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

August 24, 2020

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 usually an intractable problem, and typically involves time-discretization approximations. We propose an exact Markov chain Monte Carlo sampling algorithm that involves no such time-discretization error. Our sampler is applicable to the problem of prior simulation from an SDE, posterior simulation conditioned on noisy observations, as well as parameter inference given noisy observations. Our work recasts an existing rejection sampling algorithm for a class of 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. A diffusion evolves through time according to a stochastic differential equation (SDE) (Øksendal, 2003), and is a continuous-time Markov process 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 α, and a diffusion coefficient σ. 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), where the drift equals the negative of the difference between the current state and some constant μ, 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 results in the process taking value μ at time T with probability one. 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 astronomy (Schuecker et al., 2001), biology (Ricciardi, 2013), psychology (Tuerlinckx et al., 2001), ecology (Holmes, 2004), economics (Bergstrom, 1990), genetics (Lange, 2003), finance (Black and Scholes, 1973), physics (Keller et al., 1995), and political and social sciences (Cobb, 1981).

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 even more challenging. As a consequence, both prior and posterior simulation are typically carried out approximately by discretizing time, common approaches 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. (2006a,b). 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 simulation given noisy observations. Our focus is mostly on one-dimensional diffusions, although our ideas also apply to some multivariate diffusions which can be transformed to 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 conceptually easy to see how our basic idea extends to larger classes of diffusions (called EA2 and EA3), though these generalizations can be quite involved. We leave this for future work.

We organize our paper as follows. Section 2 briefly introduces stochastic differential equations and describes the exact EA1 algorithm of Beskos and Roberts (2005). Section 3 sets up the general Bayesian model for which we wish to carry out posterior inference, and describes our proposed MCMC algorithm in this broader setting. Section 4 shows how to extend our MCMC algorithm to incorporate parameter sampling. We discuss related work in section 5, while in section 6 and 7, we evaluate our, and three 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

$$\mathrm{d}X_t = \alpha_\theta(X_t)\mathrm{d}t + \sigma_\theta(X_t)\mathrm{d}B_t. \tag{1}$$

The process is driven by Brownian motion whose value at time t is B_t. The functions $\alpha_\theta(\cdot)$ and $\sigma_\theta(\cdot)$ are the *drift* and *diffusion* terms respectively, while θ represents parameters governing the system dynamics. For clarity, we drop dependencies on θ until section 4 on parameter sampling.

Informally, equation (1) implies that $\mathrm{d}X_t$, 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 $\mathrm{d}B_t$ scaled by $\sigma(X_t)$. In general X_t and B_t can be d-dimensional vectors, with $\alpha(X_t) \in \mathbb{R}^d$ and $\sigma(X_t) \in \mathbb{R}^{d\times d}$. For one-dimensional diffusions, all these are 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

$$\mathrm{d}X_t = \alpha(X_t)\mathrm{d}t + \mathrm{d}B_t. \tag{2}$$

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 $\eta(x) = \int_{-\infty}^{x} \frac{1}{\sigma(u)}\mathrm{d}u$. Now, the process $X'_t = \eta(X_t)$ is a diffusion with diffusion coefficient equal to 1 (Møller and Madsen, 2010). In higher-dimensions, the restriction to constant diffusion is more significant, since the Lamperti transform typically does not exist, and since there needs to exist a function $A(\mathbf{u})$ satisfying $\alpha(\mathbf{u}) = \nabla A(\mathbf{u})$ (Fearnhead et al., 2010). In what follows, we assume such a transformation has been applied to produce our SDE of interest.

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 Δt, with the change in the diffusion value $\Delta X_t := X_{t+\Delta t} - X_t$ approximated as

$$\Delta X_t \approx \alpha(X_t)\Delta t + \sigma(X_t)\Delta B_t, \tag{3}$$

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

$$\Delta X_t \sim N(\alpha(X_t)\Delta t, \sigma(X_t)\Delta t). \tag{4}$$

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 path simulation from general SDEs at the price of discretization error. The algorithm of Beskos and Roberts (2005) on the other hand allows *exact* simulation from a subclass of SDEs with diffusion coefficient 1. In follow-up work (Beskos et al., 2006a), 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 (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 π 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 Girsanov's theorem (Øksendal, 2003), that characterizes the diffusion process via a Radon-Nikodym derivative with respect to Brownian motion.

Write $\mathcal{C}$ for the space of continuous functions on $[0, T]$. We will refer to generic elements of this space as ω, with ω_t representing its value at time t. For paths ω with initial value x, write $\mathbb{W}_x$ and $\mathbb{Q}_x$ 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* $\frac{\mathrm{d}\mathbb{Q}_x}{\mathrm{d}\mathbb{W}_x}$ *satisfies*

$$\frac{\mathrm{d}\mathbb{Q}_x}{\mathrm{d}\mathbb{W}_x}(\omega) = \exp\left\{\int_0^T \alpha(\omega_t)\mathrm{d}\omega_t - \frac{1}{2}\int_0^T \alpha^2(\omega_t)\mathrm{d}t\right\}. \tag{5}$$

Let $A(u) = \int_0^u \alpha(t)\mathrm{d}t$, and recall the definition of a Brownian bridge: this is just a Brownian motion conditioned on its end points. For a density $h_x(u) \propto \widetilde{h}_x(u) := \exp(A(u) - (u-x)^2/2T)$, 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 $\mathbb{Z}_x$ for the law of

this process. Note that for this to be well defined, $\widetilde{h}_x$ must be normalizable, so that the integral $\int \widetilde{h}_x(u)\mathrm{d}u = \int \exp(A(u) - (u-x)^2/2T)\mathrm{d}u$ is finite. Then we have (Beskos and Roberts, 2005):

Proposition 2.2. *Let the drift function α satisfy the conditions of Girsanov's theorem and be continuously differentiable. Then*

$$\frac{\mathrm{d}\mathbb{Q}_x}{\mathrm{d}\mathbb{Z}_x}(\omega) \propto \exp\left\{-\frac{1}{2}\int_0^T \left(\alpha^2(\omega_t) + \alpha'(\omega_t)\right)\mathrm{d}t\right\}. \tag{6}$$

Proof. Write $A_t = A(\omega_t)$. By Itô's lemma (Øksendal, 2003),

$$\mathrm{d}A_t = \frac{\partial A_t}{\partial t}\mathrm{d}t + \frac{\partial A_t}{\partial \omega_t}\mathrm{d}\omega_t + \frac{1}{2}\frac{\partial^2 A_t}{\partial \omega_t^2}\mathrm{d}t = 0 + \alpha(\omega_t)\mathrm{d}\omega_t + \frac{1}{2}\alpha'(\omega_t)\mathrm{d}t. \tag{7}$$

Solving for $\int_0^T \alpha(\omega_t)\mathrm{d}\omega_t$ and substituting in equation (5), we get

$$\frac{\mathrm{d}\mathbb{Q}_x}{\mathrm{d}\mathbb{W}_x}(\omega) = \exp\left\{A(\omega_T) - A(\omega_0) - \frac{1}{2}\int_0^T \left(\alpha^2(\omega_t) + \alpha'(\omega_t)\right)\mathrm{d}t\right\}. \tag{8}$$

By definition, the measure $\mathbb{Z}_x$ is a reweighting of $\mathbb{W}_x$ by $h_x(\omega_T)$. Thus,

$$\frac{\mathrm{d}\mathbb{Q}_x}{\mathrm{d}\mathbb{Z}_x}(\omega) \propto \exp\left\{-A(\omega_0) - \frac{1}{2}\int_0^T \left(\alpha^2(\omega_t) + \alpha'(\omega_t)\right)\mathrm{d}t\right\}. \tag{9}$$

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

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 α satisfies $\frac{1}{2}\left(\alpha^2(\cdot) + \alpha'(\cdot)\right) \in [L, L+M]$ for finite L and M.*

We focus on SDEs of class EA1. Adding and subtracting L from the exponent in equation (9),

$$\frac{\mathrm{d}\mathbb{Q}_x}{\mathrm{d}\mathbb{Z}_x}(\omega) \propto \exp\left\{-\frac{1}{2}\int_0^T \left(\alpha^2(\omega_t) + \alpha'(\omega_t) - 2L\right)\mathrm{d}t\right\} := \exp\left\{-\int_0^T \phi(\omega_t)\mathrm{d}t\right\} := \rho(\omega). \tag{10}$$

For class EA1, the function $\phi(\cdot) = \frac{1}{2}(\alpha^2(\cdot) + \alpha'(\cdot) - 2L)$ 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 $\mathbb{Q}_x$: propose a path from the stochastic process $\mathbb{Z}_x$, and accept it with probability $\rho(\omega)$. Naively, this requires 1) simulating the entire path ω, and 2) transforming and integrating ω to calculate $\rho(\omega)$, both being impossible steps. The EA1 algorithm bypasses this by recognizing that $\exp\left\{-\int_0^T \phi(\omega_t)\mathrm{d}t\right\}$ 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 ω, 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-$\phi(\omega)$ Poisson process via the thinning theorem (Lewis and Shedler, 1979). It does this in three steps: simulate a Poisson process Ψ with intensity M on the interval $[0,T]$, instantiate an h-biased Brownian bridge ω_t on Ψ, 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-$\phi(\omega_t)$ 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-$\phi(\omega_t)$ 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_Ψ, the last being the values of the diffusion uncovered on the times in the Poisson set Ψ. 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 Ψ. We simulate X_G, the diffusion evaluated on G, from a Brownian bridge with endpoints X_{t_i} and $X_{t_{t+1}}$. We write this as $X_G \sim \mathrm{BB}_G(t_i, X_{t_i}, t_{i+1}, X_{t_{i+1}})$ (see the appendix for more details). 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 $\mathbb{Q}_x$ to differ significantly from the proposal $\mathbb{Z}_x$. Further, 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. Our proposed sampler aims to address both these problems, and brings sampling algorithms from the Gaussian process literature to applications with SDEs. Before describing our algorithm, we set up the general problem.

