Journal of Computational Physics **173,** 231–255 (2001)
doi:10.1006/jcph.2001.6874, available online at http://www.idealibrary.com on IDEAL®

Accelerating Convergence in Stochastic Particle Dispersion Simulation Codes

Richard R. Picard,[1] Mark Fitzgerald,[1,2] and Michael J. Brown

Los Alamos National Laboratory, Los Alamos, New Mexico 87545
E-mail: picard@lanl.gov, fitz@math.cudenver.edu, mbrown@lanl.gov

Received August 25, 2000; revised July 2, 2001

Recent work in adaptive importance sampling is applied to Markov chain models for Monte Carlo simulations. When this technique is incorporated into the simulation of physical processes, it can give orders-of-magnitude improvement in convergence times relative to standard approaches. We review the related methodology and illustrate its application.

Key Words: exponential convergence; importance sampling; Markov chains; Monte Carlo methods.

1. BACKGROUND

Random-walk models have long been used by researchers for turbulent diffusion calculations. In this approach, the trajectories of particles or tagged fluid elements are tracked as they evolve. Successful applications, especially in regard to particle dispersion in the atmosphere, range from early efforts by Lamb [16] and Reid [22] and continue to more recent work such as by Borgas *et al*. [6].

Much attention on the use of random-walk models has focused, understandably, on the underlying theory (see van Dop *et al*. [29] and Wilson and Sawford [30] for reviews) and on developing models that accurately represent physical behavior (e.g., Thomsen [26] and Reynolds [23]). Perhaps partly as a consequence, less attention has been paid to computational issues. Because a drawback of these models is that a large number of trajectories must be simulated in order to obtain the desired statistical accuracy in the results, simulation efficiency is important.

To address this computational problem, efforts have been made to increase the time step that can be used by assuming a homogeneous parameterization for turbulence [13] and by

[1] Supported by the Mathematical, Information, and Computational Sciences (MICS) program of the U.S. Department of Energy.

[2] Present address: University of Colorado at Denver, Denver, Colorado 80217.

using a linear-skewed form of the Langevin equation to obtain an exact particle velocity relation [19]. So-called particle puff models also have been adopted in various forms, such as by Hurley [12] and de Haan and Rotach [8], to improve efficiency.

In this paper, we present an approach to accelerating convergence based on adaptive importance sampling. This approach can be used alone or in conjunction with other techniques to substantially improve simulation efficiency. It follows in the vein of recent radiation transport work [5], where, as in turbulent dispersion, conducting controlled experiments is expensive and computer simulation is widely used.

We overcome the need for simulating large numbers of particle trajectories that conform to the model for the natural process by instead simulating *biased* particle trajectories—trajectories that evolve according to stochastic processes that differ from that for the natural model. These stochastic processes are defined using recent theoretical results in adaptive learning algorithms. The novelty of the approach lies in its use of iterative curve fitting in such a way as to produce exponentially convergent acceleration properties. (This phenomenon is explained in Section 3.4.) Because the underlying methodology applies to a large class of computational physics problems that can be modeled as transient Markov chains, its applicability is not limited to specific Monte Carlo codes, such as those for turbulent dispersion.

The bottom line is that uncertainties are greatly reduced. In the idealized examples presented here, simulation efficiency for estimating physical properties of interest is improved by factors of tens to hundreds. To be sure, there are trade-offs involved, in that efficient estimation of a property of interest may correspond to inefficient estimation of another property, but these trade-offs are usually worthwhile.

In Section 2, background for the methodology is described. Illustrative examples are presented in Sections 3 and 4, where we show significant improvement in convergence rates relative to simulation of the natural process. Additional remarks are given in Section 5, and mathematical details are contained in three appendices.

2. IMPORTANCE SAMPLING

Underpinning the methodology is importance sampling, an idea dating back some 50 years [14]. Of late, however, advances in adaptive importance sampling for Markov chains offer the potential to greatly accelerate the convergence of stochastic particle dispersion simulations. We begin by giving the mathematical foundation of importance sampling for simple integrals, then extend the concepts to random-walk models, and finally discuss the role of learning algorithms.

2.1. Simulation Estimates of Integrals

Simulation averages, such as an ensemble mean or a time-average mean, usually correspond to some type of mathematical integration. In this context, importance sampling is often useful (e.g., [24]). It helps to consider the problem of using simulation to estimate a simple integral. Suppose that y is a random variable which behaves according to probability density function $f(y)$, and that it is of interest to estimate the average value of some function

$s(y)$, i.e., to estimate the integral

$$\tilde{I} = \int s(y)\, f(y)\, dy.$$

The straightforward approach to solving the problem via simulation involves generating a set of N independent random variables $\{y_i\}$ according to the probability density function $f(y)$ and averaging the resulting "scores" $\{s(y_i)\}$, obtaining the Monte Carlo estimate

$$\hat{I}_f = \frac{1}{N} \sum_{i=1}^{N} s(y_i). \tag{1}$$

Here, the "hat" notation $\hat{I}$ is used to distinguish a simulation estimate from the theoretical quantity $\tilde{I}$ being estimated. When the sample size N is large enough to produce the desired statistical accuracy, the problem is solved.

On occasion, however, the straightforward approach is computationally inefficient. If the simulation of random variables from the probability density $f(y)$ is slow, or if the simulated scores $\{s(y_i)\}$ are near zero for the most commonly observed $\{y_i\}$ values, then much computation time is required. A way to avoid this circumstance is to choose another probability density function $g(y)$ and rewrite the integral $\tilde{I}$ of interest as

$$\tilde{I} = \int s(y)\, f(y)\, dy = \int \left\{ s(y) \frac{f(y)}{g(y)} \right\} g(y)\, dy. \tag{2}$$

Then, by simulating values $\{y_i\}$ independently from the probability density function $g(y)$, the importance sampling estimate is

$$\hat{I}_g = \frac{1}{N} \sum_{i=1}^{N} \left\{ s(y_i) \frac{f(y_i)}{g(y_i)} \right\}. \tag{3}$$

Loosely speaking, the importance sampling estimate $\hat{I}_g$ takes the values $\{s(y_i)\}$ obtained from the simulation using $g(y)$ and combines them using the weights $\{f(y_i)/g(y_i)\}$ in order to adjust for the biased sampling of the $\{y_i\}$. Under mild regularity conditions on the importance distribution $g(y)$ that preclude division by zero in Eq. (3), the resulting estimates are statistically valid.

2.2. Zero-Variance Importance Sampling

The key to making importance sampling succeed is to choose a good importance distribution $g(y)$. Illustrating an idealized case, suppose $s(y) \geq 0$ and consider the importance distribution $\tilde{g}(y) = s(y)\, f(y)/\tilde{I}$. This choice of $g(y)$ tends to preferentially sample $\{y_i\}$ values which emphasize parts of the space that are "important" to the integral $\tilde{I}$, in contrast to ordinary simulation which produces $\{y_i\}$ values based on their prevalence in the natural distribution described by $f(y)$.

Using the probability density function $\tilde{g}(y) = s(y)\, f(y)/\tilde{I}$ for random number generation, each simulated random variable y_i from $\tilde{g}(y)$ contributes the score

$$s(y_i)[\, f(y_i)/\tilde{g}(y_i)] \equiv \tilde{I}$$

to the importance sampling estimate $\hat{I}_g$ in Eq. (3). That is, the simulation estimate gives the desired answer with no simulation error, a phenomenon that may initially seem hard to believe. This choice of $g(y)$ is known as the *zero-variance solution*.

To illustrate, consider using simulation to estimate the variance σ^2 of a Gaussian random variable having mean zero. The probability density function for this example is

$$f(y) = \frac{1}{\sigma\sqrt{2\pi}} e^{-y^2/2\sigma^2},$$

and the score function of interest is

$$s(y) = y^2.$$

The integral $\tilde{I}$ is simply the parameter σ^2, and thus the zero-variance solution is

$$\tilde{g}(y) = \frac{1}{\sigma^3\sqrt{2\pi}} y^2 e^{-y^2/2\sigma^2}.$$

Figure 1 displays the difference between $f(y)$ and $\tilde{g}(y)$ for the case $\sigma^2 = 19{,}715$ (a value to arise in later examples). The distribution $\tilde{g}(y)$ highlights the values of y most relevant to the integration of $s(y)$.

Of course, construction of the zero-variance solution $\tilde{g}(y) = s(y)\,f(y)/\tilde{I}$ requires knowledge of the integral $\tilde{I}$, which is what we are trying to estimate in the first place. As such, the theoretical zero-variance result may not seem to have much practical value. We show in later sections, however, that learning algorithms based on approximations to the zero-variance solution lead to dramatic efficiency gains.

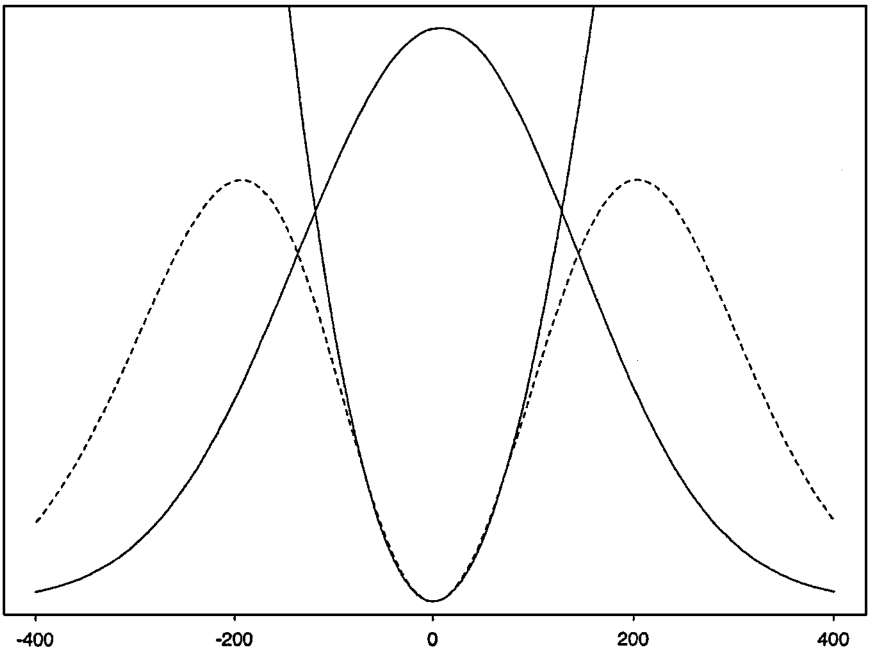

FIG. 1. Solid curves denote the Gaussian probability density function and (scaled) score function $s(y) = y^2$; the dashed curve denotes the zero-variance solution.

