THE JOURNAL OF PHYSICAL CHEMISTRY B
A JOURNAL OF THE AMERICAN CHEMICAL SOCIETY

pubs.acs.org/JPCB
Article

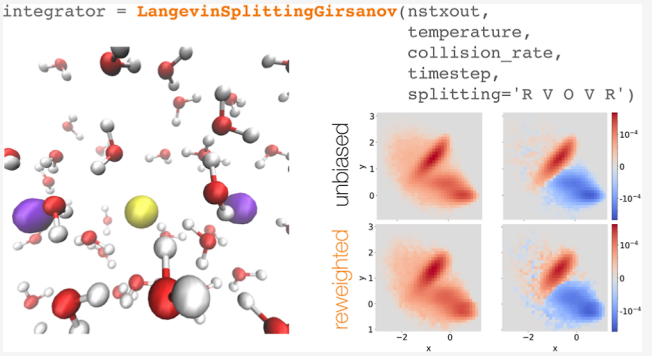
Implementation of Girsanov Reweighting in OpenMM and Deeptime

Published as part of The Journal of Physical Chemistry B virtual special issue "Recent Advances in Simulation Software and Force Fields".

Joana-Lysiane Schäfer and Bettina G. Keller*

Cite This: J. Phys. Chem. B 2024, 128, 6014–6027

ABSTRACT: Classical molecular dynamics (MD) simulations provide invaluable insights into complex molecular systems but face limitations in capturing phenomena occurring on time scales beyond their reach. To bridge this gap, various enhanced sampling techniques have been developed, which are complemented by reweighting techniques to recover the unbiased dynamics. Girsanov reweighting is a reweighting technique that reweights simulation paths, generated by a stochastic MD integrator, without evoking an effective model of the dynamics. Instead, it calculates the relative path probability density at the time resolution of the MD integrator. Efficient implementation of Girsanov reweighting requires that the reweighting factors are calculated on-the-fly during the simulations and thus needs to be implemented within the MD integrator. Here, we present a comprehensive guide for implementing Girsanov reweighting into MD simulations. We demonstrate the implementation in the MD simulation package OpenMM by extending the library openmmtools. Additionally, we implemented a reweighted Markov state model estimator within the time series analysis package Deeptime.



INTRODUCTION

Classical molecular dynamics (MD) simulations yield trajectories of a molecular system at atomistic resolution and are an excellent tool to study the dynamics of complex molecular systems. A complication emerges from the fact that relevant time scales of the molecular system, from slow conformational changes to complex formation and ultimately chemical reactions, occur on time scales that are orders of magnitude beyond the time scales that can be covered by an unbiased MD simulation. To close this time scale gap a wide variety of enhanced sampling techniques[1] have been proposed, which broadly can be classified into methods which (1) change the temperature of the system,[2,3] (2) change the classical Hamiltonian, in particular the potential energy function, of the system,[4−8] or (3) bias the initial state of the trajectory but otherwise leave the dynamics unchanged.[9,10] A necessary complement of enhanced sampling techniques are reweighting methods,[1,11,12] which recover the unperturbed stationary density and the unperturbed dynamics from biased simulations.

Girsanov reweighting[13−20] is a dynamical reweighting technique to recover the unbiased dynamics of a molecular system from simulations that were conducted with a biased potential. It differs from other dynamical reweighting techniques[21−25] in that it does not assume an effective model of the molecular dynamics but reweights the dynamics directly on the level of the stochastic MD integrator. In this sense, Girsanov reweighting is an exact reweighting technique. The method is based on the theory of stochastic path integrals.[26−28] It has been applied to model systems[13−17] and peptide dynamics.[19] Recently, it has been used for the machine learning of optimal collective variables based on enhanced sampling simulations.[29]

Girsanov reweighting can also be applied in a prospective manner. From MD trajectories obtained at a reference potential energy function, one can calculate the transition rates at a modified potential energy function and investigate the sensitivity of the transition rates with respect to certain parameters of the potential energy. In fact, while modern empirical potential energy functions often reproduce the structural and thermodynamic properties with high fidelity,[30,31]

Received: March 14, 2024
Revised: May 22, 2024
Accepted: May 22, 2024
Published: June 12, 2024

https://doi.org/10.1021/acs.jpcb.4c01702
J. Phys. Chem. B 2024, 128, 6014−6027

the dynamic properties, such as transition times across barriers, may vary considerably across different potential energy models for the same molecule.[32] In ref 33, Girsanov reweighting has been combined with the maximum caliber approach[34] to optimize potential energy functions such that the resulting dynamics reproduce a specific transition rate.

Since Girsanov reweighting is based on stochastic path integrals, it requires that the MD simulations are carried out with a stochastic MD integrator.[35] To achieve reweighting accuracy at the level of the MD integrator, the random numbers and bias force at each integration time step are collected and aggregated in path reweighting factors. The path reweighting factor can be calculated for any time interval $[t, t + \tau]$ of the entire MD trajectory and represents the relative probability of the path from t to $t + \tau$ at the target potential relative to its probability at the simulation potential.

Interpreting the path reweighting factor as the relative path probability gives an intuitive explanation of why stochastic MD integrators are needed for Girsanov reweighting. The probability of a specific deterministic path, by definition, is one at the simulation potential but zero at any other potential. Hence, the relative probability of a deterministic path, i.e., the ratio of the path probability between the target potential and the simulation potential, is always zero.

For stochastic integrators, the path reweighting factor can be calculated efficiently by formulating it in terms of the random numbers generated during a stochastic MD simulation and in terms of random number differences that account for the difference between the dynamics at the simulation potential and the dynamics at the target potential to which the simulated dynamics should be reweighted.[18,36] The equation for the random number difference depends on the integrator and can easily be derived for the Euler-Maruyama integrator,[28,35] of overdamped Langevin dynamics.[14−16] However, a much more realistic description of the molecular dynamics is achieved by underdamped Langevin dynamics. Ref 37 (unpublished results) presents a general approach to derive the relative path probability density for stochastic integrators of underdamped Langevin dynamics, and it provides a framework to analyze whether the relative path probability for a given integration algorithm is defined throughout the path space. If the relative path probability density is defined, its value can be calculated, and reweighting underdamped Langevin dynamics is thus possible.

While specialized simulation protocols for Girsanov reweighting have been published, there is a lack of ready-to-use simulation programs that allow on-the-fly estimation of reweighting factors. Calculating the relative path probability density requires access to the random numbers and forces at the time resolution of the MD integrator. Even though the implementation of the relative path probability density amounts to adding a reporter to the inner loop of the MD integrator, for most simulation packages this requires deep knowledge of the package architecture. The goal of this contribution is to provide a template for efficiently implementing the relative path probability density into an MD program. We use OpenMM[38] which provides an easy and versatile implementation of Langevin integrators via the openmmtools package,[39,40] to demonstrate the implementation.

As an example analysis of the simulation data, we reweight Markov state models (MSMs)[41−46] and implement the reweighting factor into the time-series analysis package Deeptime.[47] While Girsanov reweighting can be used to reweight other rate estimators,[33] MSMs have the advantage that they are well suited to model multistate dynamics and that, due to the short Markov lag time, a model of the long-time dynamics can be constructed from short paths. Our additions to the two software packages are open source and provide a blueprint for extending an MD simulation and analysis program to include Girsanov reweighting.

THEORY

Molecular Simulations and Path Probabilities. Consider a system of N atoms with Cartesian position vector $q \in \mathbb{R}^{3N}$ and associated momentum vector $p \in \mathbb{R}^{3N}$. Momentum and position vectors can be combined to a phase space vector $x = (q, p) \in \mathbb{R}^{6N}$. The time-evolution of such a system in a thermal bath is modeled by underdamped Langevin dynamics

$$M\ddot{q}(t) = -\nabla V(q(t)) - \xi M\dot{q}(t) + \sqrt{2RT\xi M}\eta(t) \quad (1)$$

The left-hand side of eq 1 represents the total force on the particles, where $\ddot{q}(t) = \frac{\partial^2}{\partial t^2}q(t)$ is the acceleration vector at time t. M is the $3N \times 3N$-dimensional mass matrix, which contains the masses of the particles along its diagonal.

The force consists of three terms (right-hand side of eq 1): (i) the force due to the gradient of the potential energy function $-\nabla V(q)$; (ii) the friction force, where $\dot{q}(t) = \frac{\partial}{\partial t}q(t) = v(t)$ is the velocity vector and ξ is the collision rate with unit s^{-1}; and (iii) a random force. According to the dissipation fluctuation theorem,[48] the random force along the lth degree of freedom has the mean 0 and variance $2RT\xi M_{ll}$, where R is the ideal gas constant, T is the temperature, and M_{ll} is the l diagonal element in M and represents the mass associated with the lth degree of freedom. We model this random force by an uncorrelated white Gaussian noise with unit variance centered at 0, $\eta(t)$, which is then scaled by $\sqrt{2RT\xi M_{ll}}$ in order to fulfill the fluctuation dissipation theorem.[48]

We report the potential in molar energy units, J mol^{-1}; correspondingly, the thermal energy is reported as molar quantity: RT. If energy units are used for the potential, R should be replaced by the Boltzmann constant $k_B = R/N_A$ in eq 1 and all of the following equations. N_A is the Avogadro constant.

Langevin integrators[35] are numerical schemes that provide an approximate solution to eq 1. They yield a time-discrete trajectory or path $\mathbf{x} = (x_0, x_1, x_2, \ldots x_n)$, where $x_0 = (q_0, p_0)$ is the initial state at time $t = 0$, $x_k = (q_k, p_k)$ is the state at time $t = k\Delta t$ with time step Δt. The path length is $\tau = n\Delta t$.

To model the random force, typical Langevin integrators draw either one or two random numbers from a standard normal distribution per integration time step and degree of freedom. The simulation thus additionally yields either one sequence of random number vectors, $\boldsymbol{\eta} = [\eta_0, \eta_1 \ldots \eta_{n-1}]$, or two sequences of random number vectors, $\boldsymbol{\eta}^{(1)} = [\eta_0^{(1)}, \eta_1^{(1)} \ldots \eta_{n-1}^{(1)}]$ and $\boldsymbol{\eta}^{(2)} = [\eta_0^{(2)}, \eta_1^{(2)} \ldots \eta_{n-1}^{(2)}]$, where $\eta_k \in \mathbb{R}^{3N}$, $\eta_k^{(1)} \in \mathbb{R}^{3N}$, and $\eta_k^{(2)} \in \mathbb{R}^{3N}$. The lth element of a random number vector η_k contains the random number that determines the random force on the lth degree of freedom at the time step k.

https://doi.org/10.1021/acs.jpcb.4c01702
J. Phys. Chem. B 2024, 128, 6014−6027

The path probability $\mathcal{P}[\mathbf{x}]$ is the probability of observing a specific time-discretized path $\mathbf{x}$. It can be decomposed using the Chapman−Kolmogorov equation[49] as

$$\mathcal{P}[\mathbf{x}] = p(x_0)\cdot\mathcal{P}[x_1 \ldots x_n|x_0] = p(x_0)\cdot\prod_{k=0}^{n-1} p(x_{k+1}|x_k) \tag{2}$$

where $p(x_0)$ is the probability density of the initial state. We here assume that the initial states are distributed according to the Boltzmann distribution

$$p(x_0) = p(q_0, p_0) = \frac{1}{Z}\exp\left(-\frac{V(q_0)}{RT}\right)\cdot\prod_{l=1}^{3N}\frac{1}{\sqrt{2RT\pi M_{ll}}}\exp\left(-\frac{1}{RT}\frac{p_{l,0}^2}{2M_{ll}}\right) \tag{3}$$

where $Z = \int \mathrm{d}q \exp\left(-\frac{V(q)}{RT}\right)$ is the classical configurational partition function,[50] and $p_{l,0}$ is the lth element of the initial momentum vector p_0.

In eq 2, $p(x_{k+1}|x_k)$ is the one-step transition probability to reach x_{k+1} within one integration step, given that the current state is x_k. Langevin integrators that only take current state x_k into account when updating position and momentum implement a Markov process. For these integrators, eq 2 is exact. For Langevin integrators that additionally take the previous state x_{k-1} into account, eq 2 is an approximation.

