View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by LSE Research Online

Angelos Dassios and Hongbiao Zhao

Efficient simulation of clustering jumps with CIR intensity

Article (Accepted version) (Refereed)

Original citation: Dassios, Angelos and Zhao, Hongbiao (2017) *Efficient simulation of clustering jumps with CIR intensity.* Operations Research, 65 (6). pp. 1494-1515. ISSN 0030-364X
DOI: 10.1287/opre.2017.1640

© 2017 INFORMS

This version available at: http://eprints.lse.ac.uk/74205/
Available in LSE Research Online: January 2018

LSE has developed LSE Research Online so that users may access research output of the School. Copyright © and Moral Rights for the papers on this site are retained by the individual authors and/or other copyright owners. Users may download and/or print one copy of any article(s) in LSE Research Online to facilitate their private study or for non-commercial research. You may not engage in further distribution of the material or use it for any profit-making activities or any commercial gain. You may freely distribute the URL (http://eprints.lse.ac.uk) of the LSE Research Online website.

This document is the author's final accepted version of the journal article. There may be differences between this version and the published version. You are advised to consult the publisher's version if you wish to cite from it.

http://eprints.lse.ac.uk

Efficient Simulation of Clustering Jumps with CIR Intensity

Angelos Dassios*
London School of Economics

Hongbiao Zhao†
Shanghai University of Finance and Economics

Abstract

We introduce a broad family of generalised self-exciting point processes with CIR-type intensities, and develop associated algorithms for their exact simulation. The underlying models are extensions of the classical Hawkes process, which already has numerous applications in modelling the arrival of events with clustering or contagion effect in finance, economics and many other fields. Interestingly, we find that the CIR-type intensity together with its point process can be sequentially decomposed into simple random variables, which immediately leads to a very efficient simulation scheme. Our algorithms are also pretty accurate and flexible. They can be easily extended to further incorporate externally-excited jumps, or, to a multi-dimensional framework. Some typical numerical examples and comparisons with other well known schemes are reported in detail. In addition, a simple application for modelling a portfolio loss process is presented.

Keywords: Contagion risk; Jump clustering; Stochastic intensity model; Self-exciting point process; Self-exciting point process with CIR intensity; Hawkes process; CIR process; Square-root process; Exact simulation; Monte Carlo simulation; Portfolio risk

Mathematics Subject Classification (2010): Primary: 60G55; Secondary: 60H35 · 65C05 · 60G17

JEL Classification: C15 · C53 · C63

*Department of Statistics, London School of Economics, Houghton Street, London WC2A 2AE, United Kingdom. Email: a.dassios@lse.ac.uk

†*Corresponding author*, School of Statistics and Management, Shanghai University of Finance and Economics, No. 777 Guoding Road, Shanghai 200433, China. Email: h.zhao1@lse.ac.uk

1 Introduction

The *square-root diffusion process*, later called the *Cox-Ingersoll-Ross (CIR) process* (Cox et al., 1985) in finance, was initially studied by Feller (1951), and then widely applied in finance for modelling interest rates (Cox et al., 1985) and stochastic volatilities (Heston, 1993). The distributional properties of this process and its integral over time were well documented in Dufresne (2001) and Dassios and Nagaradjasarma (2006). On the other hand, the default risk, and more recently, the jump risk are extensively investigated in finance. However, most of the existing literature used simple *Poisson processes* or *Cox point processes*, which would not be able to capture contagion effect of jumps or defaults in markets particularly during crises. Self-exiting jump processes, in particular the *Hawkes process*, were then proposed for modelling the arrival of events (such as jumps, defaults, bankruptcies, crises, loss claims and catastrophes) with clustering, contagion, ripple effects, or herd behaviors, see methodologies developed in Hawkes (1971a,b), and their associated applications to finance, insurance and economics in Chavez-Demoulin et al. (2005), Bowsher (2007), Large (2007), Errais et al. (2010), Dassios and Zhao (2012) and Aït-Sahalia et al. (2014, 2015). Therefore, it is natural for us now to explore the idea of combining the properties of the CIR diffusion and self-exciting jumps together.

In this paper, we mainly use a mean-reverting CIR-diffusion process to model the intensity of a jump process, which is additionally excited by the jumps themselves, namely, a *self-exiting point process with CIR intensity*, which extends *a point process with CIR stochastic intensity* and the classical self-exiting Hawkes process (Hawkes, 1971a,b). This is a generalised Hawkes process with exponentially decaying intensity additionally perturbed by an independent mean-reverting diffusion. By adding a diffusion to the intensity of the classical Hawkes process, the model is then easier to be benchmarked to well-developed CIR-type models in the existing literature. In addition, for applications in finance, this diffusion component could be useful in capturing noises which are persistently and externally existing in markets. This process and its variants were also discussed in Giesecke and Kim (2007), Errais et al. (2010), Dassios and Zhao (2011), Giesecke et al. (2011b), Zhu (2014) and Zhang et al. (2015).

In practice, analytic formulas for many financial quantities (such as path-dependent option prices, CDS/CDO prices) based on this generalised self-exciting jump model are rather limited or difficult to be obtained. Hence, a computationally efficient Monte Carlo simulation scheme for generating this process becomes crucial for model implementation, simulation-based statistical inference and empirical studies.

Our aim here is to develop an *exact* simulation algorithm for this point process, rather than conventionally discretising the process and simulating approximately, as the discretisation introduces bias into simulation results (Giesecke and Kim, 2007). The exact simulation method has the primary advantage of generating sample paths according to the process law exactly (Chen and Huang, 2013).

Recently, efficient simulation algorithms for CIR-type stochastic processes were intensively discussed in the literature. Beskos and Roberts (2005) developed an exact simulation based on the *acceptance/rejection* (A/R) scheme for one-dimensional state-dependent diffusions. It was then extended by Chen and Huang (2013) to exactly simulate a more general class of diffusions via a *localisation* technique. Broadie and Kaya (2006) presented an exact simulation for the classical Heston (1993)'s stochastic volatility model. This algorithm has been further enhanced by Glasserman and Kim (2011) using a *Gamma expansion*.

On the other hand, a self-exciting Hawkes process can be simulated via a branching structure without any discretisation procedure, as it can be equivalently defined as a branching process via its *Poisson cluster representation*; see the associated algorithms first developed in Brix and Kendall (2002) and then extended by Møller and Rasmussen (2005, 2006). For the simulation within a finite time interval, their algorithms all require the stationary condition, and may suffer from *edge effects* or involve approximations. As an alternative, from a point of view based on the *conditional stochastic intensity representation*, the exact simulation for the Hawkes process and its variants (on $\mathbb{R}_+$-time) was first introduced by Giesecke and Kim (2007). An obvious problem as pointed by themselves in the conclusion is that, their method needs the numerical evaluation of the inverse of conditional distribution functions of interarrival times. So, a root-finding procedure is required which involves intensive computations. This work was later refined and generalised by Giesecke and Kim (2011) and Giesecke et al. (2011b) with the aid of an A/R scheme. Moreover, Giesecke et al. (2011a) developed the *projection method* for exactly sampling point processes with general state-dependent intensities via the sequential *thinning* (Lewis and Shedler, 1979) or A/R implementation. In addition, Giesecke and Smelov (2013) adopted an A/R scheme and further extended the exact simulation method of Beskos and Roberts (2005) to a more general framework of jump-diffusion processes with state-dependent coefficients and jump intensities.

We have to admit that, the popular A/R scheme for exact simulation indeed works for a very general class of point processes, whereas the efficiency varies. An evident disadvantage is the mechanism of A/R scheme itself: it brings the uncertainty about the

number of loops needed for simulating each step moving forward, as there is a possibility of rejecting the generated candidates; depending on the frequency of these rejections, one may end up with a substantially large number of loops. To avoid this problem, Dassios and Zhao (2013) first developed an efficient, *certain* and much simpler exact simulation algorithm for the most widely-used self-exciting point process, i.e. a Hawkes process with exponentially decaying intensity (or exponential kernel) and random marks. This scheme is applicable to both one-dimensional and multi-dimensional cases. The key idea is to decompose the processes into simple random variables sequentially: each interarrival time can be generated exactly and *certainly* as the minimum of two simple intermediate random variables (one exponential random variable and one simple defective random variable). This *certain scheme* needs more detailed knowledge about the distribution of the underlying process than the conventional A/R method, and requires the targeted process to be more specific, and attempts to identify the underlying distribution as exactly as possible. So, it may not be applicable as generally as the A/R method. However, it has many advantages: it avoids the numerical inversion of distribution functions, it also does not involve any approximation or truncation error, and is free of any stationary condition. Moreover, it does not involve any A/R procedure: generating just two simple intermediate random variables can guarantee successfully generating one interarrival time without wasting (or rejecting) any candidate. Hence, the resulting simulation speed is independent of the choices of parameters.

The new approach developed in this paper makes a non-trivial extension of the certain scheme of Dassios and Zhao (2013). Obviously, our current targeted process is much more complicated and difficult to be simulated: the trajectories of the CIR-type stochastic intensity between two successive jumps (i.e. interarrival intensity processes) are no longer deterministic as in the Hawkes process. To deal with this problem, we have to use the integral transforms of joint distributions of the self-exciting point process with CIR intensity to derive the transition densities as the basis for the algorithm design of exact simulation. Our approach is mainly based on the distributional decomposition for the process. The interarrival times and the intensity levels at the jump arrival times can be sequentially decomposed into simple random variables (uniform, Poisson and Gamma), so they can be exactly and efficiently simulated, and the entire continuous-time path simulation for the intensity can be avoided. In fact, each interarrival time can be generated as the minimum of two intermediate random variables: one is a simple defective random variable, and the other one is a well-defined random variable. The later one is the only random variable that needs be generated via a modified (or transformed) A/R scheme, as to our knowledge its distribution can not be decomposed into simpler ones any fur-

ther. More interestingly and elegantly, conditional on the realisation of the interarrival time, the associated intensity level at each jump time then can be simply generated from two simple Gamma random variables by conditioning on a single realisation of Poisson random variable.

The key methodological difference from many other relevant literature is that, here our principle is to use less A/R scheme as possible, in order to make most use of simulation candidates generated. It means that, if we are able to decompose the distribution of a random variable explicitly, we will abandon the associated A/R scheme, and hence reduce the uncertainty from the number of loops needed for each step moving forward during the simulation. This principle of algorithm design leads our scheme to be very accurate, efficient and flexible: 1) It has no bias or truncation error, and does not involve any numerical inversion or root finding procedure. 2) Conventional conditions for the CIR and Hawkes processes[1] are not required. 3) Jumps are allowed to be non-additive. 4) Both of the cases of stationary and non-stationary (unbounded) intensities can be generated. 5) The path simulation can start from any arbitrary time and any arbitrary initial intensity without truncation or approximation error. 6) In particular, it is also applicable to exactly simulating a point process with pure CIR intensity as a special case. Indeed, this is an important special case, which has many useful applications in practice. For example, Glasserman and Kim (2011) in the conducing remarks of their paper also suggested using a Gamma expansion to simulate this process. However, their approach would introduce truncation errors, as the expansion creates infinite summations which need the numerical truncation. Our scheme is exact, thereby it could provide an improvement over their algorithm for simulating this process. 7) More generally, our algorithm can be easily adjusted to simulate a broad family of more general self-exciting point processes with CIR-type intensities. For example, it can further integrate an additional series of externally-excited jumps in the intensity process, which may be useful for modelling external risk factors. 8) Furthermore, it can be easily extended to a multi-dimensional mutually-exciting framework.

The paper is organised as follows. Section 2 provides the preliminaries, a definition and some basic distributional properties for this self-exciting point process with CIR intensity. Section 3 investigates the distributional properties of the transition of this process, and develops the numerical algorithm for exactly generating a sample path of the point

[1]The conventional condition for CIR-type processes is the Feller's condition granting that the processes are strictly positive almost surely and the origin is inaccessible (Feller, 1951); the conventional condition for Hawkes process is the stationary condition that grants its intensity process to be stationary. We will specify these two important conventional conditions later, however, both are not required in all of our algorithms developed in this paper.

process in one dimension. Numerical examples (including some unconventional cases that are often hard to be generated by other algorithms in the current existing literature) with the associated error and convergence analysis are also demonstrated. Our key algorithm is summarised in Algorithm 3.6. Numerical comparisons with other algorithms (such as the classical discretisation scheme and projection scheme) are conducted and reported in Section 4. Some important extensions such as a multi-dimensional version are provided in Section 5. In Section 6, we apply our method to modelling a portfolio loss process, and cumulative loss distributions for different scenarios are computed by our exact simulation.

2 Preliminaries

A self-exciting point process with CIR intensity provides a very general framework for modelling event arrivals, and it extends a *Cox process with CIR intensity* and a self-exciting Hawkes process. We provide an intensity-based definition for this point process in Definition 2.1 as below.

Definition 2.1 (Self-exciting Point Process with CIR Intensity)**. A self-exciting point process with CIR intensity** is a counting process

$$N_t = \sum_{i \geq 1} \mathbb{1}\{T_i^* \leq t\},$$

with non-negative $\mathcal{F}_t-$stochastic (conditional) intensity

$$\lambda_t = a + (\lambda_0 - a)\, e^{-\delta t} + \sigma \int_0^t e^{-\delta(t-s)} \sqrt{\lambda_s} \mathrm{d}W_s + \sum_{0 \leq T_i^* < t} Y_i e^{-\delta(t-T_i^*)}, \qquad t \geq 0, \qquad (1)$$

where

- $\mathbb{1}\{\cdot\}$ is the indicator function;
- $\left\{T_i^*\right\}_{i=1,2,\ldots}$ on $\mathbb{R}_+$ are the associated (ordered) *arrival times* of point N_t, i.e. $N \equiv \left\{T_i^*\right\}_{i=1,2,\ldots}$;
- $\{\mathcal{F}_t\}_{t \geq 0}$ is a history of the process $\{N_t\}_{t \geq 0}$ with respect to which $\{\lambda_t\}_{t \geq 0}$ is adapted;
- $\lambda_0 > 0$ is the initial intensity at time $t = 0$;
- $a \geq 0$ is the constant of *reversion level*;
- $\delta > 0$ is the constant of *reversion rate*;
- $\sigma > 0$ is the constant that governs the diffusive volatility of intensity;

- $\{W_t\}_{t\geq 0}$ is a standard Brownian motion independent of N_t;
- $\{Y_i\}_{i=1,2,...}$ are the sizes (or *marks*) of *self-excited jumps*, a sequence of nonnegative random variables with $\mathcal{F}_{T_i^{*-}}$-measurable distribution function $G(y), y \geq 0$ [2].

N_t is a point process which is completely characterised by the intensity process λ_t and satisfies

$$\Pr\left\{N_{t+\Delta t} - N_t = 1 \,\middle|\, \mathcal{F}_t\right\} = \lambda_t \Delta t + o(\Delta t), \qquad \Pr\left\{N_{t+\Delta t} - N_t > 1 \,\middle|\, \mathcal{F}_t\right\} = o(\Delta t),$$

where Δt is a sufficiently small time interval, and $o(\Delta t)/\Delta t \to 0$ when $\Delta t \to 0$. The conditional intensity function (1) can be equivalently expressed by the jump-diffusion stochastic differential equation (SDE)

$$\mathrm{d}\lambda_t = \delta(a - \lambda_t)\mathrm{d}t + \sigma\sqrt{\lambda_t}\mathrm{d}W_t + \mathrm{d}J_t, \qquad t \geq 0,$$

where the jump process $J_t := \sum_{i=1}^{N_t} Y_i$ is a *compound self-exciting point process with CIR intensity*. The $(i+1)^{\text{th}}$ *interarrival time* or *duration* (Engle and Russell, 1998) is defined by

$$S_{i+1}^* := T_{i+1}^* - T_i^* > 0, \qquad i = 0, 1, 2, ..., \qquad T_0^* = 0.$$

The cumulative intensity process (or the *compensator* of point process) is denoted by $\Lambda_t := \int_0^t \lambda_u \mathrm{d}u$, and the mean of self-excited jump sizes is denoted by $\mu_{1_G} := \int_0^\infty y \mathrm{d}G(y)$. A sample path of the intensity process λ_t simulated by the discretisation scheme is presented in Figure 1.

This self-exciting point process with CIR intensity is a special case of *affine point processes*. If N_t is an independent Poisson process, then, λ_t is a *basic affine jump-diffusion process*, which has been already widely used for pricing in finance, see Duffie et al. (2000, 2003). We provide the expectation of N_t conditional on λ_0 below in Proposition 2.1, which can be easily proved by explicitly solving ODEs, see Errais et al. (2010) in general. Note that, this formula in a simple analytic form is not subject to either the conventional condition for the CIR process or the stationary condition for the Hawkes process, and hence

[2] It means that the distribution function $G(y)$ is revealed just before the arrival time T_i^*. This distribution could have a highly general dependency structure $G(y) = G(y \mid \cdot)$. For example, it could depend on the initial intensity λ_0, the past history of intensity at or just before the jump arrival times $\{T_k^*\}_{k=1,2,...,i}$, all past jump sizes $\{Y_k\}_{k=1,2,...,i-1}$, or cumulated number of jumps N_t, and so on, as long as we can record these information, i.e. in general, we have

$$G(y) = G\left(y \mid T_1^*, T_2^*, ..., T_i^*, \lambda_0, \lambda_{T_1^{*-}}, ..., \lambda_{T_i^{*-}}, \lambda_{T_1^*}, ..., \lambda_{T_{i-1}^*}\right).$$

This is similar as the adaptive model setting of Giesecke et al. (2011a), but is different from the classical Hawkes process. We will provide some specified examples later.

