Journal of the Royal Statistical Society Series B: Statistical Methodology, 2023, **00**, 1–25
https://doi.org/10.1093/jrsssb/qkad022

ROYAL STATISTICAL SOCIETY
DATA | EVIDENCE | DECISIONS
Journal of the Statistics Society
Series B
Statistical Methodology

Original Article

Exact Monte Carlo likelihood-based inference for jump-diffusion processes

Flávio B. Gonçalves[1], Krzysztof Łatuszyński[2] and Gareth O. Roberts[2]

[1]Department of Statistics, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil
[2]Department of Statistics, University of Warwick, Coventry, UK

Address for correspondence: Flávio B. Gonçalves, Department of Statistics, Universidade Federal de Minas Gerais, Av. Antonio Carlos, 6627 – DEST, ICEx, UFMG 31270-901, Belo Horizonte, Minas Gerais, Brazil. Email: fbgoncalves@est.ufmg.br

Abstract

Statistical inference for discretely observed jump-diffusion processes is a complex problem which motivates new methodological challenges. Thus, existing approaches invariably resort to time-discretisations which inevitably lead to approximations in inference. In this paper, we give the first general collection of methodologies for exact (in this context meaning discretisation-free) likelihood-based inference for discretely observed finite activity jump-diffusions. The only sources of error involved are Monte Carlo error and convergence of expectation maximisation (EM) or Markov chain Monte Carlo (MCMC) algorithms. We shall introduce both frequentist and Bayesian approaches, illustrating the methodology through simulated and real examples.

Keywords: Barker's algorithm, MCEM, MCMC, Poisson estimator, retrospective sampling

1 Introduction

Jump-diffusion processes are used in a variety of applications in several scientific areas, especially economics (see Ball & Roma, 1993; Barndorff-Nielsen & Shephard, 2004; Cont & Tankov, 2004; Duffie et al., 2000; Eraker, 2004; Eraker et al., 2003; Feng & Linetsky, 2008; Johannes, 2004; Kennedy et al., 2009; Runggaldier, 2003). Other applications can be found, for example, in physics (see Chudley & Elliott, 1961), biomedicine (see Grenander & Miller, 1994), and object recognition (see Srivastava et al., 2002). Jump-diffusions are natural extensions to diffusions, allowing for discrete discontinuities in trajectories which are otherwise described by diffusion dynamics, thus offering additional flexibility in modelling phenomena which exhibit sudden large jumps.

Inference for jump-diffusions is a challenging problem because transition densities are almost always intractable. This problem is typically overcome using approximations based on time-discretisations which typically lead to systematic biases which are difficult to quantify (see Bruti-Liberati & Platen, 2007). Therefore, the problem requires new approaches which circumvent the need for approximation, and we shall call them *exact* solutions. They are exact in the sense of not involving any kind of discrete-time approximation although other sources of inaccuracy are involved—Monte Carlo error and convergence of EM and Markov chain Monte Carlo (MCMC) algorithms, which are more reasonable to control. Whereas efficient exact solutions have already been proposed for the context where there is no jump component (see Beskos et al., 2006, 2009; Sermaidis et al., 2013), the problem for jump-diffusions is extremely under-developed.

This paper provides a general suite of methodologies for exact likelihood-based inference for discretely observed finite activity jump-diffusions, featuring both maximum likelihood and Bayesian approaches. Some of our approaches directly use an algorithm that performs exact

Received: February 13, 2023. **Revised:** February 20, 2023. **Accepted:** February 26, 2023

simulation of a class of jump-diffusion bridges proposed in Gonçalves and Roberts (2014) and called the Jump Bridge Exact Algorithm (JBEA). However, our most general approaches involve the construction of significant extensions of JBEA.

First, we propose two methodologies (MCEM and MCMC) for the case where the drift and the jump-rate are uniformly bounded and JBEA can be directly applied. Our methodology collapses to that of Beskos et al. (2006) in the case where no jump component is present. We also propose two methods for the general case where both the drift and the jump-rate may be unbounded. One of these is an importance sampling (IS) adaptation of the first MCEM method we propose providing improved Monte Carlo variance properties. The second method introduces a new perspective for solving the inference problem in an exact framework using an infinite-dimensional MCMC algorithm based on Barker's transitions (Barker, 1965) and novel simulation techniques.

Our exact approach can be applied to a wide class of univariate models which allow nonlinear state-dependent drift and diffusion coefficients, and state-dependent and time-inhomogeneous jump rate and jump size distribution. The methodology extends in principle to certain multivariate jump-diffusions, although we do not explore this direction here.

Whilst the focus of this paper is clearly on exact inference for jump-diffusions, the use of exact Barker's transitions via a Two-Coin algorithm which we will introduce appears to be of generic interest for any context with intractable accept/reject ratios. Barker's method is rarely used, as it is known to be uniformly dominated by the much more well-known Metropolis–Hastings (MH) accept/reject formula. However, MH never beats Barker by a factor greater than 2 in Peskun order sense (Łatuszyński & Roberts, 2013), and crucially, in contrast to the MH algorithm, the particular form of the accept/reject formula in the Barker case permits the construction of the Two-Coin procedure.

This paper is organised as follows. The remainder of Section 1 provides a literature review and formally defines the class of jump-diffusion processes to be considered. Section 2 presents some technical background material used throughout the remainder of the paper. Section 3 presents two inference methods directly based on the exact simulation of jump-diffusion bridges and applicable when the drift and jump rates are uniformly bounded. Section 4 presents two other methods which make indirect use of the algorithm for exact simulation of jump-diffusion bridges and do not require the boundedness assumption. In Section 5, the methods are applied to simulated data sets to investigate their efficiency. Finally, two real data sets concerning the exchange rate GBP/USD and the S&P500 index are analysed in Section 6.

1.1 Literature review on approximate methods

Existing solutions for the inference problem that we consider are based on approximations which, in turn, rely on *path discretisation* and/or *data augmentation* strategies. These solutions follow basically three main directions: considering alternative estimators to the *maximum likelihood estimator* (MLE); using numerical approximations to the unknown likelihood function; and estimating an approximation to the likelihood by using Monte Carlo methods. Moreover, several of the methods assume state-independence of various components of the model.

Alternative estimators can be found in Duffie and Singleton (1993) and Duffie and Glynn (2004). Numerical approximations can be found in Lo (1988), Aït-Sahalia and Yu (2006), and Filipović et al. (2013). Particle filter-based Monte Carlo methods can be found in Johannes et al. (2002, 2009). Golightly (2009) proposes a refinement where the particles are propagated via MCMC. The author proposes a MH algorithm generalising the Durham and Gallant bridge (Durham & Gallant, 2002) for the jump-diffusion case.

The most promising and robust solutions among the ones cited above rely on data augmentation, via Euler schemes, and Monte Carlo methods. An important drawback in data augmentation schemes is the fact that as more data points are input between the observations, so to reduce the error when using the discrete time approximation, the computational cost grows significantly. Moreover, Bruti-Liberati and Platen (2007) provide a result regarding the error of the strong Euler approximation, stating that a discrete time approximation Y^{Δ}, with time step size Δ, approximates a jump-diffusion Y (satisfying some regularity conditions) at time T such that

$$\sqrt{E(|Y_T - Y_T^{\Delta}|^2)} \leq C\Delta^{0.5},$$

for $\Delta \in (0, \Delta_0)$, for a finite $\Delta_0 > 0$, where C is a positive constant, independent of Δ. Although the result provides the order (w.r.t. Δ) of a bound on the error, no result about how to obtain C is provided.

We emphasise that, whilst the methodologies proposed in this paper involve only Monte Carlo and EM or MCMC convergence errors and there exists a vast literature about how to control those type of errors, the most promising approximate solutions also have to deal with time discretisation errors. Furthermore, whilst improvement on the discrete-time error inevitably leads to an increase in computation cost, the exact methodologies proposed in this paper are free from such errors at a fixed and reasonable computational cost.

Recently, an exact methodology that performs unbiased Monte Carlo estimation of the transition density of a class of univariate jump-diffusions was proposed in Giesecke and Schwenkler (2019), building on the work of Pedersen (1995), Beskos et al. (2009), Chen (2009), and Casella and Roberts (2010). This work provides a methodology for exact MLE methods although suffers from two main drawbacks. Firstly, the results that assure finite variance of the transition density estimator and consistency and asymptotic normality of the simulated MLE require strong assumptions that are typically hard to check and often not met in realistic situations. Secondly, even when these assumptions are met, the method can be very inefficient due to high IS weight variances. Our methodology will circumvent these difficulties through the use of rejection sampling, IS with provably finite variances, and a novel Barker's MCMC methodology for Bayesian inference.

1.2 The jump-diffusion model

Formally, a jump-diffusion is the stochastic process $V := \{V_s : 0 \leq s \leq t\}$ that solves the stochastic differential equation (SDE):

$$dV_s = b(V_{s-})ds + \sigma(V_{s-})dW_s + \int_E g_1(z, V_{s-})m(dz, ds), \quad V_0 = v_0, \tag{1}$$

where $b : \mathbb{R} \to \mathbb{R}$, $\sigma : \mathbb{R} \to \mathbb{R}^+$ and $g_1 : E \times \mathbb{R} \to \mathbb{R}$ are assumed to satisfy the regularity conditions (locally Lipschitz, with a linear growth bound) to guarantee a unique weak solution (see Platen & Bruti-Liberati, 2010, Section 1.9). Functions b and σ can be time-inhomogeneous, but we restrict ourselves to the time homogeneous case. W_s is a Brownian motion and $m(dz, ds)$ is a random counting measure on the product space $E \times [0, t]$, for $E \subseteq \mathbb{R}$, with associated intensity measure λ_m. We assume that λ_m is absolutely continuous with respect to Lebesgue measure on $E \times [0, t]$ and Markov dependent on V:

$$\lambda_m(dz, ds; V_{s-}) = \lambda_m(z, s; V_{s-})dzds = \lambda_1(s, V_{s-})f_Z(z; s)dzds, \tag{2}$$

where, for any $v \in \mathbb{R}$, $\lambda_1(\cdot, v)$ is a non-negative real valued function on $[0, t]$ and, for any $s \in [0, t]$, $f_Z(\cdot; s)$ is a standard density function with support E. According to (1) and (2), between any two jumps, the process V behaves as a homogeneous diffusion process with drift b and diffusion coefficient σ. The jump times follow a Markov point process on $[0, t]$ with intensity function (jump rate) $\lambda_1(s; V_{s-})$. A random variable Z_j with density $f_Z(z; \tau_j)$ is associated to each of the N jump times $\tau_1, \ldots, \tau_N$ and, along with the state of the process, determines the size of the jump $g_1(Z_j, V_{\tau_j-})$ at time τ_j.

2 Necessary technical background material

Some of our methodology will rely heavily on the JBEA for the exact simulation of jump-diffusion bridges introduced in Gonçalves and Roberts (2014). JBEA simulates a finite-dimensional representation from the exact probability law of a class of univariate jump-diffusion bridges and can be used to derive methodologies to perform exact likelihood-based inference for discretely observed jump-diffusions. We refer to this finite-dimensional representation as a skeleton of the jump-diffusion bridge.

JBEA can only be directly applied to processes with unit diffusion coefficient. This is a genuine restriction in the multivariate case where not all diffusions can be reduced to this case. On the other

hand, in the one-dimensional case, under weak regularity conditions we can always apply the Lamperti transform to obtain a process as required (assuming that the diffusion coefficient is continuously differentiable). Thus, we set $X_s = \eta(V_s)$ with the Lamperti transform

$$\eta(v) := \int_{v^*}^{v} \frac{1}{\sigma(u)} \mathrm{d}u, \tag{3}$$

where v^* is some arbitrary element of the state space of V.

The transformed process $X := \{X_s : 0 \leq s \leq t\}$ is a one-dimensional jump-diffusion solving the SDE:

$$dX_s = \alpha(X_{s-})ds + dW_s + \int_E g(z, X_{s-})m(dz, ds), \quad X_0 = \eta(v_0) = x_0, \tag{4}$$

where the jump rate $\lambda(s, X_{s-})$ of X is given by the original jump rate λ_1 applied at $\eta^{-1}(X_{s-})$ (the inverse of η). The expressions for α and g are given in Appendix A of the online supplementary material, which also describes the details of JBEA.

2.1 Some notational conventions

Consider the one-dimensional jump-diffusion process $V := \{V_s : 0 \leq s \leq T\}$ solving the SDE in (1). Suppose that functions b, σ, g_1, f_Z and λ_1 depend on an unknown parameter $\theta \in \Theta$ and that we observe V at $(n+1)$ time instances $\mathbf{t} = (t_0, \ldots, t_n)$, with $t_0 = 0$ and $t_n = T$. We aim to carry out inference about the parameter vector θ based on the observations $\mathbf{v} = \{v_0, \ldots, v_n\}$ of V.