Algorithm 1 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 $G \subset [0, T]$.

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.

Calculate $A(\cdot), \phi(\cdot)$, and the constants L and M from $\alpha(\cdot)$, and set `accept` to `false`.

while `accept = false` **do** ▷ Rejection sampling

  Simulate a rate-M Poisson process $\Psi = \{t_1, t_2, \cdots, t_{|\Psi|}\}$ on $[0, T]$.

  At the start time 0, simulate the initial value X_0 of the diffusion from π.

  At the end point T, simulate X_T from $h_{X_0}(X_T) \propto \exp(A(X_T) - (X_T - X_0)^2/2T)$.

  Simulate a Brownian bridge connecting $(0, X_0)$ and (T, X_T) on the times Ψ:

$$X_\Psi \sim \mathrm{BB}_\Psi(0, X_0, T, X_T) \qquad \text{(see appendix for details).} \tag{11}$$

  For $i \in \{1, \ldots, |\Psi|\}$, simulate $u_i \sim \mathrm{Uniform}(0, 1)$. If all $u_i > \frac{\phi(X_{t_i})}{M}$, set `accept = true`.

end while

for i in $\{0, \ldots, |\Psi|\}$ **do** ▷ Impute diffusion on G

  Define $t_0 = 0, t_{|\Psi|+1} = T$ and $G_i = G \cap (t_i, t_{i+1})$. Simulate $X_{G_i} \sim \mathrm{BB}_{G_i}(t_i, X_{t_i}, t_{i+1}, X_{t_{i+1}})$.

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 $\alpha(\cdot)$, and distribution π on the initial value 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 X_O, the trajectory values at a finite set of times $O = \{o_1, \ldots, o_{|O|}\}$, so that $\ell(X) = \ell(X_O)$. Without loss of generality, we let O include 0 and T. A simple example is when we have i.i.d. additive-noise measurements $Y_O = \{y_1, \cdots, y_{|O|}\}$ at the times O, so that $\ell(X) = \prod_{o \in O} \ell_o(X_o)$ and for example, $\ell_o(X_o) = N(y_o | X_o, \sigma_Y^2)$. 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.

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

$$\frac{\mathrm{d}\mathbb{Q}_{\pi,\ell}}{\mathrm{d}\mathbb{Q}_\pi}(\omega) \propto \ell(\omega). \tag{12}$$

The EA1 algorithm, as outlined in section 2.2, can only simulate trajectories from the prior distribution $\mathbb{Q}_\pi$. In Beskos et al. (2006a, Section 6.2), the authors adapt the EA1 rejection-sampling algorithm to conditional simulation where the diffusion is exactly observed at a finite set of times. One can adapt this when the diffusion is noisily observed, repeating two steps: 1) given X_O, the diffusion imputed on the 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, update the diffusion values at the observation times O. The first step exploits the Markov property of the SDE, and runs the conditional EA1 algorithm independently for each interval $[o_i, o_{i+1}]$. The second step involves simulating each $X_o, o \in O$ from the conditional distribution resulting from a Brownian bridge prior on X_o (algorithm 1, line 10) and the likelihood $\ell_o(X_o)$.

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 (Beskos et al., 2006a, Section 4). 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. Our proposed algorithm eliminates the rejection sampling step altogether, instead allowing practitioners to use standard MCMC algorithms in a fairly straightforward fashion.

3.2 Our proposed auxiliary variable Gibbs sampler for SDEs

We describe a novel MCMC sampling algorithm that targets the posterior distribution over trajectories $\mathbb{Q}_{\pi,\ell}$ from our Bayesian model of the previous section. Note that this equals the prior $\mathbb{Q}_\pi$ when the likelihood $\ell(\cdot)$ is a constant function. To keep our notation simple, we will write $\mathbb{Q}_{\pi,\ell}$ as

$\mathbb{Q}$. Recall that $\mathbb{Z}_x$ is an h-biased Brownian bridge starting at x. In a similar manner to $\mathbb{Q}_x$, use $\mathbb{Z}_x$ to define $\mathbb{Z}_\pi$ and $\mathbb{Z}_{\pi,\ell}$. Thus, $\mathbb{Z}_\pi$ is the distribution over Brownian bridge paths, with endpoints X_0 distributed as π, and then $X_T|X_0$ distributed as h_{X_0}. Treating this as a prior over paths, $\mathbb{Z}_{\pi,\ell}$ is the posterior distribution corresponding to observations with likelihood $\ell(\cdot)$. Again, we write $\mathbb{Z}$ for $\mathbb{Z}_{\pi,\ell}$. It follows directly from equations (10) and (12) that for a path $\omega \in \mathcal{C}$,

$$\frac{\mathrm{d}\mathbb{Q}}{\mathrm{d}\mathbb{Z}}(\omega) \propto \exp\left\{-\int_0^T \phi(\omega_t)\mathrm{d}t\right\}. \tag{13}$$

Write $\mathcal{M}$ for the space of finite point process realizations on the interval $[0,T]$. Let $\mathbb{M}$ be the probability measure on $\mathcal{M}$ corresponding to a rate-1 Poisson process. Define the product measure $\mathbb{Z}^+ = \mathbb{Z} \times \mathbb{M}$. For $\Psi \in \mathcal{M}$, and recalling that M is the supremum of $\phi(\cdot)$, define the measure $\mathbb{Q}^+$ via the following Radon-Nikodym derivative with respect to $\mathbb{Z}^+$:

$$\frac{\mathrm{d}\mathbb{Q}^+}{\mathrm{d}\mathbb{Z}^+}(\omega,\Psi) = \exp(-MT)\prod_{t\in\Psi}(M-\phi(\omega_t)). \tag{14}$$

Proposition 3.1. *$\mathbb{Q}^+$ has $\mathbb{Q}$ as its marginal distribution: $\int_{\mathcal{M}} \mathrm{d}\mathbb{Q}^+(\omega,\Psi) = \mathrm{d}\mathbb{Q}(\omega)$.*

Proof. From equation (14), we have

$$\int_{\mathcal{M}} \mathrm{d}\mathbb{Q}^+(\omega,\Psi) = \int_{\mathcal{M}} \mathrm{d}\mathbb{Z}^+(\omega,\Psi)\exp(-MT)\prod_{t\in\Psi}(M-\phi(\omega_t)) = \exp(-MT)\mathrm{d}\mathbb{Z}(\omega)\mathbb{E}_{\mathbb{M}}\left[\prod_{t\in\Psi}(M-\phi(\omega_t))\right],$$

where $\mathbb{E}_{\mathbb{M}}$ is the expectation with respect to the Poisson measure $\mathbb{M}$. By Campbell's theorem (Kingman, 1992), we have $\mathbb{E}_{\mathbb{M}}\left[\prod_{t\in\Psi}(M-\phi(\omega_t))\right] = \mathbb{E}_{\mathbb{M}}\left[\exp\left\{\sum_{t\in\Psi}\log(M-\phi(\omega_t))\right\}\right] = \exp\left\{\int_0^T (M-\phi(\omega_t))\mathrm{d}t\right\}$. The result follows directly from this and equation (13). □

While our goal is to produce samples from $\mathbb{Q}$, our MCMC sampler is an auxiliary variable sampler that targets the joint distribution $\mathbb{Q}^+$. Its state-space is the SDE trajectory X as well as the random set of Poisson times Ψ. Proposition 3.1 tells us that discarding the Poisson times Ψ produces trajectories X from the desired conditional distribution $\mathbb{Q}$. Our algorithm takes a Gibbs sampling approach, and targets the distribution $\mathbb{Q}^+$ by repeating two steps: simulate Poisson times Ψ 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 point events Ψ follow an inhomogeneous Poisson process with rate $(M-\phi(X))$.*

Proof. For ω fixed to X, from equation (14), Ψ is a point process whose density with respect to $\mathbb{M}$ is proportional to $\exp(-MT)\prod_{t\in\Psi}(M-\phi(X_t))$. Write this as $\mathbb{M}_X$. For any nonnegative function g, the Laplace functional $\mathbb{E}_{\mathbb{M}_X}[\exp\{-\sum_{t\in\Psi}g(t)\}] \propto \mathbb{E}_{\mathbb{M}}\left[\exp\{-MT-\sum_{t\in\Psi}g(t)\}\prod_{t\in\Psi}(M-\phi(X_t))\right]$. From Campbell's theorem, this equals $\exp\{-MT+\int((M-\phi(X_t))e^{-g(t)}\mathrm{d}t\}$, which is proportional to the Laplace functional of a rate-$(M-\phi(X))$ Poisson process (Kingman, 1992). □

An important point to note is that conditioned on X, the distribution over Ψ does not depend on the likelihood $\ell(X)$, since the observations depend only on the path values X. Instead, they enter when we update X. Our second corollary concerns updating X given the Poisson times Ψ.