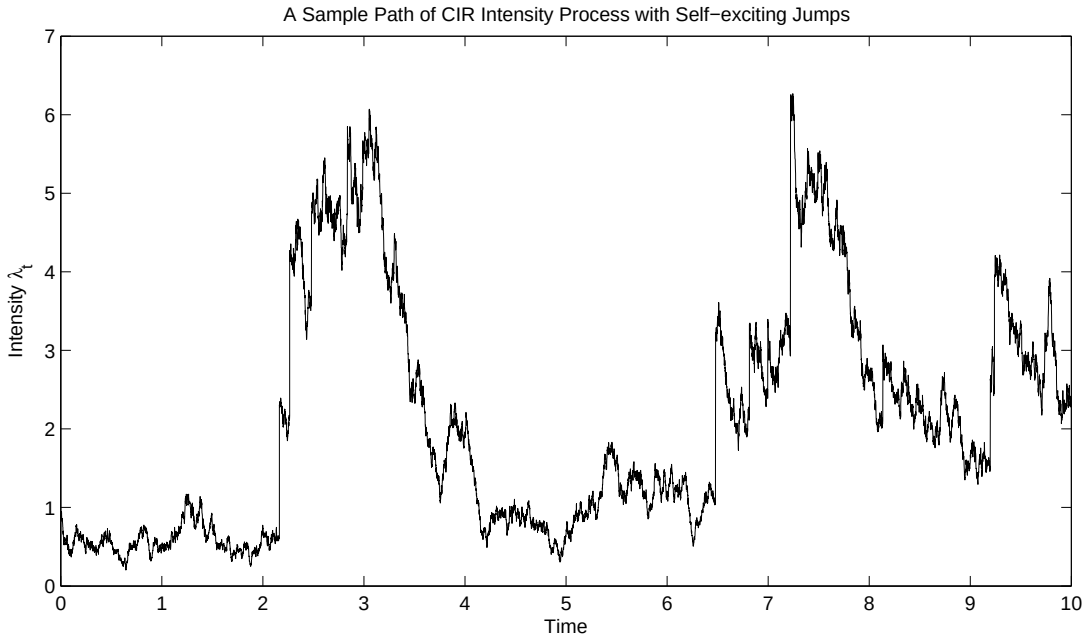


Figure 1: A sample path of the CIR intensity process with self-exciting jumps

it is convenient to be used later in Section 3 for numerically validating our simulation algorithm and measuring the associated errors.

Proposition 2.1. The expectation of N_T conditional on λ_0 for a fixed time $T > 0$ is given by

$$\mathbb{E}[N_T \mid \lambda_0] = \begin{cases} \frac{a\delta}{\xi}T + \frac{1}{\xi}\left(\lambda_0 - \frac{a\delta}{\xi}\right)\left(1 - e^{-\xi T}\right), & \xi \neq 0, \\ \lambda_0 T + \frac{1}{2}a\delta T^2, & \xi = 0, \end{cases}$$

where $\xi := \delta - \mu_{1_G}$.

3 Exact Simulation

In this section, based on the joint distributional properties of pre-jump intensity levels and interarrival times, we first develop the exact simulation for pre-jump interarrival times. Then, conditional on the realisation of these interarrival times, the pre-jump intensities at arrival times can be simulated exactly. Finally, by adding self-exited jumps to the intensity process, the entire series of interarrival times of the process $\{(N_t, \lambda_t)\}_{t\geq 0}$ can be generated exactly.

3.1 Joint Distribution of Pre-jump Intensity and Interarrival Time

We provide the transition law from the location $\left(\lambda_{T_i^*}, T_i^*\right)$ to the location $\left(\lambda_{T_i^*+S_{i+1}^{*-}}, T_i^* + S_{i+1}^{*-}\right)$ as below in Proposition 3.1.

Proposition 3.1. Conditional on the i^{th} jump time T_i^* and the intensity level $\lambda_{T_i^*}$, we can characterise the joint distribution of $\left(\lambda_{T_i^*+s^-}, \Lambda_{T_i^*+s} - \Lambda_{T_i^*}\right)$ via a special integral transform of the CIR process (without jumps) starting from T_i^*, i.e.

$$\mathbb{E}\left[e^{-v\lambda_{T_i^*+s^-}} e^{-\left(\Lambda_{T_i^*+s}-\Lambda_{T_i^*}\right)} \,\middle|\, \lambda_{T_i^*}\right] = \left(\frac{A_s}{B_s v + C_s}\right)^D \exp\left(-\frac{E_s v + F_s}{B_s v + C_s}\lambda_{T_i^*}\right), \qquad s \in [0, S_{i+1}^*), \tag{2}$$

where

$$\kappa = \sqrt{\delta^2 + 2\sigma^2}, \qquad D = \frac{2a\delta}{\sigma^2}, \qquad A_s = 2\kappa e^{\frac{\kappa+\delta}{2}s}, \qquad B_s = \sigma^2(e^{\kappa s} - 1), \tag{3}$$

$$C_s = (\kappa - \delta) + (\kappa + \delta)e^{\kappa s}, \qquad E_s = (\kappa + \delta) + (\kappa - \delta)e^{\kappa s}, \qquad F_s = 2\left(e^{\kappa s} - 1\right). \tag{4}$$

Obviously, $\kappa, D, A_s, B_s, C_s, E_s, F_s$ are all positive and $E_s C_s > F_s B_s$. Note that, (2) is a special case of the bivariate Laplace transform of the intensity and its integral $\left(\lambda_{T_i^*+s^-}, \Lambda_{T_i^*+s} - \Lambda_{T_i^*}\right)$, which is given by Proposition 6.2.4 in Lamberton and Lapeyre (2008, p.162), and also see the equation (3.76) in Glasserman (2003, p.129). We provide a proof for Proposition 3.1 using the martingale approach in the electronic companion of this paper online.

Based on the integral transform of joint distribution in Proposition 3.1, we will first simulate the pre-jump interarrival time S_{i+1}^*, and then, by conditioning on the realisation of $S_{i+1}^* = s$, the pre-jump intensity level $\lambda_{T_i^*+S_{i+1}^{*-}}$ can be simulated.

3.2 Exact Simulation of Pre-jump Interarrival Times

As later given by Theorem 3.3, any pre-jump interarrival time S_{i+1}^* can be exactly simulated as the minimum of two intermediate random variables, S^* and V_{i+1}^*, where S^* is well defined and V_{i+1}^* is defective. Note that, S^* is independent of the step index i which can be simulated separately, however, it can not be simulated by distributional decomposition. Hence, we develop a simulation algorithm for S^* based on an A/R scheme in Lemma 3.2. It should be noted that, S^* is the only intermediate random variable in this paper that involves an A/R algorithm for simulation. A numerical test is also provided which shows that this A/R algorithm for S^* is very efficient and substantially outperforms the conventional scheme of numerical inversion.

Lemma 3.2. The random variable S^* defined by the tail distribution

$$\Pr\{S^* > s\} = \left[\frac{2\kappa e^{\frac{\kappa+\delta}{2}s}}{(\kappa+\delta)\left(e^{\kappa s} - 1\right) + 2\kappa}\right]^{\frac{2a\delta}{\sigma^2}} \tag{5}$$

can be exactly simulated by a modified (or transformed) A/R scheme as follows:

1. Generate a generalised Pareto random variable W_g via

$$W_g = \frac{2\kappa}{\kappa+\delta}\left[U_1^{-\frac{\sigma^2}{a\delta\kappa(\kappa-\delta)}} - 1\right], \qquad U_1 \sim \mathcal{U}[0,1]; \tag{6}$$

2. Generate a uniformly distributed random variable $U_2 \sim \mathcal{U}[0,1]$;

3. If

$$U_2 \leq \left(\frac{\kappa+\delta}{2\kappa}\right)^{\frac{\kappa+\delta}{2\kappa}\frac{2a\delta}{\sigma^2}} \left(\frac{W_g+1}{\frac{2\kappa}{\kappa+\delta}+W_g}\right)^{\frac{a\delta(\kappa+\delta)}{\sigma^2\kappa}} \frac{W_g}{W_g+1},$$

then, accept and set

$$S^* = \frac{1}{\kappa}\ln(W_g+1); \tag{7}$$

4. Otherwise, reject and go back to Step 1.

Proof. The random variable S^* is well defined, since we have the cumulative distribution function (CDF) of S^*,

$$F_{S^*}(s) := \Pr\{S^* \leq s\} = 1 - \left[\frac{2\kappa e^{\frac{\kappa+\delta}{2}s}}{(\kappa+\delta)(e^{\kappa s}-1)+2\kappa}\right]^{\frac{2a\delta}{\sigma^2}}, \tag{8}$$

with $F_{S^*}(0) = 0$, $F_{S^*}(\infty) = 1$ and the density

$$f_{S^*}(s) := \frac{\mathrm{d}}{\mathrm{d}s}F_{S^*}(s) = \left[\frac{2\kappa e^{\frac{\kappa+\delta}{2}s}}{(\kappa+\delta)(e^{\kappa s}-1)+2\kappa}\right]^{\frac{2a\delta}{\sigma^2}} \frac{\sigma^2(e^{\kappa s}-1)}{(\kappa+\delta)(e^{\kappa s}-1)+2\kappa} > 0, \qquad s > 0.$$

We define the random variable $W := e^{\kappa S^*} - 1$. Given the tail distribution of S^* in (5), the tail distribution of W is

$$\Pr\{W > w\} = \left(\frac{2\kappa}{\kappa+\delta}\right)^{\frac{2a\delta}{\sigma^2}} (w+1)^{\frac{2a\delta}{\sigma^2}\frac{\kappa+\delta}{2\kappa}} \left(\frac{2\kappa}{\kappa+\delta}+w\right)^{-\frac{2a\delta}{\sigma^2}}, \qquad w \in [0,\infty),$$

and the density of W is

$$f_W(w) = \frac{2a\delta}{\sigma^2}\frac{\kappa-\delta}{2\kappa}\left(\frac{2\kappa}{\kappa+\delta}\right)^{\frac{2a\delta}{\sigma^2}} \left(\frac{2\kappa}{\kappa+\delta}+w\right)^{-\frac{2a\delta}{\sigma^2}-1} w(w+1)^{\frac{2a\delta}{\sigma^2}\frac{\kappa+\delta}{2\kappa}-1}.$$

Comparing this with the density function of a generalised Pareto random variable W_g with the location parameter 0, scale parameter $\frac{\sigma^2}{2a\delta}\frac{2\kappa}{\kappa-\delta}$ and shape parameter $\frac{\sigma^2}{2a\delta}\frac{2\kappa}{\kappa-\delta}\frac{2\kappa}{\kappa+\delta}$, i.e. with density

$$f_{W_g}(w) = \frac{2a\delta}{\sigma^2}\frac{\kappa-\delta}{2\kappa}\left(\frac{2\kappa}{\kappa+\delta}\right)^{\frac{2a\delta}{\sigma^2}\frac{\kappa-\delta}{2\kappa}} \left(\frac{2\kappa}{\kappa+\delta}+w\right)^{-\frac{2a\delta}{\sigma^2}\frac{\kappa-\delta}{2\kappa}-1}, \qquad w \in [0,\infty),$$

and the CDF of W_g is

$$F_{W_g}(w) = 1 - \left(1 + \frac{\kappa+\delta}{2\kappa} w\right)^{-\frac{2a\delta}{\sigma^2}\frac{\kappa-\delta}{2\kappa}}, \qquad w \in [0,\infty).$$

It is much easier to simulate W_g than W, as W_g can be simulated directly via (6).

Since

$$\frac{f_W(w)}{f_{W_g}(w)} = \left(\frac{2\kappa}{\kappa+\delta}\right)^{\frac{2a\delta}{\sigma^2}\frac{\kappa+\delta}{2\kappa}} \left(\frac{w+1}{\frac{2\kappa}{\kappa+\delta}+w}\right)^{\frac{2a\delta}{\sigma^2}\frac{\kappa+\delta}{2\kappa}} \frac{w}{w+1},$$

and

$$\frac{f_W(w)}{f_{W_g}(w)} \le \bar{c}, \qquad \forall w \in [0,\infty),$$

where

$$\bar{c} = \left(\frac{2\kappa}{\kappa+\delta}\right)^{\frac{\kappa+\delta}{2\kappa}\frac{2a\delta}{\sigma^2}} > 1, \tag{9}$$

we can generate W by the classical A/R method (Glasserman, 2003; Asmussen and Glynn, 2007). Finally, we can generate S^* by taking a simple transformation of (7). □

Note that, our A/R scheme developed in Lemma 3.2 for simulating S^* is not subject to the CIR's conventional condition (i.e. Feller's condition) $2\delta a \ge \sigma^2$ or the Hawkes' stationary condition $\delta > \mu_{1_G}$.

Provided the intermediate random variable S^* as simulated by Lemma 3.2, we can then simulate the pre-jump interarrival time S^*_{i+1} via a simple distributional decomposition as given in Theorem 3.3.

Theorem 3.3. *Conditional on the initial intensity $\lambda_{T_i^*}$, the next interarrival time S^*_{i+1} can be exactly simulated via*

$$S^*_{i+1} \overset{\mathcal{D}}{=} \begin{cases} S^* \wedge V^*_{i+1}, & d_{i+1} > 0, \\ S^*, & d_{i+1} < 0, \end{cases} \tag{10}$$

where

- *d_{i+1} is simulated via*

$$d_{i+1} := \frac{1 + \frac{\ln U_3}{2\lambda_{T_i^*}}(\kappa+\delta)}{1 - \frac{\ln U_3}{2\lambda_{T_i^*}}(\kappa-\delta)}, \qquad U_3 \sim \mathcal{U}[0,1],$$

- *S^* is exactly simulated by Lemma 3.2;*

- *V^*_{i+1} is a simple defective random variable with $\Pr\{V^*_{i+1} = \infty\} = \exp\left(-\frac{2}{\delta+\kappa}\lambda_{T_i^*}\right)$, and it is exactly simulated via*

$$V^*_{i+1} \overset{\mathcal{D}}{=} -\frac{1}{\kappa}\ln d_{i+1}. \tag{11}$$

Proof. Setting $v = 0$ in (2), we have the marginal tail distribution of S^*_{i+1} conditional on $\lambda_{T^*_i}$,

$$\begin{aligned} & \Pr\left\{S^*_{i+1} > s \mid \lambda_{T^*_i}\right\} \qquad (12)\\ = \;& \mathbb{E}\left[\exp\left(-\int_{T^*_i}^{T^*_i+s} \lambda_u \mathrm{d}u\right) \;\middle|\; \mathcal{F}_{T^*_i}\right] \\ = \;& \mathbb{E}\left[e^{-\left(\Lambda_{T^*_i+s}-\Lambda_{T^*_i}\right)} \;\middle|\; \mathcal{F}_{T^*_i}\right] \\ = \;& \left[\frac{2\kappa e^{\frac{\kappa+\delta}{2}s}}{(\kappa+\delta)\left(e^{\kappa s}-1\right)+2\kappa}\right]^{\frac{2a\delta}{\sigma^2}} \exp\left(-\frac{2\left(e^{\kappa s}-1\right)}{(\kappa+\delta)\left(e^{\kappa s}-1\right)+2\kappa}\lambda_{T^*_i}\right), \qquad (13) \end{aligned}$$

which is a well-defined distribution function, as obviously, if $s \to 0$, $\Pr\{S^*_{i+1} > s\} \to 1$ and if $s \to \infty$, $\Pr\{S^*_{i+1} > s\} \to 0$.

It is impossible to invert the probability function (13) analytically, however, we can decompose S^*_{i+1} into two simpler and independent random variables S^* and V^*_{i+1} by

$$S^*_{i+1} \overset{\mathcal{D}}{=} S^* \wedge V^*_{i+1},$$

i.e.

$$\Pr\{S^*_{i+1} > s\} = \Pr\{S^* \wedge V^*_{i+1} > s\} = \Pr\{S^* > s\} \times \Pr\{V^*_{i+1} > s\},$$

where their tail distributions are specified respectively by

$$\begin{aligned} \Pr\{S^* > s\} &= \left[\frac{2\kappa e^{\frac{\kappa+\delta}{2}s}}{(\kappa+\delta)\left(e^{\kappa s}-1\right)+2\kappa}\right]^{\frac{2a\delta}{\sigma^2}}, \\ \Pr\{V^*_{i+1} > s\} &= \exp\left(-\frac{2\left(e^{\kappa s}-1\right)}{\kappa-\delta+(\delta+\kappa)e^{\kappa s}}\lambda_{T^*_i}\right). \end{aligned}$$

Note that, S^* is given by Lemma 3.2, and it is independent of the step index i and the intensity level $\lambda_{T^*_i}$. V^*_{i+1} is a defective random variable with the CDF

$$\begin{aligned} F_{V^*_{i+1}}(s) &=: \Pr\{V^*_{i+1} \le s\} = 1 - \exp\left(-\frac{2\left(e^{\kappa s}-1\right)}{(\kappa+\delta)\left(e^{\kappa s}-1\right)+2\kappa}\lambda_{T^*_i}\right), \\ F_{V^*_{i+1}}(\infty) &= 1 - \exp\left(-\frac{2}{\delta+\kappa}\lambda_{T^*_i}\right) < 1, \end{aligned}$$

and the density

$$f_{V^*_{i+1}}(s) = \exp\left(-\frac{2\left(e^{\kappa s}-1\right)}{(\kappa+\delta)\left(e^{\kappa s}-1\right)+2\kappa}\lambda_{T^*_i}\right) 2\lambda_{T^*_i} \frac{2\kappa^2 e^{\kappa s}}{\left((\kappa+\delta)\left(e^{\kappa s}-1\right)+2\kappa\right)^2} > 0.$$

Hence, V^*_{i+1} is a simple defective random variable defined by $\Pr\{V^*_{i+1} = \infty\} = \exp\left(-\frac{2}{\delta+\kappa}\lambda_{T^*_i}\right)$, and it can be simulated via (11) conditional on $d_{i+1} > 0$.

