Inference for Coupled SDE: Metropolis Algorithms via Density Tracking by Quadrature

Harish S. Bhat, R. W. M. A. Madushani, and Shagun Rawat

1 Introduction

Stochastic Differential Equations (SDE) are a widely used powerful mathematical tool to model real-world phenomena. However, parameter inference of SDE models is still a very challenging problem, due to the fact that the likelihood function is generally unknown for the case where time-discrete observations are available [8, 5, 4]. Most existing parametric inference methods for discretely observed SDE require inter-observation time to be small, to track the transition density for the discrete time observations. As a way to facilitate approximation of the transition density for parametric inference for large inter-observation times, Bayesian methods are used to simulate missing values of the observations to form a high-frequency data set. In situations where the likelihood function is either analytically unavailable or computationally prohibitive to evaluate, Bayesian inference of SDE makes use of likelihood-free methods such as Approximate Bayesian Computation [6], variational methods [2, 9], and/or Gaussian processes [1, 7].

In our work we have developed a Markov Chain Monte Carlo Method (MCMC) algorithm for Bayesian inference of parameters in SDE. The MCMC algorithm is derived using a Metropolis scheme; our innovation is to evaluate the log likelihood efficiently using density tracking by quadrature (DTQ). The DTQ method applies quadrature to the Chapman-Kolmogorov equation associated with a time-discretization of the original SDE [3]. For the case of scalar SDE, the DTQ method's

Harish S. Bhat
University of California Merced, 5200 N. Lake Rd., Merced, CA, USA
e-mail: hbhat@ucmerced.edu

R. W. M. A. Madushani
University of California Merced, 5200 N. Lake Rd., Merced, CA, USA
e-mail: rmadushani@ucmerced.edu

Shagun Rawat
University of California Merced, 5200 N. Lake Rd., Merced, CA, USA
e-mail: srawat2@ucmerced.edu

density function converges to the true density of the SDE at a rate that is linear in the time step. The method we have developed is applicable to the case where inter-observation times are large and/or irregular. In this paper, we present a Metropolis algorithm for Bayesian inference of unknown parameters of a 2-D SDE.

We consider the setup where a player is running and another player is chasing the runner inside the court. We assume that the goal of the chaser is to get in the vicinity of the runner and thus the chaser is running towards the current position of the runner. The current spatial coordinates of the runner is (x^r, y^r) and those of the chaser is (x^c, y^c). Since the chaser is moving towards the runner, the velocity vector of the chaser is pointed towards the runner's current position.

Fig. 1: Paths for runner and chaser

The velocity of the chaser, $(\dot{x}^c, \dot{y}^c)$, can thus be given as

$$(\dot{x}^c, \dot{y}^c) = \gamma(t)(x^r - x^c, y^r - y^c)$$

where the velocity vector of the chaser is $\phi = (x^r - x^c, y^r - y^c)$ and the speed of the chaser is $\gamma(t)$. If we know the runner's position at different time points, the system of ODEs for the chaser is as follows,

$$\begin{aligned} \mathrm{d}(x^c, y^c) &= \gamma(t)\frac{\phi}{||\phi||}\mathrm{d}t + (\nu_1 \mathrm{d}W_t^1, \nu_2 \mathrm{d}W_t^2) \\ \implies \mathrm{d}x^c &= \frac{\gamma(t)(x^r - x^c)}{\sqrt{(x^r - x^c)^2 + (y^r - y^c)^2}}\mathrm{d}t + \nu_1 \mathrm{d}W_t^1 \\ \mathrm{d}y^c &= \frac{\gamma(t)(y^r - y^c)}{\sqrt{(x^r - x^c)^2 + (y^r - y^c)^2}}\mathrm{d}t + \nu_2 \mathrm{d}W_t^2 \end{aligned}$$

