AIP | The Journal of Chemical Physics

Umbrella sampling for nonequilibrium processes

Aryeh Warmflash, Prabhakar Bhimalapuram, and Aaron R. Dinner

Citation: The Journal of Chemical Physics **127**, 154112 (2007); doi: 10.1063/1.2784118

View online: http://dx.doi.org/10.1063/1.2784118

View Table of Contents: http://scitation.aip.org/content/aip/journal/jcp/127/15?ver=pdfcov

Published by the AIP Publishing

Umbrella sampling for nonequilibrium processes

Aryeh Warmflash, Prabhakar Bhimalapuram, and Aaron R. Dinner[a)]
James Franck Institute, The University of Chicago, Chicago, Illinois 60637, USA

(Received 25 June 2007; accepted 21 August 2007; published online 18 October 2007)

The authors introduce an algorithm for determining the steady-state probability distribution of an ergodic system arbitrarily far from equilibrium. By enforcing equal sampling of different regions of phase space, as in umbrella sampling simulations of systems at equilibrium, low probability regions are explored to a much greater extent than in physically weighted simulations. The algorithm can be used to accumulate joint statistics for an arbitrary number of order parameters for a system governed by any stochastic dynamics. They demonstrate the efficiency of the algorithm by applying it to a model of a genetic toggle switch which evolves irreversibly according to a continuous time Monte Carlo procedure. © *2007 American Institute of Physics*. [DOI: 10.1063/1.2784118]

I. INTRODUCTION

Many systems of significant fundamental and applied interest are irreversible. These include, but are not limited to, living systems, chemical reactors, systems with driven flows of matter and energy, and light-driven systems. For theoretical studies of such nonequilibrium processes, the steady-state distribution is of central importance because it enables calculation of static averages of observables for comparison to experimental measurements. For example, flow cytometry can be used to detect the single-cell protein levels in a large population of cells efficiently.[1,2] From these data, steady-state distributions for protein numbers can be constructed and compared with quantitative stochastic models for gene and signaling regulatory networks. Often, the observed distributions are strongly asymmetric with long tails and multiple peaks. As such, it can be important to calculate higher moments of model distributions,[24] but doing so is rarely possible analytically without approximation and can become prohibitive computationally due to the fact that low probability states contribute significantly to such averages.

For systems at equilibrium, low probability states can be explored efficiently in simulations with umbrella sampling methods, in which biasing potentials that are functions of one or more order parameters are used to enhance sampling of selected regions of phase space.[3–6] What complicates extending umbrella sampling to simulations of nonequilibrium processes is that, by definition, they do not obey detailed balance (microscopic reversibility). As such, one must account for the fact that the steady-state probability of observing particular values of the order parameters can be determined by a balance of flows in phase space through different possible transitions.

Here, we describe what we believe to be the first general umbrella sampling algorithm for steady-state distributions of nonequilibrium processes. Even sampling in a space of an arbitrary number of order parameters is obtained by discretizing it and performing a separate simulation in each resulting region. The lack of detailed balance necessitates transfer of information about fluxes and probabilities between connected regions. The computational cost of the algorithm scales linearly with the number of projected states in the system regardless of their probabilities. The algorithm can be employed with any stochastic integrator; here, we show explicitly that the relative occupancies of different states of a genetic toggle switch converge in orders of magnitude fewer total steps in continuous time Monte Carlo (MC) simulations which employ our algorithm than conventional ones.[7] Relations to other methods for enhanced sampling of nonequilibrium processes are discussed.

II. METHODS

A. Overview

In this section, we describe the algorithm and its implementation. It is motivated by the observation that if an unconstrained simulation of an ergodic nonequilibrium process were run, the probability distribution in a region of interest would depend only on the segments of the trajectory that crossed it. So long as one knew the flux from outside the region into it, one could weigh these segments correctly and perform a simulation of that part of the phase space in isolation. Below we refer to states in a region that are accessible from outside it as "boundary states," even though they need not be physically at its boundary due to jumps in the space of order parameters. By the same token, we refer to any two regions connected by allowed transitions as "neighboring."

Using these terms, the basic scheme is as follows. We divide the space into boxes defined by order parameters and run a conventional simulation in each box. Whenever the system attempts to exit a box, we return it to a boundary state selected with information obtained from a neighboring box. The simulations in different boxes are otherwise independent. The algorithm is thus reminiscent of equilibrium umbrella sampling with an infinite square well potential, except that, rather than simply rejecting transitions from the box, it is necessary to reset the system such as to account correctly for the flux into the box.

Below, we first prove that, given the flux into a box, the steady-state probability distribution can be sampled by per-

[a)]Electronic mail: dinner@uchicago.edu forming a simulation in that box alone. We then show how to compute the fluxes from states outside the box to those inside it up to a constant weight factor which compensates for the nonphysical density of walkers (replicas of the system) in each box and, in turn, how to compute this weight factor. We then summarize the algorithm and discuss technical details for its implementation. Finally, a simplified algorithm for the special case that one is interested in only a single order parameter is described.

