Exact Simulation of Jump-Diffusion Processes with Monte Carlo Applications

Casella, Bruno and Roberts, Gareth O.

University of Warwick

1 September 2011

Submitted to the Annals of Applied Probability

EXACT SIMULATION OF JUMP-DIFFUSION PROCESSES WITH MONTE CARLO APPLICATIONS*

By Bruno Casella† , and Gareth Roberts†

University of Warwick

We introduce a novel algorithm (JEA) to simulate exactly from a class of one-dimensional jump-diffusion processes with state-dependent intensity. The simulation of the continuous component builds on the recent Exact Algorithm ((1)). The simulation of the jump component instead employes a thinning algorithm with stochastic acceptance probabilities in the spirit of (14). In turn JEA allows unbiased Monte Carlo simulation of a wide class of functionals of the process' trajectory, including discrete averages, max/min, crossing events, hitting times. Our numerical experiments show that the method outperforms Monte Carlo methods based on the Euler discretization.

1. Introduction. The purpose of this paper is twofold.

1. We present an algorithm (the Jump Exact Algorithm or JEA) to simulate exactly from a general family of one-dimensional jump diffusion processes with state-dependent intensity. The algorithm can be regarded as a generalisation of the Exact algorithm (EA) developed in (4) and (1) for the simulation of diffusion processes.
2. We apply the Jump Exact Algorithm to develop suitable Monte Carlo algorithms for the simulation of a number of jump-diffusions' functionals of interest in financial applications.

Jump-diffusion processes are useful modelling tools in many applied areas ranging from neurobiology (13) to adaptive control (5) and computer science (26). In particular they have become increasingly popular in financial modeling since Merton seminal paper (20) thanks to their ability to account for some empirically observed effects like heavy tails of the returns' distribution and volatility smiles. Recently, there has been growing interest for jump-diffusion models in structural credit risk modeling (6, 25). In this context jump diffusions are able to support the empirical evidence that short

*Research supported by EPSRC.

†Both authors are indebted to Alex Beskos, Omiros Papaspiliopoulos and Stefano Peluchetti for many stimulating discussions.

AMS 2000 subject classifications: Primary 60K30; secondary 65C05

Keywords and phrases: Jump Diffusion, Simulation, Exact Algorithms, Barrier Option Pricing

maturity credit spreads are bounded away from 0. Generally speaking, by choosing the parameters of the jump process appropriately, we can generate a very wide variety of dynamics incorporating relevant empirically observed effects that otherwise would not be explained by traditional diffusion-based models. However computing with jump-diffusion processes, for example for pricing derivatives or determining default probabilities, is mathematically very challenging and it usually involves some kind of approximation. In practice most quantities of financial interest can be represented as expectations of functionals of the underlying jump-diffusion and can be estimated within a Monte Carlo simulation framework. Typically this involves the employment of a discretization scheme in order to simulate the process trajectory in between any two jumps. If the jump component is independent from the diffusion component, jumps' times and sizes can be simulated exactly and the discretization error affecting the Monte Carlo estimator is expected to be the same as for pure diffusions. Instead when the intensity of the marked point process driving the jump component is state-dependent, the discretization error generated in the simulation of the continuous component reflects on the simulation of the jumps, thus resulting in larger bias of the Monte Carlo estimator. In a recent paper (14) develope an interesting simulation procedure to deal with this case. Their method is based on a thinning algorithm with state dependent acceptance probability for the simulation of the jump component and a discretization scheme for the simulation of the diffusion component. The authors prove that the weak convergence order of their scheme equals the order of the employed discretization in a restricted context where 1) the target functional depends on the realization of the diffusion trajectory at only one fixed time point; 2) the target functional is uniformly bounded.

In this paper, after introducing the class of jump-diffusion models of interest (Section 2) and the simulation problem (Section 3), we present a novel algorithm (JEA) to simulate exactly a Skeleton from a general family of one-dimensional jump diffusion processes with a state-dependent intensity (Section 4). The algorithm can be seen as a combination of the *Exact Algorithm* and the *thinning algorithm*. The whole procedure is sequential. At each jump epoch the starting point of the Exact Algorithm is updated according to the output of the thinning experiment; conversely, the thinning probability of a candidate jump depends on the current state of the process simulated by the Exact Algorithm. Unlike (14), bias does not affect neither the simulation of the diffusion component nor the simulation of the jump component. As a consequence JEA algorithm allows unbiased Monte

Carlo simulation of a large class of target functionals, including challenging path-dependent functionals. This is exemplified in Section 5 where we apply the JEA framework to simulate Monte Carlo prices of some financial derivatives. The key is the Brownian bridge decomposition: the Skeleton generated by the JEA algorithm decomposes the jump-diffusion process in a sequence of independent Brownian bridges. In a Monte Carlo context this allows us to take advantage of the many distributional results related to Brownian bridges' functionals. Some of them are briefly reviewed in Appendix A. Results from the numerical simulation reported in Section 6 are fully satisfactory. In fact on the one side they support our main theoretical statement, confirming the unbiasedeness of the Monte Carlo estimator; on the other side they motivate a practical interest in JEA techniques showing that, if a good level of approximation is required, JEA estimator is computationally more efficient than competing estimators based on discretisation schemes.

2. Jump-diffusion processes with state-dependent intensities. Let $V := \{V_t : 0 \le t \le T\}$ be a one-dimensional jump-diffusion process solving the SDE:

$$dV_t = \mu(V_{t^-})dt + \sigma\,(V_{t^-})\,dW_t + \int_E g(z, V_{t^-})m(dz, dt), \qquad V_0 = v_0 \tag{1}$$

where μ, $\sigma : \mathbf{R} \to \mathbf{R}$ and $g : \mathbf{R} \times E \to \mathbf{R}$ are presumed to satisfy the appropriate Lipschitz and linear growth conditions for the existence of a unique weak solution (16). Here $m(dz, dt)$ is a random counting measure on the product space $E \times [0, T]$ with $E \subseteq \mathbf{R}$ with associated intensity measure λ_m. We assume that λ_m is absolutely continuous with respect to the Lebesque measure on $M \times [0, T]$ and Markov-dependent on V:

$$\lambda_m\,(dz, dt; V_{t^-}) \;=\; \lambda_m(z, t; V_{t^-})dz\,dt \;=\; \lambda(t;\, V_{t^-})\,f_Z(z;\, t)\,dz\,dt \tag{2}$$

where, for any $v \in \mathbf{R}$, $\lambda\,(\cdot; v)$ is a positive real valued function on $[0, T]$ and for any $t \in \mathbf{R}^+$, $f_Z\,(\cdot; t)$ is a standard density function with support E. According to (1) and (2), between any two jumps the process V behaves as a homogeneus diffusion process with drift μ and diffusion coefficient σ. Jumps' times and sizes are state-dependent. The jumps' times, say $(t_1, t_2, \ldots, t_M)$, are generated by a renewal process on $[0, T]$ with intensity function $\lambda(t\,; V_{t^-})$. A random variable $Z_i \sim f_Z(z\,; t_i)$ is associated to each

jump time t_i, $i = 1, 2, \ldots, M$. In turn Z_i and the state of the process $V_{t_i^-}$ determine the amplitude $g(Z_{t_i}, V_{t_i^-})$ of the jump in t_i. By an appropriate choice of the functions μ, σ, g, λ, and f, we can generate a wide variety of stochastic dynamics and, more crucially, we can incorporate information on the state of the system both in the continuous component and in the jump component.
We consider the problem of simulating exact (finite information on the) sample paths of V and the related problem of unbiased Monte Carlo estimation of:

$$\nu := \mathbb{E}\left(\phi(V) \mid V_0 = v_0\right). \tag{3}$$

for general, eventually path-dependent, functional ϕ. However, preliminarly, we shall introduce the transformed process $X := \{X_t : 0 \leq t \leq T\}$:

$$X_t = \eta\left(V_t\right) = \int_v^{V_t} \frac{1}{\sigma(u)} du \tag{4}$$

where v is an arbitrary element of the state-space of V. We apply the generalized Ito's Lemma for jump-diffusion processes to find the SDE of X. From (1) we derive:

$$dX_t = \alpha\left(X_{t^-}\right) dt + dW_t + \int_E \Delta_\eta\left(z, X_{t^-}\right) m\left(dz,\, dt\right), \quad X_0 = \eta(v_0) := x_0 \tag{5}$$

where, for $x \in \mathbf{R}$, $\alpha(x) := \frac{\mu\left(\eta^{-1}(x)\right)}{\sigma(\eta^{-1}(x))} - \frac{1}{2}\sigma'\left(\eta^{-1}(x)\right)$ and

$$\Delta_\eta\left(z, x\right) := \eta\left(\eta^{-1}(x) + g\left(z, \eta^{-1}(x)\right)\right) - x \tag{6}$$

with λ_m defined as in (2). Thus (5) and (2) define the dynamics of the transformed process $X = \eta(V)$. Transformation (4) turns (1) into a SDE with unit diffusion coefficient. As unit diffusion coefficient is a pre-requisite for the application of the Jump Exact Algorithm we shall work on (5) rather than (1). However, since $\eta(\cdot)$ is invertible we can easily recover information on the trajectories of V from the simulation of the process X by means of the inverse transformation $V_t = \eta^{-1}(X_t)$.

3. The simulation problem.

3.1. *An introductory example: Merton model.* To introduce the simulation problem we analyze briefly the standard Merton's model. Merton's model arises as a particular case of (1) and (2):

$$dV_t = \mu\, V_{t^-}\, dt + \sigma\, V_{t^-}\, dW_t + V_{t^-} \int_{\mathbf{R}} (e^z - 1)\; m(dz, dt); \quad V_0 \;=\; v_0 \tag{7}$$

with

$$\lambda_m(z,t) = \lambda\, f_{\mathcal{N}_{\gamma,\,\beta}}(z) \tag{8}$$

where λ is a strictly positive constant and $f_{\mathcal{N}_{\gamma,\,\beta}}(\cdot)$ is a gaussian density with mean γ and variance β^2. Thus the continuous component is driven by a geometric Brownian motion; the jumps' times $(t_1, t_2, \ldots, t_M)$ are distributed according to a homogeneous Poisson process with intensity function λ; for any $i = 1, 2, \ldots, M$, the jump corresponding to t_i is given by $J_i = e^{Z_i} - 1$ where $Z_1, Z_2, \ldots, Z_M$ are i.i.d. normal random variables with mean γ and variance β^2. This modelisation of the jumps preserves the positivity of the process V. We apply transformation (4) to (7) and (8) and find the following SDE representation for the transformed process $X_t \;=\; \eta\,(V_t) = \frac{1}{\sigma}\,(\log V_t - \kappa)$ (κ fixed constant):

$$dX_t = \left(\frac{\mu}{\sigma} - \frac{\sigma}{2}\right) dt + dW_t + \int_{\mathbf{R}} z\, m(dz,\, dt); \qquad X_0 = x_0 := \frac{1}{\sigma}\,(\log v_0 - \kappa) \tag{9}$$

with λ_m defined as in (8). From a simulation perspective the model (9)-(8) has very attractive properties. In particular we can construct a simple and efficient algorithm to simulate from (9) just by simulating in advance the jumps' times from a Poisson process ($\mathbb{PP}$) and then the diffusion dynamics between any two jumps from a Brownian motion with drift $\alpha := \frac{\mu}{\sigma} - \frac{\sigma}{2}$. Algorithm 1 returns a partial finite realisation (a "Skeleton") of the process X defined over the time interval $[0, T]$ with starting point x_0. We represent it as:

$$\mathcal{S}_0(x_0; 0, T) := \left\{(t_0, x_{t_0}), (t_1^-, x_{t_1^-}), (t_1, x_{t_1}), \ldots, (t_M^-, x_{t_M^-}),\; (t_M, x_{t_M}), (t_{M+1}, x_{t_{M+1}})\right\} \tag{10}$$