Corollary 3.3. *Conditioned on the Poisson times Ψ, the trajectory X has density with respect to $\mathbb{Z}_\pi$ 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 Ψ, from equation (14), we see that X has density with respect to $\mathbb{Z}$ proportional to $\prod_{t\in\Psi}\left(1-\frac{\phi(X_t)}{M}\right)$. The result follows from the definition of $\mathbb{Z}=\mathbb{Z}_{\pi,\ell}$. □

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

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

This corresponds to a fairly typical posterior distribution in applications involving Gaussian processes (Williams and Rasmussen, 2006). Here our prior over trajectories is the h-baised Brownian bridge $\mathbb{Z}_\pi$, and our likelihood is $\ell(X_O)\prod_{g\in\Psi}\left(1-\frac{\phi(X_g)}{M}\right)$. Consequently, after conditioning on the Poisson grid Ψ, 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 Ψ^* given $(X_{\Psi\cup O},\Psi)$, and then simulate a new set of diffusion values $X^*_{\Psi^*\cup O}$ given Ψ^*. 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 Ψ^* conditioned on $X_{\Psi \cup O}$: Corollary 3.2 shows that conditioned on the entire trajectory X, Ψ 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 Ψ and on the observation times O. To simulate the new times Ψ^*, 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 Γ from a rate-M Poisson process, and uncover X_Γ, 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 Γ, we keep each element $g \in \Gamma$ with probability $1 - \phi(X_g)/M$, else we discard it. The set of surviving elements of Γ is a realization from a rate $M - \phi(X)$ Poisson process, and forms the new times Ψ^*. Along the way, we have evaluated X_{Ψ^*}, 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 Ψ: Corollary 3.3 shows that conditioned on the Poisson grid Ψ^*, $X_{\Psi^* \cup O}$ has 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 that update $X_{\Psi^* \cup O}$ using a Markov kernel with equation (15) as stationary distribution. We carry out this update using Hamiltonian Monte Carlo (Neal, 2011). We can exploit the Markov structure of Brownian motion to calculate the log-likelihood and its gradient in linear time (see the experiments and appendix for details). HMC exploits this gradient information to efficiently explore the conditional distribution. 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 $\mathbb{Z}^+$. Its proof is immediate (see Meyn and Tweedie (2009)): the sampler has $\mathbb{Z}^+$ as its stationary distribution since the two Gibbs steps update the conditionals of $\mathbb{Z}^+$. The sampler

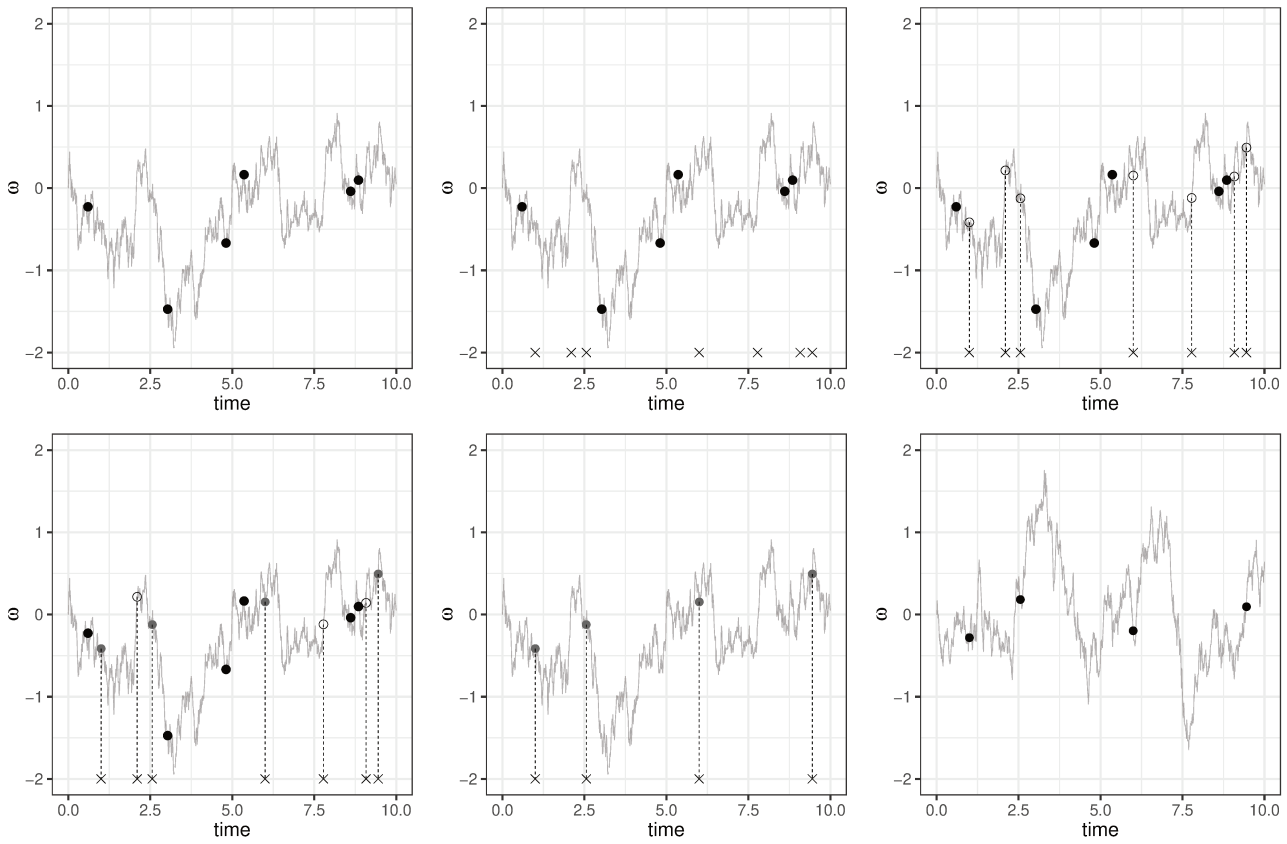


Figure 1: One iteration of our proposed Gibbs sampling algorithm. For simplicity, we do not include observations (see Algorithm 2 for the general case). From the top-left to bottom-right: 1) The iteration starts with Poisson times Ψ and the corresponding path values X_Ψ. This SDE skeleton is represented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

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

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

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

is irreducible under mild conditions on the Markov kernel used to update $X_{\Psi^* \cup O}$ given Ψ^*.

Theorem 3.4. *The Gibbs sampler described above results in a Markov chain on the state space* (Ψ, X) *with stationary distribution* $\mathbb{Z}^+(\Psi, X)$.

