The Journal of Chemical Physics ARTICLE scitation.org/journal/jcp

Adaptive Brownian Dynamics

Cite as: J. Chem. Phys. 155, 134107 (2021); doi: 10.1063/5.0062396
Submitted: 5 July 2021 • Accepted: 19 August 2021 •
Published Online: 1 October 2021

Florian Sammüller[a] and Matthias Schmidt[b]

AFFILIATIONS

Theoretische Physik II, Physikalisches Institut, Universität Bayreuth, D-95447 Bayreuth, Germany

[a]Author to whom correspondence should be addressed: florian.sammueller@uni-bayreuth.de
[b]Electronic mail: Matthias.Schmidt@uni-bayreuth.de

ABSTRACT

A framework for performant Brownian Dynamics (BD) many-body simulations with adaptive timestepping is presented. Contrary to the Euler–Maruyama scheme in common non-adaptive BD, we employ an embedded Heun–Euler integrator for the propagation of the overdamped coupled Langevin equations of motion. This enables the derivation of a local error estimate and the formulation of criteria for the acceptance or rejection of trial steps and for the control of optimal stepsize. Introducing erroneous bias in the random forces is avoided by rejection sampling with memory due to Rackauckas and Nie, which makes use of the Brownian bridge theorem and guarantees the correct generation of a specified random process even when rejecting trial steps. For test cases of Lennard-Jones fluids in bulk and in confinement, it is shown that adaptive BD solves performance and stability issues of conventional BD, already outperforming the latter even in standard situations. We expect this novel computational approach to BD to be especially helpful in long-time simulations of complex systems, e.g., in non-equilibrium, where concurrent slow and fast processes occur.

I. INTRODUCTION

Computer simulations have long become an established tool in the investigation of physical phenomena.[1–3] Complementing experimental results, they build the foundation for the exploration of increasingly complex dynamical systems. From the standpoint of classical statistical mechanics, the simulation of a many-body system consisting of discrete interacting particles can reveal information about its structural correlation as well as its thermodynamic properties. Naturally, this opens up the possibility of tackling many problems in the fields of materials science, soft matter, and biophysics, such as investigating the dynamics of macromolecules,[4] predicting rheological properties of fluids,[5–7] or exploring non-equilibrium processes that occur, e.g., in colloidal suspensions under the influence of external forcing.[8]

With the ever-increasing capabilities of computer hardware, a variety of different computational methods have emerged since the middle of the last century. Conceptually, at least three distinct classes of particle-based simulation frameworks can be identified: (i) Monte-Carlo (MC), which relies on the stochastic exploration of phase space; (ii) Molecular Dynamics (MD), in which the set of ordinary differential equations (ODEs) of Hamiltonian dynamics is integrated to obtain particle trajectories; and (iii) Langevin Dynamics, where random processes are incorporated into the Newtonian equations of motion so that the evolution of a system is obtained by numerical integration of then stochastic differential equations (SDEs). Brownian Dynamics (BD) can be seen as a special case of (iii), since the underlying stochastic Langevin equation is thereby considered in the overdamped limit where particle inertia vanishes and only particle coordinates remain as the sole microscopic degrees of freedom.

Notably, a broad range of refined methods have been developed in all three categories, sometimes even intersecting those. Important examples of such extensions are kinetic Monte-Carlo for the approximation of time evolution from reaction rates,[9] the addition of thermostats in MD to model thermodynamic coupling,[10,11] event-driven algorithms that enable both MD and BD in hard-particle systems,[12,13] and the adaptation of molecular algorithms to modern hardware.[14] Improvements in the calculation of observables from the resulting particle configurations have been made as well, e.g., by modifying their generation in MC (umbrella sampling, transition matrix MC,[15] Wang-Landau sampling[16]) or by utilizing advanced evaluation schemes in MD and BD, such as force sampling[17–19] or adaptive resolution MD.[20]

The efficiency and accuracy of a certain algorithm are always primary concerns, as these properties are essential for applicability and practicability in real-world problems. One therefore aims to design procedures that are both as fast and as precise as possible—yet

it is no surprise that those two goals might often be conflicting. In particular, in BD, where stochastic processes complicate the numerical treatment, the development of more sophisticated algorithms apparently lacks behind that of MD, for example, and one often resorts to alternative or combined simulation techniques.[21,22] If the full dynamical description of BD is indeed considered, the equations of motion are usually integrated with the simple Euler–Maruyama method,[23] where stochasticity is accounted for in each equidistant step via normally distributed random numbers. This can lead to inaccuracies and stability problems, making BD seem inferior to other computational methods.

In this work, we propose a novel approach to BD simulations, which rectifies the above shortcomings of conventional BD. To achieve this, we employ an adaptive timestepping algorithm that enables the control of the numerical error as follows. The common Euler–Maruyama method is complemented with a higher-order Heun step to obtain an embedded integrator pair for an estimation of the local discretization error per trial step. By comparison of this error estimate with a prescribed tolerance, the trial step is either accepted or it is rejected and then retried with a smaller stepsize. Particular care is required after rejections so as to not introduce a bias in the random forces, which would violate their desired properties. We therefore use Rejection Sampling with Memory (RSwM)[24] to retain a Gaussian random process even in a scenario where already determined random increments may conditionally be discarded. RSwM is a recently developed algorithm for the adaptive generation of random processes in the numerical solution of SDEs, which we improve and specialize to our context of overdamped Brownian motion and thereby formulate a method for adaptive BD simulations.

We demonstrate the practical advantages of adaptive BD over common BD in simulation results for prototypical bulk equilibrium systems and for more involved cases in non-equilibrium. A notable example that we investigate is the drying of colloidal films at planar surfaces. In particular, when dealing with non-trivial mixtures, as, e.g., present in common paints and coatings, the dynamics of this process can be inherently complex and its quantitative description turns out to be a major challenge.[25,26] This stands in contrast to the necessity of understanding and predicting stratification processes in those systems. Stratification leads to a dried film that has multiple layers differing in the concentration of constituent particle species, thereby influencing macroscopic properties of the resulting colloidal film. Therefore, controlling this process is an important measure to tailor colloidal suspensions to their field of application. Advances in this area have been made experimentally,[27,28] by utilizing functional many-body frameworks like dynamical density functional theory (DDFT),[29] and with molecular simulations such as conventional BD.[30] By employing the adaptive BD method, we are able to capture the complex dynamical processes occurring in those systems even in the final dense state. Close particle collisions and jammed states are resolved with the required adjustment of the timestep, necessary for the stability and accuracy of the simulation run in those circumstances. This cannot be achieved easily with common BD.

This paper is structured as follows. In Sec. II, a brief and non-rigorous mathematical introduction to the numerical solution of SDEs is given. Particularly, we illustrate the prerequisites for adaptive and higher-order algorithms in the case of general SDEs and emphasize certain pitfalls. In Sec. III, these considerations are applied to the case of Brownian motion. We construct the embedded Heun–Euler integration scheme in Sec. III A and incorporate RSwM in Sec. III B, which yields the adaptive BD method. Observables can then be sampled from the resulting particle trajectories with the means illustrated in Sec. III C. In Sec. IV, simulation results of the above-mentioned Lennard-Jones systems are shown and the practical use of adaptive BD is confirmed. In Sec. V, we conclude with a summary of the shown concepts, propose possible improvements for the adaptation of timesteps, and present further ideas for use cases.

II. NUMERICS OF STOCHASTIC DIFFERENTIAL EQUATIONS

Brownian dynamics of a classical many-body system of N particles in d spatial dimensions with positions $\mathbf{r}^N(t) = (\mathbf{r}^{(1)}(t), \ldots, \mathbf{r}^{(N)}(t))$ at time t and temperature T is described by the overdamped Langevin equation. The trajectory of particle i satisfies

$$\dot{\mathbf{r}}^{(i)}(t) = \frac{1}{\gamma^{(i)}}\mathbf{F}^{(i)}(\mathbf{r}^N(t)) + \sqrt{\frac{2k_BT}{\gamma^{(i)}}}\mathbf{R}^{(i)}(t), \tag{1}$$

where $\mathbf{F}^{(i)}(\mathbf{r}^N(t))$ is the total force (composed of external and interparticle contributions) acting on particle i, $\gamma^{(i)}$ is the friction coefficient of particle i, and k_B is Boltzmann's constant; the dot denotes a time derivative. In Eq. (1), the right-hand side consists of a deterministic (first summand) and a random contribution (second summand). The random forces are modeled via multivariate Gaussian white noise processes $\mathbf{R}^{(i)}(t)$ that satisfy

$$\langle \mathbf{R}^{(i)}(t) \rangle = 0, \tag{2}$$

$$\langle \mathbf{R}^{(i)}(t)\mathbf{R}^{(j)}(t') \rangle = \mathbf{I}\delta_{ij}\delta(t-t'), \tag{3}$$

where $\langle\cdot\rangle$ denotes an average over realizations of the random process, $\mathbf{I}$ is the $d \times d$ unit matrix, δ_{ij} denotes the Kronecker delta, and $\delta(\cdot)$ is the Dirac delta function.

One can recognize that Eq. (1) has the typical form of an SDE,

$$dX(t) = f(X(t),t)dt + g(X(t),t)dW(t), \tag{4}$$

if the dependent random variable X is identified with the particle positions $\mathbf{r}^N$ and W is a Wiener process corresponding to the integral of the Gaussian processes $\mathbf{R}^N = (\mathbf{R}^{(1)}, \ldots, \mathbf{R}^{(N)})$. As we do not consider hydrodynamic interactions, the random forces in Eq. (1) are obtained by a mere scaling of $\mathbf{R}^{(i)}(t)$ with the constant prefactors $\sqrt{2k_BT/\gamma^{(i)}}$. Therefore, the noise in BD is additive, since $g(X(t),t) = const.$ in the sense of Eq. (4). This is a crucial property for the construction of a simple higher-order integrator below in Sec. III.

In computer simulations, particle trajectories are obtained from Eq. (1) by numerical integration. Contrary to the numerics of ODEs, where higher-order schemes and adaptivity are textbook material, the derivation of corresponding methods for SDEs poses several difficulties that we address below. Due to the complications, SDEs of type (4) are often integrated via the Euler–Maruyama method instead, which follows the notion of the explicit Euler scheme for ODEs. Thereby, the true solution of Eq. (4) with initial value

$X(0) = X_0$ is approximated in $t \in [0, T]$ by partitioning the time interval into n equidistant subintervals of length $\Delta t = T/n$. Then, for $0 \leq k < n$, a timestep is defined by

$$X_{k+1} = X_k + f(X_k, t_k)\Delta t + g(X_k, t_k)\Delta W_k \tag{5}$$

with Wiener increments ΔW_k. An Euler–Maruyama step is also incorporated in the adaptive BD method that we construct below, applying Eq. (5) to the overdamped Langevin equation (1).

Crucially, the random increments ΔW_k in each Euler–Maruyama step (5) have to be constructed from independent and identically distributed normal random variables with expectation value $\mathrm{E}(\Delta W_k) = 0$ and variance $\mathrm{Var}(\Delta W_k) = \Delta t$. In practice, this is realized by drawing a new random number R (or vector thereof) from a pseudo-random number generator obeying the normal distribution $\mathcal{N}(0, \Delta t) = \sqrt{\Delta t}\mathcal{N}(0, 1)$ in each step k and setting $\Delta W_k = R$ in Eq. (5). The process of obtaining such a scalar (or vectorial) random increment R will be denoted in the following by

$$R \sim \mathcal{N}(\mu, \eta), \tag{6}$$

where $\mathcal{N}(\mu, \eta)$ is a scalar (or multivariate) normal distribution with expectation value μ and variance η.

As in the case of ODEs, an important measure for the quality of an integration method is its order of convergence. However, what convergence exactly means in the case of SDEs must be carefully reconsidered due to their stochasticity. We refer the reader to the pertinent literature (see, e.g., Ref. 23) and only summarize the key concepts and main results in the following.