The Euler-Maruyama approximation of the above system of ODEs is

$$x_{i+1}^c = x_i^c + \frac{\gamma(t)(x_i^r - x_i^c)}{\sqrt{(x_i^r - x_i^c)^2 + (y_i^r - y_i^c)^2}}h + \nu_1\sqrt{h}Z_1$$
$$y_{i+1}^c = y_i^c + \frac{\gamma(t)(y_i^r - y_i^c)}{\sqrt{(x_i^r - x_i^c)^2 + (y_i^r - y_i^c)^2}}h + \nu_2\sqrt{h}Z_2$$

2 Derivation of the Numerical Method

Let $W_{1,t}$ and $W_{2,t}$ denote two independent Wiener processes with $W_{1,0} = W_{2,0} = 0$ almost surely. In this work, we deal with time-dependent coupled SDE of the form:

$$\mathrm{d}X_{1,t} = f_1(t, \mathbf{X}_t, \theta)\mathrm{d}t + g_1(t, \mathbf{X}_t, \theta)\mathrm{d}W_{1,t} \tag{1a}$$
$$\mathrm{d}X_{2,t} = f_2(t, \mathbf{X}_t, \theta)\mathrm{d}t + g_2(t, \mathbf{X}_t, \theta)\mathrm{d}W_{2,t}. \tag{1b}$$

Here $\mathbf{X}_t = (X_{1,t}, X_{2,t})$ is a two-dimensional stochastic process. For $j = 1, 2$, we refer to f_j and g_j as, respectively, drift and diffusion functions. Both drift and diffusion functions may depend on a parameter vector $\theta \in \mathbb{R}^N$.

Our goal is to infer the parameter vector θ from direct observations of $\mathbf{X}_t$. Suppose that at a sequence of times $0 = t_0 < t_1 < \cdots < t_M = T$, we have observations $\mathbf{x} := \{(x_{1,m}, x_{2,m})\}_{m=0}^M$. Here $\mathbf{x}_m = (x_{1,m}, x_{2,m})$ is a sample of $\mathbf{X}_{t_m}$. In this paper, we will assume equispaced temporal observations, i.e., $t_m = m\Delta t$ for fixed step size $\Delta t > 0$. We make this assumption purely for notational simplicity; the method we describe can be easily adapted for nonequispaced temporal observations. We refer to Δt as the time step of the data.

The posterior density of the parameter vector given the observations is $p(\theta \,|\, \mathbf{x}) \propto p(\mathbf{x} \,|\, \theta)p(\theta)$, where $p(\mathbf{x} \,|\, \theta)$ is the likelihood and $p(\theta)$ is the prior. We discretize the SDE (1) in time using the Euler-Maruyama scheme:

$$X_1^{n+1} = X_1^n + f_1(t_n, X_1^n, X_2^n, \theta)h + g_1(t_n, X_1^n, X_2^n, \theta)\sqrt{h}Z_1^{n+1} \tag{2a}$$
$$X_2^{n+1} = X_2^n + f_2(t_n, X_1^n, X_2^n, \theta)h + g_2(t_n, X_1^n, X_2^n, \theta)\sqrt{h}Z_2^{n+1}. \tag{2b}$$

Here $h > 0$ is a fixed time step, the time step of our numerical method. We shall choose h to be a fraction of Δt, i.e., $Fh = \Delta t$ for integer $F \geq 2$. The random variables X_i^n for $i = 1, 2$ are approximations of $X_{i,nh}$. The Z_i^n are independent and identically distributed random variables, normally distributed with mean 0 and variance 1, i.e., $Z_i^n \sim \mathcal{N}(0, 1)$.

Let $\widetilde{p}(\mathbf{x} \,|\, \theta)$ denote the likelihood under the discrete-time model (2), an approximation to the true likelihood $p(\mathbf{x} \,|\, \theta)$. Note that (2) describes a discrete-time Markov chain. By the Markov property, the likelihood $\widetilde{p}(\mathbf{x} \,|\, \theta)$ factors and we can write:

$$p(\mathbf{x} \,|\, \theta) \approx \widetilde{p}(\mathbf{x} \,|\, \theta) = \prod_{m=0}^{M-1} \widetilde{p}(\mathbf{x}_{m+1} \,|\, \mathbf{x}_m, \theta). \tag{3}$$

The term $\widetilde{p}(\mathbf{x}_{m+1} \,|\, \mathbf{x}_m, \theta)$ is the transition density for (2), from state $\mathbf{x}_m$ at time t_m to state $\mathbf{x}_{m+1}$ at time t_{m+1}. This suggests a numerical method for computing this density, which we explore in the next subsection.