Algorithm 1 Merton model

1. `Simulate the jump times:` $(t_1, t_2, \dots t_M) \sim \mathbb{PP}([0,T],\lambda)$

2. `Set` $t_0 = 0$, $t_{M+1} = T$. `For any` $i = 0, 1, \dots, M$ `repeat the following:`

   2.1. `simulate` $X_{t_{i+1}^-} = X_{t_i} + \alpha(t_{i+1} - t_i) + \xi_{i+1}$ `with` $\xi_{i+1} \sim \mathcal{N}(0, (t_{i+1} - t_i))$

   2.2. `simulate` $X_{t_{i+1}} = Z_{i+1} + X_{t_{i+1}^-}$ with $Z_{i+1} \sim \mathcal{N}\left(\gamma, \beta^2\right)$

with $t_0 := 0$ and $t_{M+1} := T$. We can simulate further information on the process X thanks to the following representation, immediately derived from elementary Brownian motion constructions:

$$X \mid \mathcal{S}_0(x_0; 0, T) \sim \bigotimes_{i=0}^{M} \mathbb{BB}(t_i, x_{t_i}; t_{i+1}^-, x_{t_{i+1}^-}) \tag{11}$$

where we have denoted by $\mathbb{BB}(s_1, a; s_2, b)$ the measure of a Brownian bridge starting in a at time s_1 and ending in b at time s_2. The representation (11) of the conditional process $X \mid \mathcal{S}_0$ in terms of independent Brownian bridges is very convenient as the law of many functionals associated with the Brownian bridge are very well known (e.g. finite-dimensional distributions, law of max/min, crossing probabilities of a given barrier..). As an example, (21) apply the Brownian bridge factorization (11) to derive Monte Carlo prices of barrier options under Merton model.

3.2. *The general case.* In the general case (5)-(2) the simulation of X is far more challenging due to two main problems:

P.1: jump component. The jumps' times cannot be simulated in advance as they depend on the state of the process

P.2: diffusion component. Basic probabilistic properties, like transition densities, of the diffusion component in (5) usually are not known. This implies that:

- **(a)** we cannot simulate directly a skeleton (10) of the process X
- **(b)** the conditional law (11) of the process given the skeleton is not available

The Jump Exact Algorithm is designed to deal with these difficulties under rather general conditions.

4. The Jump Exact Algorithm.

4.1. *Constructing the Jump Exact Algorithm.*

4.1.1. *The jump component (Problem P.1).* We can break down the state-dependancy of the jumps' times by performing a simple transformation of the jumps' component in (5). To this end, we need to introduce our first assumption:

C.0 the function $\lambda(\cdot,\cdot)$ in (2) is bounded above by a constant λ

Let us denote by $R(t,x)$ the ratio $R(t,x) := \frac{\lambda(t;\eta^{-1}(x))}{\lambda} \leq 1$ and define $I(u,t,x) := \mathbf{I}_{\{[0,R(t,x)]\}}(u)$ with $u \in [0,1]$ where, for a given set A, $\mathbf{I}_A(u)$ is the indicator function of the event $\{u \in A\}$. Then Lemma 1 provides a suitable alternative representation of the process X (5)-(2).

Lemma 1
The jump-diffusion process X (5)-(2) is a solution of the following SDE:

$$dX_t = \alpha(X_{t^-})dt + dW_t + \int_{E^*} \Delta_\eta(z, X_{t^-})\, \mathbf{I}(u,t,X_{t^-})\, m^*(du,dz,dt) \tag{12}$$

where $X_0 = x_0$, $E^* = E \times [0,1]$ and m^* is a random Poisson measure with intensity:

$$\lambda_{m^*}(du,dz,dt) = \lambda f_Z(z;\,t)\, du\, dz\, dt \tag{13}$$

Proof:
We can represent the jump component in (12) in the following way:

$$\int_{E^*} \Delta_\eta(z, X_{t^-}) I(u,t;X_{t^-})\, m^*(du,dz,dt) = \int_E \Delta_\eta(z, X_{t^-}) \hat{m}(dz,dt) \tag{14}$$

with $\hat{m}(dz,dt) := \int_{[0,1]} I(u,t;X_{t^-})\, m^*(du,dz,dt)$. Using elementary facts from the renewal theory:

$$\lambda_{\hat{m}}(z,t) = \int_{[0,1]} I\left(u,t;X_{t^-}\right)\lambda f_Z(z;\,t)\,du = f_Z(z,t)\lambda(t;\,\eta^{-1}(X_{t^-})) = \lambda_m(z,t)$$

so that the jump component (14) in (12) is distributed as the jump component in (5).

□

Let θ be a realisation of the marked Poisson process m^* of intensity (13) driving the jump component in (12):

$$\theta := \{\tau, z, u\} = \{(\tau_1, z_1, u_1), \ldots, (\tau_m, z_m, u_m)\}, \quad m \in \mathbf{N}^+ \tag{15}$$

According to (12)-(13), the jumps' times $\tau = (\tau_1, \tau_2, \ldots, \tau_m)$, are now selected by a homogeneous Poisson process of intensity λ; each jump arrives with a mark (z_i, u_i) such that $Z_i \sim f_Z(z;\tau_i)$ and $U_i \sim \text{Unif}[0,1]$. Notice that, crucially, the jumps' times τ_i are independent from the state of the process and can be simulated in advance along with the marks (z_i, u_i) affecting the jumps' size. We now turn the attention to the process' behaviour in between any two jump times (diffusion component) and we set up a suitable framework for undertaking problem P.2.

4.1.2. *The diffusion component (Problem P.2)..* Let us denote $\mathbb{Q}\left(a;s_1,s_2\right)$ the law of the continuous process $\omega(a;s_1,s_2)$ starting in a at time s_1, finishing at time s_2 driven by the diffusion component of (12):

$$d\omega_t = \alpha\left(\omega_t\right)dt + d\,W_t \tag{16}$$

We assume that (16) satisfies the following conditions.

C.1 Given $T \in \mathbf{R}^+$, for any $a \in \mathbf{R}$, the diffusion measure $\mathbb{Q}(a;0,T)$ is absolutely continuous with respect to the Wiener measure on $[0,T]$ started in a, say $\mathbb{W}(a;0,T)$, and Girsanov representation of the Radon-Nikodym derivative holds:

$$\frac{d\mathbb{Q}(a;0,T)}{d\mathbb{W}(a;0,T)}\left(\omega\right) = \exp\left\{\int_0^T \alpha(\omega_s)d\omega_s - \int_0^T \alpha^2(\omega_s)ds\right\}$$

C.2 The drift function α is continuously differentiable.
C.3 The function $(\alpha^2 + \alpha')/2$ is bounded.

Now we observe that the dynamics (12) of the process X implies:

$$\{X_t : 0 \leq t < \tau_1\} \mid \theta, x_0 \sim \mathbb{Q}(x_0; 0, \tau_1) \tag{17}$$

and, for any $i = 1, \ldots, m$:

$$\{X_t : \tau_i \leq t < \tau_{i+1}\} \mid \theta, x_{\tau_{i^-}} \sim \mathbb{Q}(x_{\tau_i}; \tau_i, \tau_{i+1}), \quad \tau_{m+1} := T \tag{18}$$

where $x_{\tau_i} = x_{\tau_i^-} + \Delta_\eta\left(z_i, x_{\tau_i^-}\right) I\left(u_i, \tau_{i^-}, x_{\tau_i^-}\right)$. From (17) and (18) problem P.2(a) boils down to the simulation of an exact skeleton, say $\mathcal{S}(a; s_1, s_2)$, from $\mathbb{Q}(a; s_1, s_2)$ for given $a \in \mathbf{R}$ and $0 \leq s_1 \leq s_2 \leq T$. This is a challenging task because in general (16) does not have explicit solution with identifiable transition densities. Under conditions C.1-C.3 the Exact Algorihtm 1 ((4), (1)) provides a suitable solution to this problem. Notice that for ease of exposition, we have confined our analysis to the basic Exact Algorithm 1 characterized by the most restrictive set of conditions C.1-C.3. However we can easily relax condition C.3 by applying the more general Exact Algorithm 2 and Exact Algorithm 3 (see (2)).

Lemma 2
Under conditions C.1-C.3 we can apply the Exact Algorithm 1 to simulate an exact skeleton $\mathcal{S}_{EA}(a; s_1, s_2)$ *from* $\mathbb{Q}(a; s_1, s_2)$ *for given* $a \in \mathbf{R}$ *and* $0 \leq s_1 < s_2 \leq T$ *such that:*

$$\mathcal{S}_{EA}(a; s_1, s_2) := \left\{(s_1, a), \left\{\left(t_j, \omega_{t_j}\right)\right\}_{j=1,2,\ldots,N}, (s_2, \omega_{s_2})\right\} \tag{19}$$

where $s_1 < t_1 < \cdots < t_N < s_2$.

Proof:
See (2).

□

Thus the Exact Algorithm returns as output a Skeleton $\mathcal{S}_{EA}(a; s_1, s_2)$ (2) of the trajectory of $\omega(a; s_1, s_2)$ including: the (given) starting point at time s_1, (s_1, a); the ending point at time s_2, (s_2, ω_{s_2}); a (random) number N of intermediate points at (randomly located) time instants $s_1 < t_1 < \cdots < t_N < s_2$, $\left\{\left(t_j, \omega_{t_j}\right)\right\}_{j=1,2,\ldots,N}$ (problem P.2 (a)). Furthermore conditionally on the EA1 output the process' trajectory is distributed as a product of independent Brownian bridges (problem P.2 (b)). Lemma (3) formalises this crucial property.

Lemma 3
Let $\mathcal{S}_{EA}(a; s_1, s_2)$ (19) be the Skeleton of $\omega(a; s_1, s_2)$ generated by the Exact Algorithm. Then:

$$\omega(a; s_1, s_2) \mid \mathcal{S}_{EA}(a; s_1, s_2) \sim \bigotimes_{j=0}^{N} \mathbb{BB}\left(t_i, \omega_{t_i}; t_{i+1}, \omega_{t_{i+1}}\right) \tag{20}$$

where $t_0 \equiv s_1$ so that $\omega_{t_0} \equiv a$ and $t_{N+1} \equiv s_2$.

Proof:
See (2).

