An Exact Auxiliary Variable Gibbs Sampler for a Class of Diffusions

Qi Wang  
Department of Statistics, Purdue University  
and  
Vinayak Rao  
Department of Statistics, Purdue University  
and  
Yee Whye Teh  
Department of Statistics, University of Oxford  
March 27, 2019

Abstract

Stochastic differential equations (SDEs) or diffusions are continuous-valued continuous-time stochastic processes widely used in the applied and mathematical sciences. Simulating paths from these processes is an intractable problem, and usually involves time-discretization approximations. We propose an asymptotically exact Markov chain Monte Carlo sampling algorithm that involves no such time-discretization error. Our sampler is applicable both to the problem of prior simulation from an SDE, as well as posterior simulation conditioned on noisy observations. Our work recasts an existing rejection sampling algorithm for diffusions as a latent variable model, and then derives an auxiliary variable Gibbs sampling algorithm that targets the associated joint distribution. At a high level, the resulting algorithm involves two steps: simulating a random grid of times from an inhomogeneous Poisson process, and updating the SDE trajectory conditioned on this grid. Our work allows the vast literature of Monte Carlo sampling algorithms from the Gaussian process literature to be brought to bear to applications involving diffusions. We study our method on synthetic and real datasets, where we demonstrate superior performance over competing methods.

Keywords: Brownian motion, Markov chain Monte Carlo, Poisson process, stochastic differential equations
1 Introduction

Diffusion processes are a class of stochastic processes that have been deeply studied and widely applied across a variety of theoretical and applied domains. Diffusions can be expressed as solutions to stochastic differential equations (SDEs) (Øksendal, 2003), and are continuous-time Markov processes whose realizations are continuous paths. The most well-known example is Brownian motion, corresponding to a random walk through some finite-dimensional Euclidean space. Brownian motion is characterized by two fixed parameters: a drift coefficient $\alpha$, and a diffusion coefficient $\sigma$. SDEs generalize this, allowing the drift and diffusion to depend on the current state of the process. A simple example is the Ornstein-Uhlenbeck (OU) process (Uhlenbeck and Ornstein, 1930; Øksendal, 2003), where the drift equals the negative of the difference between the current state and some constant $\mu$, resulting in mean-reverting dynamics. Closely related is the Brownian bridge (Øksendal, 2003), where the drift at any time $t$ is this negative difference divided by $T - t$, the time remaining till the end of an interval $[0, T]$. This ensures that with probability one, the process ends at $\mu$ at time $T$. The OU process and the Brownian bridge are still simple Gauss-Markov processes, with the distribution over the process value at some future time following an easy-to-compute normal distribution. More general drift and diffusion dependencies allow SDEs to model rich, mechanistic, nonlinear and nonstationary phenomena from a variety of applied disciplines. Examples include applications from astronomy (Schuecker et al., 2001), the biological sciences (Ricciardi, 2013), cognitive psychology (Hanes and Schall, 1996; Tuerlinckx et al., 2001), ecology (Holmes, 2004), economics (Bergstrom, 1990), genetics (Holland, 1976; Karlin and Tavaré, 1983; Lange, 2003), geology (Ditlevsen et al., 2002), mathematical finance (Shreve, 2004; Black and Scholes, 1973), physics (Keller et al., 1995), neurology (Holden, 1976), political and social sciences (Cobb, 1981), and many other fields of science and engineering.

The flexibility that SDEs offer comes at a severe computational cost, especially in data-driven applications. With a few exceptions, the nonlinear, continuous-time dynamics of SDEs result in distributions over future values that are not just non-Gaussian, but also unavailable in closed form. If an SDE forms a prior distribution over paths, then even simulating from this distribution forms an intractable problem. Given noisy measurements via some measurement process, posterior simulation is an even more challenging problem. As a consequence, both prior and posterior simulation are typically carried out by imposing approximations through time-discretization, common
examples being Euler-Maruyama or Millstein approximations (Kloeden et al., 2012). While this allows ideas from the discrete-time literature to be used, time-discretization introduces errors into inferences, and controlling these requires fine discretization grids and expensive computation.

Our main contribution in this paper is an auxiliary variable Markov chain Monte Carlo (MCMC) algorithm that targets the posterior distribution over paths exactly without any such approximations. Our scheme builds on a rejection sampling algorithm that allows exact simulation from a class of SDEs, outlined in the papers Beskos and Roberts (2005); Beskos et al. (2006b,a). Our work recasts this rejection sampling algorithm as a latent variable model, and then derives a Gibbs sampling algorithm that at a high level involves two simulation steps: 1) simulate a random grid of times from an inhomogeneous Poisson process conditioned on a set of diffusion values, and 2) update the diffusion values on this Poisson grid. Our algorithm allows us to easily use standard tools from the vast Gaussian process literature (Titsias et al., 2008), and also allows conditional simulations from SDEs given noisy observations. Our focus is mostly on one-dimensional diffusions, although our ideas also apply to diffusions which, after a transformation, have a constant diffusion function $\sigma(\cdot)$. A more serious restriction is that like Beskos and Roberts (2005), our algorithm applies to diffusions whose Radon-Nikodym derivative with respect to a biased Brownian bridge is bounded (see section 2.2): we call these SDEs of class EA1. It is however conceptually easy to consider generalizations of our ideas to larger classes of diffusions (called EA2 and EA3).

We organize our paper as follows. Section 2 briefly introduces stochastic differential equations, and then describes the exact EA1 algorithm of Beskos et al. (2006b). Section 3 sets up the general Bayesian model for which we wish to carry out posterior inference, and then describes our proposed MCMC algorithm in this broader setting. We discuss related work in section 4, while in section 5 and 6 we evaluate our, and two other sampling algorithms on synthetic and real datasets.

2 Stochastic differential equations (SDEs)

A diffusion $X_t$ is a continuous-valued continuous-time Markov process that solves the SDE

$$dX_t = \alpha(X_t, \theta)dt + \sigma(X_t, \theta)dB_t. \quad (1)$$

The process is driven by a stochastic Brownian motion whose value at time $t$ is given by $B_t$. The functions $\alpha(\cdot, \theta)$ and $\sigma(\cdot, \theta)$ are the drift and diffusion terms respectively, while $\theta$ represents parameters that govern the dynamics of the system. This paper is concerned with path inference:
we leave parameter inference for future work, and will drop all dependencies on $\theta$ in the following.

Informally, equation (1) implies that $dX_t$, the infinitesimal change in the value of the diffusion at time $t$, is comprised of two parts, a deterministic and a stochastic component. The former is determined by the current value $X_t$ of the diffusion transformed by $\alpha(\cdot)$, while the latter is an increment of Brownian motion $dB_t$, scaled by $X_t$ transformed by diffusion term $\sigma(\cdot)$. In general $X_t$ and $B_t$ can be $d$-dimensional vectors lying in $\mathbb{R}^d$, with $\alpha(X_t) \in \mathbb{R}^d$ and $\sigma(X_t) \in \mathbb{R}^{d \times d}$. For one-dimensional diffusions, all these quantities are real scalars.

In this paper, as in [Beskos and Roberts (2005)] and follow-up papers, we will assume that the diffusion coefficient $\sigma(\cdot) = 1$. Thus, we will be dealing with diffusions solving the equation

$$dX_t = \alpha(X_t)dt + dB_t.$$  

(2)

For one-dimensional diffusions, this is a mild assumption, since a general SDE can be transformed to have a diffusion coefficient of one via the Lamperti transform [Møller and Madsen (2010)]. This involves scaling the diffusion by the function $\eta(x) = \int_{-\infty}^{x} \frac{1}{\sigma(u)} du$. Now, the process $X'_t = \eta(X_t)$ is a diffusion with diffusion coefficient equal to 1 [Møller and Madsen (2010)]. In what follows, we assume such a transformation has been applied to produce our SDE of interest. In higher-dimensions, the restriction to constant diffusion does involve loss of generality. Nevertheless, this forms a broad class with practical use in fields like finance, biology and neuroscience (see also [Beskos and Roberts, 2005][Beskos et al., 2006b]).

2.1 Simulation via the Euler-Maruyama Method

The Euler-Maruyama method ([Iacus 2009][Kloeden et al. 2012]) forms the simplest approach to simulating general diffusions over an interval $[0, T]$. This simplicity comes at the price of approximation error. Under the Euler-Maruyama scheme, one chooses a time-discretization granularity $\Delta t$, with the change in the diffusion value $\Delta X_t := X_{t+\Delta t} - X_t$ approximated as

$$\Delta X_t \approx \alpha(X_t)\Delta t + \sigma(X_t)\Delta B_t,$$

(3)

where $\Delta B_t \sim N(0, \Delta t)$. Effectively, the change $\Delta X_t$ follows a conditionally Gaussian distribution:

$$\Delta X_t \sim N(\alpha(X_t)\Delta t, \sigma(X_t)\Delta t).$$

(4)

The discretization error from the Euler-Maruyama method can be reduced by using a finer discretization grid. Alternately, more sophisticated approaches like the Millstein algorithm [Kloeden et al. 2012] can provide more accurate approximations for a fixed time resolution.
2.2 An exact simulation algorithm (EA1) for diffusion processes

Algorithms like the Euler-Maruyama method allow easy simulation of paths from general SDEs. They however come at the price of a discretization error. The algorithm of Beskos and Roberts (2005) on the other hand allows exact simulation from a class of SDEs with diffusion coefficient 1. In follow up work (Beskos et al., 2006b), this was extended to a broader class of such SDEs, though we focus on the original algorithm, called the Exact algorithm 1 or EA1. We refer to the associated family of SDEs as class EA1, which we characterize below. At a high level, EA1 is a rejection sampling scheme, where proposals are made from a simple stochastic process (a Brownian motion), and are accepted or rejected with appropriate probability. The ingenuity of the algorithm lies in a retrospective sampling scheme that only requires evaluating the paths on a finite set of times.

Assume that at time 0 the diffusion has initial value $X_0 = x$; later we will place a probability $\pi$ over $X_0$. Since the diffusion coefficient equals 1, the resulting stochastic process differs from standard Brownian motion only through the drift function $\alpha(\cdot)$. Informally, this results in paths from the SDE having the same ‘roughness’ as the Brownian motion paths. A consequence is that the probability measure over paths specified by the diffusion process is absolutely continuous with respect to the probability measure corresponding to Brownian motion. This is formalized by the Girsanov theorem (Øksendal, 2003), that characterizes the diffusion process via a Radon-Nikodym derivative with respect to Brownian motion.