Note that η and the functions α, λ and g, from the transformed process X, will also depend on θ and we shall introduce this dependence on the notation accordingly. Furthermore, we define $\mathbf{x}(\theta) = \{x_0(\theta), x_1(\theta), \ldots, x_n(\theta)\}$ as the transformed observations, which depend on θ whenever the diffusion coefficient σ of V does, and suppress the notation (θ) from $\mathbf{x}$ and x_i whenever σ does not depend on θ.

From this point on, we shall also consider the following notation and definitions. Let $\mathbb{P}$ be the probability measure of the jump-diffusion X solving the SDE in (4) but with components (drift and jump process) indexed by the parameter vector θ. Let $\mathbb{Q}$ to be the measure of a jump-diffusion which has the same initial value as $\mathbb{P}$ and is the sum of a Brownian motion and a jump process with jump rate 1 and jump size density f which does not depend on θ and such that $f_g \ll f$. We shall use the notations $\mathbb{P}$ and $\mathbb{Q}$ to refer to the original measures and all the respective measures induced by them upon (measurable) transformations and conditioning, which will be properly indicated in the brackets when writing, for example, $\mathbb{P}(\cdot|\cdot)$.

Define $\Delta t_i = t_i - t_{i-1}$ and let N_i be the number of jumps from X in $(t_{i-1}, t_i]$, $\tau_{i,j}$ and $J_{i,j}$ be the jth jump time and jump size, respectively, for $j = 1, \ldots, N_i$, and $\boldsymbol{\tau}$ be the set of all the jumps times in $[0, T]$. Let $f_g(\cdot; s, X_{s-}, \theta)$ be the jump size density of X at time s. We shall use the abbreviated notation $\ell(i, j; \theta) := \lambda(\tau_{i,j}, X_{\tau_{i,j}-}; \theta)$ and $f_g(\cdot; i, j, \theta) := f_g(\cdot; \tau_{i,j}, X_{\tau_{i,j}-}; \theta)$. Finally, define $A(u; \theta) := \int_0^u \alpha(y; \theta)\, \mathrm{d}y$ and $\Delta A(i, j; \theta) := A(X_{\tau_{i,j}}; \theta) - A(X_{\tau_{i,j}-}; \theta)$. We assume that $A(u; \theta)$ can be obtained analytically.

The inference algorithms to be presented in Section 3 require, among other things (see assumptions of the JBEA algorithm in Appendix A of the online supplementary material), that the function $(\alpha^2 + \alpha')$ is bounded below and functions α and λ are bounded above. The algorithms from Section 4, however, do not require those two conditions and can then be applied to a much wider class of jump-diffusion processes. This greater applicability, however, comes at the price of a considerably higher computational complexity.

3 Inference directly using JBEA

In this section, we present two inference algorithms based directly on JBEA to perform exact inference for discretely observed jump-diffusion processes. The first one is a Monte Carlo EM algorithm to find the MLE and the second one is a MCMC algorithm to sample from the posterior distribution of the parameters.

3.1 A Monte Carlo EM algorithm

In this algorithm, JBEA is used to obtain a Monte Carlo estimate of the expectation on the E-step. The exactness of the method is obtained by combining the exactness feature of JBEA with some auxiliary variable techniques. We present two versions of the algorithm to cover the cases where the diffusion coefficient does or does not depend on unknown parameters.

3.1.1 The case where the diffusion coefficient is known

In order to obtain the likelihood function of θ based on the observations $\mathbf{x}$ we need the finite-dimensional distributions of X, which are typically unavailable. We can, however, obtain the augmented or full likelihood function, that is obtained from observing the entire jump-diffusion trajectory in $[0, T]$. This function is obtained by writing down the Radon–Nikodym derivative of $\mathbb{P}$ w.r.t. any measure that dominates $\mathbb{P}$ and $\mathbb{Q}$ does not depend on θ (see, e.g., Gonçalves & Franklin, 2019). This is the natural environment to apply the *EM algorithm*. Note, however, that the missing data is an infinite-dimensional random variable in this case and, for that reason, we have to be particularly careful in constructing a nondegenerate algorithm.

We define X_{mis} as the unobserved part of the process X consisting of the bridges of X between the observations $\mathbf{x}$. We use $\mathbb{Q}$ as the dominating measure to obtain the likelihood of a complete path of X. Theorem 2 from Gonçalves and Roberts (2014) implies that the complete log-likelihood function is given by

$$\begin{aligned} l(X \mid \theta) = A(x_n; \theta) - A(x_0; \theta) - \sum_{i=1}^{n}\left(\sum_{j=1}^{N_i}\big(\Delta A(i, j; \theta)\big) - \int_{t_{i-1}}^{t_i} \phi(s, X_s; \theta)\, \mathrm{d}s\right) \\ + \sum_{i=1}^{n}\sum_{j=1}^{N_i} \log\big(\ell(i, j; \theta)\big) + \log\big(f_g(J_{i,j}; i, j, \theta)\big) + \kappa, \end{aligned} \tag{5}$$

where

$$\phi(s, X_s; \theta) = \left(\frac{\alpha^2 + \alpha'}{2}\right)(X_s; \theta) + \lambda(s, X_s; \theta), \tag{6}$$

and κ is a constant with respect to θ and can, therefore, be neglected in the EM algorithm. Since the expectation of (5) cannot be evaluated analytically, we rely on Monte Carlo methods to obtain an unbiased and strongly consistent estimator of it.

Given the auxiliary variable $U = (U_1, \ldots, U_n)$, where the U_i's are mutually independent with $U_i \sim U(t_{i-1}, t_i)$ and independent of X, we have that

$$\begin{aligned} \mathbb{E}_{X_{\text{mis}} \mid \mathbf{x}; \theta'}\big[l(X \mid \theta)\big] = A(x_n; \theta) - A(x_0; \theta) - \mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{N_i}\big(\Delta A(i, j; \theta); \theta\big)\right] + \mathbb{E}(\kappa) \\ - \mathbb{E}\left[\sum_{i=1}^{n} \Delta t_i \phi(U_i, X_{U_i}; \theta)\right] + \mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{N_i} \log\big(\ell(i, j; \theta)\big) + \log\big(f_g(J_{i,j}; i, j, \theta)\big)\right], \end{aligned} \tag{7}$$

with the subscript $(X_{\text{mis}}, U \mid \mathbf{x}; \theta')$ being omitted from all the expectations on the r.h.s. of (7).

On the E-step, an estimate of the r.h.s. in (7) is obtained via simple Monte Carlo integration, based on M iid samples from $(X_{\text{mis}}, U \mid \mathbf{x}; \theta')$. Due to the Markov property of X, each one of the M samples is independently obtained from the n bridges of $X \sim \mathbb{P}$ in intervals (t_{i-1}, t_i), conditional on the respective values of x_{i-1} and x_i, for $i = 1, \ldots, n$, via JBEA. This also includes the simulation of X at times U. On the M-step, the estimate obtained on the E-step is maximised w.r.t. θ. The value obtained from this maximisation becomes the new θ' for a new iteration of the algorithm. Thus, a single iteration of the algorithm inputs θ' and outputs the maximising θ value. The algorithm iterates this procedure until the output values are convergent according to the desired accuracy.

It is well documented (see, e.g., Fort & Moulines, 2003, and references therein) that the number of Monte Carlo samples should increase with the EM iterations in order to overcome Monte Carlo error. Finally, the maximisation step may require numerical methods. For well behaved likelihoods, standard numerical optimisation algorithms should work well. For example, the quasi-Newton method BFGS (see, e.g., Fletcher, 1987, Section 3.2) is used in the example presented in Section 5.1.

3.1.2 *The case where the diffusion coefficient is unknown*

We now focus on the case where the diffusion coefficient of V depends on unknown parameters. A relevant result in (jump-)diffusion theory is that a complete path of a (jump-)diffusion can be used to almost surely perfectly estimate $\sigma(V_{s-}; \theta)$. The result for the jump-diffusion case states that (Protter, 2004, II.6)

$$\lim_{n\to\infty} \sum_{l=1}^{n} \left(V_{lT/n} - V_{(l-1)T/n}\right)^2 = \int_0^T \sigma^2(V_s; \theta)\, \mathrm{d}s + \sum_{j=1}^{N} (V_{\tau_j} - V_{\tau_j -})^2. \tag{8}$$

The computational implication of this result is that we cannot construct an EM algorithm as in Section 3.1.1 because there is a perfect correlation between σ and the missing path as described in (8) (see Meng & Rubin, 1993). This problem is also relevant in MCMC algorithms, where it was first encountered in the context of inference for diffusions (see Elerian, 1999; Roberts & Stramer, 2001).

We propose a solution for this problem which combines together and then substantially generalises Roberts and Stramer (2001) and Beskos et al. (2006). The method consists of a suitable transformation of the missing data that breaks the dependence between the missing data and the parameters, when conditioned on the observed values.

In our infinite-dimensional context, this problem is equivalent to finding a reparameterisation of the missing data so that the dominating measure is independent of the parameters. We construct this reparameterisation in two easily interpreted transformations.

The first transformation considers the Lamperti transformed process $X = \eta(V; \theta)$ as described in (3). We make $\tau_{i,0} = t_{i-1}$ and $\tau_{i,N_i+1} = t_i$ to obtain the second level of path transformation $\{X_s \to \dot{X}_s;\ s \in (t_{i-1}, t_i) \setminus \{\tau_{i,1}, \ldots, \tau_{i,N_i}\}\}$, for $i = 1, \ldots, n$ and $j = 1, \ldots, N_i + 1$:

$$\dot{X}_s := X_s - \left(1 - \frac{s - \tau_{i,j-1}}{\tau_{i,j} - \tau_{i,j-1}}\right) X_{\tau_{i,j-1}} - \left(\frac{s - \tau_{i,j-1}}{\tau_{i,j} - \tau_{i,j-1}}\right) X_{\tau_{i,j}-}; \quad s \in (\tau_{i,j-1}, \tau_{i,j}), \tag{9}$$

where $X_{\tau_{i,0}} = x_{i-1}(\theta)$ and $X_{\tau_{i,N_i+1}-} = x_i(\theta)$.

Note that $\dot{X} := \{X_s; s \in [0, T]\} \setminus \{\mathbf{t}, \boldsymbol{\tau}\}$ is a collection of diffusion bridges starting and ending at 0 between the observation and jump times. Its dynamics depend on θ and are typically intractable; nevertheless it is easy to simulate $\dot{X}$ at any time s, conditionally on $\mathbf{v}$ and a specific value of θ, by firstly computing $x_{i-1}(\theta)$ and $x_i(\theta)$, then simulating X conditioned on these two values via JBEA and, finally, applying the transformation in (9). This will allow us to use a θ-free dominating measure to obtain the likelihood function of a complete path of X (see the proof of Lemma 1 in Appendix C of the online supplementary material). Furthermore, the MCEM algorithm requires the simulation of one bridge point of $\dot{X}$ at a uniformly chosen time instant U_i in every interval (t_{i-1}, t_i), as it is shown further ahead.

We define $\mathbf{x}(\theta)$ as the vector of the transformed observations. The inverse transformation to obtain X_s from $\dot{X}_s$ is given by

$$\begin{aligned} X_s = \varphi(s, V_{\text{mis}}, \mathbf{x}(\theta)) = \dot{X}_s &+ \left(1 - \frac{s - \tau_{i,j-1}}{\tau_{i,j} - \tau_{i,j-1}}\right) X_{i,j-1} \\ &+ \left(\frac{s - \tau_{i,j-1}}{\tau_{i,j} - \tau_{i,j-1}}\right) X_{i,j-}; \quad s \in (\tau_{i,j-1}, \tau_{i,j}). \end{aligned} \tag{10}$$

The data augmentation scheme is now based on $V_{\text{mis}} = (\mathbf{J}, X_J, \dot{X})$, where $\mathbf{J}$ is the jump process (jump times and sizes) of X and X_J is X at its jump times. We define $V_{\text{com}} = (\mathbf{v}, V_{\text{mis}})$.