2.3. Random-Walk Models

The above discussion can be extended from simple random variables y to the stochastic processes $\{x_n; n = 0, 1, 2, \ldots\}$ used in turbulent particle dispersion. The idea is that each step of a random walk is simulated from a probability density function that differs from its natural counterpart. That is, the direct simulation of a random walk from the model for nature involves producing, at each step, one or more stochastic components that determine how the walk will proceed.

The goal of importance sampling is to judiciously "bias" each step of the random walk to produce particle trajectories that are important to the calculation at hand, thereby reducing uncertainties relative to direct simulation. Although the general concept is by no means new, its practical use has been inhibited by past emphasis on nonadaptive techniques tailored to highly specific applications (such as those in radiation transport) and by assorted issues discussed in Section 5.

More formally, particle movement is treated as a transient Markov chain. The state space for the chain contains all of the information needed to generate each step of the random walk and may include particle location, velocity, direction, and other aspects of interest such as "real" time. The embedded multidimensional structure of the state space is not of interest to this discussion however; thus, the state of the process at step n is simply denoted as x_n.

Simulation of the nth step of the random walk is governed by the transition kernel for the Markov chain, denoted $p(x_{n-1}, x_n)$. In other words, the transition kernel is the probability density function that formalizes, in a stochastic sense, the process of moving from state x_{n-1} at step $n - 1$ to state x_n at step n. Computationally, the state x_n is obtained by simulating from $p(x_{n-1}, x_n)$, and it follows that the probability density function for an entire random walk $\{x_n; n = 0, 1, \ldots, \tau\}$ is the product of the densities for the multiple steps:

$$\prod_{n=1}^{\tau} p(x_{n-1}, x_n).$$

The purpose of most stochastic particle dispersion simulations is to understand aspects of particle movement. In what follows, the quantity of interest is assumed to be a property of an individual particle trajectory. One simple example of such a property is the theoretical centroid of a particle at a specified stage of its random walk. Here, the term "theoretical" refers to the statistical distribution of particle trajectories and can be viewed as being equal to the limit (as the number N of simulated trajectories goes to infinity) of the simulated average.

The theoretical quantity of interest is expressed mathematically as the average of a score which accumulates with each step of the random walk. Let $s(x_{n-1}, x_n)$ denote the score accumulated on step n, and let τ denote the (possibly random) number of steps in the walk. Then the accumulated score for a single random walk $\{x_n\}$ starting in state x_0 and evolving according to the natural transition kernel $p(x_{n-1}, x_n)$ is

$$S(x_0) = \sum_{n=1}^{\tau} s(x_{n-1}, x_n).$$

The corresponding theoretical average with respect to the distribution of all trajectories emanating from x_0 is denoted $\tilde{S}(x_0)$.

For example, quantities of interest could include:

(a) The plume centroid at a specified step $\tilde{n}$. In this case, the score function is $s(x_{n-1}, x_n) = \mathbf{0}$ for all $n \neq \tilde{n}$, and $s(x_{\tilde{n}-1}, x_{\tilde{n}})$ is the location $\boldsymbol{\ell}_{\tilde{n}}$ of the particle trajectory at step $\tilde{n}$.

(b) The second moment of the plume at step $\tilde{n}$ about a specified location $\boldsymbol{\ell}$. Here, scores $s(x_{n-1}, x_n) = \mathbf{0}$ for all $n \neq \tilde{n}$, and $s(x_{\tilde{n}-1}, x_{\tilde{n}})$ is the outer product $(\boldsymbol{\ell}_{\tilde{n}} - \boldsymbol{\ell})(\boldsymbol{\ell}_{\tilde{n}} - \boldsymbol{\ell})^{\mathrm{T}}$.

(c) The portion of released particles that enter a volume V. Here, the score function is $s(x_{n-1}, x_n) = 1$ if the particle arrives in V for the first time at step n. All other scores are zero. Note that, once obtained, this average score can be combined with the release rate and converted to a concentration.

(d) The portion of released particles that are deposited on a surface area A. Here, the score function is $s(x_{n-1}, x_n) = 1$ if the particle is deposited on A at step n. All other scores are zero.

(e) The average elapsed time until an event of interest occurs (at which point the random walk terminates). Here, $s(x_{n-1}, x_n) = (\Delta t)_n$, where $(\Delta t)_n$ is the elapsed clock time associated with the nth time step.

In this paper, we consider score functions $s(x_{n-1}, x_n)$ that are one-dimensional and non-negative. Cases where scores can be negative, such as for particle position, can often be handled by adding an artificial constant to the score function (akin to a Celsius-to-Kelvin temperature conversion).

The straightforward Monte Carlo approach involves generating random walks from state x_0 according to the transition kernel $p(x_{n-1}, x_n)$ for the model of nature, obtaining an accumulated score $S(x_0)$ for each random walk, and then averaging those accumulated scores over the simulation. Averaging the scores $S(x_0)$ from N trajectories originating from x_0 is the stochastic process analogue of Eq. (1) for Monte Carlo estimation of simple integrals.

Improving on the status quo via importance sampling involves choosing a transition kernel $q(x_{n-1}, x_n)$ which differs from the natural kernel $p(x_{n-1}, x_n)$ to bias the steps of the simulated random walk. The accumulated score from a biased random walk for a single particle takes the form

$$S(x_0) = \sum_{n=1}^{\tau} \left\{ s(x_{n-1}, x_n) \frac{\prod_{i=1}^{n} p(x_{i-1}, x_i)}{\prod_{i=1}^{n} q(x_{i-1}, x_i)} \right\}, \tag{4}$$

where the ratio $\prod p(x_{i-1}, x_i) / \prod q(x_{i-1}, x_i)$ in Eq. (4) plays the role of $f(y_i)/g(y_i)$ in Eq. (3). By averaging scores in Eq. (4) from N biased trajectories, the stochastic process analogue of importance sampling for simple integrals is obtained. For some problems, such as when interest lies only in the ultimate state of the process, $s(x_{n-1}, x_n)$ is zero for all but the final step τ of the random walk and Eq. (4) simplifies considerably.

2.4. Learning Algorithms

Adaptive importance sampling is a technique for reducing uncertainties. In the context of Section 2.1 regarding Monte Carlo estimation of simple integrals, the adaptive procedure starts by selecting an initial importance distribution $g(y)$. Ideally, $g(y)$ should resemble the zero-variance solution, to the extent that this can be done given the initial information; alternatively, the probability density $f(y)$ for the model of nature can be used for the initial iteration if a better estimate is not available. Next, random variables $\{y_i\}$ are simulated according to the initial density and the simulated results are used to improve the importance distribution. Simulation then continues with the improved importance distribution, producing additional data for further improvement, and so on. For an early example of such a learning algorithm, based on histogram-type probability density functions, see [17].

In the context of stochastic processes, less research has been done on adaptive methods than for simple estimation of integrals. For stochastic processes, the goal of adaptive importance sampling is to find a good importance sampling transition kernel $q(x_{n-1}, x_n)$. The general approach is the same as for simple integrals, namely, to start with an initial kernel and then use simulation data to improve it.

In what follows, we use the fact that a zero-variance solution exists for transient Markov chains with nonnegative score functions $s(x_{n-1}, x_n)$. Letting $\tilde{S}(x_0)$ denote the theoretical average score for a random walk beginning in state x_0, the zero-variance kernel is [1]

$$\tilde{q}_S(x_{n-1}, x_n) = p(x_{n-1}, x_n)\frac{s(x_{n-1}, x_n) + \tilde{S}(x_n)}{\tilde{S}(x_{n-1})}. \tag{5}$$

As with the zero-variance solution for simple integrals in Section 2.2, Eq. (5) requires knowledge of the answer, here meaning $\tilde{S}(x)$ for all states x, which is nothing short of omniscience.

Although the theoretical quantity $\tilde{S}(x)$ is unknown in practical problems (indeed, the lack of analytical solutions is a major reason for using simulation in the first place), it is often possible to approximate $\tilde{S}(x)$ with a parametric function of x. Such approximation, over all states x, amounts to a type of curve fitting, examples of which are given in Sections 3 and 4. This curve fitting serves as the basis of a flexible, general approach to accelerating convergence.

In idealized cases, where the parametric form of the approximating function contains the exact form as a special case, recent theoretical work for discrete [15] and continuous [1] state spaces has shown that a special class of adaptive learning algorithms converges *exponentially* quickly to the solution. When compared with the $N^{-1/2}$ rate of convergence for ordinary simulation estimates, substantial gains in computational efficiency are achieved. An example of exponential convergence is given in Section 3.4 for the problem of dispersion in homogeneous turbulence; Booth [5] has achieved such convergence in an idealized radiation transport example using less general methods.

To describe the learning algorithm, an initial simulation is carried out in which several particle trajectories are generated based on an initial transition kernel $\hat{q}^{(0)}(x_{n-1}, x_n)$. As noted before, the natural transition kernel $p(x_{n-1}, x_n)$ can serve as the initial kernel $\hat{q}^{(0)}(x_{n-1}, x_n)$ if need be. From the initial simulation data, an estimate $\hat{S}(x)$ of $\tilde{S}(x)$ is obtained via curve fitting. That estimate is then substituted into (5) and the result normalized to integrate to 1, thus giving an updated importance sampling kernel $\hat{q}_S^{(1)}(x_{n-1}, x_n)$. Using the updated kernel, more simulation data are obtained, which are used to improve the estimate $\hat{S}(x)$. The improved estimate then defines a new importance sampling kernel $\hat{q}_S^{(2)}(x_{n-1}, x_n)$, and so on.

Each iteration of the learning algorithm is, in principle, better than the one before. In essence, the adaptive algorithm attempts to “learn” the theoretical average $\tilde{S}(x)$ in order to use it in a zero-variance simulation. Convergence of this algorithm often tends to be exponential until the limiting performance is achieved, the performance being limited by the ability of the assumed family of curves to describe the actual function $\tilde{S}(x)$ over states x in the state space. At that point, considerable variance reduction per simulated trajectory has usually occurred relative to simulation using the model of the natural process based on the kernel $p(x_{n-1}, x_n)$.