Write $C$ for the space of continuous functions on $[0, T]$. We will refer to generic elements of this space as $\omega$. For paths $\omega$ with initial value $x$, write $W_x$ and $Q_x$ for path probability measures corresponding to Brownian motion and the SDE respectively. Then, under standard assumptions (we refer to Øksendal (2003) for more details), we have

**Theorem 2.1** (Girsanov’s theorem). The Radon-Nikodym derivative $\frac{dQ_x}{dW_x}$ satisfies

$$
\frac{dQ_x}{dW_x}(\omega) = \exp\left\{ \int_0^T \alpha(\omega_t)d\omega_t - \frac{1}{2} \int_0^T \alpha^2(\omega_t)dt \right\}.
$$

(5)

Let $A(u) = \int_0^u \alpha(t)dt$, and recall the definition of a Brownian bridge: this is just a Brownian motion conditioned on its end points. For a density $h_x(u) \propto \tilde{h}_x(u) := \exp(A(u) - (u - x)^2/2T)$, define an $h_x$-biased Brownian bridge as a stochastic process starting at $x$, ending with a value $X_T$ drawn from $h_x$, with the two points linked by a Brownian bridge. Write $Z_x$ for the law of
this process. Note that for this to be well defined, \( \tilde{h}_x \) must be normalizable, so that the integral 
\[
\int \tilde{h}_x(u)du = \int \exp(A(u) - (u - x)^2/2T)du
\]
is finite. Then we have (Beskos and Roberts, 2005):

**Proposition 2.2.** Let the drift function \( \alpha \) satisfy the conditions of Girsanov’s theorem and be continuously differentiable. Then
\[
\frac{dQ_x}{dZ_x}(\omega) \propto \exp \left\{ -\frac{1}{2} \int_0^T \left( \alpha^2(\omega_t) + \alpha'(\omega_t) \right) dt \right\}. \tag{6}
\]

**Proof.** Write \( A_t = A(\omega_t) \). By Itô’s lemma (Øksendal, 2003),
\[
dA_t = \frac{\partial A_t}{\partial t} dt + \frac{\partial A_t}{\partial \omega_t} d\omega_t + \frac{1}{2} \frac{\partial^2 A_t}{\partial \omega_t^2} dt = 0 + \alpha(\omega_t) d\omega_t + \frac{1}{2} \alpha'(\omega_t) dt. \tag{7}
\]

Solving for \( \int_0^T \alpha(\omega_t) d\omega_t \) and substituting in equation (5), we get
\[
\frac{dQ_x}{dW_x}(\omega) = \exp \left\{ A(\omega_T) - A(\omega_0) - \frac{1}{2} \int_0^T \left( \alpha^2(\omega_t) + \alpha'(\omega_t) \right) dt \right\}. \tag{8}
\]

By definition, the measure \( Z_x \) is a reweighting of \( W_x \) by \( h_x(\omega_T) \). Thus,
\[
\frac{dQ_x}{dZ_x}(\omega) \propto \exp \left\{ -A(\omega_0) - \frac{1}{2} \int_0^T \left( \alpha^2(\omega_t) + \alpha'(\omega_t) \right) dt \right\}. \tag{9}
\]

Since we are fixing \( \omega_0 = x, A(x) \) is a constant, and the result follows. \( \square \)

We now come to the key assumption of the EA1 algorithm of Beskos and Roberts (2005):

**Definition 1.** An SDE belongs to class EA1 if it satisfies the assumptions of Proposition 2.2, and its drift function \( \alpha \) satisfies \( \alpha^2(\cdot) + \alpha'(\cdot) \in [L, L + M] \) for finite \( L \) and \( M \).

We focus on SDEs of class EA1. Adding and subtracting \( L \) from the exponent in equation (9),
\[
\frac{dQ_x}{dZ_x}(\omega) \propto \exp \left\{ -\frac{1}{2} \int_0^T \left( \alpha^2(\omega_t) + \alpha'(\omega_t) - L \right) dt \right\} := \exp \left\{ -\int_0^T \phi(\omega_t) dt \right\} := \rho(\omega). \tag{10}
\]

Since \( L \) is the infimum of \( \alpha^2 + \alpha' \), the function \( \phi(\cdot) = \alpha^2(\cdot) + \alpha'(\cdot) - L \) is positive, and exponentiating its negative integral gives a number \( \rho(\omega) \) between 0 and 1. This suggests a rejection sampling scheme (Robert and Casella, 2005) to simulate from \( Q_x \): propose a path from the stochastic process \( Z_x \), and accept it with probability \( \rho(\omega) \). Naively, this requires 1) simulating the entire path \( \omega \), and 2) transforming and integrating \( \omega \) to calculate \( \rho(\omega) \), both being impossible steps. The EA1 algorithm bypasses this by recognizing that equation (10) gives the probability that
a Poisson process with intensity $\{\phi(\omega_t), t \in [0, T]\}$ produces 0 events on the interval $[0, T]$. It takes the approach of partially ‘uncovering’ the path $\omega$, simulating it on a finite set of times, until the number of Poisson events is determined. To do this, the EA1 algorithm exploits the bound $\phi(\cdot) \leq M$ to simulate a rate-$\phi(\omega_t)$ Poisson process via the thinning theorem (Lewis and Shedler, 1979). It does this in three steps: simulate a Poisson process $\Psi$ with intensity $M$ on the interval $[0, T]$, instantiate an $h$-biased Brownian bridge $\omega_t$ on $\Psi$, and keep each point $t_i \in \Psi$ with probability $\phi(\omega_{t_i})/M$. The surviving points then form an exact realization from a rate-$\phi(\omega)$ Poisson process. The probability this Poisson process has 0 events is given by equation (10).

Now, the EA1 algorithm involves repeatedly simulating from the rate-$\phi(\omega)$ Poisson process this way until a realization with no events is produced. We write the corresponding path as $X$, this forms an exact realization of the SDE of interest. Note that at this stage, we only have $X_0, X_T$ and $X_\Psi$, the last being the values of the diffusion uncovered on the times in the Poisson set $\Psi$. We will refer to the pair $(0 \cup \Psi \cup T, X_0 \cup X_\Psi \cup X_T)$ as the diffusion ‘skeleton’, this forms a sufficient statistic that allows the diffusion at any other set of times to be easily and exactly simulated. For this, we recognize that the accepted path was a proposal from a biased Brownian bridge, but which only was evaluated at times in $0 \cup \Psi \cup T$. It can retrospectively be uncovered at a set of times by conditionally simulating from a Brownian bridge. Consider a set of times $G$ between two successive elements $t_i$ and $t_{i+1}$ of $\Psi$. We simulate $X_G$, the diffusion evaluated on $G$, from a Brownian bridge with endpoints $X_{t_i}$ and $X_{t_{i+1}}$. We write this as $X_G \sim BB_G(t_i, X_{t_i}, t_{i+1}, X_{t_{i+1}})$ (see equation (25) in the appendix). Algorithm 1 describes all steps involved with the EA1 algorithm.

3 Posterior simulation for SDEs

The EA1 algorithm, while exact, can suffer from high rejection rates. This happens when dealing with long time intervals, or when the drift $\alpha(\cdot)$ causes $Q_x$ to differ significantly from the $h$-biased Brownian proposal $Z_x$. Furthermore, the EA1 algorithm is primarily designed to simulate from an SDE prior or an end-point conditioned SDE. As we describe below, extending it to simulating SDE paths from conditional distributions given noisy observations can be challenging, with current approaches depending on the EA1 rejection sampling algorithm. Our proposed sampler aims to address both these problems, and brings sampling algorithms from the Gaussian process literature to applications with SDEs. Before we describe our algorithm, we first setup the general problem.
Algorithm 1 To simulate an SDE of class EA1 with drift term $\alpha(\cdot)$ over an interval $[0, T]$

**Input:** An initial distribution over the diffusion state $\pi(\cdot)$, a finite grid of times $G \in [0, T]$.  

**Output:** A diffusion skeleton $(0 \cup \Psi \cup T, X_0 \cup X_\Psi \cup X_T)$.  

The diffusion values $X_G$ evaluated on the grid $G$.

1: Calculate $A(\cdot), \phi(\cdot)$, and the constants $L$ and $M$ from $\alpha(\cdot)$, and set accept to false.  
2: while accept = false do  
   ▷ Rejection sampling
3: Simulate a rate-$M$ Poisson process $\Psi = \{t_1, t_2, \ldots, t_{|\Psi|}\}$ on $[0, T]$.  
4: At the start time 0, simulate the initial value $X_0$ of the diffusion from $\pi$.  
5: At the end point $T$, simulate $X_T$ from $h_{X_0}(X_T) \propto \exp(A(X_T) - (X_T - X_0)^2/2T)$.  
6: Simulate a Brownian bridge connecting $(0, X_0)$ and $(T, X_T)$ on the times $\Psi$:  
   \[X_\Psi \sim \text{BB}_\Psi(0, X_0, T, X_T)\]  
   (see equation (25) for details).  
7: For $i \in \{1, \ldots, |\Psi|\}$, simulate $u_i \sim \text{Uniform}(0, 1)$. If all $u_i > \frac{\phi(X_{t_i})}{M}$, set accept = true.  
8: end while  
9: for $i$ in $\{0, \ldots, |\Psi|\}$ do  
   ▷ Impute diffusion on $G$
10: Define $t_0 = 0, t_{|\Psi|+1} = T$ and $G_i = G \cap (t_i, t_{i+1})$. Simulate $X_{G_i} \sim \text{BB}_{G_i}(t_i, X_{t_i}, t_{i+1}, X_{t_{i+1}})$.  
11: end for

3.1 Bayesian model

Consider a latent trajectory $X = \{X_t : t \in [0, T]\}$ on the interval $[0, T]$. We model this as a realization of an SDE of class EA1, with drift function $\alpha(\cdot)$, with distribution $\pi$ on the initial state $X_0$. Following our previous notation, our prior distribution on the process $X_0 \times \{X_t : t \in (0, T]\}$ equals the product measure $\pi \times \mathbb{Q}_{X_0}$. Write this as $\mathbb{Q}_\pi$. We are given noisy measurements of the latent trajectory, with likelihood $\ell(X)$. We will assume this depends only on the trajectory values at a finite set of times $O$, so that $\ell(X) = \ell(X_O)$ (without loss of generality, we will let $O$ include 0 and $T$). A simple example is when we have measurements with i.i.d. noise at the set of times $O$, so that $\ell(X) = \prod_{o \in O} \ell_o(X_o)$. We can also consider more complex likelihoods, where this condition holds after augmenting the observations with additional variables. Examples of such likelihoods include point processes [Adams et al., 2009] [Rao and Teh, 2011], jump processes [Rao and Teh, 2013], or even other diffusions modulated by the latent SDE trajectory. In this case, our MCMC sampler will include such data-augmentation as an inner step. Obviously, our setup includes the
problem of prior simulation, where there are no observations.