We obtain the likelihood function of θ by writing the joint law of $\mathbf{v}$ and a suitable transformation of V_{mis} (see the proof of Lemma 1 in Appendix C of the online supplementary material) with respect to a dominating measure that does not depend on θ. First, define

$$\dot{\phi}(s, \dot{X}_s; \theta) := \phi(s, \varphi(s, V_{\text{mis}}, \mathbf{x}(\theta)); \theta). \tag{11}$$

Lemma 1 The likelihood of θ given the complete data V_{com} is given by

$$\begin{aligned}\exp\{l(V_{\text{com}} \mid \theta)\} \propto \exp\left\{A(x_n(\theta); \theta) - A(x_0(\theta); \theta) - \sum_{i=1}^{n}\sum_{j=1}^{N_i} \Delta A(i, j; \theta)\right\} \\ \times \exp\left\{-\sum_{i=1}^{n} \int_{t_{i-1}}^{t_i} \dot{\phi}(s, \dot{X}_s; \theta)\, \mathrm{d}s\right\} \prod_{i=1}^{n}\prod_{j=1}^{N_i} \left[\ell(i, j; \theta) f_g(J_{i,j}; i, j, \theta)\right] \\ \times \prod_{i=1}^{n}\left[|\sigma(v_i; \theta)^{-1}| f_N(X_{i,1-}; x_{i-1}(\theta), \tau_{i,1} - t_{i-1}) f_N(x_i(\theta); X_{i,N_i}, t_i - \tau_{i,N_i})\right],\end{aligned} \tag{12}$$

where $f_N(u; a, b)$ is the Lebesgue density of the normal distribution with mean a and variance b evaluated at u.

Proof. See Appendix C of the online supplementary material. □

Multiplicative terms in $\exp\{l(V_{\text{com}} \mid \theta)\}$ that do not depend on θ, including some normal p.d.f.'s, are omitted in expression (12).

The algorithm is now analogous to the case where σ is known. On the E-step, M iid samples of $(V_{\text{mis}}, U \mid \mathbf{v}; \theta')$ are obtained via JBEA (and applying the transformation in (9) to obtain $\dot{X}$) and used to compute a Monte Carlo estimate of the expectation of the log of (12) w.r.t. the measure of $(V_{\text{mis}}, U \mid \mathbf{v}; \theta')$. Notice that

$$\mathbb{E}\left[\sum_{i=1}^{n} \int_{t_{i-1}}^{t_i} \dot{\phi}(s, \dot{X}_s; \theta)\, \mathrm{d}s\right] = \mathbb{E}\left[\sum_{i=1}^{n} \Delta t_i \dot{\phi}(U_i, \dot{X}_{U_i}; \theta)\right],$$

where the expectation on the l.h.s. is w.r.t. $(V_{\text{mis}} \mid \mathbf{v}; \theta')$ and the one on the r.h.s. is w.r.t. $(V_{\text{mis}}, U \mid \mathbf{v}; \theta')$. On the M-step, the estimate obtained on the E-step is maximised w.r.t. θ and the obtained value becomes the new θ' for a new iteration of the algorithm. Care must be exercised to keep track of all terms that depend on θ, for example, $\varphi(\cdot, V_{\text{mis}}, \mathbf{x}(\theta))$ and the $x_i(\theta)$'s.

3.2 A MCMC approach

We now present a Bayesian solution for the inference problem with parameter-dependent diffusion coefficient by constructing a Markov chain with stationary distribution given by the exact joint posterior distribution of θ and the variables output by JBEA. More specifically, we have a standard MH algorithm to updated θ, interspersed by moves to update a skeleton $\mathbf{S}$ of the latent jump-diffusion bridges of X, as introduced in Section 2 and described in Appendix A of the online supplementary material, in $(0, T) \setminus \mathbf{t}$, where $\mathbf{S}$ is the union of skeletons $S^{(i)}$, for all i.

In theory, to be exact, our MCMC scheme will require the imputation and storage of entire continuous-time trajectories of jump-diffusion bridges between observations. The key to the retrospective simulation approach is to note that subsequent parameter updates within our MCMC scheme can be carried out exactly by only requiring a finite but random collection of imputed bridge values (stored within a skeleton). The skeletons $S^{(i)}$ are sampled independently, by applying

JBEA to each of the intervals (t_{i-1}, t_i). The conditional independence of $S^{(i)}$'s, given $\mathbf{x}$ and θ, is implied by the Markov property. A skeleton $S^{(i)}$ contains

1. the jump times and sizes of X in $(t_{i-1}, t_i]$,
2. a random collection $\Phi^{(i)}$ of points in the rectangle $[t_{i-1}, t_i] \times [0, 1]$,
3. the value of X at the jump times and at the times given by the horizontal coordinates of $\Phi^{(i)}$, and
4. a finite-dimensional measurable function $\mathbf{L}_i$ of X, termed its *layer*, on the interval $[t_{i-1}, t_i]$ (and motivated and described below).

In order for the subsequent parameters update steps within our MCMC procedure to be carried out, it turns out that we need at least to know finite upper and lower bounds on X within $[t_{i-1}, t_i]$. $\mathbf{L}_i$ therefore needs to facilitate these bounds. However, since skeletons are dynamic (we need to be able to add extra time points at which X is evaluated) it is also vital that we can carry out conditional simulation of such extra points conditional on $\mathbf{L}_i$. A formulation for $\mathbf{L}_i$ which satisfies these two conditions was presented in Appendix E of the online supplementary material, based on the work of Beskos et al. (2008). We give an informal description here, while referring the interested reader to Online Supplementary Material, Appendix E for full details.

Writing

$$\mathbf{L}_i = (L_{i,1}, L_{i,2}, \ldots, L_{i,N_i+1})$$

and given a sequence of positive increasing thresholds $\{b_k;\ k = 0, 1, 2, \ldots\}$, with $b_0 = 0$, we define each component

$$L_{i,j} = 1 + \sup\left\{k; \sup_{s \in (\tau_{i,j-1} - \tau_{i,j})} |\dot{X}_s| \geq b_k\right\}.$$

This way, $L_{i,j} = K$ implies that $\dot{X}_s \in (-b_K, b_K)$, for $s \in (\tau_{i,j-1}, \tau_{i,j})$.

Let $\pi(\theta)$ be the prior density of θ. We shall use π as a general notation for densities. It is likely that we are primarily interested in the posterior distribution $\pi(\theta \mid \mathbf{v})$ of θ although, depending on the application, we might also be interested in the posterior distribution $\pi(\mathbf{J}, X_J, \dot{X} \mid \mathbf{v})$, which can be obtained from the joint posterior $\pi(\theta, \mathbf{S} \mid \mathbf{v})$.

The full conditional density of θ is given by Theorem 1 in Appendix B of the online supplementary material. It may be convenient to sample θ in blocks as it may be possible to simulate directly from the full conditional distributions of parameters in the jump rate and jump size distribution when conjugated priors are used. This way a MH step is only used for parameters in the drift and diffusion coefficient.

4 Inference when exact simulation of bridges is not possible

The two algorithms proposed in Section 3 require JBEA to be directly applied, meaning that the drift α and the jump rate need to be bounded and function $(\alpha^2 + \alpha')$ need to be bounded below. In order to circumvent those restrictions, we present two algorithms to perform exact inference which do not make direct use of JBEA, and, therefore, can be applied to a more general class of jump-diffusion models. Formally, the two algorithms only require assumptions (a), (c), (e) and from Appendix A of the online supplementary material to be satisfied and that $A(u)$ can be obtained analytically. One of the key points in the proposed algorithms is the use of local bounds for the proposed path, which are obtained from the simulation of functions $\mathbf{L}_i$, as introduced in Section 3.2, through an algorithm called the layered Brownian bridge (see Appendix E of the online supplementary material). These bounds allow us to obtain local lower and upper bounds for the function $((\alpha^2 + \alpha')/2 + \lambda)$, as required by the algorithms in this section.

The first approach introduces an MCEM algorithm that follows similar lines to that previously presented in Section 3.1 but uses an IS estimate for the E-step. The second approach introduces an infinite-dimensional MCMC algorithm which is quite different from that in Section 3.2.

4.1 Importance sampling MCEM

For simplicity, we present here the algorithm for the case in which the diffusion coefficient is parameter-free. The algorithm for the general case can be obtained by combining results from Section 3.1.2 with this.

The algorithm requires that we can find a dominating measure from which we can simulate and w.r.t. which the measure of bridges of $X \sim \mathbb{P}$ is absolutely continuous. However, unlike the algorithm from Section 3.1, the implied RN derivative is not required to be uniformly bounded. The algorithm proposed in this section outperforms the algorithm from Section 3.1, when the latter is feasible and given that the proposal measure from which the samples are drawn are the same in both algorithms, in the sense of having a smaller variance MC estimator of the expectation on the E-step (it is well known in the literature that IS outperforms rejection sampling in terms of MC variance). On the other hand, in our context, the IS estimator has a higher computational cost, given the extra variables and calculations involved due to the intractability of the original weights (to be detailed further ahead).

We introduce the probability measure $\mathbb{D}$ that, restricted to the interval (t_{i-1}, t_i), is the law of a bridge process conditioned to start at x_{i-1} and end at x_i, defined as the sum of a jump process and a continuous process. The former is, marginally, a jump process with a constant positive jump rate $\lambda_i(\theta)$ and jump size density $f_i(\cdot; \theta)$ and the latter is, conditional on the former, a Brownian bridge that starts at x_{i-1} at time t_{i-1} and finishes in $x_i - J_i$ at time t_i, where J_i is the sum of the jumps of the jump process. The jump components $\lambda_i(\theta)$ and $f_i(\cdot; \theta)$ are assumed to be time- and state-independent, although the extension to make them time dependent is straightforward (see Gonçalves & Roberts, 2014). We also assume $\mathbb{D}$ to be mutually independent among the intervals (t_{i-1}, t_i). In order to sample from $\mathbb{D}$ in (t_{i-1}, t_i) we use the following algorithm.

Algorithm 1: sampling from $\mathbb{D}$ in (t_{i-1}, t_i)

1. Sample the jump process with jump rate $\lambda_i(\theta)$ and jump size density $f_i(\cdot; \theta)$;
2. sample the value of the process at the jump times by sampling a Brownian bridge (starting at x_{i-1} at time 0 and ending at $(x_i - J_i)$ at time Δt_i) at the jump times and adding the jumps;
3. sample the standard Brownian bridges $\dot{X}_s$ between times $\{t_{i-1}, \tau_1, \ldots, \tau_{N_i}, t_i\}$ any required time instant;
4. obtain the value of $X(s)$ at the required time instants mentioned on the previous step by applying the transformation φ as defined in (10).

Defining X_{mis} as the missing paths of X and $X_{\text{com}} = \{\mathbf{x}, X_{\text{mis}}\}$, with $\mathbf{x} = \eta(\mathbf{v})$, the expectation of the complete log-likelihood $l(X_{\text{com}}; \theta)$ in (5) can be written as

$$\mathbb{E}_{\mathbb{P}\,|\,\mathbf{x}}[l(X_{\text{com}}; \theta)] = \mathbb{E}_{\mathbb{D}}\left[\frac{d\mathbb{P}}{d\mathbb{D}}(X_{\text{mis}}\,|\,\mathbf{x}; \theta')\, l(X_{\text{com}}; \theta)\right], \tag{13}$$

where $\mathbb{E}_{\mathbb{P}|\mathbf{x}}$ is the expectation w.r.t. to the measure of $(X_{\text{mis}}\,|\,\mathbf{x}; \theta')$ under $\mathbb{P}$, and $w := (d\mathbb{P}/d\mathbb{D})(X_{\text{mis}}\,|\,\mathbf{x}; \theta')$ is the RN derivative of that measure w.r.t. $\mathbb{D}$. The results in Lemma 4 (see Appendix B of the online supplementary material) combined with the Markov property gives that $w = \prod_{i=1}^{n} w_i$, where

$$\begin{aligned} w_i = \kappa_{1,i}(\mathbf{x}; \theta') \exp\left\{ - \int_{t_{i-1}}^{t_i} \phi(s, X_s; \theta'), \mathrm{d}s - \sum_{j=1}^{N_i} \Delta A(i, j; \theta') - \frac{(\Delta x_i - J_i)^2}{2\Delta t_i} \right\} \\ \times \prod_{j=1}^{N_i} \frac{\ell(i, j; \theta') f_g(J_{i,j}; i, j, \theta')}{\lambda_i(\theta') f_i(J_{i,j}; \theta')}, \end{aligned} \tag{14}$$

with $\kappa_{1,i}(\mathbf{x}; \theta')$ being a function that does not depend on X_{mis} and $\Delta x_i = x_i - x_{i-1}$.

Now note that the complete log-likelihood $l(X_{\text{com}}; \theta)$ can be written as $\sum_{i=1}^{n} l_i(\theta)$, where

$$l_i(\theta) = A(x_{i-1}; \theta) - A(x_i; \theta) - \sum_{j=1}^{N_i} \big(\Delta A(i, j; \theta)\big) - \int_{t_{i-1}}^{t_i} \phi(s, X_s; \theta)\, \mathrm{d}s + \sum_{j=1}^{N_i} \log\{\ell(i, j; \theta) f_g(J_{i,j}; i, j, \theta)\}. \tag{15}$$

The first natural choice of an estimator for the r.h.s. in (13) is

$$E_1(\theta) = \frac{1}{M} \sum_{k=1}^{M} \sum_{i=1}^{n} \left(\prod_{j=1}^{n} w_j^{(k)} \right) l_i^{(k)}(\theta), \tag{16}$$

where M is the number of Monte Carlo samples from $(X_{\text{mis}} \mid \mathbf{x}; \theta')$ under $\mathbb{D}$, and $w_i^{(k)}$ and $l_i^{(k)}$ are the values of w_i and $l_i(\theta)$ for the kth sampled value of X_{mis}.