B. Theory

For concreteness, we consider stochastic realizations of a master equation obtained from a continuous time Monte Carlo algorithm,[7] although any other stochastic dynamics can be treated so long as the system is ergodic. We label the states inside the box of the current simulation by Roman indices and those outside by Greek indices. In terms of these two groups of states, the master equation is

$$\frac{\partial P_i}{\partial t} = -a_i P_i + \sum_j r_{ij} P_j + \sum_\alpha r_{i\alpha} P_\alpha, \tag{1}$$

where P_i is the probability of residing in state i, r_{ij} is the transition probability to state i from state j, and $a_i = \Sigma_j r_{ji} + \Sigma_\alpha r_{\alpha i}$ is the total escape rate from state i.

Defining $\mathbf{P}$ to be the vector with components P_i for states i in the region of interest, one can rewrite Eq. (1):

$$\frac{\partial \mathbf{P}}{\partial t} = W\mathbf{P} + \mathbf{f}. \tag{2}$$

Above, W is the transition matrix with off-diagonal entries $W_{ij} = r_{ij}$ and on-diagonal entries $W_{ii} = -a_i$; $\mathbf{f}$ is a flux vector defined such that f_i is the total flux into state i from outside the region (i.e., $f_i = \Sigma_\alpha f_{i\alpha}$, where $f_{i\alpha} = r_{i\alpha} P_\alpha$). In a conventional master equation, each column of the evolution operator sums to zero to ensure that probability is conserved. In contrast, W in Eq. (2) contains the exit rates to states outside the box (through the diagonal elements) but does not account for the transfer of probability back into the box. The net flux of probability out of the box that results from W is balanced at steady state by the vector $\mathbf{f}$, which introduces new walkers to the box to represent the flux from outside. Thus, one can view $\mathbf{f}$ in Eq. (2) as a set of sources for walkers that otherwise evolve and exit the box according to the original dynamics encoded in W.

Equation (2) suggests that the density of walkers in each box fluctuates, but, in fact, we fix it. A conventional simulation is performed for each walker until it attempts to exit the box, at which point it is returned to another state in the box with a probability proportional to the flux into that state from neighboring regions. That is, the probability of being returned to state i is

$$F_i = \frac{f_i}{\Sigma_i f_i}, \tag{3}$$

where the sum is over all states inside the box (states inaccessible from outside have $f_i = 0$ and thus do not contribute). Since we are only interested in the steady-state solution to Eq. (2) given by $\mathbf{P} = -W^{-1}\mathbf{f}$ up to a constant normalization factor, the fluxes can be scaled arbitrarily. Thus, as long as we return walkers to boundary states chosen with the proper ratios of fluxes, this scheme is equivalent to the source picture above. It is preferable since it is simple to implement and allows one to control the degree to which each region is sampled through the specification of the number of walkers.

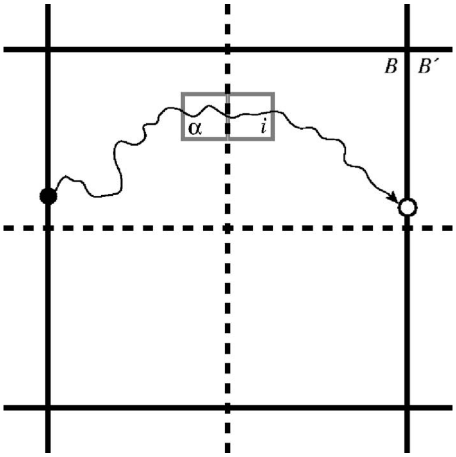

FIG. 1. Schematic of the algorithm. We show one box (labeled B) on lattice 2 (solid lines) and parts of the overlapping boxes on lattice 1 (dashed lines). The wiggly line represents one trajectory in B. It begins at a boundary point chosen using the fluxes into B (filled circle) and is generated using the integrator that defines the dynamics. When the trajectory transitions from state α to state i, it crosses a boundary on lattice 1; the flux counter $N_{i\alpha}^{(2)}$ is incremented, and the configuration upon entry to i is stored. These quantities will be used to choose boundary points and entry configurations for the simulation in box $b^{(1)}(i)$. Finally the trajectory leads to an attempt to exit the box (open circle); the weight of B is decreased, and the weight of B' is increased. Then, the configuration is reset to a boundary point in box B and the simulation is continued.

The above analysis can be extended immediately to the case in which one is interested in a projection of the steady-state probability distribution onto a subset of variables. To do so, the flux vector must be broken into two parts: the flux into regions of phase space consistent with the values of the order parameters of interest and the distribution of that flux to states within that region that differ only with regard to the remaining degrees of freedom in the system. The simulation procedure is essentially the same as above, except that i in Eq. (3) labels a set of order parameter values rather than a single state. Once the monitored variables are chosen according to the fluxes, the remaining degrees of freedom are chosen from their joint distribution at that state. In practice the latter step is accomplished by storing configurations representative of the flux distribution at each set of order parameter values and randomly picking from this list upon return to that projected state.

C. Computing fluxes

We have shown that a separate simulation can be performed for each box in the space of order parameters if the fluxes into its states are known. The array of boxes covering the space defines a lattice. To determine the fluxes, we introduce a second such lattice shifted such that the interfaces between boxes on one lattice run through the middle of boxes on the other (Fig. 1). The two lattices (indexed as 1 and 2) are otherwise the same, and simulations are performed on each as described above. We explicitly describe the transfer of information in terms of that from lattice 2 to lattice 1, but we simultaneously execute the operations obtained by swapping the lattice indices.