Given the observations, our goal is to simulate from the conditional distribution over paths under a prior $Q_\pi$, given the observations with likelihood $\ell(X)$. Write this as $Q_{\pi,\ell}$, which forms the posterior distribution over paths under our Bayesian model. Observe that this satisfies

$$\frac{dQ_{\pi,\ell}}{dQ_\pi}(\omega) \propto \ell(\omega).$$

The EA1 algorithm, as outlined in section 2.2, can only simulate trajectories from the prior distribution $Q_\pi$. To address the problem of posterior simulation, Beskos et al. (2006b) break the interval $[0, T]$ into sub-intervals $[o_i, o_{i+1}]$, where $o_i$ is the $i$th element of $O$. They then propose an MCMC algorithm that repeats two steps: 1) given $X_O$, the diffusion evaluated on the set of observation times $O$, use the rejection-sampling algorithm to simulate an SDE skeleton within each sub-interval $(o_i, o_{i+1})$, and 2) conditioned on the skeleton, use the likelihood to update the SDE values at the observation times $O$. The first step exploits the Markov property of the SDE, and involves running the EA1 rejection sampling algorithm independently for each interval $[o_i, o_{i+1}]$. The second step involves simulating each $X_o, o \in O$ from the posterior distribution resulting from a Brownian bridge prior and the likelihood associated with the observation at time $o$. Even without observations, such an approach can be useful to deal with high rejection rates from long time-intervals or strong drift terms: break up the interval into smaller sub-intervals, and alternately simulate the SDE on the ends of the sub-intervals given the skeleton, and then the skeleton within each sub-interval given the edge values. As the sub-intervals get smaller and smaller, the rejection rates of the biased Brownian bridge proposals become correspondingly smaller.

Such an approach, while useful, can scale badly with high observation-rates, as is common in fields like high-frequency finance. Even with low to moderate observation rates, it can be necessary to partition the observation interval into small sub-intervals to maintain low rejection rates. This slows down MCMC mixing, since 1) we are instantiating more of the diffusion path, and 2) rather than updating the entire path in a single step, we conditionally update part of the trajectory given the rest. The finer the sub-intervals, the stronger the coupling, and thus, the poorer the mixing. Choosing an appropriate grid resolution is thus not straightforward, involving a trade-off between fast mixing and low rejection rates. Next, we outline an algorithm that eliminates the EA1 rejection sampling step altogether, instead allowing practitioners to exploit the vast literature of MCMC algorithms for Gaussian process models in a fairly straightforward fashion.
3.2 Our proposed auxiliary variable Gibbs sampler for SDEs

In this section, we describe an MCMC sampling algorithm that targets the posterior distribution over trajectories $Q_{\pi,\ell}$ from our Bayesian model of the previous section. Note that this equals the prior $Q_{\pi}$ when the likelihood $\ell(\cdot)$ is a constant function. To keep our notation simple, we will write $Q_{\pi,\ell}$ as $Q$. Recall that $Z_x$ is an $h$-biased Brownian bridge starting at $x$. In a similar manner to $Q_x$, use $Z_x$ to define $Z_\pi$ and $Z_{\pi,\ell}$. Thus, $Z_\pi$ is the distribution over Brownian bridge paths, with values at time 0 and $T$ distributed as $\pi$ and $hX_0$ respectively. Treating this as a prior over paths, $Z_{\pi,\ell}$ is the posterior distribution corresponding to observations with likelihood $\ell(\omega)$. Again, we set $Z$ as equal to $Z_{\pi,\ell}$. It follows directly from equations (10) and (12) that for a path $\omega \in C$,

$$
\frac{dQ}{dZ}(\omega) \propto \exp \left\{ -\int_0^T \phi(\omega_t) dt \right\}. 
$$

(13)

Write $M$ for the space of all finite point process realizations on the interval $[0, T]$. Let $\mathcal{M}$ be the probability measure on $M$ corresponding to a rate-$M$ Poisson process (recall $M$ is the supremum of $\phi(\cdot)$). Define $Z^+ = Z \times \mathcal{M}$ as the product measure of the Brownian posterior $Z$ with $\mathcal{M}$. For a $\Psi \in \mathcal{M}$, define the measure $Q^+$ via the following Radon-Nikodym derivative with respect to $Z^+$:

$$
\frac{dQ^+}{dZ^+}(\omega, \Psi) = \prod_{t \in \Psi} \frac{1 - \phi(\omega_t)}{M}.
$$

(14)

Proposition 3.1. $Q^+$ has $Q$ as its marginal distribution: $\int_M dQ^+(\omega, \Psi) = dQ(\omega)$.

Proof. From equation (14), we have

$$
\int_M dQ^+(\omega, \Psi) = \int_M dZ^+(\omega, \Psi) \prod_{t \in \Psi} \left( 1 - \frac{\phi(\omega_t)}{M} \right) = dZ(\omega) \mathbb{E}_\mathcal{M} \left[ \prod_{t \in \Psi} \left( 1 - \frac{\phi(\omega_t)}{M} \right) \right],
$$

where $\mathbb{E}_\mathcal{M}$ is the expectation with respect to the Poisson measure $\mathcal{M}$, and the second equality follows from the fact that $Z^+ = Z \times \mathcal{M}$. From equation (13), we then need to show that

$$
\mathbb{E}_\mathcal{M} \left[ \prod_{t \in \Psi} \left( 1 - \frac{\phi(\omega_t)}{M} \right) \right] = \exp \left\{ -\int_0^T \phi(\omega_t) dt \right\}.
$$

This is intuitively clear from the thinning theorem, and can be formally proved by an easy application of Campbell’s theorem [Kingman 1992].

While our goal is to produce samples from the distribution $Q$, our MCMC sampler is an auxiliary variable sampler that targets the joint distribution $Q^+$. Its state-space is the SDE trajectory $X$ as well as the random set of Poisson times $\Psi$. Proposition 3.1 tells us that discarding the Poisson times $\Psi$ produces trajectories $X$ from the desired conditional distribution $Q$. Our
algorithm takes a Gibbs sampling approach, and targets the distribution \( Q^+ \) by repeating two steps: simulate Poisson times \( \Psi \) given the path \( X \), and update the path given the Poisson times. Equation (14) allows us to derive two simple corollaries that underpin our Gibbs sampler.

**Corollary 3.2.** Conditioned on the trajectory \( X \), the Poisson events follow an inhomogeneous Poisson process with rate function \( M - \phi(X) \).

**Proof.** For fixed \( X \), it follows from equation (14) that \( \Psi \) is a point process with density with respect to a rate-M Poisson process given by \( \prod_{t \in \Psi} \left( 1 - \frac{\phi(X_t)}{M} \right) \). It is clear from the thinning theorem that this is a rate \( M - \phi(X) \) Poisson process. A formal proof simply involves applying Campbell’s theorem (Kingman, 1992) to calculate the Laplace functional of this process.

An important consequence of this corollary is that conditioned on \( X \), the distribution over \( \Psi \) does not depend on the likelihood \( \ell(X) \). The observations enter only when we update \( X \). Our second corollary concerns updating the path \( X \) given the Poisson times \( \Psi \).

**Corollary 3.3.** Conditioned on the Poisson times \( \Psi \), the trajectory \( X \) has density with respect to \( Z_\pi \) given by \( h_{X_0}(X_T)\ell(X_O) \prod_{t \in \Psi} \left( 1 - \frac{\phi(X_t)}{M} \right) \).

**Proof.** Conditioned on the times \( \Psi \), from equation (14), we see that \( X \) has density with respect to \( Z \) proportional to \( \prod_{t \in \Psi} 1 - \frac{\phi(X_t)}{M} \). The result follows from the definition of \( Z = Z_\pi,\ell \).

The above result shows us that conditioned on the Poisson skeleton \( \Psi \), the probability density of the SDE path evaluated \( \Psi \) and \( O \) (write this as \( X_{\Psi,O} \)) is given by

\[
p(X_{\Psi,O}) \propto \pi(X_0)h_{X_0}(X_T)\text{BB}(X_{O,\Psi}|0, X_0, T, X_T)\ell(X_O) \prod_{g \in \Psi} \left( 1 - \frac{\phi(X_g)}{M} \right). \tag{15}
\]

This corresponds to a fairly typical posterior distribution in applications involving Gaussian processes (Williams and Rasmussen, 2006). Here our prior over trajectories is the Brownian motion prior \( \mathbb{W}_\pi \), and our likelihood is \( h_{X_0}(X_T)\ell(X_O) \prod_{g \in \Psi} \left( 1 - \frac{\phi(X_g)}{M} \right) \). Consequently, after conditioning on the Poisson grid \( \Psi \), we do not need to calculate intractable SDE transition probabilities to calculate prior probabilities over the trajectory \( X \). The SDE posterior is amenable to standard Gaussian process MCMC techniques. For a survey of such methods, see for example Titsias et al. (2008), we will use Hamiltonian Monte Carlo (Neal, 2011).
3.3 Gibbs sampler details

Corollaries 3.2 and 3.3 provide the basis of our Gibbs sampling algorithm. Each iteration of this algorithm starts with a pair \((X_{\Psi \cup O}, \Psi)\), and repeats two steps: simulate a new Poisson grid \(\Psi^*\) given \((X_{\Psi \cup O}, \Psi)\), and then simulate a new set of diffusion values \(X_{\Psi^* \cup O}^*\) given \(\Psi^*\). Recall that the set \(O\) includes the start and end times, 0 and \(T\). There are a few issues that must be resolved to translate these into a practical algorithm. We detail these below.