Nevertheless, an improved estimator can be devised based on the following result.

Proposition 1

$$\mathbb{E}_{\mathbb{P}|\mathbf{x}}\big[l(\theta)\big] = \sum_{i=1}^{n} \mathbb{E}_{\mathbb{D}}\big[w_i l_i(\theta)\big]. \tag{17}$$

Proof. See Appendix C of the online supplementary material. □

This result leads to the following Monte Carlo estimator:

$$E_2(\theta) = \frac{1}{M} \sum_{k=1}^{M} \sum_{i=1}^{n} w_i^{(k)} l_i^{(k)}(\theta). \tag{18}$$

Estimator E_1 is a global IS estimator whilst estimator E_2 is a local one. The latter is expected to have a much better behaviour than the former. In fact, we have the following result.

Proposition 2

$$Var[E_2] \leq Var[E_1]. \tag{19}$$

Proof. See Appendix C of the online supplementary material. □

In order to devise an exact algorithm we need to avoid the calculation of the integrals in the expressions of w_i and $l_i(\theta)$. This is achieved by firstly defining the auxiliary variables: $K_i \sim \text{Poi}(\mu_i(X; \theta')\Delta t_i)$, $U_{i,k} \sim \mathcal{U}(t_{i-1}, t_i)$, $\dot{U}_i \sim \mathcal{U}(t_{i-1}, t_i)$, for $k = 1, \ldots, K_i$, $i = 1, \ldots, n$. We also define $K = (K_1, \ldots, K_n)$, $U_i = (U_{i,1}, \ldots, U_{i,K_i})$, $U = (U_1, \ldots, U_n)$, and $\dot{U} = (\dot{U}_1, \ldots, \dot{U}_n)$, and assume all the components of K, U, and $\dot{U}$ to be independent. The $\mu_i(X)$'s are functions of X that must be chosen appropriately, as it is explained in the next paragraphs, to ensure the efficiency of the algorithm. Now, using the same auxiliary variable strategy from Section 3.1.1 and the ideas from the Poisson estimator (see Beskos et al., 2006), we have the following result.

Proposition 3

$$\mathbb{E}[w_i l_i(\theta)] = \mathbb{E}\left[h_{0,i}\, e^{(\mu_i(X;\theta') - b_i(X;\theta'))\Delta t_i} \mu_i(X;\theta')^{-K_i} \left(\prod_{k=1}^{K_i} (b_i(X;\theta') - h_{1,i,k}) \right) (h_{2,i} - \Delta t_i h_{3,i}) \right], \tag{20}$$

where the expectation on the l.h.s. is w.r.t. $\mathbb{D}$ and the one on the r.h.s. is w.r.t. $\mathbb{D} \otimes \mathbb{A}$, with $\mathbb{A}$ being the joint probability measure of $(K, U, \dot{U})$. Also, both of those measures consider $\theta = \theta'$. The $b_i(X;\theta')$'s are functions of X that must be chosen appropriately, as it is explained in the next paragraphs, to ensure the efficiency of the algorithm. Finally,

$$\begin{aligned} h_{0,i} &= \kappa_{1,i}(X;\theta') \prod_{j=1}^{N_i} \frac{\ell(i,j;\theta') f_g(J_{i,j};i,j,\theta')}{\lambda_i(\theta') f_i(J_{i,j};\theta')} \exp\left\{ -\sum_{j=1}^{N_i} (\Delta A(i,j;\theta')) - \frac{(\Delta x_i - J_i)^2}{2\Delta t_i} \right\}; \\ h_{1,i,k} &= \phi\left(U_{i,k}, X_{U_{i,k}};\theta'\right); \quad h_{3,i} = \phi(\dot{U}_i, X_{\dot{U}_i};\theta); \\ h_{2,i} &= A(x_{i-1};\theta) - A(x_i;\theta) - \sum_{j=1}^{N_i} (\Delta A(i,j;\theta)) + \sum_{j=1}^{N_i} \log\left(\ell(i,j;\theta)\right) + \log\left(f_g(J_{i,j};i,j,\theta)\right). \end{aligned} \tag{21}$$

Proof. See Appendix C of the online supplementary material. □

We now need to simulate K, U, $\dot{U}$, $X_{\dot{U}_i}$'s, $X_{U_{i,k}}$'s, X_J, $\mathbf{J}$, and $\mathbf{L}_i$'s to obtain IS estimates of the expectations on the r.h.s. of (17). Recall that the random variable $\mathbf{L}_i$, as introduced in Section 3.2, defines local upper and lower bounds for X in $[t_{i-1}, t_i]$. These are local bounds in the sense that they bound the specific path simulated from $\mathbb{D}$ and, therefore, dismisses the need for uniform (lower and upper) bounds on $\phi(s, u;\theta')$, for $u \in \mathbb{R}$, $s \in [0, T]$.

Naturally, the choices of $b_i(X;\theta')$ and $\mu_i(X;\theta')$ are closely related to the efficiency of the algorithm. Based on the results from Beskos et al. (2006) and Fearnhead et al. (2008), we recommend $b_i(X;\theta') = \bar{U}_i(X;\theta')$ and $\mu_i(X;\theta') = \bar{U}_i(X;\theta') - \bar{L}_i(X;\theta')$, where $\bar{L}_i(X;\theta')$ and $\bar{U}_i(X;\theta')$ are, respectively, local lower and upper bounds for $\phi(s, X_s;\theta')$, for $s \in [t_{i-1}, t_i]$, which are a function of $\mathbf{L}_i$, as it is explained in Section 3.2.

We simulate the Brownian bridge component $\dot{X}_s$ of $\mathbb{D}$ by simulating standard Brownian bridges and applying the linear transformation in (10) to obtain the desired bridges. Defining $\dot{z}_{i,j}$ and $\dot{u}_{i,j}$ as the lower and upper bounds of $\dot{X}_s$, for $s \in (\tau_{i,j-1}, \tau_{i,j})$, provided by $L_{i,j}$ as defined in Section 3.2, we set $\bar{H}_{i,j} = [\min\{X_{i,j-1}, X_{i,j-}\} + \dot{z}_{i,j}, \quad \max\{X_{i,j-1}, X_{i,j-}\} + \dot{u}_{i,j}]$ as the respective local lower and upper bounds for X_s in $(\tau_{i,j-1}, \tau_{i,j})$. These, in turn, are used to obtain the local bounds $\bar{L}_i(X;\theta')$ and $\bar{U}_i(X;\theta')$ for $\phi(s, X_s;\theta')$, in $[t_{i-1}, t_i]$ as follows:

$$\bar{U}_i(X;\theta') = \sup_j \sup \{\phi(s, X_s;\theta'); \quad s \in (\tau_{i,j-1}, \tau_{i,j}), \quad X_s \in \bar{H}_{i,j}\}; \tag{22}$$

$$\bar{L}_i(X;\theta') = \inf_j \inf \{\phi(s, X_s;\theta'); \quad s \in (\tau_{i,j-1}, \tau_{i,j}), \quad X_s \in \bar{H}_{i,j}\}. \tag{23}$$

Using the results from Proposition 3, we get the following estimator:

$$\dot{E}_2(\theta) = \frac{1}{M} \sum_{k=1}^{M} \sum_{i=1}^{n} \dot{w}_i^{(k)} \dot{l}_i^{(k)}(\theta), \tag{24}$$

where $\dot{w}_i^{(k)}$ and $\dot{l}_i^{(k)}(\theta)$ are the kth sample value of

$$\dot{w}_i = h_{0,i}\, \mathrm{e}^{(\mu_i(X;\theta') - b_i(X;\theta'))\Delta t_i} \mu_i(X;\theta')^{-K_i} \prod_{k=1}^{K_i} (b_i(X;\theta') - h_{1,i,k}), \tag{25}$$

$$\dot{l}_i(\theta) = \left(h_{2,i} - \Delta t_i h_{3,i}\right). \tag{26}$$

The efficiency of the proposed important sampling MCEM algorithm relies heavily on the variance of the IS estimator. In particular, it is important to have a finite variance estimator, which typically relies on finite variance weights. Finite variance weights do not ensure that the estimator also has finite variance (they do whenever the target function is bounded), but weights with infinite variance will typically lead to infinite variance estimators. Besides being hard to check if the weights $\dot{w}_i$ have finite variance, these depend on terms $\kappa_{1,i}(\mathbf{x};\theta')$ (through $f_{0,i}$) which, in turn, depend on the (unknown) transition density. Nevertheless, we can modify $\dot{E}_2$ in a way that terms $\kappa_{1,i}(\mathbf{x};\theta')$ can be ignored and also get a nice property regarding the variance of the estimator. The fact that $\mathbb{E}[\dot{w}_i] = 1$ suggests the following estimator:

$$\dot{E}_3(\theta) = \sum_{i=1}^{n} \sum_{k=1}^{M} \tilde{w}_i^{(k)} \dot{l}_i^{(k)}(\theta), \tag{27}$$

where $\tilde{w}_i^{(k)} = \dot{w}_i^{(k)} / \sum_{k=1}^{M} \dot{w}_i^{(k)}$. Estimator $\dot{E}_3$ is biased but has some nice properties as stated in the following lemma.

Lemma 2 $\dot{E}_3$ is a strongly consistent estimator for the expectation from the E-step and has finite variance whenever $\dot{l}_i(\theta)$ has finite variance, for all i, under $\mathbb{D} \otimes \mathbb{A}$.

Proof. See Appendix C of the online supplementary material. □

Although $\dot{E}_3$ is biased, Lemma 2 implies that it is asymptotically unbiased. Furthermore, the number of Monte Carlo samples M is expected to be typically large enough to make the bias negligible.

In the general case in which the diffusion coefficient depends on unknown parameters, we consider the complete log-likelihood $l_i(\theta)$ to be as in (12) and obtain the transformed bridges $\dot{X}$ by applying the transformation in (9).

Finally, we also construct an estimator for the covariance matrix of the MLE, which is presented in Appendix D of the online supplementary material.

4.2 An infinite-dimensional Barker's MCMC

If at least one of the assumption of the JBEA algorithm (see Appendix A of the online supplementary material) is not satisfied, we are unable to use it to exactly simulate bridges from the jump-diffusion X and, as a consequence, we are unable to find a finite-dimensional representation of those bridges for which we know and can simulate from the full conditional distribution of the parameters given this representation. In this case, we can devise a Bayesian solution based on the (infinite-dimensional) complete bridges V_{mis} (as defined in Section 3.1.2). We consider the general case in which the diffusion coefficient $\sigma(V_{s-};\theta)$ depends on unknown parameters and propose a MCMC algorithm that iterates between the update of the two following blocks via accept/reject algorithms:

$$\left(V_{\text{mis}} | \theta, \mathbf{v}\right) \quad \text{and} \quad \left(\theta | V_{\text{mis}}, \mathbf{v}\right). \tag{28}$$

The first block cannot be sampled exactly from its full conditional distribution due to the intractability of JBEA. In order to perform inference without resorting to discrete approximations we

need to sample both V_{mis} and θ in a Markov chain with the required invariant distribution, by unveiling only a finite-dimensional part of V_{mis}. That can be achieved by constructing a MCMC algorithm with accept/reject steps for both blocks—V_{mis} and θ, and adopting a Barker's accept/reject algorithm, instead of the traditional MH one. The particular form of the Barker's acceptance probability allows us to devise a Bernoulli factory algorithm to sample events with the respective (intractable) acceptance probabilities by unveiling only a (random) finite-dimensional representation of V_{mis}.

The main contribution of this algorithm when compared to the one from Section 3.2 is that it can be applied to a quite general class of jump-diffusion processes, including processes with unbounded drift and/or unbounded jump rate and unbounded function ($\alpha^2 + \alpha'$).