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

4 Parameter inference

Following Beskos et al. (2006b, Section 9), we extend our methodology to include posterior inference over the parameter θ in equation (1). Our scheme absorbs the earlier trajectory update into a

larger Gibbs sampler that also updates θ given the trajectory. Assume for the moment that there are no observations present over the interval $[0, T]$. We start with equation 8, making explicit the dependence of terms on θ. Recall $\mathbb{Q}_x$ and $\mathbb{W}_x$ are measures over paths starting at x under the SDE and Brownian motion. Let $\mathbb{W}_{x,y}$ correspond to the Brownian bridge joining $X_0 = x$ to $X_T = y$. Integrating out the path between 0 and T in equation (8) gives the transition density

$$p(X_T = y|X_0 = x,\theta) = N(y|x,T)\mathbb{E}_{\mathbb{W}_{x,y}}\left[\exp\left\{A_\theta(y) - A_\theta(x) - \frac{1}{2}\int_0^T \left(\alpha_\theta^2(X_t) + \alpha_\theta'(X_t)\right)\mathrm{d}t\right\}\right]$$
$$= N(y|x,T)\exp\left\{A_\theta(y) - A_\theta(x) - L_\theta T\right\}\mathbb{E}_{\mathbb{W}_{x,y}}\left[\exp\left\{-\int_0^T \phi_\theta(X_t)\mathrm{d}t\right\}\right]. \quad (16)$$

On the other hand, from equation (14), conditioned on its endpoints and again assuming no observations, the trajectory and Poisson events have density with respect to $\mathbb{W}_{x,y} \times \mathbb{M}$ given by

$$p(X, \Psi|X_0 = x, X_T = y, \theta) \propto \exp(-M_\theta T)\prod_{g\in\Psi}(M_\theta - \phi_\theta(X_g)) \quad (17)$$

From Campbell's theorem, the normalization constant above, obtained by integrating out X and Ψ is just the term in square brackets in equation (16). Then, multiplying equations (16) and (17),

$$p(X, \Psi|X_0 = x, \theta) = N(X_T|x,T)\exp\left\{A_\theta(X_T) - A_\theta(X_0) - (M_\theta + L_\theta)T\right\}\prod_{g\in\Psi}(M_\theta - \phi_\theta(X_g)). \quad (18)$$

Given observations at times O in an interval $[0, T]$, we break the interval into segments $[o_{i-1}, o_i]$, calculating the transition density across each segment as above. The total density is the product of these terms. With a prior $p(\theta)$ over θ, and dropping terms that do not depend on θ, we have the following expression for the posterior over θ (see also Theorem 3 in Beskos et al. (2006b)):

$$p(\theta|\psi, X) \propto p(\theta)\ell_\theta(X_O)\exp\left\{A_\theta(X_T) - A_\theta(X_0) - (M_\theta + L_\theta)T\right\}$$
$$\pi(X_0)\prod_{i=2}^{|O|} N(X_{o_i}|X_{o_{i-1}}, (o_i - o_{i-1}))\prod_{g\in\Psi}(M_\theta - \phi_\theta(X_g)) \quad (19)$$

Above, we allow the likelihood $\ell(\cdot)$ to also depend on θ. Simulating from this distribution given the skeleton (Ψ, X_Ψ) and X_O is straightforward, and we do this using a Metropolis-Hastings step. Our overall Gibbs sampler then alternates the two steps of algorithm 2 with a step to update θ. We point out that following ideas from Beskos et al. (2006b), we can use *non-centered reparametrizations* (Papaspiliopoulos et al., 2007) that reduce coupling between diffusion paths and the parameter θ. This is especially important when θ also affects the diffusion term σ: this situation requires some care, and we refer the reader to Beskos et al. (2006b) for more details.

5 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 a few approaches towards *exact* or *unbiased* estimation for diffusions to eliminate discretization error. As 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. Section 3.1 shows how this prior simulation method 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 we described, this 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 path values, reducing the coupling between the Gibbs steps. Furthermore, this extension still involves the EA1 rejection sampling algorithm, and can have high rejection rates. Controlling this requires instantiating more of the diffusion on additional grid points (Beskos et al., 2006a), which will slow down mixing. Our MCMC algorithm does not face this problem.

In Fearnhead et al. (2008), the authors propose another unbiased discretization-free algorithm, a random-weight particle filter to approximate the posterior distribution. This is a sequential Monte Carlo algorithm that targets the augmented distribution in equation (18). This algorithm is consistent as the number of particles tends to infinity, and for a finite number of particles, can be incorporated into a particle MCMC scheme, giving an MCMC algorithm that targets the posterior without any error. There have been a number of follow-up papers improving Fearnhead et al. (2008), whether by devising better proposal distributions or by developing particle smoothing

algorithms that improve effective sample sizes and allow parameter inference (Olsson and Ströjby, 2011; Gloaguen et al., 2017). Another 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, and can be used for posterior estimation (Jasra et al., 2020). These methods, along with some of the earlier particle 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 a Lamperti transformation, and thus apply to more general multi-dimensional diffusions. Our HMC based-approach is quite different from these, and an interesting line of work is to use ideas from each to improve the other.

6 Experiments

In the following, we evaluate our sampler and a number of baselines on synthetic and real datasets. Our first baseline is the EA1 rejection sampling algorithm of Beskos and Roberts (2005), we use this 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 of high rejection rates. Our second baseline is an approximate Markov chain Monte Carlo sampling algorithm, and uses Euler–Maruyama discretization to construct a particle Markov chain Monte Carlo (pMCMC) sampler. pMCMC (Andrieu et al., 2010) is a standard and relatively off-the-shelf tool to simulate from nonlinear hidden state-space models. It 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, we considered time-discretization levels of 0.1 and 0.01. Our last baseline is the unbiased random-weight particle filter of Fearnhead et al. (2010). We considered both this particle filter, as well as an exact pMCMC algorithm based on it. For both pMCMC algorithms, we considered a variety of settings for the number of particles, reporting results with 50 particles: this usually gave best performance, with run-time becoming unmanageably long with more than 200 particles. We will refer to the time-discretizated pMCMC algorithm as `EulpMCMC`, the random-weight particle filter as `FearnPF`, and the exact pMCMC algorithm as `FearnpMCMC`. All experiments were carried out on a desktop with an Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz and 16GB RAM.

6.1 Example 1: The hyperbolic bridge

Consider the hyperbolic bridge, a special case of the hyperbolic diffusion (Barndorff-Nielsen, 1978):

$$\mathrm{d}X_t = -\frac{\theta X_t}{\sqrt{1+X_t^2}}\mathrm{d}t + \sigma \mathrm{d}B_t, \quad \theta > 0. \tag{20}$$

As stated in Section 2, we fix the diffusion parameter σ to 1. When we are not updating θ, we fix it to 1. 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 $A(x) = \int_0^x \alpha(u)\mathrm{d}u = \theta - \theta\sqrt{1+x^2}$ and $\alpha'(x) = -\frac{\theta}{(1+x^2)^{3/2}}$, showing that $\frac{1}{2}(\alpha^2(x) + \alpha'(x)) = \frac{1}{2}(\frac{\theta^2 x^2}{1+x^2} - \frac{\theta}{(1+x^2)^{3/2}})$ lies in $[-\frac{\theta}{2}, \frac{\theta^2}{2}]$. We set

$$\phi(x) := \frac{1}{2}(\alpha^2(x) + \alpha'(x)) + \frac{\theta}{2} = \frac{1}{2}\left(\frac{\theta^2 x^2}{1+x^2} - \frac{\theta}{(1+x^2)^{3/2}}\right) + \frac{\theta}{2}. \tag{21}$$

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 Ψ, following equation (15). We implement a Markov kernel that targets this conditional distribution using Hamiltonian Monte Carlo (HMC) (Neal, 2011), a widely used MCMC algorithm. We provide more details of this in the appendix, at a high-level this requires computing the gradient of the log of the joint probability specified in equation (15). HMC requires tuning three parameters M, N and ϵ, 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 obtained best performance for M between 10 to 100 times the identity matrix and chose the latter (see Neal (1996); Beskos et al. (2011) for more sophisticated approaches to setting M). We tried a range of values for both the size ϵ 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 problems, corresponding to simulating the hyperbolic SDE on intervals with length T equal to $10, 20$ and 50. For each combination of ϵ, N and T, we produced 10000 samples from our sampler.

To evaluate sampler performance, we calculate the effective sample size (ESS) of $X_{T/2}$, the diffusion 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 `rcoda` (Plummer et al., 2006). To account for the different settings having different

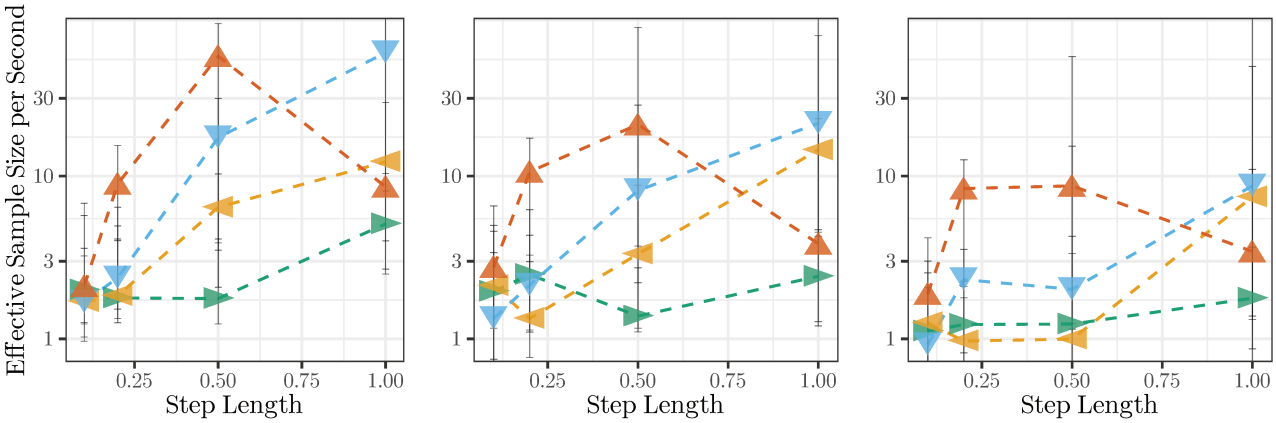


Figure 2: Effective sample size per second (ESS/s) for different settings of stepsize ϵ (x-axis), each curve being a different number of leapfrog steps N for the HMC sampler. From left to right, the three panels fix $T = 10, 20$ and 50 respectively. Different symbols represent the different values of N: ▶ represents 1 step, ◀ represents 2 steps, ▼ represents 5 steps and ▲ represents 10 steps.

