Quantitative Finance

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/rquf20

A fast algorithm for simulation of rough volatility models

Jingtang Ma & Haofei Wu

To cite this article: Jingtang Ma & Haofei Wu (2022) A fast algorithm for simulation of rough volatility models, Quantitative Finance, 22:3, 447-462, DOI: 10.1080/14697688.2021.1970213

To link to this article: https://doi.org/10.1080/14697688.2021.1970213

Quantitative Finance, 2022
Vol. 22, No. 3, 447–462, https://doi.org/10.1080/14697688.2021.1970213

A fast algorithm for simulation of rough volatility models

JINGTANG MA*† and HAOFEI WU‡

†School of Economic Mathematics and Fintech Innovation Center, Southwestern University of Finance and Economics, Chengdu 611130, People's Republic of China
‡School of Economic Mathematics, Southwestern University of Finance and Economics, Chengdu 611130, People's Republic of China

(*Received 8 November 2020; accepted 13 August 2021; published online 14 October 2021*)

A rough volatility model contains a stochastic Volterra integral with a weakly singular kernel. The classical Euler-Maruyama algorithm is not very efficient for simulating this kind of model because one needs to keep records of all the past path-values and thus the computational complexity is too large. This paper develops a fast two-step iteration algorithm using an approximation of the weakly singular kernel with a sum of exponential functions. Compared to the Euler-Maruyama algorithm, the complexity of the fast algorithm is reduced from $O(N^2)$ to $O(N \log N)$ or $O(N \log^2 N)$ for simulating one path, where N is the number of time steps. Further, the fast algorithm is developed to simulate rough Heston models with (or without) regime switching, and multi-factor approximation algorithms are also studied and compared. The convergence rates of the Euler-Maruyama algorithm and the fast algorithm are proved. A number of numerical examples are carried out to confirm the high efficiency of the proposed algorithm.

Keywords: Rough Heston models; Fractional Brownian motion; Fast simulation algorithms; Monte-Carlo methods; Regime switching

JEL Classification:: G13, C02

1. Introduction

In this paper, we develop a fast algorithm to simulate the rough volatility model

$$\mathrm{d}S_t = rS_t\,\mathrm{d}t + S_t\sqrt{V_t}\,\mathrm{d}W_t, \tag{1}$$

$$V_t = V_0 + \frac{1}{\Gamma(1-\alpha)}\int_0^t (t-s)^{-\alpha} f(V_s)\,\mathrm{d}s + \frac{1}{\Gamma(1-\alpha)}\int_0^t (t-s)^{-\alpha} g(V_s)\,\mathrm{d}B_s,$$
$$t \in [0,T],\ \alpha \in (0,1/2), \tag{2}$$

where S_0, V_0, T are positive constants denoting the initial values and the maturity time, respectively, r nonnegative constant denoting interest rate, f and g two continuous functions on $[0,+\infty)$, and W and B two Brownian motions with correlation coefficient $\rho \in [-1,1]$. Two typical examples of rough volatility models are given as follows:

(i) If

$$f(V_s) = \kappa(\theta - V_s) \quad \text{and} \quad g(V_s) = \kappa\epsilon\sqrt{V_s}, \tag{3}$$

where κ, θ, ϵ are given positive constants, then the rough volatility model (1)–(2) becomes the rough Heston model of El Euch and Rosenbaum (2018), and if further $\alpha = 0$, then it reduces to the classical Heston model of Heston (1993).

(ii) If the mean-reversion level θ in (3) depends on a continuous-time Markov chain, then it is the regime switching rough Heston model of Alfeus *et al.* (2019). More precisely, the regime switching θ is given by

$$\theta = \theta_s := \langle \boldsymbol{\theta}, \mathbf{Z}_s \rangle \tag{4}$$

where $\boldsymbol{\theta} := (\theta^1, \ldots, \theta^k)^\top$ with $\theta^i,\ i = 1, \ldots, k$ being positive constants which represent different mean-reversion level in different regime states, $\mathbf{Z}$ is a Markov chain in the set $\mathbf{Z}_s \in \{\mathbf{e}_1, \ldots, \mathbf{e}_k\}$ where $\mathbf{e}_i \in \mathbb{R}^k$ is a column vector with one in the ith position and zeros elsewhere, $i = 1, \ldots, k$, and $\langle \cdot, \cdot \rangle$ denotes the inner product of two vectors.

*Corresponding author. Email: mjt@swufe.edu.cn

It is observed from the empirical studies of Gatheral *et al.* (2018) that volatility is much rougher than that of the classical volatility model and that roughness can be characterized by fractional Brownian motion and an integral process $\int_0^t (t-s)^{H-1/2}\,\mathrm{d}W_s$ which has Hölder regularity $H-\epsilon$ for any $\epsilon > 0$, where H is the Hurst parameter less than 1/2. So in the rough Heston model, this integral process is often introduced in the squared volatility process. Based on a limit theorem for the nearly unstable Hawkes process, Jaisson and Rosenbaum (2015, 2016) prove that when $\alpha \in (0, 1/2)$, after proper re-scaling the law of a nearly unstable heavy tailed Hawkes process converges to a rough version of the squared volatility process which constitutes the rough Heston model together with the asset price process. Enlightened by Jaisson and Rosenbaum (2016), El Euch *et al.* (2018) propose a microscopic price model and prove that it converges to a rough Heston model as the scaling time T goes to infinity. Furthermore, El Euch and Rosenbaum (2018) modify the sequence of Hawkes processes to obtain a non-degenerate rough volatility and derive the Riccati equation for the characteristic function of the asset log-price. El Euch and Rosenbaum (2018) derive the rough volatility model when the mean-reversion level θ in (3) is replaced by a time-dependent one in order to be consistent with the market forward variance curve. Callegaro *et al.* (2020) propose a fast hybrid scheme to solve the option pricing problem for rough Heston models through characteristic functions, and Dastgerdi and Bastani (2020) give an efficient method to solve a fractional Riccati differential equation for the characteristic function. Alfeus *et al.* (2019) study option pricing under the regime switching rough Heston model where the mean-reversion level θ in (3) is regime-dependent. There are also other definitions of rough Heston models (see e.g. Guennoun *et al.* 2018).

It is hard to simulate the paths of the rough Heston model, as (2) contains an integral process and the kernel $(t-s)^{-\alpha}$ with $\alpha \in (0, 1/2)$ is singular at point $s = t$. To simulate a path at a time level, one needs to keep all the past information. So the simulation is extremely time-consuming and the complexity of the Euler-Maruyama algorithm for simulating one path is proportional to $O(N^2)$ which is too large. Moreover, since the integrand is singular at $s = t$, the discretization of the integral process needs special treatment to attain good accuracy. Wood and Chan (1994) propose a simulation of stationary Gaussian processes and based on it Coeurjolly (2000) gives a method for fractional Brownian motion with complexity $O(N \log N)$. But the method of Wood-Chan cannot be used for solving the stochastic Volterra integral equations (2).

The aim of this paper is to develop a fast simulation algorithm. Using the approximation of the kernel by a sum of exponential functions, a two-step algorithm is designed and the complexity of simulating one path is about $O(N \log N)$ for $T \gg 1$ and $O(N \log^2 N)$ for $T \approx 1$. The fast algorithm is used to simulate rough Heston models in El Euch and Rosenbaum (2018) and Alfeus *et al.* (2019), i.e. (i) and (ii) above, and the option values are evaluated by the average of the payoff on the sample paths at maturity. Moreover, algorithms are developed to simulate the multi-factor Heston models which approximate the rough Heston models (see Abi Jaber and El Euch 2019).

The rest of the paper is arranged as follows. In Section 2, we derive a fast algorithm for simulation of the rough volatility model with the rough Heston models of El Euch and Rosenbaum (2018) and the regime switching rough Heston models of Alfeus *et al.* (2019). In Section 3, we simulate the multi-factor Heston models which approximate the rough Heston models and the regime switching rough Heston models. In Section 4, we prove the convergence rates of the Euler-Maruyama algorithm and the fast algorithm. In Section 5, we carry out numerical examples to confirm the high efficiency of the fast simulation algorithm. Conclusions are given in the final section.

2. Euler-Maruyama algorithm and the fast algorithm

We first modify Euler-Maruyama algorithm of Liang *et al.* (2017) to simulate the stochastic Volterra integral equation (2). Let $\{t_n := n\tau,\ n = 0, 1, \ldots, N\}$ be a given uniform mesh on $[0, T]$, where N is a positive integer and $\tau = T/N$. At time level t_n, the stochastic Volterra integral equation is written as

$$V_{t_n} = V_0 + \frac{1}{\Gamma(1-\alpha)} \int_0^{t_n} (t_n - s)^{-\alpha} f(V_s)\,\mathrm{d}s + \frac{1}{\Gamma(1-\alpha)} \int_0^{t_n} (t_n - s)^{-\alpha} g(V_s)\,\mathrm{d}B_s. \tag{5}$$

We then derive that

$$\begin{aligned} V_{t_n} &= V_0 + \frac{1}{\Gamma(1-\alpha)} \sum_{k=1}^{n} \int_{t_{k-1}}^{t_k} (t_n - s)^{-\alpha} f(V_s)\,\mathrm{d}s \\ &\quad + \frac{1}{\Gamma(1-\alpha)} \sum_{k=1}^{n} \int_{t_{k-1}}^{t_k} (t_n - s)^{-\alpha} g(V_s)\,\mathrm{d}B_s \\ &\approx V_0 + \frac{1}{\Gamma(1-\alpha)} \sum_{k=1}^{n} f\left(V_{t_{k-1}}\right) \int_{t_{k-1}}^{t_k} (t_n - s)^{-\alpha}\,\mathrm{d}s \\ &\quad + \frac{1}{\Gamma(1-\alpha)} \sum_{k=1}^{n} g\left(V_{t_{k-1}}\right) \int_{t_{k-1}}^{t_k} (t_n - s)^{-\alpha}\,\mathrm{d}B_s \\ &= V_0 + \frac{1}{\Gamma(2-\alpha)} \sum_{k=1}^{n} f\left(V_{t_{k-1}}\right) \\ &\quad \times \left[(t_n - t_{k-1})^{1-\alpha} - (t_n - t_k)^{1-\alpha}\right] \\ &\quad + \frac{1}{\Gamma(1-\alpha)} \sum_{k=1}^{n} g\left(V_{t_{k-1}}\right) \int_{t_{k-1}}^{t_k} (t_n - s)^{-\alpha}\,\mathrm{d}B_s. \end{aligned} \tag{6}$$

Since

$$\int_{t_{k-1}}^{t_k} (t_n - s)^{-2\alpha}\,\mathrm{d}s = \frac{(t_n - t_{k-1})^{1-2\alpha} - (t_n - t_k)^{1-2\alpha}}{1-2\alpha}, \quad \text{with } \alpha \in (0, 1/2),$$

is bounded, it follows from the classical result in Shreve (2004, Theorem 4.4.9) that the following Itô integral

is normally distributed as follows

$$\begin{aligned}\int_{t_{k-1}}^{t_k} (t_n - s)^{-\alpha}\,\mathrm{d}B_s &\sim \mathcal{N}\left(0, \int_{t_{k-1}}^{t_k} (t_n - s)^{-2\alpha}\,\mathrm{d}s\right)\\ &= \mathcal{N}\left(0, \frac{(t_n - t_{k-1})^{1-2\alpha} - (t_n - t_k)^{1-2\alpha}}{1-2\alpha}\right). \end{aligned} \tag{7}$$

Using (6) and (7), we derive the modified Euler-Maruyama algorithm for (2) as follows.

ALGORITHM 1 (Modfied Euler-Maruyama algorithm) *Let $V_{t_n}^N$ be the approximation to the solution V_{t_n} of* (2) *at time $t = t_n$ for $n = 0, 1, \ldots, N$. Then the modified algorithm is given by*

$$\begin{aligned} V_{t_n}^N = V_0 &+ \frac{1}{\Gamma(2-\alpha)} \sum_{k=1}^{n} f\left(V_{t_{k-1}}^N\right)\\ &\times \left[(t_n - t_{k-1})^{1-\alpha} - (t_n - t_k)^{1-\alpha}\right]\\ &+ \frac{1}{\Gamma(1-\alpha)} \sum_{k=1}^{n} g\left(V_{t_{k-1}}^N\right)\\ &\times \left[\frac{(t_n - t_{k-1})^{1-2\alpha} - (t_n - t_k)^{1-2\alpha}}{1-2\alpha}\right]^{1/2} Z_{t_k}, \end{aligned} \tag{8}$$

where $Z_{t_k} \sim \mathcal{N}(0,1)$.