4.2.1 *Sampling the missing paths*

The missing paths of X are sampled via a Metropolis-type algorithm but using the Barker's acceptance probability instead of the traditional MH one. Barker's algorithm was proposed in Barker (1965) and, like the MH algorithm, defines a Markov chain with a given target invariant distribution. On each iteration of the chain, a proposal value is simulated from an arbitrary distribution and accepted according to an acceptance probability that preserves detailed balance. The probability to go from x to y when proposing from $q(y \mid x)$ is given by

$$a_B(x, y) = \frac{\pi^*(y)q(x \mid y)}{\pi^*(x)q(y \mid x) + \pi^*(y)q(x \mid y)}, \tag{29}$$

where π^* is the invariant distribution of the chain. Barker's method is not as popular as the prolific MH algorithm since it is easy to demonstrate that Barker is uniformly dominated by MH in terms of its convergence properties (see Peskun, 1973). However, it is also easy to see that Barker is no worse than twice as slow as MH and its convergence properties are broadly comparable (Łatuszyński & Roberts, 2013). More specifically, we can show that $a_{MH}(x, y)/2 \leq a_B(x, y) \leq a_{MH}(x, y)$, for all (x, y) and, as it is shown by Łatuszyński and Roberts (2013), for all measurable function $f \in L^2(\pi^*)$ for which a square root central limit theorem (CLT) holds for the MH chain with CLT asymptotic variance σ^2_{MH}, a corresponding CLT holds for f and the Barker's chain with CLT asymptotic variance σ^2_B, such that $\sigma^2_{MH} \leq \sigma^2_B \leq 2\sigma^2_{MH} + \sigma^2_{\pi^*}$, where $\sigma^2_{\pi^*} := \text{Var}_{\pi^*}(f)$. For us, the smoothness of the Barker's acceptance probability function turns out to be crucial in obtaining a feasible algorithm.

The block V_{mis} is broken into n blocks which consists of V_{mis} restricted to each of the n time intervals (t_{i-1}, t_i). Due to the Markov property, the full conditional distribution of each of these blocks depends only on θ and (v_{i-1}, v_i), which means that these n sampling steps on each iteration of the MCMC algorithm are conditionally independent and can be performed in parallel. The Barker's proposal for V_{mis} in (t_{i-1}, t_i) is sampled from $\mathbb{D}$, as defined in Section 4.1, using Algorithm 1 from that same section. Note that the initial and end values that the bridges, $x_{i-1}(\theta)$ and $x_i(\theta)$, are functions of θ.

For a given interval (t_{i-1}, t_i), let $X^{(k-1)}$ be the current state of V_{mis} in the MCMC chain and a new proposal jump-diffusion bridge $X^{(k)}$ is drawn from $\mathbb{D}$. It is convenient to use the proposal measure also as a dominating measure to obtain the acceptance probability which, from Lemma 4 (see Appendix B of the online supplementary material), is given by

$$\alpha_X = \frac{\frac{d\mathbb{P}}{d\mathbb{D}}(X^{(k)} \mid \mathbf{x}(\theta))}{\frac{d\mathbb{P}}{d\mathbb{D}}(X^{(k-1)} \mid \mathbf{x}(\theta)) + \frac{d\mathbb{P}}{d\mathbb{D}}(X^{(k)} \mid \mathbf{x}(\theta))} = \frac{s_k(X^{(k)})p_k(X^{(k)})}{s_{k-1}X^{(k-1)}p_{k-1}X^{(k-1)} + s_k(X^{(k)})p_k(X^{(k)})}, \tag{30}$$

where

$$s_k(X^{(k)}) = \exp\left\{-\sum_{j=1}^{N_i}\big(\Delta A(i, j; \theta)\big) - \mathcal{I}_{a_i}(\theta) - \frac{(\Delta x_i(\theta) - J_i)^2}{2\Delta t_i}\right\}\prod_{j=1}^{N_i}\frac{\ell(i, j; \theta)f_g(J_{i,j}; i, j, \theta)}{\lambda_i(\theta)f_i(J_{i,j}; \theta)}, \tag{31}$$

$$p_k(X^{(k)}) = \exp\left\{-\int_{t_{i-1}}^{t_i} \phi(s, X_s; \theta) - a_i(s; \theta)\, \mathrm{d}s\right\}, \tag{32}$$

for $\Delta x_i(\theta) = x_i(\theta) - x_{i-1}(\theta)$, $\mathcal{I}_{a_i} = \int_{t_{i-1}}^{t_i} a_i(s; \theta)\, \mathrm{d}s$ and $a_i(s; \theta)$ being any (local) lower bound satisfying

$$a_i(s; \theta) \leq \phi(s, X_s; \theta) \quad \text{for } s \in (t_{i-1}, t_i). \tag{33}$$

The expressions for $s_{k-1}(X^{(k-1)})$ and $p_{k-1}(X^{(k-1)})$ are obtained by replacing $X^{(k)}$ by $X^{(k-1)}$ in (31)–(33).

The choice of $a_i(s; \theta)$ has direct impact on the efficiency of the algorithm, as it is discussed further ahead in Section 4.2.2. If the function ϕ is not uniformly bounded below, $a_i(s; \theta)$ can only be a local bound. We can set, for example, $a_i(s; \theta) = \bar{L}_i(X; \theta')$, for $s \in (t_{i-1}, t_i)$, as defined in (23). This choice ought to work well in most cases. A more refined bound, however, may be obtained by adopting a time-dependent bound $a_i(s; \theta)$ (see Appendix F of the online supplementary material).

To perform the accept/reject step of the algorithm, we need to simulate a Bernoulli(α_X) random variable. Most importantly, we have to use only a finite-dimensional representation of the missing paths to do it. That is achieved by the Two-Coin algorithm given in the following proposition.

Proposition 4 Suppose we want to simulate a Bernoulli(α_X) random variable, where s_k and s_{k-1} are known positive numbers and it is possible to simulate events of unknown probabilities given by p_k and p_{k-1}. The following algorithm outputs an exact draw of this Bernoulli random variable:

Two-Coin algorithm

1. Sample C_1 from $\{0, 1\}$, where $P(C_1 = 1) = \frac{s_k}{s_{k-1}+s_k}$;
2. if $C_1 = 1$, sample C_2 from $\{0, 1\}$, where $P(C_2 = 1) = p_k$;
   - if $C_2 = 1$, output 1;
   - if $C_2 = 0$, go back to 1;
3. if $C_1 = 0$, sample C_2 from $\{0, 1\}$, where $P(C_2 = 1) = p_{k-1}$;
   - if $C_2 = 1$, output 0;
   - if $C_2 = 0$, go back to 1.

Proof. Let q be the probability that there is no output in one trial of C_1 and C_2, that is $q = (s_k(1 - p_k) + s_{k-1}(1 - p_{k-1}))/s_k + s_{k-1}$. Then, the probability that the algorithm outputs 1 is $(s_k/s_k + s_{k-1})p_k \sum_{i=0}^{\infty} q^i = s_k p_k/(s_k p_k + s_{k-1} p_{k-1})$. The fact that the algorithm is a geometric experiment implies that it finishes after a finite number of loops with probability 1, which concludes the proof. □

In order to simulate events of probability $p_k(X^{(k)})$ and $p_{k-1}(X^{(k-1)})$ we use the Poisson Coin algorithm described in Appendix A of the online supplementary material, for which we require an upper bound $r_i(\theta)$ for the integrand $\phi(s, X_{s-}; \theta) - a_i(s; \theta)$ in (32) in the interval $[t_{i-1}, t_i]$. This can be a local bound, which explains why a uniform upper bound on that function is not required here. Setting, for example, $a_i(s; \theta) = \bar{L}_i(X; \theta')$, for $s \in (t_{i-1}, t_i)$, and $r_i(\theta) = \bar{U}_i(X; \theta') - \bar{L}_i(X; \theta')$, as defined in (22) and (23), should work well in most cases. More refined bounds, however, can be obtained by adopting a time-dependent bound $a_i(s; \theta)$ (see Appendix F of the online supplementary material).

It is important to notice that all the values of the proposed path unveiled along the Two-Coin algorithm need to be kept, meaning that the simulation of the path in any new time instant has to be conditioned on all the already unveiled points.

Let X_Ψ be the process X at a finite random collection of time instants which, for each interval $[t_{i-1}, t_i]$, are sampled in the Poisson Coin algorithm used in all the loops of the Two-Coin algorithm. Each Barker's step for V_{mis} stores a finite-dimensional representation of the missing path, consisting of $\mathbf{J}$, X_J, $\mathbf{L}_i$, and X_Ψ in each interval $[t_{i-1}, t_i]$, $i = 1, \ldots, n$. It is crucial though to be able to simulate further points, given this skeleton, on the θ step of the MCMC algorithm, as we discuss in Section 4.2.2.

4.2.2 *Sampling the parameters*

Typically, the full conditional distribution of the parameters will depend on the unknown integral in (37). In order to minimise the complexity of the algorithm, the parameters should firstly be separated into two blocks: the first—θ_1, consisting of those parameters whose full conditional distributions depend on the integral in (37), and the second block—θ_2, consisting of the remaining parameters. Parameters in θ_2 are sampled as in an ordinary tractable MCMC—they may be broken into smaller blocks, sampled directly from the full conditional or via MH steps. Parameters in θ_1 are sampled via Barker's step, which may be performed separately for sub-blocks. The full conditional density of θ and, consequently, of any subvector of its coordinates is given by

$$\pi(\theta \mid V_{\text{mis}}, \mathbf{v}) \propto \pi(V_{\text{mis}}, \mathbf{v} \mid \theta)\pi(\theta), \tag{34}$$

where $\pi(V_{\text{mis}}, \mathbf{v} \mid \theta)$ is given in Lemma 1.

Proposals for θ_1 are drawn from a symmetric random walk, $\theta_1^{(k)} = \theta_1^{(k-1)} + \epsilon$, where ϵ is a r.v. symmetric around 0 with a covariance matrix Σ properly tuned to obtain suitable acceptance rates (varying from around 0.27, for unidimensional θ_1, to around 0.16 for dimension 5+; Agrawal et al., 2021). A move $\theta_1^{(k-1)} \to \theta_1^{(k)}$ is accepted with probability:

$$\alpha_\theta = \frac{\pi(\theta^{(k)} \mid V_{\text{mis}}, \mathbf{v})}{\pi(\theta^{(k-1)} \mid V_{\text{mis}}, \mathbf{v}) + \pi(\theta^{(k)} \mid V_{\text{mis}}, \mathbf{v})} = \frac{s_k(\theta^{(k)})p_k(\theta^{(k)})}{s_{k-1}(\theta^{(k-1)})p_{k-1}(\theta^{(k-1)}) + s_k(\theta^{(k)})p_k(\theta^{(k)})}, \tag{35}$$

where

$$\begin{aligned} s_k(\theta^{(k)}) = \pi(\theta^{(k)}) \exp\left\{A(x_n(\theta^{(k)}); \theta^{(k)}) - A(x_0(\theta^{(k)}); \theta^{(k)}) - \sum_{i=1}^{n} \mathcal{I}_{a_i}(\theta^{(k)})\right\} \\ \times \prod_{i=1}^{n}\left[\exp\left\{-\sum_{j=1}^{N_i} \Delta A(i, j; \theta^{(k)})\right\} \prod_{j=1}^{N_i} \ell(i, j; \theta^{(k)}) f_g(J_{i,j}; i, j, \theta^{(k)})\right] \\ \times \prod_{i=1}^{n}\left[\sigma(V_{t_i}; \theta^{(k)})^{-1} f_N(X_{i,1-}; x_{i-1}(\theta^{(k)}), \tau_{i,1} - t_{i-1}) f_N(x_i(\theta^{(k)}); X_{i,N_i}, t_i - \tau_{i,N_i})\right], \end{aligned} \tag{36}$$

$$p_k(\theta^{(k)}) = \exp\left\{-\int_0^T \dot{\phi}(s, \dot{X}_s; \theta^{(k)}) - \dot{a}(s; \theta^{(k)})\, \mathrm{d}s\right\}, \tag{37}$$

$$\dot{a}(s; \theta^{(k)}) = \dot{a}_{i,j}(s; \theta^{(k)}) \quad \text{for } s \in [\tau_{j,i-1}, \tau_{j,i}], \tag{38}$$

$$\dot{a}_{i,j}(s; \theta) \le \dot{\phi}(s, \dot{X}_s; \theta) \quad \text{for } s \in [\tau_{j,i-1}, \tau_{j,i}], \tag{39}$$

where $\theta^{(k)}$ has values $\theta_1^{(k)}$ for θ_1 and the current value of the chain for θ_2. The expressions for $s_{k-1}(\theta^{(k-1)})$ and $p_{k-1}(\theta^{(k-1)})$ are obtained by replacing $\theta^{(k)}$ by $\theta^{(k-1)}$ in (36)–(38).

The efficiency of this sampling step depends on the efficiency of the Two-Coin algorithm which ultimately relies on the probabilities of success of C_2, p_{k-1}, and p_k. The smaller these probabilities are, the higher is the expected number of trials per iteration. Moreover, at every trial, the missing paths have to be unveiled at extra time points, which also increases the computational cost. The optimisation of p_{k-1} and p_k is related to the optimisation of the lower bounds $\dot{a}_{i,j}$ from (39).