Since both the approximation X_k and the true solution $X(t_k)$ are random variables, one can define two distinct convergence criteria. For a certain method with discretization $\Delta t \to 0$, *weak convergence* captures the error of average values, whereas *strong convergence* assesses the error of following a specific realization of a random process. One can show that the Euler–Maruyama method has a strong convergence order of 0.5, i.e., when increasing or decreasing the stepsize Δt, the error of the numerical solution only scales with $\Delta t^{0.5}$. For general $g(X(t), t)$ in Eq. (4), the construction of schemes with higher strong order is complicated due to the occurrence of higher-order stochastic integrals. Practically, this means that the careful evaluation of additional random variables is necessary in each iteration step. These random variables then enable the approximation of the stochastic integrals. There exist schemes of Runge–Kutta type with strong orders up to 2 although only strong order 1 and 1.5 Runge–Kutta methods are mostly used due to practical concerns.[31,32]

In order to incorporate adaptivity, one needs a means of comparison of two integration schemes of different strong order to formulate a local error estimate for the proposed step. If the error that occurs in a specific timestep is too large (we define below precisely what we mean by that), the step is rejected and a retrial with a smaller value of Δt is performed. Otherwise, the step is accepted and, based on the previously calculated error, a new optimized stepsize is chosen for the next step, which hence can be larger than the previous one. This protocol makes an optimal and automatic control of the stepsize possible, meaning that Δt can be both reduced when the propagation of the SDE is difficult and increased when the error estimate allows us to do so. Similar to the case of ODEs, it is computationally advantageous to construct so-called embedded integration schemes, analogous to, e.g., Runge–Kutta–Fehlberg integrators,[33] which minimize the number of costly evaluations of the right-hand side of Eq. (4). Developments have been made in this direction, e.g., with embedded stochastic Runge–Kutta methods.[34]

There is still one caveat to consider when rejecting a step, in that one has to be careful to preserve the properties of the Wiener process. In the naive approach of simply redrawing new uncorrelated random increments, rejections would alter the random process implicitly. The reason lies in the introduction of an undesired bias. Since large random increments (generally causing larger errors) get rejected more often, the variance of the Wiener process would be systematically decreased, ultimately violating its desired properties. To avoid this effect, it must be guaranteed that once a random increment is chosen, it will not be discarded until the time interval it originally applied to has passed. Consequently, when rejecting a trial step and retrying the numerical propagation of the SDE with smaller time intervals, new random increments cannot be drawn independently for those substeps anymore. The new random increments must instead be created based on the rejected original timestep such that an unbiased Brownian path is still followed.

The formal framework to the above procedure is given by the so-called Brownian bridge theorem,[35] which interpolates a Wiener process between known values at two timepoints. If $W(0) = 0$ and $W(\Delta t) = R$ are given (e.g., due to the previous rejection of a timestep of length Δt where the random increment R has been drawn), then a new intermediate random value must be constructed by

$$W(q\Delta t) \sim \mathcal{N}(qR, (1-q)q\Delta t), \quad 0 < q < 1 \tag{7}$$

such that the statistical properties of the to-be-simulated Wiener process are left intact and a substep $q\Delta t$ can be tried. The value of q thereby sets the fraction of the original time interval to which the Wiener process shall be interpolated. Equation (7) extends naturally (i.e., component-wise) to the multivariate case and it hence enables the construction of interpolating random vectors in a straightforward manner.

With this idea in mind, several adaptive timestepping algorithms for SDEs have been designed.[36–42] Still, most of these approaches are quite restrictive in the choice of timesteps (e.g., only allowing discrete variations such as halving or doubling),[36,42] involve the deterministic or random part only separately into an *a priori* error estimation,[39,41,43] or store rejected timesteps in a costly manner, e.g., in the form of Brownian trees.[36] In particular, the above methods rely on precomputed Brownian paths and do not illustrate an ad hoc generation, which is desirable from a performance and memory consumption standpoint in a high-dimensional setting like BD.

In contrast, a very flexible and performant class of adaptive timestepping algorithms called Rejection Sampling with Memory (RSwM) has recently been developed by Rackauckas and Nie.[24] Their work provides the arguably optimal means for the adaptive numerical solution of SDEs while still being computationally efficient in the generation of random increments as well as in the handling of rejections. We therefore use RSwM in the construction of an adaptive algorithm and specialize the method to Brownian motion in the following.

III. APPLICATION TO BROWNIAN DYNAMICS

Based on the remarks of Sec. II, we next proceed to apply the general framework to the case of BD with the overdamped Langevin equation (1) forming the underlying SDE. An embedded integration scheme is constructed, which allows the derivation of an error estimate and an acceptance criterion in each step. Furthermore, the application of RSwM for handling rejected timesteps in BD is shown and discussed. We also illustrate how the calculation of observables from sampling of phase space functions has to be altered in a variable timestep scenario.

A. Embedded Heun–Euler method

Regarding the overdamped Langevin equation (1), a major simplification exists compared to the general remarks made in Sec. II. Due to the noise term being trivial, some higher-order schemes can be constructed by only evaluating the deterministic forces for different particle configurations. Crucially, no iterated stochastic integrals are needed, which would have to be approximated in general higher strong-order integrators by using additional random variables.[32] In the following, we apply a scheme similar to the one suggested by Lamba *et al.*[41] for general SDEs to Eq. (1) and term it *embedded Heun–Euler method* due to its resemblance to the corresponding ODE integrators. Two different approximations $\tilde{\mathbf{r}}_{k+1}^N$ and $\mathbf{r}_{k+1}^N$ are calculated in each trial step by

$$\tilde{\mathbf{r}}_{k+1}^{(i)} = \mathbf{r}_k^{(i)} + \frac{1}{\gamma^{(i)}}\mathbf{F}^{(i)}(\mathbf{r}_k^N)\Delta t_k + \sqrt{\frac{2k_BT}{\gamma^{(i)}}}\mathbf{R}_k^{(i)}, \tag{8}$$

$$\mathbf{r}_{k+1}^{(i)} = \mathbf{r}_k^{(i)} + \frac{1}{2\gamma^{(i)}}\left(\mathbf{F}^{(i)}(\mathbf{r}_k^N) + \mathbf{F}^{(i)}(\tilde{\mathbf{r}}_{k+1}^N)\right)\Delta t_k + \sqrt{\frac{2k_BT}{\gamma^{(i)}}}\mathbf{R}_k^{(i)}. \tag{9}$$

Equation (8) is the conventional Euler–Maruyama step and hence constitutes the application of Eq. (5) to the overdamped Langevin equation (1). Equation (9) resembles the second order Heun algorithm or midpoint scheme for ODEs[44] and has been formally derived for SDEs in the context of stochastic Runge–Kutta methods.[45] Since the deterministic forces $\mathbf{F}^N$ are evaluated at the initial particle configuration $\mathbf{r}_k^N$ in both Eqs. (8) and (9), we have constructed an embedded integration method. This is favorable regarding computational cost, since the numerical result of $\mathbf{F}^N(\mathbf{r}_k^N)$ is evaluated once in Eq. (8) and reused in Eq. (9). Only one additional computation of the deterministic forces at the intermediate particle configuration $\tilde{\mathbf{r}}_{k+1}^N$ is then needed in the Heun step (9).

In each trial step, the same realization of random vectors $\mathbf{R}_k^N$ must be used in both Eqs. (8) and (9). Recall that the random displacements have to obey the properties of multivariate Wiener increments to model the non-deterministic forces. In conventional BD with fixed stepsize $\Delta t_k = \Delta t = const.$, these random vectors can therefore be drawn independently via $\mathbf{R}_k^{(i)} \sim \mathcal{N}(0, \Delta t)$ for each particle i. If rejections of trial steps are possible, $\mathbf{R}_k^{(i)}$ must instead be constructed as described in Sec. III B.

When the embedded Heun–Euler step (8) and (9) is applied to BD, the improvement over the conventional method is twofold. First, the Heun step (9) can be used as a better propagation method as already analyzed by Fixman[46] and Iniesta and García de la Torre.[47] Several further higher-order schemes, mostly of Runge–Kutta type, have been used in BD simulations.[45,48] These methods often lead to increased accuracy and even efficiency due to bigger timesteps becoming achievable, which outweighs the increased computational cost per step. Since the prefactor of the random force is trivial (i.e., constant) in the overdamped Langevin equation (1), higher strong-order schemes are easily constructable. This situation stands in contrast to the more complicated SDEs of type (4) with general noise term $g(X(t), t)$.

Second, with two approximations of different order at hand, assessing their discrepancy allows us to obtain an estimate of the discretization error in each step, which is a fundamental prerequisite in the construction of adaptive timestepping algorithms. For this, we exploit the additive structure of the noise term in Eq. (1) again and recognize that such an error estimate can be obtained by only comparing the deterministic parts of Eqs. (8) and (9). Nevertheless, note that the random displacements are already contained in $\tilde{\mathbf{r}}_{k+1}^N$. This makes the deterministic part of Eq. (9) implicitly dependent on the realization of $\mathbf{R}_k^N$, which is opposed to Ref. 41 where an error estimate is defined without involving the random increments at all.

At a given step $k \to k+1$, one can construct the automatic choice of an appropriate Δt_k. For this, the error of a trial step with length Δt is evaluated.

We define a particle-wise error

$$E^{(i)} = \|\Delta\tilde{\mathbf{r}}^{(i)} - \Delta\mathbf{r}^{(i)}\| = \frac{\Delta t}{2\gamma^{(i)}}\|\mathbf{F}^{(i)}(\tilde{\mathbf{r}}_{k+1}^N) - \mathbf{F}^{(i)}(\mathbf{r}_k^N)\|, \tag{10}$$

with $\Delta\tilde{\mathbf{r}}^{(i)} = \tilde{\mathbf{r}}_{k+1}^{(i)} - \mathbf{r}_k^{(i)}$ and $\Delta\mathbf{r}^{(i)} = \mathbf{r}_{k+1}^{(i)} - \mathbf{r}_k^{(i)}$. For each particle i, this error is compared to a tolerance

$$\tau^{(i)} = \epsilon_{\text{abs}} + \epsilon_{\text{rel}}\|\Delta\mathbf{r}^{(i)}\| \tag{11}$$

consisting of an absolute and a relative part with user-defined coefficients ϵ_{abs} and ϵ_{rel}. Note that $\Delta\mathbf{r}^{(i)}$ is used in Eq. (11), since it captures the true particle displacement more accurately than $\Delta\tilde{\mathbf{r}}^{(i)}$ due to its higher order. Additionally, we stress that $\Delta\mathbf{r}^{(i)}$ decreases on average for shorter timesteps such that $\tau^{(i)}$ indirectly depends on the trial Δt, which limits the accumulation of errors after multiple small steps.

Then, a total error estimate

$$\mathscr{E} = \left\|\left(\frac{E^{(i)}}{\tau^{(i)}}\right)_{1\le i\le N}\right\| \tag{12}$$

can be calculated for the trial step. While the Euclidean norm is the canonical choice in Eq. (10), there is a certain freedom of choosing an appropriate norm $\|\cdot\|$ in Eq. (12). The two- or ∞-norm defined as

$$\left\|(x^{(i)})_{1\le i\le N}\right\|_2 = \sqrt{\frac{1}{N}\sum_{i=1}^{N}|x^{(i)}|^2}, \tag{13}$$

$$\left\|(x^{(i)})_{1\le i\le N}\right\|_\infty = \max_{1\le i\le N}|x^{(i)}|, \tag{14}$$

respectively, may both come up as natural and valid options (note that we normalize the standard two-norm by $\sqrt{N}$ to obtain an intensive quantity). However, in Eq. (12), where a reduction from

particle-wise errors to a global scalar error takes place, this has crucial implications to the kind of error that is being controlled. If the two-norm is used, then ϵ_{abs} and ϵ_{rel} set the *mean* absolute and relative tolerance for all particles. In practice for large particle number N, this can lead to substantial single-particle errors becoming lost in the global average. Therefore, it is advisable to use the ∞-norm for the reduction in Eq. (12) to be able to set a *maximum single-particle* absolute and relative tolerance, i.e., if $\mathscr{E} < 1$, $E^{(i)} < \tau^{(i)}$ for all $i = 1, \ldots, N$.