3. EXAMPLE: DISPERSION IN HOMOGENEOUS TURBULENCE

3.1. Introduction

To illustrate adaptive importance sampling, we consider virtually the simplest textbook problem, namely, determining a one-dimensional plume spread for dispersion in homogeneous turbulence. Horizontal wind velocity is assumed constant in space and time, and the domain in this example is of infinite extent, with no reflecting boundaries or other complications. For this simple problem, the analytical solution for $\tilde{S}(x)$ is known, making convergence to zero variance possible.

The model we use for the natural stochastic process is as follows: At time $t_0 = 0$, a particle is released in the atmosphere at height z_0 having vertical velocity w_0. Indexing time in units of the simulated time step Δt, the states of the Markov chain are denoted $x_t = (t, z_t, w_t)$, and the simulated natural process we consider evolves as:

(a) Update the time index to $t + 1$.

(b) Update the vertical velocity to $w_t = \phi w_{t-1} + \eta_t$, where $\phi = 1 - 1/T_L > 0$ with T_L the Lagrangian time scale, and η_t a Gaussian random variable with mean zero and standard deviation σ_w.

(c) Update the particle height to $z_t = z_{t-1} + [w_t + w_{t-1}]/2$.

(d) Terminate the particle trajectory if the event(s) of interest have occurred; otherwise, return to (a) and continue.

The goal here is to determine the plume spread at time step $\tau = 1000$, where the plume spread denotes the theoretical second moment of the vertical displacement $(z_\tau - z_0)$ of the particle height z_τ at time step τ from the release height z_0 at time step 0. For a horizontal wind velocity of 5 m/s and time step 1 s, the time step $\tau = 1000$ corresponds to a downwind distance of 5000 meters.

The straightforward approach to this problem involves simulating numerous trajectories starting at x_0 according to the model for the natural process, observing the final particle height z_τ for each trajectory, accumulating statistics on the score $(z_\tau - z_0)^2$, and continuing the run until enough simulated trajectories have been generated to obtain the desired accuracy. The standard deviation of the average score of N trajectories decreases as $N^{-1/2}$, governing the rate of convergence.

3.2. The Learning Algorithm

To illustrate use of the learning algorithm for estimating the plume spread, knowledge is needed regarding the theoretical average $\tilde{S}(x_t) = E[(z_\tau - z_0)^2 \mid x_t]$, given that the process is in state x_t at time t. It can be shown (see Appendix A) that the theoretical average for a particle in state $x_t = (t, z_t, w_t)$, where $t < \tau$, is

$$\begin{aligned}\tilde{S}(x_t) &\equiv E[(z_\tau - z_0)^2 \mid (t, z_t, w_t)] \\ &= \left[z_t - z_0 + w_t\left(\frac{1}{2} + \sum_{i=1}^{\tau-t-1}\phi^i + \frac{1}{2}\phi^{\tau-t}\right)\right]^2 \\ &\quad + \left[\sigma_w^2 \sum_{i=1}^{\tau-t-1}\left\{\frac{1}{2}\phi^{\tau-t-i} + \sum_{j=i}^{\tau-t-1}\phi^{j-i}\right\}^2 + \sigma_w^2/4\right],\end{aligned}$$

where again the simulated time step is 1 s and $\phi = 1 - 1/T_L$ where T_L is the Lagrangian time scale. Summing the geometric series above and collecting terms in the state space variables (t, z_t, w_t) of x_t gives the relation

$$\begin{aligned}\tilde{S}(x_t) &= \beta_0 + \beta_1 t + \beta_2(1/\phi)^{\tau-t} + \beta_3(1/\phi^2)^{\tau-t} + \beta_4 z_t^2 + \beta_5 w_t^2 + \beta_6 z_t w_t \\ &\quad + \beta_7 z_t w_t(1/\phi)^{\tau-t} + \beta_8 w_t^2(1/\phi)^{\tau-t} + \beta_9 w_t^2(1/\phi^2)^{\tau-t}, \qquad (6)\end{aligned}$$

where the curve-fitting parameters $\{\beta_j\}$ depend on specifics of the problem, such as σ_w. If $\tilde{S}(x_t)$ in Eq. (6) were substituted into the zero-variance kernel in Eq. (5), a single particle trajectory from the biased random walk starting at any state x would produce the solution $\tilde{S}(x) = E[(z_\tau - z_0)^2 \mid x]$ without error.

To illustrate the adaptive procedure which learns $\tilde{S}(x)$, suppose that the parameters $\{\beta_j\}$ in Eq. (6) are treated as unknowns to be learned from simulation data. Knowledge of the solution to this level of detail is unrealistic in practical applications, but the point here is to demonstrate the performance of the algorithm under ideal conditions.

In order to learn the theoretical average $\tilde{S}(x)$ for all states x, particles are released from a number of different initial states (t, z_t, w_t). Even if interest (nominally) lies only in $\tilde{S}(x_0)$ for a single initial state x_0, it is necessary to simulate trajectories originating from many states to do the curve fitting required to bias the random walks. Let the set $D = \{x^{(1)}, x^{(2)}, \ldots, x^{(d)}\}$, called the *design*, denote d states from which particles are released. That is, each design point $x^{(i)} = (t^{(i)}, z^{(i)}, w^{(i)})$ denotes a specific state from which a trajectory is initiated and a score obtained.

For this example, we choose a design having $d = 750$ initial states, where the design has a Cartesian product structure with

$$\begin{aligned} t &\in \{0\text{ s}, 50\text{ s}, 100\text{ s}, 150\text{ s}, 200\text{ s}, 250\text{ s}, 300\text{ s}, 350\text{ s}, 400\text{ s}, 450\text{ s}, 500\text{ s}, 550\text{ s}, 600\text{s}, \\ &\quad 650\text{ s}, 700\text{ s}, 750\text{ s}, 800\text{ s}, 850\text{ s}, 900\text{ s}, 910\text{ s}, 920\text{ s}, 930\text{ s}, 940\text{ s}, 950\text{ s}, 960\text{ s}, 970\text{ s}, \\ &\quad 980\text{ s}, 990\text{ s}, 995\text{ s}, 998\text{ s}\}, \\ z &\in \{z_0, z_0 \pm 200\text{ m}, z_0 \pm 400\text{ m}\}, \quad \text{and} \\ w &\in \{0\text{ m/s}, \pm 1\text{ m/s}, \pm 2\text{ m/s}\}.\end{aligned}$$

That is, all $30 \times 5 \times 5 = 750$ combinations of the above 30 time points, 5 particle release heights, and 5 vertical velocities are used for the initial particle release conditions. This design consists of somewhat regularly spaced points on a three-dimensional lattice and is chosen because it covers the space of particle trajectories reasonably well and devotes increasing attention to the more important later time points (recall that the final time step is at $\tau = 1000$ s). Through judicious choice of design points, the right amounts of information can be obtained from all relevant regions of the state space, as some regions are more crucial to efficient calculation than are others.

In the first iteration of the algorithm, 25 trajectories are simulated from each state in the design D using nature's transition kernel $p(x_{t-1}, x_t)$. The 25×750 simulated scores are then used to estimate the function $\tilde{S}(x_t)$ through a least-squares curve fitting (see Appendix B for details). Letting $z_\tau^{(i)}$ denote the height at time τ of a particle released from the ith design point $x^{(i)}$, the expected value of the score $(z_\tau^{(i)} - z_0)^2$ for that particle's trajectory is

$$E\left[\left(z_\tau^{(i)} - z_0\right)^2\right] \equiv \tilde{S}\left(x^{(i)}\right) = \sum_{j=0}^{9} \beta_j b_j\left(x^{(i)}\right),$$

where the $\{b_j(\cdot)\}$ are the basis functions of the regression model and are given in Eq. (6). Using the simulation results, estimated parameters $\{\hat{\beta}_j\}$ are obtained.

The estimate $\hat{S}(x_t)$ of $\tilde{S}(x_t)$, where both are viewed as functions of x_t, is obtained by inserting the least-squares estimates $\{\hat{\beta}_j\}$ into the above, giving

$$\hat{S}(x_t) = \sum_{j=0}^{9} \hat{\beta}_j b_j(x_t). \tag{7}$$

Upon substituting this estimated importance function $\hat{S}(x_t)$ for $\tilde{S}(x_t)$ in Eq. (5) and normalizing, an approximate zero-variance kernel $\hat{q}_S(x_{t-1}, x_t)$ is obtained for the next iteration of the learning algorithm. (Note that occasionally, it can happen that the fitted curve $\hat{S}(x_t) \leq 0$ for some x_t; this situation is discussed in Appendix C.)

Subsequent iterations of the adaptive process, following the $k = 1st$, have the form:

(a) Substitute the estimated importance function $\hat{S}(x)$ into (5) and normalize to obtain the importance sampling kernel $\hat{q}_S^{(k-1)}(x_{t-1}, x_t)$.

(b) Simulate 25 particle trajectories originating from each of the d states $\{x^{(i)}\}$ in the design D and using the transition kernel $\hat{q}_S^{(k-1)}(x_{t-1}, x_t)$.

(c) Use least-squares regression of the simulation data to estimate the parameters $\{\beta_j\}$ to update the estimate $\hat{S}(x)$ as indicated in Eq. (7).

(d) Update the iteration to $k = k + 1$.

3.3. Particle Splitting

Before giving the results for the example, we digress to describe a simulation trick known as particle splitting. Particle splitting is a method for reducing uncertainty that complements importance sampling (e.g., Hammersley and Handscomb [9]) and is effective when used in the examples of this paper.

To understand the concept, return to the setting of Section 2.1 and Monte Carlo for simple integrals. The importance sampling estimate (3),

$$\hat{I}_g = \frac{1}{N} \sum_{i=1}^{N} \left\{ s(y_i) \frac{f(y_i)}{g(y_i)} \right\},$$

combines values $\{s(y_i)\}$ with weights $\{f(y_i)/g(y_i)\}$. The average weight, with respect to the simulation distribution $g(y)$, is equal to 1. But in the event that a few weights were to dominate all others, the estimate $\hat{I}_g$ would be dominated by the few scores from the corresponding random walks, largely ignoring the information in the other random walks. Such a situation can lead to unstable estimates.