During the course of a simulation on lattice 2, we record the number of crossings of each lattice 1 boundary and the configurations that result from such events. Consider a projected state i which is in a particular box on lattice 1. Define the indicator function $b^{(x)}(i)$ for $x=1,2$ such that it returns the index of the box on lattice x in which state i resides. Then the flux from a state α outside of $b^{(1)}(i)$ into state i is computed from a simulation on lattice 2 according to

$$f_{i\alpha}^{(1)} = \frac{N_{i\alpha}^{(2)} w_{b^{(2)}(\alpha)}^{(2)}}{T_{b^{(2)}(\alpha)}^{(2)}}, \tag{4}$$

where $N_{i\alpha}^{(x)}$ is the number of crossings from state α into state i recorded by the simulation on lattice x, $T_B^{(x)}$ is the amount of time elapsed in the simulation run in box B on lattice x, and $w_B^{(x)}$ is a weight factor associated with box B on lattice x. This last parameter compensates for the restriction on the density of walkers in each box. Without the weight factors, fluxes out of low probability boxes would be inflated relative to those out of high probability boxes.

The flux into state i (f_i) is obtained by summing Eq. (4) over all α, as stated above. Thus, given the weight factors, the fluxes can be computed and the states chosen as described above. In addition, every time a boundary on lattice 1 is crossed in the simulation on lattice 2, the configuration is stored for use in setting the degrees of freedom other than the order parameter upon returning walkers to their boxes in the simulations on lattice 1.

D. Computing the box weights

To determine a means for evaluating the weight factors, it is again worth considering an unconstrained simulation. Walkers would transition from one region to another, and, once a steady state was reached, on average there would be many in the high probability parts of phase space and few in the low probability parts. In our algorithm, each walker constrained to a box represents this average occupancy, and we thus transfer portions of the weight factors for each boundary crossing. Specifically, whenever a walker attempts to leave its box (labeled B) to go to a neighboring one (B'), the configuration is reset as detailed above and a transfer of weight is made from B to B',

$$-\Delta w_B^{(x)} = \Delta w_{B'}^{(x)} = s w_B^{(x)} T^* / T_B^{(x)}, \tag{5}$$

where Δ denotes an additive change and T^* is a reference time which prevents the numerical value of the right hand side from decreasing as the simulation time increases. In practice, T^* can be chosen to be the elapsed time in any box as long as the choice of reference box is fixed throughout the simulation. The factor $w_B^{(x)}$ reflects the fact that the number of boundary crossings is linearly proportional to the number of walkers in a region. By the same token, longer simulations have more opportunity for boundary crossings, so we divide by $T_B^{(x)}$ to allow for differences in the physical times elapsed in B and B'. Finally, s is an arbitrary parameter that enables one to tune how much weight is transferred for each boundary crossing. Higher values of s lead to faster redistribution but also larger fluctuations in the instantaneous values of the weight factors.

E. Summary

To review, the algorithm proceeds operationally as follows. A separate simulation is carried out in each box, during which boundary crossings and configurations are recorded for use by simulations in overlapping boxes on the other lattice. Whenever a walker tries to exit its box, it is returned to a boundary point chosen according to Eq. (3) with the fluxes computed from the simulation on the other lattice according to Eq. (4), and the configuration is reset to one from the list of attempted entries. Additionally, a portion of the weight of the box containing the walker is shifted to the box to which it would have transitioned in an unconstrained simulation according to Eq. (5). At the end of the simulation, the steady-state probability of state i as calculated from the simulation on lattice x is given by

$$P_i^{(x)} = \frac{t_i^{(x)} w_{b^{(x)}(i)}^{(x)}}{T_{b^{(x)}(i)}^{(x)} N}, \tag{6}$$

where $t_i^{(x)}$ is the amount of time spent in projected state i and N is a normalization factor, computed from the requirement that $\Sigma_i P_i^{(x)}=1$. In order words, each walker visits the states within its box with the correct relative likelihoods, and these are then uniformly scaled by the normalized weight of the box. The probabilities computed from the simulations on the two lattices can be averaged. The overall computational cost of the algorithm scales linearly with the number of boxes.

F. Practical details

Nonequilibrium processes can be simulated using a variety of methods.[6,8] Any stochastic integrator can be employed for the simulations that take walkers from one boundary crossing to another so long as it reproduces the desired dynamics and is ergodic. Although we illustrate the algorithm with a continuous time Monte Carlo procedure for obtaining stochastic realizations of a dynamics defined by a master equation,[7] Monte Carlo or molecular dynamics integrators with discrete time steps can also be used. It is worth noting that if a continuous time algorithm is used it is unnecessary to draw a separate random number for the time increment since only steady-state properties are desired. The time for exit from state i can simply be set to $\Delta t_i = 1/a_i$.

For order parameters with finite ranges, the system can simply be divided into boxes as described above. For order parameters with infinite ranges, it is clear that it is impossible to tile the entire space with boxes. Truncation error can be avoided by choosing the outer boxes of the simulation to be infinite in some directions. The sizes of the boxes in the interior of the space of order parameters should be chosen to effectively sample all the states in each box and thus should be smaller in regions in which the probability changes rapidly. For this reason, the computational cost of the algorithm can increase somewhat faster than the volume of the projected phase space of interest. Nonetheless, this scaling compares favorably with that of a conventional simulation in which the computational cost of sampling projected states is inversely proportional to their probabilities. How to tile the space is discussed further in conjunction with the example.