Following the design of adaptive ODE solvers and ignoring stochasticity, an expansion of an embedded pair of methods with orders p and $p-1$ shows that an error estimate of type (12) is of order p. Thus, a timestep of length $q\Delta t$ with

$$q = \mathscr{E}^{-\frac{1}{p}} \tag{15}$$

could have been chosen to marginally satisfy the tolerance requirement $\mathscr{E} < 1$.

Considering the recommendation of Ref. 24, which discusses the application of such a timestep scaling factor to embedded methods for SDEs, we set

$$q = \left(\frac{1}{\alpha\mathscr{E}}\right)^2. \tag{16}$$

Here, both a more conservative exponent is chosen than in Eq. (15) and also a safety factor $\alpha = 2$ is introduced, as we want to account for stochasticity and for the low order of our integrators, which results in a low order of the error estimate (12).

With the choice (16), one can distinguish two possible scenarios in each trial step:

- $q \geq 1$: Accept the trial step, i.e., set $\Delta t_k = \Delta t$ and advance the particle positions with Eq. (9), and then continue with $k + 1 \to k + 2$.
- $q < 1$: Reject the trial step and retry $k \to k + 1$ with a smaller timestep.

In both cases, the timestep is adapted afterward via $\Delta t \leftarrow q\Delta t$. Here and in the following, the notation $a \leftarrow b$ denotes an assignment of the value b to the variable a.

It is advisable to restrict the permissible range of values for q by defining lower and upper bounds $q_{\text{min}} \leq q \leq q_{\text{max}}$ such that the adaptation of Δt is done with

$$q \leftarrow \min(q_{\text{max}}, \max(q_{\text{min}}, q)). \tag{17}$$

While commonly chosen as $q_{\text{max}} \approx 10$ and $q_{\text{min}} \approx 0.2$ for ODE solvers, due to stochasticity and the possibility of drawing "difficult" random increments, q_{min} should certainly be decreased in the case of SDEs to avoid multiple re-rejections. Vice versa, a conservative choice of q_{max} prevents an overcorrection of the timestep in the case of "fortunate" random events. We set $q_{\text{max}} = 1.2$ and $q_{\text{min}} = 0.001$ in practical applications to achieve a rapid adaptation in the case of rejected trial steps and a careful approach to larger timesteps after accepted moves.

One can also impose limits for the range of values of Δt, such as a maximum bound $\Delta t_{\text{max}} \geq \Delta t$. Restriction by a minimum value $\Delta t_{\text{min}} \leq \Delta t$, however, could lead to a forced continuation of the simulation with an actual local error that lies above the user-defined tolerance. Thus, this is not recommended. In our test cases described below, we see no need to restrict the timestep as the adaptive algorithm does not show unstable behavior without such a restriction.

Most concepts of this section can be generalized in a straightforward manner to non-overdamped Langevin dynamics, where particle inertia is explicitly considered in the stochastic equations of motion. However, the embedded Heun–Euler method (8) and (9) might not be a suitable integration scheme in this case. We therefore outline the modifications that are necessary in the construction of an adaptive algorithm for general Langevin dynamics in Appendix A.

B. Rejection sampling with memory in BD

We still have to prescribe the generation of the random vectors $\mathbf{R}_k^N$ that appear in both Eqs. (8) and (9). For clarity, we consider a single trial step and denote the set of corresponding random increments with $\mathbf{R}$ (the sub- and superscript is dropped). Obviously, $\mathbf{R}$ can no longer be chosen independently in general but has to incorporate previously rejected timesteps as long as they are relevant, i.e., as long as their original time interval has not passed yet. For this, we apply the Rejection Sampling with Memory (RSwM) algorithm to BD. Rackauckas and Nie[24] described three variants of RSwM, which they refer to as RSwM1, RSwM2, and RSwM3 and which differ in generality of the possible timesteps and algorithmic complexity. We aim to reproduce RSwM3 because it enables optimal timestepping and still ensures correctness in special but rare cases such as re-rejections.

Common to all of the RSwM algorithms is storing parts of rejected random increments onto a stack S_f. We refer to elements of this stack as *future information*, since they have to be considered in the construction of future steps. This becomes relevant as soon as a step is rejected due to a too large error. Then, a retrial is performed by decreasing Δt and drawing bridging random vectors via Eq. (7). The difference between rejected and bridging random vectors must not be forgotten but rather be stored on the future information stack, cf. Fig. 1.

On the other hand, if a step is accepted, new random vectors have to be prepared for the next time interval Δt. If no future information is on the stack, this can be done conventionally via drawing Gaussian distributed random vectors according to $\mathbf{R} \sim \mathcal{N}(0, \Delta t)$. If the stack S_f is not empty and thus future information is available, then elements of the stack are popped one after another, i.e., they are taken successively from the top. The random vectors as well as the time interval of each popped element are accumulated in temporary variables, which then hold the sum of those respective time intervals and random vectors. The stack could be empty before the accumulated time reaches Δt, i.e., there could still be a difference Δt_{gap}. In this case, one draws new random vectors $\mathbf{R}_{\text{gap}} \sim \mathcal{N}(0, \Delta t_{\text{gap}})$ to compensate for the difference and adds them to the accumulated ones, cf. Fig. 2, before attempting the trial step. Otherwise, if the future information reaches further than Δt, then there is one element that passes Δt. One takes this element, splits it in "before Δt" and "after Δt," and draws bridging random vectors for "before Δt" according to Eq. (7), which are again added to the accumulated ones. The rest of this element ("after Δt") can be pushed back to the future information stack

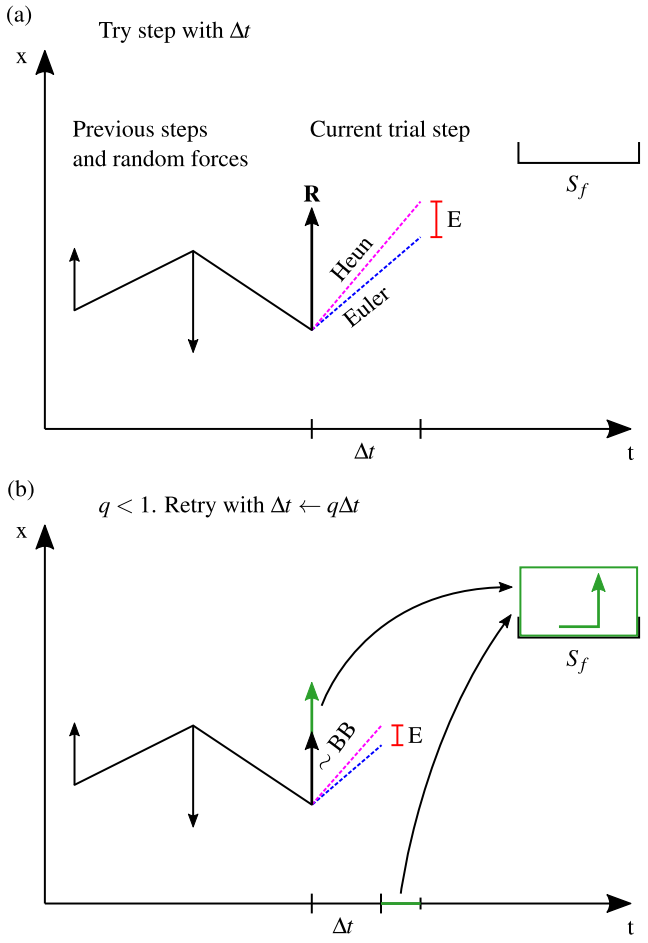

FIG. 1. The trial step is rejected (a) and a retrial with a smaller value of Δt is performed (b) if the discrepancy between Eqs. (8) and (9) is large. To preserve the properties of the Brownian motion, the Brownian bridge (BB) theorem (7) is used to interpolate the random process at the intermediate timepoint $q\Delta t$. The difference between the bridged random sample and the rejected original random sample is stored along with the remaining time difference onto the stack S_f. This is indicated in (b), where S_f now contains one element that holds the residual time interval (horizontal segment) and random increment (vertical arrow). Thus, in future steps, the Brownian path can be reconstructed from elements on S_f and the properties of the Wiener process remain intact. Note that for correctness in the case of re-rejections, one requires a second stack S_u as explained below and in Fig. 4.

and the step Δt can be tried with the accumulated vectors set as **R** in Eqs. (8) and (9), cf. Fig. 3.

At this stage, we have constructed RSwM2 for BD, which is not capable of handling all edge cases yet as pointed out in Ref. 24. If future information is popped from the stack to prepare **R** for the next step, and this next step is then rejected, we have lost all popped information unrecoverably. To circumvent this, one adds a second stack S_u that stores information that is currently used for the construction of **R**. We refer to elements of this stack as *information in use*. If a step is rejected, the information in use can be moved back to the future information stack so that no elements are lost in multiple retries, cf. Fig. 4. With this additional bookkeeping, correctness

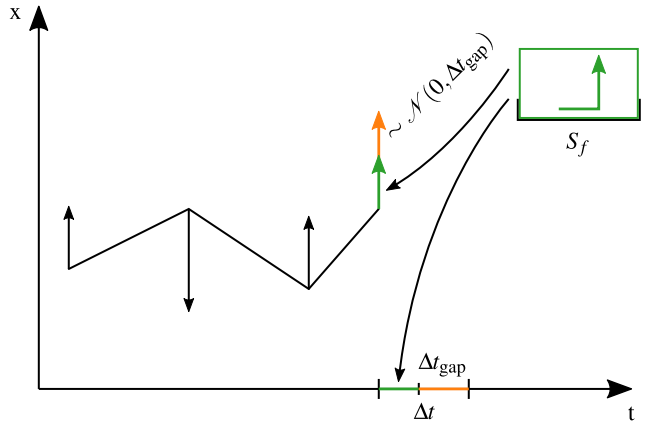

FIG. 2. After an accepted step has been performed, a new random increment is prepared for the next trial step of length Δt. If available, future information—stemming from previously rejected trial steps—has to be incorporated in the generation of new random vectors in order to retain the properties of Brownian motion. In the shown case, the future information stack contains one element, which is popped and accumulated to the new random increment and time interval. The stack is now empty and a difference Δt_{gap} to the goal timestep Δt remains. For this gap, a new uncorrelated random increment has to be generated from $\mathcal{N}(0, \Delta t_{\text{gap}})$ to complete the preparation of the next trial step.

and generality are ensured in all cases and the RSwM3 algorithm is complete.

Notably, with the structuring of information into stacks where only the top element is directly accessible ("last in first out"), the chronological time order is automatically kept intact so that one only has to store time intervals and no absolute timepoints. Furthermore, searching or sorting of elements is prevented entirely, which makes all operations $\mathcal{O}(1)$ and leads to efficient implementations.

We point out that the original RSwM3 rejection branch as given in Ref. 24 was not entirely correct and draw a comparison to our rectifications in Appendix B, which have been brought to attention[49] and have since been fixed in the reference implementation DifferentialEquations.jl.[50] Crucially, the correction not only applies to the case of BD but is rather relevant for the solution of general SDEs as well. A full pseudocode listing of one adaptive BD trial step utilizing RSwM3 is given in Algorithm 2 in Appendix C along with further explanation of technical details.[51]

C. Sampling of observables

Within BD, observables can be obtained from the sampling of configuration space functions. As an example, consider the one-body density profile $\rho(\mathbf{r})$, which is defined as the average of the density operator $\hat{\rho}(\mathbf{r}, \mathbf{r}^N) = \sum_{i=1}^{N} \delta(\mathbf{r} - \mathbf{r}_i)$.