This situation is remedied by particle splitting. For random walk models, each particle trajectory score (4) has the form

$$S(x_0) = \sum_{n=1}^{\tau} \left\{ s(x_{n-1}, x_n) \frac{\prod_{i=1}^{n} p(x_{i-1}, x_i)}{\prod_{i=1}^{n} q(x_{i-1}, x_i)} \right\},$$

where the weight at step n is the ratio of the probability density function for the natural process to that for the importance sampled process,

$$\frac{\prod_{i=1}^{n} p(x_{i-1}, x_i)}{\prod_{i=1}^{n} q(x_{i-1}, x_i)}.$$

This weight can be monitored as the particle trajectory evolves and, if it becomes too large, the particle is "split."

Splitting is not a physical reality, but instead it is a statistical device that prevents destabilization of the estimate. If, after step n of the random walk, a particle weight becomes too large, the particle is replaced by k "subparticles" at state x_n, where each subparticle has weight $1/k$ times the weight of the original particle. Upon independently simulating each subparticle from state x_n to the conclusion of its random walk, the subparticle scores are added and used in place of the score that would have been obtained for the unsplit original particle.

In the examples that follow, particles are split whenever their weight exceeds 2.0, the number k of subparticles being just large enough so that each subparticle weight is less than 2.0, which tends to split particles during the early portions of their trajectories. Note that there exists the possibility that the subparticles may themselves be split at subsequent time steps.

3.4. Numerical Results

Returning to the plume spread example, results are summarized in Fig. 2 for simulating trajectories according to the model for the natural process and for simulating biased trajectories using the adaptive approach in conjunction with particle splitting and the theoretical functional form (6) of $\tilde{S}(x)$. For both approaches, we use constant Lagrangian time scale $T_L = 10$ s, standard deviation $\sigma_w = 0.1$ m/s, and initial vertical velocity $w_0 = 0$ m/s. The plotted curves display simulation accuracy relative to the known plume spread as a function of central processing unit (CPU) time, where results are averaged over 50 runs of each approach. The term *run* refers to simulating natural trajectories from $x_0 = (0, z_0, 0)$ for 1000 time steps each, continuing to simulate trajectories for a total of 2000 CPU seconds (for the standard approach), and refers to simulating successive iterations of the learning algorithm for the same amount of time (for the adaptive approach).

Upon comparing the two approaches, it is seen that as the algorithm learns the parameters $\{\beta_j\}$, scores from simulated random walks from x_0 for the adaptive procedure converge to the known plume spread $\tilde{S}(x_0) = 19715$ m^2 exactly (well, to machine precision) and do so

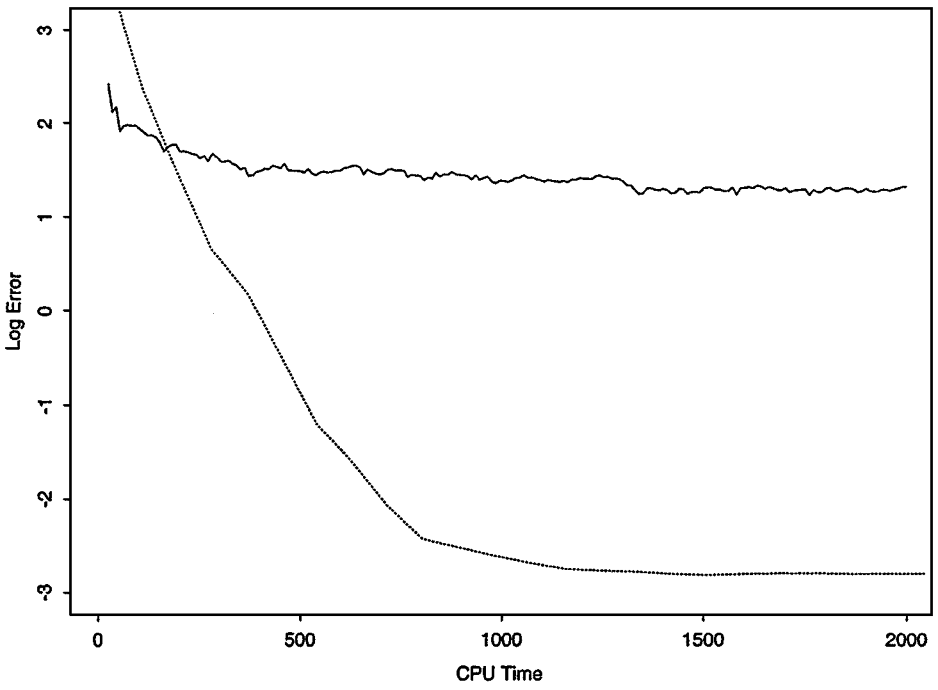

FIG. 2. Log_{10} simulation error as a function of CPU time for the plume spread example, for simulation of the model for the natural process (solid curve) and for simulation of the adaptive zero-variance process (dashed curve). The dashed curve levels off once machine precision for the calculation is reached.

in an amount of CPU time that results in only modest accuracy for the standard approach. As shown in Fig. 2, the $\log_{10}$ difference between $\tilde{S}(x_0)$ and the simulation estimate from the adaptive approach decreases to machine precision almost linearly as a function of computation time. This behavior is called *exponential convergence*.

As is apparent, simulating a biased particle trajectory requires more computational work than does simulating a natural trajectory, especially for models like this one where producing a natural trajectory is simple. The additional CPU time per simulated trajectory is more than offset, however, by the need for far fewer trajectories and hence leads to the substantial efficiency gain.

Typical particle trajectories from $x_0 = (0, z_0, 0)$ for the natural process (Fig. 3) and for the zero-variance process (Fig. 4) illustrate the difference between the two approaches. The set of possible trajectories for random walks is the same in each case, but the probability distributions on the trajectories are markedly different. Vertical displacements $z_\tau - z_0$ for the natural and zero-variance processes resemble, not surprisingly, random samples from the idealized distributions in Fig. 1.

3.5. Use of Approximating Models

We now consider a continuation of the above problem, intended to show the effects of imperfect knowledge. That is, suppose the theoretical form of $\tilde{S}(x_t)$ were not known. Instead, suppose that enough were understood about $\tilde{S}(x_t)$, from subject matter knowledge and/or from initial simulation data, to describe its general features in terms of a parametric model. For example, instead of using the theoretical functional form (6) for biasing the

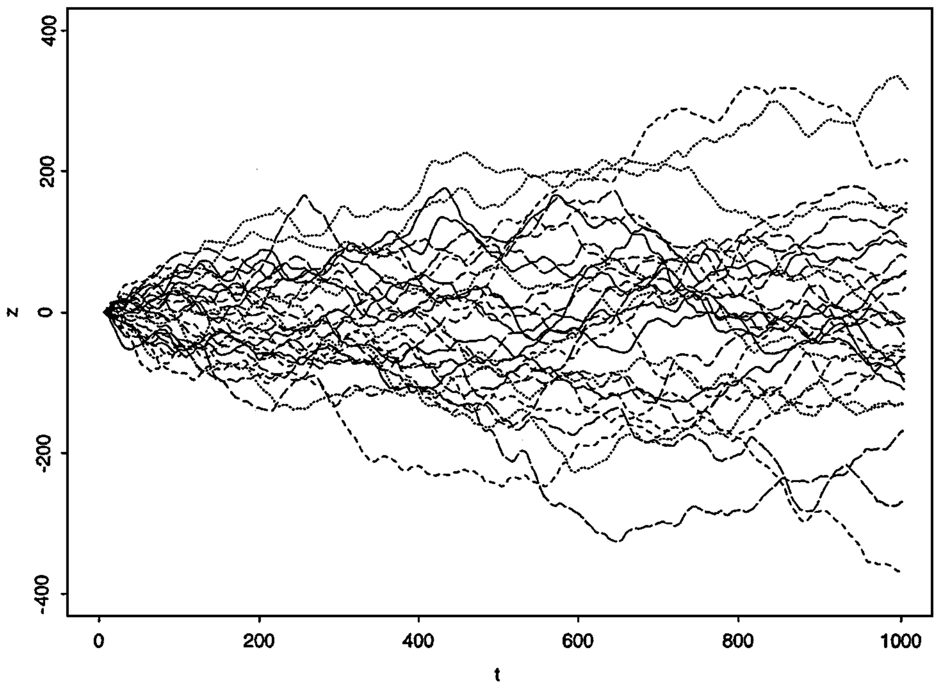

FIG. 3. Random sample of natural trajectories for the plume spread example.

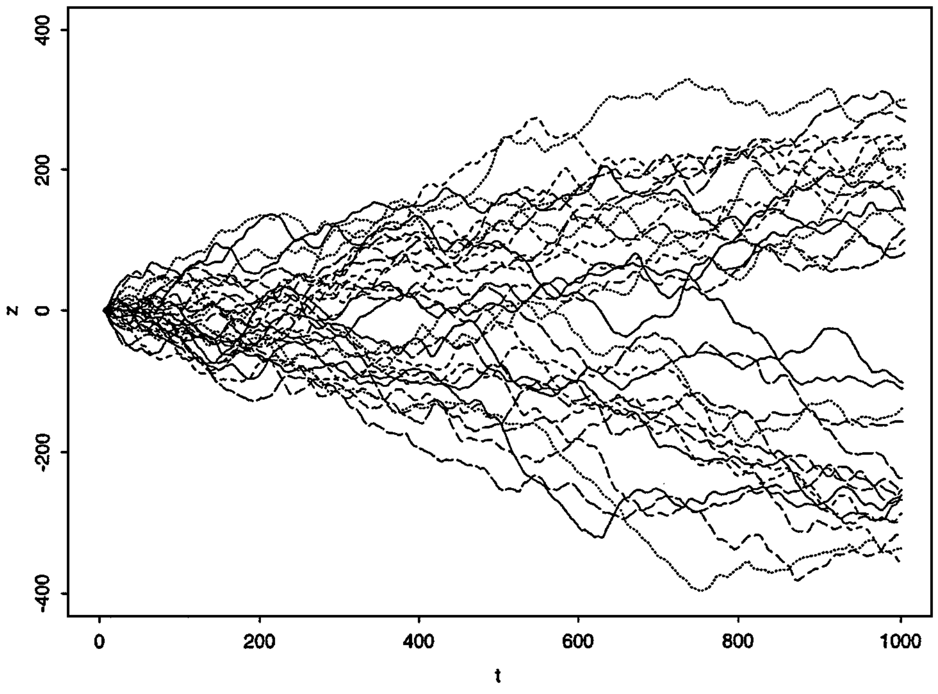

FIG. 4. Random sample of zero-variance biased trajectories for the plume spread example.

random walks, consider instead using a quadratic approximation

$$\begin{aligned} S^*_{\boldsymbol{\beta}^*}(x_t) = \beta^*_0 + \beta^*_1 t + \beta^*_2 z_t + \beta^*_3 w_t + \beta^*_4 t^2 + \beta^*_5 z_t^2 + \beta^*_6 w_t^2 \\ + \beta^*_7 t z_t + \beta^*_8 t w_t + \beta^*_9 z_t w_t, \end{aligned} \tag{8}$$

where $\boldsymbol{\beta}^* = \{\beta^*_j\}$ is a vector of curve-fitting parameters to be determined using simulation data. Because the plume spread $\tilde{S}(x_t)$ is a well-behaved function of the state space variables (t, z_t, w_t), the model (8) provides a reasonable approximation over the range of x_t that matters.