The mathematical expression for $p(x_{k+1}|x_k)$ depends on the Langevin integrator. One can however argue[36] that the single-step transition probability is equal to the probability of drawing the random number vector η_k that gives rise to this specific transition

$$p(x_{k+1}|x_k) = p(\eta_k) = \prod_{l=1}^{3N}\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{\eta_{l,k}^2}{2}\right) \tag{4}$$

where we used the random numbers $\eta_{l,k}$ that are drawn from a standard normal distribution. Equation 4 holds for the Langevin integrator, which draws a single random number per integration step and degree of freedom l. The path probability density (eq 2) can then be written as

$$\mathcal{P}[\mathbf{x}] = p(x_0)\cdot\prod_{k=0}^{n-1} p(\eta_k) = p(x_0)\cdot\prod_{k=0}^{n-1}\prod_{l=1}^{3N}\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{\eta_{l,k}^2}{2}\right) \tag{5}$$

Analogously, the path probability density (eq 2) for Langevin integrators that draw two random numbers per integration step and degree of freedom is

$$\mathcal{P}[\mathbf{x}] = p(x_0)\cdot\prod_{k=0}^{n-1} p(\eta_k^{(1)})p(\eta_k^{(2)}) = p(x_0)\cdot\prod_{k=0}^{n-1}\prod_{l=1}^{3N}\frac{1}{2\pi}\exp\left(-\frac{\eta_{l,k}^{(1)2}}{2}\right)\exp\left(-\frac{\eta_{l,k}^{(2)2}}{2}\right) \tag{6}$$

Girsanov Reweighting. Girsanov reweighting is an importance sampling technique[1] for path probability densities at different potential energy functions. Following the principles of importance sampling, a path expected value at target potential $\tilde{V}$ can be reweighted as

$$\langle s\rangle_{\mathrm{path},\tilde{V}} = \int_{\mathcal{S}} \mathcal{D}\mathbf{x}\tilde{\mathcal{P}}[\mathbf{x}]s[\mathbf{x}] = \int_{\mathcal{S}} \mathcal{D}\mathbf{x}\frac{\tilde{\mathcal{P}}[\mathbf{x}]}{\mathcal{P}[\mathbf{x}]}\mathcal{P}[\mathbf{x}]s[\mathbf{x}] \tag{7}$$

where $\tilde{\mathcal{P}}[\mathbf{x}]$ is the path probability density at the target potential $\tilde{V}(q)$, $\mathcal{P}[\mathbf{x}]$ is the path probability density at the simulation potential $V(q)$. $s[\mathbf{x}]$ is a path observable, i.e., a function that assigns a real-valued number to each path $\mathbf{x}$. $\langle\ldots\rangle$ denotes an expected value, where we added the subscript "path" to emphasize that the expected values is calculated with respect to path probability density $\mathcal{P}[\mathbf{x}]$ and not with respect to a phase space probability $p(x)$. The path integral $\int_{\mathcal{S}}\mathcal{D}\mathbf{x}\ldots$ integrates over the space $\mathcal{S}$ of all possible time-discretized paths of length $\tau = n\Delta t$.

If an analytical expression for the relative path probability $\tilde{\mathcal{P}}[\mathbf{x}]/\mathcal{P}[\mathbf{x}]$ can be found, the path expected value in eq 7 can be estimated from a set of paths $\mathrm{S} = (x_1,\ldots x_{n_{\mathrm{paths}}})$ generated at the simulation potential $V(q)$ as

$$\langle s\rangle_{\mathrm{path},\tilde{V}} = \lim_{n_{\mathrm{paths}}\to\infty}\sum_{\mathbf{x}_i\in S}\frac{\tilde{\mathcal{P}}[\mathbf{x}_i]}{\mathcal{P}[\mathbf{x}_i]}s[\mathbf{x}_i] \tag{8}$$

A critical question is under which conditions the relative path probability $\tilde{\mathcal{P}}[\mathbf{x}]/\mathcal{P}[\mathbf{x}]$ exists. For time-continuous paths, these conditions have been discussed by Girsanov[27] and Onsager and Machlup.[26] It depends on the integrator in the case of time-discretized paths; ref 37 (unpublished reference) discusses path probability ratios for Langevin splitting operators.

The potential energy $V(q)$, at which the simulation is carried out, and the target potential energy $\tilde{V}(q)$, to which the path expected value is reweighted, are related by a potential $U(q)$

$$\tilde{V}(q) = V(q) + U(q) \tag{9}$$

Equation 9 takes the viewpoint of a perturbation,[18,20,33,36] (ref 37 unpublished results), where the target potential differs from the reference (simulation) potential by a perturbation $U(q)$. Alternatively, one can take the viewpoint of enhanced sampling simulations, where a bias potential $U_{\mathrm{bias}}(q)$ needs to be subtracted from the simulation potential $V(q)$ to obtain the molecular (target) potential, thus $\tilde{V}(q) = V(q) - U_{\mathrm{bias}}(q)$. By setting $U(q) = -U_{\mathrm{bias}}(q)$, one can reconcile[19] the enhanced sampling viewpoint with eq 9. It is important to be aware of the sign-convention since it enters the equations for the reweighting factor.

Using eq 5, the path probability density ratio can be expressed as[36]

$$\frac{\tilde{\mathcal{P}}[\mathbf{x}]}{\mathcal{P}[\mathbf{x}]} = \frac{\tilde{p}(x_0)}{p(x_0)}\cdot\prod_{k=0}^{n-1}\prod_{l=1}^{3N}\frac{\exp\left(-\frac{\tilde{\eta}_{l,k}^2}{2}\right)}{\exp\left(-\frac{\eta_{l,k}^2}{2}\right)} = \frac{\tilde{p}(x_0)}{p(x_0)}\cdot\prod_{k=0}^{n-1}\prod_{l=1}^{3N}\exp\left(-\eta_{l,k}\cdot\Delta\eta_{l,k} - \frac{1}{2}\Delta\eta_{l,k}^2\right), \tag{10}$$

where $\tilde{\eta}_k$ is the random number vector that yields an update $x_k \to x_{k+1}$ at the target potential $\tilde{V}(q)$, and η_k is the random number vector that yields the same update at the simulation potential $V(q)$. The intuition is that one calculates $\tilde{\eta}_k$ and compares it to the random number vector η_k, which was used

https://doi.org/10.1021/acs.jpcb.4c01702
J. Phys. Chem. B 2024, 128, 6014−6027

in the simulation at $V(q)$. However, in our implementation, $\tilde{\eta}_k$ is never calculated directly. Instead, the two random number vectors are related by a random number difference vector $\Delta\eta_k$ with

$$\tilde{\eta}_{l,k} = \eta_{l,k} + \Delta\eta_{l,k} \tag{11}$$

The random numbers $\eta_{l,k}$ can be recorded during the simulation at simulation potential $V(q)$, but $\Delta\eta_{k,l}$ needs to be calculated at each integration time step. The equation for $\Delta\eta_{k,l}$ depends on the integrator, and the equations for various Langevin integrators are reported later in the text.

Analogously, using eq 6, the path probability density ratio can be expressed as[36]

$$\frac{\tilde{\mathcal{P}}[\mathbf{x}]}{\mathcal{P}[\mathbf{x}]} = \frac{\tilde{p}(x_0)}{p(x_0)} \cdot \prod_{k=0}^{n-1} \prod_{l=1}^{3N} \exp\left(-\eta_{l,k}^{(1)} \cdot \Delta\eta_{l,k}^{(1)} - \frac{1}{2}\Delta\eta_{l,k}^{(1)^2}\right) \exp\left(-\eta_{l,k}^{(2)} \cdot \Delta\eta_{l,k}^{(2)} - \frac{1}{2}\Delta\eta_{l,k}^{(2)^2}\right) \tag{12}$$

with random number differences

$$\begin{aligned} \tilde{\eta}_{l,k}^{(1)} &= \eta_{l,k}^{(1)} + \Delta\eta_{l,k}^{(1)} \\ \tilde{\eta}_{l,k}^{(2)} &= \eta_{l,k}^{(2)} + \Delta\eta_{l,k}^{(2)}. \end{aligned} \tag{13}$$

The formulas for $\Delta\eta_{l,k}$, $\Delta\eta_{l,k}^{(1)}$, and $\Delta\eta_{l,k}^{(2)}$ depend on the Langevin integrator[37] (unpublished reference).

The path probability ratio can be decomposed into the probability ratio of the initial state $\tilde{p}(x_0)/p(x_0)$ and the ratio of the path probabilities conditioned on the initial state

$$\frac{\tilde{\mathcal{P}}[\mathbf{x}]}{\mathcal{P}[\mathbf{x}]} = \frac{\tilde{p}(x_0)}{p(x_0)} \cdot M[\mathbf{x}|x_0] \tag{14}$$

where the conditional path probability ratio $M[\mathbf{x}|x_0]$ is given by the product over time steps k in eqs 10 or 12. If the initial state x_0 is drawn from the Boltzmann distribution eq 3, the probability ratio of the initial state in eqs 10 and 12 is given as

$$\frac{\tilde{p}(x_0)}{p(x_0)} = \frac{Z}{\tilde{Z}} \exp\left(-\frac{U(q_0)}{RT}\right) \tag{15}$$

Girsanov Reweighting for Langevin Integrators. Commonly used Langevin integrators are based on the splitting method.[51,52] Equation 1 is formulated as a vector field and separated in three terms, each of which is integrated separately, yielding three update operators,[35,51] (ref 37 unpublished results) for the position $q_{l,k}$ and the momentum $p_{l,k} = M_{ll}\dot{q}_{l,k}$

$$\begin{aligned} \mathcal{A}\begin{pmatrix} q_{l,k} \\ p_{l,k} \end{pmatrix} &= \begin{pmatrix} q_{l,k} + a p_{l,k} \\ p_{l,k} \end{pmatrix} \\ \mathcal{B}\begin{pmatrix} q_{l,k} \\ p_{l,k} \end{pmatrix} &= \begin{pmatrix} q_{l,k} \\ p_{l,k} + b(q_{l,k}) \end{pmatrix} \\ O\begin{pmatrix} q_{l,k} \\ p_{l,k} \end{pmatrix} &= \begin{pmatrix} q_{l,k} \\ d p_{l,k} + f\eta_{l,k} \end{pmatrix} \end{aligned} \tag{16}$$

where

$$\begin{aligned} a &= \Delta t \frac{1}{M_{ll}} \\ b(q) &= -\Delta t \frac{\partial}{\partial q_l} V(q) \\ d &= \exp(-\xi\Delta t) \\ f &= +\sqrt{k_B T M_{ll}(1 - \exp(-2\xi\Delta t))}. \end{aligned} \tag{17}$$

We have formulated the three update operators here as operators that act on a single degree of freedom l. In the simulation of a multidimensional system, these update operators are applied to each degree of freedom.