Finally, we can simulate S^*_{i+1} via (10). □

For simulating S^*, one may wonder whether the associated numerical inversion scheme would be more efficient than this A/R scheme in Lemma 3.2. So we carry out a numerical test and compare the performance of the A/R algorithm with the associated inversion scheme, by setting the parameters $(a, \lambda_0, \delta, \sigma) = (0.9, 0.9, 1.0, 1.0)$ with the total number of simulation[3] replications $n = 100,000$. The simulated results via the A/R scheme for $\Pr\{S^* > s\}$, the corresponding theoretical (true) values of (5) and the error percentages (Error%)[4] for measuring the accuracy are reported in Table 1. The simulation experiments for the all 10 different values of s only take about 0.37 seconds and achieve a high level of accuracy. Alternatively, the random variable S^* can be simulated by numerically inverting its CDF $F_{S^*}(s)$ of (8). With the same parameter setting and the number of replications, we can recreate Table 1. It has a similar level of accuracy but a much lower speed (69.6 seconds) than the A/R scheme. To save space, the associated table is not provided here. Hence, the A/R scheme obviously outperforms the inversion scheme.

Table 1: Comparison between the theoretical formulas and the associated simulation results for $\Pr\{S^* > s\}$ with the parameter setting $(a, \lambda_0, \delta, \sigma) = (0.9, 0.9, 1.0, 1.0)$ based on the total number of simulation replications $n = 100,000$ by the A/R scheme of Lemma 3.2

s	$\Pr\{S^* > s\}$	Simulation	Error%
0.1	99.57%	99.56%	0.01%
0.2	98.33%	98.26%	0.08%
0.3	96.42%	96.35%	0.07%
0.4	93.93%	94.00%	-0.07%
0.5	91.00%	91.03%	-0.03%
0.6	87.72%	87.78%	-0.07%
0.7	84.21%	84.23%	-0.02%
0.8	80.53%	80.63%	-0.12%
0.9	76.77%	76.86%	-0.11%
1.0	72.99%	73.07%	-0.11%

Remark 3.4. The efficiency of the A/R scheme of Lemma 3.2 can even be further enhanced. The time needed to obtain a qualified candidate for W or S^* is random, and the expected number of random variables we will need before the acceptance is the constant $\bar{c}$ as given by (9). Hence, to search the optimal efficiency, we want $\bar{c}$ to be as small as possible, such that fewer simulated candidate samples of W_g would be wasted. The efficiency

[3]The simulation experiments here and in the other parts of this paper are conducted on a normal desktop PC with Intel Core i7-3770 CPU@3.40GHz processor, 8.00GB RAM, Windows 7 64-bit Operating System, the algorithms are codded and performed in MatLab (R2013b), and the computing time is measured by the elapsed CPU time (in seconds).

[4]The error percentage for measuring the accuracy is simply calculated by taking the difference between the value estimated by simulation and the corresponding true value from (5) and then dividing by the true value, i.e.

$$\text{Error\%} = \frac{\text{Value Estimated by Simulation} - \text{True Value}}{\text{True Value}}.$$

of this mainly depends on the value of $\frac{2a\delta}{\sigma^2}$. The rest is at most $\sqrt{2}$ which is achieved when κ and therefore σ^2 is large but then $\frac{2a\delta}{\sigma^2}$ will not be. The problem is that $\bar{c}$ increases exponentially with a. This can be addressed by the following refinement when a is relatively large: we note that $\Pr\{W > w\} = \bar{H}(w)^a$, where $\bar{H}(\cdot)$ is the tail of a distribution. We can repeat the procedure m times with a replaced by $\frac{a}{m}$ and generate independent $W_1, W_2, \ldots, W_m$, then, take $W = \min\{W_1, W_2, \ldots, W_m\}$. The expected number of random variables to be generated is

$$\bar{C}(m) := m \times \left(\frac{2\kappa}{\kappa+\delta}\right)^{\frac{\kappa+\delta}{2\kappa}\frac{2a\delta}{m\sigma^2}}.$$

Therefore, we choose m to be the integer that minimises $\bar{C}(m)$. Obviously, the minimum of this function is achieved at

$$m^* := \min_m \left\{\bar{C}(m)\right\} = \frac{2a\delta}{\sigma^2}\frac{\kappa+\delta}{2\kappa}\ln\left(\frac{2\kappa}{\kappa+\delta}\right),$$

so we now take m to be the integer either side of m^* that produces the lowest value, i.e. we choose integer $m = m^*$ if $\bar{C}([m^*]) < \bar{C}([m^*]+1)$, otherwise $m = [m^*]+1$, where $[m^*]$ is the largest integer not greater than m^*. Therefore, the expected number of random variables needed grows approximately linearly with a rather than exponentially.

3.3 Exact Simulation of Pre-jump Intensities

Given the pre-jump interarrival time S^*_{i+1} as simulated by Theorem 3.3, we can derive the distribution explicitly for the pre-jump intensity level $\lambda_{T_i^*+S^{*-}_{i+1}}$, so it can be simulated exactly as given by Theorem 3.5 without any A/R scheme. Note that, only a skeleton of the intensity process (at these jump times) is exactly simulated here rather than the entire continuous-time path. However, this will not hinder us from exactly simulating the entire continuous-time path of the point process N_t afterward.

Theorem 3.5. *Conditional on the intensity level $\lambda_{T_i^*}$ and the realisation of interarrival time $S^*_{i+1} = s$, we have the distributional decomposition*

$$\lambda_{T_i^*+s^-} \overset{\mathcal{D}}{=} \begin{cases} \text{Gamma}\left(J^*_s + D + 1, \frac{C_s}{B_s}\right), & \textit{with probability } w_{1s}, \\ \text{Gamma}\left(J^*_s + D + 2, \frac{C_s}{B_s}\right), & \textit{with probability } w_{2s}, \end{cases}$$

where

$$J^*_s \sim \text{Poisson}\left(\lambda_{T_i^*}\left(\frac{E_s}{B_s} - \frac{F_s}{C_s}\right)\right), \tag{14}$$

$$w_{1s} = \frac{DB_s}{DB_s + \lambda_{T_i^*}\left(E_s - F_s\frac{B_s}{C_s}\right)}, \qquad w_{2s} = \frac{\lambda_{T_i^*}\left(E_s - F_s\frac{B_s}{C_s}\right)}{DB_s + \lambda_{T_i^*}\left(E_s - F_s\frac{B_s}{C_s}\right)}, \tag{15}$$

and B_s, C_s, D, E_s, F_s are specified by (3) and (4).

Proof. Note that, $\lambda_{T_i^*+s^-}\exp\left(-\int_{T_i^*}^{T_i^*+s}\lambda_u \mathrm{d}u\right)$ is the density of the interarrival time S_{i+1}^* conditional on $\lambda_{T_i^*+s^-}$ and $\Lambda_{T_i^*+s}-\Lambda_{T_i^*}$, and here we concisely denote it by

$$\Pr\left\{S_{i+1}^* \in \mathrm{d}s\right\} = \lambda_{T_i^*+s^-} e^{-\left(\Lambda_{T_i^*+s}-\Lambda_{T_i^*}\right)}\mathrm{d}s,$$

so, we have

$$\mathbb{E}\left[e^{-v\lambda_{T_i^*+S_{i+1}^{*-}}}\mathbb{1}\{S_{i+1}^* \in \mathrm{d}s\}\right] = \mathbb{E}\left[e^{-v\lambda_{T_i^*+s^-}}\lambda_{T_i^*+s^-}e^{-\left(\Lambda_{T_i^*+s}-\Lambda_{T_i^*}\right)}\right]\mathrm{d}s.$$

Using (2) in Proposition 3.1, we have

$$\begin{aligned}
&\mathbb{E}\left[\lambda_{T_i^*+s^-}e^{-v\lambda_{T_i^*+s^-}}e^{-\left(\Lambda_{T_i^*+s}-\Lambda_{T_i^*}\right)}\right]\\
=\;& -\frac{\partial}{\partial v}\mathbb{E}\left[e^{-v\lambda_{T_i^*+s^-}}e^{-\left(\Lambda_{T_i^*+s}-\Lambda_{T_i^*}\right)}\right]\\
=\;& \frac{1}{B_s v+C_s}\left(DB_s+\lambda_{T_i^*}\frac{E_sC_s-F_sB_s}{B_s v+C_s}\right)\left(\frac{A_s}{B_s v+C_s}\right)^D\times\exp\left(-\frac{E_s v+F_s}{B_s v+C_s}\lambda_{T_i^*}\right). \quad (16)
\end{aligned}$$

The last term of (16) in an exponential form can be rearranged and expressed in term of an infinite summation by the Taylor expansion as

$$\begin{aligned}
&\exp\left(-\frac{E_s v+F_s}{B_s v+C_s}\lambda_{T_i^*}\right)\\
=\;& \exp\left(-\lambda_{T_i^*}\frac{E_s}{B_s}\times\left[1-\left(1-\frac{F_sB_s}{E_sC_s}\right)\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right]\right) \qquad \frac{F_sB_s}{E_sC_s}\in(0,1)\\
=\;& e^{-\lambda_{T_i^*}\frac{E_s}{B_s}}\times\exp\left(\lambda_{T_i^*}\left(\frac{E_s}{B_s}-\frac{F_s}{C_s}\right)\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right)\\
=\;& e^{-\lambda_{T_i^*}\frac{E_s}{B_s}}\times\sum_{j=0}^{\infty}\frac{\left[\lambda_{T_i^*}\left(\frac{E_s}{B_s}-\frac{F_s}{C_s}\right)\right]^j}{j!}\left(\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right)^j\\
=\;& e^{-\lambda_{T_i^*}\frac{F_s}{C_s}}\times\sum_{j=0}^{\infty}e^{-\lambda_{T_i^*}\left(\frac{E_s}{B_s}-\frac{F_s}{C_s}\right)}\frac{\left[\lambda_{T_i^*}\left(\frac{E_s}{B_s}-\frac{F_s}{C_s}\right)\right]^j}{j!}\left(\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right)^j\\
=\;& e^{-\lambda_{T_i^*}\frac{F_s}{C_s}}\times\sum_{j=0}^{\infty}\Pr\{J_s^*=j\}\left(\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right)^j,
\end{aligned}$$

where J_s^* is a Poisson random variable with the given rate $\lambda_{T_i^*}\left(\frac{E_s}{B_s}-\frac{F_s}{C_s}\right)$, i.e.

$$\Pr\{J_s^*=j\} = e^{-\lambda_{T_i^*}\left(\frac{E_s}{B_s}-\frac{F_s}{C_s}\right)}\frac{\left[\lambda_{T_i^*}\left(\frac{E_s}{B_s}-\frac{F_s}{C_s}\right)\right]^j}{j!}, \qquad j=0,1,....$$

Then, we have

$$
\begin{aligned}
& \mathbb{E}\left[\lambda_{T_i^*+s^-} e^{-v\lambda_{T_i^*+s^-}} e^{-\left(\Lambda_{T_i^*+s}-\Lambda_{T_i^*}\right)}\right] \\
= & \frac{1}{B_s v + C_s}\left(DB_s + \lambda_{T_i^*}\frac{E_s C_s - F_s B_s}{B_s v + C_s}\right)\left(\frac{A_s}{B_s v + C_s}\right)^D \times e^{-\lambda_{T_i^*}\frac{F_s}{C_s}} \sum_{j=0}^{\infty} \Pr\{J_s^* = j\}\left(\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right)^j \\
= & \frac{\frac{1}{B_s}}{v+\frac{C_s}{B_s}}\left(DB_s + \lambda_{T_i^*}\frac{E_s\frac{C_s}{B_s} - F_s}{v+\frac{C_s}{B_s}}\right)\left(\frac{\frac{A_s}{B_s}}{v+\frac{C_s}{B_s}}\right)^D e^{-\lambda_{T_i^*}\frac{F_s}{C_s}} \times \sum_{j=0}^{\infty} \Pr\{J_s^* = j\}\left(\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right)^j \\
= & \frac{1}{C_s}\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\left[DB_s + \lambda_{T_i^*}\left(E_s - F_s\frac{B_s}{C_s}\right)\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right]\left(\frac{A_s}{C_s}\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right)^D e^{-\lambda_{T_i^*}\frac{F_s}{C_s}} \times \sum_{j=0}^{\infty} \Pr\{J_s^* = j\}\left(\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right)^j \\
= & G_s \times \left[w_{1s}\left(\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right) + w_{2s}\left(\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right)^2\right] \times \left(\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right)^D \times \sum_{j=0}^{\infty} \Pr\{J_s^* = j\}\left(\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right)^j \\
= & G_s \times \sum_{j=0}^{\infty} \Pr\{J_s^* = j\}\left[w_{1s}\left(\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right)^{j+D+1} + w_{2s}\left(\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right)^{j+D+2}\right],
\end{aligned}
$$

where

$$
G_s := \frac{e^{-\lambda_{T_i^*}\frac{F_s}{C_s}}}{C_s}\left(\frac{A_s}{C_s}\right)^D\left[DB_s + \lambda_{T_i^*}\left(E_s - F_s\frac{B_s}{C_s}\right)\right],
$$

and the weights w_{1s}, w_{2s} are given by (15). Obviously, we have $w_{1s} + w_{2s} = 1$, and $w_{1s}, w_{2s} > 0$.

Note that, $\left(\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right)^{j+D+1}$ and $\left(\frac{\frac{C_s}{B_s}}{v+\frac{C_s}{B_s}}\right)^{j+D+2}$ are the Laplace transforms of Gamma random variables Gamma $\left(j+D+1, \frac{C_s}{B_s}\right)$ and Gamma $\left(j+D+2, \frac{C_s}{B_s}\right)$, respectively. By inverting it with respect to v, we have the joint density of $\left(S_{i+1}^*, \lambda_{T_i^*+S_{i+1}^{*-}}\right)$,

$$
\begin{aligned}
& \Pr\left\{\lambda_{T_i^*+S_{i+1}^{*-}} \in \mathrm{d}\lambda, S_{i+1}^* \in \mathrm{d}s\right\}/(\mathrm{d}s\mathrm{d}\lambda) \\
= & G_s \times \sum_{j=0}^{\infty} \Pr\{J_s^* = j\}\left[w_{1s}\frac{\left(\frac{C_s}{B_s}\right)^{j+D+1}}{\Gamma(j+D+1)}\lambda^{j+D}e^{-\frac{C_s}{B_s}\lambda} + w_{2s}\frac{\left(\frac{C_s}{B_s}\right)^{j+D+2}}{\Gamma(j+D+2)}\lambda^{j+D+1}e^{-\frac{C_s}{B_s}\lambda}\right].
\end{aligned}
$$

Integrating out $\lambda \in [0, \infty)$, we obtain the marginal density of S_{i+1}^*,

$$
\Pr\left\{S_{i+1}^* \in \mathrm{d}s\right\}/\mathrm{d}s = G_s,
$$

which can be alternatively derived by setting $v = 0$ in (16). Note that, G_s is independent of v.

Given $S^*_{i+1} = s$, we have the conditional density of $\lambda_{T^*_i + S^{*-}_{i+1}}$,

$$
\begin{aligned}
& \Pr\left\{\lambda_{T^*_i+S^{*-}_{i+1}} \in \mathrm{d}\lambda \mid S^*_{i+1} = s\right\} / (\mathrm{d}\lambda) \\
= & \frac{\Pr\left\{\lambda_{T^*_i+S^{*-}_{i+1}} \in \mathrm{d}\lambda, S^*_{i+1} = s\right\} / (\mathrm{d}s\mathrm{d}\lambda)}{\Pr\{S^*_{i+1} \in \mathrm{d}s\}/(\mathrm{d}s)} \\
= & \sum_{j=0}^{\infty} \Pr\{J^*_s = j\} \left[w_{1s} \frac{\left(\frac{C_s}{B_s}\right)^{j+D+1}}{\Gamma(j+D+1)} \lambda^{j+D} e^{-\frac{C_s}{B_s}\lambda} + w_{2s} \frac{\left(\frac{C_s}{B_s}\right)^{j+D+2}}{\Gamma(j+D+2)} \lambda^{j+D+1} e^{-\frac{C_s}{B_s}\lambda} \right] \quad (17) \\
= & \mathbb{E}\left[w_{1s} \frac{\left(\frac{C_s}{B_s}\right)^{J^*_s+D+1}}{\Gamma(J^*_s+D+1)} \lambda^{J^*_s+D} e^{-\frac{C_s}{B_s}\lambda} + w_{2s} \frac{\left(\frac{C_s}{B_s}\right)^{J^*_s+D+2}}{\Gamma(J^*_s+D+2)} \lambda^{J^*_s+D+1} e^{-\frac{C_s}{B_s}\lambda} \right]. \quad (18)
\end{aligned}
$$

□

Gamma $\left(J^*_s + D + 1, \frac{C_s}{B_s}\right)$ in Theorem 3.5 is denoted for a Gamma random variable with the shape parameter $J^*_s + D + 1$ and rate parameter $\frac{C_s}{B_s}$ (or scale parameter $\frac{B_s}{C_s}$). Theorem 3.5 suggests that, $\lambda_{T^*_i+s^-}$ i.e. the intensity location at the next jump time after T^*_i, can be exactly simulated as a mixture of two Gamma random variables, conditional on the intensity level $\lambda_{T^*_i}$ and the realisation of interarrival time $S^*_{i+1} = s$ and a Poisson random variable $J^*_s = j$. Note that, although there is an infinite summation in (17), we do not introduce any truncation error.