At the beginning of a simulation, no data are available to use in choosing boundary points. Walkers that attempt to leave their box can be returned either to a random projected state inside the box or simply prohibited from leaving without a change in configuration until some information about the fluxes and stored configurations are accumulated. Here we adopt the latter strategy, although for systems with weakly stochastic dynamics it is important to employ the former to avoid simulating the same paths repeatedly. Initially, the sampling is not very accurate because the boundary statistics as well as the weight factors are not representative of the steady-state distribution. Convergence is significantly improved if both the fluxes $[f_i^{(x)}]$ and the accumulated probabilities $[t_i^{(x)}]$ are periodically reinitialized. Because the former are necessary to continue the simulation, we employ an iterative procedure. We accumulate boundary crossing statistics over a fixed number of simulation steps (an iteration) and, in the following iteration, use these statistics in choosing the boundary states according to Eq. (4). Convergence was also found to be accelerated by averaging the instantaneous values of the weights which fluctuate according to Eq. (5) over a fixed number of Monte Carlo (MC) steps before employing them in the flux computation in Eq. (4). We set the number of steps for this averaging equal to the number of steps in one of the iterations for computing fluxes discussed above, although these two intervals need not be the same.

In practice, the simulation is performed as follows. Initially all weights are set equal and one iteration is carried out in which walkers are prevented from exiting the box but no boundary states are chosen. For the next iteration, the boundary statistics, averaged weights, and stored configurations recorded during the first iteration are used to execute the algorithm as described in detail above. At the end of each successive iteration, these quantities are once again updated, and the simulation is continued. After each iteration, the times spent in each projected state during that iteration, along with the weight values, are used to compute the steady-state probabilities according to Eq. (6). Either the length of the iterations is increased or the results of many iterations are averaged until the desired accuracy in the probabilities is achieved.

G. One-dimensional algorithm

If the steady-state distribution is only desired as a function of one order parameter and the interfaces between boxes partition the space such that a walker in an unconstrained simulation would enter each box at the projected state that it exited, considerable simplification of the algorithm is possible. When a walker attempts to exit a box, the system is kept in the same projected state, and the remaining degrees of freedom are reset according to their joint distribution for walkers entering that boundary. This information can be obtained from the neighboring boxes on the same lattice, which obviates the need for the other lattice. In analogy to the general algorithm, whenever the system in box B attempts to transition to box B', the configuration that it would have taken upon entry into B' is stored. Because the weight factors are still needed to compute the steady-state probability distribution according to Eq. (6) at the end of the simulation, they are adjusted as in the full algorithm.

It might seem that such a single-lattice scheme could be used for obtaining the needed fluxes and configurations for sampling spaces of more than one order parameter as well. However, this is not the case due to the need to choose the projected state to which the system is returned upon attempted exit. Suppose, for example, that the weight of a box (B) fluctuates upward. By Eqs. (3) and (4), walkers in neighboring boxes will then be reset to boundary states accessible from B more often. However, if transitions from those states to ones in B are allowed, with some probability, the reset walkers will immediately attempt to enter B and increase its weight further according to Eq. (5). This positive feedback loop causes the single-lattice scheme to be unstable in simulations to obtain the steady-state probability distribution as a function of multiple variables. The use of two lattices enables boundary states on one lattice to be chosen using the fluxes from the other lattice, which breaks the feedback loop and enables convergence.

TABLE I. Reactions for the genetic toggle switch. The rate constants for the forward and backward reactions are k_f and k_b; "..." indicates that there is no backward reaction; ∅ denotes degradation. The parameters for the dimerization reactions are a factor of 2 larger than those reported by Allen *et al.* (Ref. 9) because otherwise it was not possible to reproduce their results when correctly accounting for the indistinguishability of the reactants (Ref. 7).

A reactions	B reactions	k_f	k_b
$A+A \rightleftharpoons A_2$	$B+B \rightleftharpoons B_2$	10	5
$O+A_2 \rightleftharpoons OA_2$	$O+B_2 \rightleftharpoons OB_2$	5	1
$O \rightarrow O+A$	$O \rightarrow O+B$	1	⋯
$OA_2 \rightarrow OA_2+A$	$OB_2 \rightarrow OB_2+B$	1	⋯
$A \rightarrow \varnothing$	$B \rightarrow \varnothing$	0.25	⋯

III. EXAMPLE

In this section, we demonstrate the efficiency of the algorithm for calculating steady-state properties of a model of a genetic toggle switch.[9,10] The model is defined by the reactions specified in Table I. Two proteins, A and B, can each homodimerize and then bind to an operon (O). The operon can only bind one dimer at a time. When a dimer of A (B) is bound to the operon, it represses transcription of the gene for B (A); there is no concomitant effect on the expression of A (B). As a result of these dynamics, the system has two stable states: one with abundant A and scarce B and the opposite situation. Switching between the two stable states is rare (approximately six orders of magnitude less frequent than each elementary reaction[9]).