In simulations of equilibrium or steady states, one can use a time average over a suitably long interval $[0, T]$ to measure such quantities, i.e., for a general operator $\hat{A}(X, \mathbf{r}^N)$,

$$A(X) \approx \frac{1}{T} \int_0^T dt \hat{A}(X, \mathbf{r}^N(t)). \tag{18}$$

Note that the remaining dependence on X can consist of arbitrary scalar or vectorial quantities or also be empty. For example,

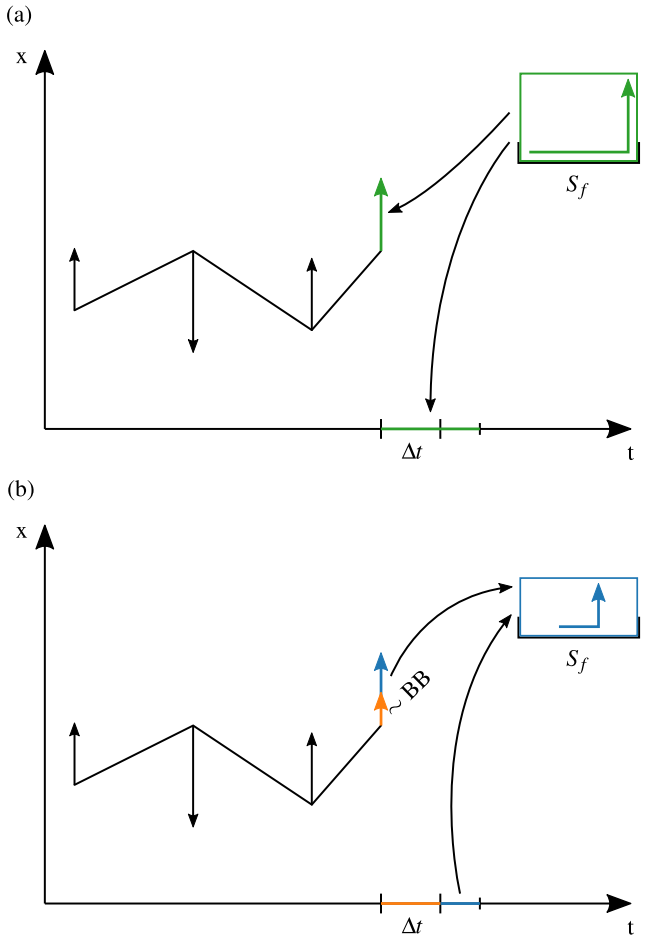

FIG. 3. The situation is similar to Fig. 2, where a step is rejected (a) and then retried (b). Here, the future information reaches further than the goal timestep Δt. In this case, the Brownian bridge (BB) has to be applied for the interpolation of an element from the future information stack. Generally, one pops the elements of the future stack S_f one after another and accumulates the random increments and time intervals until the element that crosses Δt is reached, to which the bridging theorem is then applied. In the shown example, the interpolation is done with the first element of S_f, since it already surpasses Δt. Similar to Fig. 1, the remainder of the bridged increment and time interval is pushed back onto S_f.

$X = (\mathbf{r}, \mathbf{r}')$ or $X = \mathbf{r}$ for general two- or one-body fields, $X = r$ for the isotropic radial distribution function, or $X = \emptyset$ for bulk quantities such as pressure or heat capacity.

Practically, $\hat{A}(X, \mathbf{r}^N)$ is evaluated in each step and an X-resolved histogram is accumulated, which yields $A(X)$ after normalization. Considering the numerical discretization of $[0, T]$ into n timesteps of constant length Δt within a conventional BD simulation, Eq. (18) is usually implemented as

$$A(X) \approx \frac{1}{n} \sum_{k=1}^{n} \hat{A}(X, \mathbf{r}_k^N). \tag{19}$$

In adaptive BD with varying timestep lengths, one cannot use Eq. (19) directly, since this would cause disproportionately many contributions to $A(X)$ from small timesteps and would thus lead

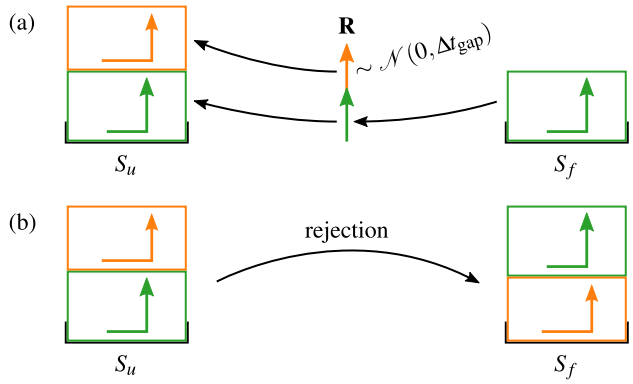

FIG. 4. To keep track of the random increments that are used in the construction of **R**, a second stack S_u is introduced. In (a), the situation of Fig. 2 is shown as an example, where **R** consists of an element of S_f (stemming from a previous rejection) and a Gaussian contribution $\mathbf{R}_{\text{gap}} \sim \mathcal{N}(0, \Delta t_{\text{gap}})$. In the case of rejection, the elements that were popped from S_f as well as newly drawn increments such as $\mathbf{R}_{\text{gap}}$ would be lost unrecoverably. By using S_u as an intermediate storage, all contributions to **R** can be transferred back to S_f such that no information about drawn random increments is lost and the same Brownian path is still followed. These considerations apply as well when random vectors are drawn via the Brownian bridge theorem, as, e.g., in Figs. 1 and 3.

to a biased evaluation of any observable. Formally, the quadrature in Eq. (18) now has to be evaluated numerically at non-equidistant integration points.

Therefore, if the time interval $[0, T]$ is discretized into n non-equidistant timepoints $0 = t_1 < t_2 < \cdots < t_n < t_{n+1} = T$ and $\Delta t_k = t_{k+1} - t_k$,

$$A(X) \approx \frac{1}{T} \sum_{k=1}^{n} \Delta t_k \hat{A}(X, \mathbf{r}_k^N) \tag{20}$$

constitutes a generalization of Eq. (19) for this case that enables a straightforward sampling of observables within adaptive BD.

An alternative technique can be employed in scenarios where the state of the system shall be sampled on a sparser time grid than the one given by the integration timesteps. Then, regular sampling points can be defined that must be hit by the timestepping procedure (e.g., by artificially shortening the preceding step). On this regular time grid, Eq. (19) can be used again. In particular, in non-equilibrium situations and for the measurement of time-dependent correlation functions such as the van Hove two-body correlation function,[52–57] this method might be beneficial since quantities can still be evaluated at certain timepoints rather than having to construct a time-resolved histogram consisting of finite time intervals. Note, however, that timestepping to predefined halting points is not yet considered in Algorithm 2.

IV. SIMULATION RESULTS

To test and illustrate the adaptive BD algorithm, we investigate the truncated and shifted Lennard-Jones fluid with interaction potential

$$\Phi_{\text{LJTS}}(r) = \begin{cases} \Phi_{\text{LJ}}(r) - \Phi_{\text{LJ}}(r_c), & r < r_c \\ 0, & r \geq r_c, \end{cases} \tag{21}$$

where

$$\Phi_{\mathrm{LJ}}(r) = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \tag{22}$$

and r is the interparticle distance. We set a cutoff radius of $r_c = 2.5\sigma$ throughout Secs. IV A and IV B and use reduced Lennard-Jones units that yield the reduced timescale $\tau = \sigma^2\gamma/\varepsilon$.

A common problem in conventional BD simulations is the choice of an appropriate timestep. Obviously, a too small value of Δt—while leading to accurate trajectories—has a strong performance impact, hindering runs that would reveal long-time behavior and prohibiting extensive sampling periods, which are desirable from the viewpoint of the time average (20). Still, Δt must be kept below a certain threshold above which results might be wrong or the simulation becomes unstable. Unfortunately, due to the absence of any intrinsic error estimates, judgment of a chosen Δt is generally not straightforward. For instance, one can merely observe the stability of a single simulation run and accept Δt if sensible output is produced and certain properties of the system (such as its energy) are well-behaved. Another possibility is the costly conduction of several simulation runs with differing timesteps, thereby cross-validating gathered results. Consequently, a true *a priori* choice of the timestep is not possible in general and test runs cannot always be avoided.

With adaptive timestepping, this problem is entirely prevented as one does not need to make a conscious choice for Δt at all. Instead, the maximum local error of a step is restricted by the user-defined tolerance (11), ensuring correctness of results up to a controllable discretization error. This does come at the moderate cost of overhead due to the additional operations per step necessary in the embedded Heun–Euler method and the RSwM algorithm. However, as we demonstrate in the following, the benefits of this method far outweigh the cost even in simple situations.

A. Lennard-Jones bulk fluid in equilibrium

In the following, we compare results from conventional BD to those obtained with adaptive BD. We first consider a bulk system of size $7 \times 7 \times 6\sigma^3$ with periodic boundary conditions at temperature $k_B T = 0.8\varepsilon$ consisting of $N = 100$ Lennard-Jones particles initialized on a simple cubic lattice. In the process of equilibration, a gaseous and a liquid phase emerge, and the system therefore becomes inhomogeneous.

With non-adaptive Euler–Maruyama BD, a timestep $\Delta t_{\mathrm{fix}} = 10^{-4}\tau$ is chosen to consistently converge to this state. This value is small enough to avoid severe problems that occur reproducibly for $\Delta t_{\mathrm{fix}} \gtrsim 5 \cdot 10^{-4}\tau$, where the simulation occasionally crashes or produces sudden jumps in energy due to faulty particle displacements.

In contrast, the timestepping of an adaptive BD simulation run is shown both as a timeseries and as a histogram in Fig. 5. The tolerance coefficients in Eq. (11) are thereby set to $\epsilon_{\mathrm{abs}} = 0.05\sigma$ and $\epsilon_{\mathrm{rel}} = 0.05$ and the ∞-norm is used in the reduction from particle-wise to global error (12). One can see that large stepsizes up to $\Delta t \approx 6 \cdot 10^{-4}\tau$ occur without the error exceeding the tolerance threshold. The majority of steps can be executed with a timestep larger than the value $\Delta t_{\mathrm{fix}} = 10^{-4}\tau$. On the other hand, the algorithm is able to detect moves that would cause large errors where it decreases Δt appropriately. It is striking that in the shown sample, minimum timesteps as small as $\Delta t = 3 \cdot 10^{-6}\tau$ occur. This is far

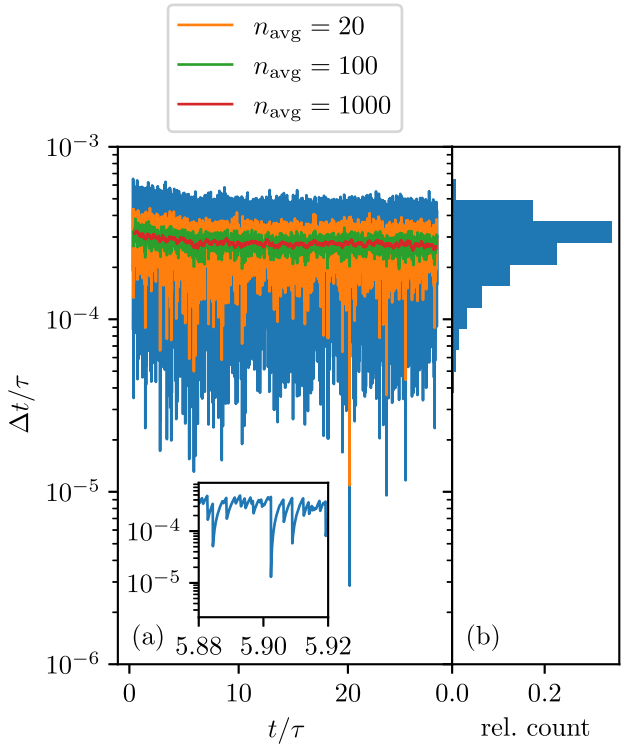

FIG. 5. The evolution of chosen timesteps for accepted moves is shown in (a). To accentuate the distribution of values of Δt further, moving averages taken over the surrounding n_{avg} points of a respective timestep record are depicted. One can see that the timestep Δt varies rapidly in a broad range between $\Delta t \approx 3 \cdot 10^{-6}\tau$ and $\Delta t \approx 6 \cdot 10^{-4}\tau$ around a mean value of $\Delta t \approx 2.8 \cdot 10^{-4}\tau$. In the inset of (a), a close-up of the timestep behavior is given, which reveals the rapid reduction of Δt at jammed states and the quick recovery afterward. In (b), the relative distribution of the data in (a) is illustrated. It is evident that the majority of steps can be executed with a large timestep, leading to increased performance of the BD simulation. On the other hand, in the rare event of a step that would produce large errors, the timestep is decreased appropriately to values far below those that would be chosen in a fixed-timestep BD run.

below the stepsize of $\Delta t_{\mathrm{fix}} = 10^{-4}\tau$ chosen in the fixed-timestep BD run above, which indicates that although the simulation is stable for this value, there are still steps that produce substantial local errors in the particle trajectories. For even larger values of Δt_{fix}, it is those unresolved collision events that cause unphysical particle displacements, which then cascade and crash the simulation run. In comparison, the adaptive BD run yields a mean timestep of $\overline{\Delta t} = 3 \cdot 10^{-4}\tau$, which is larger than the heuristically chosen fixed timestep.