A full update of $x_k \rightarrow x_{k+1}$ is obtained by sequentially applying all three update operators. Different Langevin integrators can be derived by varying the sequence in which the update operators are applied. Symmetric splitting integrators use a symmetric sequence. An example is the ABOBA algorithm, which applies the sequence $\mathcal{A}'\mathcal{B}'O\mathcal{B}'\mathcal{A}'(q_k, p_k)^T$, where update operators that appear twice in the sequence are applied with half a time step and are denoted with a prime. The corresponding parameters a', $b'(q)$, d', and f' are obtained by replacing Δt with $\frac{\Delta t}{2}$ in eq 18

$$\begin{aligned} a' &= \frac{\Delta t}{2}\frac{1}{M_{ll}} \\ b'(q) &= -\frac{\Delta t}{2}\frac{\partial}{\partial q_l} V(q) \\ d' &= \exp\left(-\xi\frac{\Delta t}{2}\right) \\ f' &= +\sqrt{k_B T M_{ll}(1 - \exp(-\xi\Delta t))}. \end{aligned} \tag{18}$$

In eq 16, we formulated the updates for position and momenta, which leads to the ABO notation of the update operators.[51,53] The update can analogously be formulated for position and velocities,[39,52]

$$\begin{aligned} \mathcal{R}\begin{pmatrix} q_{l,k} \\ v_{l,k} \end{pmatrix} &= \begin{pmatrix} q_{l,k} + \bar{a} v_{l,k} \\ v_{l,k} \end{pmatrix} \\ \mathcal{V}\begin{pmatrix} q_{l,k} \\ v_{l,k} \end{pmatrix} &= \begin{pmatrix} q_{l,k} \\ v_{l,k} + \bar{b}(q_{l,k}) \end{pmatrix} \\ O\begin{pmatrix} q_{l,k} \\ v_{l,k} \end{pmatrix} &= \begin{pmatrix} q_{l,k} \\ \bar{d} v_{l,k} + \bar{f}\eta_{l,k} \end{pmatrix}, \end{aligned} \tag{19}$$

where an $\mathcal{R}$-step is the analogue of an $\mathcal{A}$-step, a $\mathbf{V}$-step is the analogue of a $\mathcal{B}$-step, and $\bar{a} = a/M_{ll}$, $\bar{b}(q_{l,k}) = b(q_{l,k})/M_{ll}$, $\bar{d} = d$, $\bar{f} = f/M_{ll}$. Equation 19 leads to the RVO notation of Langevin splitting operators.[39,52]

A systematic approach to derive the random number difference between an update $x_k \rightarrow x_{k+1}$ at the simulation potential $V(q)$ and the same update at the target potential $\tilde{V}(q)$ is given in ref 37 (unpublished results). The reference also discusses why the relative path probability density $\tilde{\mathcal{P}}[\mathbf{x}]/\mathcal{P}[\mathbf{x}]$ may not be defined for some Langevin splitting integrators. In these cases, path reweighting is not possible. For

https://doi.org/10.1021/acs.jpcb.4c01702
J. Phys. Chem. B 2024, 128, 6014−6027

Table 1. Random Number Differences as Function of Perturbation Force $-\frac{\partial}{\partial q_l}U$ or Bias Force $-\nabla U_{\text{bias}}$, Integration Step Δt, Dissipation d, and Fluctuation f Terms for Underdamped Langevin Integrators[a]

integrator	random number difference		integrator
(ABO)	Perturbation	bias	(RVO)
ABO	$\Delta\eta_k = \frac{d}{f}\Delta t\frac{\partial}{\partial q_l}U(q_{l,k+1})$	$\Delta\eta_k = -\frac{d}{f}\Delta t\frac{\partial}{\partial q_l}U_{\text{bias}}(q_{l,k+1})$	RVO
ABOBA	$\Delta\eta_k = \frac{(d+1)}{f}\frac{\Delta t}{2}\frac{\partial}{\partial q_l}U(l, q_{k+1/2})$	$\Delta\eta_k = -\frac{(d+1)}{f}\frac{\Delta t}{2}\frac{\partial}{\partial q_l}U_{\text{bias}}(q_{l,k+1/2})$	RVOVR
AOBOA	$\Delta\eta_k^{\text{comb}} = \frac{d'}{f'}\Delta t\frac{\partial}{\partial q_l}U(q_{l,k+1/2})$	$\Delta\eta_k^{\text{comb}} = -\frac{d'}{f'}\Delta t\frac{\partial}{\partial q_l}U_{\text{bias}}(q_{l,k+1/2})$	ROVOR
	$\eta_k^{\text{comb}} = d'\eta_k^{(1)} + \eta_k^{(2)}$	$\eta_k^{\text{comb}} = d'\eta_k^{(1)} + \eta_k^{(2)}$	
BOAOB	$\Delta\eta_k^{(1)} = \frac{d'}{f'}\frac{\Delta t}{2}\frac{\partial}{\partial q_l}U(q_{l,k})$	$\Delta\eta_k^{(1)} = -\frac{d'}{f'}\frac{\Delta t}{2}\frac{\partial}{\partial q_l}U_{\text{bias}}(q_{l,k})$	VOROV
	$\Delta\eta_k^{(2)} = \frac{1}{f'}\frac{\Delta t}{2}\frac{\partial}{\partial q_l}U(q_{l,k+1})$	$\Delta\eta_k^{(2)} = -\frac{1}{f'}\frac{\Delta t}{2}\frac{\partial}{\partial q_l}U_{\text{bias}}(q_{l,k+1})$	
OBABO/BP	$\Delta\eta_k^{(1)} = \frac{1}{f'}\frac{\Delta t}{2}\frac{\partial}{\partial q_l}U(q_{l,k})$	$\Delta\eta_k^{(1)} = -\frac{1}{f'}\frac{\Delta t}{2}\frac{\partial}{\partial q_l}U_{\text{bias}}(q_{l,k})$	OVRVO
	$\Delta\eta_k^{(2)} = \frac{d'}{f'}\frac{\Delta t}{2}\frac{\partial}{\partial q_l}U(q_{l,k+1})$	$\Delta\eta_k^{(2)} = -\frac{d'}{f'}\frac{\Delta t}{2}\frac{\partial}{\partial q_l}U_{\text{bias}}(q_{l,k+1})$	

[a] $-\frac{\partial}{\partial q_l}U(q_{l,k})$: perturbation force is evaluated before position update; $-\frac{\partial}{\partial q_l}U(q_{l,k+1})$: perturbation force is evaluated after position update; $-\frac{\partial}{\partial q_l}U(q_{l,k+1/2})$: perturbation force is evaluated at an intermediate position during update sequence of the Langevin integrator.

Langevin splitting integrators, which have a finite relative path probability density, the expressions for the random number differences are summarized in Table 1. The same equations for the random number difference are obtained when deriving them from the RVO update operators in eq 19. As an example, consider the RVO algorithm, whose random number difference is $\Delta\eta_{l,k} = (\overline{d}/\overline{f})\Delta t\frac{1}{M_{ll}}\frac{\partial}{\partial q_l}U(q_{l,k+1})$. But since $\overline{d} = d$ and $\overline{f} = f/M_{ll}$, we obtain $\Delta\eta_{l,k} = (d/f)\Delta t\frac{\partial}{\partial q_l}U(q_{l,k+1})$, which is the equation given in Table 1.

Reweighting a Markov State Model. In a Markov state model (MSM),[41−46] the $3N$-dimensional position space Ω is discretized into n_{state} nonoverlapping states Ω_i, i.e., $\cup_{i=1}^{n_{\text{states}}}\Omega_i = \Omega \subset \mathbb{R}^{3N}$. The probability vector $\mathbf{p}(t)$ contains the time-dependent probabilities $p_i(t)$ of finding the system in Ω_i at time t. The time-evolution of this discrete probability vector is then modeled as a Markov process

$$\mathbf{p}^T(t+\tau) = \mathbf{p}^T(t)\mathbf{P}(\tau) \tag{20}$$

where $\mathbf{P}(\tau)$ is the MSM transition matrix with dimension $n_{\text{states}} \times n_{\text{states}}$. τ is the MSM lag time. Its elements $P_{ij}(\tau) = \mathbb{P}(q(t+\tau) \in \Omega_j | q(t) \in \Omega_i)$ contain the conditional probability of finding the system in state Ω_j at time $t+\tau$, given that it has been in state Ω_i at time t. By definition, the matrix is row-normalized, such that $\sum_{j=1}^{n_{\text{states}}} P_{ij}(\tau) = 1$.

The conditional transition probability $P_{ij}(\tau)$ can be calculated from the absolute transition probability $C_{ij}(\tau)$ between states Ω_i and Ω_j as

$$P_{ij}(\tau) = \frac{C_{ij}(\tau)}{\sum_{j=1}^{n_{\text{states}}} C_{ij}(\tau)} \tag{21}$$

The absolute transition probability can be formulated as a path integral.[17,18]

$$C_{ij}(\tau) = \int_{\mathcal{S}} \mathcal{D}\mathbf{x} h_i(q_0)\mathcal{P}[\mathbf{x}]h_j(q_n) \tag{22}$$

where $h_i(q)$ is the indicator function of state Ω_i

$$h_i(q) = \begin{cases} 1, & q \in \Omega_i \\ 0, & \text{otherwise} \end{cases} \tag{23}$$

Analogously, $h_j(q)$ is the indicator function of state Ω_j. The path observable $s[\mathbf{x}] = h_i(q_0)h_j(q_n)$ thus evaluates to one if the path $\mathbf{x}$ starts in Ω_i and ends in Ω_j and to zero otherwise. $C_{ij}(\tau)$ can be reweighted according to eq 8, and the reweighted estimator is

$$\tilde{C}_{ij}(\tau) = \lim_{n_{\text{paths}}\to\infty}\frac{1}{n_{\text{paths}}}\sum_{\mathbf{x}_m \in S}\frac{\tilde{\mathcal{P}}[\mathbf{x}_m]}{\mathcal{P}[\mathbf{x}_m]}h_i(q_0^{(m)})h_j(q_n^{(m)}) \tag{24}$$

where $S = (x_1,...x_{n_{\text{paths}}})$ is a set of paths of length $\tau = n\Delta t$ generated at the simulation potential $V(q)$, $q_0^{(m)}$ is the initial position of the mth path and $q_n^{(m)}$ is its last position. $\tilde{C}_{ij}(\tau)$ is the absolute transition probability at a target potential $\tilde{V}(q)$. Note that due to the normalization of the transition probability in eq 21, the constant ratio of the partition functions $Z/\tilde{Z}$ appearing in eq 15 cancels.[18]

Once the reweighted MSM transition matrix $\tilde{\mathbf{P}}(\tau)$ has been calculated, the MSM is analyzed via the eigenvalue and eigenvectors of $\tilde{\mathbf{P}}(\tau)$

$$\tilde{\mathbf{l}}^T\tilde{\mathbf{P}}(\tau) = \tilde{\lambda}_i(\tau)\tilde{\mathbf{l}}^T \qquad i = 0, 1 ...(n_{\text{states}} - 1) \tag{25}$$

where $\mathbf{l}_i$ is the ith left eigenvector and $\lambda_i(\tau)$ is the associated eigenvalue. Due to the row-normalization of $\tilde{\mathbf{P}}(\tau)$, the leading

https://doi.org/10.1021/acs.jpcb.4c01702
J. Phys. Chem. B 2024, 128, 6014−6027

eigenvalue is always $\lambda_0 = 1$ and its associated eigenvector $\mathbf{l}_0$ is the stationary density of the MSM. For a molecular system evolving according to eq 1, it should be equal to the (discretized) configurational Boltzmann density. The slow dynamic processes are represented by eigenvectors with eigenvalues close to 1.

In Markovian dynamics, the eigenvalues of the MSM transition matrix decay exponentially with the lag time τ: $\tilde{\lambda}_i(\tau) = \exp(-\tau/\tilde{t}_i^{\mathrm{ITS}})$. Thus, if the time-evolution of $\mathbf{p}(t)$ can indeed be modeled by a Markov process, the implied time scale should be independent of τ

$$\tilde{t}_i^{\mathrm{ITS}} = -\frac{\tau}{\ln(\tilde{\lambda}_i(\tau))} \tag{26}$$

The implied time scale test[43] uses eq 26 to check the approximation quality of an MSM by calculating MSM transition matrices at various lag times and checking whether the right-hand side of eq 26 is indeed independent of τ.

METHODS

General Approach. Performing Girsanov path reweighting for MSMs requires two main steps: (i) computing the relative path probability (eqs 10 and 15) and (ii) estimating the reweighted MSM transition matrix (eq 24). The first step is handled by the MD simulation program on-the-fly during the simulation. The second step is handled by an MD analysis program. The data that is exchanged between the simulation and the analysis program are the position trajectory and an associated trajectory of path reweighting factors. Figure 1 illustrates the data flow.