3.4 Exact Simulation of Self-excited Jumps

Finally, the exact scheme for simulating self-exiting jumps with CIR intensity via induction is summarised as below in Algorithm 3.6.

Algorithm 3.6 (Exact Scheme). *Based on Theorem 3.3 and Theorem 3.5, we can exactly simulate* $\left(\lambda_{T^*_{i+1}}, T^*_{i+1}\right)$ *conditional on* $\left(\lambda_{T^*_i}, T^*_i\right)$ *via the following steps:*

1. *Simulate the* $(i+1)^{\text{th}}$ *interarrival time* $S^*_{i+1} = s$ *via (10).*
2. *Set the* $(i+1)^{\text{th}}$ *self-excited jump time* T^*_{i+1} *by*

$$T^*_{i+1} = T^*_i + s.$$

3. *Simulate a Poisson random variable* $J^*_s = j$ *of (14).*
4. *Conditional on* s *and* j, *simulate* $\lambda_{T^{*-}_{i+1}}$ *as a mixture of two Gamma random variables* Gamma $\left(j + D + 1, \frac{C_s}{B_s}\right)$ *and* Gamma $\left(j + D + 2, \frac{C_s}{B_s}\right)$ *with weights* w_{1s}, w_{2s} *of (15).*
5. *Add a self-excited jump in the intensity process at the jump time* T^*_{i+1} *by*

$$\lambda_{T^*_{i+1}} = \lambda_{T^{*-}_{i+1}} + Y_{i+1}, \quad (19)$$

where Y_{i+1} is the $(i+1)^{\text{th}}$ self-excited jump size.

6. *Add one unit in the point process at the jump time T^*_{i+1} by*

$$N_{T^*_{i+1}} = N_{T^{*-}_{i+1}} + 1.$$

It is interesting to see that, the original randomness we need to simulate throughout the whole procedure of sampling one path by Algorithm 3.6 only involves uniform, Poisson and Gamma random variables. The original randomness of normal distributions driven by the Brownian motion $\{W_t\}_{t\geq 0}$ in the intensity process apparently disappears.

Algorithm 3.6 can be numerically validated by the associated theoretical (true) results, for instance, the conditional expectation $\mathbb{E}[N_T \mid \lambda_0]$ provided in Proposition 2.1. For the demonstration purpose, we further assume that the self-excited jump sizes follow an exponential distribution with constant rate $\beta > 0$. We implement the simulation for four different parameter settings of

$$\Theta := (a, \lambda_0, \delta, \sigma, \beta),$$

including the unconventional cases when the stationary condition $\delta > \mu_{1_G} = 1/\beta$ for the Hawkes process or the conventional condition $2\delta a \geq \sigma^2$ for the CIR process is invalid, i.e. the Case II, Case III and Case IV, which are often hard to be generated by other algorithms[5] in the current existing literature:

Case I: $\Theta_I = (0.9, 0.9, 1.0, 1.0, 1.2)$ when the conventional conditions for the Hawkes process and CIR process both hold, i.e. $\delta > \mu_{1_G}$ and $2\delta a \geq \sigma^2$;

Case II: $\Theta_{II} = (0.9, 0.9, 1.0, 1.0, 0.9)$ when $\delta < \mu_{1_G}$ and $2\delta a \geq \sigma^2$;

Case III: $\Theta_{III} = (0.9, 0.9, 1.0, 1.0, 1.0)$ when $\delta = \mu_{1_G}$ and $2\delta a \geq \sigma^2$;

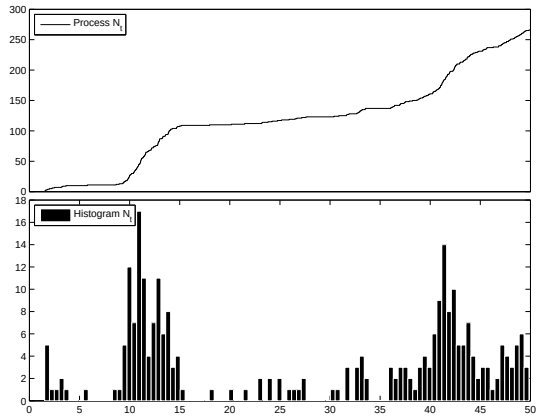
Case IV: $\Theta_{IV} = (0.9, 0.9, 1.0, 2.0, 1.2)$ when $\delta > \mu_{1_G}$ and $2\delta a < \sigma^2$, note that, the origin is accessible only if $2\delta a < \sigma^2$.

All four cases are based on the total number of simulated sample paths $n = 100,000$. The simulated results, the corresponding theoretical values, error percentages and standard

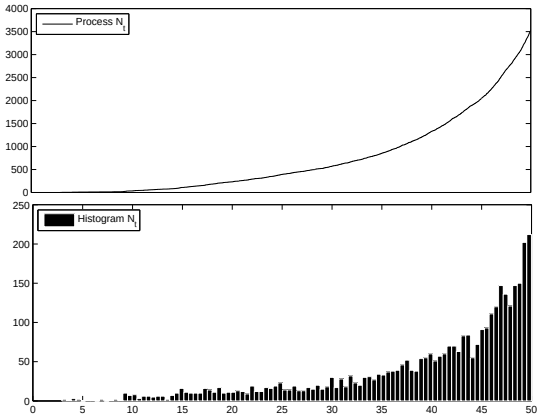
[5]To simulate this type of specified process, the condition $2\delta a \geq \sigma^2$ is usually required in other algorithms such as the classical discretisation scheme, inverse-CDF scheme of Giesecke and Kim (2007), projection scheme of Giesecke et al. (2011a), and A/R-sampling scheme of Giesecke et al. (2011b), and these schemes relies on this condition to guarantee that the intensity process would never touch zero level. The condition $\delta > \mu_{1_G}$ is required in other algorithms such as perfect scheme of Møller and Rasmussen (2005) and approximation scheme of Møller and Rasmussen (2006), this is to guarantee that the intensity process would never go explosive.

errors (SE)[6] for different time T are given by Table 2, where we can find that the simulation can achieve a high level of accuracy. One simulated sample path of point process $\left\{N_t\right\}_{0\leq t\leq 50}$ with the associated histogram for each case is represented in Figure 2, where the clustering jumps over the time horizon are evident. The convergence analysis of the standard errors against the total numbers of simulated sample paths as well as the computing time for the four cases are also provided in Figure 3 respectively with the detailed data reported in Table 3.

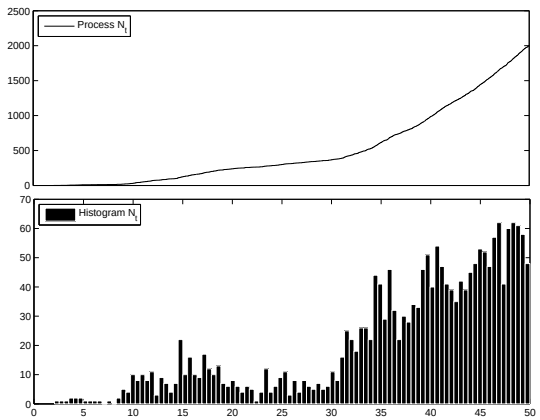
(a) **Case** I: $\delta > \mu_{1_G}, 2\delta a \geq \sigma^2$



(b) **Case** II: $\delta < \mu_{1_G}, 2\delta a \geq \sigma^2$



(c) **Case** III: $\delta = \mu_{1_G}, 2\delta a \geq \sigma^2$



(d) **Case** IV: $\delta > \mu_{1_G}, 2\delta a < \sigma^2$

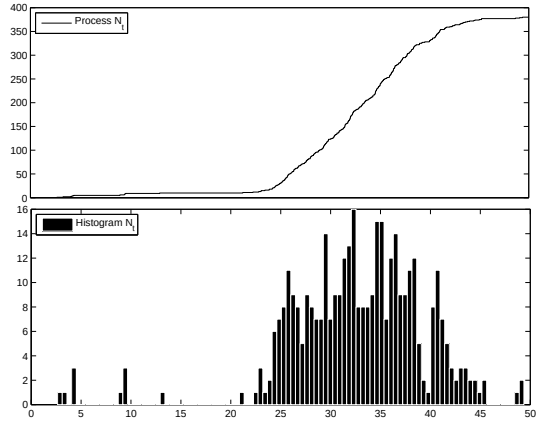


Figure 2: Sample paths of the point process $\left\{N_t\right\}_{0\leq t\leq 50}$ simulated by Algorithm 3.6 with the associated histograms for four sets of parameters $\Theta_I = (0.9, 0.9, 1.0, 1.0, 1.2)$, $\Theta_{II} = (0.9, 0.9, 1.0, 1.0, 0.9)$, $\Theta_{III} = (0.9, 0.9, 1.0, 1.0, 1.0)$, $\Theta_{IV} = (0.9, 0.9, 1.0, 2.0, 1.2)$, respectively

4 Comparisons with Other Important Algorithms

In this section, we conduct and report the comparisons of numerical performance with the classical discretisation scheme and the recently-developed projection scheme in the

[6]The standard error is estimated as the sample standard deviation of the simulation output divided by the square root of the total number of simulated sample paths.

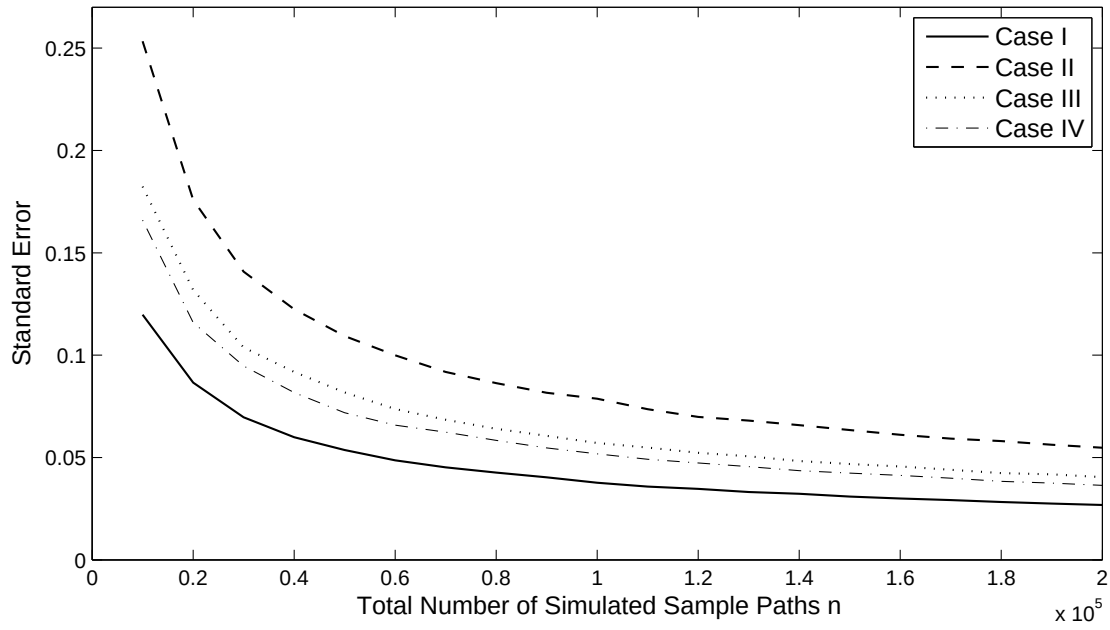
Table 2: Comparison between the theoretical formulas and the associated simulation results for $\mathbb{E}[N_T|\lambda_0]$ with four sets of parameters $\Theta_I = (0.9, 0.9, 1.0, 1.0, 1.2)$, $\Theta_{II} = (0.9, 0.9, 1.0, 1.0, 0.9)$, $\Theta_{III} = (0.9, 0.9, 1.0, 1.0, 1.0)$, $\Theta_{IV} = (0.9, 0.9, 1.0, 2.0, 1.2)$, respectively, based on the number of simulated sample paths $n = 100,000$ by Algorithm 3.6

Time T	$\mathbb{E}[N_T\|\lambda_0]$	Simulation	Error%	SE	Time (sec)	$\mathbb{E}[N_T\|\lambda_0]$	Simulation	Error%	SE	Time (sec)
	Case I					**Case** II				
2	3.1463	3.1506	0.14%	0.0117	67	3.9568	3.9578	0.03%	0.0165	87
4	8.4623	8.4705	0.10%	0.0283	156	12.9295	12.8922	-0.29%	0.0504	267
6	15.3327	15.3965	0.42%	0.0483	285	28.1665	28.2685	0.36%	0.1110	608
8	23.3171	23.3578	0.17%	0.0704	434	51.2265	51.4166	0.37%	0.2019	1,145
10	32.0996	32.1776	0.24%	0.0935	605	84.0563	84.2589	0.24%	0.3326	1,997
12	41.4541	41.4898	0.09%	0.1160	782	129.0871	128.8164	-0.21%	0.5078	3,158
14	51.2182	51.3052	0.17%	0.1403	977	189.3551	189.3875	0.02%	0.7580	4,739
16	61.2761	61.1809	-0.16%	0.1609	1,165	268.6522	267.5076	-0.43%	1.0712	7,106
18	71.5443	71.6200	0.11%	0.1835	1,360	371.7135	371.6251	-0.02%	1.5209	10,044
20	81.9632	81.5863	-0.46%	0.2037	1,609	504.4530	505.5166	0.21%	2.0635	13,643
	Case III					**Case** IV				
2	3.6000	3.5808	-0.53%	0.0143	77	3.1463	3.1317	-0.47%	0.0146	67
4	10.8000	10.8427	0.40%	0.0402	205	8.4623	8.4817	0.23%	0.0378	163
6	21.6000	21.5983	-0.01%	0.0775	425	15.3327	15.3445	0.08%	0.0667	280
8	36.0000	36.0496	0.14%	0.1282	727	23.3171	23.3663	0.21%	0.0983	429
10	54.0000	54.1481	0.27%	0.1898	1,159	32.0996	32.1231	0.07%	0.1323	595
12	75.6000	75.8490	0.33%	0.2650	1,584	41.4541	41.5570	0.25%	0.1645	776
14	100.8000	101.4370	0.63%	0.3534	2,147	51.2182	51.1153	-0.20%	0.1964	969
16	129.6000	129.8397	0.18%	0.4458	2,911	61.2761	61.2148	-0.10%	0.2310	1,165
18	162.0000	162.3701	0.23%	0.5606	3,624	71.5443	71.7533	0.29%	0.2641	1,375
20	198.0000	198.0550	0.03%	0.6812	4,535	81.9632	82.0178	0.07%	0.2927	1,578

Table 3: Convergence analysis on the simulation results for $\mathbb{E}[N_{T=5}|\lambda_0]$ with four sets of parameters $\Theta_I = (0.9, 0.9, 1.0, 1.0, 1.2)$, $\Theta_{II} = (0.9, 0.9, 1.0, 1.0, 0.9)$, $\Theta_{III} = (0.9, 0.9, 1.0, 1.0, 1.0)$, $\Theta_{IV} = (0.9, 0.9, 1.0, 2.0, 1.2)$, respectively, based on the number of simulated sample paths n by Algorithm 3.6

n	$\mathbb{E}[N_{T=5}\|\lambda_0]$	Simulation	Error%	SE	Time (sec)	$\mathbb{E}[N_{T=5}\|\lambda_0]$	Simulation	Error%	SE	Time (sec)
	Case I					**Case** II				
20,000	11.7342	11.9804	2.10%	0.0866	44	19.6756	19.8296	0.78%	0.1759	83
40,000	11.7342	11.6749	-0.51%	0.0599	88	19.6756	19.7735	0.50%	0.1224	168
60,000	11.7342	11.7532	0.16%	0.0486	131	19.6756	19.7067	0.16%	0.0999	245
80,000	11.7342	11.7472	0.11%	0.0426	172	19.6756	19.6267	-0.25%	0.0863	335
100,000	11.7342	11.6755	-0.50%	0.0377	214	19.6756	19.7216	0.23%	0.0787	416
120,000	11.7342	11.7846	0.43%	0.0347	258	19.6756	19.5723	-0.53%	0.0698	492
140,000	11.7342	11.8067	0.62%	0.0323	305	19.6756	19.6965	0.11%	0.0658	583
160,000	11.7342	11.7637	0.25%	0.0300	351	19.6756	19.6725	-0.02%	0.0611	664
180,000	11.7342	11.7495	0.13%	0.0283	392	19.6756	19.7323	0.29%	0.0580	739
200,000	11.7342	11.7465	0.11%	0.0268	435	19.6756	19.6417	-0.17%	0.0548	818
	Case III					**Case** IV				
20,000	15.7500	15.8829	0.84%	0.1318	66	11.7342	11.6952	-0.33%	0.1159	43
40,000	15.7500	15.9381	1.19%	0.0918	133	11.7342	11.6847	-0.42%	0.0817	86
60,000	15.7500	15.6563	-0.60%	0.0737	194	11.7342	11.6553	-0.67%	0.0658	130
80,000	15.7500	15.7119	-0.24%	0.0640	260	11.7342	11.8371	0.88%	0.0584	176
100,000	15.7500	15.6770	-0.46%	0.0571	322	11.7342	11.7331	-0.01%	0.0518	213
120,000	15.7500	15.7850	0.22%	0.0523	389	11.7342	11.7910	0.48%	0.0473	256
140,000	15.7500	15.7184	-0.20%	0.0483	449	11.7342	11.7546	0.17%	0.0436	298
160,000	15.7500	15.8258	0.48%	0.0456	518	11.7342	11.7620	0.24%	0.0413	341
180,000	15.7500	15.6714	-0.50%	0.0424	576	11.7342	11.6443	-0.77%	0.0384	380
200,000	15.7500	15.7022	-0.30%	0.0404	642	11.7342	11.6949	-0.33%	0.0364	426