□

4.2. *The JEA algorithm: Thinning and Exact Algorithm.* The JEA simulation procedure follows easily from the results in Lemma 1 and Lemma 2. Under conditions C.0-C.3, Algorithm (2) returns an exact skeleton of the process X (5)-(2) up to time $T > 0$.
An intuitively appealing way to see the Jump Exact Algorithm is as a combination of the *Exact Algorithm* and the *Thinning Algorithm*. In fact let us focus our attention on the model X under the original representation (5)-(2). Given the assumption C.0, we can apply a thinning algorithm in order to simulate the jumps' times $(t_1, t_2, \ldots, t_M)$ from the appropriate renewal process of intensity $\lambda(t; \eta^{-1}(X_{t^-}))$, $t > 0$. This involves the usual two steps.

Proposal step: we propose a set of candidate jumps' times $\tau = (\tau_1, \tau_2, \ldots, \tau_m)$ from a Poisson process of intensity λ such that for any $t > 0$ and $x \in \mathbf{R}$, $\lambda \geq \lambda(t; x)$.

Acc./Rej. step: for any $i = 1, 2, \ldots, m$, given $X_{\tau_i^-} = x_{\tau_i^-}$, we accept τ_i (as jumps' time) with probability:

$$R\left(\tau_i, x_{\tau_i^-}\right) = Pr\left\{I\left(U_i, \tau_i, x_{\tau_i^-}\right) = 1\right\}; \qquad U_i \sim \text{Unif}[0, 1] \tag{21}$$

Algorithm 2 Jump Exact Algorithm (JEA)

1. `Simulate` θ `from the marked Poisson process` Θ `with intensity` (13):

   $\theta = \{\tau, z, u\} = \{(\tau_1, z_1, u_1), (\tau_2, z_2, u_2), \ldots, (\tau_m, z_m, u_m)\}$

   such that $\tau_1 \leq \tau_2 \leq \cdots \leq \tau_m$.

2. `Set` $\tau_0 = 0$ and $\tau_{m+1} = T$. `For` $i = 0, 1, \ldots, m$ `repeat:`

   2.1. `apply the Exact Algorithm (EA) to` $\omega(x_{\tau_i}; \tau_i, \tau_{i+1})$ `to simulate:`

   $$\mathcal{S}_{EA}\left(x_{\tau_i}; \tau_i, \tau_{i+1}\right) = \left\{\left(\tau_i, x_{\tau_i}\right), \left\{\left(t_{j,i}, x_{t_{j,i}}\right)\right\}_{j=1,2,\ldots,N_i}, \left(\tau_{i+1}^-, x_{\tau_{i+1}^-}\right)\right\}$$

   2.2. `set` $x_{\tau_{i+1}} = x_{\tau_{i+1}^-} + \Delta_\eta\left(z_i, x_{\tau_{i+1}^-}\right) I\left(u_i, \tau_{i+1}^-, x_{\tau_{i+1}^-}\right) \mathbf{I}_{\{i<m\}}$

3. `Output:`

   $$\mathcal{S}_{JEA}(x_0; 0, T) := \bigcup_{i=0}^{m} \mathcal{S}_{EA}\left(x_{\tau_i}; \tau_i, \tau_{i+1}\right)$$

The proposal step is straightforward. The set of resulting candidate jumps' times τ split the interval $[0, T)$ in $m+1$ sub-intervals $\{[\tau_{i-1}, \tau_i)\}_{i=1,2,\ldots,m+1}$ ($\tau_0 \equiv 0$, $\tau_{m+1} \equiv T$) on each of which the process X follows a pure diffusion with SDE (16). Consider the first interval $[0, \tau_1)$. Under assumptions A.1-A.3 we can apply the Exact Algorithm 1 to simulate a Skeleton $\mathcal{S}_{EA}\left(x_0; 0, \tau_1\right)$ (x_0 given) from the diffusion $\omega(x_0; 0, \tau_1)$ such that, from Lemma 2:

$$\mathcal{S}_{EA}\left(x_0; 0, \tau_1\right) = \left\{\left(0, x_0\right), \left\{\left(t_{j,1}, x_{t_{j,1}}\right)\right\}_{j=1,2,\ldots,N_1}, \left(\tau_1^-, x_{\tau_1^-}\right)\right\} \tag{22}$$

with $0 \leq t_{1,1} \leq t_{2,1} \leq \cdots \leq t_{N_1,1} \leq \tau_1$. Thus we can use the information provided by the EA-Skeleton to compute exactly the thinning acceptance probability (21) corresponding to the first candidate jump time τ_1. According to (5):

$$X_{\tau_1} = x_{\tau_1} = \begin{cases} x_{\tau_1^-} + \Delta_\eta(Z_1, x_{\tau_1^-}) & \text{if } I\left(U_1, \tau_1, x_{\tau_1^-}\right) = 1 \\ x_{\tau_1^-} & \text{if } I\left(U_1, \tau_i, x_{\tau_1^-}\right) = 0 \end{cases}$$

where $U_1 \sim Unif[0, 1]$ and $Z_1 \sim f_Z(z; \tau_1)$. Given $X_{\tau_1} = x_{\tau_1}$ we can then repeat the same procedure for the interval $[\tau_1, \tau_2)$ and so forth until the

last time interval $[\tau_m, \tau_{m+1} \equiv T)$. Finally the Jump Exact Algorithm manages to construct a Skeleton $\mathcal{S}_{JEA}$ of the process X by merging the EA-skeletons $\mathcal{S}_{EA}(x_{\tau_i}, \tau_i, \tau_{i+1})$ defined on the time-intervals $[\tau_i, \tau_{i+1})$ selected by the thinning proposal. By construction, the skeleton $\mathcal{S}_{JEA}$ will have istantaneous jumps at those times $(t_1, t_2, \ldots, t_M) \subseteq (\tau_1, \tau_2, \ldots, \tau_m)$ accepted by the thinning accept/reject test. For convenience of notation, we shall represent $\mathcal{S}_{JEA}(x_0; 0, T)$ in the following way:

(23)
$$\mathcal{S}_{JEA} = \bigcup_{i=0}^{m} \mathcal{S}_{EA}\left(x_{\tau_i}; \tau_i, \tau_{i+1}\right) = \bigcup_{i=0}^{m} \left\{ \left(t_{0,i}, x_{t_{0,i}}\right), \left(t_{1,i}, x_{t_{1,i}}\right), \ldots, \left(t_{N_i+1,i}, x_{t_{N_i+1,i}}\right) \right\}$$

where, $\tau_0 = 0$, $\tau_{m+1}^- = T$ and $\tau_i := t_{0,i} \leq t_{1,i} \leq \cdots \leq t_{N_i+1,i} := \tau_{i+1}^-$ for any $i = 0, 1, \ldots, m$. In Figure 1 we have provided a graphical representation of the JEA Skeleton under notation (23).

5. Monte Carlo applications.

5.1. *Preliminaries.* In this Section we give some examples of applications of the Jump Exact Algorithm to concrete Monte Carlo problems (Jump Exact Monte Carlo Algorithms). Thus we assume that the process V (1)-(2) is such that the transformed process $X = \eta(V)$ (5)-(2) satisfies conditions C.0-C.3 and we consider the problem of the Monte Carlo estimation of:

$$\nu := \mathbb{E}\left(\phi(V) \mid V_0 = v_0\right) \tag{24}$$

for some functionals of interest ϕ. The goal in this Section is to show by a number of examples that in the JEA framework an appropriate combination of standard distributional result from Brownian motion theory and suitable Monte Carlo simulation techniques allow unbiased Monte Carlo estimation of (24) even in presence of complex, highly path-dependent functonals ϕ. The functionals we will consider are inspired by financial applications. In fact we can interpret the problem of the Monte Carlo estimation of (24) as a pricing problem where (24) is the price of a derivative defined on the underlying V (1)-(2) with (discounted) payoff ϕ. However we point out that the analogy with option pricing is merely motivated by computational purposes. In particular, we will not question the major problem of the characterization of the model (1) under the pricing (or "martingale") measure and the related implications of the market uncompleteness (due to the presence of jumps).

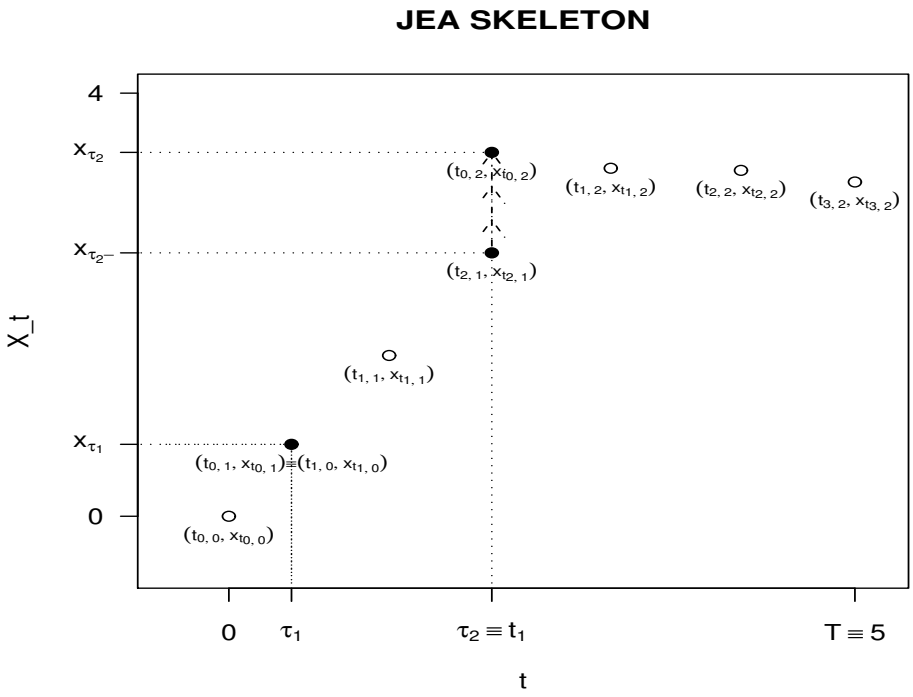

FIG 1. *This is a graphic illustration of the JEA skeleton for the model (38)-(39) with parameters* $x_0 = 0$, $T = 5$, $\alpha = 0.5$, $\beta = 0.5$, $l = 0.3$, $\sigma = 1$, $\lambda_0 = 1$. *The solid circles are the realised trajectory at the candidate jumps' times* τ_1 *and* τ_2 *where the thinning takes place. The first jump in* τ_1 *is rejected and the second in* τ_2 *is accepted. Thus in* $t_1 = \tau_2$ *the first and unique jump occurs. The labels in round brackets attached to each circle are meant to exemplify the notation for the Skeleton introduced in (23).*

We will need the following Brownian bridge-related notation. Given a Brownian bridge starting in a at time s_1 and ending in b at time s_2, say $BB(s_1, y_1; s_2, y_2)$, we denote by $\mathcal{M}(s_1, y_1; s_2, y_2)$ its maximum and by $p(s_1, y_1, s_2, y_2; l_1, l_2)$ its exit probability from the interval (l_1, l_2) conditionally on the event $\{y_1 \in (l_1, l_2)\}$:

$$\begin{aligned} \mathcal{M}(s_1, y_1; s_2, y_2) &:= \max_t \{BB_t(s_1, y_1; s_2, y_2) : s_1 \le t \le s_2\} \\ p(s_1, y_1, s_2, y_2; l_1, l_2) &:= 1 - Pr(BB(s_1, y_1; s_2, y_2) \in (l_1, l_2) \mid \{y_1 \in (l_1, l_2)\}) \end{aligned}$$

Additionally, given the process X (5)-(2) and an open interval (l_1, l_2) such

that $x_0 \in (l_1, l_2)$, we will denote by $\tau^X_{(l_1,l_2)}$ its first exit time from the set (l_1, l_2), i.e.

$$\tau^X_{(l_1,l_2)} := \inf\{t \in [0,T] : X_t \notin (l_1, l_2)\} \tag{25}$$

under the convention that $\inf \emptyset = \infty$.