Despite its apparent simplicity, this system provides a challenging test for our method for a number of reasons. The bistability not only makes convergence of the relative population of the stable states slow because there are few events connecting them but leads to a hysteresis in which the flow between stable states takes a different path through phase space in each direction.[9] As a result of the latter feature, the system enters and exits many of the boxes through different projected states, which requires that the boundary states be weighted properly when walkers are reset. We determine the steady-state probability distribution as a function of the total numbers of A and B ($N_x = n_x + 2n_{x_2} + 2n_{\mathrm{O}x_2}$, where n_x is the population of species x for $x = \mathrm{A}, \mathrm{B}$). It is thus necessary to also take into account the joint distribution for the state of the operon and the fractions of A and B in dimers when choosing boundary states for returning walkers.

In calculating the probability distribution, each discrete $(N_\mathrm{A}, N_\mathrm{B})$ pair was considered a separate projected state; for the umbrella sampling, we tiled a space of 80×80 with 4×4 boxes for a total of 400 boxes on each lattice (on one lattice, these were shifted by two states in each direction). The boxes on the outer edges are infinite in that the order parameters are allowed to increase to arbitrarily large values to avoid truncation errors. Within each box, walkers evolved according to the Gillespie algorithm.[7] Initially, all the boxes were given equal weights. As mentioned above, the fluxes [Eq. (4)] and changes to the weights [Eq. (5) with $s = 10^{-2}$] were accumulated over a fixed number of executed reactions (an iteration) and then the values actually used to reset walkers were updated at the end of each such interval; a list of the 100 most recent configurations from attempted transitions was maintained for each boundary state. The first iteration was 10^4 steps per box; it was followed by 10 iterations of 10^5 steps per box and then iterations of 10^6 steps per box until a total of 5×10^7 steps was performed for each walker. After the first 10^7 steps per box, the probability distribution was calculated using Eq. (6) after each iteration and these results were averaged at the end of the simulation.

For comparison, a conventional simulation with the same integrator[7] was run for 4×10^{10} steps, the total number employed in the umbrella sampling simulation (we estimate the overhead associated with the algorithm to be less than 10% of the computational time). The results are shown in Fig. 2(a). The error in the conventional simulation can be estimated by noting that symmetry requires the two peaks in the probability distribution to have equal heights. In the region where both simulations have good statistics, the results from the two methods agree to within this error. However, the conventional simulation only samples regions in which the probability is greater than about 10^{-9} compared with a maximum probability value of 2.2×10^{-2}. Our method accurately samples the space over the entire region of interest, regardless of the probability of each projected state. The outermost contour in Fig. 2(a) marks a probability of 10^{-35}.

We also computed the probability distribution in one projected dimension along the order parameter $\Delta = N_\mathrm{A} - N_\mathrm{B}$. Each projected state corresponded to an integral value of Δ and the region between $\Delta = -120$ and $\Delta = 120$ was tiled with boxes of width four states. The scale factor for Eq. (5) was $s = 10^{-1}$, and 100 configurations were stored for each boundary state to reset walkers that attempted to leave their boxes.

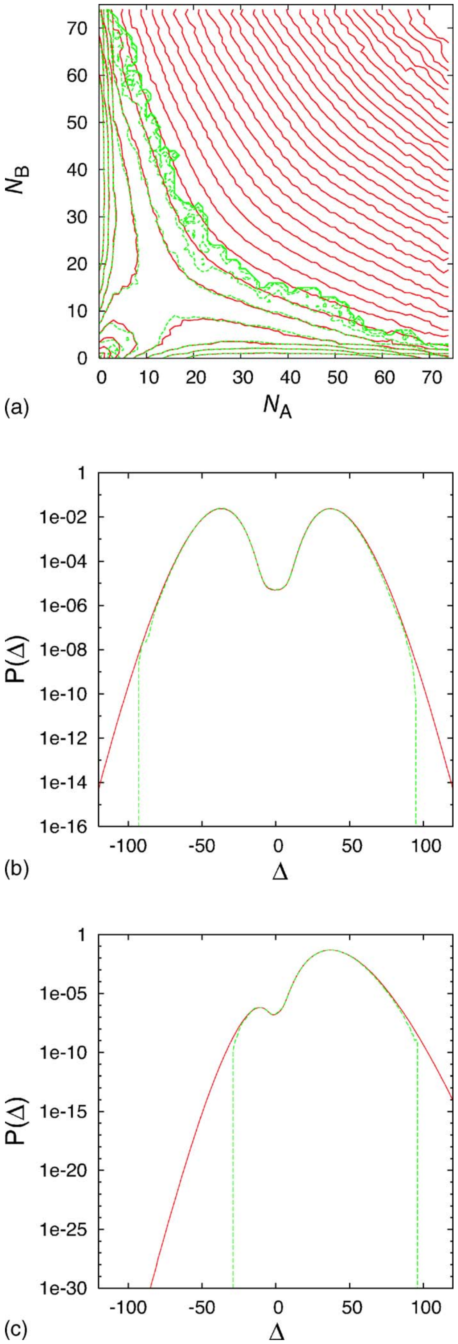

FIG. 2. (Color online) (A) Steady-state probability distributions for the toggle switch (Table I) computed with a conventional simulation (dashed lines) and umbrella sampling (solid lines). Contours are logarithmically spaced by factors of 10 with the innermost and outermost contours corresponding to probabilities of 10^{-3} and 10^{-35}, respectively. Results from the umbrella sampling are averaged over the two lattices. (B) Probability distribution along the variable $\Delta = N_A - N_B$ computed with a conventional simulation (dashed line) and umbrella sampling algorithm (solid line). (C) Same as (B) except with the degradation rate for B increased by a factor of 2.