Usually, we count the number of the multiplication for the complexity. From (8), we can see that the complexity is accumulated with the time-stepping. The complexity of the Euler-Maruyama algorithm (8) to simulate a whole path from t_0 to t_N is proportional to $2\sum_{n=1}^{N} n = N(N+1) = O(N^2)$. This high complexity makes it impossible to simulate many paths.

We now develop a fast algorithm to reduce the complexity. To this end, we start from the approximation of the kernel function by the sum of exponential functions. From the definition of Γ function (see Olver *et al.* 2010), we can write the kernel function $t^{-\alpha}$ for $\alpha > 0$ into an integral form

$$t^{-\alpha} = \frac{1}{\Gamma(\alpha)} \int_0^\infty e^{-ts} s^{\alpha-1}\,\mathrm{d}s. \tag{9}$$

This is further written as for $t \in [\tau, T]$ and $\tau = T/N \in (0, 1]$,

$$\begin{aligned}\Gamma(\alpha) t^{-\alpha} &= \int_0^\infty e^{-ts} s^{\alpha-1}\,\mathrm{d}s\\ &= \int_0^{2^{-M}} e^{-ts} s^{\alpha-1}\,\mathrm{d}s + \int_{2^{-M}}^{2^{N+1}} e^{-ts} s^{\alpha-1}\,\mathrm{d}s\\ &\quad + \int_{2^{N+1}}^\infty e^{-ts} s^{\alpha-1}\,\mathrm{d}s\\ &\approx \int_0^{2^{-M}} e^{-ts} s^{\alpha-1}\,\mathrm{d}s + \sum_{j=-M}^{-1} \int_{2^j}^{2^{j+1}} e^{-ts} s^{\alpha-1}\,\mathrm{d}s\\ &\quad + \sum_{j=0}^{N} \int_{2^j}^{2^{j+1}} e^{-ts} s^{\alpha-1}\,\mathrm{d}s, \end{aligned} \tag{10}$$

where $M = [O(\log T)]$ and $N = [O(\log\log\frac{1}{\xi} + \log\frac{1}{\tau})]$, $[\cdot]$ denotes taking integer operation and $\xi > 0$ is a given absolute error tolerance for the approximation of the kernel function $t^{-\alpha}$.

The Gauss integration formulas are applied to the definite integrals in (10). To be more detailed, we let $s_{o,1}, \ldots, s_{o,n_o}$ and $w_{o,1}, \ldots, w_{o,n_o}$ be the nodes and weights for the n_o-point Gauss-Jacobi quadrature on the interval $[0, 2^{-M}]$, where $n_o = [O(\log\frac{1}{\xi})]$, let $s_{j,1}, \ldots, s_{j,n_s}$ and $w_{j,1}, \ldots, w_{j,n_s}$ be the nodes and weights for the n_s-point Gauss-Legendre quadrature on the small interval $[2^j, 2^{j+1}]$, $j = -M, \ldots, -1$, where $n_s = [O(\log\frac{1}{\xi})]$, and let $s_{j,1}, \ldots, s_{j,n_l}$ and $w_{j,1}, \ldots, w_{j,n_l}$ be the nodes and weights for the n_l-point Gauss-Legendre quadrature on the small interval $[2^j, 2^{j+1}]$, $j = 0, \ldots, N$, where $n_l = [O(\log\frac{1}{\xi} + \log\frac{1}{\tau})]$. Then (10) is approximated by

$$\begin{aligned}\Gamma(\alpha) t^{-\alpha} \approx &\sum_{k=1}^{n_o} e^{-s_{o,k}t} w_{o,k} + \sum_{j=-M}^{-1} \sum_{k=1}^{n_s} e^{-s_{j,k}t} s_{j,k}^{\alpha-1} w_{j,k}\\ &+ \sum_{j=0}^{N} \sum_{k=1}^{n_l} e^{-s_{j,k}t} s_{j,k}^{\alpha-1} w_{j,k}. \end{aligned} \tag{11}$$

This is re-written into the following compact form:

$$t^{-\alpha} \approx \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l t}, \tag{12}$$

where

$$\begin{aligned} N_{\exp} = \Bigg[O\Bigg(&\log\frac{1}{\xi}\left(\log\log\frac{1}{\xi} + \log\frac{T}{\tau}\right)\\ &+ \log\frac{1}{\tau}\left(\log\log\frac{1}{\xi}\log\frac{1}{\tau}\right)\Bigg)\Bigg], \end{aligned} \tag{13}$$

and

$$\begin{aligned}\left\{x_l, l = 1, \ldots, N_{\exp}\right\} &= \left\{s_{o,1}, \ldots, s_{o,n_o}\right\}\\ &\cup \left(\bigcup_{j=-M}^{-1} \{s_{j,1}, \ldots, s_{j,n_s}\}\right)\\ &\cup \left(\bigcup_{j=0}^{N} \{s_{j,1}, \ldots, s_{j,n_l}\}\right), \end{aligned} \tag{14}$$

$$\begin{aligned}&\left\{\omega_l, l = 1, \ldots, N_{\exp}\right\}\\ &= \left\{w_{o,1}/\Gamma(\alpha), \ldots, w_{o,n_o}/\Gamma(\alpha)\right\}\\ &\quad \cup \left(\bigcup_{j=-M}^{-1} \left\{s_{j,1}^{\alpha-1} w_{j,1}/\Gamma(\alpha), \ldots, s_{j,n_s}^{\alpha-1} w_{j,n_s}/\Gamma(\alpha)\right\}\right)\\ &\quad \cup \left(\bigcup_{j=0}^{N} \left\{s_{j,1}^{\alpha-1} w_{j,1}/\Gamma(\alpha), \ldots, s_{j,n_l}^{\alpha-1} w_{j,n_l}/\Gamma(\alpha)\right\}\right). \end{aligned} \tag{15}$$

The following error estimation of the approximation (12) is given by Jiang *et al.* (2015):

$$\left| t^{-\alpha} - \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l t} \right| \leq \xi, \quad \text{for } t \in [\tau, T].$$

Now we derive the fast algorithm to solve the stochastic Volterra integral equation (2). Denote

$$I_1^n := \int_0^{t_n} (t_n - s)^{-\alpha} f(V_s)\, \mathrm{d}s,$$
$$I_2^n := \int_0^{t_n} (t_n - s)^{-\alpha} g(V_s)\, \mathrm{d}B_s. \tag{16}$$

Then we use the approximation of the kernel (12), we obtain that

$$\begin{aligned} I_1^n &= \int_{t_{n-1}}^{t_n} (t_n - s)^{-\alpha} f(V_s)\, \mathrm{d}s + \int_0^{t_{n-1}} (t_n - s)^{-\alpha} f(V_s)\, \mathrm{d}s \\ &\approx \int_{t_{n-1}}^{t_n} (t_n - s)^{-\alpha} f(V_s)\, \mathrm{d}s + \int_0^{t_{n-1}} \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l(t_n - s)} f(V_s)\, \mathrm{d}s \\ &= \int_{t_{n-1}}^{t_n} (t_n - s)^{-\alpha} f(V_s)\, \mathrm{d}s \\ &\quad + \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l \tau} \int_0^{t_{n-1}} e^{-x_l(t_{n-1} - s)} f(V_s)\, \mathrm{d}s \\ &\approx \frac{\tau^{1-\alpha}}{1-\alpha} f\left(V_{t_{n-1}}\right) + \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l \tau} H_l(t_{n-1}), \end{aligned} \tag{17}$$

where $\tau = t_n - t_{n-1} \equiv T/N$ and

$$\begin{aligned} H_l(t_{n-1}) &:= \int_0^{t_{n-1}} e^{-x_l(t_{n-1} - s)} f(V_s)\, \mathrm{d}s \\ &= \int_{t_{n-2}}^{t_{n-1}} e^{-x_l(t_{n-1} - s)} f(V_s)\, \mathrm{d}s \\ &\quad + e^{-x_l \tau} \int_0^{t_{n-2}} e^{-x_l(t_{n-2} - s)} f(V_s)\, \mathrm{d}s \\ &= \int_{t_{n-2}}^{t_{n-1}} e^{-x_l(t_{n-1} - s)} f(V_s)\, \mathrm{d}s + e^{-x_l \tau} H_l(t_{n-2}) \\ &\approx \frac{1 - e^{-x_l \tau}}{x_l} f\left(V_{t_{n-2}}\right) + e^{-x_l \tau} H_l(t_{n-2}), \end{aligned} \tag{18}$$

with $H_l(t_0) = 0$ for $1 \leq l \leq N_{\exp}$. Similarly, we derive that

$$\begin{aligned} I_2^n &= \int_{t_{n-1}}^{t_n} (t_n - s)^{-\alpha} g(V_s)\, \mathrm{d}B_s + \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l \tau} J_l(t_{n-1}) \\ &\approx \tau^{-\alpha} g\left(V_{t_{n-1}}\right)\left(B_{t_n} - B_{t_{n-1}}\right) + \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l \tau} J_l(t_{n-1}), \\ &= \tau^{\frac{1}{2}-\alpha} g\left(V_{t_{n-1}}\right) Z_{t_n} + \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l \tau} J_l(t_{n-1}), \end{aligned} \tag{19}$$

with $Z_{t_n} \sim \mathcal{N}(0,1)$, where

$$\begin{aligned} J_l(t_{n-1}) &:= \int_0^{t_{n-1}} e^{-x_l(t_{n-1} - s)} g(V_s)\, \mathrm{d}s \\ &= \int_{t_{n-2}}^{t_{n-1}} e^{-x_l(t_{n-1} - s)} g(V_s)\, \mathrm{d}B_s \\ &\quad + e^{-x_l \tau} \int_0^{t_{n-2}} e^{-x_l(t_{n-2} - s)} g(V_s)\, \mathrm{d}B_s \\ &= \int_{t_{n-2}}^{t_{n-1}} e^{-x_l(t_{n-1} - s)} g(V_s)\, \mathrm{d}B_s + e^{-x_l \tau} J_l(t_{n-2}) \\ &\approx e^{-x_l \tau} g\left(V_{t_{n-2}}\right)\left(B_{t_{n-1}} - B_{t_{n-2}}\right) + e^{-x_l \tau} J_l(t_{n-2}) \\ &= e^{-x_l \tau} g\left(V_{t_{n-2}}\right) \sqrt{\tau}\, Z_{t_{n-1}} + e^{-x_l \tau} J_l(t_{n-2}), \end{aligned} \tag{20}$$

with $J_l(t_0) = 0$ for $1 \leq l \leq N_{\exp}$ and $Z_{t_{n-1}} := \frac{B_{t_{n-1}} - B_{t_{n-2}}}{\sqrt{\tau}} \sim \mathcal{N}(0,1)$.

Combining (17) and (19) leads to the fast algorithm for simulation of (2).

Algorithm 2 (Fast algorithm) *The fast algorithm for simulation of* (2) *is given by*

$$\begin{aligned} V_{t_n}^N &= V_0 + \frac{\tau^{1-\alpha}}{\Gamma(2-\alpha)} f\left(V_{t_{n-1}}^N\right) \\ &\quad + \frac{1}{\Gamma(1-\alpha)} \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l \tau} H_l^N(t_{n-1}) \\ &\quad + \frac{\tau^{\frac{1}{2}-\alpha}}{\Gamma(1-\alpha)} g\left(V_{t_{n-1}}^N\right) Z_{t_n} \\ &\quad + \frac{1}{\Gamma(1-\alpha)} \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l \tau} J_l^N(t_{n-1}), \\ &\quad n = 1, \ldots, N, \\ H_l^N(t_{n-1}) &= \frac{f\left(V_{t_{n-2}}^N\right)}{x_l}\left(1 - e^{-x_l \tau}\right) + e^{-x_l \tau} H_l^N(t_{n-2}), \\ &\quad n = 2, \ldots, N, \\ J_l^N(t_{n-1}) &= e^{-x_l \tau} g\left(V_{t_{n-2}}^N\right) \sqrt{\tau}\, Z_{t_{n-1}} + e^{-x_l \tau} J_l^N(t_{n-2}), \\ &\quad n = 2, \ldots, N, \end{aligned}$$

with $V_{t_0}^N = V_0$, $H_l^N(t_0) = 0$, $J_l^N(t_0) = 0$, $\tau = T/N$, $Z_{t_n} \sim \mathcal{N}(0,1)$ *for* $n = 1, \ldots, N$, *and* $N_{\exp}, x_l, \omega_l$ *are given by* (13), (14), (15), *respectively.*

The complexity of Algorithm 2 is $O(NN_{\exp})$ and according to (13), $N_{\exp} = O(\log N)$ if $T \gg 1$ and $N_{\exp} = O(\log^2 N)$ if $T \approx 1$. In either case, Algorithm 2 greatly increases the computational efficiency as compared to the Euler-Maruyama algorithm (8) with complexity $O(N^2)$.