5.2. *Brownian bridge decomposition.* The Jump Exact Algorithm allows a desirable factorization of the conditional process given $\mathcal{S}_{JEA}$ in terms of independent Brownian bridges. This crucial property is formalised by Proposition 1 and illustrated in Figure 2 where we have completed the Skeleton of Figure 1 by the appropriate Brownian bridge interpolation.

Proposition 1
Let $\mathcal{S}_{JEA}(x_0; 0, T)$ (23) be the Skeleton of the process X generated by the Jump Exact Algorithm. Then:

$$X \mid \mathcal{S}_{JEA}(x_0; 0, T) \sim \bigotimes_{i=0}^{m} \bigotimes_{j=0}^{N_i} \mathbb{BB}\left(t_{j,i}, x_{t_{j,i}}; t_{j,i+1}, x_{t_{j,i+1}}\right) \tag{26}$$

Proof:
From the JEA construction of the Skeleton (23) and the EA Brownian bridge decomposition (20), for any $i = 0, 1, \ldots, m$:

$$\begin{aligned} X^{(i)} \mid \mathcal{S}_{JEA}(x_0; 0, T) &\stackrel{d}{=} \omega\left(x_{t_{0,i}}; t_{0,i}, t_{N_{i+1},i}\right) \mid \mathcal{S}_{EA}\left(x_{t_{0,i}}; t_{0,i}, t_{N_{i+1},i}\right) \\ &\sim \bigotimes_{j=0}^{N_i} \mathbb{BB}\left(t_{j,i}, x_{t_{j,i}}; t_{j+1,i}, x_{t_{j+1,i}}\right) \end{aligned}$$

where we have used the following notation $X^{(i)} := \{X_t : t_{0,i} \le t < t_{N_{i+1},i}\}$. Representation (26) then follows from the Markov property of the proocess X.

□

Proposition 1 implies that, by conditioning on $\mathcal{S}_{JEA}$, we reduce the problem of the simulation from the highly complex jump-diffusion process (1) to the problem of the simulation from (well known) Brownian bridge measures.

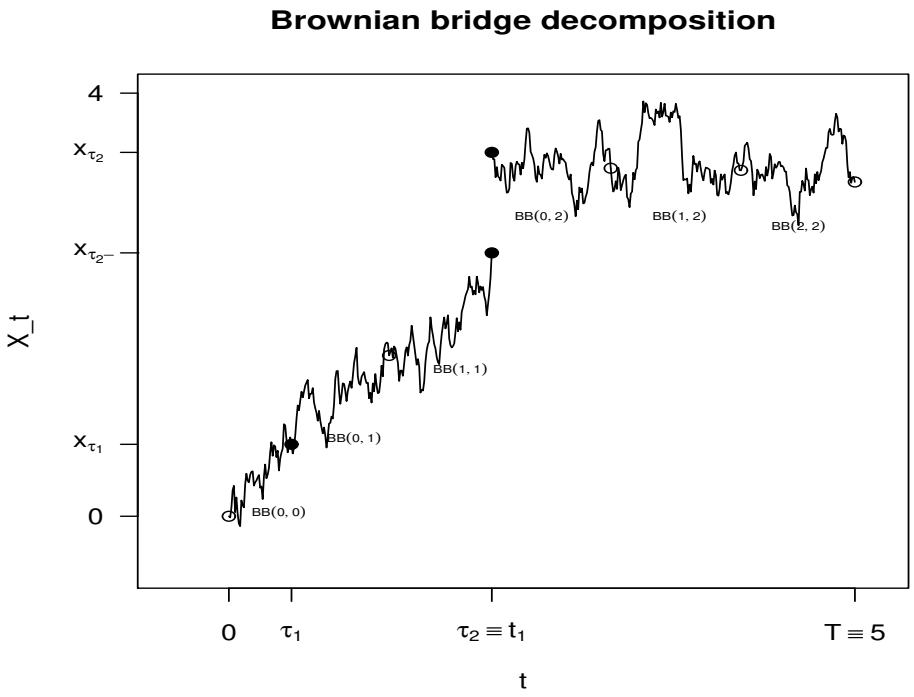

FIG 2. *This is the complete realisation of the trajectory corresponding to the Skeleton in Figure 1. The continuous component of the trajectory is obtained by simulating the appropriate bridge (according to Lemma 3) on a very fine discrete grid ($\delta = 0.01$). We have introduced the convenient notation $BB(j,i) \equiv BB\left(t_{j,i}, x_{t_{j,i}}; t_{j+1,i}, x_{t_{j+1,i}}\right)$ for any j, i.*

This is very valuable in Monte Carlo applications as many interesting functionals associated with Brownian bridge can be simulated exactly. In the rest of the current Section we will provide a sample of a number of challenging Monte Carlo problems that can be solved thanks to the Brownian bridge characterization (26).

5.3. *Functional evaluated at a discrete set of time points.* We consider the following class of functionals:

$$\phi(V) := \phi\left(V_{T_1}, V_{T_2}, \ldots, V_{T_n}\right) = \phi^*\left(X_{T_1}, X_{T_2}, \ldots, X_{T_n}\right) \tag{27}$$

for given $0 \leq T_1 \leq T_2 \leq \cdots \leq T_n$ and $n \in \{1, 2, \ldots\}$ where $\phi^*(u_1, u_2, \ldots, u_n) =$

$\phi\left(\eta^{-1}(u_1), \eta^{-1}(u_2), \ldots, \eta^{-1}(u_n)\right)$, $\eta^{-}(\cdot)$ being the inverse of $\eta(\cdot)$ (4). Popular examples of these functionals in finance are the payoffs of *digital options*, *european options* and *discrete (geometric and arithmetic) asian options*. The Monte Carlo problem involves the joint simulation of $X_{T_1}, X_{T_2}, \ldots, X_{T_n}$. To this end the JEA framework offers two main options.

1. We can split $[0, T_n]$ in n sub-intervals $[T_i, T_{i+1})$, $i = 0, 1, \ldots, n-1$, apply the Exact Algorithm to each of the sub-intervals and then simply merge the individual sub-skeletons so that the resulting skeleton $\mathcal{S}_{JEA}(x_0; 0, T_n)$ includes the target vector $X_{T_1}, X_{T_2}, \ldots, X_{T_n}$.
2. We can choose an arbitrary splitting scheme to generate the Skeleton on $[0, T_n]$ and simulate the variables of interest $X_{T_1}, X_{T_2}, \ldots, X_{T_n}$ from the appropriate Brownian bridge according to (26). There are several ways to simulate a discrete realisation from a Brownian bridge trajectory; a very convenient one resorts to trivial Brownian motion simulation through representation (43).

It is worth to point out here that the choice of the "splitting" can have a relevant effect on the efficiency of the Exact Algorithm (and thus also of the Jump Exact Algorithm). We refer to (22) for a complete treatment of this point.

5.4. *Path dependent functionals.* The Brownian bridge decomposition (26) allows us to construct Monte Carlo estimators of (24) also for a number of interesting path-dependent functionals. We provide some examples suggested by applications in option pricing.

5.4.1. *Lookback options.* Lookback options are a type of path-dependent options where the payoff depends on the maximum (or minimum) asset price over the life of the option $[0, T]$ and eventually (floating strike type) by the value of the underlying at some maturity time T. Since the function η (4) is strictly increasing we can write the lookback options' payoff as:

$$\phi(V) = \phi\left(\max_{0\le t\le T} V_t, V_T\right) = \phi^*\left(\max_{0\le t\le T} X_t, X_T\right) \tag{28}$$

where $\phi^*(v, u) = \phi\left(\eta^{-1}(v), \eta^{-1}(u)\right)$. Hence the crucial step of the Monte Carlo procedure involves the simulation of the maximum of X given the Skeleton. From (26) we have trivially that:

$$\max_{0\le t\le T} X_t \mid \mathcal{S}_{JEA}(x_0; 0, T) \stackrel{d}{=} \max_{i,j}\left\{\mathcal{M}\left(t_{j,i}, x_{t_{j,i}}; t_{j,i+1}, x_{t_{j,i+1}}\right)\right\}$$

As the distribution function of the maximum of a given Brownian bridge can be derived from simple properties of Brownian motion , we can easily simulate from it by applying the inverse transform method. See Appendix A ((48) and (49)).

5.4.2. *Single Barrier options.* Barrier options are an extremely popular class of path-dependent options which come in many flavours and forms. Their key characteristic is that they are either initiated ("knocked-in") or exterminated ("knocked-out") if the underlying process V exits a given set $H = (a, b)$ such that $V_0 = v_0 \in H$. Firstly we discuss *single barrier* options ($a = -\infty$ or $b = +\infty$). To fix the ideas let us consider a *knock-out* option on the underlying V (1) with upper barrier $b > v_0$ (i.e. $H = (-\infty, b)$). If the underlying does not cross the upper barrier b between 0 and T the option pays off a function g of the asset process V at maturity time T. If the process crosses the barrier the option is killed and a rebate R is paid to its holder at the killing time. In this context, assuming a fixed continuously compounded interest rate r, the payoff of the option under the transformed process X is:

$$\phi^*\left(X_T,\ \tau^X_{H^*}\right) = e^{-rT} g^*(X_T)(1 - \mathbf{I}_{\{\tau^X_{H^*} \le T\}}) + e^{-r\tau^X_{H^*}}\, R\, \mathbf{I}_{\{\tau^X_{H^*} \le T\}} \tag{29}$$

where $g^*(\cdot) := g(\eta^{-1}(\cdot))$ and $H^* := (-\infty, b^*)$ with $b^* = \eta^{-1}(b)$. Thus conditionally on the Skeleton (23) exact simulation of ϕ^* involves the two steps: 1) the simulation of the one-sided crossing event of a Brownian bridge $\mathbf{I}_{\{\dots\}}$ and 2) the simulation of the crossing time τ^X given the crossing event. Both steps can be performed efficiently applying the standard results from Brownian motion theory collected in Appendix A ((47) and (48)). Interestingly if we assume that the rebate is paid at the (fixed) maturity time T, i.e. $\tau^X_{H^*} \equiv T$ in (29), we can perform more efficient Rao-Blackwellised Monte Carlo estimation of η (24) by averaging over the conditional expectations:

$$\mathbb{E}\left(\phi^*\ (30)_{JEA}(x_0, 0, T)\right) = e^{-r\,T}\left[g^*(x_T) + (R - g^*(x_T))\, Pr\left(\tau^X_{H^*} \le T \mid \mathcal{S}_{JEA}\right)\right]$$

where by Proposition 1 explicit expression for the conditional crossing probability can be derived from the one-sided crossing probability of the Brownian bridge (47).