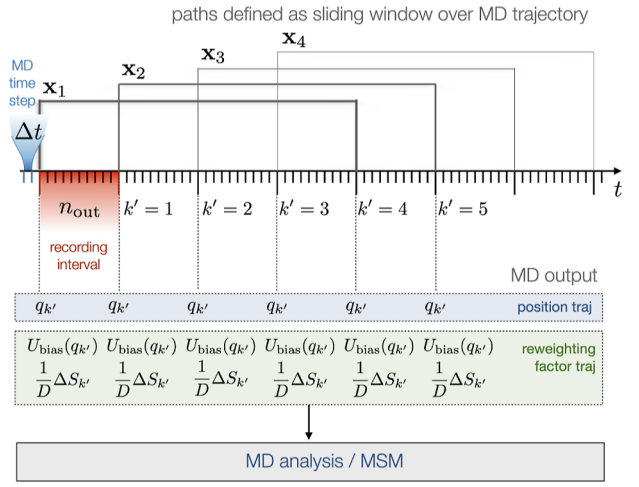


Figure 1. Overview of path reweighting.

In path reweighting, and in Girsanov reweighting in particular, the role of MD simulation is to generate a set of paths $S = (x_1,...x_{n_{\mathrm{paths}}})$ at the simulation potential $V(q)$. While it is possible to run a separate MD simulation for each path $\mathbf{x}_i$, in the context of MSMs the paths are usually extracted from a long trajectory via a sliding window. The path length is then $\tau = n\Delta t = n_{\mathrm{out}} \cdot n_{\mathrm{interval}}\Delta t$, where n_{out} represents the interval (in number of MD time steps Δt) between writing coordinates to the output trajectory file, and n_{interval} is the number of intervals $n_{\mathrm{out}}\Delta t$ that fit into the path length τ. To illustrate, in Figure 1 coordinates are written to file every $n_{\mathrm{out}} = 10$ time steps, and $n_{\mathrm{interval}} = 4$ of these recording intervals fit into a path, yielding a path length of $\tau = 40\Delta t$. We denote the iteration of the MD integrator with index k and the index of the recorded coordinates by k'. The indices are related via $k = k' n_{\mathrm{out}}$.

path probability density is calculated and written to a file at the same interval as the coordinates. To do this efficiently, the relative path probability for a path starting at $k = k' = 0$ is decomposed into a product of two factors as shown in eq 14. Inserting eq 15 and omitting the factor $Z/\tilde{Z}$ yields

$$\begin{aligned}\frac{\tilde{\mathcal{P}}[\mathbf{x}]}{\mathcal{P}[\mathbf{x}]} &\propto \exp\left(-\frac{U(q_0)}{RT}\right)\cdot M[\mathbf{x}|x_0].\\ &\propto \exp\left(\frac{U_{\mathrm{bias}}(q_0)}{RT}\right)\cdot M[\mathbf{x}|x_0].\end{aligned} \tag{27}$$

We omitted the factor $Z/\tilde{Z}$ because it cancels in the equation for the MSM transition probability (eq 21). The factor also cancels in other rate estimators.[33]

Using a sliding window, path $\mathbf{x}_i$ may start at any time-point $k = k' n_{out}$ at which coordinates are written to file. The indices in eq 27 then shift accordingly, i.e., $q_0 \to q_{k'}$ and $[x_0, x_1...x_n] \to [x_{k'}, x_{k'+1}...x_{k'+n}]$. To be able to calculate the relative probability of the initial state of a sliding window path, the perturbation energy $U(q_{k'})$ (or equivalently the bias energy $U_{\mathrm{bias}}(q_{k'})$) is written to the reweighting file, whenever coordinates $q_k = q_{k' n_{\mathrm{out}}}$ are written to file.

For a Langevin integrator which draws a single random number per degree of freedom and simulation time step, $M[\mathbf{x}|x_0]$ is given by the double product in eq 10. To account for the recording interval, we reformulate the factor as

$$\begin{aligned}M[\mathbf{x}|x_0] &= \prod_{k'=0}^{n_{\mathrm{interval}}-1}\prod_{k''=0}^{n_{\mathrm{out}}-1}\prod_{l=1}^{3N}\exp\Bigg(-\eta_{l,k' n_{\mathrm{interval}}+k''}\cdot\\ &\qquad \Delta\eta_{l,k' n_{\mathrm{interval}}+k''} - \frac{1}{2}\Delta\eta^2_{l,k' n_{\mathrm{interval}}+k''}\Bigg)\\ &= \exp\left(\sum_{k'=0}^{n_{\mathrm{interval}}-1} -\frac{1}{D}\Delta S_{k'}\right),\end{aligned} \tag{28}$$

where we split the product over the integration time steps k into an outer product over the recording intervals k' and an inner product over the simulation time steps k'' within a given recording interval. The indices in eqs 10 and 29 are related by $k = k' n_{\mathrm{interval}} + k''$. Furthermore, we have

$$\begin{aligned}\frac{1}{D}\Delta S_{k'} = \sum_{k''=0}^{n_{\mathrm{out}}-1}\sum_{l=1}^{3N} &\eta_{l,k' n_{\mathrm{interval}}+k''}\cdot\Delta\eta_{l,k' n_{\mathrm{interval}}+k''}\\ &+ \frac{1}{2}\Delta\eta^2_{l,k' n_{\mathrm{interval}}+k''}\end{aligned} \tag{29}$$

$\frac{1}{D}\Delta S_{k'}$ is calculated in the inner loop of the MD simulation program. At each integration time step k and for each degree of freedom, the random number $\eta_{l,k} = \eta_{l,k' n_{\mathrm{interval}}+k''}$ is recorded, and the random number difference $\Delta\eta_{l,k} = \Delta\eta_{l,k' n_{\mathrm{interval}}+k''}$ according to the appropriate equation in Table 1. The terms $\eta_{l,k}\Delta\eta_{l,k} + \frac{1}{2}\Delta\eta^2_{l,k}$ are then summed over all degrees of freedom l, and the result is added to a buffer variable $\frac{1}{D}\Delta S_{k'}$. The buffer variable is accumulated throughout the recording

https://doi.org/10.1021/acs.jpcb.4c01702
J. Phys. Chem. B 2024, 128, 6014−6027

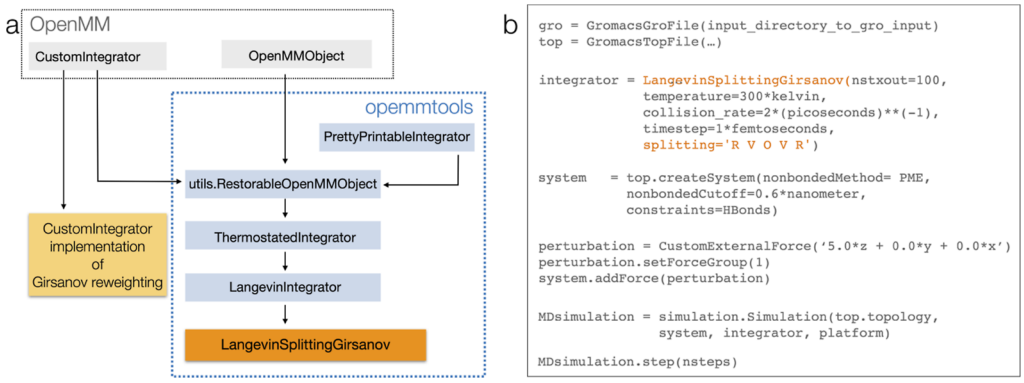


Figure 2. Implementation of the Girsanov reweighting in OpenMM via openmmtools. (a) Inheritance structure for the class LangevinSplittingGirsanov; (b) sketch of an OpenMM simulation script. `LangevinSplittingGirsanov` is called analogously to `LangevinInegrator`.

interval, i.e., for n_{out} steps. When the recording interval has elapsed and coordinates are written to the position trajectory, the value of the buffer variable $\frac{1}{D}\Delta S_{k'}$ is written to the reweighting factor trajectory, and the buffer variable is reset to zero.

To calculate $\Delta\eta_k$, one needs the bias force $-\nabla U_{\text{bias}}$ or the perturbation force $-\nabla U$. When Girsanov reweighting is used to unbias a biased simulation, the bias force is already available within the inner loop of the simulation. When Girsanov reweighting is used in a prospective manner to predict the influence of a perturbation $U(q)$ on a reference potential $V(q)$, perturbation forces $-\nabla U(q)$ need to be calculated in addition to the forces $-\nabla V(q)$ that are used to propagate the system.

For Langevin integrators that draw two random numbers per integration time step, $\frac{1}{D}\Delta S_{k'}$ is given as

$$\begin{aligned}\frac{1}{D}\Delta S_{k'} = &\sum_{k''=0}^{n_{\text{out}}-1}\sum_{l=1}^{3N}\eta^{(1)}_{l,k'n_{\text{interval}}+k''}\cdot\Delta\eta^{(1)}_{l,k'n_{\text{interval}}+k''}\\ &+\frac{1}{2}(\Delta\eta^{(1)}_{l,k'n_{\text{interval}}+k''})^2\\ &+\sum_{k''=0}^{n_{\text{out}}-1}\sum_{l=1}^{3N}\eta^{(2)}_{l,k'n_{\text{interval}}+k''}\cdot\Delta\eta^{(2)}_{l,k'n_{\text{interval}}+k''}\\ &+\frac{1}{2}(\Delta\eta^{(2)}_{l,k'n_{\text{interval}}+k''})^2.\end{aligned} \tag{30}$$

The variable name $\frac{1}{D}\Delta S_{k'}$ is a nod to the path action, where D is the diffusion constant. When formulating the path probability in terms of the path action,[15,26,54] $S[\mathbf{x}]$, eqs 29 and 30 can be interpreted as the action difference between the simulation and the target potential, scaled by the diffusion constant.

Given the position trajectory at the simulation potential $V(q)$ and the trajectory of path reweighting factors, an MSM at the target potential $\tilde{V}(q)$ can be obtained via eq 24. The relative path probability density is calculated from the path reweighting factors by summing up $\frac{1}{D}\Delta S_{k'}$ for all $n_{\text{intervals}}$ recording intervals that make up the path $\mathbf{x}$ (eq 28) and multiplying the resulting conditional path reweighting factor $M[\mathbf{x}|x_0]$ with the relative probability density of the initial state of the path (eq 27).

Implementation in OpenMM via Openmmtools. Openmmtools is a Python library layer that provides tools that allow for a user-friendly setup of complex simulation protocols. For our purpose, the most important openmmtools class is `LangevinIntegrator`. This class ultimately extends the OpenMM `CustomIntegrator` class (Figure 2) and provides access to Langevin Splitting integrators.[35,52,53] Openmmtools used the RVO notation. The precise integrator can be conveniently specified by passing the sequence of update operators (eq 16) as a string. `LangevinIntegrator` implements the update operators as member functions `_add_R_step()`, `_add_V_step()`, and `_add_O_step()`. The sequence of update operations is realized by the function `_add_integrator_steps()`, which reads a string input, e.g., 'R V O V R' and outputs the corresponding update sequence.

To implement the reweighting factor trajectory, we created the new class `LangevinSplittingGirsanov` within openmmtools. With this new class, Girsanov reweighting simulations can be set up simply by specifying the Langevin integrator as a string, as one would normally do for a Langevin simulation with openmmtools. Figure 2b shows a sample OpenMM script for a Girsanov reweighting simulation.

`LangevinSplittingGirsanov` extends `LangevinIntegrator` and thus inherits all methods from `LangevinIntegrator` and its parent classes (Figure 2). In this way, variables for M_{ll}, T, ξ, Δt, etc. are automatically set. The random number differences $\Delta\eta_{l,k}$ are implemented as a dispatch table, which can be accessed via the class function `_delta_eta_table(self)`. When initializing the `LangevinSplittingGirsanov` with a string of update operators, the `__init__`-function checks whether Girsanov reweighting is possible for this particular Langevin integrator and whether the random number difference has been implemented. The variable $\frac{1}{D}\Delta S_{k'}$ is provided by the function `_get_logM()`, which performs the summation over the $3N$ degrees of freedom, the recording interval n_{out}, and, if necessary, the number of random numbers drawn per integration step. An additional factor sets the buffer variable for $\frac{1}{D}\Delta S_{k'}$ to zero before the next recording step is performed.