Now we consider the fast simulation of the regime switching rough Heston model of Alfeus *et al.* (2019), i.e. (1)–(2) with f given by (3) and the mean-reversion level θ in (3) depending on a continuous-time Markov chain given by (4). The definitions of I_1^n and I_2^n in (16) now read as

$$I_1^n = \int_0^{t_n} (t_n - s)^{-\alpha} \kappa(\theta_s - V_s)\, \mathrm{d}s,$$

$$I_2^n = \int_0^{t_n} (t_n - s)^{-\alpha} g(V_s)\, \mathrm{d}B_s. \tag{21}$$

The approximation I_2^n in (21) has the same form as (19). The approximation I_1^n in (21) is derived as follows:

$$\begin{aligned}
I_1^n &= \int_{t_{n-1}}^{t_n} (t_n - s)^{-\alpha} \kappa\, (\theta_s - V_s)\, \mathrm{d}s \\
&\quad + \int_0^{t_{n-1}} (t_n - s)^{-\alpha} \kappa\, (\theta_s - V_s)\, \mathrm{d}s \\
&\approx \int_{t_{n-1}}^{t_n} (t_n - s)^{-\alpha} \kappa\, (\theta_s - V_s)\, \mathrm{d}s \\
&\quad + \int_0^{t_{n-1}} \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l (t_n - s)} \kappa\, (\theta_s - V_s)\, \mathrm{d}s \\
&= \int_{t_{n-1}}^{t_n} (t_n - s)^{-\alpha} \kappa\, (\theta_s - V_s)\, \mathrm{d}s \\
&\quad + \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l \tau} \int_0^{t_{n-1}} e^{-x_l (t_{n-1} - s)} \kappa\, (\theta_s - V_s)\, \mathrm{d}s \\
&\approx \sum_{m=0}^{M_n - 1} \int_{s_m^n}^{s_{m+1}^n} (t_n - s)^{-\alpha} \kappa \left(\theta_{s_m^n} - V_{t_{n-1}}\right) \mathrm{d}s \\
&\quad + \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l \tau} H_l\, (t_{n-1}) \\
&= \sum_{m=0}^{M_n - 1} \frac{\left(t_n - s_m^n\right)^{1-\alpha} - \left(t_n - s_{m+1}^n\right)^{1-\alpha}}{1 - \alpha} \kappa \left(\theta_{s_m^n} - V_{t_{n-1}}\right) \\
&\quad + \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l \tau} H_l\, (t_{n-1}), \qquad (22)
\end{aligned}$$

with

$$\begin{aligned}
H_l\, (t_{n-1}) &:= \int_0^{t_{n-1}} e^{-x_l (t_{n-1} - s)} \kappa\, (\theta_s - V_s)\, \mathrm{d}s \\
&= \int_{t_{n-2}}^{t_{n-1}} e^{-x_l (t_{n-1} - s)} \kappa\, (\theta_s - V_s)\, \mathrm{d}s \\
&\quad + e^{-x_l \tau} \int_0^{t_{n-2}} e^{-x_l (t_{n-2} - s)} \kappa\, (\theta_s - V_s)\, \mathrm{d}s \\
&= \sum_{m=0}^{M_{n-1} - 1} \frac{e^{-x_l \left(t_{n-1} - s_{m+1}^{n-1}\right)} - e^{-x_l \left(t_{n-1} - s_m^{n-1}\right)}}{x_l} \\
&\quad \times \kappa \left(\theta_{s_m^{n-1}} - V_{t_{n-2}}\right) + e^{-x_l \tau} H_l\, (t_{n-2}), \qquad (23)
\end{aligned}$$

where $M_n - 1$ is the number of regime switching and s_m^n ($m = 1, \ldots, M_n - 1$) are regime switching times on $[t_{n-1}, t_n)$ and positive constants $\theta_{s_m^n}$ are the values of θ_s on $[s_m^n, s_{m+1}^n)$ for $m = 0, 1, \ldots, M_n - 1$ and $n = 1, \ldots, N$. In addition, $s_0^n = t_{n-1}$ and $s_{M_n}^n = t_n$ for $n = 1, \ldots, N$.

Using (22) and (23), we give the fast algorithm for simulation of the regime switching rough Heston model.

ALGORITHM 3 (Fast algorithm for regime switching rough Heston model)

The fast algorithm for simulation of (2) *with* f *and* θ *respectively given by* (3) *and* (4) *is defined as*

$$\begin{aligned}
V_{t_n}^N &= V_0 + \frac{\tau^{\frac{1}{2} - \alpha}}{\Gamma(1 - \alpha)} g(V_{t_{n-1}}^N) Z_{t_n} \\
&\quad + \sum_{m=0}^{M_n - 1} \frac{\left(t_n - s_m^n\right)^{1-\alpha} - \left(t_n - s_{m+1}^n\right)^{1-\alpha}}{\Gamma(2 - \alpha)} \\
&\quad \times \kappa \left(\theta_{s_m^n} - V_{t_{n-1}}^N\right) \\
&\quad + \frac{1}{\Gamma(1 - \alpha)} \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l \tau} H_l^N\, (t_{n-1}) \\
&\quad + \frac{1}{\Gamma(1 - \alpha)} \sum_{l=1}^{N_{\exp}} \omega_l e^{-x_l \tau} J_l^N\, (t_{n-1}), \\
&\quad \times n = 1, \ldots, N, \\
H_l^N\, (t_{n-1}) &= \sum_{m=0}^{M_{n-1} - 1} \frac{e^{-x_l \left(t_{n-1} - s_{m+1}^{n-1}\right)} - e^{-x_l \left(t_{n-1} - s_m^{n-1}\right)}}{x_l} \\
&\quad \times \kappa \left(\theta_{s_m^{n-1}} - V_{t_{n-2}}^N\right) + e^{-x_l \tau} H_l^N\, (t_{n-2}), \\
&\quad n = 2, \ldots, N, \\
J_l^N\, (t_{n-1}) &= e^{-x_l \tau} g(V_{t_{n-2}}^N) \sqrt{\tau}\, Z_{t_{n-1}} + e^{-x_l \tau} J_l^N\, (t_{n-2}), \\
&\quad n = 2, \ldots, N,
\end{aligned}$$

where the notations or variables are the same as the previous discussion without specific explanation.

Now we discuss the simulation of the stock price (1). Let $X_t = \log S_t$ where S follows (1). Then Itô's lemma gives that

$$\mathrm{d}X_t = \left(r - \frac{1}{2} V_t\right) \mathrm{d}t + \sqrt{V_t}\, \mathrm{d}W_t. \tag{24}$$

The Euler-Maruyama algorithm for discretization of (24) is then given by

$$\begin{aligned}
X_{t_n}^N &= X_{t_{n-1}}^N + \left(r - \frac{1}{2} V_{t_{n-1}}^N\right) \tau + \sqrt{V_{t_{n-1}}^N}\, \widetilde{Z}_{t_n} \sqrt{\tau}, \\
&\quad n = 1, \ldots, N, \qquad (25)
\end{aligned}$$

where $\widetilde{Z}_{t_n} \sim \mathcal{N}(0, 1)$ and it is correlated with Z_{t_n} in Algorithms 1, 2, or Algorithm 3 with coefficient $\rho \in [-1, 1]$. The generation of $\widetilde{Z}_{t_n}$ and Z_{t_n} can be realized by the well-known inverse transform sampling:

$$\begin{aligned}
Z_{t_n} &= \Phi^{-1} \left(U_{t_n}^1\right), \\
\widetilde{Z}_{t_n} &= \rho Z_{t_n} + \sqrt{1 - \rho^2}\, \Phi^{-1} \left(U_{t_n}^2\right),
\end{aligned}$$

where $U_{t_n}^1$ and $U_{t_n}^2$ are independent uniform samples and Φ^{-1} is the inverse cumulative normal distribution function.

The simulation of the rough volatility model of El Euch and Rosenbaum (2018), i.e. (1)–(2) with f and g given by (3), is realized by combining Algorithm 2 with (25) and $S_{t_n}^N = e^{X_{t_n}^N}$. The simulation of the regime switching rough volatility model of Alfeus *et al.* (2019), i.e. (1)–(2) with f and g given by (3)

and θ by (4), is implemented by combining Algorithm 3 with (25) and $S_{t_n}^N = e^{X_{t_n}^N}$. To ensure the squared volatility is positive, in practice $V_{t_n}^N$ is replaced by $(V_{t_n}^N)^+ := \max(0, V_{t_n}^N)$ (see Andersen 2008).

3. Multi-factor approximation algorithms

Abi Jaber and El Euch (2019) approximate the kernel function by

$$\frac{t^{-\alpha}}{\Gamma(1-\alpha)} \approx \sum_{j=1}^{\widetilde{N}_{\exp}} c_j e^{-\gamma_j t}, \tag{26}$$

with

$$c_j := \int_{\eta_{j-1}}^{\eta_j} \mu(d\gamma), \quad \gamma_j := \frac{1}{c_j}\int_{\eta_{j-1}}^{\eta_j} \gamma\,\mu(d\gamma),$$
$$\eta_j := j\frac{\widetilde{N}_{\exp}^{-1/5}}{T}\left(\frac{\sqrt{10}\alpha}{2+\alpha}\right)^{2/5}, \tag{27}$$

for $j = 1, \ldots, \widetilde{N}_{\exp}$, and

$$\mu(d\gamma) := \frac{\gamma^{\alpha-1}}{\Gamma(1-\alpha)\Gamma(\alpha)}\,\mathrm{d}\gamma.$$

Meanwhile, the error of the approximation (26) is estimated by Abi Jaber and El Euch (2019)

$$\left\| \frac{t^{-\alpha}}{\Gamma(1-\alpha)} - \sum_{j=1}^{\widetilde{N}_{\exp}} c_j e^{-\gamma_j t} \right\|_{L^2(0,T)} \le C\widetilde{N}_{\exp}^{-\frac{4}{5}(\frac{1}{2}-\alpha)}, \tag{28}$$

where C is a positive constant which is independent of $\widetilde{N}_{\exp}$ and $0 < \alpha < 1/2$.

Abi Jaber and El Euch (2019) use (26) to derive the approximation of the rough Heston model

$$V_t = V_0 + \frac{1}{\Gamma(1-\alpha)}\int_0^t (t-s)^{-\alpha}\kappa(\theta - V_s)\,\mathrm{d}s + \frac{1}{\Gamma(1-\alpha)}\int_0^t (t-s)^{-\alpha} g(V_s)\,\mathrm{d}B_s, \tag{29}$$

by the following multi-factor Heston model

$$V_t^{\widetilde{N}_{\exp}} = V_0 + \kappa\theta\sum_{j=1}^{\widetilde{N}_{\exp}} \frac{c_j}{\gamma_j}(1 - e^{-\gamma_j t}) + \sum_{j=1}^{\widetilde{N}_{\exp}} c_j V_t^{\widetilde{N}_{\exp},j}, \tag{30}$$

$$dV_t^{\widetilde{N}_{\exp},j} = (-\gamma_j V_t^{\widetilde{N}_{\exp},j} - \kappa V_t^{\widetilde{N}_{\exp}})\,\mathrm{d}t + g(V_t^{\widetilde{N}_{\exp}})\,\mathrm{d}B_t, \tag{31}$$

with $V_0^{\widetilde{N}_{\exp},j} = 0$, for $j = 1, \ldots, \widetilde{N}_{\exp}$.

We now give the semi-implicit Euler-Maruyama algorithm for simulation of the multi-factor models (30)–(31). This thus gives the so-called multi-factor approximation algorithm for simulation of the rough Heston model (29) (see Abi Jaber 2019).