We simulate the second coin piecewise by simulating a sequence of 'subcoins', forward in time from 0 to T, with probability

$$p_{k,i,j} = \exp\left\{-\int_{\tau_{i,j-1}}^{\tau_{i,j}} \dot{\phi}(s, \dot{X}_s; \theta^{(k)}) - \dot{a}_{i,j}(s; \theta^{(k)})\, \mathrm{d}s\right\} \tag{40}$$

or $p_{k-1,i,j}$, which replaces $\theta^{(k)}$ by $\theta^{(k-1)}$ in (40). If we get 0 at some point, the remaining subcoins do not need to be simulated and the Two-Coin algorithm starts a new loop.

In order to simulate each subcoin $p_{k,i,j}$, we apply the Poisson Coin algorithm by simulating a Poisson process with rate $\dot{r}_{i,j}(\theta^{(k)})$, where $\dot{r}_{i,j}(\theta)$ is a local upper bound for the integrand $\dot{\phi}(s, X_{s-}; \theta) - \dot{a}(s; \theta)$ in (37) in the interval $[\tau_{i,j-1}, \tau_{i,j}]$. An efficient solution to obtain $\dot{r}_{i,j}(\theta)$ is presented in Appendix F of the online supplementary material.

The overall MCMC algorithm is the following.

Barker's MCMC for jump-diffusions

1. Provide initial values for all the parameters θ;
2. make k=1;
3. sample $V_{\text{mis}}^{(k)}$ (retrospectively and including $\mathbf{L}_i$) once on every interval between consecutive observations via Barker's by proposing from the measure $\mathbb{D}$ and accepting with probability α_X in (30) using the Two-Coin algorithm;
4. sample each block of $\theta_1^{(k)}$ via Barker's by proposing from a symmetric random walk and accepting with probability α_θ in (35) using the Two-Coin algorithm;
5. sample the remaining parameters $\theta_2^{(k)}$ directly from their full conditionals or via Gaussian random walk MH;
6. to continue running the chain, make $k = k + 1$ and `GOTO` 3, otherwise, `STOP`.

4.2.3 *Improving the MCMC algorithm*

The efficiency of the proposed MCMC algorithm in terms of convergence and computing cost depends on many factors. Here, we shall discuss strategies for improvements.

In order to avoid numerical problems when computing the probability of C_1 in the Two-Coin algorithm, compute $(1 + \exp(\log(s_{k-1}) - \log(s_k)))^{-1}$.

A way to improve the mixing of the chain is to add an extra step to the Gibbs sampler to sample the missing (continuous) paths between observation and jump times using the EA algorithm (see Beskos et al., 2006). We only require the extra condition that $(\alpha^2 + \alpha')(u)$ is bounded below for all u in the state space of X. EA performs rejection sampling by proposing from a Brownian bridge with the same initial and ending values as the target diffusion bridge and accepting with probability given by $\exp\{-\int_{t_{i-1}}^{t_i} ((\alpha^2 + \alpha')/2)(X_s) - l\,\mathrm{d}s\}$, where $l = \inf_u ((\alpha^2 + \alpha')/2)(u)$. Since this algorithm will output an exact draw of the missing paths between observation and jump times, it guarantees that these bridges are updated on every iteration of the chain at least once. This strategy ought to improve the mixing of the chain considerably. Furthermore, it eliminates the problem of accumulating too many bridge points along the iterations of the chain due to rejections on the Barker's step. It is recommended to add one EA step after each Barker's step—for X or θ-blocks. The example in Section 5.2 implements this extra update step.

Further improvements in the chain mixing could be obtained by modifying the Barker's step for V_{mis}, which is the only step where the jump process is updated. One idea is to perform multiple steps of this type. Another simple and virtually costless strategy is to adopt a pilot analysis to tune the jump rate based on the average number of jumps in each interval. Moreover, if one has reasonable choices for initial values of the parameters it may be a good idea to warm up the chain on the first iterations before start updating the parameters.

In order to increase the success probability of C_2 in the Two-Coin algorithm we need to improve the lower bound $\dot{a}(s, \theta)$, which can be done by tightening the lower and upper bounds for the missing paths in V_{mis}. That is achievable by performing what we call the *layer refinement algorithm* which, instead of simulating upper and lower bounds (through $\mathbf{L}_i$) for the proposal standard bridges between the merged observation and jump times, simulates upper and lower bounds for shorter intervals. For a constant $m \in \mathbb{N}$ and a time interval $(\tau_{i,j-1}, \tau_{i,j})$, we first simulate a standard BB at times $\tau_{i,j-1} + k(\tau_{i,j} - \tau_{i,j-1})/m$, for $k = 1, \ldots, m-1$, and then obtain upper and lower bounds for each of the subintervals. This strategy will provide tighter bounds for the standard bridge and, consequently, for the X path. In particular, for a standard bridge of length t, the range is $\mathcal{O}(\sqrt{t})$. One may also set different refinement levels along the intervals of consecutive observations. It is

typically straightforward to identify the relation between the cost of the Two-Coin algorithm and the values assumed by the process X, so the level of refinement may be set as a function of the extreme observations of each interval. The detailed algorithms to simulate $\mathbf{L}_i$ and obtain the bounds for the X path are presented in Appendices E and F of the online supplementary material. A cheaper version of the EA step described above consists of updating the missing continuous bridges between observation, jump and refinement points, conditioned on the auxiliary variable $\mathbf{L}_i$.

The layer refinement algorithm suggests that the algorithm that simulates only the first coin C_1 may be seen as a discretised method and it is the second coin that guarantees the exactness of the algorithm. In that case, the larger m is the smaller the error due to the discretisation.

Two other strategies may increase the probability of the second coins. The first one is to break θ_1 into smaller blocks (the extreme case being one parameter per block). This may help in the sense that a smaller block may simplify the function inside the integral in (37) and allow for a more efficient lower bound $a(s;\cdot)$. The second strategy is to divide the numerator and denominator of the Barker's acceptance probability by p_{k-1} or p_k. This will make the probability of one of the second coins equal to 1 and possibly increase the probability of the other one. In order to choose between p_{k-1} and p_k one has to look at the resulting ratio of p's and recognise which choice has a ratio of the form $s_0 \int_0^T h(X_s)ds$, where s_0 is computable and h is a non-negative function. This choice may vary between iterations of the chain, depending on the proposed value of the parameters. If this strategy is adopted, one of the C_2 coins will have success probability 1 and the other one will typically get smaller as the size of the random walk step increases. For this reason, it may be wise to truncate the random walk proposal for θ_1 (if it is Gaussian) between say ± 3.5 or ± 4 standard deviations to avoid the algorithm from collapsing after an average number of $1/\epsilon$ iterations—ϵ being the probability of proposing extreme values. This would have very little effect on the algorithms's convergence properties. Two other reasonable strategies are to use a uniform random walk proposals and to perform multiple updates using smaller variance proposals. The multiple proposals ought to compensate the slower mixing but with considerable gains in computational cost.

4.3 Practical implementation

Identifiability is a particular problem when dealing with jump-diffusions. It is crucial to have enough information to distinguish well between continuous and jump variation. Practical strategies to tackle the problem include fixing some of the parameters at reasonable values, which is not always easy, or using informative priors under a Bayesian approach. A general idea that should always be considered is that of admitting the least possible number of jumps necessary to get a good fit so that the jumps only occur when a pure diffusion process is not flexible enough to model the phenomenon of interest. This not only mitigates the identifiability problem but also favours the interpretability of jumps and parameters and, finally, ought to improve computational efficiency.

Computational cost is another important issue when dealing with the algorithms proposed in this paper. Since the MCEM and MCMC algorithms from Section 3 are considerably cheaper than the algorithms presented in Section 4, we may consider some practical strategies to allow the use of the former ones. For example, we may occasionally truncate the jump rate. However, this strategy ought to be adopted with care as it may seriously compromise the analysis. On one hand, a severe truncation could significantly compromise desirable properties of the original (without truncation) model. On the other hand, a conservative bound (that uses a truncation value that is expected to be hardly reached by the model) is bound to seriously compromise the computational cost of the algorithm. The bounds for the jump rate and drift obtained from the truncation are used to specify λ_0 and κ_0, as defined in Appendix A of the online supplementary material. The use of conservative bounds then will lead to low acceptance probabilities for the JBEA algorithm.

We can also reduce the computational cost of JBEA by making its proposal as similar as possible to the target. This is also a good idea when using the algorithms from Section 4—it would reduce the variance of the weights in the ISMCEM algorithm and increase the success probability of C_2 in the MCMC one. The idea is basically to make the proposal jump process (jump rate and jump size distribution) depend on time and/or on the initial and ending values of the interval whenever the target jump process is time and/or state dependent. The time dependence can always be mimicked from the target and the state replaced by a function of the initial and ending values, for example, their mean.

5 Simulated examples

In this section, we present results from some simulated examples. Firstly, we present an example with bounded drift and bounded jump-rate to apply the algorithms from Section 3. Secondly, the algorithms from Section 4 are applied to an unbounded drift example. Three data sets are simulated from each model and analysed with the respective algorithms. The three replications in each example present similar results when applying the proposed methodologies. For that reason, results for only one of them are reported here whilst results for the other ones are reported in Appendix H of the online supplementary material.

5.1 An introductory example

We consider the following model:

$$\begin{aligned} dV_s &= -\tanh(V_s-\delta)ds + \sigma dW_s + dJ_s, \quad V_0 = v, \quad s \in [0, 1{,}000], \\ \lambda_1(s; V_{s-}) &= \lambda(1-\tanh^2(V_s-\delta)), \quad g_1(Z_j, V_{t_j-}) = Z_j \sim \mathcal{N}(\mu, \tau^2), \end{aligned} \tag{41}$$

with $(\delta, \sigma^2, \lambda, \mu, \tau^2) = (0, 1, 0.1, 2, 0.35^2)$. We chose these parameters to be potentially problematical for the algorithm, as there will be identifiability problems in distinguishing the presence of jumps from continuous volatility. The MCEM algorithm of Section 3.1.2 and the MCMC algorithm of Section 3.2 are applied to the same dataset and results are presented in Tables 1 and 2, respectively. The M-step from the MCEM algorithm is performed numerically via the quasi-Newton method BFGS (see, e.g., Fletcher, 1987, Section 3.2).

The data set presents 56 jumps, with mean 2.125 and variance 0.104. The distance between the MLE estimates and the posterior means in terms of the respective posterior standard deviations (how many posterior s.d.'s the MLE estimate is from the posterior mean) are 0.51 for δ, 0.11 for σ^2, 0.73 for λ, 0.83 for μ, and 1.06 for τ^2.

The MCMC algorithm also outputs a sample from the posterior distribution of the jump process. Some interesting posterior statistics can be obtained from this distribution. Figure 1 shows an example.

We also run the algorithms fixing parameters μ and τ^2 at their real values. Table 3 shows the results obtained for the two algorithms. They reinforce the issues concerning identifiability—note the improvement in the estimation of the jump rate by the MCMC algorithm. Without fixing those parameters, the posterior distribution estimates a higher average number of jumps with smaller average size.

5.2 An unbounded drift example: the Ornstein Uhlenbeck process

We now present a simulated example where the drift is unbounded and the methodology from Section 4 is applied. We consider the following model:

$$\begin{aligned} dV_s &= -\rho(V_s-\mu)ds + dW_s + dJ_s, \quad V_0 = v, \quad s \in [0, 500], \\ \lambda_1(s; V_{s-}) &= \lambda, \quad g_1(Z_j, V_{t_j-}) = Z_j \sim \text{Exp}(\theta), \end{aligned} \tag{42}$$

with $(\rho, \mu, \lambda, \theta) = (1, 0, 0.07, 1)$. Identifiability problems are likely to occur due to the ambiguity involving variation of the continuous part and small jumps from the exponential distribution. We consider the (constant) diffusion coefficient to be known. Although the proposed methodologies can be used to estimated parameters in the diffusion coefficient, identifiability issues would be severely worsen. In real application, in which assuming the diffusion coefficient to be known may be not reasonable, we can use some specific strategies to mitigate the identifiability problem as it is done for the example in Section 6.

The data set presents 36 jumps, with mean $(1/0.733) = 1.364$. Results from the MCEM algorithm of Section 4.1 are presented in Table 4, considering the estimator $\dot{E}_3$. The M-step is performed analytically. An algorithm based on estimator E_1 for fixed λ was also implemented and the Monte Carlo variance was too high even for 8×10^5 samples.

