-exponential distribution) has a higher *kurtosis* than a normal distribution [5]. Hence, deterministic samples of the symmetric exponential distribution defined in Eq. (7) have some values less, and some values greater, than the corresponding samples of a normal distribution. These low-value samples have smaller step sizes which are more likely to satisfy the accuracy condition $\Delta x < L$, and are thus able to "regulate" the fidelity of the solution in cases for which the deterministic samples from the normal distribution cannot.

An important extension to the present study would be to apply random timestepping to plasma or fluid simulations. A straightforward way of doing this would to transform the timestep in the field equations to a random timestep as done in Eq. (4). For example, consider the applicability of the random timestep method to fluid simulations using QDSMC as described in [16]. The fluid equations are derived from the first three moments of the Fokker–Planck kinetic equation, which is itself equivalent to the Ornstein–Uhlenbeck (O–U) process equations [23]. The O–U process consists of a diffusive process (as discussed here) with the addition of a drift term. Using operator splitting as described in [16], the thermalization term (which describes the relaxation of the particle velocity to the local fluid velocity) would now include a bi-exponential probability density function instead of a Gaussian. The transport term need not undergo any changes. Again, the reason for this is simply that the change in the thermalization term can be interpreted as an update with the same fixed timestep $\mathrm{d}t$ as used in the transport term, but now with a random displacement described by a bi-exponential probability density function. Such an analysis could be presented in future work.

In summary, when applied to Quiet DSMC simulations, the random timestep method can be run at timesteps at least double that of fixed-timestep simulations for the example of the one-dimensional diffusion equation. Preliminary two- and three-dimensional simulations seem to indicate that this performance gain scales roughly linearly as the number of dimensions increase. Hence, 2-d is roughly four times faster, and 3-d is roughly six times faster. Although this performance gain has only been shown for the one-dimensional diffu-

sion equation (which is relevant to a wide number of applications), the concept of accelerating QDSMC by deterministic sampling of a probability density function of high kurtosis has been demonstrated.

Finally, another benefit in using an exponentially-distributed random timestep over a fixed timestep is that the intersection of a particle path with a given point can be determined exactly from excursion theory [24,25]. This may be important for simulations involving special boundary conditions. For example, consider the case of gas or fluid motion in a region bounded by a fixed wall. Using DSMC, the particle position $x(t)$ is known initially at t, and, by the update Eq. (1), at a time $t + \mathrm{d}t$ later. There are cases for which the boundary was penetrated by the particle between t and $t + \mathrm{d}t$, although at the end of the timestep the particle position $x(t + \mathrm{d}t)$ is on the same side of the boundary as $x(t)$ [25]. This case cannot be determined exactly by using fixed timestep methods, but can be determined from excursion theory for exponentially-distributed timesteps.

Acknowledgements

I would like to thank Don Lemons of Bethel College and Brian Albright of Los Alamos National Laboratory for illuminating discussions. I also acknowledge the helpful suggestions of the referees in improving the content of the final manuscript.

References

[1] T. Tajima, Computational Plasma Physics: With Applications to Fusion and Astrophysics, Addison-Wesley, New York, 1989.
[2] C.K. Birdsall, A.B. Langdon, Plasma Physics Via Computer Simulation, Adam Hilger, Bristol, 1991.
[3] K. Jansons, G.D. Lythe, J. Statist. Phys. 100 (2000) 1097.
[4] C.W. Gardiner, Handbook of Stochastic Methods, second ed., Springer, New York, 1985, pp. 106–107.
[5] D.S. Lemons, An Introduction to Stochastic Processes in Physics, Johns Hopkins, Baltimore, 2002.
[6] G.A. Bird, Phys. Fluids 6 (1963) 1518;
G.A. Bird, Ann. Rev. Fluid Mech. 10 (1978) 11.
[7] E.P. Muntz, Ann. Rev. Fluid Mech. 21 (1989) 387.
[8] D.I. Pullin, J. Comput. Phys. 34 (1980) 231.
[9] S. Roy, R. Raju, H.F. Chuang, B.A. Cruden, M. Meyyappan, J. Appl. Phys. 93 (2003) 4870.
[10] M. Wang, Z. Li, Phys. Fluids 16 (2004) 2122.
[11] E.Y.-K. Ng, N. Liu, J. Micromech. Microeng. 12 (2002) 567.
[12] D. Burnett, Proc. London Math. Soc. 40 (1935) 382.
[13] J. Fan, C. Shen, J. Comput. Phys. 167 (2001) 393.
[14] C. Cai, I.D. Boyd, J. Fan, J. Thermophys. Heat Transfer 14 (2000) 368.
[15] L.S. Pan, G.R. Liu, B.C. Khoo, B. Song, J. Micromech. Microeng. 11 (2001) 181.
[16] B.J. Albright, D.S. Lemons, M.E. Jones, D. Winske, Phys. Rev. E. 65 (2002) 055302/1-4.
[17] B.J. Albright, W. Daughton, D.S. Lemons, D. Winske, M.E. Jones, Phys. Plasmas 9 (2002) 1898.
[18] A.L. Garcia, Numerical Methods for Physics, second ed., Prentice-Hall, Upper Saddle River, NJ, 2000.
[19] M. Abramowitz, I. Stegun, Handbook of Mathematical Functions, Dover, New York, 1972.
[20] C. Lanczos, Applied Analysis, Prentice Hall, Englewood Cliffs, N.J., 1956.
[21] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipes in C: The Art of Scientific Computing, second ed., Cambridge University Press, Cambridge, 1992.
[22] More information on these utilities can be obtained from http://www.gnu.org.
[23] D.S. Lemons, B.J. Albright, J. Quant. Spectrosc. Radiat. Transfer 74 (2002) 719.
[24] S. Karlin, H.M. Taylor, A First Course in Stochastic Processes, second ed., Academic Press, New York, 1975.
[25] K. Jansons, G.D. Lythe, Siam J. Sci. Comput. 24 (2003) 1809.