1. Performance and overhead

Per step, due to an additional evaluation via the Heun method (9), twice the computational effort is needed to calculate the deterministic forces compared to a single evaluation of the Euler–Maruyama step (8). This procedure alone has the benefit of increased accuracy though, and it hence makes larger timesteps feasible.

The computational overhead due to adaptivity with RSwM3 comes mainly from storing random increments on both stacks and

applying this information in the construction of new random forces. Therefore, potential for further optimization lies in a cache-friendly organization of the stacks in memory as well as in circumventing superfluous access and copy instructions altogether. The latter considerations suggest that avoiding rejections, and more importantly, avoiding re-rejections, is crucial for good performance and reasonable memory consumption of the algorithm. As already noted, in practice, this can be accomplished with a small value of $q_{\min}$ that allows for a rapid reduction of Δt in the case of unfortunate random events and a conservative value of $q_{\max}$ to avoid too large timestepping after moves with fortunate random increments. In our implementation, the cost of RSwM routines is estimated to lie below 10% of the total runtime in common situations.

B. Non-equilibrium—Formation of colloidal films

While the benefits of adaptive BD are already significant in equilibrium, its real advantages over conventional BD become particularly clear in non-equilibrium situations. Due to the rich phenomenology—which still lacks a thorough understanding—and the important practical applications, the dynamics of colloidal suspensions near substrates and interfaces has been the center of attention in many recent works.[25–30,58] Nevertheless, the simulation of time-dependent interfacial processes is far from straightforward, and especially for common BD, stability issues are expected with increasing packing fraction.

In the following, we apply adaptive BD to systems of Lennard-Jones particles and simulate evaporation of the implicit solvent. This is done by introducing an external potential that models the fixed substrate surface as well as a moving air–solvent interface. As in Ref. 29, we set

$$V_{\text{ext}}^{(i)}(z,t) = B\left(\mathrm{e}^{-\kappa(z-\sigma^{(i)}/2)} + \mathrm{e}^{\kappa(z+\sigma^{(i)}/2-L(t))}\right) \tag{23}$$

to only vary in the z-direction and assume periodic boundary conditions in the remaining two directions. The value of κ controls the softness of the substrate and the air–solvent interface, while B sets their strength. We distinguish between the different particle sizes $\sigma^{(i)}$ to account for the offset of the particle centers where the external potential is evaluated. The position $L(t)$ of the air–solvent interface is time-dependent and follows a linear motion $L(t) = L_0 - vt$ with initial position L_0 and constant velocity v.

In the following, systems in a box that is elongated in the z-direction are considered. To ensure dominating non-equilibrium effects, values for the air–solvent interface velocities are chosen, which yield large Péclet numbers $\mathrm{Pe}^{(i)} = L_0 v/D^{(i)} \gg 1$, where $D^{(i)} = k_B T/\gamma^{(i)}$ is the Einstein–Smoluchowski diffusion coefficient and $\gamma^{(i)} \propto \sigma^{(i)}$ due to Stokes.

When attempting molecular simulations of such systems in a conventional approach, one is faced with a non-trivial choice of the timestep length Δt_{fix} since it has to be large enough to be efficient in the dilute phase but also small enough to capture the motion of the dense final state of system. A cumbersome solution would be a subdivision of the simulation into subsequent time intervals, thereby choosing timesteps that suit the density of the current state. This method is beset by problems as the density profile becomes inhomogeneous and is not known *a priori*.

We show that by employing the adaptive BD method of Sec. III, these issues become non-existent. Concerning the physical results of the simulation runs, the automatically chosen timestep is indeed closely connected to the increasing packing fraction as well as to the structural properties of the respective colloidal system, as will be discussed below. Similar test runs as the ones shown in the following but carried out with constant timestepping and a naive choice of Δt_{fix} frequently lead to instabilities in the high-density regimes, ultimately resulting in unphysical trajectories or crashes of the program. Due to the possibility of stable and accurate simulations of closely packed phases with adaptive BD, we focus on the investigation of the final conformation of the colloidal suspension.

1. Single species, moderate driving

First, a single species Lennard-Jones system is studied and the box size is chosen as $8 \times 8 \times 50\sigma^3$. The Lennard-Jones particles are initialized on a simple cubic lattice with lattice constant 2σ and the velocity of the air–solvent interface is set to $v = 1\sigma/\tau$. We set $\epsilon_{\text{abs}} = 0.01\sigma$ to accommodate for smaller particle displacements in the dense phase and relax the relative tolerance to $\epsilon_{\text{rel}} = 0.1$.

As one can see in Fig. 6, the timestep is automatically adjusted as a reaction to the increasing density. In the course of the simulation run, the average number density increases from $\sim 0.07\sigma^{-3}$ to $1.4\sigma^{-3}$ although the particles first accumulate near the air–solvent interface. Astonishingly, even the freezing transition at the end of the simulation run can be captured effortlessly. This illustrates the influence of collective order on the chosen timestep. With rising density of the Lennard-Jones fluid, the timestep decreases on average due to the shorter mean free path of the particles and more frequent collisions. At this stage, Δt varies significantly and very small timesteps maneuver the system safely through jammed states of the disordered fluid. In the process of crystallization, the timestep decreases rapidly to accommodate for the reduced free path of the particles before it shortly relaxes to a plateau when crystal order is achieved. Additionally, the variance of Δt decreases and, contrary to the behavior in the liquid phase, fewer jammed states can be observed. This is due to the crystal order of the solid phase, which prevents frequent close encounters of particles and hence alleviates the need for a rapid reduction of Δt.

2. Single species, strong driving

If the evaporation rate is increased via a faster moving air–solvent interface, the final structure of the colloidal suspension is altered. Particularly, with rising air–solvent velocity v, a perfect crystallization process is hindered and defects in the crystal structure occur. This is reflected in the timestepping evolution, as jammed states still happen frequently in the dense regime due to misaligned particles. Therefore, unlike in the previous case of no defects, sudden jumps to very small timesteps can still be observed as depicted in Fig. 7.

If the velocity of the air–solvent interface is increased even further, amorphous states can be reached, where no crystal order prevails. Nevertheless, our method is still able to resolve the particle trajectories of those quenched particle configurations so that the simulation remains both stable and accurate even in such demanding circumstances.

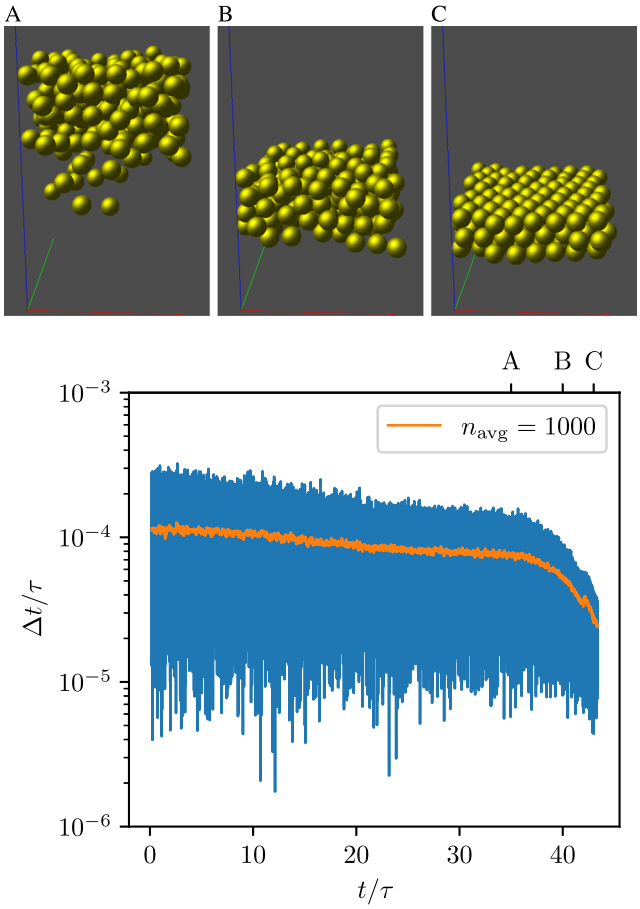

FIG. 6. The timestepping evolution in a system with moving interface modeled via Eq. (23) with parameters $B = 7000\varepsilon$, $\kappa = 1/\sigma$ (adapted from Ref. 29), and $v = 1\sigma/\tau$ is shown. We depict both the actual timeseries whose envelope visualizes the maximum and minimum values of Δt and a moving average over the surrounding n_{avg} points to uncover the mean chosen timestep. As the density increases and the propagation of the overdamped Langevin equation becomes more difficult, the timestep is systematically decreased. This is an automatic process that needs no user input and that can even handle the freezing transition which occurs at the end of the simulation. To illustrate this process, typical snapshots of the system are given at different timepoints, which are marked in the timestepping plot as A, B, and C and correspond to a dilute, a prefreezing, and a crystallized state. In particular, in the transition from B to C, frequent particle collisions occur, which are resolved carefully by the adaptive BD method.

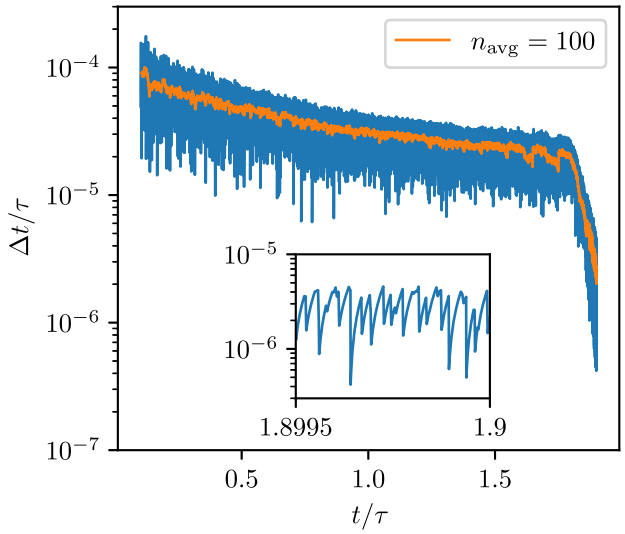

FIG. 7. Time evolution and moving average of the timestep Δt for the air–solvent interface velocity being increased to $v = 50\sigma/\tau$ in a larger system of box size $10 \times 10 \times 100\sigma^3$. In this case, defects are induced during the crystallization process, which prevent a perfect crystal order of the final particle configuration. Jammed states still occur frequently, and the timestep has to accommodate rapidly to resolve particle displacements correctly. This is depicted in the inset, which shows that sudden jumps to small values of Δt still occur in the high-density regime, indicating error-prone force evaluations due to prevailing defects in the crystal structure.

3. Binary mixture

Returning to stratification phenomena, we consider mixtures of particles differing in size. Depending on the Péclet numbers of the big (subscript b) and small (subscript s) particle species and their absolute value (i.e., if $\mathrm{Pe} \ll 1$ or $\mathrm{Pe} \gg 1$), different structures of the final phase can emerge, ranging from "small-on-top" or "big-on-top" layering to more complicated conformations.[29] For large Péclet numbers $1 \ll \mathrm{Pe}_s < \mathrm{Pe}_b$, studies of Fortini *et al.*[58] have shown the formation of a "small-on-top" structure, i.e., the accumulation of the small particle species near the moving interface. Additionally, in the immediate vicinity of the air–solvent boundary, a thin layer of big particles remains trapped due to their low mobility.