(a) Convergence analysis: standard error v.s. number of simulated paths



(b) Convergence analysis: standard error v.s. simulation time

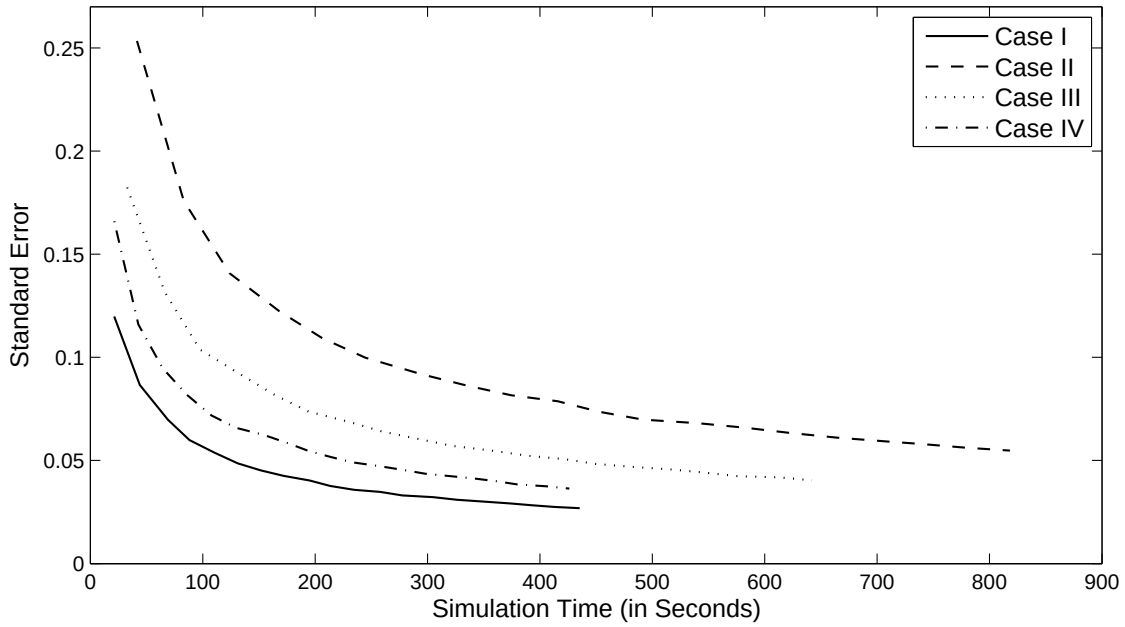


Figure 3: Convergence analysis of the standard error against the total number of simulated paths and the simulation time (in seconds) for $\mathbb{E}[N_{T=5}|\lambda_0]$ with four sets of parameters $\Theta_I = (0.9, 0.9, 1.0, 1.0, 1.2)$, $\Theta_{II} = (0.9, 0.9, 1.0, 1.0, 0.9)$, $\Theta_{III} = (0.9, 0.9, 1.0, 1.0, 1.0)$, $\Theta_{IV} = (0.9, 0.9, 1.0, 2.0, 1.2)$ by Algorithm 3.6, respectively

literature, respectively. To compare the performance among different algorithms, we adopt the conventional measure of *root mean square error* (RMSE)[7]. The numerical examples for our exact simulation in this section are implemented using Algorithm 3.6. Remind that, its efficiency can be further improved by the adjustment proposed by Remark 3.4.

4.1 Comparison with Discretisation Scheme

The classical *time-scaling* method for the discretisation scheme is based on the *change of time* developed by Meyer (1971). Given the previous intensity level $\lambda_{T_i^*}$ at the i^{th} jump arrival time T_i^*, the following interarrival intensity process $\{\lambda_t\}_{T_i^* \le t < T_{i+1}^*}$ (i.e. the intensity of interarrival time S_{i+1}^*) follows a CIR process (or Feller diffusion) with the SDE

$$\mathrm{d}\lambda_t = \delta(a - \lambda_t)\mathrm{d}t + \sigma\sqrt{\lambda_t}\mathrm{d}W_t, \qquad t \in [T_i^*, T_{i+1}^*)\,. \tag{20}$$

By the change of time (Meyer, 1971), the $(i+1)^{\text{th}}$ jump arrival time T_{i+1}^* is given by

$$T_{i+1}^* \overset{\mathcal{D}}{=} \inf\left\{t \ge T_i^* : \Lambda_t - \Lambda_{T_i^*} \ge \mathcal{E}_{i+1}\right\}, \tag{21}$$

where $\left\{\mathcal{E}_i\right\}_{i=1,2,\ldots}$ is a sequence of *i.i.d.* standard exponential random variables, i.e. $\mathcal{E}_i \sim \mathrm{Exp}(1)$. To numerically implement (21) for simulating T_{i+1}^*, the path of interarrival intensity process $\{\lambda_t\}_{T_i^* \le t < T_{i+1}^*}$ needs to be discretised. The procedures are summarised in Algorithm 4.1.

Algorithm 4.1 (Discretisation Scheme)**.** *Based on (21), we can approximately simulate* $\left(\lambda_{T_{i+1}^*}, T_{i+1}^*\right)$ *conditional on* $\left(\lambda_{T_i^*}, T_i^*\right)$ *via the following steps:*

1. *The continuous-time interarrival intensity process* λ_t *of (20) is approximated by* $\hat{\lambda}_t$ *via the Euler discretisation scheme*

$$\hat{\lambda}_{t_j} = \hat{\lambda}_{t_{j-1}} + \delta\left(a - \hat{\lambda}_{t_{j-1}}\right)\hbar + \sigma\sqrt{\hat{\lambda}_{t_{j-1}}}\sqrt{\hbar}\mathcal{N}_{j-1}, \qquad j = \frac{\hat{T}_i^*}{\hbar}, \frac{\hat{T}_i^*}{\hbar}+1, \ldots, \frac{\hat{T}_{i+1}^*}{\hbar}-1, \quad t_0 = 0, \quad \hat{\lambda}_0 = \lambda_0,$$

   with the approximated initial condition $\hat{\lambda}_{\hat{T}_i^*}$*, where*

   - $j = 1, 2, \ldots, \bar{J}$ *is the index of the time-discretisation grid;*
   - $\bar{J}$ *is the total number of grids within the time interval* $[0, T]$*;*
   - $\hbar = T/\bar{J}$ *is the length of each equally-spaced time grid, i.e.* $t_j = j\hbar$ *for any* j*;*

[7] RMSE is calculated by

$$\mathrm{RMSE} = \sqrt{\mathrm{SE}^2 + \mathrm{Bias}^2},$$

where the bias is the difference between the expectation of the estimator and the associated true (theoretical) value.

- $\{\mathcal{N}_j\}$ *is a sequence of i.i.d. standard normal random variables, i.e.* $\mathcal{N}_j \sim \mathcal{N}(0,1)$.

2. *The associated compensator* Λ_t *is approximated by* $\hat{\Lambda}_t$ *as*

$$\hat{\Lambda}_{t_j} = \hbar \sum_{k=1}^{j} \hat{\lambda}_{t_k}, \qquad \hat{\Lambda}_{t_0} = 0.$$

3. *The* $(i+1)^{\text{th}}$ *jump arrival time can be approximately simulated by*

$$\hat{T}^*_{i+1} = \inf\left\{j\hbar : \hat{\Lambda}_{t_j} - \hat{\Lambda}_{T^*_i} \geq \mathcal{E}_{i+1}\right\}.$$

4. *Add a self-excited jump in the discretised intensity process at the jump time* $\hat{T}^*_{i+1}$ *by*

$$\hat{\lambda}_{\hat{T}^*_{i+1}} = \hat{\lambda}_{\hat{T}^{*-}_{i+1}} + Y_{i+1}.$$

5. *Add one unit in the approximated point process at the jump time* $\hat{T}^*_{i+1}$ *by*

$$\hat{N}_{\hat{T}^*_{i+1}} = \hat{N}_{\hat{T}^{*-}_{i+1}} + 1.$$

This discretisation algorithm introduces bias, as $\mathbb{E}[\hat{N}_t] \neq \mathbb{E}[N_t]$ for any time $t > 0$ in practice. One sample path of intensity process $\left\{\hat{\lambda}_t\right\}_{t\in[0,10]}$ represented in Figure 1 is simulated via Algorithm 4.1 based on the parameter setting $\Theta_I = (0.9, 0.9, 1.0, 1.0, 1.2)$ of Case I and $\hbar = 0.001$. Based on the principle of optimal allocation of computation budget proposed by Duffie and Glynn (1995), in our numerical experiment, the number of time-discretisation grids is set equal to the square root of the number of sample paths, i.e. $\bar{J} = \sqrt{n}$ where n is the total number of sample paths. The numerical results of the comparison between the discretisation scheme and our exact scheme for the parameter setting Θ_I of Case I and $T = 1, 2, 5, 10$, respectively, are provided in Table 4. By conventionally following Giesecke et al. (2011b) and other relevant literature, the bias column for the exact scheme is set to be zero, and the bias column for the discretisation scheme is estimated based on a very large number of 10^8 sample paths. The associated true values are calculated based on the theoretical analytic formula in Proposition 2.1. The graphic comparison of convergence via the RMSE v.s. CPU time is plotted in Figure 4. It shows that our exact scheme produces much smaller RMSE for a given computational budget, in particular for a longer time horizon. Hence, our exact scheme obviously outperforms the discretisation scheme for all different time horizons in term of the convergence rate.

The discretisation scheme is easy to be implemented. However, comparing with our exact scheme of Algorithm 3.6, it has some obvious disadvantages:

Table 4: Numerical comparison between the discretisation scheme and our exact scheme, based on the parameter setting $\Theta_I = (0.9, 0.9, 1.0, 1.0, 1.2)$ of Case I for $T = 1, 2, 5, 10$, respectively

Scheme	Paths	Grids	$\mathbb{E}[N_t \mid \lambda_0]$	Simulation	Bias	SE	RMSE	Time (sec)
Discretisation	10,000	100	1.2550	1.2054	-0.0406	0.0160	0.0436	1.53
$T = 1$	40,000	200	1.2550	1.2441	-0.0207	0.0084	0.0223	7.22
	90,000	300	1.2550	1.2506	-0.0139	0.0056	0.0150	20.00
	160,000	400	1.2550	1.2389	-0.0105	0.0042	0.0113	40.67
	250,000	500	1.2550	1.2458	-0.0081	0.0034	0.0088	72.61
	360,000	600	1.2550	1.2466	-0.0071	0.0028	0.0077	119.31
	490,000	700	1.2550	1.2502	-0.0065	0.0024	0.0069	176.17
	640,000	800	1.2550	1.2464	-0.0054	0.0021	0.0058	252.89
	810,000	900	1.2550	1.2527	-0.0046	0.0019	0.0050	348.64
	1,000,000	1000	1.2550	1.2517	-0.0045	0.0017	0.0048	466.78
Exact	10,000		1.2550	1.2571	0	0.0171	0.0171	3.33
$T = 1$	40,000		1.2550	1.2404	0	0.0085	0.0085	13.16
	90,000		1.2550	1.2513	0	0.0057	0.0057	30.19
	160,000		1.2550	1.2546	0	0.0043	0.0043	55.03
	250,000		1.2550	1.2532	0	0.0034	0.0034	84.55
	360,000		1.2550	1.2560	0	0.0028	0.0028	120.91
	490,000		1.2550	1.2564	0	0.0025	0.0025	163.03
	640,000		1.2550	1.2532	0	0.0021	0.0021	212.33
	810,000		1.2550	1.2545	0	0.0019	0.0019	266.88
	1,000,000		1.2550	1.2541	0	0.0017	0.0017	329.84
Discretisation	10,000	100	3.1463	2.9253	-0.2302	0.0324	0.2325	1.67
$T = 2$	40,000	200	3.1463	3.0353	-0.1211	0.0172	0.1224	7.73
	90,000	300	3.1463	3.0629	-0.0820	0.0118	0.0828	20.38
	160,000	400	3.1463	3.0814	-0.0624	0.0089	0.0630	43.30
	250,000	500	3.1463	3.0954	-0.0497	0.0072	0.0502	75.42
	360,000	600	3.1463	3.1043	-0.0425	0.0060	0.0429	120.44
	490,000	700	3.1463	3.1128	-0.0368	0.0052	0.0372	187.41
	640,000	800	3.1463	3.1156	-0.0313	0.0045	0.0316	258.95
	810,000	900	3.1463	3.1104	-0.0282	0.0040	0.0285	356.53
	1,000,000	1000	3.1463	3.1195	-0.0262	0.0036	0.0265	480.84
Exact	10,000		3.1463	3.1224	0	0.0365	0.0365	6.36
$T = 2$	40,000		3.1463	3.1344	0	0.0185	0.0185	25.77
	90,000		3.1463	3.1455	0	0.0123	0.0123	58.48
	160,000		3.1463	3.1444	0	0.0092	0.0092	101.16
	250,000		3.1463	3.1434	0	0.0073	0.0073	156.47
	360,000		3.1463	3.1389	0	0.0061	0.0061	223.23
	490,000		3.1463	3.1412	0	0.0053	0.0053	304.09
	640,000		3.1463	3.1521	0	0.0046	0.0046	398.86
	810,000		3.1463	3.1488	0	0.0041	0.0041	504.31
	1,000,000		3.1463	3.1445	0	0.0037	0.0037	619.42
Discretisation	10,000	100	11.7342	8.8240	-2.8711	0.0720	2.8720	2.02
$T = 5$	40,000	200	11.7342	9.9319	-1.7365	0.0442	1.7370	9.36
	90,000	300	11.7342	10.4655	-1.2536	0.0322	1.2541	24.23
	160,000	400	11.7342	10.7748	-0.9839	0.0254	0.9842	49.41
	250,000	500	11.7342	10.9215	-0.8128	0.0208	0.8131	84.70
	360,000	600	11.7342	11.0273	-0.6902	0.0177	0.6904	136.00
	490,000	700	11.7342	11.1270	-0.5925	0.0154	0.5927	199.08
	640,000	800	11.7342	11.1940	-0.5292	0.0136	0.5294	284.23
	810,000	900	11.7342	11.2638	-0.4769	0.0123	0.4771	396.25
	1,000,000	1000	11.7342	11.3185	-0.4491	0.0111	0.4493	515.75
Exact	10,000		11.7342	11.5227	0	0.1170	0.1170	20.89
$T = 5$	40,000		11.7342	11.7919	0	0.0598	0.0598	89.34
	90,000		11.7342	11.6653	0	0.0397	0.0397	191.52
	160,000		11.7342	11.7320	0	0.0300	0.0300	336.38
	250,000		11.7342	11.7110	0	0.0239	0.0239	524.39
	360,000		11.7342	11.7225	0	0.0199	0.0199	766.48
	490,000		11.7342	11.7486	0	0.0171	0.0171	1,058.94
	640,000		11.7342	11.7252	0	0.0150	0.0150	1,356.11
	810,000		11.7342	11.7352	0	0.0133	0.0133	1,740.38
	1,000,000		11.7342	11.7493	0	0.0120	0.0120	2,119.53
Discretisation	10,000	100	32.0996	16.7565	-15.2934	0.0944	15.2937	2.58
$T = 10$	40,000	200	32.0996	21.0183	-10.9613	0.0680	10.9615	11.52
	90,000	300	32.0996	23.3115	-8.6655	0.0542	8.6657	29.70
	160,000	400	32.0996	24.9134	-7.2055	0.0455	7.2057	59.48
	250,000	500	32.0996	25.9162	-6.1867	0.0389	6.1868	104.30
	360,000	600	32.0996	26.6985	-5.4266	0.0341	5.4267	163.67
	490,000	700	32.0996	27.3107	-4.8393	0.0304	4.8394	238.89
	640,000	800	32.0996	27.7387	-4.3668	0.0275	4.3669	342.64
	810,000	900	32.0996	28.1089	-3.9821	0.0251	3.9822	461.17
	1,000,000	1000	32.0996	28.4272	-3.6703	0.0230	3.6704	630.36
Exact	10,000		32.0996	32.2119	0	0.2970	0.2970	62.36
$T = 10$	40,000		32.0996	32.1947	0	0.1472	0.1472	248.14
	90,000		32.0996	32.0534	0	0.0980	0.0980	548.92
	160,000		32.0996	32.1711	0	0.0739	0.0739	982.52
	250,000		32.0996	32.2147	0	0.0590	0.0590	1,551.08
	360,000		32.0996	32.1601	0	0.0491	0.0491	2,252.56
	490,000		32.0996	32.1256	0	0.0420	0.0420	3,039.61
	640,000		32.0996	32.0723	0	0.0367	0.0367	3,954.58
	810,000		32.0996	32.0977	0	0.0327	0.0327	4,955.09
	1,000,000		32.0996	32.0684	0	0.0294	0.0294	6,093.38