2.1 Density Tracking by Quadrature (DTQ)

Equation (2) describes a Markov chain over a continuous state space. If we let $\widetilde{p}^n(x_1, x_2 \,|\, \theta)$ denote the joint probability density function of X_1^n and X_2^n given θ, then the Chapman-Kolmogorov equation associated with (2) is

$$\widetilde{p}^{n+1}(x_1, x_2, t_{n+1} \,|\, \theta) = \int_{y_1, y_2 \in \mathbb{R}^2} K^n(x_1, x_2, y_1, y_2, t_n; \theta) \widetilde{p}^n(y_1, y_2, t_n \,|\, \theta)\, dy, \tag{4}$$

where

$$\begin{aligned} &K^n(x_1, x_2, y_1, y_2, t_n; \theta) = \widetilde{p}^{n+1|n}(x_1, x_2 | y_1, y_2, t_n; \theta) \\ &= (2\pi\sigma_1^2)^{-1/2} \exp\left[-(x_1 - \mu_1)^2/(2\sigma_1^2)\right] (2\pi\sigma_2^2)^{-1/2} \exp\left[-(x_2 - \mu_2)^2/(2\sigma_2^2)\right]. \end{aligned}$$

Here $\mu_1 = y_1 + f_1(y_1, y_2, t_n; \theta)h, \mu_2 = y_2 + f_2(y_1, y_2, t_n; \theta)h, \sigma_1^2 = g_1^2(y_1, y_2, t_n; \theta)h$ and $\sigma_2^2 = g_2^2(y_1, y_2, t_n; \theta)h$. That is, $K(x_1, x_2, y_1, y_2; \theta)$ is the conditional density of X_1^{n+1} and X_2^{n+1} given $X_1^n = y_1$, $X_2^n = y_2$ and $\theta = \theta$, evaluated at the point (x_1, x_2). The fact that the conditional density is a product of normal distributions with means μ_1, μ_2 and variances σ_1^2, σ_2^2 can be shown using (2) together with the fact that X_1^{n+1} and X_2^{n+1} are conditionally independent given X_1^n and X_2^n. This conditional independence is a direct consequence of having two independent random variables Z_1^n and Z_2^n in (2).

The crux of the DTQ method is to apply quadrature to (4) to evolve an initial density forward in time. Consider a spatial grid with fixed spacing $k > 0$ and grid points $x_1^i = ik$, $x_2^j = jk$, $y_1^{i'} = i'k$, and $y_2^{j'} = j'k$. Then we apply the trapezoidal rule in both the y_1 and y_2 variables to obtain:

$$\hat{p}^{n+1}(x_1^i, x_2^j, t_{n+1}; \theta) = k^2 \sum_{i'=-\infty}^{\infty} \sum_{j'=-\infty}^{\infty} K^n(x_1^i, x_2^j, y_i^{i'}, y_2^{j'}, t_n; \theta) \hat{p}^n(y_1^{i'}, y_2^{j'}, t_n; \theta) \tag{5}$$

It is unnecessary to sum over all of $\mathbb{Z}^2$. We know that a two-dimensional Gaussian decays to zero far from its mean so we sum over a window of $2\gamma+1$. Since the mean (μ_1, μ_2) is approximately (y_1, y_2), we sum only from $y_1 = x_1 - \gamma k$ to $y_1 = x_1 + \gamma k$ and similarly for y_2, where $\gamma = \frac{2\sqrt{h}\sup(g)}{k}$:

$$\hat{p}^{n+1}(x_1^i, x_2^j, t_{n+1}; \theta) = k^2 \sum_{i'=i-\gamma}^{i+\gamma} \sum_{j'=j-\gamma}^{j+\gamma} K^n(x_1^i, x_2^j, y_i^{i'}, y_2^{j'}, t_n; \theta) \hat{p}^n(y_1^{i'}, y_2^{j'}, t_n; \theta) \tag{6}$$