ALGORITHM 4 (Multi-factor approximation algorithm, Abi Jaber 2019)

Let $V_{t_n}^{\widetilde{N}_{\exp},N}$ and $V_{t_n}^{\widetilde{N}_{\exp},j,N}$ denote the approximation of $V_{t_n}^{\widetilde{N}_{\exp}}$ and $V_{t_n}^{\widetilde{N}_{\exp},j}$, respectively, for $n = 0, 1, \ldots, N$. Then the algorithm for simulation of rough Heston model given by

$$V_{t_n}^{\widetilde{N}_{\exp},N} = V_{t_0}^{\widetilde{N}_{\exp},N} + \kappa\theta\sum_{j=1}^{\widetilde{N}_{\exp}} \frac{c_j}{\gamma_j}\left(1 - e^{-\gamma_j t_n}\right) + \sum_{j=1}^{\widetilde{N}_{\exp}} c_j\, V_{t_n}^{\widetilde{N}_{\exp},j,N},$$

$$V_{t_n}^{\widetilde{N}_{\exp},j,N} = \frac{V_{t_{n-1}}^{\widetilde{N}_{\exp},j,N} - \kappa V_{t_{n-1}}^{\widetilde{N}_{\exp},N}\tau + g(V_{t_{n-1}}^{\widetilde{N}_{\exp},N})\sqrt{\tau}Z_{t_n}}{1+\gamma_j\tau},$$
$$j = 1, 2, \ldots, \widetilde{N}_{\exp},$$

where $V_{t_0}^{\widetilde{N}_{\exp},N} = V_0$, $V_{t_0}^{\widetilde{N}_{\exp},j,N} = 0$, $\tau = T/N$, $Z_{t_n} \sim \mathcal{N}(0,1)$ for $n = 1, \ldots, N$, and c_j, γ_j, η_j are given by (27). *It thus gives the simulation of the rough Heston model* (29) *with $V_{t_n} \approx V_{t_n}^{\widetilde{N}_{\exp},N}$ for $n = 0, 1, \ldots, N$.*

REMARK 3.1 The complexity of Algorithm 4 for simulating one path of the rough Heston model (29) is $O(\widetilde{N}_{\exp}N)$ compared to $O(N_{\exp}N)$ of Algorithm 2. The error of the multi-factor approximation algorithm is composed by two parts: the approximation error for the multi-factor models (30)–(31) approximating the rough Heston model (29) and the simulation error of simulating the path of the multi-factor models (30)–(31). So to maintain the accuracy of path simulation, $\widetilde{N}_{\exp}$ has to be increased with N, which is shown by the numerical examples in Section 5. Unlike Algorithms 2, 4 does not give the relation between $\widetilde{N}_{\exp}$ and N. However noting that the error of kernel approximation (28) is proportional to $O(\widetilde{N}_{\exp}^{-\frac{4}{5}(\frac{1}{2}-\alpha)})$ and error of Euler-Maruyama algorithm $O(N^{-(\frac{1}{2}-\alpha)})$, to balance these two errors, the relation between $\widetilde{N}_{\exp}$ and N is roughly like $\widetilde{N}_{\exp}^{-\frac{4}{5}(\frac{1}{2}-\alpha)} \approx O(N^{-(\frac{1}{2}-\alpha)})$, i.e. $\widetilde{N}_{\exp} \approx O(N^{\frac{5}{4}})$. So the increasing speed for $\widetilde{N}_{\exp}$ is much larger than that for $N_{\exp}$. Therefore, Algorithm 2 has much less complexity than Algorithm 4 and thus is much faster in the simulation.

We now derive the multi-factor approximation algorithm for simulation of the regime switching rough Heston model

$$V_t = V_0 + \frac{1}{\Gamma(1-\alpha)}\int_0^t (t-s)^{-\alpha}\kappa(\theta_s - V_s)\,\mathrm{d}s + \frac{1}{\Gamma(1-\alpha)}\int_0^t (t-s)^{-\alpha} g(V_s)\,\mathrm{d}B_s, \tag{32}$$

where θ_s is defined by (4). We re-write (32) as

$$V_t = V_0 + \int_0^t \frac{(t-s)^{-\alpha}}{\Gamma(1-\alpha)}\kappa\theta_s\,\mathrm{d}s + \int_0^t \frac{(t-s)^{-\alpha}}{\Gamma(1-\alpha)}\left(-\kappa V_s\,\mathrm{d}s + g(V_s)\,\mathrm{d}B_s\right).$$

Compared to the no regime switching model, the only difference is on the term

$$G(t) := \int_0^t \frac{(t-s)^{-\alpha}}{\Gamma(1-\alpha)}\kappa\theta_s\,\mathrm{d}s.$$

The discretization of this term on $t = t_n$ for $n = 0, 1, \ldots, N$ is given by

$$\begin{aligned} G(t_n) &= \sum_{m=0}^{P-1} \int_{s_m}^{s_{m+1}} \frac{(t_n - s)^{-\alpha}}{\Gamma(1-\alpha)} \kappa \theta_{s_m} \, \mathrm{d}s \\ &\approx \sum_{m=0}^{P-1} \kappa \theta_{s_m} \int_{s_m}^{s_{m+1}} \sum_{j=1}^{\widetilde{N}_{\exp}} c_j e^{-\gamma_j (t_n - s)} \, \mathrm{d}s \\ &= \sum_{m=0}^{P-1} \left[\kappa \theta_{s_m} \sum_{j=1}^{\widetilde{N}_{\exp}} \frac{c_j}{\gamma_j} \left(e^{-\gamma_j (t_n - s_{m+1})} - e^{-\gamma_j (t_n - s_m)} \right) \right], \end{aligned}$$

where $P-1$ is the number of regime switching and s_m ($m = 1, \ldots, P-1$) are regime switching times on $[0, t_n)$ and positive constants θ_{s_m} is the value of θ_s on $[s_m, s_{m+1})$ for $m = 0, 1, \ldots, P-1$, specially $s_0 = 0$ and $s_P = t_n$. So we can obtain the multi-factor approximation algorithm for simulation of the regime switching rough Heston model as follows.

ALGORITHM 5 (Multi-factor approximation algorithm for regime switching rough Heston model)

The multi-factor approximation algorithm for simulation of (2) *with f, g and θ respectively given by* (3) *and* (4) *is defined as*

$$\begin{aligned} V_{t_n}^{\widetilde{N}_{\exp},N} &= V_{t_0}^{\widetilde{N}_{\exp},N} + \sum_{j=1}^{\widetilde{N}_{\exp}} c_j \, V_{t_n}^{\widetilde{N}_{\exp} j,N} \\ &\quad + \sum_{m=0}^{P-1} \left[\kappa \theta_{s_m} \sum_{j=1}^{\widetilde{N}_{\exp}} \frac{c_j}{\gamma_j} \left(e^{-\gamma_j (t_n - s_{m+1})} - e^{-\gamma_j (t_n - s_m)} \right) \right], \\ V_{t_n}^{\widetilde{N}_{\exp} j,N} &= \frac{V_{t_{n-1}}^{\widetilde{N}_{\exp} j,N} - \kappa V_{t_{n-1}}^{\widetilde{N}_{\exp},N} \tau + g(V_{t_{n-1}}^{\widetilde{N}_{\exp},N}) \sqrt{\tau} Z_{t_n}}{1 + \gamma_j \tau}, \\ &\quad j = 1, 2, \ldots, \widetilde{N}_{\exp}, \end{aligned}$$

where the notations or variables are the same as the previous discussions without specification.

4. Convergence analysis

In this section, we prove the convergence rate of Euler-Maruyama scheme and the fast algorithms for the volatility, stock log-price and the option price at time 0 under the rough Heston model (1)–(2). In Li *et al.* (2020), it has been proved that well-posedness of solutions and the convergence order of the Euler-Maruyama scheme of equation (2), if f and g in (2) satisfy the global Lipschitz condition and the linear growth condition. The results are summarized as the following lemma.

LEMMA 4.1 *There is a unique solution V_t to* (2) *which satisfies*

$$\sup_{0 \le t \le T} E[V_t]^2 \le C, \quad E[V_t - V_s]^2 \le C(t-s)^{2(\frac{1}{2}-\alpha)},$$

where C is a generic positive constant which is independent of time. Moreover let $V_{t_n}^N$ be the approximation to the solution V_{t_n} of (2) *at time $t = t_n$ for $n = 0, 1, \ldots, N$ by Algorithm* 1. *Then*

$$\sup_{1 \le n \le N} E[V_{t_n} - V_{t_n}^N]^2 \le \widetilde{C} \tau^{2(\frac{1}{2}-\alpha)}, \tag{33}$$

where $\widetilde{C}$ is a positive constant which is independent of time step $\tau := T/N$.

Proof The proofs are given by Li *et al.* (2020). ■

Now we prove that the convergence of the Euler-Maruyama scheme (25) for the stock log-price process (24).

THEOREM 4.2 *Let $X_{t_n}^N$ be the approximation to the solution X_{t_n} of* (24) *at time $t = t_n$ for $n = 0, 1, \ldots, N$ by Euler-Maruyama scheme* (25). *Then we have*

$$\sup_{1 \le n \le N} E[X_{t_n} - X_{t_n}^N]^2 \le C \tau^{2(\frac{1}{2}-\alpha)},$$

where C is a positive constant which is independent of τ.

Proof The Euler-Maruyama scheme (25) can be equivalently written as

$$\begin{aligned} X_{t_n}^N &= X_{t_{n-1}}^N + \int_{t_{n-1}}^{t^n} \left(r - \frac{1}{2} \underline{V_s^N} \right) \mathrm{d}s + \int_{t_{n-1}}^{t^n} \sqrt{\underline{V_s^N}} \, \mathrm{d}W_s, \\ &\quad n = 1, \ldots, N, \end{aligned} \tag{34}$$

where $\underline{V_s^N} := V_{t_k}^N$ for $s \in [t_k, t_{k+1})$, $k = 0, 1, \ldots, n-1$. Then subtracting (34) by the stock log-price process (24) gives that

$$X_{t_n} - X_{t_n}^N = \int_0^{t_n} \frac{1}{2} \left(\underline{V_s^N} - V_s \right) \mathrm{d}s + \int_0^{t_n} \left(\sqrt{V_s} - \sqrt{\underline{V_s^N}} \right) \mathrm{d}W_s.$$

Using the geometric inequality, Cauchy-Schwarz's inequality, Itô isometry and the global Lipschitz condition, we get

$$\begin{aligned} E[X_{t_n} - X_{t_n}^N]^2 &\le 2E \left[\int_0^{t_n} \frac{1}{2} \left(\underline{V_s^N} - V_s \right) \mathrm{d}s \right]^2 \\ &\quad + 2E \left[\int_0^{t_n} \left(\sqrt{V_s} - \sqrt{\underline{V_s^N}} \right) \mathrm{d}W_s \right]^2 \\ &\le \frac{t_n}{2} E \left[\int_0^{t_n} \left(\underline{V_s^N} - V_s \right)^2 \mathrm{d}s \right] \\ &\quad + 2E \left[\int_0^{t_n} \left(\sqrt{V_s} - \sqrt{\underline{V_s^N}} \right)^2 \mathrm{d}s \right] \\ &\le \frac{t_n}{2} E \left[\int_0^{t_n} \left(\underline{V_s^N} - V_s \right)^2 \mathrm{d}s \right] \\ &\quad + 2L^2 E \left[\int_0^{t_n} \left(V_s - \underline{V_s^N} \right)^2 \mathrm{d}s \right] \\ &= \left(\frac{t_n}{2} + 2L^2 \right) \\ &\quad \times E \left[\int_0^{t_n} \left(V_s - \underline{V_s} + \underline{V_s} - \underline{V_s^N} \right)^2 \mathrm{d}s \right] \\ &\le (T + 4L^2) \\ &\quad \times E \left[\int_0^{t_n} \left[(V_s - \underline{V_s})^2 + (\underline{V_s} - \underline{V_s^N})^2 \right] \mathrm{d}s \right] \end{aligned}$$

$$= (T + 4L^2) \sum_{i=1}^{n} \int_{t_{i-1}}^{t_i} \times \left[E(V_s - V_{t_{i-1}})^2 + E(V_{t_{i-1}} - V_{t_{i-1}}^N)^2 \right] \mathrm{d}s,$$

where $\underline{V_s} := V_{t_k}$ for $s \in [t_k, t_{k+1})$, $k = 0, 1, \ldots, n-1$. Applying Lemma 4.1, it follows that

$$\begin{aligned} \sup_{1 \le n \le N} E[X_{t_n} - X_{t_n}^N]^2 &\le \sup_{1 \le n \le N} (T + 4L^2) \\ &\quad \times \sum_{i=1}^{n} \left(\int_{t_{i-1}}^{t_i} C_1 \tau^{1-2\alpha} + C_2 \tau^{1-2\alpha} \, \mathrm{d}s \right) \\ &\le C \tau^{2(\frac{1}{2}-\alpha)}. \end{aligned}$$

The proof is thus complete. ∎

Now we prove the convergence rate of European call option price based on path simulation.

THEOREM 4.3 *Let P_0^N be the numerical option price based on paths simulation by Euler-Maruyama scheme and P_0 the true option price at time 0. Then we have*

$$[P_0 - P_0^N]^2 \le C \tau^{2(\frac{1}{2}-\alpha)},$$

where C is a positive constant which is independent of τ.