Generally speaking (e.g., [31]), the two objectives in multivariate importance sampling are reduced variance and ease of simulation. Relative to choosing a model $S^*_{\boldsymbol{\beta}^*}(x_t)$ to guide biased random walks, these two objectives imply

(a) It is vital that, for a "good" parameter vector $\boldsymbol{\beta}^*$, the approximating function $S^*_{\boldsymbol{\beta}^*}(x_t)$ resembles $\tilde{S}(x_t)$ as a function of x_t, so that the biased random walk has scores with variance close to zero, and

(b) the functional form of $S^*_{\boldsymbol{\beta}^*}(x_t)$ is such that biased random walks using approximate zero-variance kernel $\hat{q}_S(x_{t-1}, x_t)$ can be simulated in minimal CPU time.

In this example, these objectives are met.

Although the second-order response surface approximation (8) is not formally correct in a theoretical sense, the adaptive learning algorithm can be implemented as if it were. To illustrate, consider repeating the approach in the previous section. That is, in the first iteration of the algorithm, 25 particles are released from each of the 750 design points in D. Trajectories evolve according to the natural process. The simulated plume spreads at $\tau = 1000$ time steps are used to obtain the best-fitting approximation of the form $S^*_{\boldsymbol{\beta}^*}(x_t)$. Here, the term "best" again means that the parameter vector $\boldsymbol{\beta}^*$ defining $S^*_{\boldsymbol{\beta}^*}(x_t)$ is optimal in a least-squares sense.

The estimate $S^*_{\boldsymbol{\beta}^*}(x_t)$ is substituted for $\tilde{S}(x_t)$ in Eq. (5) and then normalized to define the transition kernel $\hat{q}_S^{(1)}(x_{t-1}, x_t)$ for the next iteration of the algorithm. Then, 25 trajectories again are simulated from each of the 750 design points $\{x^{(i)}\}$, this time using the updated kernel. The resulting simulation data are used to improve the estimated parameters $\{\beta^*_j\}$ through another curve fitting, the improved parameters in turn are used to improve the estimated importance function, and so on.

Because the model (8) is only approximate, convergence to zero variance is not possible. Further, once the limiting performance of the model has been learned, there is no benefit to additional adaptation. At that point, $\tilde{S}(x_0)$ is estimated by averaging results from trajectories originating from x_0, using the transition kernel corresponding to the limiting performance. Iterations of the learning algorithm are terminated upon stabilization of the sum of squares SS_D for the design D,

$$SS_D = \sum_{x^{(i)} \in D} \left[\bar{s}\big(x^{(i)}\big) - S^*_{\boldsymbol{\beta}^*}\big(x^{(i)}\big)\right]^2, \tag{9}$$

where $\bar{s}(x^{(i)})$ denotes the average of the 25 replicated scores $S(x^{(i)})$ from biased trajectories sourced from the ith design point $x^{(i)}$ and the sum is taken over all $d = 750$ design points. When the moving average (over three consecutive iterations of the algorithm) of SS_D increases, the learning is discontinued. At that point, multiple random walks from $x_0 = (0, z_0, 0)$ are simulated independently using the most recent transition kernel $\hat{q}_S(x_{t-1}, x_t)$.

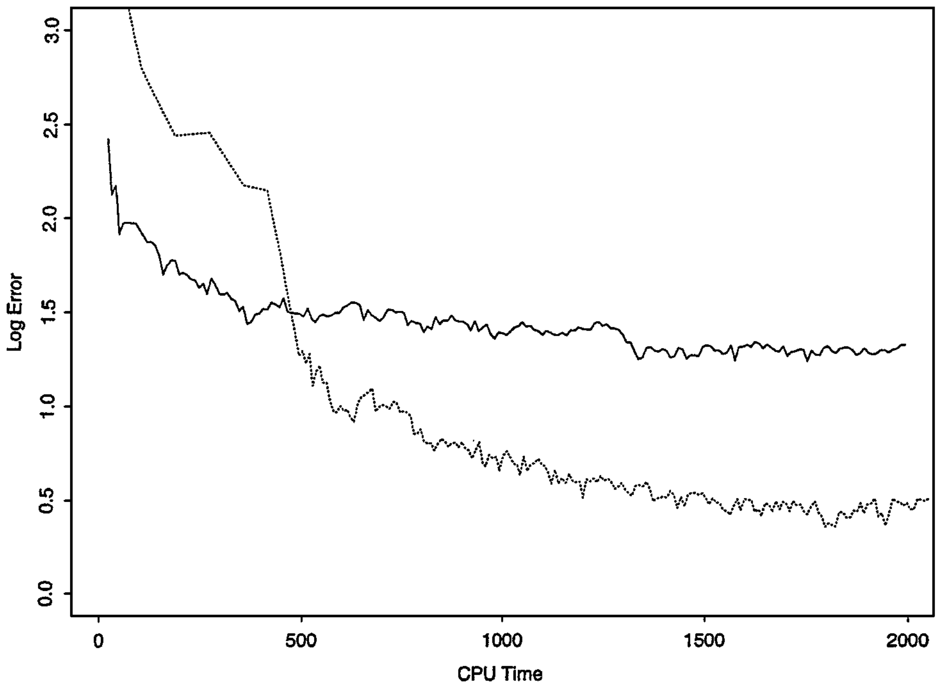

FIG. 5. Log_{10} simulation error as a function of CPU time for the plume spread example, for simulation of the model for the natural process (solid curve) and for simulation of the adaptive approximate zero-variance process (dashed curve).

The mean and standard deviation of the scores then give the estimated plume spread and its uncertainty.

Results are displayed in Fig. 5, which summarizes 50 runs (where each run consists of at least 2000 CPU seconds of simulating trajectories) for the adaptive approach and for the approach simulating the natural process. The learning algorithm requires roughly 500 s, on the average, to learn parameter values, during which time its estimates $\hat{S}(x_0)$ of $\tilde{S}(x_0)$ have comparatively large variability. And upon learning, generating a biased trajectory takes roughly 6 times more CPU time than generating a natural trajectory.

The acceleration in convergence, however, is substantial. Note from Fig. 5 that the difference in $\log_{10}$ error between the adaptive approach and the simulation of the model for the natural process at a CPU time of 2000 s is approximately equal to 0.83, averaged over the 50 runs for each approach. This difference is equivalent to a factor of $10^{0.83} \approx 6.7$ decrease in absolute error for the adaptive approach. Because of the $N^{-1/2}$ behavior of the standard deviation, this means that the model for the natural process would have to be simulated roughly $6.7^2 \approx 45$ times longer in order to produce enough trajectories to match this 6.7-fold improvement in accuracy.

Like most iterative algorithms, convergence of adaptive learning is accelerated through the use of good starting values. That is, instead of using the natural process as the basis for the first iteration of the algorithm, preliminary estimates of the parameters $\{\beta_j^*\}$ could, if available, be used to define the transition kernel for the initial iteration. Such preliminary estimates can sometimes be extrapolated from solutions to similar problems. In other cases, simulation codes of "lower resolution" could provide preliminary estimates. If starting values for the learning algorithm provided a reasonable facsimile of $\tilde{S}(x_t)$, biased trajectories would give more accurate results than would natural trajectories in the first iteration and would further accelerate convergence.

As an example of this type of approach, Turner and Larsen [27] used deterministic methods for a neutron transport application to solve a discretized version of the actual (continuous) problem, doing so with less computational work than analog Monte Carlo would have entailed. Then, using their solution to the discretized case, they defined approximate zero-variance biasing parameters to guide a nonadaptive Monte Carlo for the continuous problem, thereby coupling the advantages of deterministic and Monte Carlo methods in the same analysis. Such coupling could similarly help define biasing parameters for the initial iteration for adaptive Monte Carlo.

4. EXAMPLE: REFLECTING BOUNDARIES AND PARTICLE DEPOSITION

4.1. Introduction

Adaptive importance sampling offers its greatest benefits in the study of rare-event behavior—events such as a particle being deposited on a small surface area A, or passing through a small volume V, or traveling an unusually great distance from its release point, and so on. The straightforward approach to the rare-event problem involves simulating numerous natural trajectories, selecting from them the few that are "rare" and then examining the behavior of the events in that subset. While such an approach has the advantage that simulating a large number of natural trajectories guarantees that the subset of rare events has the correct statistical properties, there is an obvious drawback in that much simulation effort is wasted. Importance sampling provides a method to preferentially simulate uncommon trajectories and reweight the results; see, e.g., [3, 10, 18, and 28] for discussions of nonadaptive importance sampling.

Consider a generalization of the previous example, intended to illustrate rare-event behavior relevant to dispersion of heavy particles. Such issues arise in problems related to migration of airborne pollutants, to cross-pollination of agricultural crops, to resuspension of aerosols, to aspects of chemical/biological warfare, and to other applications. In this example, the goal is to estimate the probability that a pollen particle travels a great distance without being deposited on the ground. When such a rare-event probability is coupled with the total number of released particles, deposition fluxes can be obtained.

Specifically, we assume that a pollen particle is released from a point source at height $z_0 = 5$ m at time $t = 0$ having initial vertical velocity $w_0 = 0$ m/s. The particle is subject to a downward drift of $\delta = 0.5$ m/s; this drift is consistent with values for certain pollen reported by Sehmel [25]. The ground, represented by the plane $z = 0$, acts as a partially reflecting lower boundary. That is, if the height z_t of a trajectory drops below 0, the particle is deposited on the ground with a certain probability π and is reflected upward with probability $1 - \pi$. For simplicity, the horizontal wind velocity is assumed constant in time and space, so that the downwind distance is just a multiple of the number of time steps.