We now have our method to evaluate $\widetilde{p}(\mathbf{x}_{m+1}\,|\,\mathbf{x}_m,\theta)$. Let us take $n=0$ in (6) to correspond to the time t_m. We start with the deterministic initial condition $\mathbf{X}^0=\mathbf{x}_m$, corresponding to the density $\widetilde{p}^0(\mathbf{x})=\delta(\mathbf{x}-\mathbf{x}_m)$. Inserting this point mass into (4), we obtain a Gaussian density for $\widetilde{p}^1(\mathbf{x})$. For each $i,j\in[-y_M/k,y_M/k]$, we set

$$\hat{p}^1(x_1^i,x_2^j;\theta)=\widetilde{p}^1(x_1^i,x_2^j;\theta).$$

Now that we have $\hat{p}^1$, we use (6) repeatedly to compute $\hat{p}^2$, $\hat{p}^3$, and so on until we reach $\hat{p}^F$. The object $\hat{p}^F$ is then a spatially discrete approximation of the transition density from time t_m to time $t_m+Fh=t_{m+1}$. Interpolating this density at the point $\mathbf{x}_{m+1}$, we compute a numerical approximation of $\widetilde{p}(\mathbf{x}_{m+1}\,|\,\mathbf{x}_m,\theta)$, as required for the likelihood function.

TODO: Add last interpolation step

2.2 Metropolis Algorithm

Once we have an efficient method to compute the likelihood, we can use the method in a Metropolis algorithm to sample from the posterior. In the Metropolis algorithm, we construct an auxiliary Markov chain $\{\hat{\theta}_N\}_{N\geq 0}$ which is designed to have an invariant distribution given by the posterior $p(\theta\,|\,\mathbf{x})$. This Markov chain is constructed as $\hat{\theta}_{N+1}=\hat{\theta}_N+Z_{N+1}$, where Z_{N+1} is a random vector with dimension equal to that of the parameter vector θ. In this paper, we choose all components of Z_{N+1} to be independent normal random variables with known means and variances.

The Metropolis algorithm is as follows:

- Choose value q_0 for $\hat{\theta}_0$.
- Once the values $q_0,\cdots,q_N$ of $\hat{\theta}_0,\cdots,\hat{\theta}_N$ have been found:
  - Generate a proposal value q^*_{N+1} from the auxiliary Markov chain: $q^*_{N+1}=q_N+Z_{N+1}$.
  - Calculate the ratio $\rho=\frac{p(q^*_{N+1}\,|\,\mathbf{x})}{p(q_N\,|\,\mathbf{x})}$, where $p(q^*_{N+1}\,|\,\mathbf{x})\approx\widetilde{p}(\mathbf{x}\,|\,q^*_{N+1})p(q^*_{N+1})=p(q^*_{N+1})\prod_{m=0}^{M-1}\widetilde{p}(\mathbf{x}_{m+1}\,|\,\mathbf{x}_m,q^*_{N+1})$. Now each term $\widetilde{p}(\mathbf{x}_{m+1}\,|\,\mathbf{x}_m,q^*_{N+1})$ can be computed using the DTQ method discussed in Section 2.1.
  - Sample a uniform random variable $u_N\sim\mathscr{U}(0,1)$. If $\rho>u_N$ set $\hat{\theta}_{N+1}=q^*_{N+1}$; in this case, the proposal is accepted. Else set $\hat{\theta}_{N+1}=q_N$ and the proposal is rejected.

Once we have obtained all the samples $q_0,q_1,\cdots,q_N$ from the Metropolis algorithm, we discard a sufficient number of initial samples to ensure the Markov chain has converged to its invariant distribution.

3 Numerical Results

We implement the Metropolis algorithm in R. Inside the Metropolis algorithm, we evaluate the likelihood function using the DTQ method, which is implemented in C++ as an R package.