Data are presented for the full two-lattice algorithm because simulations with the simplified one-lattice algorithm appeared to become trapped in some trials. Excellent agreement with a conventional simulation of the same length is obtained in the region of high probability and the umbrella sampling algorithm accurately samples the low probability regions as well [Fig. 2(b)]. We also examined the case in which the switch paramters were not symmetic. In particular,

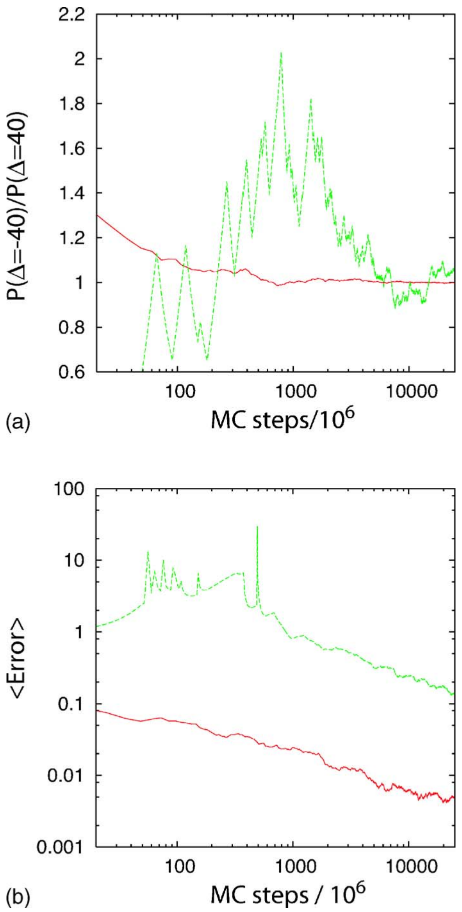

FIG. 3. (Color online) Convergence of the umbrella sampling (solid line) and conventional simulation (dashed line) for the ratio of the occupancy of the two stable states $[P(\Delta=-40)/P(\Delta=40)]$. (A) Representative simulations. (B) Average error $(\langle|P(\Delta=-40)/P(\Delta=40)-1|\rangle)$ over 20 independent simulations for each method.

we increased the degradation rate for B by a factor of 2. The results in the high probability regions are again in agreement with the conventional simulation, and the umbrella sampling samples regions of arbitrarily low probability [Fig. 2(c)].

To compare the efficiency of the umbrella sampling algorithm with a conventional simulation, we examined the convergence of the relative occupancy of the two stable states. By symmetry the states should be occupied to the same extent. The results for typical runs of the conventional and umbrella sampling simulations are shown in Fig. 3(a). The sudden jumps in the conventional simulation curve reflect switching events; their rarity accounts for the slow convergence of the conventional simulation. The umbrella sampling simulation does not suffer from this problem because it enforces significant sampling at $\Delta\approx0$. To quantify the rates of convergence, we examined the average error as a function of MC step for 20 simulations for each method [Fig. 3(b)]. After a total of 10^9 MC steps in each simulation, the ratio computed from the umbrella sampling simulations was 1.00 ± 0.03 (compared with an exact answer of 1 from symmetry), while that computed from the conventional simulation was 1.42 ± 0.88 (the reported uncertainties are standard deviations over the 20 simulations). The conventional simulation failed to achieve the former level of accuracy within 10^{11} MC steps, reaching only 1.03 ± 0.09. Thus the umbrella sampling converges to a given degree of accuracy in two orders of magnitude fewer MC steps than a conventional simulation for this example.

For this example, we obtained the best results when the space of order parameters was discretized in a manner that reflects the symmetry of the system. Otherwise, some fraction of trials were observed to become trapped, even though simulations reaching the correct steady-state probability distribution were stable. Such situations could be detected in a case in which the exact solution was not known by varying the positions of the boundaries, and, indeed, averaging over different boundary choices would be expected to yield the correct steady-state distribution. Moreover, these difficulties appear to be specific to systems with a high degree of symmetry since the same switch with asymmetric parameters converges within error to the distribution sampled by the conventional simulation even when the space was tiled symmetrically around $\Delta=0$ [Fig. 2(c)]. If the parameters were selected to make switching less frequent, the conventional simulation would converge more slowly, while the umbrella sampling simulation would remain essentially the same because its computational cost scales with the number of boxes, not inversely with the probabilities of the projected states. The umbrella sampling algorithm thus makes possible the computation of steady-state properties that would otherwise be intractable.

IV. DISCUSSION

We have introduced an algorithm for determining the steady-state distribution of a system arbitrarily far from equilibrium. The phase space is discretized, and equal sampling in different regions is enforced by restricting walkers from leaving their regions. The method is thus analogous to umbrella sampling with an infinite square well bias potential, except that information about fluxes between neighboring regions must be used to overcome the lack of detailed balance. The algorithm can be employed with essentially any stochastic integrator, and no assumptions are made with regard to the extent of memory in the projected dynamics or the form of the steady-state distribution. The computational cost scales linearly with the number of walkers. We thus believe that the algorithm will yield significant computational savings over a conventional simulation in any situation in which low probability states must be sampled with accuracy, including ones in which they are not of interest in themselves but mediate transitions between high probability states.