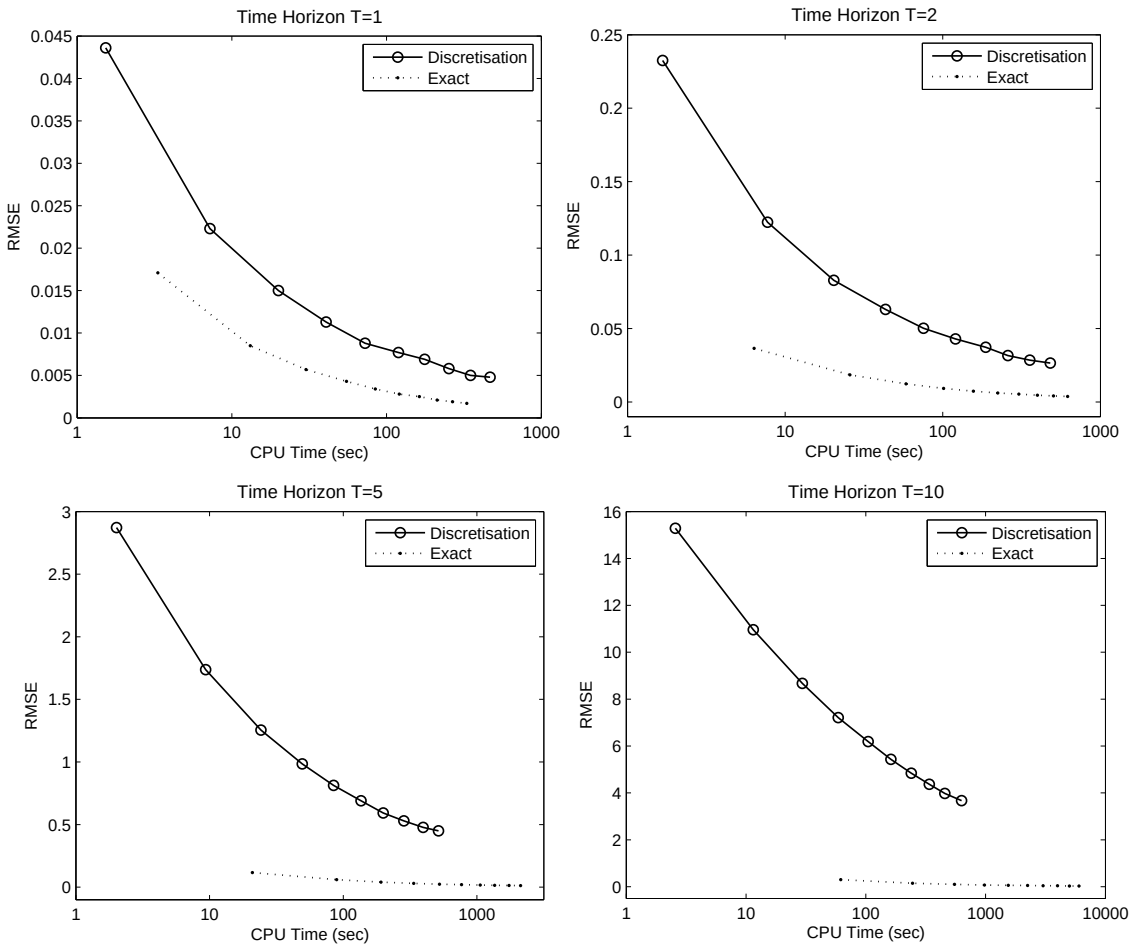


Figure 4: Graphical comparison of the convergence via the RMSE v.s. CPU time between the discretisation scheme and our exact scheme for the parameter setting $\Theta_I = (0.9, 0.9, 1.0, 1.0, 1.2)$ of Case I and $T = 1, 2, 5, 10$, respectively

1. Discretisation introduces bias which is hard to be quantified and measured.

2. The bias and errors are accumulating when time horizon T is increasing. This is evident from the plots in Figure 4: the RMSE of discretisation scheme becomes much larger when T increases from $T = 1$ to $T = 10$. It would be very time-consuming to achieve a high level of accuracy, especially for a large time T, as finer grids for time discretisation (i.e. smaller $\hbar$) are required. The accuracy of our algorithm does not much depends on T as observed from Table 2.

3. The Feller's condition $2\delta a \geq \sigma^2$ may be required by the discretisation method. The simulated discretised intensity process still has a small probability to be negative even when the Feller's condition holds. This is a well known problem and requires further adjustments.

4.2 Comparison with Projection Scheme

Giesecke et al. (2011a) proved that,

$$\Pr\left\{T_{i+1}^* > t \mid \mathcal{F}_{T_i^*}\right\} = \exp\left(-\int_{T_i^*}^{t} h_i(s)\mathrm{d}s\right), \qquad t \in [T_i^*, T_{i+1}^*), \qquad i = 0, 1, 2, ..., \quad (22)$$

where $h_i(t)$ is the projected i^{th} interarrival intensity function (or *projection*). The idea of the projection method is that, here $h_i(t)$ is a time-deterministic function, and if it can be computed exactly (i.e. without any numerical approximation, or numerically exact) for any i, then, in theory, the next arrival time T_{i+1}^* can be exactly simulated via the classical thinning scheme (Lewis and Shedler, 1979) subject to some upper-bound restriction, just like simulation for a nonhomogeneous Poisson process. Hence, the entire point process can be exactly simulated piecewisely by sequently implementing the thinning scheme. It is just like the *Ogata's modified thinning scheme* (Ogata, 1981) for exactly simulating the classical Hawkes process. However, for the numerical implementation in practice, the crucial problem is that function h_i may be difficult (or even impossible) to be computed exactly for any i, especially for a large value of i. Giesecke et al. (2011a) developed a recursive scheme for calculating h_i in theory as provided by Algorithm 4.2, see also Giesecke et al. (2008) for the version of a fixed initial intensity λ_0.

Algorithm 4.2 (Recursive Scheme)**.** *The projection h_i can be calculated recursively:*

1. *Initialisation: for $i = 0$, at time $T_0^* = 0$, compute $M_0(z) = e^{-z\lambda_0}, z \geq 0$.*

2. *For $i = 0, 1, 2, ...$, the i^{th} projection h_i can be computed by*

$$h_i(t) = -M_t'(z)\Big|_{z=0}, \qquad t \in [T_i^*, T_{i+1}^*), \qquad i = 0, 1, 2, ...,$$

where

$$M_t(z) := \frac{e^{a(t-T_i^*,z)} \times M_{T_i^*}\Big(b(t-T_i^*,z)\Big)}{e^{a(t-T_i^*,0)} \times M_{T_i^*}\Big(b(t-T_i^*,0)\Big)}, \qquad t \in [T_i^*, T_{i+1}^*), \qquad i = 0,1,2,..., \quad (23)$$

and

$$a(s,z) = D\ln\left(\frac{A_s}{B_s z + C_s}\right), \qquad b(s,z) = \frac{E_s z + F_s}{B_s z + C_s}.$$

3. *Given $h_i(t)$, the next arrival time T_{i+1}^* can be simulated according to (22) via thinning (see details later in Algorithm 4.3).*

4. *Recursion, at T_{i+1}^* compute*

$$M_{T_{i+1}^*}(z) = e^{-zY_{i+1}} \times \frac{M'_{T_{i+1}^{*-}}(z)}{M'_{T_{i+1}^{*-}}(z)\Big|_{z=0}}, \qquad t = T_{i+1}^*, \qquad i = 0,1,2,..., \qquad (24)$$

where

$$M_{T_{i+1}^{*-}}(z) = \lim_{t \uparrow T_{i+1}^*} M_t(z),$$

and $M_t(z)$ is specified by (23).

Here, $M_t(z)$ is the conditional Laplace transform of intensity λ_t, which does not directly have an analytic form. It can be calculated explicitly by (23) for the time period $[T_i^*, T_{i+1}^*)$ and updated at the jump time point T_{i+1}^* by (24). Hence, the entire function of $M_t(z)$ for the time horizon $[0,T]$ can be piecewisely obtained by the recursions starting with $M_0(z)$. Note that, the projected intensity h_i of our process is a decreasing function of time. The associated procedure of projection scheme is provided by Algorithm 4.3, which is a slightly simplified version of the original Algorithm 3.2 in Giesecke et al. (2011a).

Algorithm 4.3 (Projection Scheme)**.** *Given h_i by the recursive scheme of Algorithm 4.2, we can exactly simulate T_{i+1}^* conditional on T_i^* via thinning:*

1. *Initialise $t = T_i^*$.*

2. *Set the upper bound B_t^i adaptively by $B_t^i = h_i(t)$.*

3. *Generate candidate arrival time $\tilde{t} = t + \mathcal{E}$ where $\mathcal{E} \sim \text{Exp}\left(B_t^i\right)$, and draw $U \sim \mathcal{U}[0,1]$.*

   - *If $UB_t^i \le h_i(\tilde{t})$, then the $(i+1)^{\text{th}}$ jump occurs at time point $\tilde{t}$, so accept the candidate, set $T_{i+1}^* = \tilde{t}$.*
   - *If $UB_t^i > h_i(\tilde{t})$, then no jump occurs within $(t,\tilde{t}]$, so reject the candidate, set $t = \tilde{t}$, go back to Step 2 and continue.*

Table 5: Numerical comparison between the projection scheme and our exact scheme for estimating $\Pr\{J_T \leq 1 \mid \lambda_0\}$, based on the parameter setting $(a, \lambda_0, \delta, \sigma) = (1, 1, 1, 1)$ and uniformly distributed jump sizes $Y_i \sim \mathcal{U}\{0.4, 0.8\}$ for $T = 1, 2, 3, 4$, respectively

			Projection			**Exact**		
Time T	Paths	$\Pr\{J_T \leq 1 \mid \lambda_0\}$	Simulation	RMSE	CPU Time (sec)	Simulation	RMSE	CPU Time (sec)
$T = 1$	5,000	0.71490	0.71500	0.00639	6.25	0.70760	0.00643	3.81
	10,000	0.71490	0.71390	0.00452	12.77	0.71770	0.00450	6.73
	50,000	0.71490	0.71910	0.00201	61.81	0.71640	0.00202	33.91
	100,000	0.71490	0.71740	0.00142	127.69	0.71462	0.00143	67.27
$T = 2$	5,000	0.42821	0.43740	0.00702	17.63	0.42360	0.00699	4.58
	10,000	0.42821	0.43020	0.00495	35.20	0.42930	0.00495	9.31
	50,000	0.42821	0.43156	0.00222	176.81	0.43048	0.00221	47.22
	100,000	0.42821	0.43076	0.00157	355.81	0.42869	0.00156	92.25
$T = 3$	5,000	0.25280	0.24420	0.00608	28.05	0.24280	0.00606	5.94
	10,000	0.25280	0.25280	0.00435	55.84	0.25720	0.00437	12.31
	50,000	0.25280	0.25214	0.00194	279.63	0.24984	0.00194	59.63
	100,000	0.25280	0.25284	0.00137	561.84	0.25408	0.00138	122.88
$T = 4$	5,000	0.14670	0.15160	0.00507	35.47	0.14320	0.00495	8.14
	10,000	0.14670	0.15040	0.00357	70.14	0.14530	0.00352	15.42
	50,000	0.14670	0.14342	0.00157	348.39	0.14478	0.00157	78.61
	100,000	0.14670	0.14504	0.00111	708.81	0.14451	0.00111	154.58

For the numerical comparison between the projection scheme and our exact scheme, the targeted estimation is set to be $\Pr\{J_T \leq 1 \mid \lambda_0\}$. It is a simplified version used in the original numerical experiment of Giesecke et al. (2011a), as we attempt to keep the targeted estimation simple, without introducing additional complexity irrelevant to the algorithm comparison itself. Let us start with the same setup as Giesecke et al. (2011a):

- The jump sizes Y_i are assumed to follow a uniform distribution over two discrete points $\{0.4, 0.8\}$, i.e. $\Pr\{Y_i = 0.4\} = \Pr\{Y_i = 0.8\} = 1/2$ for any jump index i, which is denoted as $Y_i \sim \mathcal{U}\{0.4, 0.8\}$;
- Parameters are set as $(a, \lambda_0, \delta, \sigma) = (1.0, 1.0, 1.0, 1.0)$.

The results of numerical comparison are reported in Table 5, with the associated graphical comparison in Figure 5. As both of the two schemes are exact, the biases are set to be zero. Each of the true value $\Pr\{J_T \leq 1 \mid \lambda_0\}$ in the second column of Table 5 is estimated by a very large number of 10^6 sample paths based on our exact algorithm[8]. We can observe that, our exact scheme achieves a similar level of accuracy (as measured by RMSE) but a much faster computing speed than the projection scheme.

More importantly, our exact scheme is substantially better at exactly simulating sample paths with a larger number of jumps. Here, to estimate $\Pr\{J_T \leq 1 \mid \lambda_0\}$, the maxi-

[8]Note that, $\Pr\{J_T \leq 1 \mid \lambda_0\}$ does not have an analytic form to be calculated exactly, however, the true value of $\mathbb{E}[N_T \mid \lambda_0]$ can be calculated exactly by Proposition 2.1, since we have $\mu_{1_G} = (0.4 + 0.8)/2 = 0.6$ for this case. Then, the associated error% for $T = 1, 2, 3, 4$ can be reported as -0.1815%, -0.1142%, -0.0011%, 0.0004%, respectively. This benchmark makes sure that the associated estimations for $\Pr\{J_T \leq 1 \mid \lambda_0\}$ in the second column are accurate enough.

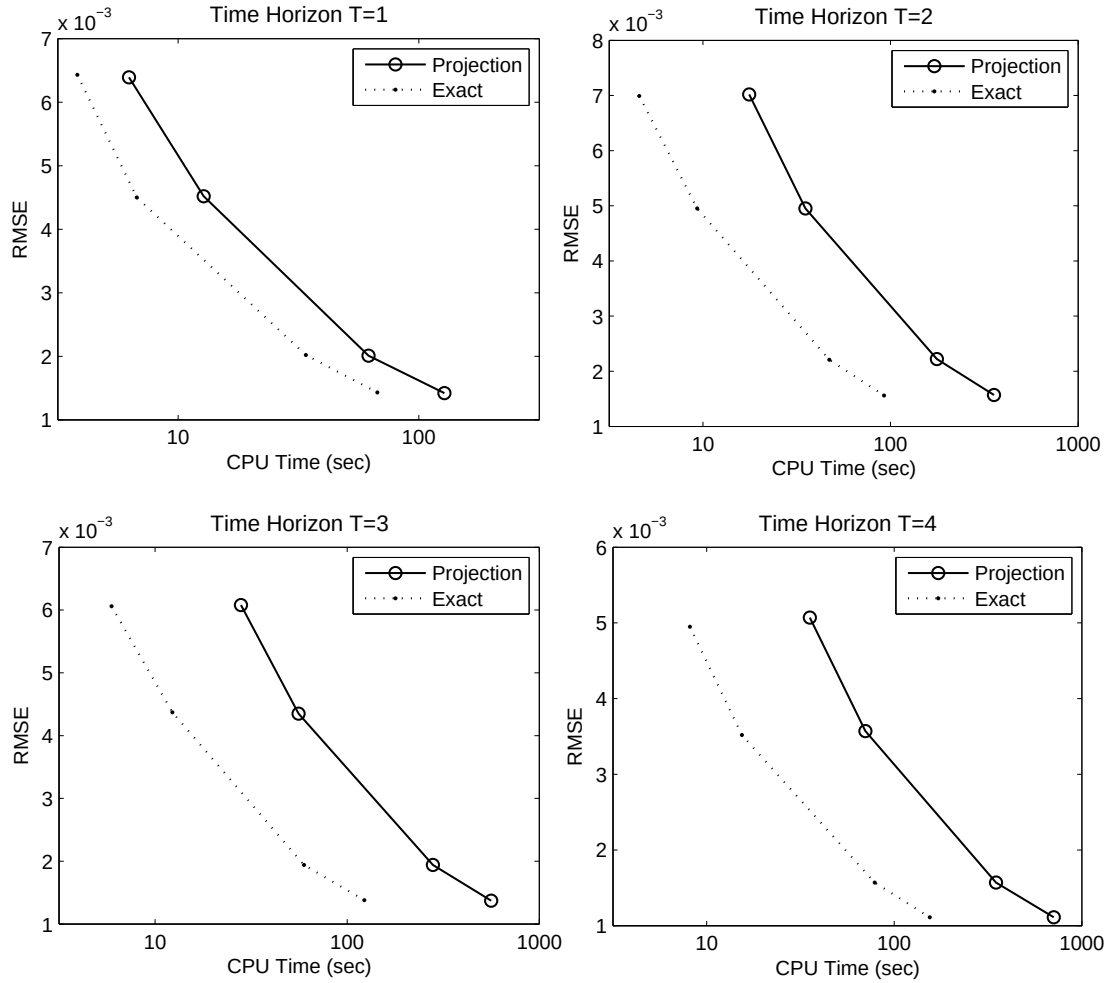


Figure 5: Graphical comparison of convergence via the RMSE v.s. CPU time between the projection scheme and our exact scheme for estimating $\Pr\{J_T \leq 1\}$, based on the parameter setting $(a, \lambda_0, \delta, \sigma) = (1, 1, 1, 1)$ and uniformly distributed jump sizes $Y_i \sim \mathcal{U}\{0.4, 0.8\}$ for $T = 1, 2, 3, 4$, respectively