Proof From (24), (25), Lemma 4.1 and Theorem 4.2, we know that X_T and X_T^N are bounded with squared expectation and we derive that

$$\begin{aligned} \left[P_0 - P_0^N\right]^2 &= \left[e^{-rT} E\left[(S_T - K)^+ - (S_T^N - K)^+\right]\right]^2 \\ &\le e^{-2rT} E\left[(S_T - K)^+ - (S_T^N - K)^+\right]^2 \\ &\le e^{-2rT} E\left[S_T - S_T^N\right]^2 \\ &= e^{-2rT} E\left[e^{X_T} - e^{X_T^N}\right]^2 \\ &\le e^{-2rT + C_1} E\left[X_T - X_T^N\right]^2 \\ &\le C \tau^{2(\frac{1}{2}-\alpha)}. \end{aligned}$$

The proof is thus complete. ∎

Now we prove the convergence of the fast algorithm (e.g. Algorithm 2) for the rough volatility process (2). Although the idea of the proof is similar to that of Dai and Xiao (2020), our fast algorithm is different from Dai and Xiao (2020, equation (14)) and thus the proof also presents different.

THEOREM 4.4 *Let $\widetilde{V}_{t_n}^N$ be the approximation to the solution V_{t_n} of (2) at time $t = t_n$ for $n = 0, 1, \ldots, N$ by Algorithm 2, and f and g satisfy global Lipschitz conditions. Then there is a constant C which is independent of τ and ξ such that*

$$\sup_{1 \le n \le N} E[V_{t_n} - \widetilde{V}_{t_n}^N]^2 \le C \left(\tau^{\frac{1}{2}-\alpha} + \xi \right)^2,$$

where ξ is the absolute tolerance error for the evaluation of kernel function.

Proof We first prove the existence of $\widetilde{V}_{t_n}^N$. For Algorithm 2, we can reformulate $\widetilde{V}_{t_n}^N$ as the following equality

$$\begin{aligned} \widetilde{V}_{t_n}^N &= V_0 + \int_{t_{n-1}}^{t_n} (t_n - s)^{-\alpha} f(\underline{\widetilde{V}_s^N}) \, \mathrm{d}s \\ &\quad + \int_{t_{n-1}}^{t_n} (t_n - \underline{s})^{-\alpha} g(\underline{\widetilde{V}_s^N}) \, \mathrm{d}B_s \\ &\quad + \int_0^{t_{n-1}} \sum_{k=1}^{N_{\exp}} \omega_k e^{-x_k(t_n - s)} f(\underline{\widetilde{V}_s^N}) \, \mathrm{d}s \\ &\quad + \int_0^{t_{n-1}} \sum_{k=1}^{N_{\exp}} \omega_k e^{-x_k(t_n - \underline{s})} g(\underline{\widetilde{V}_s^N}) \, \mathrm{d}B_s, \end{aligned}$$

where $\underline{s} := t_k$, $\underline{\widetilde{V}_s^N} := \widetilde{V}_{t_k}^N$ for $s \in [t_k, t_{k+1})$, $k = 0, 1, \ldots, n-1$. It follows from the mean inequality, Cauchy-Schwarz's inequality, Itô isometry and the linear growth condition that

$$\begin{aligned} E[\widetilde{V}_{t_n}^N]^2 &\le E\Bigg[V_0 + \int_0^{t_n} (t_n - s)^{-\alpha} f(\underline{\widetilde{V}_s^N}) \, \mathrm{d}s \\ &\quad + \int_0^{t_n} (t_n - \underline{s})^{-\alpha} g(\underline{\widetilde{V}_s^N}) \, \mathrm{d}B_s \\ &\quad + \int_0^{t_{n-1}} \left(\sum_{k=1}^{N_{\exp}} \omega_k e^{-x_k(t_n - s)} - (t_n - s)^{-\alpha} \right) f(\underline{\widetilde{V}_s^N}) \, \mathrm{d}s \\ &\quad + \int_0^{t_{n-1}} \left(\sum_{k=1}^{N_{\exp}} \omega_k e^{-x_k(t_n - \underline{s})} - (t_n - \underline{s})^{-\alpha} \right) \\ &\quad \times g(\underline{\widetilde{V}_s^N}) \, \mathrm{d}B_s \Bigg]^2, \\ &\le 5E \Bigg[V_0^2 + \int_0^{t_n} (t_n - s)^{-\alpha} \, \mathrm{d}s \int_0^{t_n} (t_n - s)^{-\alpha} f^2(\underline{\widetilde{V}_s^N}) \, \mathrm{d}s \\ &\quad + \int_0^{t_n} (t_n - \underline{s})^{-2\alpha} g^2(\underline{\widetilde{V}_s^N}) \, \mathrm{d}s + \int_0^{t_{n-1}} \xi^2 g^2(\underline{\widetilde{V}_s^N}) \, \mathrm{d}s \\ &\quad + \int_0^{t_{n-1}} \xi \, \mathrm{d}s \int_0^{t_{n-1}} \xi f^2(\underline{\widetilde{V}_s^N}) \, \mathrm{d}s \Bigg]^2 \\ &\le 5E \Bigg[V_0^2 + C_1 \int_0^{t_{n-1}} (1 + (\underline{\widetilde{V}_s^N})^2) \, \mathrm{d}s \\ &\quad + C_2 \int_0^{t_n} (t_n - s)^{-\alpha} (1 + (\underline{\widetilde{V}_s^N})^2) \, \mathrm{d}s \\ &\quad + C_3 \int_0^{t_n} (t_n - s)^{-2\alpha} (1 + (\underline{\widetilde{V}_s^N})^2) \, \mathrm{d}s \Bigg] \\ &\le C_4 \Bigg[1 + \int_0^{t_n} E[\underline{\widetilde{V}_s^N}]^2 \, \mathrm{d}s + \int_0^{t_n} (t_n - s)^{-\alpha} E[\underline{\widetilde{V}_s^N}]^2 \, \mathrm{d}s \\ &\quad + \int_0^{t_n} (t_n - s)^{-2\alpha} E[\underline{\widetilde{V}_s^N}]^2 \, \mathrm{d}s \Bigg], \end{aligned}$$

by Gronwall's inequality, we can deduce that there is a constant C which is independent of time t such that

$$\sup_{1 \le n \le N} E[\widetilde{V}_{t_n}^N]^2 \le C. \tag{35}$$

Now we estimate the mean-squared error

$$E[V_{t_n} - \widetilde{V}_{t_n}^N]^2 = E[V_{t_n} - V_{t_n}^N + V_{t_n}^N - \widetilde{V}_{t_n}^N]^2 \le 2E[V_{t_n} - V_{t_n}^N]^2 + 2E[V_{t_n}^N - \widetilde{V}_{t_n}^N]^2, \quad (36)$$

where $V_{t_n}^N$ is the numerical solution of V_{t_n} by Euler-Maruyama scheme. The first expectation could be evaluated by Lemma 4.1. Thus we only need to estimate

$$\begin{aligned}
&\Gamma(1-\alpha)E[V_{t_n}^N - \widetilde{V}_{t_n}^N]^2 \\
&= E\Bigg[\int_0^{t_{n-1}} \Bigg[(t_n - s)^{-\alpha} f(\underline{V_s^N}) \\
&\quad - \sum_{k=1}^{N_{\exp}} \omega_k e^{-x_k(t_n - s)} f(\underline{\widetilde{V}_s^N})\Bigg] \mathrm{d}s \\
&\quad + \int_{t_{n-1}}^{t_n} (t_n - s)^{-\alpha} \left(f(\underline{V_s^N}) - f(\underline{\widetilde{V}_s^N})\right) \mathrm{d}s \\
&\quad + \int_0^{t_{n-1}} \Bigg[(t_n - \underline{s})^{-\alpha} g(\underline{V_s^N}) \\
&\quad - \sum_{k=1}^{N_{\exp}} \omega_k e^{-x_k(t_n - \underline{s})} g(\underline{\widetilde{V}_s^N})\Bigg] \mathrm{d}B_s \\
&\quad + \int_{t_{n-1}}^{t_n} (t_n - \underline{s})^{-\alpha} \left(g(\underline{V_s^N}) - g(\underline{\widetilde{V}_s^N})\right) \mathrm{d}B_s\Bigg]^2 \\
&\le 2E\Bigg[\int_0^{t_{n-1}} \Bigg[(t_n - s)^{-\alpha} f(\underline{V_s^N}) \\
&\quad - \sum_{k=1}^{N_{\exp}} \omega_k e^{-x_k(t_n - s)} f(\underline{\widetilde{V}_s^N})\Bigg] \mathrm{d}s \\
&\quad + \int_{t_{n-1}}^{t_n} (t_n - s)^{-\alpha} \left(f(\underline{V_s^N}) - f(\underline{\widetilde{V}_s^N})\right) \mathrm{d}s\Bigg]^2 \\
&\quad + 2E\Bigg[\int_0^{t_{n-1}} \Bigg[(t_n - \underline{s})^{-\alpha} g(\underline{V_s^N}) \\
&\quad - \sum_{k=1}^{N_{\exp}} \omega_k e^{-x_k(t_n - \underline{s})} g(\underline{\widetilde{V}_s^N})\Bigg] \mathrm{d}B_s \\
&\quad + \int_{t_{n-1}}^{t_n} (t_n - \underline{s})^{-\alpha} \left(g(\underline{V_s^N}) - g(\underline{\widetilde{V}_s^N})\right) \mathrm{d}B_s\Bigg]^2 \\
&=: 2E\left[I_1^2 + I_2^2\right].
\end{aligned}$$

Using the mean inequality, Cauchy-Schwarz's inequality, the global Lipschitz condition and the linear growth condition, we derive that

$$\begin{aligned}
E[I_1]^2 &= E\Bigg[\int_0^{t_{n-1}} \left((t_n - s)^{-\alpha} - \sum_{k=1}^{N_{\exp}} \omega_k e^{-x_k(t_n - s)}\right) \\
&\quad \times f(\underline{\widetilde{V}_s^N}) \mathrm{d}s + \int_0^{t_n} (t_n - s)^{-\alpha} \\
&\quad \times \left(f(\underline{V_s^N}) - f(\underline{\widetilde{V}_s^N})\right) \mathrm{d}s\Bigg]^2 \\
&\le 2E\Bigg[\int_0^{t_{n-1}} \left((t_n - s)^{-\alpha} - \sum_{k=1}^{N_{\exp}} \omega_k e^{-x_k(t_n - s)}\right)^2 \\
&\quad \mathrm{d}s \int_0^{t_{n-1}} f^2(\underline{\widetilde{V}_s^N}) \mathrm{d}s\Bigg] + 2E\Bigg[\int_0^{t_n} (t_n - s)^{-\alpha} \mathrm{d}s \\
&\quad \times \int_0^{t_n} (t_n - s)^{-\alpha} \left(f(\underline{V_s^N}) - f(\underline{\widetilde{V}_s^N})\right)^2 \mathrm{d}s\Bigg] \\
&\le 2C_1 t_{n-1} \xi^2 \int_0^{t_{n-1}} E\left[1 + (\underline{\widetilde{V}_s^N})^2\right] \mathrm{d}s \\
&\quad + C_2 \int_0^{t_n} (t_n - s)^{-\alpha} E\left[\underline{V_s^N} - \underline{\widetilde{V}_s^N}\right]^2 \mathrm{d}s \\
&\le C_3 \left[\xi^2 + \int_0^{t_n} (t_n - s)^{-\alpha} E\left[\underline{V_s^N} - \underline{\widetilde{V}_s^N}\right]^2 \mathrm{d}s\right], \quad (37)
\end{aligned}$$

where the last inequality has used the conclusion (35). Using geometric inequality and Itô isometry gives that

$$\begin{aligned}
E[I_2]^2 &\le 2E \int_0^{t_{n-1}} \left[(t_n - \underline{s})^{-\alpha} - \sum_{k=1}^{N_{\exp}} \omega_k e^{-x_k(t_n - \underline{s})}\right]^2 g^2(\underline{\widetilde{V}_s^N}) \mathrm{d}s \\
&\quad + 2E \int_0^{t_n} (t_n - \underline{s})^{-2\alpha} \left[g(\underline{V_s^N}) - g(\underline{\widetilde{V}_s^N})\right]^2 \mathrm{d}s.
\end{aligned}$$

Then similarly to the estimation of (37), we obtain that

$$E[I_2]^2 \le C_4 \left[\xi^2 + \int_0^{t_n} (t_n - s)^{-2\alpha} E\left[\underline{V_s^N} - \underline{\widetilde{V}_s^N}\right]^2 \mathrm{d}s\right]. \quad (38)$$

Consequently using (37) and (38), we arrive at

$$\begin{aligned}
E[V_{t_n}^N - \widetilde{V}_{t_n}^N]^2 &\le C_5 \Bigg[\xi^2 + \int_0^{t_n} (t_n - s)^{-\alpha} E\left[\underline{V_s^N} - \underline{\widetilde{V}_s^N}\right]^2 \mathrm{d}s \\
&\quad + \int_0^{t_n} (t_n - s)^{-2\alpha} E\left[\underline{V_s^N} - \underline{\widetilde{V}_s^N}\right]^2 \mathrm{d}s\Bigg].
\end{aligned}$$

It then follows from the weakly singular Gronwall's inequality that

$$E[V_{t_n}^N - \widetilde{V}_{t_n}^N]^2 \le C_6 \xi^2. \quad (39)$$

Finally combining (33) in Lemma 4.1 and (39) with (36) completes the proof. ■

THEOREM 4.5 *Let $\widetilde{X}_{t_n}^N$ be the approximation to the solution X_{t_n} of (24) at time $t = t_n$ for $n = 0, 1, \ldots, N$ with $\widetilde{V}_{t_n}^N$ given by Algorithm 2. Then we have*

$$\sup_{1 \le n \le N} E[X_{t_n} - \widetilde{X}_{t_n}^N]^2 \le C(\tau^{\frac{1}{2} - \alpha} + \xi)^2,$$

where C is a positive constant which is independent of τ.