5.4.3. *Double barrier options.* We consider the payoff function ϕ^* (29) with $\tau^X_{H^*} \equiv T$ under the assumption that $H^* = (a^*, b^*)$ for $-\infty < a^* < x_0 < b^* < +\infty$. In the JEA framework, the simulation problem involves the two-sided crossing probability of the Brownian bridge. It is known that closed-form expressions for such probability are not available. This makes the simulation task much harder than in the single barrier case. In particular: 1) Rao-Blakwelised estimation (30) is not feasible; 2) trivial simulation strategies based on the knowledge of the crossing event's probability cannot be performed. An *ad hoc* algorithm to deal with this case has been proposed by (8). We recall it briefly. For given $i = 0, 1, \ldots, m$ and $j = 0, 1, \ldots, N_i$ we are looking for a simulation procedure that allows us to simulate events of (unknown) probability $p\left(t_{j,i}, x_{t_{j,i}}, t_{j+1,i}, x_{t_{j+1,i}}; a^*, b^*\right)$. To this end, we identify two sequences of (known) real numbers:

(31)
$$\left\{\underline{n}_k\left(t_{j,i}, x_{t_{j,i}}, t_{j+1,i}, x_{t_{j+1,i}}; a^*, b^*\right); \overline{n}_k\left(t_{j,i}, x_{t_{j,i}}, t_{j+1,i}, x_{t_{j+1,i}}; a^*, b^*\right)\right\}_{k=1,2,\ldots}$$

such that, as $k \to +\infty$:

$$\underline{n}_k\left(t_{j,i}, x_{t_{j,i}}, t_{j+1,i}, x_{t_{j+1,i}}; a^*, b^*\right) \uparrow p\left(t_{j,i}, x_{t_{j,i}}, t_{j+1,i}, x_{t_{j+1,i}}; a^*, b^*\right) \quad (32)$$

$$\overline{n}_k\left(t_{j,i}, x_{t_{j,i}}, t_{j+1,i}, x_{t_{j+1,i}}; a^*, b^*\right) \downarrow p\left(t_{j,i}, x_{t_{j,i}}, t_{j+1,i}, x_{t_{j+1,i}}; a^*, b^*\right) \quad (33)$$

The explicit functional form of the sequences (31) can be found in Appendix A ((53) and (54)) supported by a sketch of the proof of the limits (32) and (33). Limits (32) and (33) suggest that we can simply simulate a $[0,1]-$uniformly distributed random variable $U = u$ and compare it with (increasing indeces) couples of values $(\underline{n}_k, \overline{n}_k)$ until either $u \leq \underline{n}_k$ (crossing event accepted) or $u > \overline{n}_k$ (crossing event rejected). This procedure turns out to be efficient as the sequences $\{\overline{n}_k\}$ and $\{\underline{n}_k\}$ converge to p faster than exponentially (see also Proposition 2 in (8)).

5.4.4. *Bond pricing with stochastic interest rate.* Finally we outline a general simulation algorithm for unbiased Monte Carlo estimation of:

$$\nu = \mathbb{E}\left[e^{-\int_0^T V_s\, ds} \mid V_0 = v_0\right] = \mathbb{E}\left[e^{-\int_0^T \eta^{-1}(X_s)\, ds} \mid X_0 = \eta(v_0)\right] \quad (34)$$

for jump-diffusion processes V (1)-(2) and (its transformation) X (5)-(2)

under the usual conditions C.0-C.3. This is a very relevant computational problem in financial applications. For instance we can think of ν as the price of a bond with unit face value and maturity time T under a jump-diffusion stochastic interest rate pricing framework (see e.g. (9) or (17)) or alternatively as the (complementary) default probability in a reduced form approach to credit risk modeling (see e.g. (12)) where the default arrival process is a Cox process driven by an exogeneous jump-diffusion. The computational task is very demanding, involving the (exponential of the) integral of a jump-diffusion trajectory. However, conditioning on the JEA Skeleton (23) and applying the Brownian bridge decomposition yields:

$$\nu = \mathbb{E}_{\mathcal{S}}\left[\prod_{i=0}^{n}\prod_{j=0}^{N_i}\mathbb{E}_{\mathbb{BB}(i,j)}\left(\exp\left\{-\int_{t_{j,i}}^{t_{j,i+1}}\eta^{-1}\left(\omega_u\right)\,du\right\}\right)\right] := \mathbb{E}\left[\prod_{i=0}^{n}\prod_{j=0}^{N_i}\mathcal{E}_{i,j}(\mathcal{S}_{JEA})\right] \tag{35}$$

There is no immediate way to simulate an unbiased estimator of (35). In fact we are not able to express the conditional expectation $\mathcal{E}_{i,j}$ as an explicit function of $\mathcal{S}_{JEA}$, neither we can simulate the target functionals conditionally on the Skeleton. However we can still take advantage of the specific form of the target functional and apply the Poisson estimator; i.e. for any $c_{i,j} \in \mathbf{R}$ and $\lambda_{i,j} > 0$, $i = 0, 1, \ldots, m$, $j = 0, 1, \ldots, N_i$:

$$\mathcal{E}_{i,j}(\mathcal{S}_{JEA}) = \mathbb{E}\left(c^*_{i,j}\,\lambda_{i,j}^{-\kappa_{i,j}}\prod_{h=1}^{\kappa_{i,j}}\left\{c_{i,j} - \eta^{-1}(BB_{\pi_{i,j;h}}(i,j)\right\}\right) \tag{36}$$

where $c^*_{i,j} := \exp\left\{\left(\lambda_{i,j} - c_{i,j}\right)\left(t_{j,i+1} - t_{j,i}\right)\right\}$ constant and $\left\{\pi_{i,\,j;1}, \pi_{i,\,j\,;2}, \ldots, \pi_{i,j\,;\,\kappa_{i,j}}\right\}$ is a Poisson process of intensity $\lambda_{i,j}$ on the time interval $t_{j,i+1} - t_{j,i}$. Finally combining (35) and (36) gives an unbiased estimator of ν:

$$\nu^* = \prod_{i=0}^{n}\prod_{j=0}^{N_i}\left\{c^*_{i,j}\,\lambda_{i,j}^{-\kappa_{i,j}}\prod_{h=1}^{\kappa_{i,j}}\left\{c_{i,j} - \eta^{-1}(BB_{\pi_{i,j;h}}(i,j)\right\}\right\} \tag{37}$$

which can be conveniently simulated in a Monte Carlo framework.

6. Numerical example.

6.1. *The model.* We test the JEA algorithm on a jump-diffusion model $X := \{X_t : 0 \leq t \leq T\}$ (5) defined by the SDE:

$$dX_t = \sin(X_{t^-})\, dt + dW_t + \int_{\mathbf{R}} (\sigma z + l\, X_{t^-})\, m(dz, dt) \tag{38}$$

with intensity function λ_m (2):

$$\lambda_m(z\,,\, t;\, X_{t^-};\, \alpha,\, \beta,\, \lambda_0) = \lambda_0\, F_{\mathcal{N}_{0,1}}(\alpha + \beta\, X_{t^-})\; f_{\mathcal{N}_{0,1}}(z) \tag{39}$$

for $\alpha, \beta, l \in \mathbf{R}$ and $\lambda_0 > 0$ where $F_{\mathcal{N}_{0,1}}$ and $f_{\mathcal{N}_{0,1}}$ are (resp.) the distribution function and the density function of a standard normal random variable. The continuous component is a diffusion model without explicit solution characterized by `SINUS` drift and unit diffusion coefficient (`SINUS` model). The jump component is state-dependent both in the intensity of the jumps' arrival process $\lambda(t; X_{t^-};\, \alpha,\, \beta,\, \lambda_0) := \lambda_0\, F_{\mathcal{N}_{0,1}}(\alpha + \beta\, X_{t^-})$ and in the amplitude of the jumps $g(Z, X_{t^-}; l, \sigma) := \sigma\, Z + l\, X_{t^-}$ with $Z \sim \mathcal{N}_{0,1}$. The exogenous parameters α, β, l, λ_0, σ provide us with a very general and flexible modeling framework. In particular for $\lambda_0 = 0$ or $\sigma, l = 0$, model (38) and (39) reduces to the `SINUS` diffusion; for $\beta = 0$ and $l = 0$ the jump component is independent from the state of the process as for Merton model (9)-(8).

6.2. *Simulation schemes.*

6.2.1. *The JEA scheme.* For any given set of the exogenous parameters $(\alpha, \beta, l, \lambda_0)$ the process defined by (38) and (39) satisfies conditions C.0-C.3. In particular the `SINUS` model satisfies conditions C.1-C.3 and it allows the application of EA1. The `SINUS` model turns to be a convenient and easily implementable choice for numerical testing of EA1-related algorithms (see also (4), (3), (8)). The state dependent intensity function driving the jumps' arrival process satisfies condition C.0 as $\lambda(t; X_{t^-};\, \alpha,\, \beta,\, \lambda_0) \leq \lambda_0$. In this context the application of the JEA algorithm (Algorithm 2) entails the following three main steps.

1. We select the vector of time points $\tau := (\tau_0, \tau_1, \ldots, \tau_{m+1})$ where $\tau_0 \equiv 0$, $\tau_{m+1} \equiv T$ and $(\tau_1, \tau_2, \ldots, \tau_m)$ is the vector of the candidate jumps' times simulated from the proposal Poisson process of intensity λ_0.
2. For any $i = 0, 1, \ldots, m$, we apply EA1 on the `SINUS` model between any two time points τ_i and τ_{i+1}.

3. We extract from the EA1 Skeleton the (left-limit) realisation of the process' trajectory at the candidate jumps' time $(\tau_1, \tau_2, \ldots, \tau_m)$ and we evaluate the acceptance thinning probability $F_{\mathcal{N}_{0,1}}(\alpha + \beta X_{\tau_i^-})$ and (in case the thinning test is positive) the amplitude of the jumps $\sigma\, Z + l\, X_{\tau_i^-}$.

We test numerically the Monte Carlo performances of the JEA algorithm against two alternative Euler schemes with jumps.