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 ϵ. Based on this, we choose a fairly standard setting, with the stepsize ϵ equal to 0.2 and number of steps N equal to 5. 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.

Prior simulation: Using these HMC parameters, we compare the efficiency of our sampler with 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 diffusion. For each setting, we carried out 10 repetitions to produce error bars (not visible sometimes due to low variance), and plot ESS/s of $X_{T/2}$ against T 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

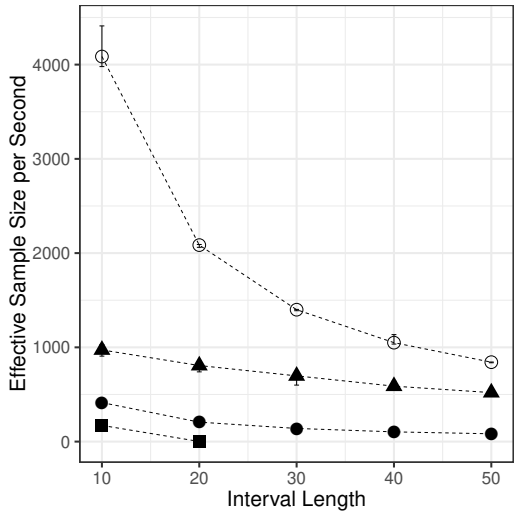


Figure 3: ESS/s against simulation interval T for the hyperbolic bridge prior. ▲ represents our Gibbs sampler, ● and ○ represents Euler–Maruyama method when stepsize equals 0.01 and 0.1 respectively, and ■ represents EA1. Due to low acceptance rates, we did not run EA1 for interval lengths longer than 20.

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 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 diffusion 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 $[0, T]$. We compare with the two particle Markov Chain Monte Carlo algorithms, `EulpMCMC` and `FearnpMCMC`, running these with 50 particles and `EulpMCMC` with a discretization level of 0.01. We considered two settings, first where we varied the length of the time interval T, keeping the number of observations fixed at 20 (figure 4, left panel), and second, where we varied the number of observations keeping $T = 20$ (figure 4, middle panel). All samplers were run for 10000 iterations, and each setting was repeated 10 times to produce error bars. For both setups, we see that our method is more than an order of magnitude more efficient than both pMCMC algorithms. This performance gain increases as T increases, where the increasingly long time-series both increase the run-time of the pMCMC algorithms, as well as reduce acceptance probabilities. As we note in section 5, it is possible to improve pMCMC performance with more careful choice of proposal distribution, though a more

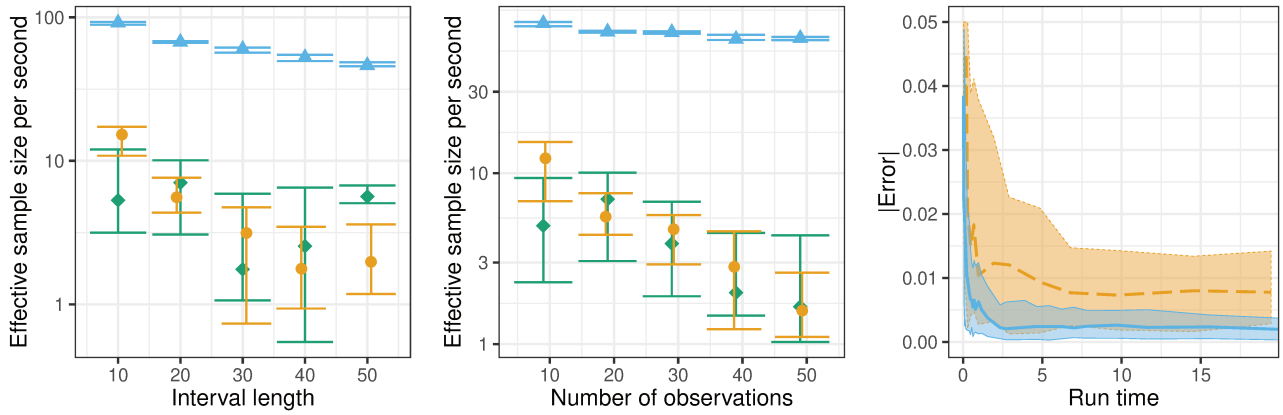


Figure 4: ESS/s of our Gibbs sampler ▲ and 50-particle pMCMC samplers `FearnpMCMC` ● and `EulpMCMC` ♦ for (left) increasing interval T with $N = 20$ observations, and (middle) increasing N with $T = 20$. The right panel shows error in filtering distributions of our method (solid line) and particle filtering `FearnPF` (dashed line) for increasing computational budget.

challenging issue is the long run-times involved with both pMCMC algorithms.

To address the last point, we note that `FearnpMCMC` is built on an unbiased particle filtering algorithm `FearnPF`. The latter forms a much cheaper baseline to compare against our MCMC sampler, with a run of 1000 particles taking slightly less time than 1000 iterations of our MCMC sampler. Unlike our MCMC sampler, the particle filter returns weighted samples, making comparison of the two algorithms trickier. To understand the speed accuracy trade-offs involved, we used both algorithms in a filtering task: we simulated a diffusion over an interval of length 20, generating 20 equally spaced Gaussian observations centered on the path, and with standard deviation 0.2. We looked at the ability of both algorithms to reconstruct statistics of the posterior distribution at the last observation: $p(X_N|y_1, \cdots, y_N)$. We considered the posterior mean, posterior variance as well as the posterior expectation of $\exp(X_N)$.

The rightmost panel in figure 4 show the results for the absolute difference between the estimated variances and the 'true' posterior variance, obtained from a long run of `EulpMCMC`. The plot shows the errors of our algorithm and `FearnPF` as computational budget (i.e. run-time) increases. The other two statistics gave similar results. For our MCMC algorithm, increasing the computational budget just involved generating more samples (up to 5000 samples in the figure), for particle filtering, this involved running the algorithm with increasing number of particles (up to 5000 particles). We can see that for the same budget, our method produces more accurate

results, converging quickly to reasonable estimates that improve with more samples. Although the random-weight particle filtering algorithm is consistent, it involves a significant amount of variance. Producing comparable errors to MCMC after many iterations will requires a large number of particles, raising issues with memory. We emphasize that particle filtering can also be improved in other ways, for example by choosing better proposal distributions, though we did not investigate this. We note though that our filtering task was designed to favor particle filtering, and a more general smoothing task would require further extensions of the basic particle filter.

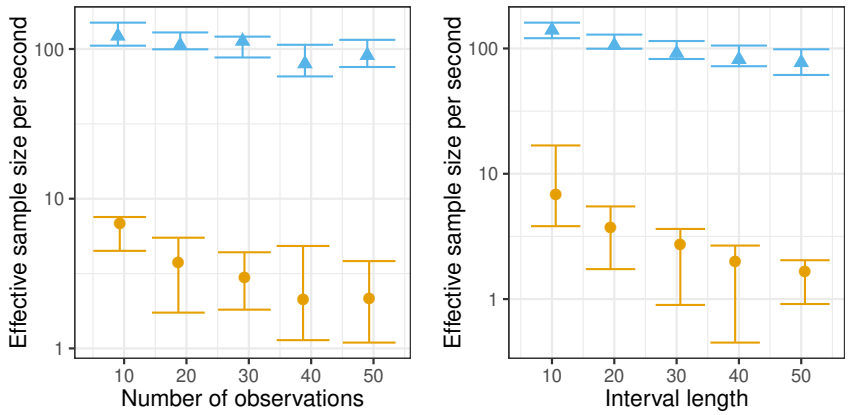


Figure 5: ESS/s for posterior samples of θ for our sampler and a PIMH `EulpMCMC` sampler, as we increase number of observations N with interval-length T fixed at 20 (left), and as T increases with $N = 20$.

In our final experiment, we place an exponential prior on the parameter θ, and look at sampling from its posterior distribution. We compare our MCMC sampler from section 6.1 with `EulpMCMC` extended to include parameter inference. This is a particle-independent Metropolis-Hastings (PIMH) sampler that proposes a new parameter θ^*, and uses `EulpMCMC` to calculate the MH acceptance probability (see appendix). Figure 5 shows the results using the prior over θ as the MH proposal distribution for both algorithms. Again our sampler is significantly more efficient than `EulpMCMC`. We expect performance to further improve if we exploit ideas from Beskos et al. (2006b) to reduce coupling between path and parameter, or jointly update path and parameter in the HMC step. The PIMH sampler on the other hand does not suffer from such coupling, and its relative performance will improve when this is a significant factor.