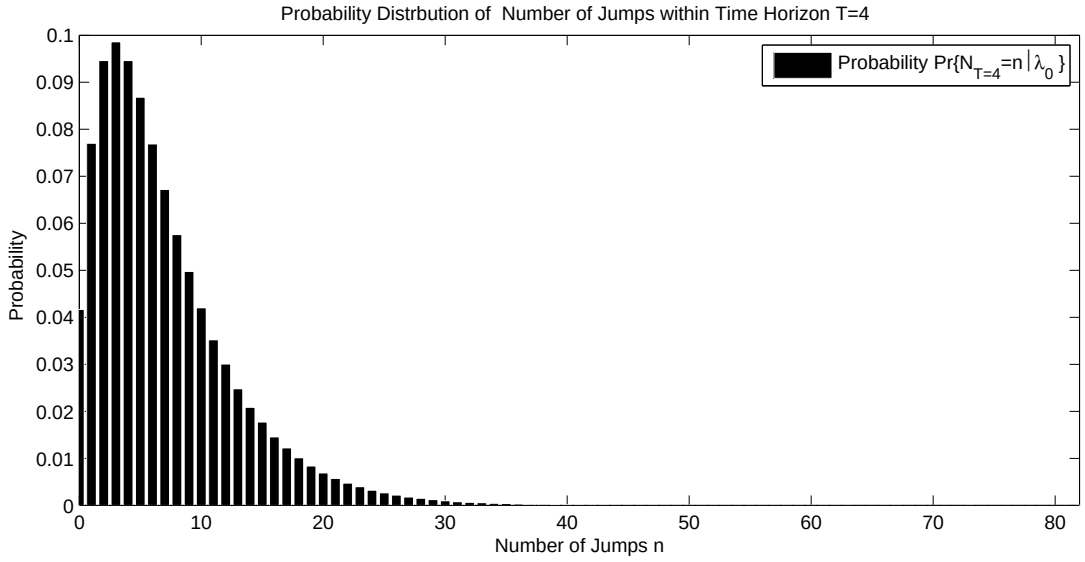


Figure 6: The estimated probability distribution for the number of jumps within the time horizon $T = 4$, based on 1,000,000 sample paths by Algorithm 3.6 and the parameter setting $(a, \lambda_0, \delta, \sigma) = (1, 1, 1, 1)$ and uniformly distributed jump sizes $Y_i \sim \mathcal{U}\{0.4, 0.8\}$

mum number of jumps needed to generate within each sample path is only two[9]. However, if one wants exactly simulate sample paths containing many jumps, the projection scheme would become very slow. This problem was also pointed by Giesecke et al. (2011a). It is due to the intrinsic recursive execution of differentiation in (23) of Step 2 and (24) of Step 4 within Algorithm 4.2. In fact, to accurately calculate the projection h_i in a computer when i is large for maintaining the algorithm still *exact*, these recursions would cumulate a huge number of analytic terms in the function $M_t(z)$ and function $M_{T_{i+1}^*}(z)$. This would be both very time-consuming and memory-consuming, and might be even hard to be handled by a normal computer. Approximations may be required to deal with this problem, however, the resulting scheme would not be *exact* anymore. Indeed, there is no such issue for our scheme. For instance, it is straightforward to obtain the entire distribution of the jump number N_T, i.e. $\Pr\{N_T = n \mid \lambda_0\}$, which has numerous applications in finance (e.g. portfolio risk management and asset pricing). For example, the estimated probability distribution for the case $T = 4$ based on 10^6 sample paths is provided in Figure 6, with the total number of jumps $N_{T=4}$ ranging from minimum 0 to maximum 82.

In summary, comparing with the projection scheme, our exact scheme of Algorithm 3.6 has some advantages:

1. In practice, the projection scheme is hard to maintain truly exact for simulating s-

[9]This is because, the jump sizes Y_i only can take two possible values, 0.4 and 0.8 here, for estimating $\Pr\{J_T \leq 1\} = \mathbb{E}\left[\mathbb{1}\left\{\sum_{i=1}^{N_T} Y_i \leq 1\right\}\right]$, only paths having 0, 1 and 2 jumps are needed to be simulated to save time.

cenarios where large numbers of jumps occur within a given time horizon $[0, T]$. These scenarios are quite common in the real world, and often important to be modelled accurately in practice, in particular for the cluttering arrivals of many events. For example, this point process may be well equipped for modelling arrivals of a large number of trades in a high-frequency trading environment. Since all of the interarrival times and intensity levels are sequentially decomposed into simple random variables (of only uniform, Poisson and Gamma) to generate, therefore the overall computing speed of our exact simulation does not much depend on the level of jump number i.

2. The Feller's condition $2\delta a \geq \sigma^2$ may be required by the projection method.

4.3 Summary for Numerical Comparisons

We admit that the discretisation scheme and projection scheme could be applicable to a more general family of point processes than our new exact scheme. However, for such an important process, the efficiency of our algorithm proposed in this paper exceeds that of these existing methods in the literature: our scheme is both theoretically and numerically *exact*, and also very fast. In fact, our approach is not only restricted to this specified process, we provide some important and useful extensions of our exact scheme in the next section.

5 Extensions

Our Algorithm 3.6 can be easily adjusted to exactly and efficiently simulate a broad family of self-exiting jumps (points) with CIR-type intensities. Some important extensions are listed as follows:

1. The self-excited jump sizes $\{Y_i\}_{i=1,2,...}$ in the intensity process (19) are flexible to be either fixed or following any arbitrary distribution, and they are not restricted to be positive as long as the zero lower bound of the intensity is not overshot, i.e. $Y_i \in [-\lambda_{T_i^*}, \infty), i = 1, 2, ...$

2. In particular, if $Y_{i+1} \equiv 0$ for any i in (19), then, this recovers the important special case of a point process with pure CIR intensity.

3. It works for a more general class of processes where the jumps can be anything not just additive: for example, at any jump time T_i^*, the jump in the intensity process from a level $\lambda_{T_{i-}^*}$ to $\lambda_{T_i^*}$ can follow a very general conditional CDF, say,

$$G(y \mid v) := \Pr\left\{\lambda_{T_i^*} \leq y \mid \lambda_{T_i^{*-}} = v\right\}, \qquad v \geq 0.$$

The simulation for this general case might be even more useful to generate different features of self excitements or contagion effects, as the nonlinear structure of the process makes a theoretical treatment very difficult.

4. It can also be adjusted to simulate self-exciting jumps with stationary CIR intensity. For instance, if jump sizes follow an exponential distribution, say, $Y_i \sim \text{Exp}(\beta), \beta > 0$, and the stationary condition holds, i.e. $\delta\beta > 1$, then, we can implement the simulation by setting the distribution of the initial intensity as

$$\lambda_0 \sim \text{Gamma}\left(\frac{2a\delta}{\sigma^2}b_1, -c_-\right) + \text{Gamma}\left(\frac{2a\delta}{\sigma^2}b_2, -c_+\right),$$

where the constants $b_1, b_2 > 0$ and $c_-, c_+ < 0$ are given by

$$b_1 = \frac{c_- + \beta}{c_- - c_+}, \qquad b_2 = -\frac{c_+ + \beta}{c_- - c_+}, \qquad c_\pm = \frac{-\left(\frac{2\delta}{\sigma^2} + \beta\right) \pm \sqrt{\left(\frac{2\delta}{\sigma^2} - \beta\right)^2 + \frac{8}{\sigma^2}}}{2}.$$

The associated proof is provided by Zhao (2012) and Dassios and Zhao (2017).

5. It is straightforward to integrate an additional series of externally-excited jumps in the intensity process to Algorithm 3.6, which may be very useful for modelling some external risk factors, see some similar models in Ogata and Akaike (1982), Brémaud and Massoulié (2002) and Dassios and Zhao (2011). For example, if a series of Poisson shot-noise jumps (Dassios and Jang, 2003) are added in the intensity process, then, the conditional intensity (1) is extended to be

$$\lambda_t = a + (\lambda_0 - a)\, e^{-\delta t} + \sigma \int_0^t e^{-\delta(t-s)} \sqrt{\lambda_s} \mathrm{d}W_s + \sum_{0 \le T_i^* < t} Y_i e^{-\delta(t - T_i^*)} + \sum_{0 \le \tau_k^* < t} X_k e^{-\delta(t - \tau_k^*)}, \quad t \ge 0, \tag{25}$$

where

- $\{\tau_k\}_{k=1,2,\ldots}$ are the arrival times of a Poisson process M_t of constant rate $\varrho > 0$;
- $\{X_k\}_{k=1,2,\ldots}$ are the sizes of externally-excited jumps.

A point process with this generalised intensity (25) can be exactly sampled by Algorithm 5.1 as follows.

Algorithm 5.1 (Exact Scheme)**.** *Conditional on (λ_{T_n}, T_n) where T_n is the n^{th} jump time in the intensity (which is either a self-excited jump or an externally-excited jump), we can simulate $\left(\lambda_{T_{n+1}}, T_{n+1}\right)$ via the following steps:*

(a) Simulate the $(n+1)^{\text{th}}$ interarrival time

$$s = \min\left\{S_{n+1}^*, E_{n+1}^*\right\},$$

where the $(n+1)^{\text{th}}$ *self-excited interarrival time* S^*_{n+1} *is simulated via (10); and* E^*_{n+1} *is the* $(n+1)^{\text{th}}$ *externally-excited interarrival time following an exponential distribution of rate* ϱ*, i.e.* $E^*_{n+1} \sim \text{Exp}(\varrho)$ *which can be simulated via*

$$E^*_{n+1} \overset{\mathcal{D}}{=} -\frac{1}{\varrho}\ln U, \qquad U \sim \mathcal{U}[0,1].$$

(b) Set the $(n+1)^{\text{th}}$ *jump time* T_{n+1} *by*

$$T_{n+1} = T_n + s.$$

(c) Simulate a Poisson random variable $J^*_s = j$ *as (14).*

(d) Conditional on s and j, simulate $\lambda_{T^{*-}_{n+1}}$ *as a mixture of two Gamma random variables* $\text{Gamma}\left(j+D+1, \frac{C_s}{B_s}\right)$ *and* $\text{Gamma}\left(j+D+2, \frac{C_s}{B_s}\right)$ *with weights* w_{1s}, w_{2s} *as (15).*

(e) Add a jump in the intensity process at the jump time by

$$\lambda_{T_{n+1}} = \begin{cases} \lambda_{T^-_{n+1}} + Y_{n+1}, & \text{if } \min\{S^*_{n+1}, E^*_{n+1}\} = S^*_{n+1}, \\ \lambda_{T^-_{n+1}} + X_{n+1}, & \text{if } \min\{S^*_{n+1}, E^*_{n+1}\} = E^*_{n+1}. \end{cases}$$

(f) Change the value in the point process at the jump time by

$$N_{T_{n+1}} = \begin{cases} N_{T^-_{n+1}} + 1, & \text{if } \min\{S^*_{n+1}, E^*_{n+1}\} = S^*_{n+1}, \\ N_{T^-_{n+1}}, & \text{if } \min\{S^*_{n+1}, E^*_{n+1}\} = E^*_{n+1}. \end{cases}$$

Inspired by the work of Dassios et al. (2015), Algorithm 5.1 may be further extended to the version, where the additional externally excited jumps arrive as a general renewal process rather than a simple Poisson process M_t. Then, the interarrival times of external shocks could follow any distribution rather than the exponential one.

6. Our exact algorithm is also flexible to be generalised to a multi-dimensional framework incorporating self-excited and mutually -excited jumps: A $\bar{D}-$dimensional point process $\left\{N^{[\jmath]}_t\right\}_{\jmath=1,2,...,\bar{D}}$ where $N^{[\jmath]}_t \equiv \left\{T^{[\jmath]*}_i\right\}_{i=1,2,...}$ can be constructed via the $\jmath^{\text{th}}$ intensity process for any $\jmath \in \{1,2,...,\bar{D}\}$ as

$$\lambda^{[\jmath]}_t = a_\jmath + \left(\lambda^{[\jmath]}_0 - a_\jmath\right)e^{-\delta_\jmath t} + \sigma_\jmath \int_0^t e^{-\delta_\jmath(t-s)}\sqrt{\lambda^{[\jmath]}_s}\mathrm{d}W^{[\jmath]}_s + \sum_{\ell=1}^{\bar{D}} \sum_{0\le T^{[\ell]*}_i < t} Y^{[\jmath,\ell]}_i e^{-\delta_\jmath(t-T^{[\ell]*}_i)},$$

where $\left\{Y^{[\jmath,\ell]}_i\right\}_{\jmath=\ell}$ are the sizes of self-excited jumps, and $\left\{Y^{[\jmath,\ell]}_i\right\}_{\jmath\neq\ell}$ are the sizes of *mutually-excited* (or *cross-excited*) jumps. Upon the arrival of one jump in point process, say, $N^{[\ell]}_t$, each marginal intensity process $\left\{\lambda^{[\jmath]}_t\right\}_{\jmath=1,2,...,\bar{D}}$ experiences a simul-

taneous *co-jump* of any nonnegative size. These co-jump sizes are free to be either mutually independent or dependent, so one can freely structure any dependency (such as correlation or copula models) for the co-jump sizes. Brownian motions $\left\{W_t^{[\jmath]}\right\}_{\jmath=1,2,...,\bar{D}}$ are assumed to be mutually independent. The ordered arrival times of co-jumps in the intensity processes are denoted by $\{T_k^*\}_{k=1,2,...}$. By extending our exact scheme of Algorithm 3.6, it is straightforward to exactly simulate the joint paths of $\left\{N_t^{[\jmath]}\right\}_{\jmath=1,2,...,\bar{D}}$ with any parameter setting $\Theta_\jmath := \left(a_\jmath, \delta_\jmath, \sigma_\jmath\right)$ piecewisely by Algorithm 5.2.

Algorithm 5.2 (Exact Scheme)**.** *For each $\jmath \in \{1, 2, ..., \bar{D}\}$, we can exactly simulate $\left(\lambda_{T_{k+1}^*}^{[\jmath]}, T_{k+1}^*\right)$ conditional on $\left(\lambda_{T_k^*}^{[\jmath]}, T_k^*\right)$ via the following steps:*

(a) Simulate the $(k+1)^{\text{th}}$ interarrival time of co-jumps in intensity processes by

$$\omega = \min\left\{S_{k+1}^{[1]*}, S_{k+1}^{[2]*}, ..., S_{k+1}^{[\bar{D}]*}\right\},$$

where each candidate $S_{k+1}^{[\jmath]*}$ can exactly simulated in the same way as S_{i+1}^* via (10) of Theorem 3.3 by simply replacing the index i by k and the parameter setting Θ by $\Theta_\jmath$ for each $\jmath$. Say, it is the ℓ^{th} point process that experiences a jump and triggers co-jumps in all intensities, i.e.

$$\omega = S_{k+1}^{[\ell]*}.$$

(b) Record the $(k+1)^{\text{th}}$ co-jump arrival time T_{k+1}^* in each intensity $\lambda_t^{[\jmath]}$ by

$$T_{k+1}^* = T_k^* + \omega.$$

(c) For each $\jmath$, simulate a Poisson random variable $J_\omega^* = j$ of (14) using the associated parameter setting $\Theta_\jmath$ instead.

(d) For each $\jmath$, conditional on ω and j, simulate $\lambda_{T_{k+1}^{*-}}^{[\jmath]}$ as a mixture of two Gamma random variables $\text{Gamma}\left(j+D+1, \frac{C_\omega}{B_\omega}\right)$ and $\text{Gamma}\left(j+D+2, \frac{C_\omega}{B_\omega}\right)$ with weights $w_{1\omega}, w_{2\omega}$ of (15) using the associated parameter setting $\Theta_\jmath$ instead.

(e) Record the change of each intensity process $\lambda_t^{[\jmath]}$ at the co-jump time T_{k+1}^* by

$$\lambda_{T_{k+1}^*}^{[\jmath]} = \lambda_{T_{k+1}^{*-}}^{[\jmath]} + Y_{k+1}^{[\jmath,\ell]}, \qquad \jmath \in \{1, 2, ..., \bar{D}\}.$$

(f) Record the change of each point process $N_t^{[\jmath]}$ at the co-jump time T_{k+1}^* by

$$N_{T_{k+1}^{*+}}^{[\jmath]} = \begin{cases} N_{T_{k+1}^{*-}}^{[\jmath]} + 1, & \jmath = \ell, \\ N_{T_{k+1}^{*-}}^{[\jmath]}, & \jmath \neq \ell, \end{cases} \qquad \jmath \in \{1, 2, ..., \bar{D}\}.$$

If Brownian motions $\left\{W_t^{[j]}\right\}_{j=1,2,\ldots,\bar{D}}$ are dependent, then, the joint process of inter-arrival intensities between two successive co-jumps generally becomes a general Wishart process (i.e. the multi-dimensional version of CIR processes) (Bru, 1991). This case is rather complicated, as the fundamental result for the integral transform in Proposition 3.1 should be completely re-derived, and it could be a matter for future research.