6.2.2. *The naive Euler scheme with jumps.* This scheme is the natural analog of the naive Euler approximation in a jump diffusion context. The time setting is fully discrete: both the diffusion component and the jump component are simulated at a fixed discrete grid of time points $\{i\Delta\}_{i=1,2,\ldots,n}$ where $\Delta = T/n$ is the discretisation interval. The resulting approximation to (38)-(39) has dynamics:

(40)
$$\tilde{X}_{(i+1)\,\Delta} = \tilde{X}_{i\,\Delta} + \sin(\tilde{X}_{i\,\Delta})\Delta + \sqrt{\Delta}\,\xi_{i+1} + J_{i+1}\left(\sigma\, Z_{i+1} + l\,\tilde{X}_{i\,\Delta}\right), \quad i = 0, \ldots, n-1$$

where $\xi_{i+1}, Z_{i+1} \sim \mathcal{N}(0,1)$, $J_{i+1} \sim Bernoulli(\Delta\lambda_0\, F_{\mathcal{N}_{0,1}}(\alpha + \beta\,\tilde{X}_{i\,\Delta}))$ under the additional condition $\Delta\lambda_0 \leq 1$. Under standard regularity conditions, as $\Delta \to 0$, the scheme converges weakly to (38)-(39) (see (23)). Although rather crude, approximation (40) has been employed in various applied contexts because of its simplicity. Among the others, (25) applied this type of discretisation to simulate the Merton model in a structural credit risk context. More interestingly, (18) proposed a state dependent jump scheme similar to (40) to model US equity indices' dynamics in a discrete time framework.

6.2.3. *The Euler scheme with thinned jumps.* We can construct a more sophisticated approximation of (38)-(39) by combining the Euler scheme with the thinning idea. Loosely, after selecting τ as in JEA (Step 1) we apply the Euler scheme instead of the EA1 algorithm to simulate the process' trajectory between any two times τ_i and τ_{i+1}. As a consequence the (state dependent) thinning acceptance probability and the amplitude of the jumps will be also biased. Nevertheless this scheme seems more sensible than the fully discrete Euler scheme since it does not force the jumps to take place only at the deterministic time points of the discretisation grid. Formally, if we denote by $T_0, T_1, \ldots, T_K$ the set of ordered time points resulting from the superposition of the simulated candidate jumps' times $\tau_1, \tau_2, \ldots, \tau_m$ and the

deterministic discretisation grid $t_0, t_1, \ldots, t_n$ (i.e. $t_i := i\Delta$, $i = 0, 1, \ldots, n$), we can represent the dynamics of the scheme in the following compact way:

$$\begin{aligned} \hat{X}_{T_{k+1}^-} \quad (41)\ \hat{X}_{T_k} &+ \sin\left(\hat{X}_{T_k}\right)(T_{k+1} - T_k) + \sqrt{T_{k+1} - T_k}\,\xi_{k+1} \\ \hat{X}_{T_{k+1}} \quad = \quad \hat{X}_{T_{k+1}^-} &+ H^*(Z_{k+1}, U_{k+1}, \hat{X}_{T_{k+1}^-})\,\mathbf{I}_{\{T_{k+1} \in \{\tau_1, \tau_2, \ldots, \tau_m\}\}}, \quad k = 0, \ldots, K-1 \end{aligned}$$

such that $H^*(z, u, x) := (\sigma\, z + l\, x)\, \mathbb{I}_{\left\{u \le F_{\mathcal{N}_{0,1}}(\alpha + \beta\, x)\right\}}$ with $\xi_{k+1}, Z_{k+1} \sim \mathcal{N}_{0,1}$ and $U_{k+1} \sim Unif\,[0, 1]$. The Monte Carlo application of this scheme as opposed to the crude (40) has been proposed by (14) for the simplest case when the target functional ϕ depends on the trajectory of the process only at a single fixed time point. However for Monte Carlo simulation of path-dependent functionals we will rather employ a more appropriate continuous version of (41) based on a Brownian bridge interpolation of the Euler trajectory (see (15)). This can be easily obtained from (41) by setting:

$$\hat{X}_t = \hat{X}_{T_k} + \sin(\hat{X}_{t_k})\,(t - T_k) + (W_t - W_{T_k}), \quad t \in [T_k, T_{k+1}) \tag{42}$$

where $\{W_t\}$ is the usual standard Brownian motion. We will refer to this scheme as continuous Euler with thinning.

6.3. *Results.* We have compared the Monte Carlo estimator generated by the Jump Exact Algorithm (say $E1$) with the estimators generated by the two Euler schemes (40) and (41) (say respectively $E2$ and $E3$) for the discrete average (or asian option) functional and the maximum (or lookback option) functional. For the maximum, $E3$ employes the continuous version (42) of the Euler scheme. To address our comparison, we require that the Monte Carlo 95%-confidence interval generated by $E2$ and $E3$ has non empty intersection with the corresponding interval generated by $E1$. This seems a very reasonable (although somewhat arbitrary) accuracy. Figures 3-4 and Table 1 summarize the relevant results from a Monte Carlo numerical test for a selection of values of the two parameters l and β affecting the jump component. In order to ensure negligible Monte Carlo error we have chosen a Monte Carlo sample size of 5 10^5 iterations. In Figures 3-4 we have plotted the Monte Carlo estimates of $E1$ and $E3$ Vs the number of discretisation steps (per time unit) $\{1, 2, 4, \ldots, 2^6\}$. To preserve graphics' readibility we omitted to display the behavior of $E2$; however it turns out to be analogous to $E3$ but characterised by slower convergence to $E1$. In Table 1 we have reported the numeric values of the punctual estimate, the 95%-confidence

interval and the computational time for $E1$, $E2$ and $E3$.

The results of the numeric experiment supports the following important arguments in favour of the JEA estimator.

i $E1$ is not affected by discretization bias; in fact it turns out to be the limit of discretisation schemes as the length of the discretization interval tends to 0.

ii $E1$ is not prone to the trade-off between accuracy of the estimator and computational effort. As a consequence, for sufficiently high level of accuracy it is computationally more efficient than estimators based on discretisation schemes.

iii $E1$ provides a reliable benchmark to evaluate the convergence of competing discretisation methods and the accuracy of the related Monte Carlo estimators.

7. Conclusion. We have developed a simple algorithm (JEA) for Monte Carlo simulation of a wide class of functionals of jump-diffusion processes with (markov) state-dependent intensity. Its core consists of the Exact Algorithm for the simulation of the continuous component and on a stochastic thinning algorithm for the simulation of the jump component. In comparison with currently available Monte Carlo estimators, JEA estimator is unbiased, and, for sufficiently high level of accuracy, computationally more efficient.

It is worth to mention at least two research areas related to JEA that deserve further investigation.

1. **Extension of C0-C3.** The domain of JEA can be extended far bejond conditions $C0 - C3$. Trivially, we can weaken conditions $C1 - C3$ by replacing $EA1$ with the more recent $EA2$ or $EA3$ for the simulation of the diffusion component. More interestingly, we believe we can remove also condition $C0$ thus allowing unboundedness of the intensity of the jump component; we are currently working on this.
2. **Financial applications.** In finance the application of jump-diffusion with state-dependant intensity has been limited so far due to the the lack of adequate computational techniques. However the common Merton assumption that jumps in assets' values are fully exogeneous (i.e. independent on the current value of the assets) is often far too restrictive. Now JEA provides a computational method to deal efficiently with state-dependancy, thus motivating a renovated interest for jump diffusion processes with state-dependent intensity in financial modeling. We believe this is a very promising area of applied research. For

a first example of application of the JEA framework to credit risk modeling see (7), Chapter 4.

APPENDIX A: APPENDIX 1: BROWNIAN BRIDGE RESULTS.

We briefly review the Brownian bridge results supporting the Monte Carlo algorithms presented in Section 5. We consider a generic $(s_1, a) \to (s_2, b)$ Brownian bridge and an interval $(l_1, l_2) \subset \mathbf{R}$ such that $a \in (l_1, l_2)$. We adopt the notation:

$$\tau(s_1, a; s_2, b; l_1, l_2) := \inf \{u \in [s_1, s_2] : BB_u(s_1, a; s_2, b) \notin (l_1, l_2)\}$$

where $BB(s_1, a; s_2, b) := \{BB_u(s_1, a; s_2, b) : s_1 \leq u \leq s_2\}$ and $\inf \emptyset = \infty$. Throughout we will rely on the following representation of the Brownian bridge in terms of Brownian motion:

$$BB_u^{(s_1,a;s_2,b)} \stackrel{d}{=} a + \frac{u - s_1}{s_2 - s_1}(b - a) + \frac{s_2 - u}{\sqrt{s_2 - s_1}} W_{\frac{u-s_1}{s_2-u}}; \qquad s_1 \leq u \leq s_2 \tag{43}$$

Given $z \in [s_1, s_2]$ it follows from (43):

$$Pr\left(\tau(s_1, a; s_2, b; l_1, l_2) > z\right) = Pr\left(W_u \in (\alpha_1 + \beta_1 u, \alpha_2 + \beta_2 u), u \in \left(0, \frac{z - s_1}{s_2 - z}\right)\right) \tag{44}$$

with $\alpha_1 := \frac{l_1 - a}{\sqrt{s_2 - s_1}}$, $\alpha_2 := \frac{l_2 - a}{\sqrt{s_2 - s_1}}$, $\beta_1 := \frac{l_1 - b}{\sqrt{s_2 - s_1}}$ and $\beta_2 := \frac{l_2 - b}{\sqrt{s_2 - s_1}}$. Let us consider first the one sided case with $l_2 > a$ and $l_1 = -\infty$. The theory for the one sided case is rather standard; references are the classic (24) and (19). Firstly we rewrite (44) in the following way:

$$Pr\left(\tau(s_1, a; s_2, b; -\infty, l_2) > z\right) = Pr\left(\tau_{\alpha_2,\beta_2} > \frac{z - s_1}{s_2 - z}\right) = Pr\left(\frac{s_1 + \tau_{\alpha_2,\beta_2} s_2}{1 + \tau_{\alpha_1,\beta_1}} > z\right). \tag{45}$$

where $\tau_{\alpha_2,\beta_2} := \inf \{u \geq 0 : W_u \geq \alpha_2 + \beta_2 u\}$. It is time to recall the celebrated Bachelier-Levy formula which gives the density of the hitting time τ_{α_2,β_2} of a sloping line $\alpha_2 + \beta_2 u$ for Brownian motion:

$$f_{\alpha_2,\beta_2}(u) = \begin{cases} e^{-2\alpha_2\beta_2} IG\left(u; \frac{\alpha_2}{\beta_2}, \alpha_2^2\right) & \text{if} \quad \alpha_2\beta_2 > 0 \\ IG\left(u; -\frac{\alpha_2}{\beta_2}, \alpha_2^2\right) & \text{if} \quad \alpha_2\beta_2 \leq 0 \end{cases} \tag{46}$$

where $IG(u;k,g)$ is an inverse gaussian density with parameters k and g. Plugging in $z = s_2$ in (44) and using (46) we find the one sided crossing probability of the Brownian bridge:

$$(47)\quad p(s_1,a;s_2,b;-\infty,l_2) = \begin{cases} 1 & \text{if} \quad b \geq l_2 \\ \exp\left\{-\frac{2}{s_2-s_1}(l_2-a)(l_2-b)\right\} & \text{if} \quad b > l_2 \end{cases}$$