Proof Using the result of Theorem 4.4, the proof is similar to that of Theorem 4.2. ■

THEOREM 4.6 *Let $\widetilde{P}_0^N$ be the numerical option price based on paths simulation by the fast algorithm, namely, Algorithm 2 and P_0 the true option price at time 0. Then we have*

$$[P_0 - \widetilde{P}_0^N]^2 \leq C(\tau^{\frac{1}{2}-\alpha} + \xi)^2,$$

where C is a positive constant which is independent of τ.

Proof Using the result of Theorem 4.4, the proof is similar to that of Theorem 4.3. ■

REMARK 4.1 The convergence results of the fast algorithm (Algorithm 3) for the regime switching rough Heston model are similar to the above analyses. Thus the details are omitted here.

5. Numerical examples

In this section, we simulate the rough Heston models in El Euch and Rosenbaum (2018) (see also Callegaro *et al.* 2020) and Alfeus *et al.* (2019), i.e. (i) and (ii) stated in Section 1 using Algorithms 1, 2, 3 and 4 and compare the computational efficiency of these algorithms. All the tests are based on the use of MATLAB under Inter Core i7-7500U CPU (2.70 GHz).

EXAMPLE 5.1 Consider the rough volatility model (1)–(2) with f and g given by (3) and values of parameters from Callegaro *et al.* (2020):

$$S_0 = 100, \quad V_0 = 0.0392, \quad \rho = -0.681, \quad \kappa = 0.1,$$
$$\theta = 0.3156, \quad \epsilon = 0.331, \quad r = 0.$$

We test roughness of the paths by taking $\alpha = 0.02$ and $\alpha = 0.38$. We compare the fast algorithm with the Euler-Maruyama algorithm in the simulation of the paths and also compare with the Euler-Maruyama algorithm and the Adams algorithms of Callegaro *et al.* (2020) in the computation of the call options. We also compare the fast algorithm with the multi-factor approximation algorithm for rough Heston model. In the test, the absolute tolerance error for Algorithm 2 is taken as $\xi = 0.0001$ except where noted.

We first generate the paths for rough volatility models using number of the time-steps $N = 1000$ on time interval $[0, T]$. As shown in figure 1, the paths with $\alpha = 0.38$ is much rougher than that with $\alpha = 0.02$. To be noted that, if α is approaching

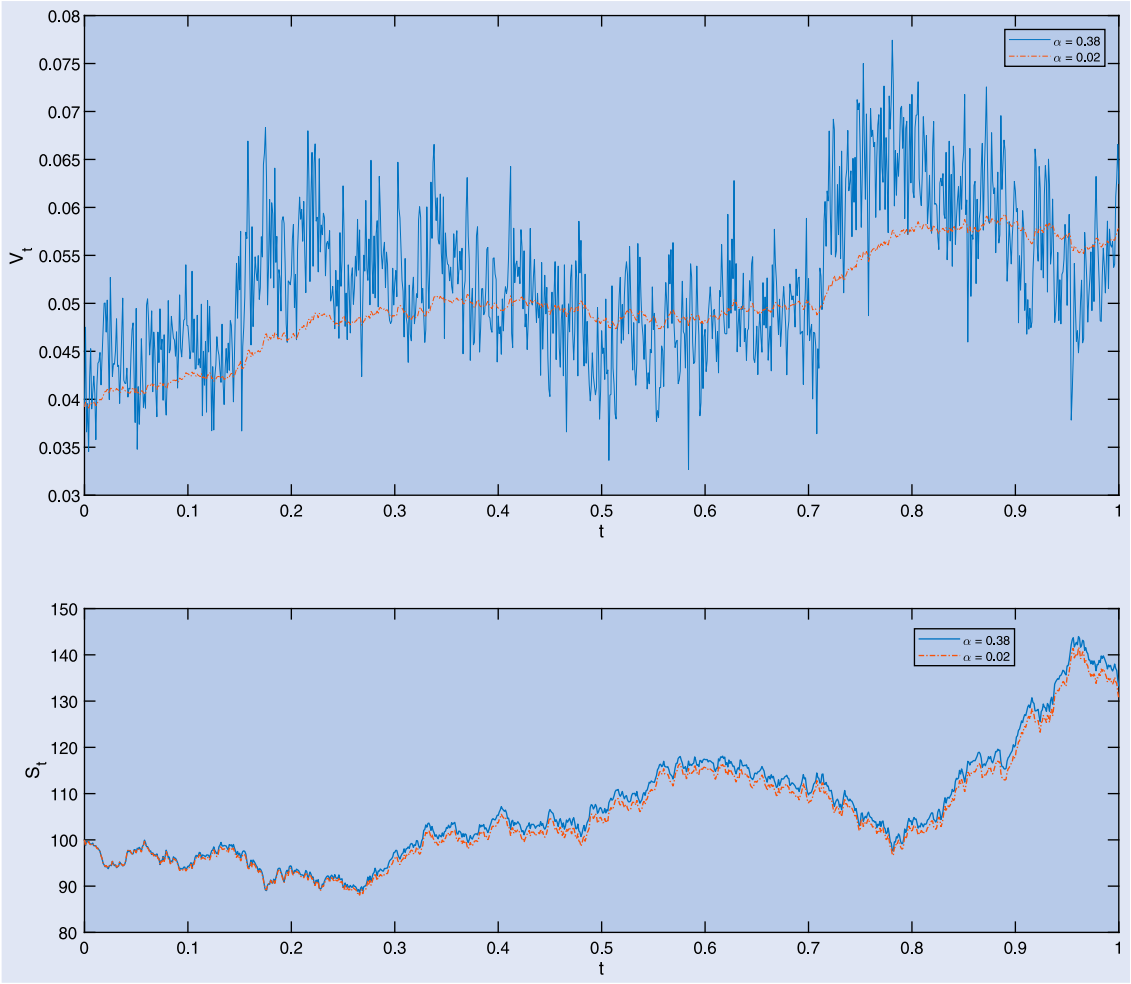

Figure 1. Paths of rough volatility model in Example 5.1: the upper figure is for the squared volatility and the lower one asset price.

to zero, the rough volatility model is close to the classical Heston model. Obviously α measures the roughness of the paths and the roughness increases with the value of α.

We now test the convergence of fast algorithms for simulation of the rough volatility model by calculating the root mean square errors

$$\mathrm{err}_N(V_T) := \sqrt{E\left[V_T^N - V_T\right]^2}, \quad \mathrm{err}_N(S_T) := \sqrt{E\left[S_T^N - S_T\right]^2},$$

with number of paths $M = 10\,000$ and $T = 1$. The computational value using $N = 64\,000$ is taken as the benchmark value, and in order to minimize the influence of the approximation of kernel function on the error the absolute tolerance error for Algorithm 2 is taken as $\xi = 10^{-12}$. We plot the errors in log-log scales and the negative slope of the line is the order of convergence. From figure 2, we observe that the order of convergence is approximately equal to $1/2 - \alpha$ with $\alpha \in (0, 1/2)$.

In the following table 1, we compare the CPU time between the Euler-Maruyama (E-M) algorithm and the fast algorithm (Algorithm 2) for simulation of the squared volatility process with $\alpha = 0.38$. According to table 1, we can see that Algorithm 2 is much faster than the Euler-Maruyama algorithm and the increasing speed of CPU time for Algorithm 2 is almost proportional to that for N which is consistent to the estimated complexity $O(N \log N)$, meanwhile the CPU time for the Euler-Maruyama algorithm is increasing with double speed of N which also agrees with the estimated complexity $O(N^2)$.

Next, we compare the fast algorithm with the Euler-Maruyama algorithm and the Adams algorithm of Callegaro *et al.* (2020) in the computation of the call options with $T = 2$. For the Euler-Maruyama algorithm and the fast algorithm, we take the number of time-steps $N = 250$ and number of paths $M = 100\,000$. The confidence intervals corresponding to 95% confidence for Monte Carlo based on the simulation are given in table 2. The results with Adams algorithm are from Callegaro *et al.* (2020). From table 2, it verifies again that the fast algorithm is the fastest one.

In the final, we compare the fast algorithm (Algorithm 2) with multi-factor approximation algorithm (Algorithm 4) in regards to the computational time under the same root mean square errors with number of paths $M = 10\,000$ and $T = 1$. In the computation of Algorithm 2, the absolute tolerance error for the kernel approximation is taken as $\xi = 0.1$. N_{exp} is estimated by (13) as $N_{\mathrm{exp}} = 10$ for $N = 2000$, $N_{\mathrm{exp}} = 11$ for $N = 4000$, 8000, and $N_{\mathrm{exp}} = 12$ for $N = 16\,000, 32\,000$. Observed from table 3 to keep the accuracy of path simulation by Algorithm 4, $\widetilde{N}_{\mathrm{exp}}$ has to be increased with N,

Table 1. Comparisons of simulation time between Euler–Maruyama (E-M) algorithm and the fast algorithm (Algorithm 2) for Example 5.1.

	CPU time (ms)			
	$T = 1$		$T = 100$	
N	Fast algorithm	E-M algorithm	Fast algorithm	E-M algorithm
250	1.51	56.37	1.54	57.18
500	3.19	223.82	3.18	236.39
1000	6.66	891.43	6.40	901.89
2000	13.21	3489.08	13.01	3529.03
4000	26.13	14 636.64	25.84	14 950.11
8000	53.80	57 283.13	51.66	57 814.46

Table 2. European call option price and computational time (CPU time (s)) of Euler-Maruyama (E-M) algorithm, fast algorithm and Adams algorithm for Example 5.1.

K	Adams algorithm	E-M algorithm	Fast algorithm
80	25.43	25.48	25.48
		[25.32, 25.65]	[25.32, 25.64]
90	19.28	19.34	19.34
		[19.19, 19.49]	[19.19, 19.49]
100	14.33	14.37	14.37
		[14.24, 14.50]	[14.24, 14.51]
110	10.45	10.48	10.49
		[10.37, 10.59]	[10.38, 10.61]
120	7.50	7.53	7.55
		[7.43, 7.63]	[7.45, 7.64]
CPU time (s)	246	5732	157

Note: The second row gives the confidence interval.

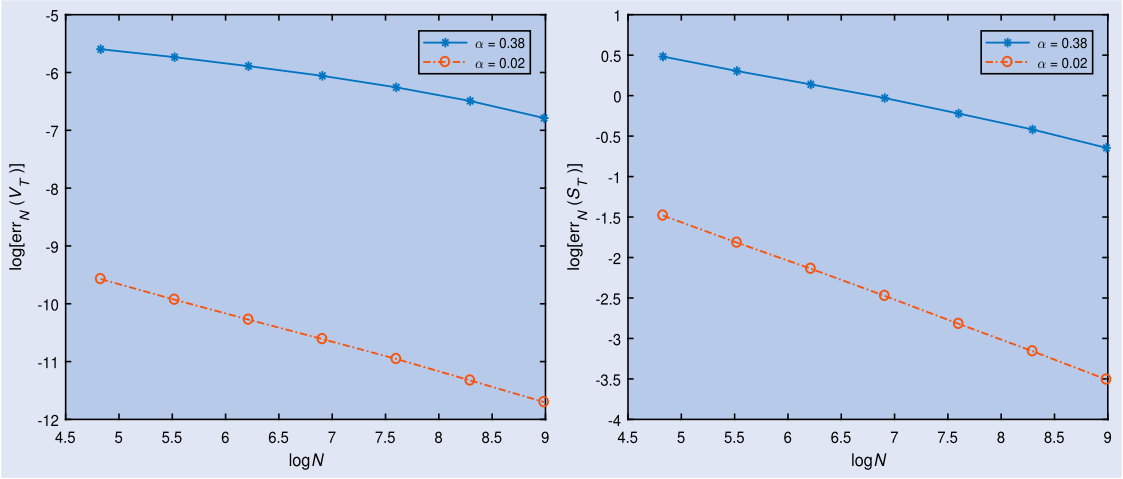

Figure 2. Errors for the fast algorithms for Example 5.1: the left figure is for squared volatility and the right one asset price.