In the following, a binary mixture of Lennard-Jones particles with diameters $\sigma_b = \sigma$ and $\sigma_s = 0.5\sigma$ is simulated in a system of size $10 \times 10 \times 100\sigma^3$ and the velocity of the air–solvent interface is set to $v = 1\sigma/\tau$ as before. We initialize $N_b = 768$ big and $N_s = 4145$ small particles uniformly in the simulation domain and particularly focus in our analysis on the structure of the final dense phase.

As the simulation advances in time, the observations of Fortini *et al.*[58] can be verified. A thin layer of big particles at the air–solvent interface followed by a broad accumulation of small particles emerges. The timestepping evolution shows similar behavior to the single species case shown in Fig. 6. On average, the value of Δt decreases and throughout the simulation, jumps to low values occur repeatedly when interparticle collisions have to be resolved accurately.

As the system approaches the dense regime, the finalizing particle distribution of the dried colloidal suspension can be investigated. One can see that a thin layer of big particles develops in close proximity to the substrate, similar to the one forming throughout the simulation at the air–solvent interface. This process can again be explained by the lower mobility of the big particles compared to the small ones, which prevents a uniform diffusion away from the substrate.

Moreover, as the packing fraction increases further, the structure of the interfacial layers of the trapped big species changes. While only a single peak is visible at first, a second peak develops in the last stages of the evaporation process. This phenomenon occurs both at

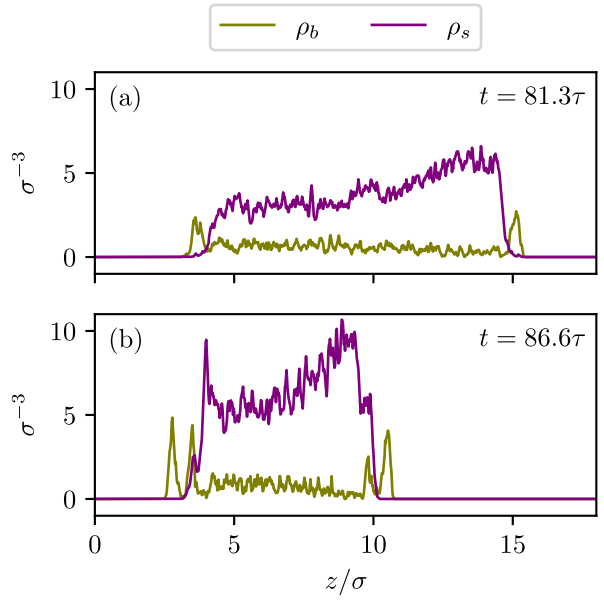

FIG. 8. The density profiles $\rho_b(z)$ and $\rho_s(z)$ of the big and small particle species in a stratifying colloidal suspension of a binary Lennard-Jones mixture with particle diameters $\sigma_b = \sigma$ and $\sigma_s = 0.5\sigma$ are shown at two timepoints of the simulation. In (a), single layers of the big species have already emerged near the substrate and the air–solvent interface, which enclose the dominating small particles in the middle of the box. At a later time (b) when the air–solvent interface has moved further toward the substrate and the packing fraction has hence increased, a second layer of the big particles forms at both interfaces and the final concentration gradient of the small species manifests within the dried film. Crucially, the intricate details of the final conformation demand an accurate numerical treatment of the dynamics of the closely packed colloidal suspension, to which adaptive BD offers a feasible solution.

the substrate and at the air–solvent interface although its appearance happens earlier and more pronounced at the former.

Even for this simple model mixture of colloidal particles that differ only in diameter, the final conformation after evaporation of the implicit solvent possesses an intricate structure. Both at the substrate and at the top of the film, primary and secondary layers of the big particle species build up. Those layers enclose a broad accumulation of the small particle species, which is by no means uniform but rather develops a concentration gradient in the positive z-direction, outlined by peaks of the respective density profile close to the big particle layers. The formation of the described final state is illustrated in Fig. 8, where the density profiles of both particle species are shown for two timepoints at the end of the simulation run.

V. CONCLUSION

In this work, we have constructed a novel method for BD simulations by employing recently developed algorithms for the adaptive numerical solution of SDEs to the case of Brownian motion as described on the level of the overdamped Langevin equation (1). For the evaluation of a local error estimate in each trial step, we have complemented the simple Euler–Maruyama scheme (8) found in common BD with a higher-order Heun step (9). By comparison of their discrepancy with a user-defined tolerance (11) composed of an absolute and a relative contribution, we were able to impose a criterion (16) for the acceptance or rejection of the trial step and for the adaptation of Δt. Special care was thereby required in the reduction from particle-wise errors (10) to a global scalar error estimate (12).

Due to the stochastic nature of Brownian motion, the rejection and subsequent retrial of a timestep could not be done naively by redrawing uncorrelated random vectors in this case. Instead, a sophisticated method had to be employed to ensure the validity of the statistical properties of the random process in Eq. (1) and hence to avoid biased random forces from a physical point of view. We have illustrated that the Brownian bridge theorem (7) resolves this problem formally and that RSwM[24] provides a feasible implementation method based on this theorem. Hence, we specialized RSwM to the case of Brownian motion in Sec. III B and constructed the fully adaptive BD scheme outlined in Algorithm 2. A correction to the original algorithm of Ref. 24 is given in Appendix B, and a generalization of adaptive BD to non-overdamped Langevin dynamics is outlined in Appendix A.

To test our framework, we applied the adaptive BD method to both equilibrium and non-equilibrium Lennard-Jones systems focusing on the analysis of individual trajectories. Even in the standard case of a phase-separating bulk fluid, we could verify that the use of RSwM induced no significant computational overhead and that a performance gain could be achieved compared to the fixed-timestep Euler–Maruyama method. This is complemented by practical convenience, since Δt needs not be chosen *a priori*.

The real advantages of adaptive BD become clear in more demanding situations where a fixed timestep would ultimately lead to inaccuracy and instability of the simulation without manual intervention. We have shown this with non-equilibrium systems of drying colloidal suspensions and have modeled the rapid evaporation of the implicit solvent by a moving interface as in Ref. 29. With our method, we achieved efficient timestepping and unconditional stability of the simulation even in the final stages where the packing fraction increases rapidly and the final structural configuration of the dried film develops. Particularly, in a single-species Lennard-Jones system, the freezing transition could be captured effortlessly even if crystal defects remained as a result of strong external driving. In the more elaborate case of a binary mixture of different-size particles, the intricate structure of the colloidal film could be resolved accurately. Here, we reported the development of a dual layer of the big particle species both at the substrate and at the top interface, while a concentration gradient of the small particle species could be observed in between. This shows that the method can be used to predict the arrangement of stratified films efficiently and with great detail, which could be helpful in the determination of their macroscopic properties.

We expect similar benefits in all sorts of situations where fundamental changes within the system call for more accurate or more relaxed numerical timestepping. For instance, this could be the case in sheared systems,[59,60] for sedimenting[61,62] or cluster-forming[63] colloidal suspensions, in microrheological simulations,[64] for phase transitions and especially the glass transition,[65] in the formation of crystals and quasicrystals,[66,67] in active matter,[68–71] possibly including active crystallization,[72,73] as well as for responsive colloids.[74] In any respect, adaptive BD can help in a more accurate measurement of observables, since the evaluation of the average (20) is not

hampered by erroneous particle configurations, which might occur in conventional BD. In particular, for quantities with a one-sided bias such as absolute or square values of particle forces and velocities, the abundance of outliers in the set of samples used in Eq. (20) is essential to yield valid results.

We finally illustrate possible improvements of the adaptive BD method itself, where several aspects come to mind. First, the choices of the parameters α in Eq. (16) as well as $q_{\min}$ and $q_{\max}$ in Eq. (17) were made mostly heuristically. We note that in Ref. 24, the influence of those parameters on the performance of the algorithm has been investigated but emphasize that results could vary for our high-dimensional setting of Brownian motion.

Second, recently developed adaptation methods for the stepsize that employ control theory could be helpful. Thereby, Δt is not merely scaled after each trial step by a locally determined factor q as evaluated in Eq. (16). Instead, one constructs a proportional-integral-derivative (PID) controller for the selection of new timesteps. This means that the adaptation is now not only influenced proportionally by the error estimate of the current step, but rather it is also determined by integral and differential contributions, i.e., the memory of previous stepsizes and the local change of the stepsize. Adaptive timestepping with control theory has already been applied to ODEs[75–77] and SDEs.[40] This method could help in our case to reduce the number of unnecessarily small steps even further. In our present implementation, a conservatively chosen $q_{\max}$ in Eq. (17) keeps the number of rejections low because it restricts the growth of Δt after moves with coincidentally low error. However, it also has the effect of preventing a fast relaxation of Δt after a sudden drop, e.g., due to an unfortunate random event. Then, many steps are needed for Δt to grow back to its nominal value because the maximum gain in each step is limited by $q_{\max}$. With a control theory approach, a mechanism could be established, which permits sudden drops to low values of Δt but still ensures a rapid relaxation afterward.

Finally, the adaptive BD method could be augmented to include hydrodynamic interactions. This requires a careful treatment of the random forces, as they now incorporate the particle configuration $\mathbf{r}^N$. Therefore, the noise is no longer additive, which complicates the numerical scheme necessary for a correct discretization of the overdamped Langevin equation. Still, instead of Eqs. (8) and (9), existing methods for SDEs with non-additive noise could be employed within the presented framework to yield an adaptive timestepping procedure for BD with hydrodynamic interactions.

In summary, to capture the dynamics of systems that undergo structural changes such as the ones shown above, and to gain further understanding of the implications regarding, e.g., the resulting macroscopic properties, sophisticated simulation methods are needed. Adaptive BD provides the means to treat these problems with great accuracy while still being computationally feasible and robust, which is a crucial trait from a practical standpoint.

ACKNOWLEDGMENTS

We thank Christopher Rackauckas, Tobias Eckert, and Daniel de las Heras for useful comments. This work was supported by the German Research Foundation (DFG) via Project No. 436 306 241.

APPENDIX A: GENERALIZATION TO LANGEVIN DYNAMICS

In the following, we transfer the concepts of Sec. III to general (non-overdamped) Langevin dynamics. Thereby, the momenta of particles with masses $m^{(i)}$, $i = 1, \ldots, N$, are explicitly considered in the Langevin equation

$$m^{(i)}\ddot{\mathbf{r}}^{(i)}(t) = \mathbf{F}^{(i)}(\mathbf{r}^N(t)) - \gamma^{(i)}\dot{\mathbf{r}}^{(i)}(t) + \sqrt{2\gamma^{(i)}k_BT}\mathbf{R}^{(i)}(t), \quad \text{(A1)}$$

where the notation is that of Eq. (1). Depending on the choice of $\gamma^{(i)}$, two special cases can be identified. BD is recovered in the diffusive regime for large $\gamma^{(i)}$. On the other hand, for vanishing friction and random forces, i.e., $\gamma^{(i)} = 0$, deterministic Hamiltonian equations of motion are obtained.

The SDE (A1) can be reformulated as a pair of first-order equations for the particle positions $\mathbf{r}^N$ and velocities $\mathbf{v}^N = \dot{\mathbf{r}}^N$,

$$m^{(i)}\dot{\mathbf{v}}^{(i)}(t) = \mathbf{F}^{(i)}(\mathbf{r}^N(t)) - \gamma^{(i)}\mathbf{v}^{(i)}(t) + \sqrt{2\gamma^{(i)}k_BT}\mathbf{R}^{(i)}(t), \quad \text{(A2a)}$$

$$\dot{\mathbf{r}}^{(i)}(t) = \mathbf{v}(t). \quad \text{(A2b)}$$

From a numerical perspective, the treatment of Eq. (A2) differs significantly from that of the overdamped Langevin equation (1), since the second-order nature makes more involved timestepping procedures for $\mathbf{r}^N$ and $\mathbf{v}^N$ possible.