Let $E := \{\tau(s_1,a;s_2,b;-\infty,l_2) \in [s_1,s_2]\}$ be the crossing event with probability (47). From (45) and (46), we derive a suitable reprsentation to simulate the hitting times conditionally on the crossing event:

$$\tau(s_1,a;s_2,b;-\infty,l_2) \mid E \stackrel{d}{=} \frac{s_1 + Z\, s_2}{1+Z} : \quad Z \sim \begin{cases} IG\left(-\frac{l_2-a}{l_2-b}, \frac{(l_2-a)^2}{s_2-s_1}\right) & \text{if} \quad b \geq l_2 \\ IG\left(\frac{l_2-a}{l_2-b}, \frac{(l_2-a)^2}{s_2-s_1}\right) & \text{if} \quad b < l_2 \end{cases}$$

The simulation of an inverse gaussian random variable is standard (e.g. see (10)). Finally from (47) we derive the distribution function of the maximum of the Brownian bridge:

$$(48)$$
$$Pr\left\{\mathcal{M}(s_1,a;s_2,b) \leq m\right\} = 1 - \exp\left\{-\frac{2\left((m-a)(m-b)\right)}{s_2-s_1}\right\}, \quad m \geq \max(a,b)$$

so that an immediate application of the inverse transform method (e.g. see (10)) provides the recipe for the simulation of $\mathcal{M}$:

$$(49)$$
$$\mathcal{M}(s_1,a,s_2,b) \stackrel{d}{=} \frac{1}{2}\left(\sqrt{(a-b)^2 - 2\,(s_2-s_1)\log(U)} + a + b\right) : \quad U \sim \text{Unif}[0,1]$$

We turn the attention now to the double barrier case $|l_1|, |l_2| < \infty$, $a \in (l_1,l_2)$ in order to justify the simulation method in Section 5.4.3. This is a succint excerpt from (8) (Section 2.4.2 and Appendix 1). From (44):

$$(50)$$
$$p(s_1,a;s_2,b;l_1,l_2) = \begin{cases} 1 & \text{if} \quad b \notin (l_1,l_2) \\ Pr\left(W_u \in (\alpha_1 + \beta_1\, u, \alpha_2 + \beta_2\, u),\, u \geq 0\right) & \text{if} \quad b \in (l_1,l_2) \end{cases}$$

where $a, b \in (l_1, l_2)$ implies that $\alpha_1, \beta_1 < 0$ and $\alpha_2, \beta_2 > 0$. We introduce the two sequences of functions:

$$\begin{aligned} P_j(s,x;t,y;l_1,l_2) &:= p_j(s,x;t,y;l_2-l_1,l_1) + p_j(s,x;t,y;l_2-l_1,l_2) \\ Q_j(s,x;t,y;l_1,l_2) &:= q_j(s,x;t,y;l_2-l_1,l_1) + q_j(s,x;t,y;l_2-l_1,l_2); \quad (j=1,2,\dots) \end{aligned}$$

where

(51)
$$p_j(s,x;t,y;\delta,l) = e^{-\frac{2}{t-s}[j\delta+(l-x_1)][j\delta+(l-y)]}; \quad q_j(s,x;t,y;\delta,l) = e^{-\frac{2j}{t-s}\left[j\delta^2-\delta(l-x)\right]}$$

From the representation of the two sided crossing probability (of two sloping lines) for Brownian motion in (11), using (50), for a, $b \in (l_1, l_2)$:

$$(52) \qquad p(s_1,a;s_2,b;l_1,l_2) = \sum_{j=1}^{\infty} \left[P_j(s,x;t,y;l_1,l_2) - Q_j(s,x;t,y;l_1,l_2)\right]$$

From (51) we can easily verify the two following crucial facts:

i. for any $j = 1, 2, \dots$: $P_j > Q_j > P_{j+1}$
ii. for $j \to \infty$: $P_j \downarrow 0$ and $Q_j \downarrow 0$

In this context it is immediate that the two sequences:

$$\begin{aligned} (53)\, \underline{n}_k(s,x;t,y;l_1,l_2) &= \sum_{j=1}^{k} \left[P_j(s,x;t,y;l_1,l_2) - Q_j(s,x;t,y;l_1,l_2)\right] \\ (54)\, \overline{n}_k(s,x;t,y;l_1,l_2) &= \underline{n}_{k-1}(s,x;t,y;l_1,l_2) + P_k(s,x;t,y;l_1,l_2) \end{aligned}$$

converge respectively from below and from above to the target probability p. Thus we can establish the final result supporting the Monte Carlo algorithm in Section 5.4.3: for $a, b \in (l_1, l_2)$, $-\infty < l_1 < l_2 < +\infty$ as $k \to \infty$:

(55)
$$\underline{n}_k(s,x;t,y;l_1,l_2) \uparrow p(s,x;t,y;l_1,l_2); \quad \overline{n}_k(s,x;t,y;l_1,l_2) \downarrow q(s,x;t,y;l_1,l_2)$$

More details on the Doob's approach to the two-sided crossing probability problem (52) and an alternative proof of (55) based on probabilistic arguments can be found in (8).

REFERENCES

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

[2] Beskos, A., Papaspiliopoulos, O., and Roberts, G. O. (2007). A new factorisation of diffusion measure and sample path reconstruction. Submitted.

[3] Beskos, A., Papaspiliopoulos, O., Roberts, G. O., and Fearnhead, P. (2006b). Exact and efficient likelihood based inference for discretely observed diffusions (with discussion). *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 68(3):333–382.

[4] Beskos, A. and Roberts, G. O. (2005). Exact simulation of diffusions. *Ann. Appl. Probab.*, 15(4):2422–2444.

[5] Caines, P. E. and Zhang, J. F. (1995). On adaptive control for jump parameter systems via nonlinear filtering. *SIAM J. Control Optim.*, 33:1758–1777.

[6] Cariboni, J. and Schoutens, W. (2004). Pricing credit default swaps under lévy models. available from www.defaultrisk.com.

[7] Casella, B. (2006). *Exact Monte Carlo simulation of Diffusion and Jump-diffusion processes with financial applications*. PhD thesis, Università Commerciale L. Bocconi, Istituto Metodi Quantitativi.

[8] Casella, B. and Roberts, G. O. (2008). Exact monte carlo simulation of diffusions with barriers. To appear in Advances in Applied Probability.

[9] Das, S. (2002). The surprise element: Jumps in interest rates. *Journal of Econometrics*, 106:27–65.

[10] Devroye, L. (1986). *Nonuniform random variate generation*. Springer-Verlag, New York.

[11] Doob, J. L. (1949). Heuristic approach to the Kolmogorov-Smirnov theorems. *Ann. Math. Statistics*, 20:393–403.

[12] Giesecke, K. (2004). Credit risk modeling and evaluation: an introduction. In *Credit risk: models and management*, volume 2 of *Risk Books*. D. Shimko.

[13] Giraudo, M. and Sacerdote, L. (1997). Jump-diffusion processes as models for neuronal activity. *Biosystems*, 40:75–82.

[14] Glasserman, P. and Merener, N. (2004). Convergence of a discretization scheme for jump-diffusion processes with state-dependent intensities. *Proc. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci.*, 460(2041):111–127. Stochastic analysis with applications to mathematical finance.

[15] Gobet, E. (2000). Weak approximation of killed diffusion using Euler schemes. *Stochastic Process. Appl.*, 87(2):167–197.

[16] Ikeda, N. and Watanabe, S. (1989). *Stochastic differential equations and diffusion processes*, volume 24 of *North-Holland Mathematical Library*. North-Holland Publishing Co., Amsterdam, second edition.

[17] Johannes, M. (2004). The statistical and economic role of jumps in continuous-time interest rate models. *The Journal of Finance*, 59(1):227–260.

[18] Johannes, M., Kumar, R., and Polson, N. (1999). State dependent jump models: How do us equity indices jump? Working paper. University of Chicago.

[19] Karatzas, I. and Shreve, S. E. (1991). *Brownian motion and stochastic calculus*,

volume 113 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, second edition.

[20] Merton, R. C. (1976). Option pricing when undrlying stock returns are discontinuous. *Journal of Financial Economics*, 3:125–144.

[21] Metwally, S. and Atiya, A. (2002). Using brownian bridge for fast simulation of jump-diffusion processes and barrier options. *Journal of Derivatives*, 10:43–54.

[22] Pelucchetti, S. (2008). *An analysis of the efficiency of the Exact Algorithm*. PhD thesis, Università Commerciale L. Bocconi, Istituto Metodi Quantitativi.

[23] Platen, E. and Rebolledo, R. (1985). Weak convergence of semimartingales and discretisation methods. *Stochastic Process. Appl.*, 20(1):41–58.

[24] Revuz, D. and Yor, M. (1991). *Continuous Martingales and Brownian Motion*. Springer-Verlag.

[25] Zhou, G. (2001). The term structure of credit spreads with jump risk. *Journal of Banking and Finance*, 25:504–531.

[26] Zhu, S. C. (1999). Stochastic jump-diffusion process for computing medial axes in markov random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21:1158–1169.

Address of the Second author
Department of Statistics
University of Warwick
Coventry CV4 7AL
UK

E-mail: gareth.o.roberts@warwick.ac.uk
URL: http://www2.warwick.ac.uk/fac/sci/statistics/staff/academic/roberts/