**Simulating a new Poisson grid \(\Psi^*\) conditioned on \(X_{\Psi \cup O}\):** Corollary 3.2 shows that conditioned on the entire trajectory \(X\), \(\Psi\) is a Poisson process with rate \(\{M - \phi(X_t), t \in [0, T]\}\). In practice, our sampler will only evaluate \(X\) on the current set of Poisson times \(\Psi\) and on the observation times \(O\). To simulate the new times \(\Psi^*\), we exploit two facts: i) that the SDE skeleton summarizes the entire trajectory, whose values at other times can be retrospectively simulated from a Brownian bridge (steps 9 and 10 in algorithm 1), and ii) that \(M - \phi(\cdot) \leq M\). We will use these along with the thinning theorem to simulate from the rate \(M - \phi(X)\) inhomogeneous Poisson process. We first simulate a random set of times \(\Gamma\) from a rate-\(M\) Poisson process, and uncover \(X_\Gamma\), the trajectory on this set of times. This second step just involves simulating from Brownian bridges over intervals defined by successive elements of \(\Psi \cup O\) (algorithm 1, steps 9 and 10). Having imputed \(X\) on \(\Gamma\), we keep each element \(g \in \Gamma\) with probability \(1 - \frac{\phi(X_g)}{M}\), else we discard it. The set of surviving elements of \(\Gamma\) is a realization from a rate \(M - \phi(X)\) Poisson process, and forms the new times \(\Psi^*\). Along the way, we have evaluated \(X_{\Psi^*}\), the trajectory on this set of times. Finally, we discard the path evaluations on the old skeleton, since, under the new skeleton, these can easily be resampled (again, from a Brownian bridge). The first five panels in figure 1 shows these steps, where for simplicity we have ignored observations.

**Updating \(X\) conditioned on \(\Psi\):** Corollary 3.3 shows that conditioned on the Poisson grid \(\Psi^*\), \(X_{\Psi^* \cup O}\) has conditional probability density given by equation (15). This distribution, while intractable, can be evaluated up to a normalization constant, and is thus amenable to standard MCMC techniques. In particular, since equation (15) expresses a density with respect to Brownian motion, we can use any of the many techniques from the mature literature on Gaussian process posterior simulation, see (Titsias et al., 2008). In our experiments, we carry out this conditional update using Hamiltonian Monte Carlo (Neal, 2011) (see the experiments and appendix for more details). We note that rather than producing an independent sample of \(X_{\Psi^* \cup O}\), we are condition-
Algorithm 2 One iteration of the proposed auxiliary variable Gibbs sampler for EA1 diffusions

Input: A distribution $\pi(\cdot)$ over $X_0$, the initial value of the SDE
The drift term $\alpha(\cdot)$, and the associated quantities $A(\cdot), \phi(\cdot)$ and $M$
The Poisson times $\Psi$ and the corresponding path values $X_\Psi$
The SDE path values $X_O$ on the observation times $O$ (recall $O$ includes 0 and $T$).

Output: A new SDE skeleton $(\Psi^*, X_{\Psi^*})$, and new path values on $O$, $X^*_O$.

1: Simulate $\Gamma$ from a rate-$M$ Poisson process on $[0, T]$.
2: Define $\Gamma = \Psi \cup O$. Write its $i$th element as $g_i$, with $g_1 = 0$ and $g_{|\Gamma|} = T$.
3: for $i$ in 1 to $|\Gamma| - 1$ do
4: Define $\Gamma_i = \Gamma \cap (G_i, G_{i+1})$. Impute $X$ on $\Gamma_i$ from a Brownian bridge:
5: $X_{\Gamma_i} \sim BB(\Gamma_i(t_i, X_{t_i}, t_{i+1}, X_{t_{i+1}}))$.
6: end for
7: Discard each point $g \in \Gamma$ with probability $\frac{\phi(X_g)}{M}$. Write $(\Psi^*, X_{\Psi^*})$ for the set of surviving times and the associated path values.
8: Discarding everything other than $\Psi^*, X_{\Psi^*}$ and $X_O$.
9: Update $(X_{\Psi^*}, X_O) \equiv X_{\Psi^* \cup O}$ on $\Psi^* \cup O$ with a Markov kernel having stationary distribution

$$p(X_{\Psi^*}, X_O) \propto \pi(X_0)h_{X_0}(X_T)BB(X_O \mid \Psi^*)|X_0, X_T|\ell(X_O) \prod_{g \in \Psi^*} \left(1 - \frac{\phi(X_g)}{M}\right)$$ (see eq. (15)).

We use Hamiltonian Monte Carlo. Write the new values as $(X_{\Psi^*}, X^*_O)$. Return $(\Psi^*, X_{\Psi^*}, X^*_O)$.

ally updating this using a Markov kernel that has equation (15) as its stationary distribution. Our overall algorithm is thus a Metropolis-within-Gibbs (MWG) sampler. At the end of this step, we have a new set of path values $(X_{\Psi^*}, X^*_O)$. This is shown in the last panel of figure 1. Again, we can impute the SDE path $X^*$ at any other set of times from a Brownian bridge. Algorithm 2 outlines one iteration of our Gibbs sampler.

For completeness, we include the following theorem which states that our sampler targets the joint measure $Z^+$. Its proof is immediate (see Meyn and Tweedie (2009)): the sampler has $Z^+$ as its stationary distribution since the two Gibbs steps update the conditionals of $Z^+$. The sampler is irreducible under mild conditions on the Markov kernel used to update $X_{\Psi^* \cup O}$ given $\Psi^*$.

**Theorem 3.4.** The Gibbs sampler described above results in a Markov chain on the state space
Figure 1: One iteration of our proposed Gibbs sampling algorithm. For simplicity, we do not include observations (see Algorithm 2 for the general case). From the top-left to bottom-right:
1) The iteration starts with Poisson times $\Psi$ and the corresponding path values $X_{\Psi}$. This SDE skeleton is represented with the bold dots. Also shown in grey is the SDE path. This has not instantiated by the algorithm, but can easily be simulated at any finite set of times from a Brownian bridge.
2) Simulate $\Gamma$ from a rate-$M$ Poisson process on $[0, T]$ (shown as crosses).
3) Uncover $X_{\Gamma}$, the SDE on $\Gamma$, by simulating from a Brownian bridge (shown with hollow circles). 4) Discard each element $g \in \Gamma$ with probability $\frac{\phi(X_g)}{M}$. The points marked for deletion are kept hollow, while the surviving points filled in. 5) Discard the hollow dots and the original skeleton $(\Psi, X_{\Psi})$. The remaining times and values form the new skeleton, write this as $(\Psi^*, X_{\Psi^*})$. 6) Update $X_{\Psi^*}$ via some standard MCMC kernel (we use Hamiltonian Monte Carlo). The rest of the trajectory has also been refreshed, and can be simulated from a Brownian bridge.
with stationary distribution $Z^+ (\Psi, X)$.

Proof. This follows immediately from the fact that two steps of the Gibbs sampler target the conditional distributions of $Z^+ (\Psi, X)$.

4 Related work

Traditional approaches to simulating from an SDE involve time-discretization methods like the Euler-Maruyama method or Millstein’s method. Time-discretization also simplifies posterior simulation, opening up the vast literature on MCMC sampling for discrete-time time-series models. Example methods include particle MCMC (Andrieu et al., 2010), the embedded HMM (Neal et al., 2004), Hamiltonian Monte Carlo (Neal, 2011) among many others. Discrete-time approximations however introduce bias into the simulations, and characterizing their effect in hierarchical models is not easy. This makes it necessary to work with fine grids, resulting in long time-series and expensive computation. Further, controlling bias in this manner uncovers more of the diffusion, increasing coupling and degrading MCMC mixing (Liu, 1994; Roberts and Stramer, 2001).

There are two main approaches towards exact or unbiased estimation for diffusions to eliminate discretization error. As we described in subsection 2.2, our approach builds on a line of work starting from Beskos and Roberts (2005), who proposed a rejection sampling algorithm allowing exact simulation from the EA1 class of SDEs. Their EA1 rejection sampling algorithm can be viewed as a hierarchical model, where the Poisson grid $\Psi$ is sampled first, conditioned on which the associated path values $X_\Psi$ are sampled. After this, and conditioned on acceptance, the path values $X_O$ at the observation times are sampled, and then the observations themselves are generated. Conditioned on the diffusion path values $X_\Psi$ and $X_O$, the Poisson times $\Psi$ are independent of the observations. Our Gibbs sampler, which updates $X_{\Psi\cup O}$ given $\Psi$, and then $\Psi$ given $X_{\Psi\cup O}$ then operates on a centered parametrization (Papaspiliopoulos et al., 2007) or sufficient augmentation (Yu and Meng, 2011) of the joint distribution.

In Beskos et al. (2006b), the authors also show how prior simulation methods can be extended to posterior simulation given noisy observations. Like our method, this involves instantiating the diffusion skeleton (the Poisson times and associated diffusion values), as well as the diffusion values on observation times. However, as described in section 3.1, their algorithm alternately updates the diffusion skeleton given the values at observation times, and vice versa. By contrast, our algorithm
updates the entire set of path values given the Poisson times, and then Poisson times given these values, reducing the coupling between the Gibbs steps. Furthermore, the first Gibbs step of the algorithm of [Beskos et al. (2006b)] involves the EA1 rejection sampling algorithm, and can have high rejection rates. Controlling this requires instantiating more of the diffusion on additional grid points, and introducing these additional variables to control rejection rates will slow down mixing. Our MCMC algorithm does not face this problem.

In [Fearnhead et al. (2008)], the authors propose another discretization-free exact algorithm for posterior simulation. Their approach can be viewed as an exact version of our particle MCMC baseline (see section 5), with the simple Euler-Maruyama based particle filtering step replaced with a random-weight particle filter. Such an approach is a bit more involved than ours, and moreover comes with the same problems associated with particle filtering-based approaches, viz. small effective samples sizes and low MCMC acceptance rates with informative observations.