6.2 Example 2: Periodic Drift

Our second example considers the sine-diffusion, an SDE with periodic drift $\alpha(x) = \sin(x - \theta)$:

$$\mathrm{d}X_t = \sin(X_t - \theta)\mathrm{d}t + \mathrm{d}B_t. \tag{22}$$

Now $A(x) = \int_0^x \alpha(u)\mathrm{d}u = \cos(\theta) - \cos(x - \theta)$, and $h_x(u) \propto \exp(A(u) - A(x) - (u - x)^2/2T) = \exp(-\cos(u - \theta) + \cos(x - \theta) - (u - x)^2/2T)$. Now, $\sin^2(x) + \cos(x)$ lies in $[-1, 5/4]$ and we set

$\phi(x) = \sin^2(x-\theta)/2 + \cos(x-\theta)/2 + 1/2$. 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 potential challenge to our MCMC methodology due to the bimodality around zero when $\theta = 0$. 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 X_t and $-X_t$ 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 that exploits the problem's symmetry works well if we want prior samples from the sine-diffusion, producing similar results to figure 3 (see the appendix).

For posterior simulation given fairly informative observations, our experiments show that this bimodality in the prior presents less of a problem. In settings where it might be a problem, 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. In the appendix, we introduce a more general and flexible *tempering* scheme (Swendsen and Wang, 1986; Neal, 1996) to explore the trajectory space more effectively, but do not discuss it in the main document.

Figure 6 fixes $\theta = 0$, and plots effective sample sizes per second of posterior samples of $X_{T/2}$. Again, the samplers are given equally spaced noisy observations of the diffusion trajectory, 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 20 as we vary the interval length T, while in the middle panel, we vary the number of observations with $T = 20$. Once again, our sampler significantly outperforms the two 50-particle pMCMC baselines, `FearnpMCMC` and `EulpMCMC` (with a discretization level of 0.01). In all our experiments, we verified our samplers were exploring the posterior distribution by running a Kolmogorov-Smirnov two-sample test on outputs from different MCMC algorithms, and in all cases, the test failed to reject the null that the samples come from the same distribution. This indicates that our sampler is not stuck in a mode of the posterior because of the bimodal drift function. The rightmost panel in the figure compares our sampler with the particle filter `FearnPF` on a filtering task, and we get results similar to the earlier experiment: for the same computational cost, our MCMC sampler is more accurate with less variance.

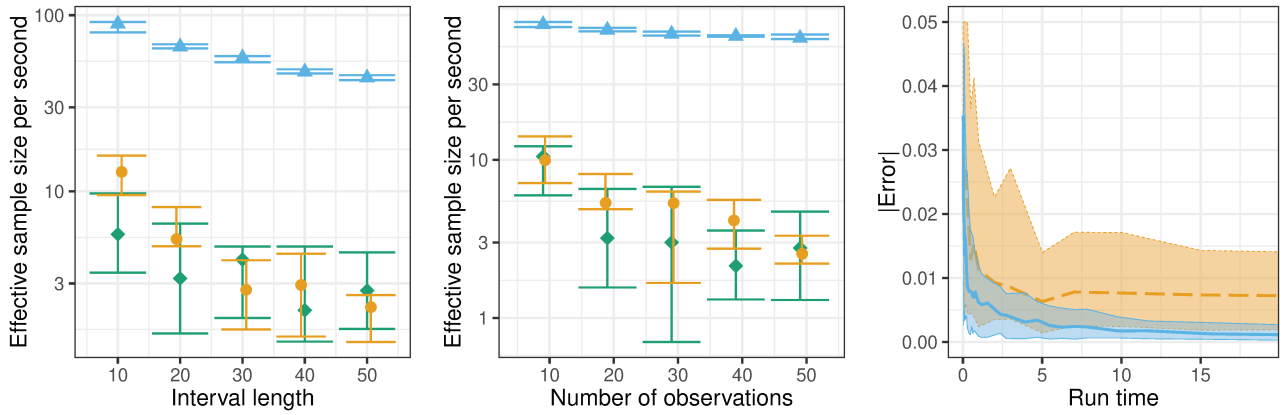


Figure 6: ESS/s of our Gibbs sampler ▲ and 50-particle `FearnpMCMC` ● and `EulpMCMC` ♦ for (left) increasing T with $N = 20$ observations, and (middle) increasing N with $T = 20$. The right panel shows error in filtering distributions of our method (solid line) and particle filtering `FearnPF` (dashed line) for increasing computational budget.

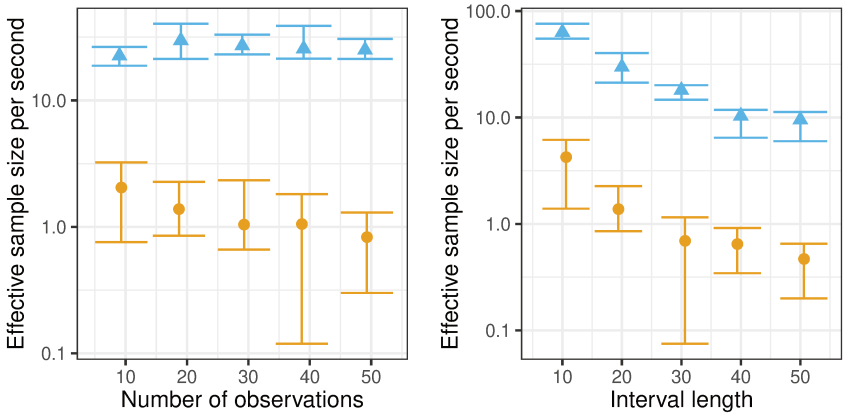


Figure 7: ESS/s for the sine diffusion as we increase the number of observations (left) and the interval-length T (right). ▲ represents our Gibbs sampler, and ● represents a particle-independent MH sampler based on an Euler–Maruyama discretization with stepsize 0.01.

The earlier experiments fixed the parameter θ to 0, in figure 7 we place of uniform Unif$(-\pi, \pi)$ on θ, and compare performance of our sampler and a PIMH sampler based on `EulpMCMC`. Once again, our sampler is about an order of magnitude more efficient.

7 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

[1]Obtained from `https://finance.yahoo.com/quote/GOOG?p=GOOG&.tsrc=fin-srch`

observations, the first 146 of which we used as the training set, and the last 33 as the test set. We plot the data in the left panel of Figure 8. As is typical, we preprocess the data, removing the linear trend, and then taking the logarithm of the detrended stock price. We also rescale time between 0 and 10. 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 8 plots this transformed data.

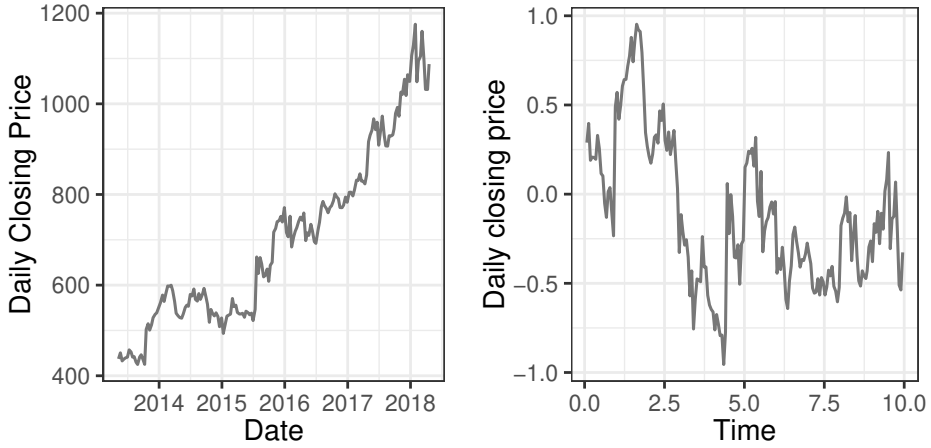


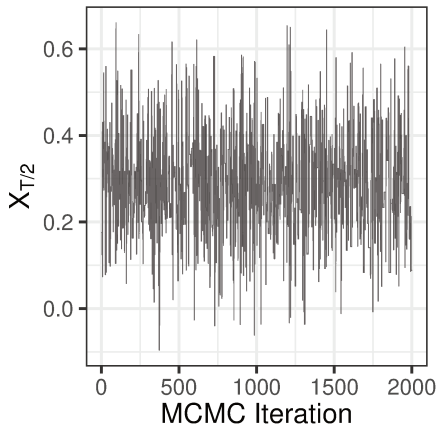
Figure 8: Weekly stock prices for Alphabet Inc, from April 2013 to April 2018. The left panel shows the raw data, and the right one shows the transformed data which we model.