An OpenMM simulation requires the input of an external force field file, which is processed to the OpenMM `System` object and can be called within the inner loop of the simulation; for more details cf. Figure S1. Additional bias forces can be provided by an interface like openmm-plumed[55] and are stored in a separate force group of the OpenMM `Force` object. To ensure the bias force update at the

https://doi.org/10.1021/acs.jpcb.4c01702
J. Phys. Chem. B 2024, 128, 6014−6027

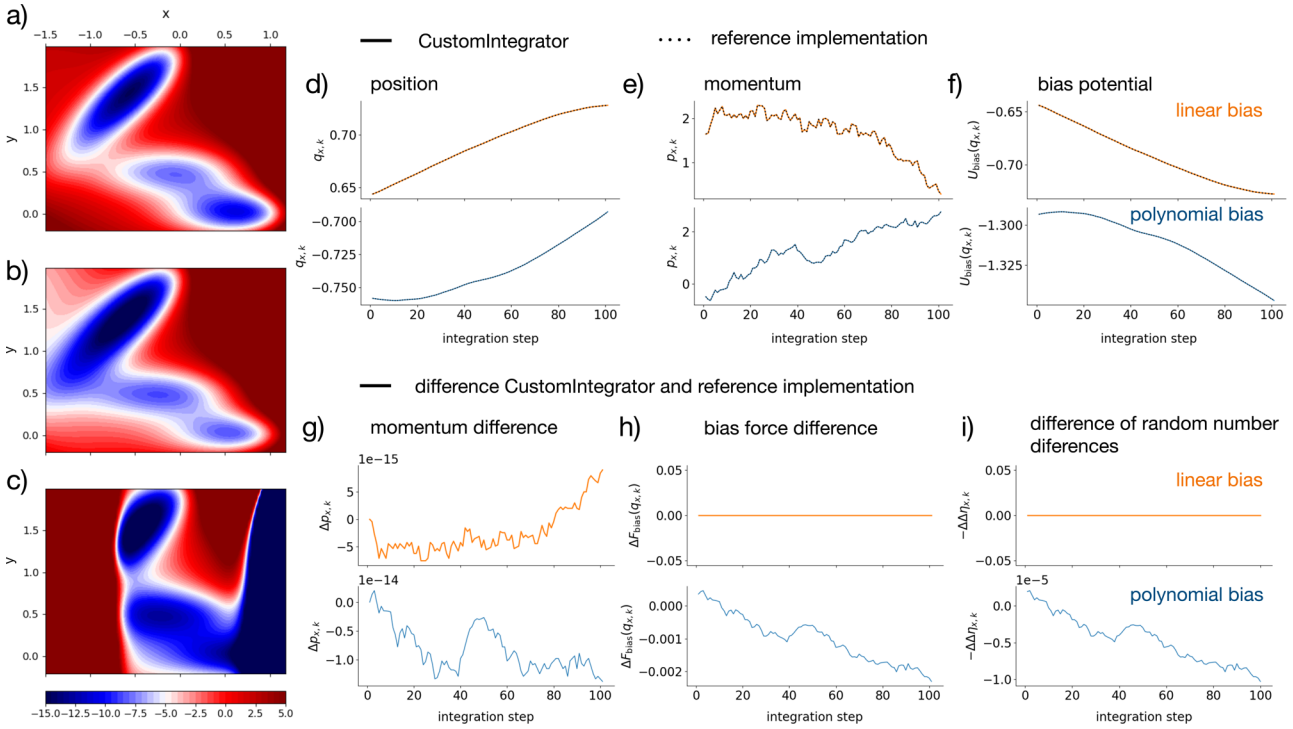


Figure 3. Comparison of the reference implementation (dotted) and the OpenMM CustomIntegrator class (solid) (a) Müller−Brown potential; (b) Müller−Brown potential and linear bias; (c) Müller−Brown potential and polynomial bias; Trajectories for 100 time steps of simulation: (d) x-component of the position, (e) x-component of the momentum, and (f) potential energy due to the bias. Difference between reference implementation and the OpenMM CustomIntegrator class (solid) for 100 time steps of simulation: (g) difference in x-component of thethe momentume momentum, (h) difference in the bias force $F_{\text{bias}}(q_k) = -\frac{\partial}{\partial x}U_{\text{bias}}(q_k)$, and (i) difference in the random number difference $\Delta\eta_k$.

integration step corresponding to the Langevin splitting scheme cf. Table 1, we have adapted `_add_integrator_steps()` to call the bias force group accordingly. Furthermore, we provide reporter class `ReweightingReporter` to record the path reweighting factors.

Implementation in Deeptime. Deeptime[47] is a Python library for the analysis of time series data and offers a wide range of tools to construct and analyze Markov models via the module `markov`. It implements estimation of a count matrix from a discretized trajectory via the function `count_matrix_coo2_mult(⋯)`, which is provided by the module `deeptime.markov.tools.estimation`. The function currently implements the direct maximum-likelihood estimator for the matrix elements $C_{ij}(\tau)$, which is obtained by setting the path probability ratio in eq 8 to one. We modified this function to implement the reweighted estimator in eq 24. The modified function accepts a trajectory of reweighting factors as an argument in addition to the discretized trajectory. If a trajectory of reweighting factors is passed to the function, the reweighted estimator is calculated; otherwise, it calculates the direct estimator. For each lag time τ, the reweighting factors are aggregated according to eqs 28 and 27 and used to calculate the reweighted count estimate according to eq 24.

The extended function is called by the class `GirsanovReweightingEstimator`, which is a child class of the class `TransitionCountEstimator`. The `MaximumLikelihoodMSM` class is available as a higher-level estimator of the MSM transition probabilities, which inputs the `GirsanovReweightingEstimator`, like the `TransitionCountEstimator`, and transforms the reweighted count matrix (21) into a transition matrix (eq 21). From there, all functionalities available in Deeptime, including the evaluation of the dominant eigenvalues (eq 25) and implied time scales (eq 26) for a series of lag times, can be achieved by feeding this transition matrix into the `MarkovStateModel` class.

RESULTS

In this section, the performance of the LangevinSplittingGirsanov class is investigated. Being part of the openmmtools package, the LangevinSplittingGirsanov class receives its forces from an OpenMM simulation object, which requires a molecular force field as the input (Figure 2). The LangevinSplittingGirsanov class thus cannot be directly compared to a reference implementation of a Langevin dynamics on an arbitrary low-dimensional potential. The OpenMM CustomIntegrator class can handle arbitrary low-dimensional potential as well as molecular force fields. To ensure consistency, we first compare the reference implementation to the OpenMM CustomIntegrator class using the Müller−Brown potential.[56] Then we compare the OpenMM CustomIntegrator class to our LangevinSplittingGirsanov class by using a dissociation process in water.

Müller−Brown Potential. We use the Müller−Brown potential $\tilde{V}_{\text{MB}}(x, y)$[56] (Figure 3a, parametrized according to eq 31 and Table 2) to compare our reference implementation to the implementation using the OpenMM CustomIntegrator class. The potential is characterized by a global minimum around $(-0.5, 1.5)$ and two local minima around $(0.0,0.5)$ and $(0.5,0.0)$. The system was propagated on two biased potentials $V(x, y) = \tilde{V}_{\text{MB}}(x, y) + U_{\text{bias}}(x)$, where the first potential had a

https://doi.org/10.1021/acs.jpcb.4c01702
J. Phys. Chem. B 2024, 128, 6014−6027

Table 2. Parameters for Müller–Brown Potential

	$n = 1$	$n = 2$	$n = 3$	$n = 4$	units
A_n	−20.0	−10.0	−17.0	1.5	kJ/mol
a_n	−1.0	−1.0	−6.5	0.7	nm^{-2}
b_n	0.0	0.0	11.0	0.6	nm^{-2}
c_n	−10.0	−10.0	−6.5	0.7	nm^{-2}
x_n	1.0	0.0	−0.5	−1.0	nm
y_n	0.0	0.5	1.5	1.0	nm

linear bias along x (Figure 3b), and the second potential had a strong polynomial bias along x (Figure 3c), cf. eq 32. The second form is motivated by a plumed-like bias, which is applied along a reaction coordinate and thus approximates a polynomial of the potential function. The linear bias was chosen to avoid position dependence in the force calculation. A consequence of applying the bias along x is that the path probabilities for the y-component of the path are the same in the simulation and the target potential.

The reference implementation uses the analytical forces from this potential and the Langevin splitting integrator ABOBA[35] to propagate the system. The OpenMM CustomIntegrator class uses numerical forces provided by OpenMM and also the Langevin splitting integrator ABOBA[35] to propagate the system. Similarly, analytical bias forces are used to calculate the path reweighting factors $\frac{1}{D}\Delta S_{k'}$ (eq 29) in the reference implementation, whereas numerical forces are used for this purpose in the OpenMM CustomIntegrator class.

Figure 3d–f shows the x-component of the position and momentum trajectory, as well as the trajectory of the bias energy for 100 simulation steps. We used the same random number sequence in both implementations. The simulations with the OpenMM CustomIntegrator class are shown as colored solid lines: orange for the linear bias and blue for the polynomial bias. The trajectories from the reference implementation are shown as black dotted lines. Visually the trajectories from the two implementations coincide. In fact, the difference between the two implementations is in the range of floating point precision. Figure 3g shows the difference of the momentum trajectories between the two implementations, which is on the order of 10^{-14} nm amu over 100 time steps, as an example. The difference of the position trajectory and the difference of the bias energy are shown in Figure S2. However, we do find that the bias force can differ in the two implementations (Figure 3h). While for the linear bias, analytical and numerical bias forces are identical (orange line in Figure 3h), the numerical force deviates from the analytical force for a nonlinear bias potential (blue line in Figure 3h). Since the bias force is used to calculate the random number difference, $\Delta\eta_k$ differs between the two implementations for the nonlinear bias (Figure 3i).

In summary, apart from the difference between analytical and numerical forces, the OpenMM CustomIntegrator implementation reproduces the trajectories and path reweighting factor trajectories from our reference implementation.

Next, we confirmed that we can reweight an MSM using the OpenMM CustomIntegrator class. Figure 4b shows the left dominant MSM eigenvectors $\tilde{l}_0$ and $\tilde{l}_1$ of the unbiased Müller–Brown potential, which is our target potential. $\tilde{l}_0$ represents the stationary density of the Müller–Brown potential. $\tilde{l}_1$ changes sign at the largest barrier of the potential and represents equilibration across this barrier. The MSM was obtained by a simulation at the unbiased Müller–Brown potential $\tilde{V}_{\mathrm{MB}}(x, y)$. Figure 4c shows the same eigenvectors obtained by simulating at a linearly biased potential $V(x, y) = \tilde{V}_{\mathrm{MB}}(x, y) + U_{\mathrm{bias}}(x)$ and then reweighting the MSM estimator (eq 24). The reweighted eigenvectors are in excellent agreement with the eigenvectors obtained from the unbiased simulation. Figure 4a shows the implied time scales of the MSM from the unbiased simulations (gray dashed line) and the MSM of the biased simulation without reweighting (dotted orange line) and with reweighting (solid orange line). Also for the implied time scales, the reweighted results agree very well with the results obtained from unbiased simulations.