The convergence of the MCEM algorithm is slow, caused by irregularities of the likelihood function related to the identifiability problems described above. If we fix the jump rate at its real value,

Table 1. Iterations from the MCEM algorithm

Iteration	δ	σ^2	λ	μ	τ^2
0	0.5	1.5	0.2	2	0.25
1	0.149	1.299	0.104	1.771	0.212
5	0.078	1.128	0.096	1.685	0.197
20	0.023	1.031	0.128	1.770	0.203
100	0.005	0.996	0.133	1.848	0.175
200	0.008	0.996	0.128	1.889	0.146
500	0.019	1.001	0.120	1.942	0.100
Real	0	1	0.1	2	0.1225

Note. 1,000 Monte Carlo samples are used in all iterations. Results obtained with other initial conditions (away from the real values) suggest the presence of local modes.

we get the estimates $\hat{\rho} = 1.019$, $\hat{\mu} = 0.112$, and $\hat{\theta} = 0.780$ (after just seven iterations) and the estimated covariance matrix

$$\begin{pmatrix} 0.0025367 & 0.00022448 & -0.0011035 \\ 0.00022448 & 0.0015771 & 0.00059542 \\ -0.0011035 & 0.00059542 & 0.038611 \end{pmatrix}.$$

For the MCMC algorithm of Section 4.2, an EA step between observation, jump and refinement points is performed after each Barker's step. The chain is warmed-up for $5k$ iterations before the parameters start to be updated. The layer refinement idea described in Section 4.2.3 is applied for $m = 4$. The parameter vector is broken into four individual blocks. Parameters from the jump process (λ, θ) are sampled directly from their full conditionals and the other two parameters are sampled via Two-Coin Barker's. We adopt uniform prior distributions for μ and ρ and the following informative priors: $\lambda \sim \text{Exp}(50)$ and $\theta \sim \mathcal{G}\text{amma}(7, 6)$. As it is explained in Section 4.3, these informative priors play a crucial role in the identifiability of the model. They work in the direction of favouring a few larger jumps instead of many smaller jumps.

We set uniform random walk proposals for μ and ρ having a higher acceptance rate then optimal but perform six alternate updates of each one on each iteration of the MCMC. Also, three consecutive updates of X are performed on each iteration. Trace plots indicate good convergence. Posterior statistics are presented in Table 5. The chosen prior distributions seem to correct the irregularities of the likelihood and lead to reasonably good results.

6 Application

6.1 Exchange rate USD × GBP

We consider the exchange rate between USD and GBP. The USD suffered a considerable depreciation during the 2008 world economic crisis. A few months later it had a moderate recovery and oscillated between 1.45 and 1.7 until beginning of 2016. We will use daily data from 21 May 2009, which was right after the recovery, up to 27 March 2013. This constitutes 1201 data points and is shown in Figure 2 in the log-scale. We shall adopt the MCMC procedure of Section 3.2.

We choose to fit a scaled Brownian motion to model the continuous part of the process and proceed as follows to specify the jump part: (1) Plot time versus differences between consecutive log-observations. (2) Identify outlier differences as the values outside an interval that is defined by the (constant along time) limits of a dense cloud of points. (3) Consider the outlier values to be indicators of jumps in the process and use them to empirically estimate the jump rate and jump size distribution. This analysis suggests a piecewise constant jump rate dividing the observed time interval in five parts and the absolute value of the jump sizes being well accommodated by a Gamma distribution, in particular, a $\mathcal{G}(8, 1, 000)$. A mixture of a positive and a negative gamma

Table 2. Posterior statistics from the MCMC output

	δ	σ^2	λ	μ	τ^2
Mean	−0.025	0.993	0.179	1.69	0.291
Median	−0.021	0.991	0.164	1.67	0.275
Mode	−0.023	0.987	0.135	1.54/1.66	0.025/0.280
SD	0.086	0.068	0.080	0.30	0.180
Real	0	1	0.1	2	0.1225

Note. Uniform improper priors are adopted for all parameters. The prior of τ^2 had to be truncated to be above 0.008 to avoid getting trapped in small values. The chain runs for $500k$ iterations. The parameters are jointly sampled using an adaptive Gaussian random walk MH step which had a 0.31 acceptance rate. Trace plots suggest convergence has been achieved. Parameters μ and τ^2 present two marginal modes each.

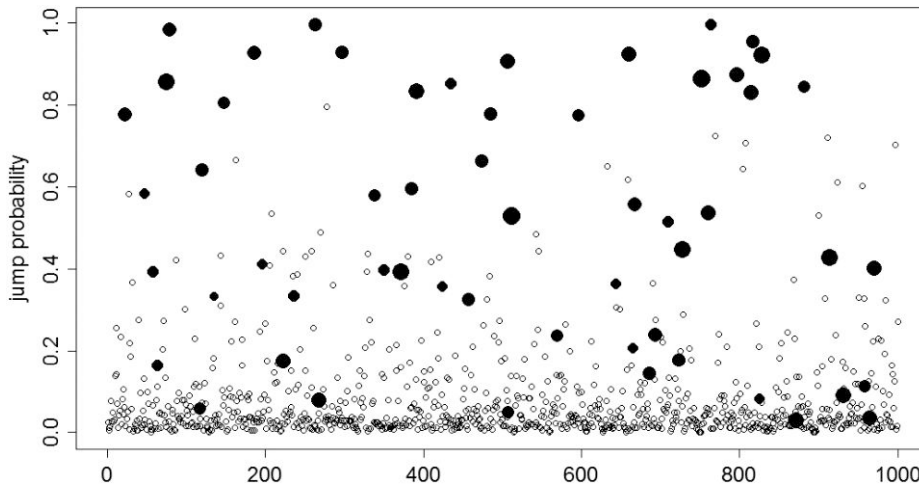

Figure 1. Posterior probability of jump between each pair of consecutive observations. Solid circles represent intervals where a jump really exists and have size proportional to the size of the jump.

Table 3. Results for the case where μ and τ^2 are fixed at their real values

	δ	σ^2	λ
MCEM	0.024	1.004	0.113
MCMC			
Mean	0.021	1.009	0.117
Median	0.022	1.007	0.115
Mode	0.024	1.004	0.113
SD	0.064	0.062	0.028
95% Cred. int.	(−0.107,0.146)	(0.895,1.137)	(0.066,0.177)
Real	0	1	0.1

Note. The specifications of the two algorithms are the same as in the previous run. The output of the MCEM algorithm corresponds to iteration 55, for which it has clearly converged.

distributions takes the probability mass of the jump size distribution away from zero and, consequently, helps to avoid identifiability problems. All of the empirical assumptions described in this paragraph are used only to elicit the model and are not actually imposed to the analysis. Naturally, some of the intervals could be likely to have more than one jump but, given the way the jump

Table 4. Iterations from the MCEM algorithm using estimator $\dot{E}_3$

Iteration	ρ	μ	λ	θ	MC samples ($\times 10^3$)
0	1.5	−0.5	0.5	1.5	
1	1.459	−0.231	0.651	1.245	20
10	1.185	−0.105	0.605	1.667	50
100	1.053	0.021	0.281	1.486	50
200	1.005	0.099	0.094	0.926	50
300	0.989	0.117	0.066	0.801	50
312	0.996	0.114	0.067	0.748	600
316	1.001	0.114	0.066	0.760	600
Real	1	0	0.07	1	

Table 5. Posterior statistics from the MCMC output

	ρ	μ	λ	θ
Mean	1.010	0.086	0.050	0.725
Median	1.010	0.086	0.046	0.709
Mode	1.013	0.086	0.039	0.667
SD	0.0719	0.051	0.025	0.189
95% Cred. int.	(0.872,1.152)	(−0.014,0.188)	(0.013,0.111)	(0.396,1.131)
Real	1	0	0.07	1

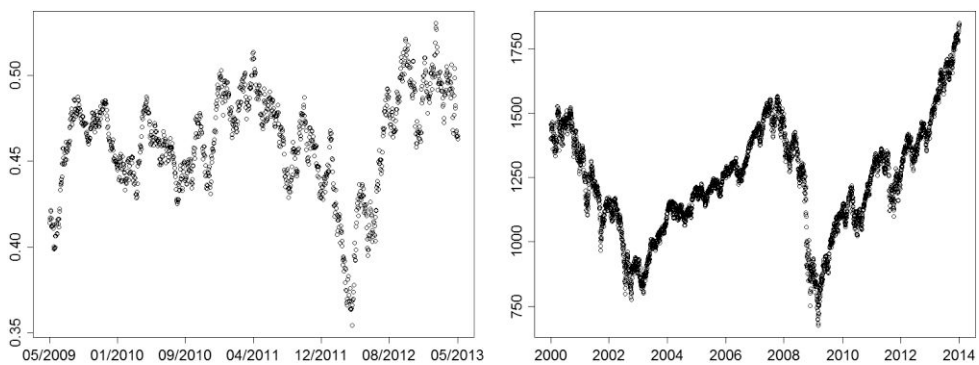

Figure 2. Left: USD×GBP log-rate, source: http://www.Investing.com. Right: S&P500 index, source: http://research.stlouisfed.org.

distribution is chosen and the fact that there are no restrictions in the number of jumps, this should happen only to a minority of the intervals estimated to have jumps and, therefore, not compromise the analysis.

The chosen model for the log-rate V is given by

$$\begin{aligned} dV_s &= bdW_s + dJ_s, \quad V_0 = v, \quad s \in [0, 1{,}201], \\ \lambda_1(s; V_{s-}) &= \lambda_i \quad \text{for } s \in A_i, \quad i = 1, \ldots, 5, \\ g_1(Z_j, V_{t_j-}) &= Z_j \sim p\mathcal{G}(8, 1{,}000) + (1-p)(-\mathcal{G}(8, 1{,}000)), \end{aligned} \tag{43}$$

Table 6. Posterior statistics of the parameters for the last $40k$ iterations

	b^2	λ_1	λ_2	λ_3	λ_4	λ_5	p
Mean	0.000017	0.269	0.069	0.014	0.008	0.025	0.481
Median	0.000017	0.267	0.067	0.010	0.005	0.019	0.482
SD	0.0000014	0.046	0.029	0.013	0.008	0.022	0.059

for $A_1 = [0, 370]$, $A_2 = (370, 825]$, $A_3 = (825, 1{,}000]$, $A_4 = (1{,}000, 1{,}120]$, and $A_5 = (1{,}120, 1{,}201]$.

We try to avoid identifiability problems by minimising the number of jumps and maximising b. This is done by setting informative priors to all the λ_i's and by adopting the jump size distribution in (43). Parameters are assumed to be mutually independent with marginal priors: $b^2 \sim \mathcal{U}(0, \infty)$, $\lambda_i \sim \mathcal{G}(1, 50)$, for all i, $p \sim \mathcal{U}(0, 1)$.

It is enough to simulate only jump times and sizes in JBEA to derive the full conditional distributions of the parameters, which all have closed forms. Note, however, that although this is a fairly simple model, exact inference is only feasible due to the JBEA algorithm.

We start the chain at values $b^2 = 0.002^2$, $\lambda_1 = 0.40$, $\lambda_2 = 0.28$, $\lambda_3 = 0.20$, $\lambda_4 = 0.07$, $\lambda_5 = 0.22$, $p = 0.5$, and run $50k$ iterations. Standard diagnostics suggest that convergence is rapidly attained. Table 6 shows the posterior statistics of the parameters.

6.2 S&P500

We now apply the methodology from Section 3.1.2 to fit the Pareto-Beta Jump-Diffusion (PBJD) to daily data from the S&P500 index. The PBJD model was proposed in Ramezani and Zeng (1998) to model stock price behaviour. It allows for up and down jumps using a mixture jump size distribution to account for good and bad news. The model is the following:

$$\begin{gathered} dV_s = \mu V_s ds + \sigma V_s dW_s + dJ_s, \quad V_0 = v, \\ \lambda_1(s; V_{s-}) = \lambda, \quad g_1(Z_j, V_{t_j-}) = (Z_j - 1)V_{t_j-}, \\ Z_j \sim p\text{Pareto}(\eta_u) + (1-p)\text{Beta}(\eta_d, 1), \end{gathered} \tag{44}$$

where the Pareto distribution takes values in $(1, \infty)$.

We consider daily data from 3 January 2000 to 31 December 2013 which consists of 3532 observations. This period incorporates the WTC 09/11 episode in 2001 and the 2008 economic crisis. The data are shown in Figure 2. The two periods mentioned are clear in the graph and it seems reasonable to assume $\mu = 0$, which considerably simplifies the algorithm. We apply the MCEM algorithm from Section 3.1. Initial values are chosen through an empirical analysis of the data. Results are presented in Table 7.

The estimated covariance matrix of the MLE returned a negative variance for parameter p. In order to resolve this we use a different parameterisation to compute this estimate. We make $\lambda_u = p\lambda$ and $\lambda_d = (1-p)\lambda$—the rates of up and down jumps, respectively.

7 Conclusions

In this paper, we looked at the challenging problem of likelihood-based statistical inference for discretely observed jump-diffusion processes. Our main contribution is to provide, for the first time, a comprehensive suite of methodologies for both (likelihood-based) frequentist and Bayesian *exact* inference for discretely observed jump-diffusion processes.