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, have been well documented. Bibby and Sørensen (1996) used a hyperbolic distribution to model the increments of the process, and we use the hyperbolic diffusion of equation (20). We treat this as a latent process underlying the observed stock prices S. The observations themselves are modeled as additive Gaussian perturbations of the underlying diffusion. The overall model is

$$X_0 \sim \pi, \qquad \mathrm{d}X_t = -\frac{\theta X_t}{\sqrt{1+X_t^2}}\mathrm{d}t + \mathrm{d}B_t, \qquad s_t \sim \mathcal{N}(X_t, \sigma^2), \quad t \in \{o_1, \ldots, o_n\}. \tag{23}$$

For simplicity, we fix the standard deviation of the measurement noise to 0.2, though we could easily place a conjugate prior on this. We consider two settings, one with θ fixed to 1, and the second with a rate-1 exponential prior over θ. We apply our MCMC algorithm to the data in both settings. Figure 9 shows the MCMC traceplot and autocorrelation function of the trajectory value at $X_{T/2}$, the midpoint of the interval, for the sampler that updates θ. The results with θ fixed are similar: our sampler mixes well, with no significant autocorrelation at lags larger than 5.

The left and middle panels in figure 10 plot the posterior distribution over diffusion paths, showing the median and a 90% posterior credible interval. The left is with θ fixed to 1 and



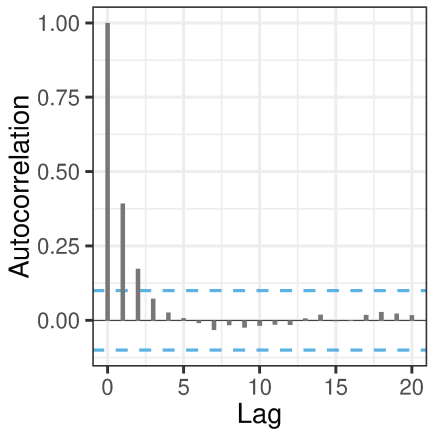
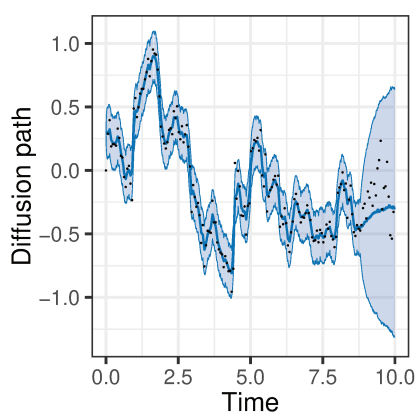
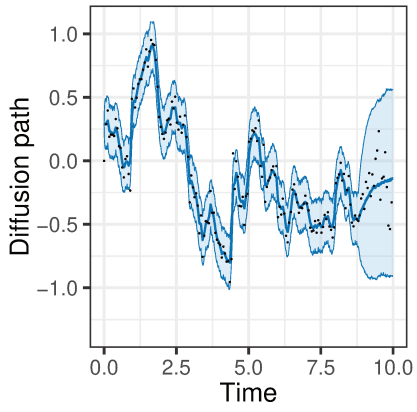


Figure 9: (left) MCMC trace of $X_{T/2}$, the diffusion value at the midpoint of the observation interval, and (right) its autocorrelation function.





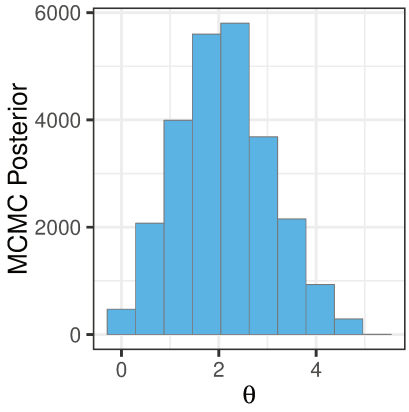
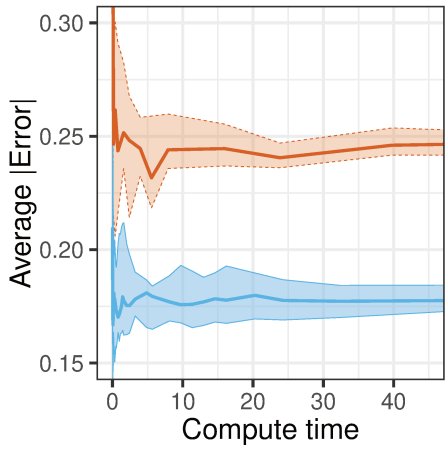


Figure 10: Median and 90% quantiles of the posterior over paths with (left) $\theta = 1$, and (center) θ updated after placing a rate-1 exponential prior. (Right) is the corresponding posterior over θ.



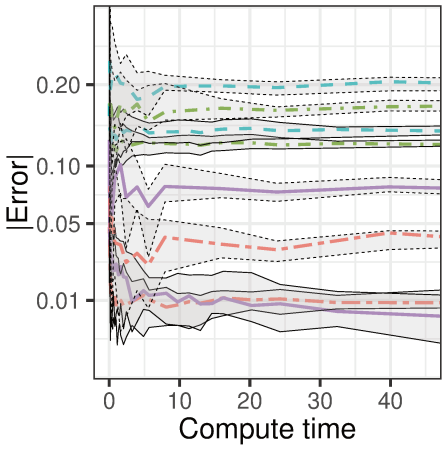


Figure 11: Absolute error (left) averaged across all test datapoints, and (right) for first 4 test datapoints (each a different line-type). Solid/dashed ribbons are our method/`FearnPF`.

the middle is when we update θ. We see that the posterior spread includes most observations, suggesting that our model, viz. the hyperbolic diffusion with Gaussian noise produces a good fit for this dataset. We note that we only fit the models on training data (until about time 8), the remaining datapoints are held-out test data that the models never see. This explains the increase in uncertainty towards the end of the interval, though both models cover the observations. Updating θ provides a tighter fit by virtue of inferring stronger zero-reverting dynamics. This is reflected in the posterior over θ (the rightmost panel) which concentrates on values larger than 1.

Our algorithm is significantly faster than the two pMCMC baselines `EulpMCMC` and `FearnpMCMC`, and we do not include speed-accuracy comparisons with these. We note though that posterior approximations produced by all 3 MCMC algorithms agreed with each other, with 2-sample Kolmogorov-Smirnov tests failing to reject the null that they come from the same distribution. Instead, we compare our algorithm with the random-weight particle filtering algorithm `FearnPF`. We set θ to 1 for both algorithms, and ran them on the training dataset. We then imputed path values on the test times, and calculated the absolute difference of the posterior predictive means at test times from the observed test values. We repeated this 10 times for each algorithm for different random seeds, plotting the errors for increasing computational budgets. As before, increasing the computational budget of our method involved running for more MCMC iterations, while for particle filtering, this involved rerunning the algorithm with more particles. The left panel in figure 11 shows the average absolute error across all 33 test datapoints: the ribbons are 90% quantiles across multiple initializations, and the thick line is the median. Our algorithm (solid ribbons) outperformed `FearnPF` (dashed ribbons), producing more accurate results for comparable runtimes. The right limit of the x-axis (at 45 seconds) corresponds to our method run for about 20000 samples, and `FearnPF` run with 5000 particles. The right panel disaggregates these results, showing the absolute error for the first 4 test datapoints. These are typical of the results we observed: our algorithm outperformed `FearnPF` on most test data points.

8 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 prior simulation from the SDE, conditional

simulation given noisy observations as well as parameter inference.

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. (2006a,b), 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, though much more involved: 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 will involve simulating from Bessel processes instead of Brownian bridges. We are currently exploring efficient ways to do this. 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, and to other diffusions not absolutely continuous with respect to Brownian motion. Finally, it is interesting to better understand theoretically the convergence properties of our proposed MCMC algorithm.

9 Supplementary material

Appendix This file includes details of the particle MCMC and Hamiltonian Monte Carlo algorithms, the tempering scheme to improve mixing, as well as experimental results not included in the main text. [`Appendix_WangRaoTeh.pdf`].

R code This includes code implementing the proposed algorithm. `README.txt` includes instructions. The github repository `https://github.com/varao/WangRaoTeh_JCGS_code` also contains the code. [`Code_WangRaoTeh.tar.gz`].

10 Acknowledgements

We thank the anonymous reviewers whose suggestions helped to significantly improve this manuscript. VR acknowledges the National Science Foundation for funding under grants RI/1816499 and DMS/1812197.

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.

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.

Fearnhead, P., Papaspiliopoulos, O., Roberts, G. O., and Stuart, A. (2010). Random-weight particle filtering of continuous time processes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):497–512.

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.

Gloaguen, P., Etienne, M.-P., and Corff, S. L. (2017). Online sequential monte carlo smoother for partially observed stochastic differential equations. *arXiv preprint arXiv:1703.01776*.