3.1 LC circuit test problem

To test the performance of our method, we consider the SDE

$$\mathrm{d}X_{1,t} = -\frac{X_{2,t}}{L}\mathrm{d}t + \frac{s_1^2}{L}\mathrm{d}W_{1,t}, \qquad \mathrm{d}X_{2,t} = \frac{X_{1,t}}{C}\mathrm{d}t + \frac{s_2^2}{C}\mathrm{d}W_{2,t}. \tag{7}$$

This system describes a noisy electrical oscillator with one inductor (with inductance L) and one capacitor (with capacitance C). The dependent variables $X_{1,t}$ and $X_{2,t}$ represent, respectively, the current and voltage of the circuit at time t.

Our goal here is to test the performance of the algorithm using simulated data. To generate this data, we start with known values of the parameters L, C, s_1, and s_2. Using a fixed initial condition $(X_{1,0}, X_{2,0})$, we then use the Euler-Maruyama method to step (7) forward in time until a final time $T > 0$. When we carry out this time-stepping, we use a step size of 0.001 and then retain only those samples at times $t_m = m\Delta t$, from $m = 0$ to $m = M$, where $M\Delta t = T$.

For the tests presented here, the known values of the parameters are $L = C = (2\pi)^{-1}$ and $s_1 = s_2 = .4/\sqrt{2\pi}$. The simulated data is taken over two periods of the oscillator ($T = 2$) with a full resolution of $\Delta t = 0.01$. By, for example, taking every other row of this data set, we can obtain data with a resolution of $\Delta t = 0.02$.

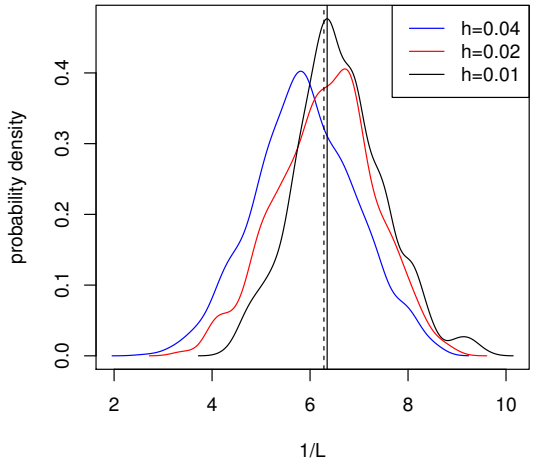
Using the samples $\{\mathbf{x}_m\}_{m=0}^M$ thus constructed, we run the Metropolis algorithm. Because capacitance and inductance are physically constrained to be positive, we take $1/L = \theta_1^2$, $1/C = \theta_2^2$, $s_1^2/L = \theta_3^2$, and $s_2^2/C = \theta_4^2$. In this way, each θ_j is allowed to take a random walk on $\mathbb{R}$ rather than be restricted to the positive half of the real line. For each θ_j, we use a diffuse Gaussian prior with mean 0 and standard deviation 100. For the proposal distribution Z_{N+1} in the auxiliary Markov chain, we choose i.i.d. Gaussians with mean 0 and standard deviation 0.35.

For the tests presented here, we infer only one parameter, θ_1, keeping the other parameters fixed at their known values. When we run the Metropolis algorithm, we discard the first 100 samples and retain the next 1000 samples. For each value of Δt and the DTQ time step h, we compute both the mean of the samples of θ_1^2 and the mode of the kernel density estimate of θ_1^2. We compare these values against the true value of the parameter $1/L = 2\pi$ and record the relative errors as, respectively, e_1 and e_2 in the following table:

Δt	h	e_1 (relative error of mean)	e_2 (relative error of mode)
0.04	0.04	6.1%	7.6%
0.04	0.02	0.54%	6.8%
0.04	0.01	5.1%	1.1%
0.02	0.02	12%	14%
0.02	0.01	4.9%	2.3%

When $h = \Delta t$, only one step of the method described in Section 2.1 is required to go from time t_m to t_{m+1}. This step does not use any quadrature at all—one merely evaluates (4) using a point mass for the density at time t_m. The resulting likelihood function is a product of Gaussians. On the other hand, when h is strictly less than Δt, we must use quadrature (i.e., the actual DTQ method) to step forward in time from t_m to t_{m+1}.