Formally, we showed that, given the fluxes into a region, the steady-state distribution can be sampled by running a simulation in that region in isolation [Eq. (2)]. Walkers that attempt to exit a box are returned according to fluxes computed from boundary crossing statistics weighted by the density of walkers that would be in the originating box if an unconstrained simulation with many walkers were run. The algorithm is self-consistent: given the correct boundary statistics and weight factors, the steady-state distribution will be sampled properly, and, given the actual steady-state distribution, the boundary statistics and weight factors will be computed correctly. Although we have not proven that the simulation will converge starting from incorrect weight factors, we have found this to be the case for the genetic toggle switch described, which we believe to be a demanding test for the reasons outlined in the previous section.

It is important to appreciate the similarities and differences of our algorithm to the handful of others for enhanced sampling of nonequilibrium processes. In our algorithm, one can view an attempt by a walker to leave its box as a transition to an absorbing state followed by initiation of a new trajectory according to the flux into the box. This aspect of the algorithm is very similar to a procedure introduced by de Oliveira and Dickman[11] for study of a quasistationary state, in which a system with an absorbing state takes infinitely long to relax to it at criticality in the thermodynamic limit. Simulations which accessed the absorbing state due to finite size effects were reset to previously sampled configurations; doing so led to about a tenfold increase in efficiency. In principle, the two algorithms could be combined to improve the sampling of the quasistationary distribution: that of de Oliveira and Dickman would be used to reset the system following transition to the true absorbing state and ours would be used to ensure even sampling of order parameters for characterizing the quasistationary state. In this way, our algorithm could be used to study a class of nonequilibrium processes that are not actually ergodic.

The discussion above highlights the fact that our algorithm is a form of path sampling[12,13] in which one harvests segments of trajectories that cross a region of phase space. Previous path sampling studies focused on transitions between two attractors. In particular, an algorithm based on perturbing the sequence of random variables employed in a Langevin simulation of a nonequilibrium process was introduced by Crooks and Chandler.[14] Subsequently, a more general algorithm, forward flux sampling (FFS),[9,10] was introduced to enable the efficient calculation of rates for a nonequilibrium process. The space is divided into regions, which is similar to the discretization of phase space in our algorithm except that, in FFS, the interfaces between regions must be nonintersecting. This stipulation enables the system to be ratcheted from one attractor to another. Specifically, in each stage of a FFS simulation, trajectories are initiated from an interface and only those that move forward towards the second attractor prior to returning to the first are kept and used as the starting points for the next stage. This procedure correctly samples the ensemble of transition paths, not the steady-state probability distribution, and thus it is fundamentally different from that introduced here. In this regard, it is worth noting that Bandrivskyy *et al.*[15] presented a ratchet-like algorithm for sampling the steady-state probability distribution of a two-dimensional system, but determining the distribution for different regions sequentially requires that those explored later do not influence those explored earlier, which is in general not the case.

In reversible systems, path sampling simulations can be used to obtain transition rates and free energies simultaneously.[16] Given that our algorithm samples segments of actual trajectories, it is of interest to consider how our method could be extended to obtain transition rates for nonequilibrium processes. Of course, the umbrella sampling algorithm could be used as part of a procedure analogous to the Bennett-Chandler approach.[6,17,18] During the umbrella sampling simulation, configurations in the transition region and the stable states from which the trajectories that generated them came could be stored. Given this information, the flux of trajectories leading from the reactant to the transition region could be computed, and structures from those trajectories could be used to initiate unconstrained simulations to obtain the transmission coefficient for reaching the product region. Alternatively, milestoning[19–21] could be used in conjunction with our algorithm to characterize the time evolution of the probability distribution of the order parameters and properties that depend on it. In milestoning, information about the kinetics of transitions between coarse-grained states is accumulated in short simulations and this information is then combined through a non-Markovian hopping model. Since the local dynamics in our algorithm reproduce the dynamics of the unconstrained system, such a hybrid method would not rely on equilibrium assumptions or *a priori* knowledge of the distributions of fluxes through the boundaries of the coarse-grained states, in contrast to published versions of milestoning;[19–21] it thus would be exact up to approximations associated with the loss in resolution upon projection to coarse-grained states.

It is worth noting that it is also straightforward to implement a single-step algorithm. The last stable state visited by each trajectory is stored and used to compute the probability that a trajectory that leaves one stable state enters the other without first returning. The flux out of each stable state can be easily computed from the simulation in the box containing that stable state. Unfortunately, this algorithm is inefficient in practice. To see that this is the case, consider the transition from the left to the right stable state in Fig. 2(b). The vast majority of paths entering the right stable state originate in that stable state itself and do not pass through the transition region near $\Delta=0$. Because our algorithm samples paths with their proper weights in the steady-state distribution, not only the ensemble of transition paths [as in FFS (Refs. 9 and 10)], rates converge slowly despite the fact that the region near $\Delta=0$ is populated at all times during the simulation. The design of an algorithm for transition rates that, like ours, allowed one to constrain sampling with intersecting interfaces is worth further consideration as it would provide a significant practical advantage over FFS for the study of systems without strong attractors.