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.

Jasra, A., Law, K. J. H., and Yu, F. (2020). Unbiased filtering of a class of partially observed diffusions. *ArXiv*, abs/2002.03747.

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 *Adv. in Neural Information Processing Systems*, pages 401–408.

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

Olsson, J. and Ströjby, J. (2011). Particle-based likelihood inference in partially observed diffusion processes using generalised Poisson estimators. *Electronic Journal of Statistics*, 5:1090–1122.

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.

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.

11 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 π, 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**

11.1 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 π, 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 π, $k = 1, ..., N$.
2: **for** i in 1 to $|O|$ **do**
3: For $k = 1, 2, ..., N$, update particle k from $[0, o_{i-1}]$ to $[0, o^i]$ by forward simulating via the Euler-Maruyama algorithm on the grid.
4: Calculate the weights $w_i^k = \ell_i(X^k_{o_i})$ and normalize $W_i^k = \frac{w_i^k}{\sum_{k=1}^N w_i^k}$, $k = 1, 2, ..., N$.
5: Sample $J_i^k \sim \text{Multi}(\cdot|(W_i^1, \ldots, W_i^N))$,$k = 1, 2, ..., N$.
6: Set $X^k_{[0,o_i]} := X^{J_i^k}_{[0,o_i]}$, $k = 1, 2, ..., N$..
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 $P_\ell(X^*)$, the conditional probability of X^* given the observations:

$P_\ell(X^*) = \prod_{i=1}^n \left[\sum_{k=1}^N \frac{1}{N} w_i^k\right]$.

11.1.1 Particle MCMC algorithm for diffusions

Algorithm 5 The particle MCMC algorithm for SDE trajectories

Input: A regular grid $G = \{0, t_1, t_2, \cdots, T\}$ on a time interval $[0, T]$,
An initial distribution over states π, 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, parameter θ, and current estimate of probability $P(X_G|O)$.

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

1: Propose a parameter θ^* from some distribution $q(\theta^*|\theta)$
2: Run the particle filtering algorithm to generate a sample X_G^* along with the estimate $P_\ell(X_G^*)$.
3: Accept (θ^*, X_G^*) with probability $\texttt{acc} = 1 \wedge \frac{P_\ell(X_G^*)p(\theta^*)q(\theta|\theta^*)}{P_\ell(X_G)p(\theta)q(\theta^*|\theta)}$.

11.2 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 $H(q, p) = U(q) + K(p)$ In our settings, the variables of interest are the SDE path evaluated on the Poisson grid Ψ, 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:

$$L(q) \equiv p(X_{\Psi\cup O}) \propto \pi(X_0) h_{X_0}(X_T) \mathrm{BB}(X_{O\cup\Psi}|0, X_0, T, X_T) \ell(X_O) \prod_{g\in\Psi} \left(1 - \frac{\phi(X_g)}{M}\right). \tag{24}$$

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 $\mathrm{BB}(X_{O\cup\Psi}|0, X_0, T, X_T)$

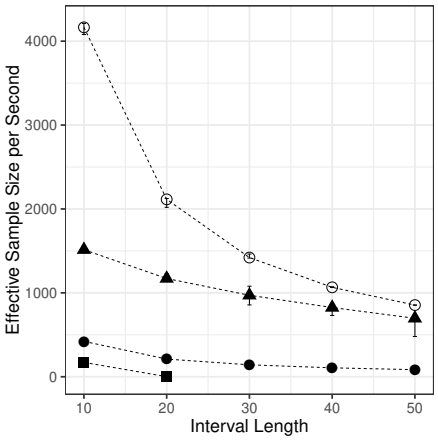


Figure 12: ESS/s for different samplers against interval length T for the SDE with a periodic drift function. ▲ represents our method, ● and ○ represents Euler-Maruyama method with stepsize 0.01 and 0.1 respectively and ■ represents EA1. Because of low acceptance rates, we did not run EA1 for interval lengths longer than 20.

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

$$\mathrm{BB}(\{X_{t_1}, \ldots, X_{t_S}\}|0, X_0, T, X_T) = P(X_{t_S}|X_0, X_T) \times P(X_{t_{S-1}}|X_0, X_{t_S}) \times \cdots \times P(X_{t_1}|X_0, X_{t_2}),$$
$$\text{where } P(X_{t_i}|X_0, X_{t_j}) \sim N\left(\frac{(t_j - t_i)X_0 + t_i X_{t_j}}{t_j}, \frac{(t_j - t_i)t_i}{t_j}\right) \text{ for any } t_j > t_i > 0. \quad (25)$$

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.

11.3 Improved mixing via tempering

Figure 12 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 T, while our asymptotically exact sampler performs between the crude and fine discretizations.

We introduce a more general and flexible *tempering* scheme (Swendsen and Wang, 1986; Neal, 1996) to explore the trajectory space more effectively. For concreteness, consider the periodic diffusion in equation (22). We introduce an 'inverse temperature' parameter c, and define a family of SDEs indexed by c:

$$\mathrm{d}X_t = c\sin(X_t)\mathrm{d}t + \mathrm{d}B_t, \quad c \in [0, 1]. \quad (26)$$

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:

$$A_c(u) = c - c\cos(u), \quad \phi_c(x) = \frac{c^2\sin^2(x)}{2} + \frac{c\cos(x)}{2} + \frac{c}{2}, \quad M_c = \max(\phi_c(x)) = \frac{c^2+c}{2} + \frac{1}{8}. \quad (27)$$

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 (26) 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 (27) includes the terms needed for this. This by itself does not solve the problem of poor mixing. However as 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 $(\Psi^{(i)}, X^{(i)}_{\Psi^{(i)}})$ and $(\Psi^{(j)}, X^{(j)}_{\Psi^{(j)}})$.

Write $P_c(\Psi, X_\Psi)$ for the probability of the skeleton (Ψ, X_Ψ) under the measure ${}_c\mathbb{Q}^+$ 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 (27). Then the swap proposal is accepted with Metropolis-Hastings probability given by

$$\text{acc} = \min\left(1, \frac{P_{c_i}(\Psi^{(j)}, X^{(j)}_{\Psi^{(j)}}) \cdot P_{c_j}(\Psi^{(i)}, X^{(i)}_{\Psi^{(i)}})}{P_{c_i}(\Psi^{(i)}, X^{(i)}_{\Psi^{(i)}}) \cdot P_{c_j}(\Psi^{(j)}, X^{(j)}_{\Psi^{(j)}})}\right). \quad (28)$$

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, and it is possible to be more systematic doing this.

11.4 Miscellaneous results

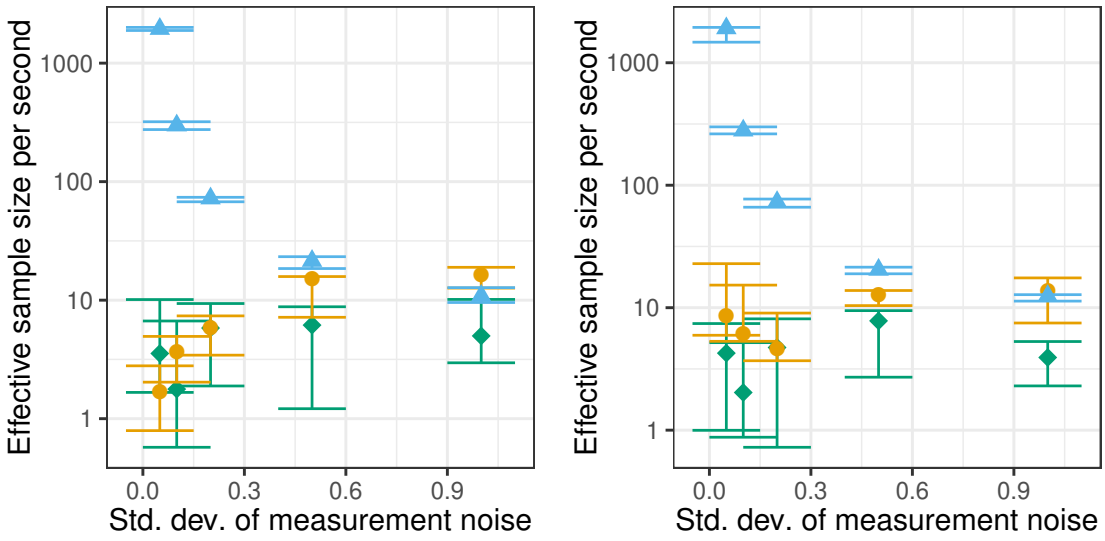


Figure 13: ESS/s of our Gibbs sampler ▲ and 50-particle pMCMC samplers `FearnpMCMC` ● and `EulpMCMC` ♦ for $T = 20$ and $N = 20$ as the standard deviation of the measurement noise increases. (Left) is for the hyperbolic diffusion, and (right) is for the periodic diffusion.