A second main line of work for unbiased estimation with SDEs (Rhee and Glynn, 2015) builds on multi-level Monte Carlo (MLMC) methods (Giles, 2008). These methods involve picking a random time-discretization granularity, so that the interval $[0, T]$ is uniformly split into $2^g$ subintervals for a random $g$. With some care, the resulting algorithms allow unbiased estimation of path functionals of the SDE. These methods have the advantage of being applicable to a wider class of SDEs than we considered here, in particular they do not require the availability of Lamperti transformation, and thus apply to more general multi-dimensional diffusions. Moreover these algorithms sometimes come with rigorous theoretical guarantees about error performance. However, they are primarily designed for unbiased estimation of path functionals of SDEs, rather than to simulate from SDEs. Furthermore, the focus here has mostly been on expectations under the SDE itself, rather than under the conditional distribution given observations, and extending to hierarchical models with SDEs remains an open question for future research. Some recent work in this direction includes [Franks et al.] (2018).

5 Experiments

In the following, we evaluate our method on a number of datasets, both synthetic and real. Besides comparing different settings of our algorithm, we also compare with two baselines: the EA1 rejection sampling algorithm of [Beskos and Roberts (2005)], and a particle Markov chain Monte
Carlo sampling algorithm (pMCMC) \cite{Andrieu2010}. We use the former in settings where we want to simulate from an SDE prior, allowing us to study trade-offs between producing cheap but dependent samples from our MCMC algorithm, and producing independent samples at the possible cost high rejection rates. Our second baseline, the pMCMC sampling algorithm, is a standard and relatively off-the-shelf tool to simulate from nonlinear state-space models. Effectively, pMCMC makes proposals from a particle filtering algorithm, which are then accepted or rejected with appropriate probability. Algorithm 4 in the appendix outlines the details of the algorithm. With SDEs, we run the particle filter on a discrete-time approximation based on the Euler-Maruyama method. This is thus an approximate MCMC algorithm, and will allow us to study what sort of computational cost our exact MCMC algorithm comes at. For pMCMC, we considered different settings for the discretization level as well as the number of particles. Performance was best for discretization levels between 0.1 and 0.01, and with the number of particles between 50 and 100, however we report a few additional settings as well. All experiments were carried out on a desktop with an Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz and 16GB RAM.

5.1 Example 1: The hyperbolic bridge

Our first example follows \cite{BeskosRoberts2005}, and considers the hyperbolic bridge:

\[
dX_t = -\frac{\theta X_t}{\sqrt{1 + X_t^2}} dt + \sigma dB_t, \quad \theta > 0. \tag{16}
\]

This is a special case of the general hyperbolic diffusion introduced in \cite{Barndorff-Nielsen1978}. As stated in Section 2, we fix the parameter \( \sigma \) to 1. In our experiments, we also set \( \theta = 1 \), though below, we consider the general setting. It is easy to verify that the drift \( \alpha(x) = -\frac{\theta x}{\sqrt{1 + x^2}} \) satisfies the assumptions of Girsanov’s theorem. We can calculate \( A(x) = \int_0^u \alpha(u) du = -\theta \sqrt{1 + x^2} \) and \( \alpha'(x) = -\frac{\theta}{(1+x^2)^{3/2}} \), showing that \( \frac{1}{2} (\alpha^2(x) + \alpha'(x)) = \frac{1}{2} \left( \frac{\theta^2 x^2}{1 + x^2} - \frac{\theta}{(1+x^2)^{3/2}} \right) \) lies in \( [\theta^2, \frac{\theta^2}{2}] \). We set

\[
\phi(x) := \frac{1}{2} (\alpha^2(x) + \alpha'(x)) + \frac{\theta}{2} = \frac{1}{2} \left( \frac{\theta^2 x^2}{1 + x^2} - \frac{\theta}{(1+x^2)^{3/2}} \right) + \frac{\theta}{2}. \tag{17}
\]

This lies in the interval \( [0, \frac{\theta^2}{2} + \frac{\theta}{2}] \). Accordingly, the EA1 Poisson process intensity \( M \) equals \( \frac{\theta^2}{2} + \frac{\theta}{2} \), and the associated \( h \)-biased Brownian bridge has \( h_x(\omega_T) \propto \exp \left( -\theta \sqrt{1 + \omega_T^2} + \theta \sqrt{1 + x^2} - \frac{(\omega_T-x)^2}{2T} \right) \).

**Tuning the HMC sampler:** A key step of our Gibbs sampler involves conditionally updating the SDE trajectory \( X \) given the Poisson grid \( \Psi \), following equation (15). We implement a Markov
Figure 2: Effective sample size per second (ESS/s) for different settings of stepsize $\epsilon$ (x-axis), each curve being a different number of leapfrog steps $N$ for the HMC sampler. From left to right, the three panels fix $T = 10, 20$ and 50 respectively. Different symbols represent the different values of $N$: ■ represents 1 step, • represents 2 steps, ▲ represents 5 steps and ♦ represents 10 steps.

kernel that targets this conditional distribution using Hamiltonian Monte Carlo (HMC) (Neal 2011), an widely used MCMC algorithm. We provide more details of our use of this algorithm in the appendix. HMC requires tuning three parameters $M, N$ and $\epsilon$, corresponding respectively to a mass matrix, the number of ‘leapfrog’ steps and the leapfrog stepsize. The latter two govern the leapfrog symplectic approximation to the Hamiltonian dynamics that HMC uses to update $X$. We fix $M$ to the identity matrix (see Neal (1996); Beskos et al. (2011) for more sophisticated approaches), and try a range of values for both the size $\epsilon$ and number $N$ of leapfrog steps ($\{0.1, 0.2, 0.5, 1, 2\}$ and $\{1, 2, 5, 10\}$ respectively). We evaluate these for three different problems, corresponding to simulating the hyperbolic SDE on intervals with length $T$ equal to 10, 20 and 50. For each combination of $\epsilon, N$ and $T$, we produced 10000 samples from our MCMC sampler.

To evaluate sampler performance, we calculate the effective sample size (ESS) of the SDE trajectory evaluated at $T/2$, the midpoint of the simulation interval. ESS estimates the number of independent samples that the MCMC output is equivalent to, and we calculated this using the R package rcdoa (Plummer et al. 2006). To account for the different algorithms and settings having different computational cost, we divide ESS by the compute time, yielding effective sample size per second (ESS/s) as our metric of sampler efficiency.

From left to right, the three panels of figure 2 fix $T$ to 10, 20 and 50, and plot ESS/s for
different settings of $N$ and $\epsilon$. We see that for all three settings of $T$, our sampler performs best when the stepsize $\epsilon$ equals to 0.2 and number of steps $N$ equals to 10. This configuration also performs adequately for other SDEs that we consider, moreover the algorithm does not show strong sensitivity to the choice of these tuning parameters. We therefore use these settings for the rest of our experiments. We note that these are fairly standard default settings.

**Prior simulation:** Using these HMC parameters, we next compare the efficiency of our sampler with the two baselines, a simple approximation based on Euler-Maruyama discretization, and the exact EA1 rejection sampler. We used each method to simulate 10000 trajectories from the hyperbolic bridge of equation (16). For each setting, we carried out 10 repetitions to produce error bars, and plot ESS/s at $T/2$ against interval length $T$ for the different algorithms in Figure 3.

Unsurprisingly, we observe that the Euler-Maruyama approximation with the coarsest time-discretization of 0.1 is the most efficient algorithm computationally. Note though that this is an approximate algorithm. We can improve its accuracy by using a finer grid, a typical setting being a grid with resolution 0.01. Interestingly, for this setting, even after correcting for the dependent samples, our MCMC algorithm is more efficient that Euler-Maruyama, with the Poisson grid allowing much fewer evaluations of the SDE trajectory. Additionally, the gap between our sampler and the 0.1-grid Euler-Maruyama sampler reduces with $T$ through a combination of two factors: faster run-times and reduced dependency between MCMC samples. All algorithms were significantly more efficient that the exact EA1 algorithm, and for interval lengths greater than 20, the acceptance rates (which decay exponentially with interval length) became too small to produce samples in a reasonable amount of time. As mentioned in Section 3.1, it is possible to
reduce rejection rates by breaking the interval into smaller segments. However, noting the poor performance of the algorithm even for smaller intervals, we did not investigate this further.

**Posterior simulation:** Our main interest is in settings where we wish to simulate from the SDE posterior conditioned on noisy measurements. We consider the following setting: additive Gaussian noise with mean 0 and standard deviation 0.2, at regularly spaced times on $[0, T]$. To study performance in this scenario, we compare with the particle Markov Chain Monte Carlo (pMCMC) algorithm of Andrieu et al. (2010) described earlier. We ran this with 100 particles and discretization levels of 0.1 and 0.01 (we tried different numbers of particles, but achieved best performance with 100). For each MCMC algorithm we first varied the length of the time interval $T$, keeping the number of observations fixed at 10 (left panel). We also varied the number of observations keeping $T = 10$ (right panel). In both cases, we generated 10000 SDE paths from the posterior distribution. Each setting was repeated 10 times to produce error bars. For both setups, our method is up to two orders of magnitude more efficient than pMCMC. Further, this performance gain only increases as $T$ increases, where the increasingly long time-series both increase the run-time of pMCMC, as well as reduce the acceptance probability.

The particle MCMC sampler is an approximate MCMC sampling algorithm, building on the time-discretized Euler-Maruyama approximation. To better understand the advantage our exact MCMC algorithm affords, we evaluated the two samplers using another metric: predictive accuracy on a held-out test dataset. Here, we simulated a trajectory from the hyperbolic bridge, and generated 100 observations through additive Gaussian noise with mean 0 and standard deviation 0.2. The first seventy percent was treated as training data and the rest served as test data.