This can be illustrated easily when Eqs. (A2a) and (A2b) are considered in the Hamiltonian case. Then, via the separation of $\mathbf{r}^N$ and $\mathbf{v}^N$, simple semi-implicit methods are readily available, such as the Euler-Cromer or the well-known velocity Verlet method that is commonly used in MD.[78,79] From a fundamental point of view, the use of symplectic algorithms is necessary for Hamiltonian systems to ensure the correct description of physical conservation laws and hence to achieve numerical stability for long-time behavior. While the above-mentioned integrators possess this property, most explicit algorithms like the forward Euler and classic Runge–Kutta method fail in this regard.

When going from Hamiltonian to Langevin dynamics, one is faced with an appropriate generalization of the familiar deterministic integration schemes to include dissipative and random forces. This is addressed by so-called quasi-symplectic methods, which are integrators for SDEs of type (A2) that degenerate to symplectic ones when both the noise and the friction term vanish.[80] Still, it is not straightforward to construct schemes that are suitable for arbitrary $\gamma^{(i)}$, and different strategies have been used in Langevin-based molecular simulations.[81–86]

To incorporate adaptive timestepping, one must again choose an integrator pair of different order to calculate an error estimation per step. The above considerations suggest, however, that at least for the higher-order method, a quasi-symplectic scheme is advisable for optimal efficiency and accuracy as it is expected to perform better than a naive application of the embedded Heun–Euler method (8) and (9) to general Langevin dynamics. Nevertheless, the latter could still be an adequate option when friction dominates.

Then, with the formalism described in Sec. III, the acceptance or rejection of trial steps and the use of RSwM is straightforward so that adaptive algorithms involving the Langevin equation (A1)

could be a feasible means for the simulation of dissipative particle dynamics in the future.

APPENDIX B: CORRECTNESS OF RSWM3 REJECTION BRANCH

Although the effect is likely subtle or even unmeasurable in most situations, in the original RSwM3 algorithm as described in Ref. 24 and formerly implemented in DifferentialEquations.jl,[50] the $q < 1$ branch was handled incorrectly. We list a pseudocode version thereof in Algorithm 1, where a specialization to our case of BD is already performed for ease of comparison with Algorithm 2.

To see why Algorithm 1 fails in some cases, consider multiple elements present on the stack S_u, e.g., as depicted in Fig. 9 where S_u contains three elements. Then, in lines 2–11 of Algorithm 1, elements are transferred successively from S_u back to S_f and their time intervals are accumulated in Δt_s as long as the remainder $\Delta t - \Delta t_s$ is still larger than the next goal timestep $q\Delta t$ (this is only the case for the green element in Fig. 9). After that, the Brownian bridge theorem is applied to $(\Delta t_K, \mathbf{R}_K) = (\Delta t - \Delta t_s, \mathbf{R} - \mathbf{R}_s)$. Therefore, the interpolation includes all the remaining elements of the stack S_u, i.e., the blue and violet one in Fig. 9, which implies that intermediate values of the Brownian path are not considered if S_u holds two or more elements at this point.

To yield a valid interpolation in this situation, the Brownian bridge must instead only be applied to the immediately following single element at the top of S_u (the blue one in Fig. 9). In Algorithm 2, lines 4–19 therefore replace the logic of Algorithm 1, which fixes the above issue by always considering only the single element that crosses $q\Delta t$ in the application of Eq. (7). This element is then interpolated at $q\Delta t$, for which the variable Δt_M is introduced to yield the corresponding fraction q_M of the element. Note that S_u always contains at least one element when entering the rejection branch (assuming a proper initialization of the simulation, cf. Appendix C).

ALGORITHM 1. Former $q < 1$ branch of RSwM3 where the bridge theorem might be applied to a wrong interval.

1: $\Delta t_s \leftarrow 0, \mathbf{R}_s \leftarrow 0$
2: **while** S_u not empty **do**
3: Pop top of S_u as $(\Delta t_u, \mathbf{R}_u)$
4: **if** $\Delta t_s + \Delta t_u < (1-q)\Delta t$ **then**
5: $\Delta t_s \leftarrow \Delta t_s + \Delta t_u, \mathbf{R}_s \leftarrow \mathbf{R}_s + \mathbf{R}_u$
6: Push $(\Delta t_u, \mathbf{R}_u)$ onto S_f
7: **else**
8: Push $(\Delta t_u, \mathbf{R}_u)$ back onto S_u
9: **break**
10: **end if**
11: **end while**
12: $\Delta t_K \leftarrow \Delta t - \Delta t_s, \mathbf{R}_K \leftarrow \mathbf{R} - \mathbf{R}_s$
13: $q_K \leftarrow q\Delta t/\Delta t_K$
14: $\mathbf{R}_{\text{bridge}} \sim \mathcal{N}(q_K \mathbf{R}_K, (1-q_K) q_K \Delta t_K)$
15: Push $((1-q_K)\Delta t_K, \mathbf{R}_K - \mathbf{R}_{\text{bridge}})$ onto S_f
16: $\Delta t \leftarrow q\Delta t, \mathbf{R} \leftarrow \mathbf{R}_{\text{bridge}}$

ALGORITHM 2. Embedded Heun–Euler trial step with RSwM3.

1: Calculate $\tilde{\mathbf{r}}_{k+1}$ and $\mathbf{r}_{k+1}$ via Eqs. (8) and (9)
2: Calculate q via Eqs. (10)–(12), (16), and (17)
3: **if** $q < 1$ **then** ▷ cf. Fig. 1, Appendix B
4: $\Delta t_s \leftarrow 0, \mathbf{R}_s \leftarrow 0$
5: **while** S_u not empty **do**
6: Pop top of S_u as $(\Delta t_u, \mathbf{R}_u)$
7: $\Delta t_s \leftarrow \Delta t_s + \Delta t_u, \mathbf{R}_s \leftarrow \mathbf{R}_s + \mathbf{R}_u$
8: **if** $\Delta t_s < (1-q)\Delta t$ **then**
9: Push $(\Delta t_u, \mathbf{R}_u)$ onto S_f
10: **else**
11: $\Delta t_M \leftarrow \Delta t_s - (1-q)\Delta t$
12: $q_M \leftarrow \Delta t_M / \Delta t_u$
13: $\mathbf{R}_{\text{bridge}} \sim \mathcal{N}(q_M \mathbf{R}_u, (1-q_M) q_M \Delta t_u)$
14: Push $((1-q_M)\Delta t_u, \mathbf{R}_u - \mathbf{R}_{\text{bridge}})$ onto S_f
15: Push $(q_M \Delta t_u, \mathbf{R}_{\text{bridge}})$ onto S_u
16: **break**
17: **end if**
18: **end while**
19: $\Delta t \leftarrow q\Delta t, \mathbf{R} \leftarrow \mathbf{R} - \mathbf{R}_s + \mathbf{R}_{\text{bridge}}$
20: **else**
21: Do step: $t \leftarrow t + \Delta t, \mathbf{r}_k \leftarrow \mathbf{r}_{k+1}, \Delta t \leftarrow q\Delta t$
22: Empty S_u, $\Delta t_s \leftarrow 0, \mathbf{R} \leftarrow 0$
23: **while** S_f not empty **do**
24: Pop top of S_f as $(\Delta t_f, \mathbf{R}_f)$
25: **if** $\Delta t_s + \Delta t_f < \Delta t$ **then**
26: $\Delta t_s \leftarrow \Delta t_s + \Delta t_f, \mathbf{R} \leftarrow \mathbf{R} + \mathbf{R}_f$
27: Push $(\Delta t_f, \mathbf{R}_f)$ onto S_u
28: **else** ▷ cf. Fig. 3
29: $q_M \leftarrow (\Delta t - \Delta t_s)/\Delta t_f$
30: $\mathbf{R}_{\text{bridge}} \sim \mathcal{N}(q_M \mathbf{R}_f, (1-q_M) q_M \Delta t_f)$
31: Push $((1-q_M)\Delta t_f, \mathbf{R}_f - \mathbf{R}_{\text{bridge}})$ onto S_f
32: Push $(q_M \Delta t_f, \mathbf{R}_{\text{bridge}})$ onto S_u
33: $\Delta t_s \leftarrow \Delta t_s + q_M \Delta t_f, \mathbf{R} \leftarrow \mathbf{R} + \mathbf{R}_{\text{bridge}}$
34: **break**
35: **end if**
36: **end while**
37: $\Delta t_{\text{gap}} \leftarrow \Delta t - \Delta t_s$
38: **if** $\Delta t_{\text{gap}} > 0$ **then** ▷ cf. Fig. 2
39: $\mathbf{R}_{\text{gap}} \sim \mathcal{N}(0, \Delta t_{\text{gap}})$
40: $\mathbf{R} \leftarrow \mathbf{R} + \mathbf{R}_{\text{gap}}$
41: Push $(\Delta t_{\text{gap}}, \mathbf{R}_{\text{gap}})$ onto S_u
42: **end if**
43: **end if**

APPENDIX C: IMPLEMENTATION OF EMBEDDED HEUN–EULER TRIAL STEP WITH RSWM3

The embedded Heun–Euler trial step with RSwM3 is the integral part of the adaptive BD method, for which we provide a pseudocode implementation in the following and explain its technical details as well as its setting in a full simulation framework. At the beginning of the simulation, S_f must be empty and a finite positive value of the first timestep Δt is chosen heuristically (this choice is irrelevant, however, since the algorithm immediately adapts Δt to acceptable values). Gaussian random increments are drawn for the

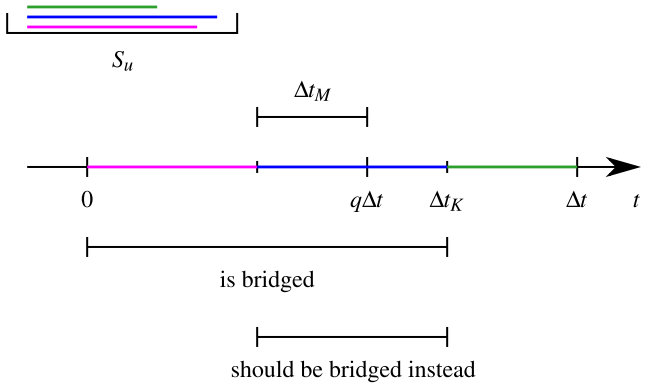

FIG. 9. A scenario is shown for which the original implementation of the RSwM3 rejection branch fails. For clarity, we only display the time intervals that reside on the stack S_u without their corresponding random vectors. In the presented case, the wrong interval ("is bridged") is selected in the original RSwM3 implementation to which the Brownian bridge theorem (7) is applied. This is fixed in Algorithm 2, where the correct element ("should be bridged") is considered via the calculation of Δt_M.

initial trial step via $\mathbf{R} \sim \mathcal{N}(0, \Delta t)$, which are pushed onto the stack S_u. This completes the initialization of the simulation run and a loop over trial steps can be started (until a certain simulation time or number of steps is reached).

Then, in each trial step, which is listed in Algorithm 2, the Euler and Heun approximations $\tilde{\mathbf{r}}_{k+1}$ and $\mathbf{r}_{k+1}$ are evaluated via Eqs. (8) and (9) and the adaptation factor q is calculated via Eqs. (16) and (17). This requires an error estimate (12), which is obtained from the two approximations via Eqs. (10) and (11).

If $q < 1$, the proposed step is rejected and a retrial must be performed. For a detailed description of the rejection branch (lines 4–19), we refer to Appendix B, where our changes to the original RSwM3 $q < 1$ case are illustrated as well. If a step is accepted, i.e., $q > 1$, the particle positions and physical time of the system are updated accordingly (line 21) before resetting random increments $\mathbf{R}$ and stack S_u (line 22). Then, in lines 23–42, new random increments are constructed for the next trial step.