The last method worth mentioning is that introduced recently for calculating a large deviation function, an analog of a Gibbs free energy that is a function of a "pressure" conjugate to the average of a static or dynamic order parameter over a long trajectory.[22,23] As in our algorithm, low probability states are sampled to a greater degree than in physically weighted simulations. However, the methods are quite different operationally. To obtain a large deviation function, statistics are accumulated during a diffusion Monte Carlo simulation biased with a nonequilibrium analog of a linear umbrella potential. More importantly, the Legendre transformation to obtain an analog of a Helmholtz free energy yields a distribution for an expectation value of an order parameter (i.e., an average over a long trajectory) rather than a distribution for its instantaneous value, as obtained in the present work. The two algorithms are thus complementary: that for large deviations provides enhanced sampling of rare trajectories, while ours enables one to focus sampling on a projected region of phase space to obtain a steady-state probability distribution with a desired accuracy. In contrast to the path sampling methods reviewed above,[9,10,12,13,15] neither algorithm relies on relaxation to strong attractors. Given that both algorithms can be employed with any (quasi)ergodic dynamics, we believe that together they open up the possibility of efficiently addressing a broad range of problems concerning irreversible systems that were previously intractable.

Note added in proof. Following acceptance of this paper, it was shown that it is possible to obtain the steady-state probability distribution from two FFS calculations performed in opposite directions [C. Valeriani, R. J. Allen, M. J. Morelli, D. Frenkel, and P. R. ten Wolde, J. Chem. Phys. **127**, 114109 (2007)]. Although statistics can be accumulated for multiple order parameters in these simulations, the sampling can only be constrained in one coordinate, and the algorithm is limited in applicability to systems with only two strong attractors. As such, our algorithm can be used to obtain steady-state distributions in a much wider class of problems than can that based on FFS.

ACKNOWLEDGMENTS

The authors thank David Chandler for a critical reading of the manuscript and Ron Elber for making us aware of the steady-state relaxation method (Ref. 20). One of the authors (A.W.) is a National Science Foundation Predoctoral Fellow. Another author (P. B.) was supported by a Searle Scholarship to one of the authors (A.R.D) and the NSF Materials Research Science and Engineering Center at The University of Chicago (Grant No. NSF-DMR-0213745).

[1] N. J. Guido, X. Wang, D. Adalsteinsson, D. McMillen, J. Hasty, C. R. Cantor, T. C. Elston, and J. J. Collins, Nature (London) **439**, 856 (2006).
[2] J. T. Mettetal, D. Muzzey, J. M. Pedraza, E. M. Ozbudak, and A. van Oudenaarden, Proc. Natl. Acad. Sci. U.S.A. **103**, 11549 (2006).
[3] J. P. Valleau and D. N. Card, J. Chem. Phys. **57**, 5457 (1972).
[4] J. M. Torrie and J. P. Valleau, J. Comput. Phys. **23**, 187 (1977).
[5] D. Chandler, *Introduction to Modern Statistical Mechanics* (Oxford University Press, New York, 1987).
[6] D. Frenkel and B. Smit, *Understanding Molecular Simulation: From Algorithms to Applications* (Academic, London, 2002).
[7] D. T. Gillespie, J. Phys. Chem. **81**, 2340 (1977).
[8] M. E. J. Newman and G. T. Barkema, *Monte Carlo Methods in Statistical Physics* (Oxford University Press, New York, 1999).
[9] R. J. Allen, P. B. Warren, and P. R. ten Wolde, Phys. Rev. Lett. **94**, 018104 (2005).
[10] R. J. Allen, D. Frenkel, and P. R. ten Wolde, J. Chem. Phys. **124**, 024102 (2006).
[11] M. M. de Oliviera and R. Dickman, Phys. Rev. E **71**, 016129 (2005).
[12] P. G. Bolhuis, D. Chandler, C. Dellago, and P. L. Geissler, Annu. Rev. Phys. Chem. **53**, 291 (2002).
[13] C. Dellago, P. G. Bolhuis, and P. L. Geissler, Adv. Chem. Phys. **123**, 1 (2002).
[14] G. E. Crooks and D. Chandler, Phys. Rev. E **64**, 026109 (2001).
[15] A. Bandrivskyy, S. Beri, D. G. Luchinsky, R. Mannella, and P. V. E. McClintock, Phys. Rev. Lett. **90**, 210201 (2003).
[16] D. Moroni, T. S. van Erp, and P. G. Bolhuis, Phys. Rev. E **71**, 056709 (2005).
[17] C. H. Bennett, in *Algorithms for Chemical Computations*, edited by R. E. Christofferson (American Chemical Society, Washington, DC, 1977), ACS Symposium Series No. 46.
[18] D. Chandler, J. Chem. Phys. **68**, 2959 (1978).
[19] A. K. Faradjian and R. Elber, J. Chem. Phys. **120**, 10880 (2004).
[20] D. Shalloway and A. K. Faradjian, J. Chem. Phys. **124**, 054112 (2006).
[21] A. M. A. West, R. Elber, and D. Shalloway, J. Chem. Phys. **126**, 145104 (2007).
[22] C. Giardina, J. Kurchan, and L. Peliti, Phys. Rev. Lett. **96**, 120603 (2006).
[23] V. Lecomte and J. Tailleur, J. Stat. Mech.: Theory Exp. P03004.
[24] A. Warmflash and A. R. Dinner (unpublished).