Letting other aspects of the simulation be as before, the simulated natural process evolves as follows:

(a) Update the time to $t + 1$.

(b) Update the vertical velocity to $w_t = \phi w_{t-1} + \eta_t$, where $\phi = 1 - 1/T_L > 0$ with T_L the Lagrangian time scale, and η_t a Gaussian random variable with mean zero and standard deviation σ_w.

(c) Update the particle height to $z_t = z_{t-1} + [w_t + w_{t-1}]/2 - \delta$.

(d) If $z_t < 0$, the particle is deposited on the ground with probability π; if it is not deposited, then set $z_t = |z_t|$ and set $w_t = |w_t + \delta|$, thereby ensuring that $z_t \geq 0$ and $w_t \geq 0$ after reflection.

(e) Terminate the trajectory if the particle has been deposited on the ground or if the time step $t > 1000$; otherwise, return to (a) and continue.

In this example, we use deposition probability $\pi = 0.75$ for each reflection in (d).

For these parameter values, most particles are deposited on the ground well in advance of the 1000th simulated time step, and the probability that a particle is airborne after 1000 time steps is approximately 5×10^{-5}. For more realistic problems, where particles move in a three-dimensional space, similarly small probabilities can occur easily. As noted above, estimation of such rare-event probabilities via simulation of the natural process is inefficient.

4.2. Survival Biasing

For problems involving particle deposition, a useful simulation trick is survival biasing, also known as *implicit capture* [4] in radiation transport. To illustrate the concept, recall from the discussion of particle splitting that each trajectory score (4) has the form

$$S(x_0) = \sum_{t=1}^{\tau} \left\{ s(x_{t-1}, x_t) \frac{\prod_{i=1}^{t} p(x_{i-1}, x_i)}{\prod_{i=1}^{t} q(x_{i-1}, x_i)} \right\},$$

and the particle weight at time step t is

$$\frac{\prod_{i=1}^{t} p(x_{i-1}, x_i)}{\prod_{i=1}^{t} q(x_{i-1}, x_i)}.$$

When a simulated particle reaches the ground during time step $t < 1000$, there are two possibilities: either it is deposited with probability π (and contributes a zero score) or it is reflected with probability $1 - \pi$ (and contributes the score that it would receive upon completion of the remainder of its random walk). Allowing particles to be deposited before time step $\tau = 1000$ shares the same weakness as does simulation of the natural process, namely, that much computation is wasted in generating trajectories that do not produce the rare event of interest.

With survival biasing, whenever a particle reaches the ground before time step $\tau = 1000$, it is *always* reflected. To correct for the particle never being deposited, the particle weight is reduced (through multiplication by $1 - \pi$) upon each reflection. In a sense, this is equivalent to having a $1 - \pi$ fraction of the particle be reflected at the boundary. The resulting estimate is statistically unbiased as a result of this weight reduction. Note that some trajectories could involve survival biasing at several individual time steps if a particle repeatedly impacts the ground surface.

On the final time step, analytical results are used in place of simulated ones. That is, instead of simulating the τth time step of the trajectory and obtaining $s(x_{\tau-1}, x_\tau) = 1$ or 0, depending on whether the particle is airborne following the τth time step, it is simple to compute the one-step probability that the particle is airborne given its height and vertical velocity. The value of the probability is then used as $s(x_{\tau-1}, x_\tau)$ for the trajectory, in place of the 1 or 0 that would otherwise be used. It is not hard to show that variability is reduced by this technique.

4.3. Particle Deposition and Adaptive Importance Sampling

The goal of importance sampling for the deposition problem is to preferentially simulate trajectories having a high probability of being airborne following time step $\tau = 1000$ ("high" relative to the population of trajectories resulting from simulating from the model for the natural process). This might seem difficult, because unlike the plume spread example in Section 3.1, there is no simple formula for the eventual deposition probability owing to the partially reflecting boundary $z = 0$. Fortunately, good results can be obtained for rare-event probabilities without a formally correct model for the functional relation between deposition probability and state space variables.

Particle trajectories having high probability of being airborne following time step $\tau = 1000$ tend to have heights z_t well above zero. One way to preferentially generate such trajectories that gives substantial improvement over the (grossly inefficient) simulation of the natural process is to model the log probability $\tilde{S}(x_t)$ that a particle in state x_t will be airborne after $\tau - t$ additional time steps using the quadratic approximation as in Eq. (8),

$$\begin{aligned} \ln \tilde{S}(x_t) &\approx \ln S^*_{\beta^*}(x_t) \\ &= \beta^*_0 + \beta^*_1 t + \beta^*_2 z_t + \beta^*_3 w_t + \beta^*_4 t^2 + \beta^*_5 z_t^2 + \beta^*_6 w_t^2 + \beta^*_7 t z_t + \beta^*_8 t w_t + \beta^*_9 z_t w_t, \end{aligned}$$

where, as before, the $\{\beta^*_j\}$ are curve-fitting parameters to be estimated. As in Section 3.5, this second-order response surface is intended to roughly describe the theoretical average of interest as a function of x_t.

Given this approximate model, the approach is the same as for the plume spread example. Trajectories are simulated starting from a set of design points; a curve fitting is carried out to estimate the parameters $\{\beta^*_j\}$; the resulting parameter estimates are used to update the importance sampling, which is then used as the basis for simulating still more trajectories; and so on.

The differences between the rare-event example and plume spread example are such that a simpler design provides good results. A useful design D^* for the rare-event example has a Cartesian product structure with

$$\begin{aligned} t &\in \{0\,\text{s},\ 250\,\text{s},\ 500\,\text{s},\ 750\,\text{s},\ 995\,\text{s}\}, \\ z &\in \{1.33\,\text{m},\ 5\,\text{m},\ 15\,\text{m},\ 45\,\text{m},\ 135\,\text{m}\}, \qquad \text{and} \\ w &\in \{-1\,\text{m/s},\ 0\,\text{m/s},\ 1\,\text{m/s},\ 2\,\text{m/s},\ 3\,\text{m/s}\}. \end{aligned}$$

That is, all 125 combinations of the 5 time points, 5 heights, and 5 vertical velocities are used as initial release conditions, with 50 simulated trajectories per design point. Recall that the release height of interest is $z_0 = 5$ m, so that the above design points roughly cover the set of plausible trajectories of particles being airborne after 1000 time steps. As in the plume spread example, more complex designs coupled with more complex approximations $S^*_{\beta^*}(x)$ to the theoretical function $\tilde{S}(x)$ would provide slightly better results.

The first iteration of the adaptive algorithm involves simulation of the model for the natural process in conjunction with survival biasing. Adapting as before (by monitoring the moving average of the sum of squares SS_{D^*} for the design D^*) leads to the performance summarized in Fig. 6. As in the plume spread example, 50 runs of the natural process and of the adaptive approach with survival biasing and particle splitting are simulated, each run

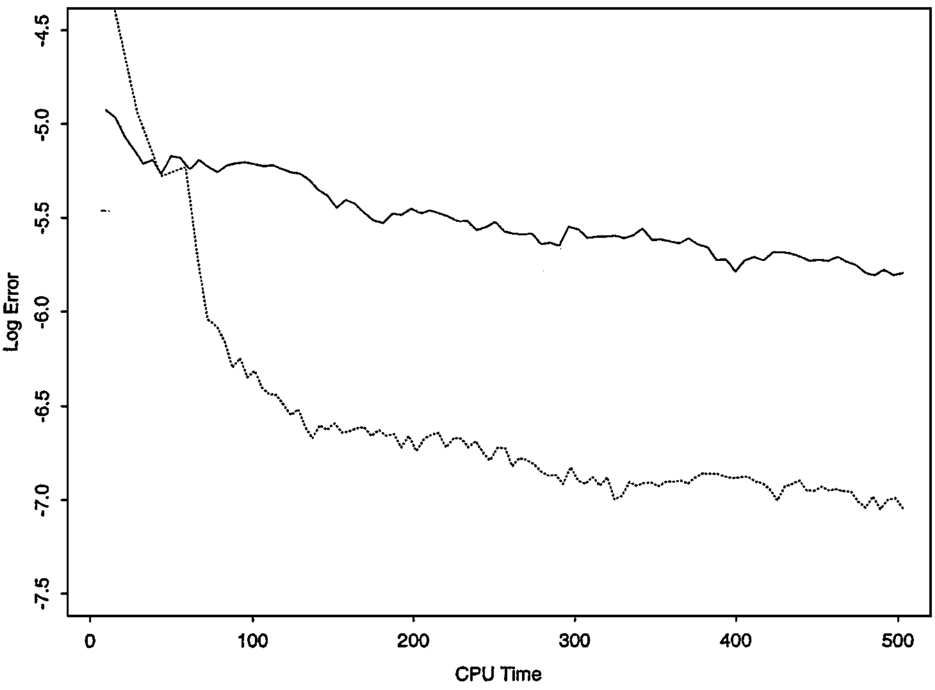

FIG. 6. Log_{10} simulation error as a function of CPU time for the pollen deposition example, for simulation of the model for the natural process (solid curve) and for simulation of the adaptive approximate zero-variance process (dashed curve).

for a CPU time exceeding 500 s. The learning stage of the adaptive algorithm takes roughly 100 s, on the average. At that point, simulation of a biased trajectory requires a factor of 35 more CPU time than does simulation of a natural particle trajectory (recall that most natural trajectories involve particles depositing on the ground after relatively few time steps).

As with most rare-event examples, however, the inherent inefficiency of simulating rare events by subselecting from a set of natural events is such that efficiency gain is considerable. After a CPU time of 500 s, for example, the difference in absolute error is $10^{1.23} \approx 17$, so that the natural process would need to be run a factor of $17^2 \approx 289$ times longer to simulate enough natural trajectories to achieve the same accuracy.

Use of the importance sampling scheme $S^*_{\beta^*}(x)$ computed in the above scenario also produces significant efficiency gains when used for biasing random walks in scenarios "near" the one considered here. That is, for scenarios whose defining parameters are close to the ones above (those values being $\sigma_w = 0.1$ m/s, $\delta = 0.5$ m, and $\pi = 0.75$, for example) or for scenarios whose events of interest are of similar character (such as the event that a particle travels $\tau = 800$ or 1200 time steps from its release point). By preferentially simulating long particle trajectories and reweighting the results, quantities of interest are determined far more efficiently.