Table 3. Comparison of the fast algorithm (Algorithm 2) and multi-factor approximation algorithm (Algorithm 4) for Example 5.1.

N				2000	4000	8000	16 000	32 000
Algorithm 2	$N_{\exp}$	10,11,12	Error	6.61e − 3	5.93e − 3	5.35e − 3	4.68e − 3	4.12e − 3
			Time (s)	103.8	205.9	404.8	821.2	1632.1
Algorithm 4	$\widetilde{N}_{\exp}$	9	Error	8.12e − 3	8.12e − 3	8.12e − 3	8.12e − 3	8.12e − 3
			Time (s)	19.1	35.6	69.0	134.7	266.3
		16	Error	7.23e − 3	7.23e − 3	7.23e − 3	7.23e − 3	7.23e − 3
			Time (s)	31.1	58.2	121.5	235.9	457.8
		27	Error	6.61e − 3	6.61e − 3	6.61e − 3	6.61e − 3	6.61e − 3
			Time (s)	55.1	105.0	204.5	405.5	811.7
		57	Error	–	5.93e − 3	5.93e − 3	5.92e − 3	5.92e − 3
			Time (s)	–	218.2	434.7	871.3	1733.0
		126	Error	–	–	5.35e − 3	5.35e − 3	5.35e − 3
			Time (s)	–	–	961.8	1925.7	3816.9
		400	Error	–	–	–	4.68e − 3	4.68e − 3
			Time (s)	–	–	–	6056.0	12 164.4
		1200	Error	–	–	–	–	4.12e − 3
			Time (s)	–	–	–	–	38 788.4

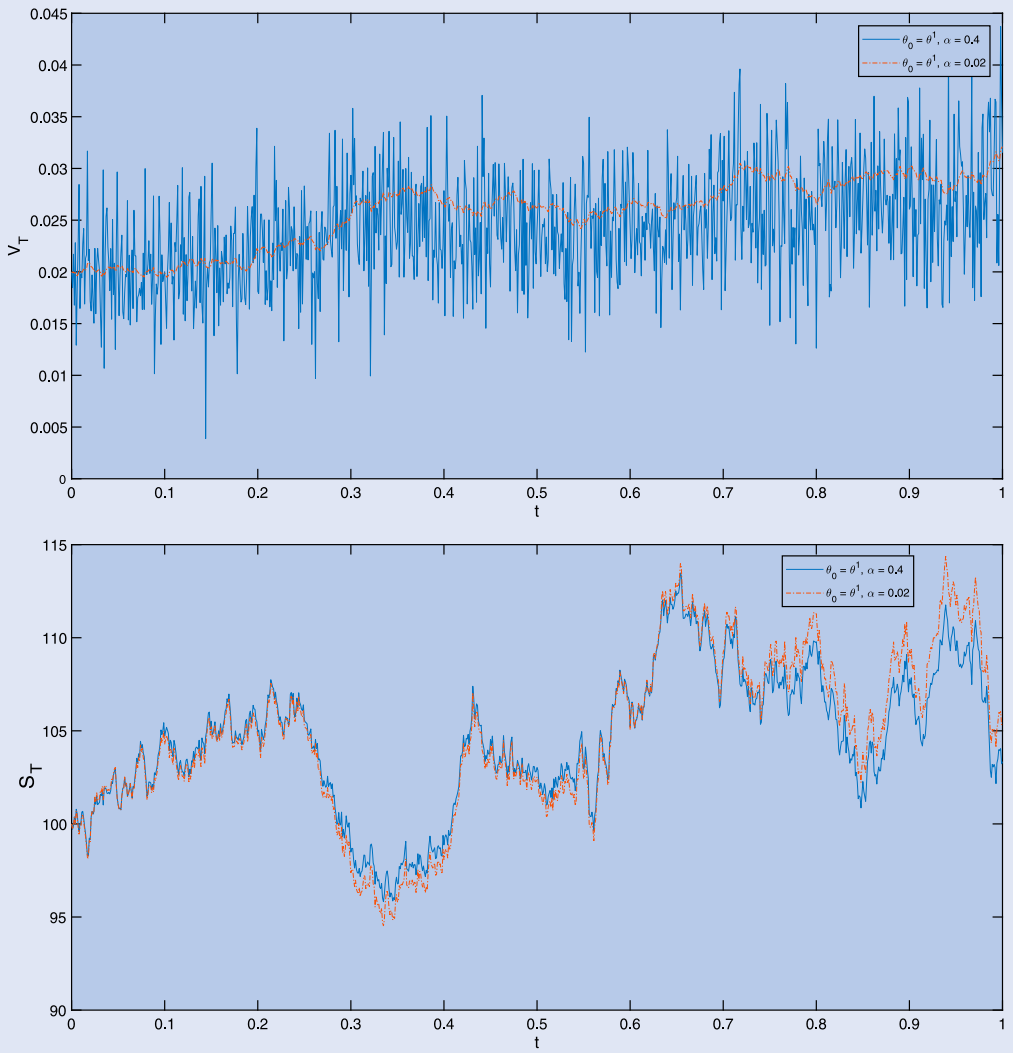

Figure 3. Paths of regime switching rough volatility model for Example 5.2 with initial regime state 1: the upper figure is for the squared volatility and the lower one asset price.

and the increasing speed is much faster than that of $N_{\exp}$ for Algorithm 2. Furthermore when $N = 4000$, the complexity (CPU time) of Algorithm 4 exceeds that of Algorithm 2 and the difference becomes larger and larger with N increasing. The numerical results are consistent with the discussions in Remark 3.1.

EXAMPLE 5.2 Consider the fast simulation of the regime switching rough Heston model of Alfeus *et al.* (2019), i.e. (1)–(2) with f and g given by (3) and the mean-reversion level θ by (4). The values of parameters are as follows:

$$S_0 = 100, \quad V_0 = 0.02, \quad \rho = -0.5, \quad r = 0.05, \quad \kappa = 0.3,$$
$$\epsilon = \frac{0.04}{\kappa}, \quad [\theta^1, \theta^2] = [0.025, 0.075].$$

and the generation matrix of the regime switching

$$Q = \begin{bmatrix} -1 & 1 \\ 0.5 & -0.5 \end{bmatrix}.$$

We test roughness of the paths by taking $\alpha = 0.02$ and $\alpha = 0.40$. We compare the fast algorithm with the Euler-Maruyama algorithm in the simulation of the paths and the computation of the call options. We also compare the fast algorithm with the multi-factor approximation algorithm for regime switching rough Heston model. In the tests, the absolute tolerance error for Algorithm 3 is taken as $\xi = 0.0001$ except where noted.

As in Alfeus *et al.* (2019), the generation of the regime switching θ_s with $k \geq 2$ states is given as follows:

$$\theta_s = \sum_{m=0}^{M_n - 1} 1_{[s_m^n(\omega), s_{m+1}^n(\omega)) \cap [t_{n-1}, t_n)}(s) \vartheta_m, \quad n = 1, \ldots, N,$$

where s_m^n is regime switching time and $s_{m+1}^n(\omega) = s_m^n + \varrho_m$, where the waiting time ϱ_m between switching time s_m^n and s_{m+1}^n is successively drawn from an exponential distribution with parameter $-q_{i,i}$ which is the negative diagonal element of the regime switching generation matrix Q whose elements satisfy $q_{i,i} < 0, q_{i,j} > 0\,(i \neq j)$ and $\sum_{j=1}^{k} q_{i,j} = 0$, that is

$$\varrho_m \sim \text{Exp}\left(-q_{i,i}\right),$$

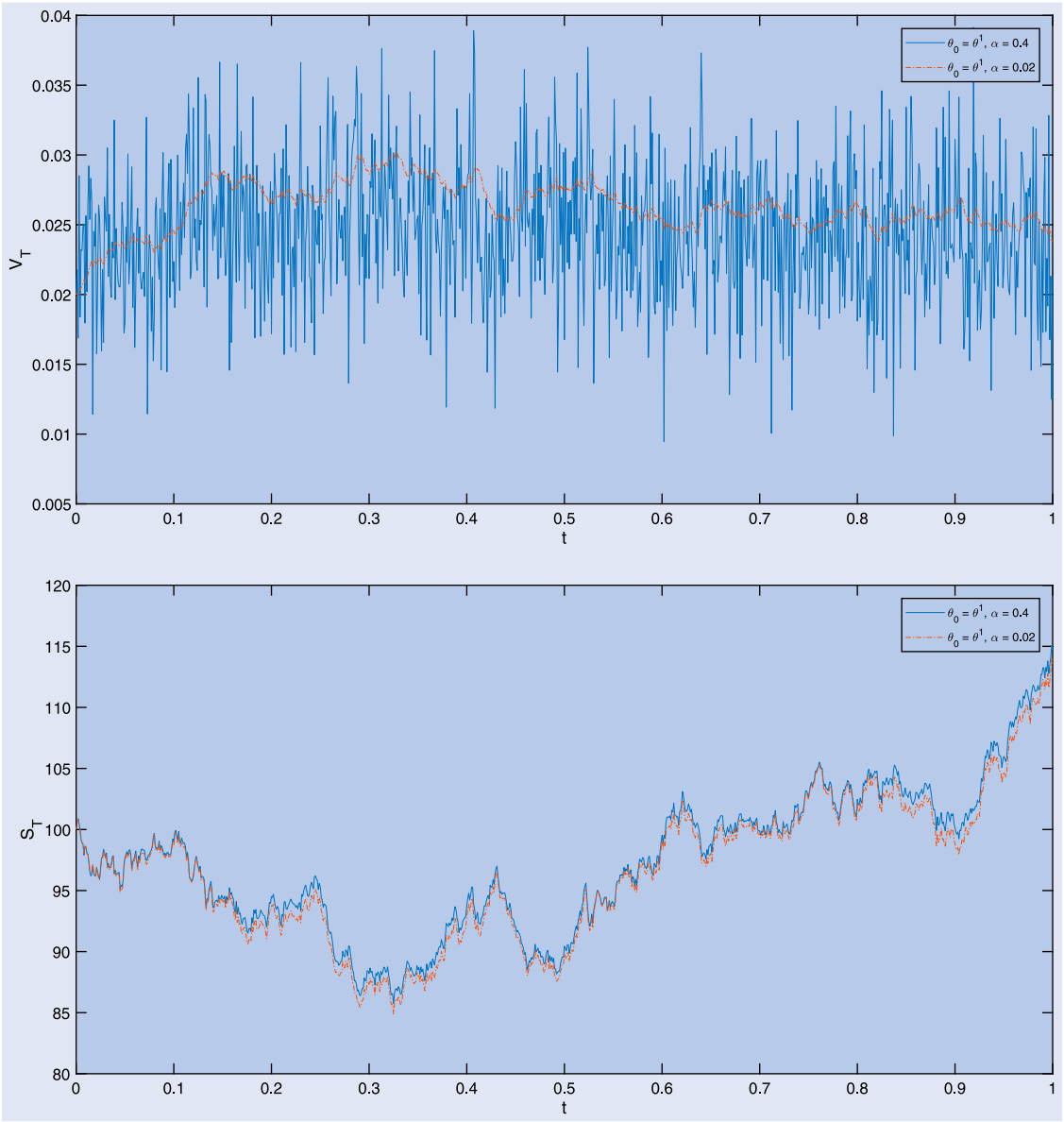

Figure 4. Paths of regime switching rough volatility model for Example 5.2 with initial regime state 2: the upper figure is for the squared volatility and the lower one asset price.

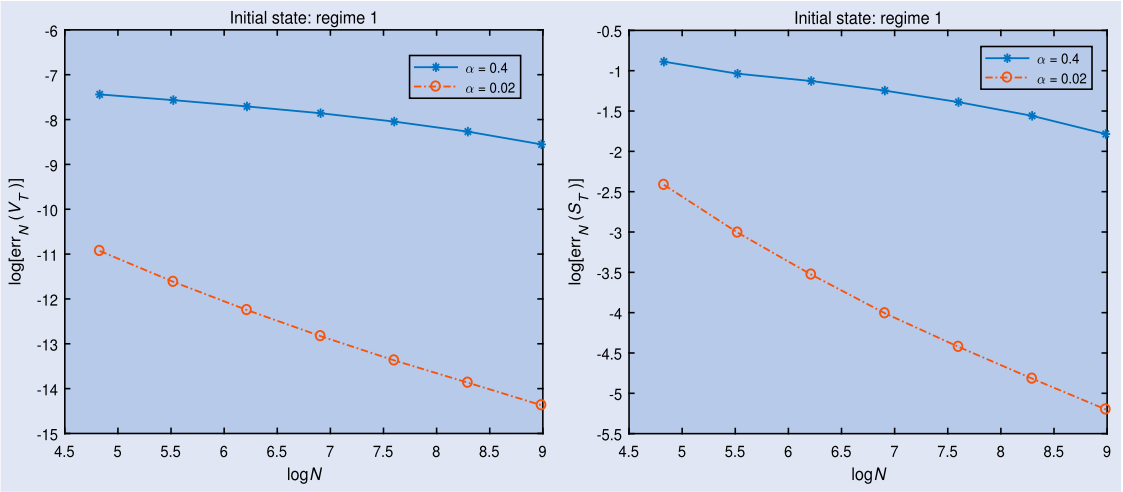

Figure 5. Errors for the fast algorithms for Example 5.2 with initial regime state 1: the left figure is for squared volatility and the right one asset price.