To visualize the results, we present the following plot of the posterior densities of $1/L$ corresponding to the first three lines of the table above. The true value of $1/L = 2\pi$ is indicated by the dashed black line. The posterior mode for $h = 0.01$ is indicated by the solid black line:



Overall, the findings above support our view that using the DTQ method to compute the likelihood yields more accurate posteriors than using a purely Gaussian likelihood.

3.2 Pursuit Problem

To simulate the pursuit model, we consider the SDE

$$\mathrm{d}x^c = \frac{s(t)(x^r - x^c)}{\sqrt{(x^r - x^c)^2 + (y^r - y^c)^2}}\mathrm{d}t + \nu_1 \mathrm{d}W_t^1$$
$$\mathrm{d}y^c = \frac{s(t)(y^r - y^c)}{\sqrt{(x^r - x^c)^2 + (y^r - y^c)^2}}\mathrm{d}t + \nu_2 \mathrm{d}W_t^2$$

The variable $\mathbf{X}^r = (x^r, y^r)$ represents the runner's trajectory and the variable $\mathbf{X}^c = (x^c, y^c)$ represents the chaser's trajectory at time t. Our goal is to test the performance of the algorithm, first using simulated data and then using real-life data. To generate the data, we start with known values of the parameters ν_1, ν_2 and $s(t), 0 \leq t \leq T$. The runner's trajectory is simulated as a sinusoidal curve from $T = 0$ to $T = 8$. The chaser's trajectory is simulated using the Euler-Maruyama method to step the SDE forward in time from a fixed initial condition $\mathbf{X}_0 = (x_0^c, y_0^c)$. During the generation of the data, we use a step size of 10^{-4} and then retain those samples at times $t_m = m\Delta t$ from $m = 0$ to $m = M$, where $M\Delta t = T$. Sample paths are saved from the same trajectory at increasing finer time intervals ($M = 4000, 2000, 1000, 500, 400, 200, 100$) to study the convergence of the algorithm.

TODO: Plot of runner and chaser

For the tests presented here, the known parameters $s(t) = \begin{cases} 0.4 \text{ if } 0 \leq t < 4 \\ 1.0 \text{ if } 4 \leq t \leq 8 \end{cases}$, $\nu_1 = 0.15, \nu_2 = 0.1$

Acknowledgements We gratefully acknowledge support from UC Merced's Committee on Research.

References

1. Archambeau, C., Cornford, D., Opper, M., Shawe-Taylor, J.: Gaussian process approximations of stochastic differential equations. Journal of Machine Learning Research: Workshop and Conference Proceedings **1**, 1–16 (2007)
2. Archambeau, C., Opper, M., Shen, Y., Cornford, D., Shawe-Taylor, J.: Variational inference for diffusion processes. In: Advances in Neural Information Processing Systems 20, pp. 17–24 (2007)
3. Bhat, H.S., Madushani, R.W.M.A.: Density tracking by quadrature for stochastic differential equations (2016). Preprint
4. Fuchs, C.: Inference for Diffusion Processes: With Applications in Life Sciences. Springer, Berlin (2013)
5. Iacus, S.M.: Simulation and Inference for Stochastic Differential Equations: With R Examples. Springer Series in Statistics. Springer, New York (2009)
6. Picchini, U.: Inference for SDE models via approximate Bayesian computation. Journal of Computational and Graphical Statistics **23**(4), 1080–1100 (2014)
7. Ruttor, A., Batz, P., Opper, M.: Approximate Gaussian process inference for the drift function in stochastic differential equations. In: Advances in Neural Information Processing Systems 26, pp. 2040–2048 (2013)

8. Sørensen, H.: Parametric inference for diffusion processes observed at discrete points in time: a survey. International Statistical Review **72**(3), 337–354 (2004)
9. Vrettas, M.D., Opper, M., Cornford, D.: Variational mean-field algorithm for efficient inference in large systems of stochastic differential equations. Physical Review E **91**(012148) (2015)