I^-–Ca^{2+}–I^- Dissociation Process in Water. We use simulations of CaI_2 in TIP3P water (Figure 5a) to compare the implementation using the OpenMM CustomIntegrator class to our new LangevinSplittingGirsanov class (Figure 5b–g). The I^- anions were position restrained such that they maintained a

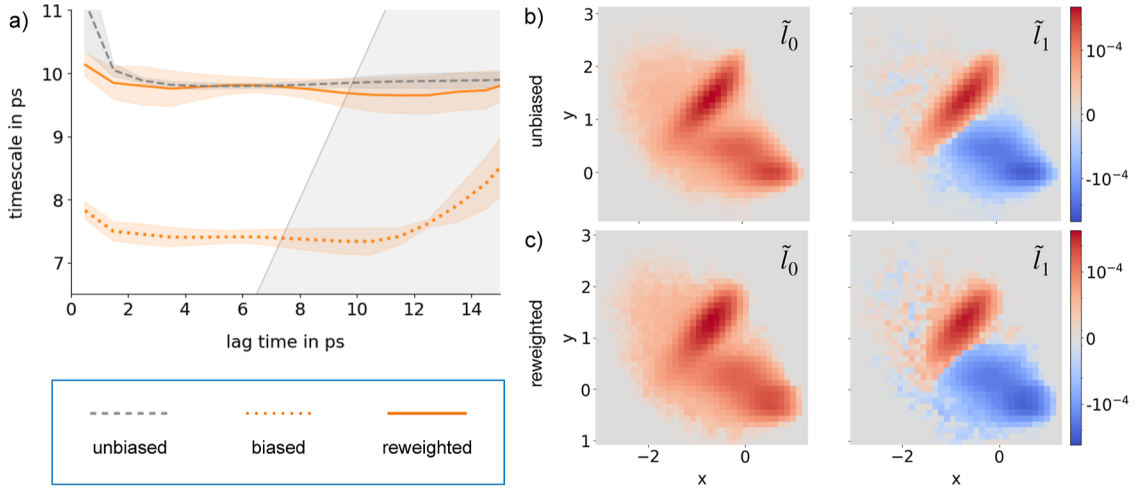


Figure 4. MSMs of the dynamics on the Müller–Brown potential. (a) Implied time scale $\tilde{t}_1^{\mathrm{ITS}}$ associated with the slowest process. (b) MSM eigenvectors $\tilde{l}_0$ and $\tilde{l}_1$ from an unbiased simulation on the Müller–Brown potential. (c) Reweighted MSM eigenvectors $\tilde{l}_0$ and $\tilde{l}_1$ from a simulation on the linearly biased Müller–Brown potential.

https://doi.org/10.1021/acs.jpcb.4c01702
J. Phys. Chem. B 2024, 128, 6014–6027

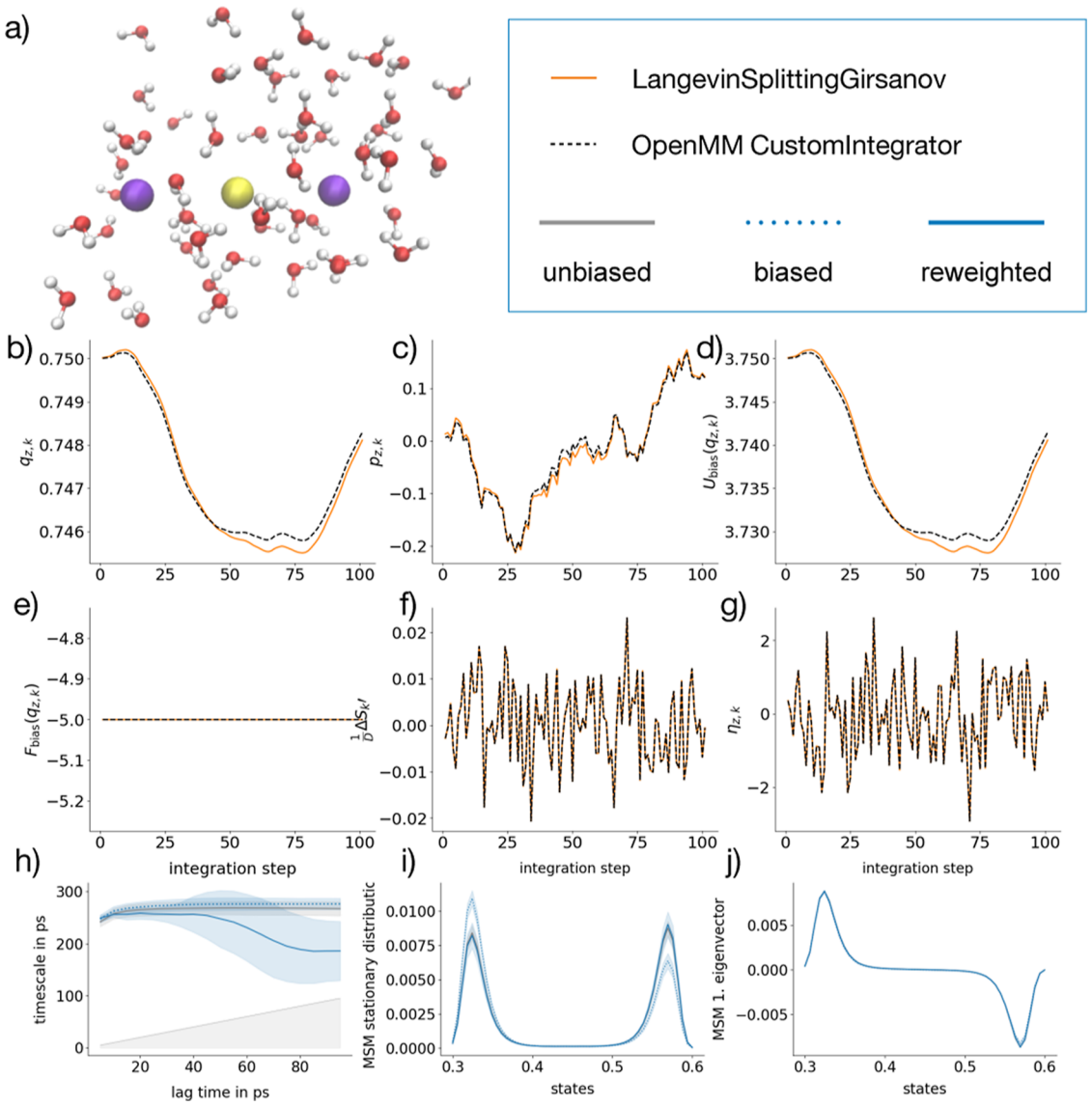


Figure 5. Molecular system of $Ca^{2+}I_2^-$ ion pair in TIP3P water is shown in (a). Comparison between OpenMM CustomIntegrator (black dashed) and LangevinSplittingGirsanov class (orange solid) for 100 time steps of simulation: (b) position $q_{z,k}$, (c) momentum $p_{z,k}$, (d) bias potential $U_{\text{bias}}(q_{z,k})$, (e) bias force $F_{\text{bias}}(q_{z,k})$, (f) reweighting factors $\frac{1}{D}\Delta S_{k'}$, and (g) random number $\eta_{z,k}$. Implied time scales from I^-−Ca^{2+}−I^- reference simulation at a unbiased potential (gray solid), as well as from a reweighted MSM (solid) and a standard MSM (dotted) from linearly biased simulations with k_z = 5 kJ/mol/nm in blue (h). First (i) and second (j) left dominant MSM eigenvectors are shown, with the same color code used in (h).

distance of 0.9 nm along the z-coordinate. The Ca^{2+} cation was allowed to diffuse between these two anions, by applying medium strong harmonic restraints along the y and x coordinates of the Ca^{2+} cation. The resulting dynamics resembles a double-well dynamics along z, where the Ca^{2+} cation alternates between being bound to the I^- anion and being bound to the other I^- anion. In between the two bound states, there is a considerable free-energy barrier. The target potential $\tilde{V}(q)$ consists of the molecular potential for this system including the above-mentioned restraints. In the potential $V(q) = \tilde{V}(q) + U_{\text{bias}}(q)$, a linear bias potential along z is added, where q is the position vector of all atoms in the system.

Similar to the Müller−Brown potential (Figure 3), we compared the two implementations for a trajectory of 100 time steps, which was generated at the biased potential $V(q)$. The two simulations use the same random number sequence for each degree of freedom. We observe a slight difference in the position and momentum trajectories between the OpenMM CustomIntegrator and the LangevinSplittingGirsanov. We suspect this can be attributed to slight algorithmic differences between our OpenMM CustomIntegrator implementation of the ABOBA algorithm and the implementation of the ABOBA algorithm via the parent class of LangevinSplittingGirsanov, the class LangevinIntegrator. Apart from this slight drift, the trajectory from our LangevinSplittingGirsanov class follows the trajectory from the OpenMM CustomIntegrator class very closely (Figure 5b−d).

Figure 5e−g shows trajectories of the bias force $F_{\text{bias}}(q_{z,k})$, of $\frac{1}{D}\Delta S_{k'}$, and of the random number $\eta_{z,k}$. The trajectories from the two implementations are identical for the linear bias (Figure S3a,b). Thus, the LangevinSplittingGirsanov imple-

https://doi.org/10.1021/acs.jpcb.4c01702
J. Phys. Chem. B 2024, 128, 6014−6027

mentation reproduces the trajectories and path reweighting factors from the OpenMM CustomIntegrator implementation.

Figure 5h–j tests whether we can reweight an MSM of the CaI_2 dynamics using the LangevinSplittingGirsanov class. The MSM has been constructed by discretizing the z-coordinate, thereby neglecting the solvent degrees of freedom. Figure 5i shows the eigenvector $\tilde{l}_0$ (gray solid line), which represents the stationary density at the target potential $\tilde{V}(q)$, and the eigenvector l_1 (blue dotted line), which represents the stationary density at the biased potential. Due to the bias potential, the two stationary densities differ. Reweighting the simulation data from the biased simulation recovers the stationary density $\tilde{l}_0$ at the target potential $\tilde{V}(q)$ with high accuracy. Figure 5j shows the eigenvectors $\tilde{l}_1$ and l_1, which represent the equilibration between the two bound states. Interestingly, the process seems to be insensitive to the bias, and thus the results from all three MSMs coincide. Figure 5h shows the associated implied time scales from simulations at the target potential $\tilde{V}(q)$, simulations at the biased potential $V(q)$, and simulations at the biased potential $V(q)$ reweighted to the target potential $\tilde{V}(q)$. While the MSM eigenvectors can be reweighted with high accuracy, the reweighted implied time scale exhibits a large statistical uncertainty. We will discuss possible reasons for this in Conclusions.

COMPUTATIONAL DETAILS

Müller–Brown Potential. The Müller–Brown potential[56] is

$$V_{\mathrm{MB}}(x, y) = \sum_{n=1}^{4} A_n \exp(a_n(x - x_n)^2 + b_n(x - x_n)(y - y_n) + c_n(y - y_n)^2) \tag{31}$$

its parameters are reported in Table 2.

We used two different bias potentials along the x-coordinate

$$\begin{aligned} U_{\mathrm{bias,linear}}(x) &= k_{\mathrm{linear}} \cdot x \\ U_{\mathrm{bias,polynom}}(x) &= k_{\mathrm{polynom}} \cdot \sum_{n=1}^{m} a_n x_n^n \end{aligned} \tag{32}$$

with k_{linear} = 1 kJ/mol/nm and k_{polynom} = 50 kJ/mol/nm, m = 13, $a_1 = 2.06e-02\ \mathrm{nm}^{-1}$, $a_2 = -2.32e-02\ \mathrm{nm}^{-2}$, $a_3 = 3.83e-03\ \mathrm{nm}^{-3}$, $a_4 = 3.92e-02\ \mathrm{nm}^{-3}$, $a_5 = -1.39e-02\ \mathrm{nm}^{-3}$, $a_6 = -3.39e-02\ \mathrm{nm}^{-3}$, $a_7 = 3.82e-04\ \mathrm{nm}^{-3}$, $a_8 = 1.24e-02\ \mathrm{nm}^{-3}$, $a_9 = 3.37e-03\ \mathrm{nm}^{-3}$, $a_{10} = -1.18e-03\ \mathrm{nm}^{-3}$, $a_{11} = -7.50e-04\ \mathrm{nm}^{-3}$, $a_{12} = -1.38e-04\ \mathrm{nm}^{-3}$, $a_{13} = -8.81e-06\ \mathrm{nm}^{-3}$. The particle mass was set to m_p = 1 amu. The ideal gas constant was set to R = 8.314 J/(K mol). The system was simulated using the Langevin splitting algorithm ABOBA[35] with time step Δt = 0.5 fs, friction coefficient ξ = 5 ps^{-1}, and temperature T = 300 K.