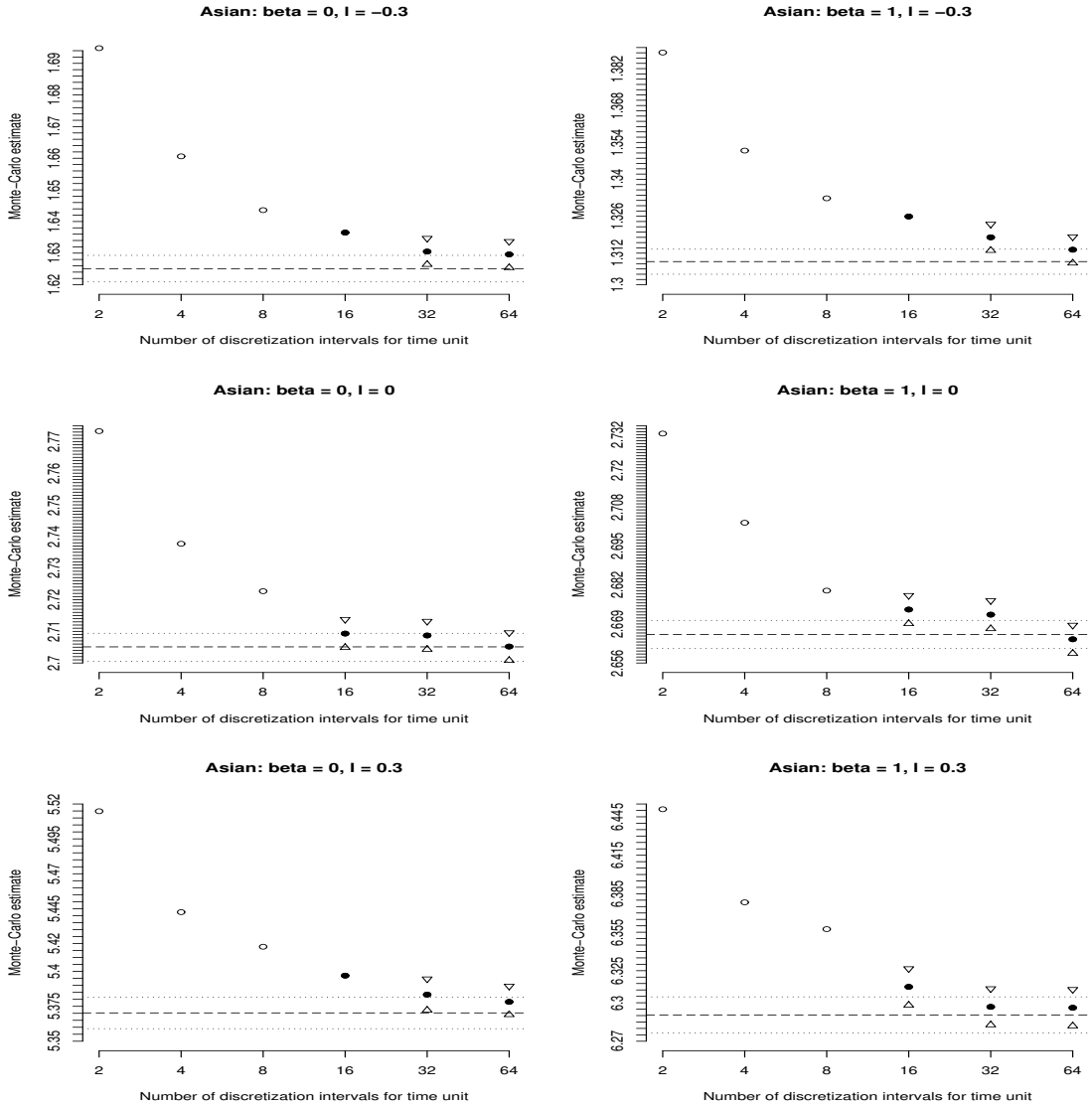

FIG 3. *Plot of the Monte Carlo estimates and* 95%*-confidence intervals of* $E1$ *(resp. dashed line and dotted lines) and* $E3$ *(resp. circle and triangles) Vs the number of discretisation intervals per time unit* (2^n, $n = 0, 1, \ldots, 6$)*. Functional: ASIAN option* $\phi = X_1 + X_2 + X_3 + X_4 + X_5$*. Model:* X *(38) with fixed parameters* $\alpha = \sigma = \lambda_0 = 1$ *and* $x_0 = 2$ *and variable parameters* β *and* l*. Monte Carlo sample size:* 5×10^5*. Notes: We have reported only the confidence interval of* $E3$ *characterised by non-empty intersection with the corresponding confidence interval of* $E1$*. The solid circles represent* $E3$ *estimates whose computational time is lower than* $E1$*.*

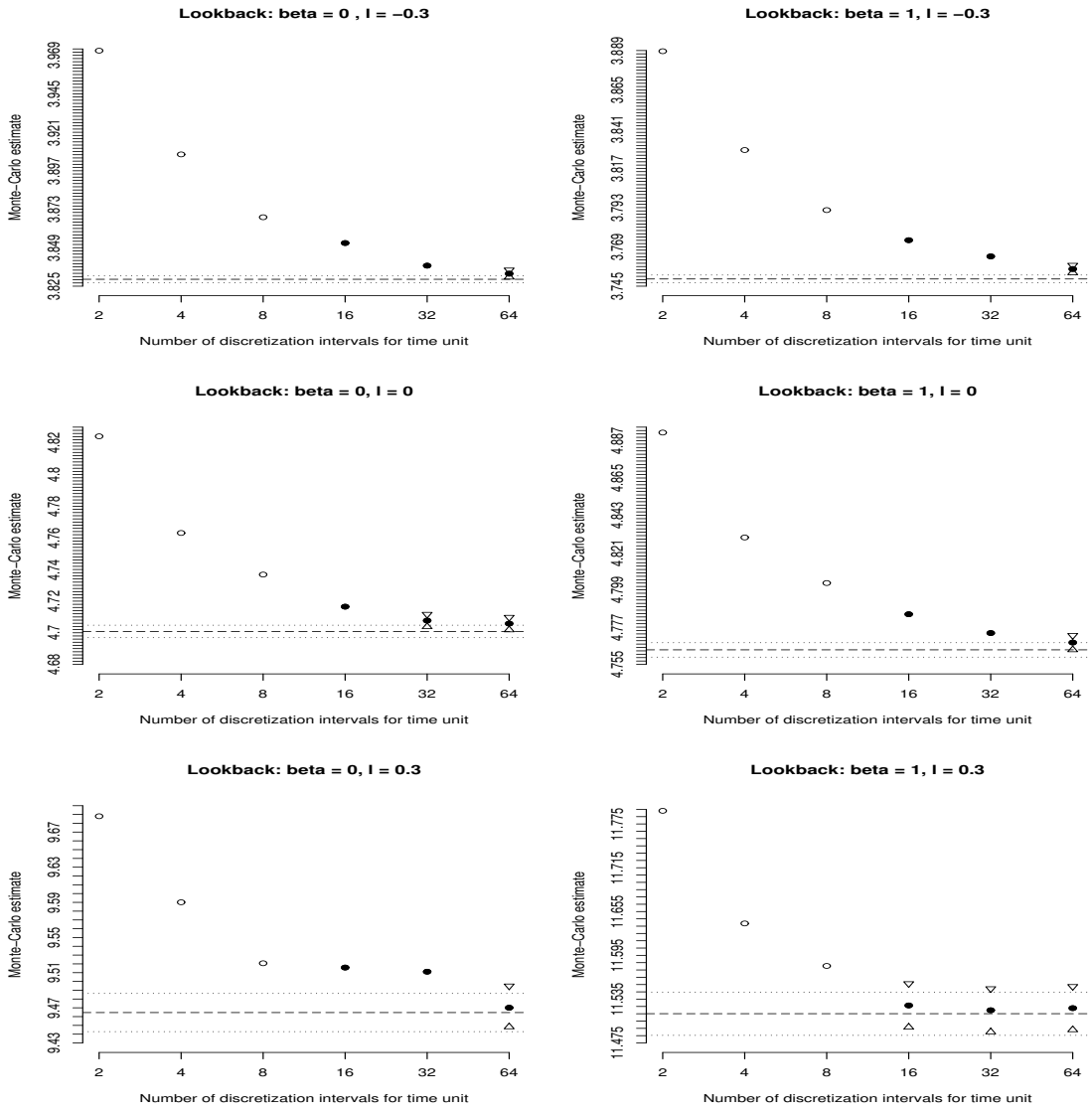

FIG 4. *Plot of the Monte Carlo estimates and 95%-confidence intervals of $E1$ (resp. dashed line and dotted lines) and $E3$ (resp. circle and triangles) Vs the number of discretisation intervals per time unit (2^n, $n = 0, 1, \ldots, 6$). Functional: LOOKBACK option $\phi = \max_{\{0 \le t \le 5\}} X_t$. Model: X (38) with fixed parameters $\alpha = \sigma = \lambda_0 = 1$ and $x_0 = 2$ and variable parameters β and l. Monte Carlo sample size: 5×10^5. Notes: We have reported only the confidence interval of $E3$ characterised by non-empty intersection with the corresponding confidence interval of $E1$. If no intersection occurs (plot: beta =1, l=-0.3) we have reported the confidence interval corresponding to the smalles discretisation step ($1/2^6$). The solid circles represent $E3$ estimates whose computational time is lower than $E1$.*

Numerical comparison							
Functional	β	l	Method	$\hat{\nu}$	C.I.(95%)	N	time
Asian	0	-0.3	E1	1.6251	[1.6209, 1.6293]		40
			E2	1.6323	[1.6281, 1.6364]	2^5	218
			E3	1.6305	[1.6264, 1.6347]	2^5	106
		0	E1	2.705	[2.7005, 2.7094]		35
			E2	2.7120	[2.7075, 2.7164]	2^5	216
			E3	2.7093	[2.7049, 2.7138]	2^4	58
		0.3	E1	5.3703	[5.359, 5.3816]		39
			E2	5.36	[5.3491, 5.3709]	2^4	110
			E3	5.3834	[5.3722, 5.3947]	2^5	107
	1	-0.3	E1	1.3088	[1.304, 1.3135]		41
			E2	1.3134	[1.3095, 1.319]	2^6	312
			E3	1.3180	[1.3131, 1.3229]	2^5	107
		0	E1	2.6655	[2.6608, 2.6702]		37
			E2	2.6703	[2.6656, 2.675]	2^5	160
			E3	2.6738	[2.6691, 2.6785]	2^4	58
		0.3	E1	6.2905	[6.2763, 6.3046]		40
			E2	6.2851	[6.2708, 6.2991]	2^5	161
			E3	6.3124	[6.2981, 6.3266]	2^4	58
Lookback	0	-0.3	E1	3.8293	[3.8271, 3.8316]		44
			E2	*3.7636*	[*3.7613*, *3.7658*]	2^6	*280*
			E3	3.8328	[3.8306, 3.8351]	2^6	244
		0	E1	4.7008	[4.6969, 4.7048]		39
			E2	*4.6306*	[*4.6266*, *4.6345*]	2^6	*281*
			E3	4.7077	[4.7038, 4.7117]	2^5	130
		0.3	E1	9.4646	[9.4429, 9.4863]		43
			E2	*9.3795*	[*9.3579*, *9.4011*]	2^6	*281*
			E3	9.4701	[9.4483, 9.4949]	2^6	247
	1	-0.3	E1	3.7494	[3.7471, 3.7518]		46
			E2	*3.6844*	[*3.6821*, *3.6867*]	2^6	*289*
			E3	*3.7555*	[*3.7532*, *3.7578*]	2^6	*243*
		0	E1	4.7636	[4.7593, 4.7678]		34
			E2	*4.7003*	[*4.6960*, *4.7043*]	2^6	*292*
			E3	4.7679	[4.7636, 4.7721]	2^6	244
		0.3	E1	11.5151	[11.4854, 11.5447]		43
			E2	*11.3992*	[*11.3699*, *11.4185*]	2^6	*275*
			E3	11.5264	[11.4966, 11.5563]	2^4	66

TABLE 1

Monte Carlo estimate, 95%-confidence intervals and computational time of E1 (JEA estimator), E2 (Euler estimator) and E3 (Euler + Thinning) with discretisation step $1/N$. Functionals: ASIAN ($\phi = \sum_{i=1}^{5} X_i$) and LOOKBACK option ($\phi = \max_{\{0 \le t \le 5\}} X_t$). Model: X (38) with fixed parameters $\alpha = \sigma = \lambda_0 = 1$ and $x_0 = 2$ and variable parameters β and l. Monte Carlo sample size: 5×10^5. Notes: The discretisation step selected for E2 or E3 is the largest in the range $\{1/2^n;\, n = 0, 1, \ldots, 6\}$ generating a confidence interval with non-empty intersection with the corresponding confidence interval of E1. Alternatively if no intersection occurs, we set $N = 2^6$ and the corresponding values of the estimate, confidence interval and computational time are identified by the italic font.