For this, elements on S_f stemming from previously rejected steps have to be accounted for as long as parts of them lie within the goal timestep Δt. Therefore, the elements are transferred successively from S_f to S_u and their time intervals and random increments are accumulated in Δt_s and $\mathbf{R}$. If at some point Δt_s exceeds Δt, the corresponding element is interpolated at Δt via the Brownian bridge theorem (7) in lines 29–34 and one proceeds with the next trial step. Conversely, if all elements of S_f were popped and a gap Δt_{gap} remains between Δt_s and the goal timestep Δt, new Gaussian random increments $\mathbf{R}_{\text{gap}} \sim \mathcal{N}(0, \Delta t_{\text{gap}})$ are drawn, added to $\mathbf{R}$, and pushed onto S_u in lines 39–41 before attempting the next trial step. Note that the values of $\mathbf{R}$ persist across trial steps and are only reset after accepted moves.

Throughout the algorithm, bookkeeping of random increments and corresponding time intervals is established by pushing those elements that are involved in the current random increment $\mathbf{R}$ onto S_u and by storing elements that become relevant beyond the current step on S_f. This procedure is especially important after bridging an element at the goal timestep, where the two resulting parts are pushed onto S_u and S_f, respectively (lines 14, 15 and 31, 32). Consequently, no drawn random increments are ever lost.

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

[1] D. Frenkel and B. Smit, *Understanding Molecular Simulation: From Algorithms to Applications*, Computational Science Series No. 1, 2nd ed. (Academic Press, San Diego, 2002).
[2] J.-P. Hansen and I. R. McDonald, *Theory of Simple Liquids: With Applications of Soft Matter*, 4th ed. (Elsevier; Academic Press, Amstersdam, 2013).
[3] M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids*, reprinted edition (Oxford Science Publications, Clarendon Press, Oxford, 2009).
[4] M. Karplus and J. A. McCammon, Nat. Struct. Biol. **9**, 646 (2002).
[5] T. M. Squires and J. F. Brady, Phys. Fluids **17**, 073101 (2005).
[6] J. Reinhardt, A. Scacchi, and J. M. Brader, J. Chem. Phys. **140**, 144901 (2014).
[7] A. M. Puertas and T. Voigtmann, J. Phys.: Condens. Matter **26**, 243101 (2014).
[8] T. Geigenfeind, D. de las Heras, and M. Schmidt, Commun. Phys. **3**, 23 (2020).
[9] D. T. Gillespie, J. Comput. Phys. **22**, 403 (1976).
[10] H. C. Andersen, J. Chem. Phys. **72**, 2384 (1980).
[11] S. Nosé, Mol. Phys. **52**, 255 (1984).
[12] B. J. Alder and T. E. Wainwright, J. Chem. Phys. **31**, 459 (1959).
[13] A. Scala, T. Voigtmann, and C. De Michele, J. Chem. Phys. **126**, 134109 (2007).
[14] P. H. Colberg and F. Höfling, Comput. Phys. Commun. **182**, 1120 (2011).
[15] J.-S. Wang, T. K. Tay, and R. H. Swendsen, Phys. Rev. Lett. **82**, 476 (1999).
[16] D. P. Landau, S.-H. Tsai, and M. Exler, Am. J. Phys. **72**, 1294 (2004).
[17] D. de las Heras and M. Schmidt, Phys. Rev. Lett. **120**, 218001 (2018).
[18] B. Rotenberg, J. Chem. Phys. **153**, 150902 (2020).
[19] S. W. Coles, E. Mangaud, D. Frenkel, and B. Rotenberg, J. Chem. Phys. **154**, 191101 (2021).
[20] S. Fritsch, S. Poblete, C. Junghans, G. Ciccotti, L. Delle Site, and K. Kremer, Phys. Rev. Lett. **108**, 170602 (2012).
[21] P. J. Rossky, J. D. Doll, and H. L. Friedman, J. Chem. Phys. **69**, 4628 (1978).
[22] M. Jardat, O. Bernard, P. Turq, and G. R. Kneller, J. Chem. Phys. **110**, 7993 (1999).
[23] P. E. Kloeden and E. Platen, *Numerical Solution of Stochastic Differential Equations*, 1st ed. (Springer-Verlag Berlin Heidelberg, 1992).
[24] C. Rackauckas and Q. Nie, Discrete Contin. Dyn. Syst. - B **22**, 2731 (2017).
[25] H. M. van der Kooij and J. Sprakel, Soft Matter **11**, 6353 (2015).
[26] M. Schulz and J. L. Keddie, Soft Matter **14**, 6181 (2018).
[27] D. K. Makepeace, A. Fortini, A. Markov, P. Locatelli, C. Lindsay, S. Moorhouse, R. Lind, R. P. Sear, and J. L. Keddie, Soft Matter **13**, 6969 (2017).
[28] R. E. Trueman, E. Lago Domingues, S. N. Emmett, M. W. Murray, J. L. Keddie, and A. F. Routh, Langmuir **28**, 3420 (2012).
[29] B. He, I. Martin-Fabiani, R. Roth, G. I. Tóth, and A. J. Archer, Langmuir **37**, 1399 (2021).
[30] A. Fortini and R. P. Sear, Langmuir **33**, 4796 (2017).
[31] K. Burrage and P. M. Burrage, Appl. Numer. Math. **22**, 81 (1996).
[32] A. Rößler, SIAM J. Numer. Anal. **48**, 922 (2010).
[33] E. Fehlberg, Computing **6**, 61 (1970).
[34] A. Rößler, Proc. Appl. Math. Mech. **2**, 461 (2003).
[35] O. C. Ibe, *Markov Processes for Stochastic Modeling* (Elsevier, 2013).
[36] J. G. Gaines and T. J. Lyons, SIAM J. Appl. Math. **57**, 1455 (1997).
[37] S. Mauthner, J. Comput. Appl. Math. **100**, 93 (1998).
[38] P. M. Burrage and K. Burrage, SIAM J. Sci. Comput. **24**, 848 (2002).
[39] H. Lamba, J. Comput. Appl. Math. **161**, 417 (2003).
[40] P. M. Burrage, R. Herdiana, and K. Burrage, J. Comput. Appl. Math. **170**, 317 (2004).

[41] H. Lamba, J. C. Mattingly, and A. M. Stuart, IMA J. Numer. Anal. **27**, 479 (2006).
[42] V. Sotiropoulos and Y. N. Kaznessis, J. Chem. Phys. **128**, 014103 (2008).
[43] S. Ilie, J. Chem. Phys. **137**, 234110 (2012).
[44] A. M. Stuart and A. R. Humphries, *Dynamical Systems and Numerical Analysis*, Cambridge Monographs on Applied and Computational Mathematics No. 2 (Cambridge University Press, Cambridge, NY, 1996).
[45] R. L. Honeycutt, Phys. Rev. A **45**, 600 (1992).
[46] M. Fixman, Macromolecules **19**, 1195 (1986).
[47] A. Iniesta and J. García de la Torre, J. Chem. Phys. **92**, 2015 (1990).
[48] D. M. Heyes and A. C. Brańka, Mol. Phys. **98**, 1949 (2000).
[49] This issue has been discussed in https://github.com/SciML/DiffEqNoiseProcess.jl/pull/80.
[50] C. Rackauckas and Q. Nie, J. Open Res. Software **5**, 15 (2017).
[51] The corresponding implementation in C++ can be found in https://gitlab.uni-bayreuth.de/bt306964/mbd.
[52] L. Yeomans-Reyna, H. Acuña-Campa, F. d. J. Guevara-Rodríguez, and M. Medina-Noyola, Phys. Rev. E **67**, 021108 (2003).
[53] A. J. Archer, P. Hopkins, and M. Schmidt, Phys. Rev. E **75**, 040501 (2007).
[54] P. Hopkins, A. Fortini, A. J. Archer, and M. Schmidt, J. Chem. Phys. **133**, 224505 (2010).
[55] D. Stopper, R. Roth, and H. Hansen-Goos, J. Phys.: Condens. Matter **28**, 455101 (2016).
[56] D. Stopper, A. L. Thorneywork, R. P. A. Dullens, and R. Roth, J. Chem. Phys. **148**, 104501 (2018).
[57] L. L. Treffenstädt and M. Schmidt, Phys. Rev. Lett. **126**, 058002 (2021).
[58] A. Fortini, I. Martín-Fabiani, J. L. De La Haye, P.-Y. Dugas, M. Lansalot, F. D'Agosto, E. Bourgeat-Lami, J. L. Keddie, and R. P. Sear, Phys. Rev. Lett. **116**, 118301 (2016).
[59] M. Laurati, K. J. Mutch, N. Koumakis, J. Zausch, C. P. Amann, A. B. Schofield, G. Petekidis, J. F. Brady, J. Horbach, M. Fuchs, and S. U. Egelhaaf, J. Phys.: Condens. Matter **24**, 464104 (2012).
[60] N. Jahreis and M. Schmidt, Colloid Polym. Sci. **298**, 895 (2020).
[61] A. J. Archer and A. Malijevský, Mol. Phys. **109**, 1087 (2011).
[62] C. P. Royall, J. Dzubiella, M. Schmidt, and A. van Blaaderen, Phys. Rev. Lett. **98**, 188304 (2007).
[63] J. Bleibel, S. Dietrich, A. Domínguez, and M. Oettel, Phys. Rev. Lett. **107**, 128302 (2011).
[64] I. C. Carpen and J. F. Brady, J. Rheol. **49**, 1483 (2005).
[65] H. Löwen, J.-P. Hansen, and J.-N. Roux, Phys. Rev. A **44**, 1169 (1991).
[66] A. J. Archer, A. M. Rucklidge, and E. Knobloch, Phys. Rev. E **92**, 012324 (2015).
[67] A. J. Archer, A. M. Rucklidge, and E. Knobloch, Phys. Rev. Lett. **111**, 165501 (2013).
[68] G. Volpe, S. Gigan, and G. Volpe, Am. J. Phys. **82**, 659 (2014).
[69] T. F. F. Farage, P. Krinninger, and J. M. Brader, Phys. Rev. E **91**, 042310 (2015).
[70] S. Paliwal, V. Prymidis, L. Filion, and M. Dijkstra, J. Chem. Phys. **147**, 084902 (2017).
[71] S. Paliwal, J. Rodenburg, R. van Roij, and M. Dijkstra, New J. Phys. **20**, 015003 (2018).
[72] A. K. Omar, K. Klymko, T. GrandPre, and P. L. Geissler, Phys. Rev. Lett. **126**, 188002 (2021).
[73] F. Turci and N. B. Wilding, Phys. Rev. Lett. **126**, 038002 (2021).
[74] U. Baul and J. Dzubiella, J. Phys.: Condens. Matter **33**, 174002 (2021).
[75] K. Gustafsson, ACM Trans. Math. Software **20**, 496 (1994).
[76] G. Söderlind, Numer. Algorithms **31**, 281 (2002).
[77] G. Söderlind, ACM Trans. Math. Software **29**, 1 (2003).
[78] L. Verlet, Phys. Rev. **159**, 98 (1967).
[79] W. C. Swope, H. C. Andersen, P. H. Berens, and K. R. Wilson, J. Chem. Phys. **76**, 637 (1982).
[80] G. N. Milstein, IMA J. Numer. Anal. **23**, 593 (2003).
[81] K. Burrage, I. Lenane, and G. Lythe, SIAM J. Sci. Comput. **29**, 245 (2007).
[82] W. F. van Gunsteren and H. J. C. Berendsen, Mol. Phys. **45**, 637 (1982).
[83] A. Brünger, C. L. Brooks, and M. Karplus, Chem. Phys. Lett. **105**, 495 (1984).
[84] R. D. Skeel and J. A. Izaguirre, Mol. Phys. **100**, 3885 (2002).
[85] N. Goga, A. J. Rzepiela, A. H. de Vries, S. J. Marrink, and H. J. C. Berendsen, J. Chem. Theory Comput. **8**, 3637 (2012).
[86] B. Leimkuhler and C. Matthews, *Molecular Dynamics, Interdisciplinary Applied Mathematics* (Springer International Publishing, Cham, 2015), Vol. 39.