In figure 5, we plot how predictive log-likelihood on the test data evolves as we increase the
number of MCMC samples. For $n$ MCMC samples, we calculated the test log-likelihood as follows:

$$\exp(\text{Test log-likelihood}(n)) = \frac{1}{n} \sum_{i=1}^{n} \ell(X_s^{(i)}) = \frac{1}{n} \sum_{i=1}^{n} \left( \prod_{s \in S} N(y_s|X_s^{(i)}, 0.2^2) \right), \tag{18}$$

where $S$ is the set of test observation times, $X_s^{(i)}$ is the diffusion path from the $i$th MCMC sample imputed at time $s$, and $N(y|\mu, \sigma^2)$ is the probability density of $y$ under a Gaussian with mean $\mu$ and standard deviation $\sigma$. The $y$-axis in figure 5 shows this test log-likelihood, while the $x$ axis shows the run time in seconds of the different algorithms. Each curve traces the median test log-likelihood (over 10 repetitions) against run-time in seconds for the different algorithms. We plot against run-time rather than number of MCMC samples since different algorithms take different time to produce $n$ samples. Shown as shaded ribbons are 90% credible intervals.

Our proposed algorithm, to the upper-left of the plot clearly achieves best predictive performance for a given computational budget. Also shown are results for three pMCMC algorithms, with discretization steps of 0.05 (the best setting), and with 50, 100 and 200 particles. For clarity, we do not include results for discretization of 0.01, which achieved much poorer performance, and 0.1, which had slightly poorer performance. The superiority of our algorithm follows from faster run-times and the fact that it is asymptotically exact as the number of Monte Carlo samples (or as run-time) increases.

## 5.2 Example 2: Periodic Drift

For our second example, we consider an SDE with peridodic drift: $dX_t = \sin(X_t)dt + dB_t$. \tag{19}
Figure 6: ESS/s for different combinations of stepsizes $\epsilon$ and the number of leapfrog steps $N$ of the HMC sampler. From left to right, we have $T = 10, 20$ and $50$, respectively. Different symbols represent the different number of steps: ■ represents 1 step, • represents 2 steps, ▲ represents 5 steps and ♦ represents 10 steps.

The drift $\alpha(x) = \sin(x)$, so that $A(u) = \int_0^u \alpha(t)dt = 1 - \cos(u)$, and $h(u) \propto \exp(A(u) - A(x) - (u - x)^2/2T) = \exp(-\cos(u) + \cos(x) - (u - x)^2/2T)$. Now, $\sin^2(x) + \cos(x)$ lies in $[-1, 5/4]$ and we set $\phi(x) = \sin^2(x)/2 + \cos(x)/2 + 1/2$. This lies in $[0, 9/8]$, so that this SDE is of class EA1.

The periodic drift term $\alpha(\cdot)$ in this SDE presents a challenge to our MCMC methodology due to the presence of a bimodality around zero. For positive values of $X_t$ in the interval $(0, \pi)$, the drift term is also positive, and the SDE experiences a repulsive push away from 0. A similar effect, but in the opposite direction, occurs when $X_t$ lies in $(-\pi, 0)$. The symmetry of the problem means that two trajectories, $X_t$ and its negative $-X_t$ are equally likely, however the repulsion away from 0 can make it difficult for an MCMC algorithm to cross from one to the other. We can overcome this with a simple additional MCMC step: at the end of each iteration, flip the sign of the entire trajectory with probability 0.5. This approach, which exploits the symmetry of the problem, works well if we just want to sample from the SDE in equation (19). Figure 6 shows the effective sample size per second for different combinations of stepsizes $\epsilon$, number of steps $N$ and observation intervals $T$. The setting is the same as the experiment described for Figure 2 and justifies the use of the same HMC parameters for this diffusion: $N = 10$ and $\epsilon = .2$.

Figure 7 compares of our sampler with these settings with EA1 and the Euler-Maruyama approximation. The results are similar to the previous experiment: EA1 does not scale to large
Figure 7: ESS/s for different samplers against interval length $T$ for the SDE with a periodic drift function. ▲ represents our method, ● and ○ represents Euler-Maruyama method with stepsize 0.01 and 0.1 respectively and ■ represents EA1. Because of low acceptance rates, we did not run EA1 for interval lengths longer than 20.

Given noisy observations, the conditional distribution over SDE paths is no longer exactly symmetric around 0, and the simple approach of flipping the path signs will require an MH correction step. The efficacy of this will depend on the degree of asymmetry introduced by the likelihood. We instead introduce a more general and flexible tempering scheme (Swendsen and Wang, 1986; Neal, 1996) to explore the trajectory space more effectively. Specifically, we introduce an ‘inverse temperature’ parameter $c$, and define a family of SDEs indexed by $c$:

$$dX_t = c\sin(X_t)dt + dB_t, \quad c \in [0, 1].$$  \hspace{1cm} (20)

Observe that $c = 0$ sets the drift term to 0, and reduces the SDE to Brownian motion, while $c = 1$ recovers the SDE of interest. Intermediate values of $c$ interpolate between these two processes, with smaller values of $c$ having smaller repulsion away from 0, and thus being easier for MCMC exploration. It is easy to derive the EA1 sampling functions associated with an arbitrary $c$:

$$A_c(u) = c - c\cos(u), \quad \phi_c(x) = \frac{c^2\sin^2(x)}{2} + \frac{c\cos(x)}{2} + \frac{c}{2}, \quad M_c = \max(\phi_c(x)) = \frac{c^2 + c}{2} + \frac{1}{8}. \hspace{1cm} (21)$$

Our parallel tempering scheme picks a set of values for $c$, spanning the interval $[0, 1]$ and including 1. We focus here on using six values, \{0, .2, .4, .6, .8, 1\}. Our approach is then to develop an MCMC sampler which targets a joint distribution over six independent trajectories, each marginally distributed according to equation (20) for one of the settings of $c$. The target distribution is thus a product distribution over the individual SDEs for each $c$. A simple MCMC step that targets this uses our Gibbs sampler to update each of the paths independently. Equation (21) includes the terms needed for this. This by itself does not solve the problem of poor mixing. However as
Figure 8: ESS/s against interval length (left) and number of observations (right). ▲ is our Gibbs sampler with 5 auxiliary tempered chains, and ● and ■ represents pMCMC with 100 particles and discretization of 0.10 and 0.01. Not included is Gibbs without tempering, since mixing was not always adequate (see main text).

mentioned earlier, we expect samplers corresponding to small c’s to explore the trajectory space better. We exploit this to improve mixing for larger c’s, and thus for our SDE of interest, with c = 1. In particular, we intersperse the previous trajectory-wise update steps with a swap proposal that uniformly picks two neighboring c’s, and proposes to exchange their associated MCMC states. In other words, for a chosen pair i and j, with inverse-temperatures, c(i) and c(j), we propose swapping the associated skeletons (Ψ(i),X(i)) and (Ψ(j),X(j)).

Write P_c(Ψ, X_Ψ) for the probability of the skeleton (Ψ, X_Ψ) under the measure e_Q^c corresponding to inverse temperature c. This is just the product of equation (14) with the probability of Ψ under a rate M_c Poisson process, with all terms given in equation (21). Then the swap proposal is accepted with Metropolis-Hastings probability given by

\[
\text{acc} = \min \left( 1, \frac{P_c(\Psi^{(j)}, X_{\Psi^{(j)}})}{P_c(\Psi^{(i)}, X_{\Psi^{(i)}})} \cdot \frac{P_c(\Psi^{(i)}, X_{\Psi^{(i)}})}{P_c(\Psi^{(j)}, X_{\Psi^{(j)}})} \right).
\] (22)

Having a larger number of c’s will mean that the SDEs corresponding to two adjacent c’s will be similar, increasing the probability of acceptance. Of course, this comes at the price of more computation. Our choice of 6 values (and thus 5 auxiliary tempered chains) was made without too much care (but see figure 9), and it is possible to be more systematic doing this. Nevertheless, our experiments demonstrate that this is already effective.

Figure 8 plots effective sample sizes per second of X_{T/2} for the task of posterior simulation
given equally spaced noisy observations of the SDE trajectory, each observation having mean equal to the trajectory value, and standard deviation equal to 0.2. In the left panel, we keep the number of observations fixed at 10 as we vary the interval length $T$, while in the right panel, we vary the number of observations with $T = 10$. Plotted with triangles is of our tempered Gibbs sampler with 5 auxiliary chains. Also included is particle MCMC, again with 100 particles, and with discretization levels of 0.1 and 0.01. We see that other than for the shortest interval length, our sampler significantly outperforms particle MCMC. Thus, the improved mixing of the tempering scheme outweights the added computational overhead, allowing it to outperform particle MCMC over a range of settings. We do not include our Gibbs sampler without tempering, since this sampler sometimes failed to mix adequately, and sometimes resulted in ESS/s scores that were deceptively good. In particular, running a Kolmogorov-Smirnov two-sample test on outputs from different runs of this MCMC algorithm sometimes rejected the null that both come from the same distribution. This never occurred for samples produced using tempering. MCMC samples from the tempered sampler also agreed with samples from particle MCMC with a fine gridsize of 0.01.

Figure 9: Predictive test log-likelihood of the different methods against run-time. Red represents our original sampler without tempering. Yellow, brown, purple and blue represent 4, 5, 10 and 50 auxiliary tempered chains. For clarity, we do not include pMCMC results, which were significantly worse than all the displayed results.

We next evaluated the samplers by measuring predictive performance on a held-out test dataset. Here, we first generated a diffusion path from the SDE over the interval $[0, 10]$, and generated 100 observations as before. We held out the last 30 observations to form the test dataset. Figure [9] shows the predictive test-likelihood from the different samplers against run-time. Here, we considered four tempering schemes, with 4, 5, 10 and 50 auxiliary tempered chains. Among these, the settings with 4 and 5 tempered distributions performed best. With more tempered chains, mixing improved only slightly, and did not compensate the increased computational burden. We
also see that while the best tempering schemes improve predictive performance slightly over our original approach without tempering, the difference is no longer significant. In other words, for this task of prediction, the fact that the simple MCMC approach did not thoroughly explore all posterior modes did not significantly hurt predictive performance. Particle MCMC performed worse than all the samplers displayed here, and for clarity, we did not include it in the plot.

6 Modeling stock prices

In our final experiment, we consider a real dataset of stock prices of Alphabet Inc\(^1\) selecting one observation each week from April 2013 to Aug 2017. The resulting dataset consists of 179 observations, of which we used the first 146 as training, and the last 33 as test. We plot the data in the left panel of Figure 10. As is typical in such applications, we preprocess the data by removing the linear trend present in the data, and then calculate the logarithm of the detrended stock price. Write \( S_t \) for the transformed measurement at time \( t \). For \( n \) trading days \( O = \{ o_1, o_2, \ldots, o_n \} \), our observations are \( S = \{ s_{o_1}, s_{o_2}, \ldots, s_{o_n} \} \). The right panel in Figure 10 plots this transformed data.