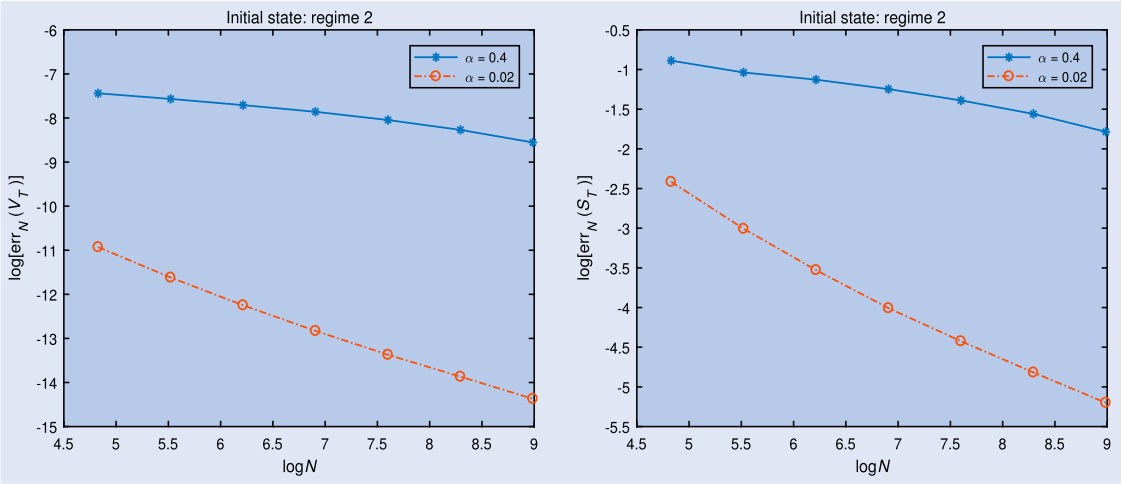

Figure 6. Errors for the fast algorithms for Example 5.2 with initial regime state 2: the left figure is for squared volatility and the right one asset price.

Table 4. Comparisons of simulation time between Euler–Maruyama (E-M) algorithm and the fast algorithm (Algorithm 3) for Example 5.2.

	CPU time (ms)			
	$T = 1$		$T = 100$	
N	Fast algorithm	E-M algorithm	Fast algorithm	E-M algorithm
250	1.98	54.91	2.13	62.97
500	3.85	218.98	4.12	255.68
1000	8.01	873.94	7.90	960.49
2000	16.85	3479.81	16.16	3714.47
4000	34.68	14 059.28	31.50	14 459.72
8000	66.95	56 429.14	65.01	57 241.94

moreover ϑ_m is a discrete random variable with values in the set $\{\theta^1, \ldots, \theta^k\}$ and conditional probability

$$\text{Prob}[\vartheta_{m+1} = \theta^j \mid \vartheta_m = \theta^i] = -\frac{q_{i,j}}{q_{i,i}}, \quad j = 1, \ldots, k.$$

We first generate the paths for regime switching rough volatility models with the number of the time-steps $N = 1000$ on time interval $[0, 1]$. Obviously, the paths with $\alpha = 0.4$ is much rougher than that with $\alpha = 0.02$, as shown in figures 3 and 4.

Next, we test the convergence of fast algorithms for simulation of the regime switching rough volatility model under $\xi = 10^{-12}$ by plotting the errors in log-log scales. From figures 5 and 6, we observe that the negative slope of the lines is approximately equal to $1/2 - \alpha$ which is just the order of convergence.

We then compare the CPU time between the Euler-Maruyama (E-M) algorithm and the fast algorithm (Algorithm 3) with $\alpha = 0.4$. From table 4, we can see that Algorithm 3 is much faster than the Euler-Maruyama algorithm and the complexity of both algorithms are consistent to the theoretical estimation.

We now compare the fast algorithm with the Euler-Maruyama algorithm of Alfeus *et al.* (2019) in the computation of the call options with $\alpha = 0.4$. For the Euler-Maruyama algorithm and the fast algorithm, we take the number of time-steps $N = 250$ and number of paths $M = 100\,000$. The confidence intervals corresponding to 95% confidence for Monte Carlo based on the simulation are given in table 5.

Table 6. Comparison of the fast algorithm (Algorithm 3) with the multi-factor approximation algorithm (Algorithm 5) for simulation of regime switching rough volatility model (Example 5.2).

N				2000	4000	8000	16 000	32 000
Algorithm 3	N_{exp}	10,11,12	Error	5.99e − 3	5.43e − 3	5.00e − 3	4.43e − 3	3.97e − 3
			Time (s)	140.6	273.2	548.3	1106.7	2174.7
Algorithm 5	$\widetilde{N}_{\text{exp}}$	5	Error	5.95e − 3	5.95e − 3	5.95e − 3	5.95e − 3	5.95e − 3
			Time (s)	44.2	91.3	175.4	344.8	679.2
		17	Error	–	5.43e − 3	5.43e − 3	5.43e − 3	5.43e − 3
			Time (s)	–	152.5	323.2	640.4	1281.3
		51	Error	–	–	4.99e − 3	4.99e − 3	4.99e − 3
			Time (s)	–	–	707.7	1498.0	2970.8
		240	Error	–	–	–	4.41e − 3	4.41e − 3
			Time (s)	–	–	–	6057.2	13 415.9
		800	Error	–	–	–	–	3.99e − 3
			Time (s)	–	–	–	–	45 915.9

Table 5. European call option price and computational time (CPU time (s)) of Euler-Maruyama (E-M) algorithm and fast algorithm for Example 5.2.

		$T = 1$	
Initial state	K	E-M	Fast
Regime 1	80	24.12	24.06
		[24.05, 25.19]	[23.99, 24.14]
	90	15.47	15.37
		[15.40, 15.53]	[15.30, 15.44]
	100	8.42	8.39
		[8.36, 8.47]	[8.33, 8.44]
	110	3.73	3.83
		[3.69, 3.76]	[3.79, 3.87]
CPU time (s)		5744.7	185.5
Regime 2	80	24.29	24.22
		[24.21, 24.37]	[24.13, 24.30]
	90	15.92	15.83
		[15.85, 16.00]	[15.76, 15.91]
	100	9.17	9.15
		[9.11, 9.23]	[9.09, 9.21]
	110	4.55	4.64
		[4.50, 4.59]	[4.59, 4.68]
CPU time (s)		5826.2	187.1

Note: The second row gives the confidence interval.

From table 5, it verifies again that the fast algorithm is much faster than the Euler-Maruyama algorithm.

Now we compare the fast algorithm (Algorithm 3) with multi-factor approximation algorithm (Algorithm 5) for the regime switching rough volatility model in regards to the computational time under the same root mean square errors with number of paths $M = 10\,000$ and $T = 1$. In the computation of Algorithm 3, the absolute tolerance error for the kernel approximation is taken as $\xi = 0.1$ and N_{exp} is estimated as that in table 3. Observed from table 6, Algorithm 3 is much faster than that of Algorithm 5. Moreover the numerics in table 6 confirm the discussions in Remark 3.1.

6. Conclusions

The rough volatility model has been increasingly popular as it fits the financial market better as shown by the empirical studies in Gatheral *et al.* (2018). However the simulation of the paths with the classical Euler-Maruyama algorithm is not efficient as the cost of simulation is too expensive with a complexity of $O(N^2)$ for simulating one path. This paper develops a fast algorithm for simulation of rough volatility models using the approximation of the weakly singular kernel by a sum of exponential functions. The complexity of the fast algorithm for simulation of one path is about $O(N \log N)$ for maturity time $T \gg 1$ and $O(N \log^2 N)$ for $T \approx 1$. This fast algorithm thus greatly increases the efficiency of the simulation. Both the rough volatility model of El Euch and Rosenbaum (2018) and the regime switching rough volatility model of Alfeus *et al.* (2019) are simulated, and a variety of examples confirm that the fast algorithm is much more efficient. Moreover, the fast algorithm are shown to be faster than the multi-factor approximation algorithms which are developed by the discretization of the multi-factor Heston models in Abi Jaber and El Euch (2019) and Abi Jaber (2019) when the number of time steps is large.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

The work was supported by National Natural Science Foundation of China (Grant No. 12071373) and the Fundamental Research Funds for the Central Universities, P.R. China (JBK1805001).

References

Abi Jaber, E., Lifting the Heston model. *Quant. Finance*, 2019, **19**, 1995–2013.

Abi Jaber, E. and El Euch, O., Multifactor approximation of rough volatility models. *SIAM J. Financ. Math.*, 2019, **10**, 309–349.

Alfeus, M., Overbeck, L. and Schlögl, E., Regime switching rough Heston model. *J. Futures Mark.*, 2019, **39**, 538–552.

Andersen, L., Simple and efficient simulation of the Heston stochastic volatility model. *J. Comput. Finance*, 2008, **11**, 1–42.

Callegaro, G., Grasselli, M. and Pagès, G., Fast hybrid schemes for fractional Riccati equations (Rough is not so tough). *Math. Oper. Res.*, 2020. Available at https://doi.org/10.1287/moor.2020.1054.

Coeurjolly, J.-F., Simulation and identification of the fractional Brownian motion: A bibliographical and comparative study. *J. Stat. Softw.*, 2000, **5**, 1–53.

Dai, X.J. and Xiao, A.G., Lévy-driven stochastic Volterra integral equations with doubly singular kernels: Existence, uniqueness, and a fast EM method. *Adv. Comput. Math.*, 2020, **46**, 29–51.

Dastgerdi, M.V. and Bastani, A.F., Solving parametric fractional differential equations arising from the rough Heston model using quasi-linearization and spectral collocation. *SIAM J. Financ. Math.*, 2020, **11**, 1063–1097.

El Euch, O. and Rosenbaum, M., Perfect hedging in rough Heston models. *Ann. Appl. Probab.*, 2018, **28**, 3813–3856.

El Euch, O. and Rosenbaum, M., The characteristic function of rough Heston models. *Math. Finance*, 2018, **29**, 3–38.

El Euch, O., Fukasawa, M. and Rosenbaum, M., The microstructural foundations of leverage effect and rough volatility. *Finance Stoch.*, 2018, **22**, 241–280.

Gatheral, J., Jaisson, T. and Rosenbaum, M., Volatility is rough. *Quant. Finance*, 2018, **18**, 933–949.

Guennoun, H., Jacquier, A., Roome, P. and Shi, F., Asymptotic behavior of the fractional Heston model. *SIAM J. Financ. Math.*, 2018, **9**, 1017–1045.

Heston, S.L., A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Rev. Financ. Stud.*, 1993, **6**, 327–343.

Jaisson, T. and Rosenbaum, M., Limit theorems for nearly unstable Hawkes processes. *Ann. Appl. Probab.*, 2015, **25**, 600–631.

Jaisson, T. and Rosenbaum, M., Rough fractional diffusions as scaling limits of nearly unstable heavy tailed Hawkes processes. *Ann. Appl. Probab.*, 2016, **26**, 2860–2882.

Jiang, S.D., Zhang, J.W., Zhang, Q. and Zhang, Z.M., Fast evaluation of the Caputo fractional derivative and its applications to fractional diffusion equations. *Commun. Comput. Phys.*, 2015, **21**, 650–678.

Li, M., Huang, C.M. and Hu, Y.Z., Numerical methods for stochastic Volterra integral equations with weakly singular kernels, 2020. Available at arXiv:2004.04916.

Liang, H., Yang, Z.W. and Gao, J.F., Strong superconvergence of the Euler-Maruyama method for linear stochastic Volterra integral equations. *J. Comput. Appl. Math.*, 2017, **317**, 447–457.

Olver, F.W.J., Lozier, D.W., Boisvert, R.F. and Clark, C.W., *NIST Handbook of Mathematical Functions*, 2010 (Cambridge University Press: New York).

Shreve, S.E., *Stochastic Calculus for Finance II: Continuous-Time Models*, 2004 (Springer-Verlag: New York).

Wood, A. and Chan, G., Simulation of stationary Gaussian processes in $[0, 1]^d$. *J. Comput. Graph. Statist.*, 1994, **3**, 409–432.