6 Applications to Finance: Portfolio Loss Distribution

Efficient numerical algorithms for simulating portfolio loss processes are extensively discussed in the literature, see Glasserman et al. (2005) and Giesecke and Kim (2011). Now, we make a simple application of our Algorithm 3.6 to estimate the portfolio loss distribution. Suppose that, we have a portfolio of investments, and the arrival of loss is modelled by a self-exciting point process $N_t \equiv \left\{T_i^*\right\}_{i=1,2,\ldots}$ with the stochastic CIR intensity λ_t as defined by Definition 2.1. The cumulative portfolio loss process at time t is

$$L_t = \sum_{i \geq 1} L_i \mathbb{1}\{T_i^* \leq t\},$$

where L_i is the individual loss (e.g. the loss-given-default) that occurs at time T_i^*.

As previously discussed, the choice for the sizes of self-excited jumps $\{Y_i\}_{i=1,2,\ldots}$ could be very flexible. To illustrate a simple implementation, here we follow Errais et al. (2010) by setting $Y_i = bL_i$, where $b > 0$ is a constant multiplier governing the sensitivity to defaults, and L_i could be assumed to be either random or constant. The loss process then has self-exciting effects, and the loss frequency has the desirable mean-reverting property: when each individual loss occurs, the intensity of loss arrival jumps up by a magnitude proportional to this realised individual loss; after the loss, the intensity of loss arrival tends to diffusively revert back to its long-term level a at a constant rate δ, since the economy is assumed to have its capability to recover to a normal state from crisis eventually. The evidence for this mean-reverting property in the default intensity can be found in Duffie et al. (2009). Comparing to the original Hawkes point process, the additional component of diffusion $\{\sigma W_t\}_{t \geq 0}$ in the portfolio intensity (1) in our model could be able to capture a certain degree of noises or factors, which are consistently existing and are making fluctuations in markets. This model could also capture the well known credit risk phenomena of the negative dependence between defaults and recovery rates, see empirical evidences in Altman et al. (2005). Overall, this provides a realistic model, particularly for the loss due to default, and can be applied to portfolio risk management or pricing multi-name credit derivatives, see more arguments in Giesecke et al. (2011b).

For a simple numerical implementation for our exact simulation via Algorithm 3.6, we further assume individual losses follow exponential distribution, say, $L_i \sim \text{Exp}(\beta)$ with the parameter setting $(a, \lambda_0, \delta, \sigma, \beta, b) = (0.9, 0.9, 1.0, 1.0, 1.2, 1.0)$. One simulated sample path of the joint point process N_t and loss process L_t for the time period $t \in [0, 100]$ is plotted in Figure 7. We can observe that the histogram of N_t reproduces the empirically observed clustering losses in the time horizon. We can also generally compute various quantities for the arrival process $\{N_t\}_{t\geq 0}$ and the loss process $\{L_t\}_{t\geq 0}$, such as the CDF of the cumulative loss process at time T, $\Pr\{L_T \leq l\}$, and call options on the portfolio loss, $\mathbb{E}[(L_T - K)^+]$ where $K > 0$ is the strike price. Here, we take the loss distribution $\Pr\{L_T \leq l\}$ for instance. We assume that, the individual losses are fixed, or, follow a standard uniform distribution or an exponential distribution:

Case 1 (Constant): $L_i \equiv 0.5$ with $(a, \lambda_0, \delta, \sigma, b) = (0.9, 0.9, 1.0, 1.0, 1.0)$;

Case 2 (Uniform Distribution): $L_i \sim \mathcal{U}[0, 1]$ with $(a, \lambda_0, \delta, \sigma, b) = (0.9, 0.9, 1.0, 1.0)$;

Case 3 (Exponential Distribution): $L_i \sim \text{Exp}(\beta)$ with $(a, \lambda_0, \delta, \sigma, \beta, b) = (0.9, 0.9, 1.0, 1.0, 2.0, 1.0)$.

Estimated probabilities for $\Pr\{L_T \leq l\}$ based on the simulation of 100,000 sample paths for each case at $T = 1, 2, 5, 10, 20$, respectively, are given by Figure 8. Note that, for all cases above, they have the same average (expected) loss of 0.5. However, the Case 3 of exponential distribution obviously products the heaviest tailed loss distributions, whereas the Case 1 has the lightest ones.

Acknowledgments

The authors would like to thank all reviewers for extremely helpful and constructive comments and suggestions, which have significantly improved our paper. They are particularly grateful to Prof. Mohammad Mousavi at the University of Pittsburgh, for sharing his detailed insights on numerically implementing the projection scheme. The corresponding author Hongbiao Zhao acknowledges the financial support from the *National Natural Science Foundation of China* (#71401147).

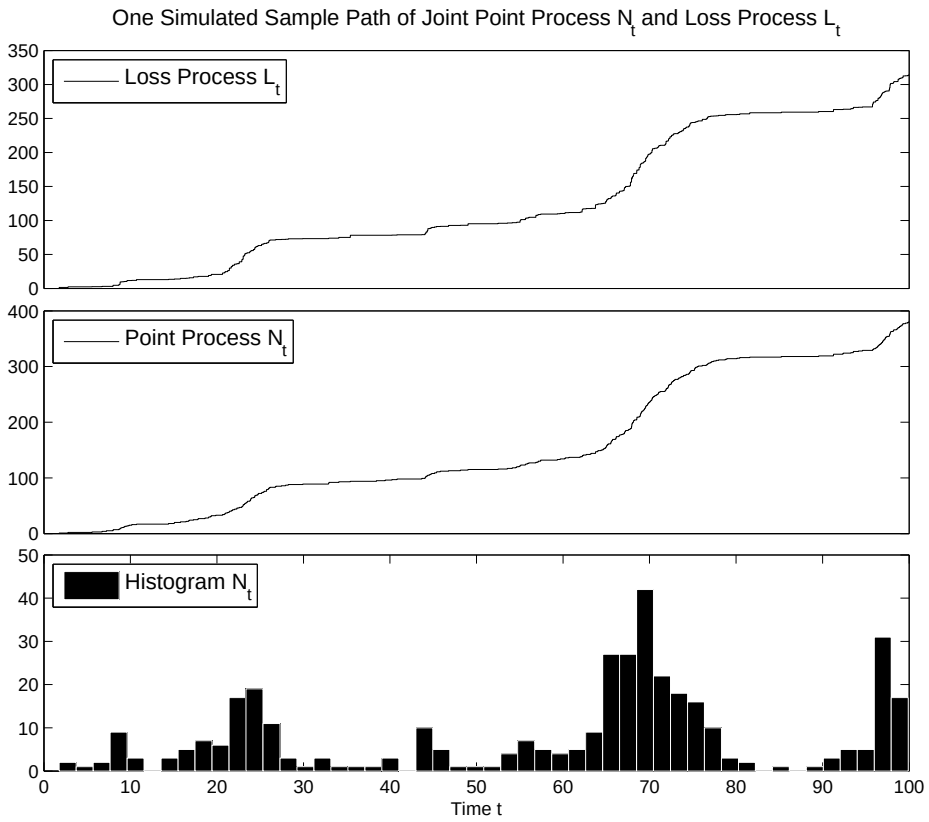


Figure 7: A simulated sample path of the joint point process N_t and loss process L_t based on Algorithm 3.6

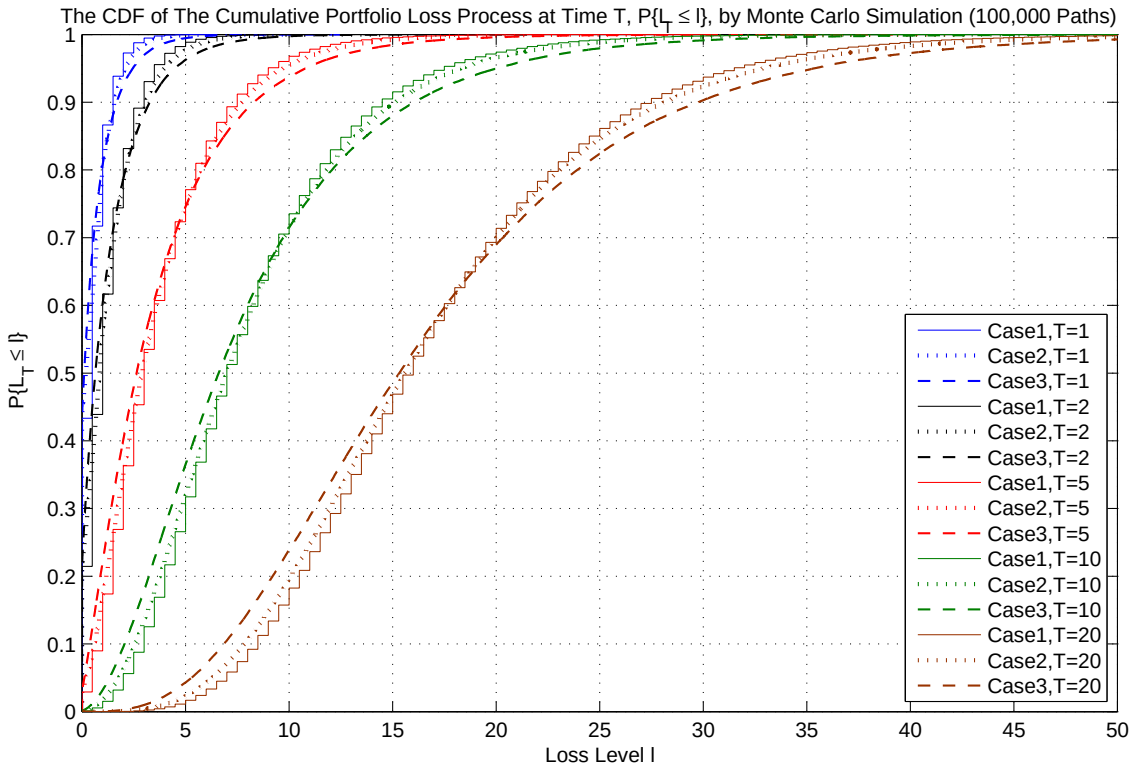


Figure 8: The CDF of cumulative portfolio loss process at time T estimated by our exact simulation

References

Aït-Sahalia, Y., Cacho-Diaz, J., and Laeven, R. J. (2015). Modeling financial contagion using mutually exciting jump processes. *Journal of Financial Economics*, 117(3):585–606.

Aït-Sahalia, Y., Laeven, R. J., and Pelizzon, L. (2014). Mutual excitation in Eurozone sovereign CDS. *Journal of Econometrics*, 183(2):151–167.

Altman, E. I., Brady, B., Resti, A., and Sironi, A. (2005). The link between default and recovery rates: Theory, empirical evidence, and implications. *The Journal of Business*, 78(6):2203–2228.

Asmussen, S. and Glynn, P. W. (2007). *Stochastic Simulation: Algorithms and Analysis*. Springer.

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

Bowsher, C. G. (2007). Modelling security market events in continuous time: Intensity based, multivariate point process models. *Journal of Econometrics*, 141(2):876–912.

Brémaud, P. and Massoulié, L. (2002). Power spectra of general shot noises and Hawkes point processes with a random excitation. *Advances in Applied Probability*, 34(1):205–222.

Brix, A. and Kendall, W. S. (2002). Simulation of cluster point processes without edge effects. *Advances in Applied Probability*, 34(2):267–280.

Broadie, M. and Kaya, Ö. (2006). Exact simulation of stochastic volatility and other affine jump diffusion processes. *Operations Research*, 54(2):217–231.

Bru, M.-F. (1991). Wishart processes. *Journal of Theoretical Probability*, 4(4):725–751.

Chavez-Demoulin, V., Davison, A. C., and McNeil, A. J. (2005). Estimating value-at-risk: a point process approach. *Quantitative Finance*, 5(2):227–234.

Chen, N. and Huang, Z. (2013). Localization and exact simulation of Brownian motion-driven stochastic differential equations. *Mathematics of Operations Research*, 38(3):591–616.

Cox, J. C., Ingersoll Jr, J. E., and Ross, S. A. (1985). A theory of the term structure of interest rates. *Econometrica*, 53(2):385–407.

Dassios, A. and Jang, J. (2003). Pricing of catastrophe reinsurance and derivatives using the Cox process with shot noise intensity. *Finance and Stochastics*, 7(1):73–95.

Dassios, A., Jang, J., and Zhao, H. (2015). A risk model with renewal shot-noise Cox process. *Insurance: Mathematics and Economics*, 65:55–65.

Dassios, A. and Nagaradjasarma, J. (2006). The square-root process and Asian options. *Quantitative Finance*, 6(4):337–347.

Dassios, A. and Zhao, H. (2011). A dynamic contagion process. *Advances in Applied Probability*, 43(3):814–846.

Dassios, A. and Zhao, H. (2012). Ruin by dynamic contagion claims. *Insurance: Mathematics and Economics*, 51(1):93–106.

Dassios, A. and Zhao, H. (2013). Exact simulation of Hawkes process with exponentially decaying intensity. *Electronic Communications in Probability*, 18(62):1–13.

Dassios, A. and Zhao, H. (2017). A generalised contagion process with an application to credit risk. *International Journal of Theoretical and Applied Finance*, 20(1):1–33.

Duffie, D., Eckner, A., Horel, G., and Saita, L. (2009). Frailty correlated default. *The Journal of Finance*, 64(5):2089–2123.

Duffie, D., Filipovic, D., and Schachermayer, W. (2003). Affine processes and applications in finance. *Annals of Applied Probability*, 13(3):984–1053.

Duffie, D. and Glynn, P. (1995). Efficient Monte Carlo simulation of security prices. *The Annals of Applied Probability*, 5(4):897–905.

Duffie, D., Pan, J., and Singleton, K. (2000). Transform analysis and asset pricing for affine jump-diffusions. *Econometrica*, 68(6):1343–1376.

Dufresne, D. (2001). The integrated square-root process. Working paper. University of Melbourne.

Engle, R. F. and Russell, J. R. (1998). Autoregressive conditional duration: A new model for irregularly spaced transaction data. *Econometrica*, 66(5):1127–1162.

Errais, E., Giesecke, K., and Goldberg, L. R. (2010). Affine point processes and portfolio credit risk. *SIAM Journal on Financial Mathematics*, 1(1):642–665.

Feller, W. (1951). Two singular diffusion problems. *The Annals of Mathematics*, 54(1):173–182.

Giesecke, K., Kakavand, H., and Mousavi, M. (2008). Simulating point processes by intensity projection. In *Proceedings of the 2008 Winter Simulation Conference*, pages 560–568. IEEE Press.

Giesecke, K., Kakavand, H., and Mousavi, M. (2011a). Exact simulation of point processes with stochastic intensities. *Operations Research*, 59(5):1233–1245.

Giesecke, K. and Kim, B. (2007). Estimating tranche spreads by loss process simulation. In *Proceedings of the 2007 Winter Simulation Conference*, pages 967–975. IEEE Press.

Giesecke, K. and Kim, B. (2011). Risk analysis of Collateralized Debt Obligations. *Operations Research*, 59(1):32–49.

Giesecke, K., Kim, B., and Zhu, S. (2011b). Monte Carlo algorithms for default timing problems. *Management Science*, 57(12):2115–2129.

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

Glasserman, P., , and Li, J. (2005). Importance sampling for portfolio credit risk. *Management Science*, 51(11):1643–1656.

Glasserman, P. (2003). *Monte Carlo Methods in Financial Engineering*. Springer.

Glasserman, P. and Kim, K.-K. (2011). Gamma expansion of the Heston stochastic volatility model. *Finance and Stochastics*, 15(2):267–296.

Hawkes, A. G. (1971a). Point spectra of some mutually exciting point processes. *Journal of the Royal Statistical Society. Series B (Methodological)*, 33(3):438–443.

Hawkes, A. G. (1971b). Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90.

Heston, S. L. (1993). A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Review of Financial Studies*, 6(2):327–343.

Lamberton, D. and Lapeyre, B. (2008). *Introduction to Stochastic Calculus Applied to Finance*. Chapman & Hall.

Large, J. (2007). Measuring the resiliency of an electronic limit order book. *Journal of Financial Markets*, 10(1):1–25.

Lewis, P. A. and Shedler, G. S. (1979). Simulation of nonhomogeneous Poisson processes by thinning. *Naval Research Logistics Quarterly*, 26(3):403–413.

Meyer, P.-A. (1971). Démonstration simplifiée d'un théorème de Knight. *Séminaire de Probabilités V Université de Strasbourg*, 191:191–195.

Møller, J. and Rasmussen, J. G. (2005). Perfect simulation of Hawkes processes. *Advances in Applied Probability*, 37:629–646.

Møller, J. and Rasmussen, J. G. (2006). Approximate simulation of Hawkes processes. *Methodology and Computing in Applied Probability*, 8(1):53–64.

Ogata, Y. (1981). On Lewis' simulation method for point processes. *IEEE Transactions on Information Theory*, 27(1):23–31.

Ogata, Y. and Akaike, H. (1982). On linear intensity models for mixed doubly stochastic Poisson and self-exciting point processes. *Journal of the Royal Statistical Society. Series B (Methodological)*, 44(1):102–107.

Zhang, X., Blanchet, J., Giesecke, K., and Glynn, P. W. (2015). Affine point processes: Approximation and efficient simulation. *Mathematics of Operations Research*, 40(4):797–819.

Zhao, H. (2012). *A Dynamic Contagion Process for Modelling Contagion Risk in Finance and Insurance*. PhD thesis, The London School of Economics and Political Science (LSE).

Zhu, L. (2014). Limit theorems for a Cox-Ingersoll-Ross process with Hawkes jumps. *Journal of Applied Probability*, 51(3):699–712.