![Figure 10: Weekly stock prices for Alphabet Inc, from April 2013 to April 2018. The left panel shows the raw data, and the right one shows the transformed data which we model.](https://finance.yahoo.com/quote/GOOG?p=GOOG&.tsrc=fin-srch)

While stock prices have classically been modeled by geometric Brownian motion (Black and Scholes, 1973), limitations of such models, such as their inability to capture empirically observed heavy tails, has been well documented. In Bibby and Sørensen (1996), the authors recommend a hyperbolic distribution to model the increments of the process, and we use the hyperbolic diffusion of equation (16) from section 5. We treat this as a latent stochastic process underlying the observed stock prices \( S \). The observations themselves are modeled as additive Gaussian perturbations of

\(^1\)Obtained from [https://finance.yahoo.com/quote/GOOG?p=GOOG&.tsrc=fin-srch](https://finance.yahoo.com/quote/GOOG?p=GOOG&.tsrc=fin-srch)
an underlying hyperbolic bridge $X_t$. Our overall model is given below:

$$X_0 \sim \pi, \quad dX_t = -\frac{X_t}{\sqrt{1 + X_t^2}} dt + dB_t, \quad s_t \sim \mathcal{N}(X_t, \sigma^2), \quad t \in \{o_1, \ldots, o_n\}. \quad (23)$$

For simplicity, we fix the standard deviation of the measurement noise to 0.2, though we could easily place a conjugate inverse-Gamma prior on this parameter. We set $\pi$ to $\mathcal{N}(0, 1)$. We evaluate our MCMC algorithm on the data with the above model. Figure 11 shows the traceplot and the autocorrelation function of the trajectory value at the midpoint of the interval, $X_{T/2}$ for the Gibbs sampler without tempering. Clearly, our sampler mixes well, with no autocorrelations at large lags. This suggests that for this SDE and for this dataset, as in section 5.1, our sampler without tempering faces no real mixing issues.

The left panel in Figure 12 plots the posterior distribution over diffusion paths, including the median and a 90% confidence interval. We see that this spread includes most observations, suggesting that our model, viz. the hyperbolic diffusion with Gaussian noise produces a good fit for this dataset. The prediction includes an increase in uncertainty towards the end of the interval, where fewer observations are present due to the construction of the test dataset. For completeness, we compare two settings of our Gibbs sampler, without and with parallel tempering. We compare
both of these with pMCMC with a discretization of 0.05 with 50, 100 and 200 particles. As before, pMCMC with a gridsize of 0.1 has similar performance, while a gridsize of 0.01 performs considerably worse, and we do not include either of these results here. The plot on the right shows the predictive test log-likelihood against run time for the different methods. We see that our method without tempering (shown in red) achieves the best prediction in the shortest time. Including tempering results in a slight, but barely noticeable drop in performance (the brown curve). By contrast, all pMCMC settings achieve significantly poorer performance for the same computational budget.

7 Discussion

In this paper, we proposed a computationally efficient auxiliary variable Gibbs sampling algorithm that allows simulation from the EA1 class of SDEs without any discretization error. Our sampler builds on the EA1 rejection sampling algorithm for diffusions, described in Beskos and Roberts (2005) and follow-up work. Our method allows both prior simulation from the SDE, as well as conditional simulation given noisy observations. Our algorithm runs on an augmented space that includes a set of random Poisson times, the diffusion evaluated on these times as well as at the observation times. Conditioned on the Poisson times, we show how standard techniques like Hamiltonian Monte Carlo can be used to update the trajectory. We also show that it is easy to update the set of times given the trajectory values. We apply the resulting Gibbs sampler to synthetic and real datasets, showing improvement over competing methods.

There are a number of avenues for future research. In follow-up work, Beskos and collaborators developed exact rejection sampling algorithms for larger classes of SDEs. Recall that the EA1 class
is limited to SDEs where \( \phi(\cdot) = \alpha^2(\cdot) + \alpha'(\cdot) \in [L, M + L] \) where \( L \) and \( M \) are finite. In Beskos et al. (2006b), the authors also consider two broader classes, EA2 where \( \phi(\cdot) \) is bounded only from one side, and EA3, where it is not bounded at all. Extending our MCMC scheme to such situations is conceptually straightforward: we need to augment our MCMC state-space to include the maximum and/or minimum of the trajectory. Conditioned on these, imputing the SDE trajectory is a bit more involved than simulating from a Brownian bridge, and will involve simulating from Bessel processes. We are currently exploring efficient ways to do this. Our paper also focused only on path inference, and extending it to parameter inference, following Beskos et al. (2006b) is another important direction. Work in Giesecke and Smelov (2013); Pollock et al. (2016) has extended ideas from Beskos and Roberts (2005) to other stochastic processes, such as jump-diffusions processes, and similar ideas to this paper can be applied in that context. Finally, it is interesting to better understand theoretically the convergence properties of our proposed MCMC algorithm.

References

Adams, R. P., Murray, I., and MacKay, D. J. C. (2009). Tractable nonparametric Bayesian inference in Poisson processes with Gaussian process intensities. In Proceedings of the 26th International Conference on Machine Learning (ICML).

Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle Markov chain Monte Carlo methods. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 72(3):269–342.

Barndorff-Nielsen, O. (1978). Hyperbolic distributions and distributions on hyperbolae. Scandinavian Journal of statistics, pages 151–157.

Bergstrom, A. R. (1990). Continuous time econometric modelling. Oxford University Press.

Beskos, A., Papaspiliopoulos, O., Roberts, G. O., et al. (2006a). Retrospective exact simulation of diffusion sample paths with applications. Bernoulli, 12(6):1077–1098.

Beskos, A., Papaspiliopoulos, O., Roberts, G. O., and Fearnhead, P. (2006b). Exact and computationally efficient likelihood-based estimation for discretely observed diffusion processes (with discussion). Journal of the Royal Statistical Society: Series B, 68(3):333–382.
Beskos, A., Pinski, F. J., Sanz-Serna, J. M., and Stuart, A. M. (2011). Hybrid Monte Carlo on Hilbert spaces. *Stochastic Processes and their Applications*, 121(10):2201–2230.

Beskos, A. and Roberts, G. O. (2005). Exact simulation of diffusions. *The Annals of Applied Probability*, 15(4):2422–2444.

Bibby, B. M. and Sørensen, M. (1996). A hyperbolic diffusion model for stock prices. *Finance and Stochastics*, 1(1):25–41.

Black, F. and Scholes, M. (1973). The pricing of options and corporate liabilities. *Journal of political economy*, 81(3):637–654.

Cobb, L. (1981). Stochastic differential equations for the social sciences. *Mathematical frontiers of the social and policy sciences*, pages 37–68.

Ditlevsen, P. D., Ditlevsen, S., and Andersen, K. K. (2002). The fast climate fluctuations during the stadial and interstadial climate states. *Annals of Glaciology*, 35:457–462.

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987). Hybrid Monte Carlo. *Physics letters B*, 195(2):216–222.

Fearnhead, P., Papaspiliopoulos, O., and Roberts, G. O. (2008). Particle filters for partially observed diffusions. *Journal of the Royal Statistical Society: Series B*, 70(4):755–777.

Franks, J., Jasra, A., Law, K., and Vihola, M. (2018). Unbiased inference for discretely observed hidden Markov model diffusions. *arXiv preprint arXiv:1807.10259*.

Giesecke, K. and Smelov, D. (2013). Exact sampling of jump diffusions. *Operations Research*, 61(4):894–907.

Giles, M. B. (2008). Multilevel Monte Carlo path simulation. *Operations Research*, 56(3):607–617.

Hanes, D. P. and Schall, J. D. (1996). Neural control of voluntary movement initiation. *Science*, 274(5286):427–430.

Holden, A. V. (1976). Models of the stochastic activity of neural aggregates. In *Models of the Stochastic Activity of Neurones*, pages 286–333. Springer.
Holland, C. J. (1976). On a formula in diffusion processes in population genetics. *Proceedings of the American Mathematical Society*, 54(1):316–318.

Holmes, E. E. (2004). Beyond theory to application and evaluation: diffusion approximations for population viability analysis. *Ecological Applications*, 14(4):1272–1293.

Iacus, S. M. (2009). *Simulation and inference for stochastic differential equations: with R examples*. Springer Science & Business Media.

Karlin, S. and Tavaré, S. (1983). A class of diffusion processes with killing arising in population genetics. *SIAM Journal on Applied Mathematics*, 43(1):31–41.

Keller, J., McLaughlin, D., and Papanicolaou, G. (1995). Diffusion in random media. In *Surveys in applied mathematics vol. I*. Plenum Press, New York.

Kingman, J. F. C. (1992). *Poisson processes*, volume 3. Clarendon Press.

Kloeden, P. E., Platen, E., and Schurz, H. (2012). *Numerical solution of SDE through computer experiments*. Springer Science & Business Media.

Lange, K. (2003). *Mathematical and statistical methods for genetic analysis*. Springer Science & Business Media.

Lewis, P. A. W. and Shedler, G. S. (1979). Simulation of nonhomogeneous Poisson processes with degree-two exponential polynomial rate function. *Operations Research*, 27(5):1026–1040.

Liu, J. S. (1994). The fraction of missing information and convergence rate for data augmentation. *Computing Science and Statistics*, pages 490–490.

Meyn, S. and Tweedie, R. L. (2009). *Markov Chains and Stochastic Stability*. Cambridge University Press, New York, NY, USA, 2nd edition.

Møller, J. K. and Madsen, H. (2010). *From state dependent diffusion to constant diffusion in stochastic differential equations by the Lamperti transform*. DTU Informatics.

Neal, R. M. (1996). Sampling from multimodal distributions using tempered transitions. *Statistics and computing*, 6(4):353–366.
Neal, R. M. (2011). MCMC using Hamiltonian dynamics. Handbook of Markov Chain Monte Carlo, 2:113–162.