We used two different implementations of ABOBA: (i) a reference stand-alone implementation in Python and (ii) an abstraction using the OpenMM CustomIntegrator class. Both implementations are available via GitHub.[57,58]

In the simulations using the reference implementation, the initial positions were drawn randomly from a uniform distribution and the initial velocities were set to v_0 = (0.0, 0.0). We then generated 5 trajectories with 2.5×10^8 time steps each for $V_{\mathrm{MB}}(x, y)$ and 5 trajectories with 2.5×10^8 time steps each for $V_{\mathrm{MB}}(x, y) + U_{\mathrm{bias,linear}}(x)$.

In the simulations using the OpenMM CustomIntegrator class, the initial positions were drawn randomly from a uniform distribution, and the initial velocities were chosen according to the Boltzmann distribution at 300 K. We then generated 5 trajectories with 10^8 time steps each for $V_{\mathrm{MB}}(x, y)$, 5 trajectories with 10^8 time steps each for $V_{\mathrm{MB}}(x, y) + U_{\mathrm{bias,linear}}(x)$.

From the trajectories, we constructed two-dimensional MSMs on a grid with 36 states, where the x-dimension was discretized in the range $-3.5 \le x \le 1.5$, and the y-dimension was discretized in the range $-1.5 \le y \le 3.5$. The lag time was varied between 0.5 and 24.5 ps in steps of 1 ps. We calculated un-reweighted MSMs for the trajectories at $V_{\mathrm{MB}}(x, y)$ and $V_{\mathrm{MB}}(x, y) + U_{\mathrm{bias,linear}}(x)$ using the estimators provided by Deeptime.[47] We calculated reweighted MSMs for the trajectories at $V_{\mathrm{MB}}(x, y) + U_{\mathrm{bias,linear}}(x)$ using our implementation of the reweighted estimators which extends Deeptime.[57]

I^-–Ca^{2+}–I^-. The model system is a $Ca^{2+}I_2^-$ ion pair (m_{Ca} = 40.08 amu, m_{I} = 126.90 amu) in an explicit TIP3P water box, containing 49 hydrogen bonds restrained water molecules. The nonbonded interaction parameters are chosen according to the AMBER99 force field,[59] $\sigma_{\mathrm{Ca}} = 3.05 \times 10^{-1}$, $\varepsilon_{\mathrm{Ca}}$ = 1.92, $\sigma_{\mathrm{I}} = 4.19 \times 10^{-1}$, and ε_{I} = 1.67.

ABOBA simulations were performed either with openmmtools LangevinIntegrator[40] or for biased runs with LangevinSplittingIntegrator.[58] The initial positions were drawn randomly from a uniform distribution, and the initial velocities were chosen according to the Boltzmann distribution at 300 K. The step size of the integrator was 1 fs, with the friction coefficient of 2 ps^{-1}. The particle mesh Ewald summation was used to calculate the interaction of all particle pairs within a cutoff of 0.49 nm.

Five trajectories with 2.5×10^8 time steps each were created for the reference simulations at the unbiased potential. A total of 5 trajectories with 2×10^8 time steps each were created at the biased potential.

To apply the bias linearly along one Cartesian coordinate of the Ca^{2+} ion

$$U_{\mathrm{bias,linear}}(z) = k_{\mathrm{linear}} \cdot z \tag{33}$$

with k_{linear} = 5 kJ/mol/nm, the degrees of freedom of the three atoms are restricted in the other two dimensions by a harmonic potential of the strength $k_x = k_y = 200 \times 10^3$ kJ/mol/nm. The displacement of the two iodide atoms is restricted in all directions, $k_z = 500 \times 10^3$ kJ/mol/nm. Restraining the degrees of freedom of the three atoms and the reduction to the collective variable of the iodide-calcium distance $d_{\mathrm{I-Ca}}$ means that the slowest process takes place on a double-well potential.

The trajectories were used to construct one-dimensional MSMs on a grid with 50 states of the iodide-calcium distance $d_{\mathrm{I-Ca}}$ in the range $0.3 \le d_{\mathrm{I-Ca}} \le 0.6$. The lag time was varied between 5 and 100 ps in steps of 5 ps. The unreweighted MSMs for both the reference and biased trajectories are calculated using the estimators provided by Deeptime,[47] cf. Figure S4. The reweighted MSMs based on trajectories at a biased potential are performed with our implementation of the reweighted estimator.[57]

CONCLUSIONS

This paper presents a guide for implementing Girsanov reweighting in MD simulation and analysis programs using the OpenMM with openmmtools and Deeptime as examples.

https://doi.org/10.1021/acs.jpcb.4c01702
J. Phys. Chem. B 2024, 128, 6014–6027

In openmmtools, we extended an existing Langevin integrator class such that the reweighting factors are calculated on-the-fly and, at regular intervals, are written to a reweighting factor trajectory file. In Deeptime, we extended the MSM estimator class such that the transition counts are reweighted according to the reweighting factor trajectory. We demonstrated the correct functioning and error-free applicability of the newly implemented functions and classes using both a low-dimensional test system and a molecular system. The implementation can be readily used for larger systems. The extended software as well as instructions on how to use it are freely available.[57,58,60,61]

The computational cost of the Girsanov reweighting is usually small. During the simulation, it consists of recording the bias forces and energies and the random numbers for the degrees of freedom that are affected by the bias. During the analysis, it consists of evaluating eqs 29 or 30. Since the bias is typically low-dimensional, i.e., it affects only a few atoms in the simulation box, $\Delta\eta_{l,k'n_{\text{interval}}+k''}$ is zero for most dimensions l, and the evaluation of the sums in eqs 29 and 30 incurs no relevant computational cost. In our example of calcium−iodine ion dissociation in water, the bias was applied along the z-dimension of the Ca^{2+} cation. Thus, the bias force was one-dimensional and the sum over the dimensions in eq 29 contained a single term. This is negligible compared to the cost of evaluating a single MD simulation step with a 450-dimensional force vector (49 water molecules and 3 ions in the simulation box).

A more critical question is under which circumstances Girsanov reweighting is efficient. One condition for efficiency is that the length of the reweighted paths has to be considerably shorter than the slow processes in the system. MSMs are one way to construct Girsanov-reweighted kinetic models from short paths because in these models the path length is equal to the MSM lag time.[18,19,36,62] This is the approach we used here. Alternatively, one can reweight transition rates within the framework of transition path sampling or transition interface sampling, in which case only the typically short transition paths are reweighted.[33] The second factor that influences the efficiency of Girsanov reweighting is the bias. Finding an optimal bias is closely connected to optimal control theory and to finding an optimal reaction coordinate along which the bias is applied.[29,63]

When reweighting MSMs, the reweighted dominant eigenvectors tend to be more accurate than the reweighted implied time scales.[18,36] This might be due to several reasons. First, the relative path probability decreases exponentially with an increasing path length, falling below the numerical floating point accuracy. One simple remedy is to use a library which allows for a higher precision in the floating point numbers when analyzing the reweighting factor trajectories (for generating the path reweighting trajectory, the usual double precision should be sufficient). Second, in high-dimensional systems, the bias might shift the transition path ensemble for transitions across the barriers. A shifted transition path ensemble at the biased potential is then not representative of paths at the unbiased potential, which causes the relative path probability to be analytically close to zero. To avoid a shift in the transition paths, the bias can be deposited exclusively in the minima of the potential energy function.[22] Finally, drifting implied time scales with large statistical uncertainties also occur in discrete Markov models which are estimated from unbiased simulations. This effect might be magnified by reweighting. More accurate Markov models are obtained by using an arbitrary ansatz function instead of a crisp discretization, such as tICA-Markov models,[64,65] variational Markov models,[66] or core-set Markov models.[45,62,67] Since it is straightforward to reweight the estimators for these Markov models by the Girsanov relative path probability density, using suitable ansatz functions instead of crisp states will likely improve the accuracy of the reweighted implied time scales.

OpenMM provides very clean and clear access to the inner loop of the MD simulation via the CustomIntegrator class, and in previous studies, we have taken advantage of this class to implement Girsanov reweighing.[18,19,36] However, this approach requires a detailed understanding of both the numerical schemes used to implement Langevin splitting integrators and the equations for the relative path probability. With the LangevinSplittingIntegrator class presented here, the running of a Girsanov reweighting simulation is simplified to one line of code in the OpenMM simulation script. A simple and error-resistant way to set up a Girsanov reweighting simulation is the starting point for more reweighting studies on large molecular systems.

■ ASSOCIATED CONTENT

Data Availability Statement

We provide the software for MD simulation with Girsanov reweighting as a pull request to the original openmmtools and openmm repository. added class `LangevinSplittingGirsanov`: https://github.com/choderalab/openmmtools/pull/729, added class `ReweightingReporter`: https://github.com/openmm/openmm/pull/4533. The extensions of the deeptime[47] package for Girsanov reweighted Markov state models are also available as a pull request. modified class `count_matrix_coo2_mult` and added `GirsanovReweightingEstimator`: https://github.com/deeptime-ml/deeptime/pull/290. All simulation scripts and input files can be viewed in our reweightingtools repository. https://github.com/bkellerlab/reweightingtools.

Supporting Information

The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acs.jpcb.4c01702.

Illustration of simulation software setup with tools for Girsanov path reweighting; representation of difference in x-dimension of 100-steps long trajectories; difference between the OpenMM CustomIntegrator class and LangevinSplittingGirsanov for 100 time steps of simulation; and dynamical properties of the I^--Ca^{2+}-I^- system (PDF)

■ AUTHOR INFORMATION

Corresponding Author

Bettina G. Keller − *Department of Biology, Chemistry, and Pharmacy, Freie Universität Berlin, Berlin 14195, Germany;* orcid.org/0000-0002-7051-0888; Email: bettina.keller@fu-berlin.de

Author

Joana-Lysiane Schäfer − *Department of Biology, Chemistry, and Pharmacy, Freie Universität Berlin, Berlin 14195, Germany*

Complete contact information is available at: https://pubs.acs.org/10.1021/acs.jpcb.4c01702

https://doi.org/10.1021/acs.jpcb.4c01702
J. Phys. Chem. B 2024, 128, 6014−6027

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

The authors thank Sascha Jähnigen for helpful comments on the manuscript. The work was funded by the German Research Foundation (DFG, GRK2473 "Bioactive Peptides"—project number 392923329 and GRK2662 "Charging into the future"—project number 434130070).