Our simplest methods are based on the JBEA algorithm for jump-diffusion bridge simulation and are presented in Section 3. However, more general methods are required for the case of unbounded drift and jump-rate for the (transformed) process. To this end, we develop a more general approach in Section 4 which involve the construction of novel IS and Barker MCMC algorithms. Both these approaches are useful: the JBEA approach is far more computationally efficient, while the methods presented in Section 4 are more generally applicable.

Table 7. Maximum likelihood estimate and asymptotic 95% confidence interval for the parameters

	σ	λ	p	λ_u	λ_d	η_u	η_d
MLE	0.00577	0.841	0.532	0.447	0.394	144.5	125.2
C.I. 95%	(0.00497, 0.00656)	–	–	(0.342, 0.552)	(0.291, 0.497)	(124.8, 164.2)	(104.9, 145.5)

We also address some important identifiability issues related to the inference problem and some practical implementation strategies. In particular, we propose (see Appendix E of the online supplementary material) an algorithm to simulate upper and lower bounds for a Brownian bridge and to simulate the bridge given these bounds and possibly other bridge points. We also concluded that prior distributions play a crucial role to solve identifiability problems. The Barker MCMC introduced in Section 4.2 seems also to be of independent interest.

The methods were empirically tested in some simulated examples and performed robustly and efficiently. The examples also addressed key identifiability issues. Finally, jump-diffusion models were used to fit some financial data.

Although beyond the scope of this paper, there is no intrinsic reason why this approach cannot be developed for many multidimensional contexts, at least for relatively small-dimensional models. However, note that the exact simulation methodology that is used here, is fundamentally limited to reducible, gradient drift diffusions. Moreover, the computational overheads associated with moderately high-dimensional problems are likely to be prohibitive.

Although we give the first general efficient methodology for exact likelihood-based inference for discretely observed jump-diffusions in this paper, we also acknowledge the restrictions and complexity involved in the methodology, which reflects the complexity of the inference problem. Most importantly though, we hope that our work will stimulate further work on our ambitious aim to solve the problem exactly.

Acknowledgments

We would particularly like to thank the two anonymous referees who provided excellent and detailed comments on earlier versions of this paper.

Supplementary material

Supplementary material are available at *Journal of the Royal Statistical Society: Series B* online.

Conflict of interests: None declared.

Funding

F.B.G. would like to thank FAPEMIG—grants PPM-00745-18 and APQ-01837-22, CNPq—grant 310433/2020-7 and the University of Warwick for financial support. K.Ł. is supported by the Royal Society through the Royal Society University Research Fellowship. G.O.R. is supported by the EPSRC grants—*ilike* (EP/K014463/1), CoSInES (EP/R034710/1) and Bayes for Health (EP/R018561/1).

Data availability

Computer codes and data are available upon request to the corresponding author.

References

Agrawal S., Vats D., Łatuszyński K., & Roberts G. O. (2021). 'Optimal scaling of MCMC beyond Metropolis', to appear in *Advances in Applied Probability*.

Aït-Sahalia Y., & Yu J. (2006). Saddlepoint approximations for continuous-time Markov processes. *Journal of Econometrics*, *134*(2), 507–551. https://doi.org/10.1016/j.jeconom.2005.07.004

Ball C. A., & Roma A. (1993). A jump diffusion model for the European monetary system. *Journal of International Money and Finance*, *12*(5), 475–492. https://doi.org/10.1016/0261-5606(93)90035-A

Barker A. A. (1965). Monte Carlo calculations of the radial distribution functions for a protonelectron plasma. *Australian Journal of Physics*, *18*(2), 119–133. https://doi.org/10.1071/PH650119
Barndorff-Nielsen O. E., & Shephard N. (2004). Power and bipower variation with stochastic volatility and jumps (with discussion). *Journal of Financial Econometrics*, *2*(1), 1–48. https://doi.org/10.1093/jjfinec/nbh001
Beskos A., Papaspiliopoulos O., & Roberts G. O. (2006). Retrospective exact simulation of diffusion sample paths with applications. *Bernoulli*, *12*(6), 1077–1098. https://doi.org/10.3150/bj/1165269151
Beskos A., Papaspiliopoulos O., & Roberts G. O. (2008). A new factorisation of diffusion measure and sample path reconstruction. *Methodology and Computing in Applied Probability*, *10*(1), 85–104. https://doi.org/10.1007/s11009-007-9060-4
Beskos A., Papaspiliopoulos O., & Roberts G. O. (2009). Monte Carlo maximum likelihood estimation for discretely observed diffusion processes. *The Annals of Statistics*, *37*(1), 223–245. https://doi.org/10.1214/07-AOS550
Beskos A., Papaspiliopoulos O., Roberts G. O., & Fearnhead P. (2006). Exact and computationally efficient likelihood-based inference for discretely observed diffusion processes (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *68*(3), 333–382. https://doi.org/10.1111/j.1467-9868.2006.00552.x
Bruti-Liberati N., & Platen E. (2007). Approximation of jump diffusions in finance and economics. *Computational Economics*, *29*(3–4), 283–312. https://doi.org/10.1007/s10614-006-9066-y
Casella B., & Roberts G. O. (2010). Exact simulation of jump-diffusion processes with Monte Carlo applications. *Methodology and Computing in Applied Probability*, *12*(3), 449–473. https://doi.org/10.1007/s11009-009-9163-1
Chen N. (2009). *Localization and exact simulation of Brownian motion driven stochastic differential equations*. Working Paper, Chinese University of Hong Kong.
Chudley C. T., & Elliott R. J. (1961). Neutron scattering from a liquid on a jump diffusion model. *Proceedings of the Physical Society*, *77*(2), 353–361. https://doi.org/10.1088/0370-1328/77/2/319
Cont R., & Tankov P. (2004). *Financial modelling with jump processes*. Chapman & Hall/CRC Financial Mathematics Series. Chapman & Hall/CRC.
Duffie D., & Glynn P. (2004). Estimation of continuous-time Markov processes sampled at random time intervals. *Econometrica*, *72*(6), 1773–1808. https://doi.org/10.1111/j.1468-0262.2004.00553.x
Duffie D., Pan J., & Singleton K. (2000). Transform analysis and asset pricing for affine jump-diffusions. *Econometrica*, *68*(6), 1343–1376. https://doi.org/10.1111/1468-0262.00164
Duffie D., & Singleton K. J. (1993). Simulated moments estimation of Markov models of asset prices. *Econometrica*, *61*(4), 929–952. https://doi.org/10.2307/2951768
Durham G. B., & Gallant R. A. (2002). Numerical techniques for maximum likelihood estimation of continuous-time diffusion processes. *Journal of Business & Economic Statistics*, *20*(3), 297–338. https://doi.org/10.1198/073500102288618397
Elerian O. (1999). *Simulation estimation of continuous time series models with applications to finance* [Ph.D. thesis]. Nuffield College, Oxford.
Eraker B. (2004). Do stock prices and volatility jump? Reconciling evidence from spot and option prices. *The Journal of Finance*, *59*(3), 1367–1403. https://doi.org/10.1111/j.1540-6261.2004.00666.x
Eraker B., Johannes M. S., & Polson N. G. (2003). The impact of jumps in volatility and returns. *The Journal of Finance*, *58*(3), 1269–1300. https://doi.org/10.1111/1540-6261.00566
Fearnhead P., Papaspiliopoulos O., & Roberts G. O. (2008). Particle filters for partially observed diffusions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *70*(4), 755–777. https://doi.org/10.1111/j.1467-9868.2008.00661.x
Feng L., & Linetsky V. (2008). Pricing options in jump-diffusion models: An extrapolation approach. *Operations Research*, *56*(2), 304–325. https://doi.org/10.1287/opre.1070.0419
Filipović D., Mayerhofer E., & Schneider P. (2013). Density approximations for multivariate affine jump-diffusion processes. *Journal of Econometrics*, *176*(2), 93–111. https://doi.org/10.2139/ssrn.1851511
Fletcher R. (1987). *Practical methods of optimization* (2nd ed.). John Wiley & Sons.
Fort G., & Moulines E. (2003). Convergence of the Monte Carlo expectation maximization for curved exponential families. *The Annals of Statistics*, *31*(4), 1220–1259. https://doi.org/10.1214/aos/1059655912
Giesecke K., & Schwenkler G. (2019). Simulated likelihood estimators for discretely observed jump-diffusions. *Journal of Econometrics*, *213*(2), 297–320. https://doi.org/10.1016/j.jeconom.2019.01.015
Golightly A. (2009). Bayesian filtering for jump-diffusions with application to stochastic volatility. *Journal of Computational and Graphical Statistics*, *18*(2), 384–400. https://doi.org/10.1198/jcgs.2009.07137
Gonçalves F. B., & Franklin P. (2019). 'On the definition of the likelihood function', arXiv, arXiv:1906.10733, preprint: not peer reviewed.
Gonçalves F. B., & Roberts G. O. (2014). Exact simulation problems for jump-diffusions. *Methodology and Computing in Applied Probability*, *16*(4), 907–930. https://doi.org/10.1007/s11009-013-9330-2
Grenander U., & Miller M. I. (1994). Representations of knowledge in complex systems. *Journal of the Royal Statistical Society. Series B*, *56*(4), 549–581. https://doi.org/10.1111/j.2517-6161.1994.tb02000.x

Johannes M. (2004). The statistical and economic role of jumps in continuous-time interest rate models. *The Journal of Finance*, *59*(1), 227–260. https://doi.org/10.1111/j.1540-6321.2004.00632.x

Johannes M. S., Polson N. G., & Stroud J. R. (2002). Nonlinear filtering of stochastic differential equations with jumps. Available at SSRN: http://ssrn.com/abstract=334601 or doi:10.2139/ssrn.334601.

Johannes M. S., Polson N. G., & Stroud J. R. (2009). Optimal filtering of jump diffusions: Extracting latent states from asset prices. *Review of Financial Studies*, *22*(7), 2759–2799. https://doi.org/10.1093/rfs/hhn110

Kennedy J. S., Forsyth P. A., & Vetzal K. R. (2009). Dynamic hedging under jump diffusion with transaction costs. *Operations Research*, *57*(3), 541–559. https://doi.org/10.1287/opre.1080.0598

Łatuszyński K. G., & Roberts G. O. (2013). CLTs and asymptotic variance of time sampled Markov chains. *Methodology and Computing in Applied Probability*, *15*(1), 237–247. https://doi.org/10.1007/s11009-011-9237-8

Lo A. W. (1988). Maximum likelihood estimation of generalized Itô processes with discretely sampled data. *Econometric Theory*, *4*(2), 231–247. https://doi.org/10.1017/S0266466600012044

Meng X. L., & Rubin D. B. (1993). Maximum likelihood estimation via the ECM algorithm: A general framework. *Biometrika*, *80*(2), 267–278. https://doi.org/10.1093/biomet/80.2.267

Pedersen A. (1995). A new approach to maximum likelihood estimation for stochastic differential equations based on discrete observations. *Scandinavian Journal of Statistics*, *22*(1), 55–71.

Peskun P. H. (1973). Optimum Monte Carlo sampling using Markov chains. *Biometrika*, *60*(3), 607–612. https://doi.org/10.1093/biomet/60.3.607

Platen E., & Bruti-Liberati N. (2010). *Numerical solution of stochastic differential equations with jumps in finance*. Springer.

Protter P. E. (2004). *Stochastic integrations and differential equations* (2nd ed.). Springer.

Ramezani C., & Zeng Y. (1998, December). *Maximum likelihood estimation os asymmetric jump-diffusion processes: Application to security prices*. Working Paper, Department of Mathematics and Statistics, University of Missouri.

Roberts G. O., & Stramer O. (2001). On inference for partially observed nonlinear diffusion models using the Metropolis–Hastings algorithm. *Biometrika*, *88*(3), 603–621. https://doi.org/10.1093/biomet/88.3.603

Runggaldier W. J. (2003). Jump-diffusion models. In *Handbook of heavy tailed distributions in finance* (pp. 169–209). Handbooks in Finance, Book 1. Elsevier/North-Holland.

Sermaidis G., Papaspiliopoulos O., Roberts G. O., Beskos A., & Fearnhead P. (2013). Markov chain Monte Carlo for exact inference for diffusions. *Scandinavian Journal of Statistics*, *40*(2), 294–321. https://doi.org/10.1111/j.1467-9469.2012.00812.x

Srivastava A., Grenander U., Jensen G. R., & Miller M. I. (2002). Jump-diffusion Markov processes on orthogonal groups for object recognition. *Journal of Statistical Planning and Inference*, *103*(1–2), 15–37. https://doi.org/10.1016/S0378-3758(01)00195-1