Neal, R. M., Beal, M. J., and Roweis, S. T. (2004). Inferring state sequences for non-linear systems with embedded hidden Markov models. In Advances in neural information processing systems, pages 401–408.

Øksendal, B. (2003). Stochastic differential equations: An introduction with applications.

Papaspiliopoulos, O., Roberts, G. O., and Sköld, M. (2007). A general framework for the parametrization of hierarchical models. Statistical Science, pages 59–73.

Plummer, M., Best, N., Cowles, K., and Vines, K. (2006). CODA: convergence diagnosis and output analysis for MCMC. R news, 6(1):7–11.

Pollock, M., Johansen, A. M., and Roberts, G. O. (2016). On the exact and ε-strong simulation of jump diffusions. Bernoulli, 22(2):794–856.

Rao, V. and Teh, Y. W. (2011). Gaussian process modulated renewal processes. In Advances in Neural Information Processing Systems 23.

Rao, V. and Teh, Y. W. (2013). Fast MCMC sampling for Markov jump processes and extensions. Journal of Machine Learning Research.

Rhee, C.-h. and Glynn, P. W. (2015). Unbiased estimation with square root convergence for SDE models. Operations Research, 63(5):1026–1043.

Ricciardi, L. M. (2013). Diffusion processes and related topics in biology, volume 14. Springer Science & Business Media.

Robert, C. P. and Casella, G. (2005). Monte Carlo Statistical Methods (Springer Texts in Statistics). Springer-Verlag, Berlin, Heidelberg.

Roberts, G. O. and Stramer, O. (2001). On inference for partially observed nonlinear diffusion models using the metropolis–hastings algorithm. Biometrika, 88(3):603–621.

Schuecker, P., Böhringer, H., Arzner, K., and Reiprich, T. (2001). Cosmic mass functions from Gaussian stochastic diffusion processes. Astronomy & Astrophysics, 370(3):715–728.
Shreve, S. E. (2004). *Stochastic calculus for finance II: Continuous-time models*, volume 11. Springer Science & Business Media.

Swendsen, R. H. and Wang, J.-S. (1986). Replica Monte Carlo simulation of spin-glasses. *Physical review letters*, 57(21):2607.

Titsias, M. K., Lawrence, N., and Rattray, M. (2008). Markov chain Monte Carlo algorithms for Gaussian processes. *Inference and Estimation in Probabilistic Time-Series Models*, 9.

Tuerlinckx, F., Maris, E., Ratcliff, R., and De Boeck, P. (2001). A comparison of four methods for simulating the diffusion process. *Behavior Research Methods, Instruments, & Computers*, 33(4):443–456.

Uhlenbeck, G. E. and Ornstein, L. S. (1930). On the theory of the Brownian motion. *Physical review*, 36(5):823.

Williams, C. K. and Rasmussen, C. E. (2006). *Gaussian processes for machine learning*, volume 2. MIT Press Cambridge, MA.

Yu, Y. and Meng, X.-L. (2011). To center or not to center: That is not the questionan Ancillarity–Sufficiency Interweaving Strategy (ASIS) for boosting MCMC efficiency. *Journal of Computational and Graphical Statistics*, 20(3):531–570.
8 Appendix

Algorithm 3 Euler-Maruyama algorithm (Kloeden et al., 2012) to simulate a diffusion process

**Input:** A regular grid \( G = \{0, t_1, t_2, \cdots, t_{n-1}, T\} \) on a time interval \([0, T]\).

An initial distribution over states \( \pi \), a drift term \( \alpha(\cdot) \) and a diffusion term \( \beta(\cdot) \).

**Output:** A diffusion trajectory \( \{X_0, X_{t_1}, X_{t_2}, \cdots, X_T\} \) evaluated on \( G \).

1: Simulate \( X_0 \sim \pi \)
2: for \( i \) in 1 to \( N \) do
3: Simulate \( y_i \) from the standard normal distribution.
4: Set \( X_{t_i+1} \leftarrow X_{t_i} + \alpha(X_{t_i})(t_{i+1} - t_i) + \beta(X_{t_i})\sqrt{t_{i+1} - t_i}y_i \)
5: end for

8.1 A particle MCMC algorithm for path inference

We first describe a particle filtering algorithm to propose a new path \( X^* \)

Algorithm 4 Particle filtering algorithm to simulate a diffusion process

**Input:** A regular grid \( G = \{0, t_1, t_2, \cdots, T\} \) on a time interval \([0, T]\),

An initial distribution over states \( \pi \), a drift term \( \alpha(\cdot) \),

Observations at times \( O = \{o_1, \ldots, o_{|O|}\} \), with observation \( i \) having likelihood \( \ell_i(X_{o_i}) \)

**Output:** A new trajectory \( X^*_G \) from the SDE conditioned on the observations.

1: Sample initial states for \( N \) particles \( X^k(0) \) from \( \pi \), \( k = 1, \ldots, N \).
2: for \( i \) in 1 to \( |O| \) do
3: (a) For \( k = 1, 2, \ldots, N \), update particle \( k \) from \([0, o_{i-1}] \) to \([0, o_i]\) by forward simulating via the Euler-Maruyama algorithm on the grid.
4: (b) Calculate the weights \( w^k_i = \ell_i(X^k_{o_i}) \) and normalize \( W^k_i = \frac{w^k_i}{\sum_{k=1}^N w^k_i} \), \( k = 1, 2, \ldots, N \).
5: (c) Sample \( J^k_i \sim \text{Multi}(\cdot|\{W^1_i, \ldots, W^N_i\}) \),\( k = 1, 2, \ldots, N \).
6: (d) Set \( X^k_{[0,o_i]} := X^i_{J^k_i [0,o_i]} \), \( k = 1, 2, \ldots, N \).
7: end for

Assume no observations at the end-time \( T \). Then uniformly pick one of the \( N \) particles, call this \( X^* \). We have an estimate of \( P_t(X^*) \), the conditional probability of \( X^* \) given the observations:
\[
P_t(X^*) = \prod_{i=1}^n \left[ \frac{\sum_{k=1}^N \frac{1}{N} w^k_i}{\sum_{k=1}^N \frac{1}{N} w^k_i} \right].
\]
8.1.1 Particle MCMC algorithm for diffusions

**Algorithm 5** The particle MCMC algorithm for SDE trajectories

**Input:**
- A regular grid \( G = \{0, t_1, t_2, \ldots, T\} \) on a time interval \([0, T]\),
- An initial distribution over states \( \pi \), a drift term \( \alpha(\cdot) \),
- Observations at times \( O = \{o_1, \ldots, o_{|O|}\} \), with observation \( i \) having likelihood \( \ell_i(X_{o_i}) \)
- Current trajectory \( X_G \) from the SDE, and current estimate of probability \( P(X_G|O) \).

**Output:** A new trajectory \( X_G^* \) from the SDE, and new estimate \( P(X_G^*|O) \).

1: Run the particle filtering algorithm above to generate a sample \( X_G^* \) along with the estimate \( P_t(X_G^*) \).
2: Accept \( X_G^* \) with probability

\[
\text{acc} = 1 \wedge \frac{P_t(X_G^*)}{P_t(X_G)}.
\]

8.2 Details of Hamiltonian Monte Carlo updates

The Hamiltonian Monte Carlo ([Duane et al., 1987](#) [Neal, 2011](#)) sampling algorithm defines a Hamiltonian function, using the target distribution as the potential energy term, and introducing a kinetic energy term parameterized by a set of auxiliary momentum variables. The algorithm proceeds by updating the variables of interest (‘position’) as well as the momentum variables according to the Hamiltonian dynamics, keeping the Hamiltonian approximately constant. In particular, if we want to sample from a distribution \( L(q) \), first, define \( U(q) = -\log(L(q)) \) to be the potential energy of position \( q \). Then introduce an auxiliary variable called \( p \) of the same dimension as \( p \) and define \( K(p) = \frac{1}{2}p^T M^{-1} p \) to be the kinetic energy. Here \( M \) is a symmetric, positive-definite mass matrix, which is typically diagonal, and is often a scalar multiple of the identity matrix. The Hamiltonian is then defined as follows:

\[
H(q,p) = U(q) + K(p)
\]

In our settings, the variables of interest are the SDE path evaluated on the Poisson grid \( \Psi \), as well as the observation times \( O \): \( q \equiv X_{\Psi\cup O} \). The distribution of interest is given in equation (15), and we repeat it below:

\[
L(q) \equiv p(X_{\Psi\cup O}) \propto \pi(X_0) h_{X_0}(X_T) \mathbb{B}(X_{O\cup \Psi}|0, X_0, T, X_T) \ell(X_O) \prod_{g \in \Psi} \left(1 - \frac{\phi(X_g)}{M}\right). \tag{24}
\]
Recall that $\pi(X_0)$ is the distribution over the initial value of the diffusion, $h_{X_0}(X_T)$ is the bias term in the $h$-biased Brownian bridge, while $\ell(\cdot)$ is the likelihood term. The term $BB(X_{O\cup\Psi}|0, X_0, T, X_T)$ gives the probability of imputing values $X_{O\cup\Psi}$ on $O \cup \Psi$ under a Brownian Bridge with values $X_0$ and $X_T$ at times 0 and $T$. Writing $O \cup \Psi \equiv \{t_1, \ldots, t_S\}$, we have

$$BB(\{X_{t_1}, \ldots, X_{t_S}\}|0, X_0, T, X_T) = P(X_{t_S}|X_0, X_T) \times P(X_{t_{S-1}}|X_0, X_{t_S}) \times \cdots \times P(X_{t_1}|X_0, X_{t_2}),$$

where $P(X_{t_i}|X_0, X_{t_j}) \sim N\left(\frac{(t_j - t_i)X_0 + t_iX_{t_j}}{t_j}, \frac{(t_j - t_i)t_i}{t_j}\right)$ for any $t_j > t_i > 0$. (25)

The potential energy is the logarithm of equation (24), and factors into a summation of straightforward terms. The Brownian bridge term in particular decomposes into a sum of quadratic terms. The gradient of equation (24) with respect to $X_{O\cup\Psi}$ is thus also straightforward to calculate, allowing an easy implementation of the HMC algorithm. We refer the reader to Neal (2011) for more details, which are now completely standard.