5. CAVEATS

Previous sections have emphasized the role of adaptive importance sampling using approximate zero-variance kernels in accelerating convergence of Monte Carlo particle dispersion simulations. The drawbacks of this approach also warrant mention. The first drawback is that implementation of the approach is still in its infancy. Similar to nonadaptive importance sampling, existing adaptive methods "often require users to have a fair degree of experience, or to employ a significant amount of trial and error, to obtain accurate efficient results" [27, p. 22]. The need for such knowledge has, perhaps, inhibited the development of adaptive algorithms that transcend specific applications, such as those in radiation transport.

Part of the novelty of the approach described herein is that its biasing parameters are obtained through a simple curve fitting of simulation data, a subject that largely transcends specific applications. This novelty, however, comes with a lack of experience in dealing with real problems. The examples of the previous sections, though simple in many regards, reflect the state of the art in use of the adaptive zero-variance approach to random-walk models. Experience with more realistic problems can only be gained with time.

A second drawback is that, in most simulations, interest lies in multiple physical properties. As may be apparent from Figs. 1 and 4, a good importance sampling for one quantity of interest $\tilde{S}(x_0)$ at one initial state x_0 is achieved by one reweighting of the natural distribution for particle trajectories, but such a reweighting can lead to efficiency losses for other quantities $S^\dagger(x^\dagger)$ at other states $x^\dagger$, for which different reweightings are needed. One way to deal with this problem is to simply repeat the learning algorithm for each property of interest. Another way is to hedge against poor estimation through the mixing of importance distributions (as discussed by Hesterberg [11] and by Raghavan and Cox [21] with respect to simulation approximation of simple integrals). Such mixing provides a balance across multiple quantities of interest. Either alternative is generally superior to the standard approach of simulating the model for the natural process.

The third drawback is that efficient implementation of the approach here requires knowledge beyond that related to direct simulation of the physical process of interest. Computer programming skills, including familiarity with simulation tricks such as splitting, roulette, and survival biasing, are helpful for most problems. And because of the curve-fitting aspect of adaptive importance sampling based on approximate zero-variance kernels, it is essential to have expertise in statistics in

(a) experimental design to select good designs D from which to originate particle trajectories;

(b) multiple regression and curve fitting to identify models $S^*_{\beta^*}(x_t)$ giving good approximations to simulation data, an issue that takes on greater significance in applications that entail complex models having a large number of parameters; and

(c) multivariate random number generation to simulate steps for biased random walks from approximate zero-variance kernels of the form (5), where again the challenges become greater in high-dimensional settings.

In cases where results from simulation of the model for the natural process are sufficiently fast and accurate, the need for such skills diminishes and finding the expertise to accelerate convergence may not be worthwhile.

6. CONCLUSION

Caveats notwithstanding, the prospect of one or more orders-of-magnitude reduction in convergence times for turbulent diffusion calculations is worth pursuing. For simulation codes that are run often (e.g., with different input values), whose estimates converge slowly, or from which results of high precision are required, the potential benefits of adaptive importance sampling are considerable.

APPENDIX A

Dispersion in Homogeneous Turbulence

In this section, the stochastic process for homogeneous turbulence is reviewed and the plume spread calculations for biased random walks are illustrated. Begin with a particle at height z_0 with vertical velocity w_0 at time $t = 0$. In what follows, the time step is denoted Δt, though the examples herein use $\Delta t = 1$ s for simplicity, so that time is indexed in units of the simulated time step. As noted in Section 3.1, the stochastic simulation for the natural process updates the height z_t of the particle as follows:

(a) update the time to $t + \Delta t$;

(b) update the vertical velocity to $w_t = \phi w_{t-1} + \eta_t$, where $\phi = 1 - \Delta t / T_L > 0$ with T_L the Lagrangian time scale and η_t a Gaussian random variable with mean zero and standard deviation σ_w;

(c) update the particle height to $z_t = z_{t-1} + \Delta t\, [w_t + w_{t-1}]/2$; and

(d) if $t < \tau$, return to (a).

Here, the vertical velocity w_t and particle height z_t behave according to standard time series models [7].

Of interest is the vertical plume spread $\tilde{S}(t_0, z_0, w_0)$ at future time step τ given the initial conditions $x_0 = (t_0, z_0, w_0)$, formally equal to the expected value $E[(z_\tau - z_0)^2 \mid x_0]$. Because of the idealized character of the problem, the plume spread can be calculated analytically. That is, the recursive relations (b) and (c) lead to

$$w_t = \phi^t w_0 + \sum_{i=1}^{t} \phi^{t-i} \eta_i$$

and the conditional relation

$$\begin{aligned} z_\tau \mid (t_0 = 0, z_0, w_0) &= z_0 + \Delta t \left[w_0/2 + \sum_{t=1}^{\tau-1} w_t + w_\tau/2 \right] \\ &= z_0 + \Delta t w_0 \left(\frac{1}{2} + \sum_{i=1}^{\tau-1} \phi^i + \frac{1}{2}\phi^T \right) \\ &\quad + \Delta t \sum_{i=1}^{\tau-1} \eta_i \left\{ \frac{1}{2}\phi^{\tau-i} + \sum_{j=i}^{\tau-1} \phi^{j-i} \right\} + \frac{\Delta t}{2} \eta_\tau, \end{aligned}$$

which imply that the vertical plume spread is

$$\begin{aligned} \tilde{S}(t_0 = 0, z_0, w_0) &\equiv E[(z_\tau - z_0)^2 \mid x_0] \\ &= \left[(\Delta t)^2 (w_0)^2 \left(\frac{1}{2} + \sum_{i=1}^{\tau-1} \phi^i + \frac{1}{2}\phi^\tau \right)^2 \right] \\ &\quad + \left[(\Delta t)^2 \sigma_w^2 \sum_{i=1}^{\tau-1} \left\{ \frac{1}{2}\phi^{\tau-i} + \sum_{j=i}^{\tau-1} \phi^{j-i} \right\}^2 + (\Delta t)^2 \sigma_w^2 / 4 \right]. \end{aligned}$$

Using a similar derivation, a random walk in state $x_t = (t, z_t, w_t)$, where $t \le \tau - 2$, proceeds for $\tau - t$ steps until time τ, and so conditional on the particle being in state x_t

$$\begin{aligned} z_\tau \mid (t, z_t, w_t) = z_t &+ \Delta t w_t \left(\frac{1}{2} + \sum_{i=1}^{\tau-t-1} \phi^i + \frac{1}{2}\phi^{\tau-t} \right) \\ &+ \Delta t \sum_{i=t+1}^{\tau-1} \eta_i \left\{ \frac{1}{2}\phi^{\tau-i} + \sum_{j=i}^{\tau-1} \phi^{j-i} \right\} + \frac{\Delta t}{2}\eta_\tau, \end{aligned}$$

leading to plume spread

$$\begin{aligned} \tilde{S}(t, z_t, w_t) &\equiv E[(z_\tau - z_0)^2 \mid (t, z_t, w_t)] \\ &= \left[z_t - z_0 + (\Delta t w_t)\left(\frac{1}{2} + \sum_{i=1}^{\tau-t-1} \phi^i + \frac{1}{2}\phi^{\tau-t} \right) \right]^2 \\ &\quad + \left[(\Delta t)^2 \sigma_w^2 \sum_{i=1}^{\tau-t-1} \left\{ \frac{1}{2}\phi^{\tau-t-i} + \sum_{j=i}^{\tau-t-1} \phi^{j-i} \right\}^2 + (\Delta t)^2 \sigma_w^2 / 4 \right]. \end{aligned}$$

Simplifying the summations using

$$\sum_{i=1}^{\tau-t-1} \phi^t = (\phi - \phi^{\tau-t})/(1-\phi),$$

and collecting terms in (t, z_t, w_t), Eq. (6) as cited in Section 3.1 is obtained.

APPENDIX B

Estimating $\tilde{S}(x)$ from Simulation Data

At the heart of the learning algorithm is the use of simulated data from biased random walks in estimating the theoretical quantity $\tilde{S}(x)$ as a function of x. In this section, the process of estimating $\tilde{S}(x)$ is described.

For the design $D = \{x^{(1)}, x^{(2)}, \ldots, x^{(d)}\}$, let $\overline{s}(x^{(i)})$ denote the average score of the replicated trajectories having initial state $x^{(i)}$. The parameters $\{\beta_j^*\}$ defining the model $S_{\beta^*}^*(x)$ for $\tilde{S}(x)$ can be chosen to minimize the sum of squares

$$SS_D = \sum_{x^{(i)} \in D} \left[\overline{s}\left(x^{(i)}\right) - S_{\beta^*}^*\left(x^{(i)}\right) \right]^2,$$

where the sum is taken over all d design points. For models linear in their parameters such as (6) and (8), a closed-form solution exists for the least-squares estimates, standard software can be used, and estimates are obtained quickly. For nonlinear models, a simplex search procedure [20] or derivative-based optimization routines, such as Levenberg-Marquardt [2], can be used for the optimization, although usually at considerable cost in CPU time relative to linear models. Upon substitution of the parameter estimates into the functional form for $\tilde{S}(x)$, an estimated function $\hat{S}(x)$ is obtained from each iteration of the learning algorithm.

Upon reaching the limiting accuracy of the model (this is reflected by equilibration in the plot of the sum of squares SS_D against iteration number), the estimated score function $\hat{S}(x)$ is obtained. For the results herein, the adaptation is terminated once the moving average of SS_D over three consecutive iterations of the algorithm increases, the rationale being that the algorithm has learned nearly all it can about the approximating model by that point. Additional computation should then be devoted to simulating trajectories for determination of quantities of interest. Parameter estimates for the importance sampling kernel to be used for final estimates are obtained from the final iteration of the adaptation.

APPENDIX C

Simulating Biased Random Walks

In this section, simulation from biased random walks is summarized. Computational aspects of this process can be nontrivial at times, but the details matter: given the number of particle trajectories involved in most simulations and the number of time steps per trajectory, any computational inefficiencies in generating random-walk steps from the transition kernel $\hat{q}_S(x_{t-1}, x_t)$ will accumulate.