■ REFERENCES

(1) Hénin, J.; Lelièvre, T.; Shirts, M. R.; Valsson, O.; Delemotte, L. Enhanced sampling methods for molecular dynamics simulations. *Living J. Comp. Mol. Sci.* **2022**, *4* (1), 1583.
(2) Swendsen, R. H.; Wang, J.-S. Replica monte carlo simulation of spin-glasses. *Phys. Rev. Lett.* **1986**, *57* (21), 2607−2609.
(3) Sugita, Y.; Okamoto, Y. Replica-exchange molecular dynamics method for protein folding. *Chem. Phys. Lett.* **1999**, *314* (1−2), 141−151.
(4) Torrie, G. M.; Valleau, J. P. Nonphysical sampling distributions in monte carlo free-energy estimation: Umbrella sampling. *J. Comput. Phys.* **1977**, *23* (2), 187−199.
(5) Wang, L.; Friesner, R. A.; Berne, B. Replica exchange with solute scaling: A more efficient version of replica exchange with solute tempering (rest2). *J. Phys. Chem. B* **2011**, *115* (30), 9431−9438.
(6) Huber, T.; Torda, A. E.; Van Gunsteren, W. F. Local elevation: A method for improving the searching properties of molecular dynamics simulation. *J. Comput. Aided Mol. Des.* **1994**, *8*, 695−708.
(7) Grubmüller, H. Predicting slow structural transitions in macromolecular systems: Conformational flooding. *Phys. Rev. E* **1995**, *52* (3), 2893−2906.
(8) Laio, A.; Parrinello, M. Escaping free-energy minima. *Proc. Natl. Acad. Sci. U.S.A.* **2002**, *99* (20), 12562−12566.
(9) Dellago, C.; Bolhuis, P. G. Transition path sampling and other advanced simulation techniques for rare events. *Advanced computer simulation approaches for soft matter sciences*; Springer 2009, pp 167−233..
(10) Zuckerman, D. M.; Chong, L. T. Weighted ensemble simulation: Review of methodology, applications, and software. *Annu. Rev. Biophys.* **2017**, *46*, 43−57.
(11) Kamenik, A. S.; Linker, S. M.; Riniker, S. Enhanced sampling without borders: On global biasing functions and how to reweight them. *Phys. Chem. Chem. Phys.* **2022**, *24* (3), 1225−1236.
(12) Kieninger, S.; Donati, L.; Keller, B. G. Dynamical reweighting methods for markov models. *Curr. Opin. Struct. Biol.* **2020**, *61*, 124−131.
(13) Zuckerman, D. M.; Woolf, T. B. Efficient dynamic importance sampling of rare events in one dimension. *Phys. Rev. E* **2000**, *63* (1), 016702.
(14) Athenes, M. A path-sampling scheme for computing thermodynamic properties of a many-body system in a generalized ensemble. *Eur. Phys. J. B* **2004**, *38*, 651−663.
(15) Adib, A. B. Stochastic actions for diffusive dynamics: Reweighting, sampling, and minimization. *J. Phys. Chem. B* **2008**, *112* (19), 5910−5916.
(16) Xing, C.; Andricioaei, I. On the calculation of time correlation functions by potential scaling. *J. Chem. Phys.* **2006**, *124* (3), 034110.
(17) Schütte, C.; Nielsen, A.; Weber, M. Markov state models and molecular alchemy. *Mol. Phys.* **2015**, *113* (1), 69−78.
(18) Donati, L.; Hartmann, C.; Keller, B. G. Girsanov reweighting for path ensembles and markov state models. *J. Chem. Phys.* **2017**, *146* (24), 244112.
(19) Donati, L.; Keller, B. G. Girsanov reweighting for metadynamics simulations. *J. Chem. Phys.* **2018**, *149* (7), 072335.
(20) Donati, L.; Weber, M.; Keller, B. G. A review of girsanov reweighting and of square root approximation for building molecular markov state models. *J. Math. Phys.* **2022**, *63* (12), 123306.
(21) Wolf, S.; Stock, G. Targeted molecular dynamics calculations of free energy profiles using a nonequilibrium friction correction. *J. Chem. Theory Comput.* **2018**, *14* (12), 6175−6182.
(22) Tiwary, P.; Parrinello, M. From metadynamics to dynamics. *Phys. Rev. Lett.* **2013**, *111* (23), 230602.
(23) Badaoui, M.; Kells, A.; Molteni, C.; Dickson, C. J.; Hornak, V.; Rosta, E. Calculating kinetic rates and membrane permeability from biased simulations. *J. Phys. Chem. B* **2018**, *122* (49), 11571−11578.
(24) Stelzl, L. S.; Kells, A.; Rosta, E.; Hummer, G. Dynamic histogram analysis to determine free energies and rates from biased simulations. *J. Chem. Theory Comput.* **2017**, *13* (12), 6328−6342.
(25) Galama, M. M.; Wu, H.; Kramer, A.; Sadeghi, M.; Noé, F. Stochastic approximation to mbar and tram: Batchwise free energy estimation. *J. Chem. Theory Comput.* **2023**, *19* (3), 758−766.
(26) Machlup, S.; Onsager, L. Fluctuations and irreversible process. ii. systems with kinetic energy. *Phys. Rev.* **1953**, *91* (6), 1512−1515.
(27) Girsanov, I. V. On transforming a certain class of stochastic processes by absolutely continuous substitution of measures. *Theory Probab. Its Appl.* **1960**, *5* (3), 285−301.
(28) Øksendal, B.; Øksendal, B. *Stochastic differential equations*; Springer, 2003.
(29) Shmilovich, K.; Ferguson, A. L. Girsanov reweighting enhanced sampling technique (grest): On-the-fly data-driven discovery of and enhanced sampling in slow collective variables. *J. Phys. Chem. A* **2023**, *127* (15), 3497−3517.
(30) Wang, Y.; Fass, J.; Kaminow, B.; Herr, J. E.; Rufa, D.; Zhang, I.; Pulido, I.; Henry, M.; Bruce Macdonald, H. E.; Takaba, K.; et al. End-to-end differentiable construction of molecular mechanics force fields. *Chem. Sci.* **2022**, *13* (41), 12016−12033.
(31) Lu, C.; Wu, C.; Ghoreishi, D.; Chen, W.; Wang, L.; Damm, W.; Ross, G. A.; Dahlgren, M. K.; Russell, E.; Von Bargen, C. D.; et al. Opls4: Improving force field accuracy on challenging regimes of chemical space. *J. Chem. Theory Comput.* **2021**, *17* (7), 4291−4300.
(32) Vitalini, F.; Mey, A. S.; Noé, F.; Keller, B. G. Dynamic properties of force fields. *J. Chem. Phys.* **2015**, *142* (8), 084101.
(33) Bolhuis, P. G.; Brotzakis, Z. F.; Keller, B. G. Optimizing molecular potential models by imposing kinetic constraints with path reweighting. *J. Chem. Phys.* **2023**, *159* (7), 074102.
(34) Hazoglou, M. J.; Walther, V.; Dixit, P. D.; Dill, K. A. Communication: Maximum caliber is a general variational principle for nonequilibrium statistical mechanics. *J. Chem. Phys.* **2015**, *143* (5), 051104.
(35) Leimkuhler, B.; Matthews, C. Molecular dynamics. *Interdiscip. Appl. Math* **2015**, *39*, 443.
(36) Kieninger, S.; Keller, B. G. Path probability ratios for langevin dynamics—exact and approximate. *J. Chem. Phys.* **2021**, *154* (9), 094102.
(37) Kieninger, S.; Ghysbrecht, S.; Keller, B. G. Girsanov reweighting for simulations of underdamped langevin dynamics. theory. *arXiv* **2023**, arXiv:2303.14696.
(38) Eastman, P.; Pande, V. Openmm: A hardware-independent framework for molecular simulations. *Comput. Sci. Eng.* **2010**, *12* (4), 34−39.
(39) Fass, J.; Sivak, D. A.; Crooks, G. E.; Beauchamp, K. A.; Leimkuhler, B.; Chodera, J. D. Quantifying configuration-sampling error in langevin simulations of complex molecular systems. *Entropy* **2018**, *20* (5), 318.
(40) Choderalab/openmmtools: A batteries-included toolkit for the gpu-accelerated openmm molecular simulation engine. https://github.com/choderalab/openmmtools, 2014.
(41) Prinz, J.-H.; Wu, H.; Sarich, M.; Keller, B.; Senne, M.; Held, M.; Chodera, J. D.; Schütte, C.; Noé, F. Markov models of molecular kinetics: Generation and validation. *J. Chem. Phys.* **2011**, *134* (17), 174105.
(42) Schütte, C.; Fischer, A.; Huisinga, W.; Deuflhard, P. A direct approach to conformational dynamics based on hybrid monte carlo. *J. Comput. Phys.* **1999**, *151* (1), 146−168.

https://doi.org/10.1021/acs.jpcb.4c01702
J. Phys. Chem. B 2024, 128, 6014−6027

(43) Swope, W. C.; Pitera, J. W.; Suits, F. Describing protein folding kinetics by molecular dynamics simulations. 1. theory. *J. Phys. Chem. B* **2004**, *108* (21), 6571−6581.
(44) Chodera, J. D.; Singhal, N.; Pande, V. S.; Dill, K. A.; Swope, W. C. Automatic discovery of metastable states for the construction of markov models of macromolecular conformational dynamics. *J. Chem. Phys.* **2007**, *126* (15), 155101.
(45) Buchete, N.-V.; Hummer, G. Coarse master equations for peptide folding dynamics. *J. Phys. Chem. B* **2008**, *112* (19), 6057−6069.
(46) Bowman, G. R.; Pande, V. S.; Noé, F. *An introduction to Markov state models and their application to long timescale molecular simulation*; Springer Science & Business Media, 2013; Vol. *797*.
(47) Hoffmann, M.; Scherer, M.; Hempel, T.; Mardt, A.; de Silva, B.; Husic, B. E.; Klus, S.; Wu, H.; Kutz, N.; Brunton, S. L.; et al. Deeptime: A python library for machine learning dynamical models from time series data. *Mach. Learn.: Sci. Technol.* **2021**, 3 (1), 015 009.
(48) Zwanzig, R. *Nonequilibrium statistical mechanics*; Oxford University Press, 2001; .
(49) Gardiner, C. W.; et al. *Handbook of stochastic methods*; Springer: Berlin, 1985; Vol. *3*.
(50) Zwanzig, R. W. Transition from quantum to" classical" partition function. *Phys. Rev.* **1957**, *106* (1), 13−15.
(51) Leimkuhler, B.; Matthews, C. Rational construction of stochastic numerical methods for molecular sampling. *Appl. Math. Res. Express* **2013**, *2013* (1), 34−56.
(52) Sivak, D. A.; Chodera, J. D.; Crooks, G. E. Using nonequilibrium fluctuation theorems to understand and correct errors in equilibrium and nonequilibrium simulations of discrete langevin dynamics. *Phys. Rev. X* **2013**, *3* (1), 011007.
(53) Leimkuhler, B.; Matthews, C. Robust and efficient configurational molecular sampling via langevin dynamics. *J. Chem. Phys.* **2013**, *138* (17), 174102.
(54) Wio, H. S. *Path integrals for stochastic processes: An introduction*; World Scientific, 2013.
(55) Galvelis, R.; Eastman, P. a., Openmm/openmm-plumed: Openmm plugin to interface with plumed, https://github.com/openmm/openmm-plumed, 2016.
(56) Müller, K.; Brown, L. D. Location of saddle points and minimum energy paths by a constrained simplex optimization procedure. *Theor. Chem. Acc.* **1979**, *53*, 75−93.
(57) Deeptime with girsanov path reweighting, https://github.com/deeptime-ml/deeptime/pull/290, accessed March 13, 2024.
(58) Openmmtools with girsanov path reweighting, https://github.com/choderalab/openmmtools/pull/729, accessed May 6, 2024.
(59) Ponder, J. W.; Case, D. A. Force fields for protein simulations. *Adv. Protein Chem.* **2003**, *66*, 27−85.
(60) Openmm with girsanov path reweighting, https://github.com/openmm/openmm/pull/4533, accessed May 6, 2024.
(61) Reweightingtools, https://github.com/bkellerlab/reweightingtools, accessed May 6, 2024.
(62) Schütte, C.; Noé, F.; Lu, J.; Sarich, M.; Vanden-Eijnden, E. Markov state models based on milestoning. *J. Chem. Phys.* **2011**, *134* (20), 204105.
(63) Das, A.; Rose, D. C.; Garrahan, J. P.; Limmer, D. T. Reinforcement learning of rare diffusive dynamics. *J. Chem. Phys.* **2021**, *155* (13), 134105.
(64) Schwantes, C. R.; Pande, V. S. Improvements in markov state model construction reveal many non-native interactions in the folding of ntl9. *J. Chem. Theory Comput.* **2013**, *9* (4), 2000−2009.
(65) Pérez-Hernández, G.; Paul, F.; Giorgino, T.; De Fabritiis, G.; Noé, F. Identification of slow molecular order parameters for markov model construction. *J. Chem. Phys.* **2013**, *139* (1), 015102.
(66) Nuske, F.; Keller, B. G.; Pérez-Hernández, G.; Mey, A. S.; Noé, F. Variational approach to molecular kinetics. *J. Chem. Theory Comput.* **2014**, *10* (4), 1739−1752.
(67) Lemke, O.; Keller, B. G. Density-based cluster algorithms for the identification of core sets. *J. Chem. Phys.* **2016**, *145* (16), 164104.

https://doi.org/10.1021/acs.jpcb.4c01702
J. Phys. Chem. B 2024, 128, 6014−6027