As noted in Section 2.2, for $p(x_{t-1}, x_t)$ denoting the natural transition probability density from x_{t-1} to x_t, the estimated zero-variance transition kernel has the form

$$\hat{q}_S(x_{t-1}, x_t) \propto p(x_{t-1}, x_t) \frac{s(x_{t-1}, x_t) + \hat{S}(x_t)}{\hat{S}(x_{t-1})}.$$

The existence of this functional form does not necessarily imply that it is easy to simulate random walks from it.

In the rare-event example, such simulation is straightforward. The tth time step involves simulation of the vertical velocity w_t from the biased probability density function $\hat{q}_S(x_{t-1}, x_t)$. Recall that the approximating model for $\tilde{S}(x_t)$ is

$$\begin{aligned}\ln S^*_{\beta^*}(x_t) &= \beta_0^* + \beta_1^* t + \beta_2^* z_t + \beta_3^* w_t + \beta_4^* t^2 + \beta_5^* z_t^2 + \beta_6^* w_t^2 + \beta_7^* t z_t + \beta_8^* t w_t + \beta_9^* z_t w_t \\ &= \beta_6^* w_t^2 + (\beta_3^* + \beta_8^* t + \beta_9^* z_t) w_t + (\text{terms not involving } w_t) \\ &= \gamma_0 + \gamma_1 w_t + \gamma_2 w_t^2,\end{aligned}$$

where the $\{\gamma_j\}$ depend on t, z_t, and the $\{\beta_j^*\}$ in the obvious way. Importantly, the log probability is just a quadratic function in w_t. The probability density function $p(x_{t-1}, x_t)$ for w_t, according to the model for the natural process, is Gaussian, with mean $\phi\, w_{t-1}$ and standard deviation σ_w, implying

$$p(x_{t-1}, x_t) \propto e^{(w_t - \phi\, w_{t-1})^2 / 2\sigma_w^2}.$$

Upon noting that the score function $s(x_{t-1}, x_t) = 0$ for all but the final step of the biased random walk, it follows that the probability density function $\hat{q}_S(x_{t-1}, x_t)$ has the form

$$\hat{q}_S(x_{t-1}, x_t) \propto p(x_{t-1}, x_t) S^*_{\beta^*}(x_t) \propto e^{(w_t - \phi w_{t-1})^2/2\sigma_w^2} e^{\gamma_0 + \gamma_1 w_t + \gamma_2 w_t^2} \propto e^{(w_t - \mu)^2/2\sigma^2},$$

where the parameters μ and σ follow upon collecting terms in w_t and depend on ϕ, w_{t-1}, σ_w, and the $\{\gamma_i\}$. In other words, the vertical velocity w_t for the biased random walk is distributed as a Gaussian, with mean and standard deviation that depend on the biasing. As such, simulation of the biased random walk for the rare-event example is relatively simple. Once w_t is obtained, z_t is derived from w_t as in the model for the natural process.

Simulation for the plume spread example is only slightly more complicated. In this case, the probability density function has the form

$$\hat{q}_S(x_{t-1}, x_t) \propto p(x_{t-1}, x_t) S^*_{\beta^*}(x_t) \propto \left(\gamma_0 + \gamma_1 w_t + \gamma_2 w_t^2\right) e^{(w_t - \phi w_{t-1})^2 / 2\sigma_w^2},$$

where the $\{\gamma_i\}$ depend on the parameters $\{\beta_j^*\}$ as above. Given the simple form of this probability density function, w_t can be simulated by rejection methods (e.g., [9]) using a mixture of Gaussians as the dominating distribution. The particle height $z_t = z_{t-1} + (w_t + w_{t-1})/2$ is then obtained from w_t just as for the natural process.

In some applications, it is possible that the curve fitting for $\tilde{S}(x_t)$ can lead to situations where $\hat{q}_S(x_{t-1}, x_t) \leq 0$ for some values of x_t. When this happens, $\hat{q}_S(x_{t-1}, x_t)$ as defined will: (a) not be a probability density function because of the negative values, or (b) assign zero importance to a region of positive importance because of the zero values. Either event destroys the validity of the approach.

If the theoretical quantity $\tilde{S}(x_t)$ can be negative (such as for flux), this can be accommodated by using absolute values of $\tilde{S}(x_t)$ in Eq. (5), leading to a low-variance (as opposed to zero-variance) importance distribution. Negative values of $\hat{S}(x_t)$ can also occur when many particle trajectories give a zero score and the fitted curve $\hat{S}(x_t)$ for $\tilde{S}(x_t)$ is not positive over certain regions of the state space. This problem can be overcome by adding a constant to the fitted curve. Taking Eq. (7), e.g., the fitted curve would be modified as

$$\hat{S}(x_t) \;=\; \sum_{j=0}^{9} \hat{\beta}_j\, b_j(x_t) \;+\; c,$$

where the constant $c > 0$ is chosen to "raise" the fitted curve above zero for all values of x_t in the state space. Most often, the fitted curve $\sum_{j=0}^{9} \hat{\beta}_j\, b_j(x)$ is already greater than zero for all states x, and no such modification is necessary.

REFERENCES

1. K. Baggerly, D. Cox, and R. Picard, Exponential convergence of adaptive importance sampling for Markov chains, *J. Appl. Probab.* **37**, 342 (2000).
2. D. M. Bates and D. G. Watts, *Nonlinear Regression Analysis and Its Applications* (Wiley, New York, 1988).
3. I. Beichl and F. Sullivan, Approximating the permanent via importance sampling with application to the dimer covering problem, *J. Comput. Phys.* **149**, 128 (1999).
4. T. E. Booth, *A Sample Problem for Variance Reduction in MCNP*, Technical Report LA-10363-MS (Los Alamos National Laboratory, 1985).
5. T. E. Booth, Exponential convergence on a continuous Monte Carlo transport problem, *Nucl. Sci. Eng.* **127**, 338 (1997).
6. M. S. Borgas, T. K. Flesch, and B. L. Sawford, Turbulent dispersion with broken reflectional symmetry, *J. Fluid Mech.* **332**, 141 (1997).
7. G. E. P. Box and G. M. Jenkins, *Time Series Analysis: Forecasting and Control* (Holden Day, San Francisco, 1976).
8. P. de Haan and M. W. Rotach, A novel approach to atmospheric dispersion modelling: The puff-particle model, *Q. J. R. Meteorol. Soc.* **124**, 2771 (1998).

9. J. M. Hammersley and D. C. Handscomb, *Monte Carlo Methods* (Chapman and Hall, London, 1983).
10. P. Heidelberger, Fast simulation of rare events in queueing and reliability models, *ACM Trans. Model. Comput. Sim.* **5**, 43 (1995).
11. T. Hesterberg, Weighted average importance sampling and defensive mixture distributions, *Technometrics* **37**, 185 (1995).
12. P. Hurley, Partpuff—A Lagrangian particle puff approach for plume dispersion modeling applications, *J. Appl. Meteorol.* **33**, 285 (1994).
13. P. Hurley and W. Physick, A skewed homogeneous Lagrangian particle model for convective conditions, *Atmos. Environ. A* **27**, 619 (1993).
14. H. Kahn, Random sampling (Monte Carlo) techniques in neutron attenuation problems, I and II, *Nucleonics* **6**(5), 27 and **6**(6), 60 (1950).
15. C. Kollman, K. Baggerly, D. Cox, and R. Picard, Adaptive importance sampling on discrete Markov chains, *Ann. Appl. Probab.* **9**, 391 (1999).
16. R. G. Lamb, A numerical simulation of dispersion from an elevated point source in the convective planetary boundary layer, *Atmos. Environ.* **12**, 1297 (1978).
17. G. P. LePage, A new algorithm for adaptive multidimensional integration, *J. Comput. Phys.* **27**, 192 (1978).
18. O. Mazonka, C. Jarzynski, and J. Blocki, Computing probabilities of very rare events for Langevin processes: A new method based on importance sampling, *Nucl. Phys. A* **641**, 335 (1998).
19. J. S. Nasstrom and D. L. Ermak, A homogeneous langevin equation model, Part I: Simulation of particle trajectories in turbulence with a skewed velocity distribution, *Boundary-Layer Meteorol.* **92**, 343 (1999).
20. J. A. Nelder and R. Mead, A simplex method for function minimization, *Comput. J.* **7**, 308 (1965).
21. N. Raghavan and D. D. Cox, Adaptive mixture importance sampling, *J. Statist. Comput. Sim.* **60**, 237 (1998).
22. J. D. Reid, Markov chain simulations of vertical dispersion in the neutral surface layer for surface and elevated releases, *Boundary-Layer Meteorol.* **16**, 3 (1979).
23. A. M. Reynolds, On trajectory curvature as a selection criterion for valid Lagrangian stochastic dispersion models, *Boundary-Layer Meteorol.* **88**, 77 (1998).
24. R. Y. Rubenstein and B. Melamed, *Modern Simulation and Modeling* (Wiley, New York, 1988).
25. G. A. Sehmel, Deposition and resuspension, in *Atmospheric Science and Power Production*, edited by D. Randerson (Technical Information Center, U.S. Dept. of Energy Oak Ridge, TN, 1984), Chapter 12.
26. D. J. Thomson, Criteria for the selection of stochastic models of particle trajectories in turbulent flows, *J. Fluid Mech.* **180**, 529 (1987).
27. S. A. Turner and E. W. Larsen, Automatic variance reduction for three-dimensional Monte Carlo simulations by the local importance function transform—I. Analysis, *Nucl. Sci. Eng.* **127**, 23 (1997).
28. T. Ueki and E. W. Larsen, A kinetic theory for nonanalog Monte Carlo particle transport algorithms: Exponential transform with angular biasing in planar geometry anisotropically scattering media, *J. Comput. Phys.* **145**, 406 (1998).
29. H. van Dop, F. T. M. Nieuwstadt, and J. C. R. Hunt, Random walk models for particle displacements in inhomogeneous unsteady turbulent flows, *Phys. Fluids* **28**, 1639 (1985).
30. J. D. Wilson and B. L. Sawford, Review of Lagrangian stochastic models for trajectories in the turbulent atmosphere, *Boundary-Layer Meteorol.* **78**, 191 (1996).
31. P. Zhang, Nonparametric importance sampling, *J. Am. Stat. Assoc.* **91**, 1245 (1996).