Hierarchical adaptive sparse grids for option pricing under the rough Bergomi model

Christian Bayer* Chiheb Ben Hammouda† Raul Tempone‡§

November 26, 2018

Abstract

The rough Bergomi (rBergomi) model, introduced recently in [4], is a promising rough volatility model in quantitative finance. This new model exhibits consistent results with the empirical fact of implied volatility surfaces being essentially time-invariant, and ability to capture the term structure of skew observed in equity markets. In the absence of analytical European option pricing methods for the model, and due to the non-Markovian nature of the fractional driver, the prevalent option is to use Monte Carlo (MC) simulation for pricing. Despite the recent advances in MC method in this context, pricing under the rBergomi model is still a time consuming task. To overcome this issue, we design a novel, alternative, hierarchical approach, based on adaptive sparse grids quadrature, specifically multi-index stochastic collocation (MISC) as in [19], coupled with Brownian bridge construction and Richardson extrapolation. By uncovering the available regularity, our hierarchical method demonstrates substantial computational gains with respect to the standard MC method, assuming a sufficiently small error tolerance in the price estimates, across different parameter constellations, even for very small values of the Hurst parameter. Our work opens a new research direction in this field, to investigate the performance of other methods than Monte Carlo, for pricing and calibrating under the rBergomi model.

Keywords Rough volatility, Monte Carlo, multi-index stochastic collocation, Brownian bridge construction, Richardson extrapolation.

2010 Mathematics Subject Classification 91G60, 91G20, 65C05, 65D30, 65D32.

1 Introduction

Modeling the volatility to be stochastic, rather than deterministic as in the Black-Scholes model, enabled quantitative analysts to explain certain phenomena observed in option price data, in particular the implied volatility smile. However, this family of models has a main drawback consisting in failing to capture the true steepness of the implied volatility smile close to maturity. To overcome

*Weierstrass Institute for Applied Analysis and Stochastics (WIAS), Berlin, Germany.

†King Abdullah University of Science and Technology (KAUST), Computer, Electrical and Mathematical Sciences & Engineering Division (CEMSE), Thuwal 23955 − 6900, Saudi Arabia (chiheb.benhammouda@kaust.edu.sa).

‡King Abdullah University of Science and Technology (KAUST), Computer, Electrical and Mathematical Sciences & Engineering Division (CEMSE), Thuwal 23955 − 6900, Saudi Arabia (raul.tempone@kaust.edu.sa).

§Alexander von Humboldt Professor in Mathematics of Uncertainty Quantification, RWTH Aachen University, Germany.

this undesired feature, one may add jumps to stock price models, for instance modeling the stock price process as an exponential Lévy process. Unfortunately, the presence of jumps in stock price processes remains controversial [12, 3], and is one of the major criticisms of such models.

Motivated by the statistical analysis of realized volatility by Gatheral, Jaisson and Rosenbaum [17] and the theoretical results on implied volatility [2, 15], rough stochastic volatility has emerged as a new paradigm in quantitative finance, overcoming the observed limitations of diffusive stochastic volatility models. In these models, the trajectories of the volatility has lower Hölder regularity than those of the standard Brownian motion [17, 4]. In fact, they are based on fractional Brownian motion (fBm), which is a centered Gaussian process, whose covariance structure depends on the so called Hurst parameter, H (we refer to [23, 13, 10] for more details regarding the fBm processes). In the rough volatility case, where $0 < H < 1/2$, the fBm has negatively correlated increments and rough sample paths. Gatheral, Jaisson, and Rosenbaum [17] empirically demonstrate the advantages of such models. For instance, they show that the log-volatility in practice has a similar behavior to fBm with the Hurst exponent $H \approx 0.1$ at any reasonable time scale (see also [16]). These results were confirmed by Bennedsen, Lunde and Pakkanen [7], who studied over a thousand individual US equities and showed that the Hurst parameter H lies in $(0, 1/2)$ for each equity. Other works [17, 7, 4] showed further benefits of such rough volatility models over standard stochastic volatility ones, in terms of explaining crucial phenomena observed in financial markets.

One of the recent rough volatility models is the rough Bergomi (rBergomi) model, developed by Bayer, Friz and Gatheral [4]. This model showed consistent behavior with the stylized fact of implied volatility surfaces being essentially time-invariant. It was also proven that this model is able to capture the term structure of skew observed in equity markets. The construction of the rBergomi model was performed by moving from physical to pricing measure and simulating prices under that model to fit the implied volatility surface well in the case of the S&P 500 index with few parameters. The model may be seen as a non-Markovian extension of the Bergomi variance curve model [9].

Despite the promising features of the rBergomi model, pricing and hedging under such a model still constitutes a challenging, and time consuming task due to the non-Markovian nature of the fractional driver. In fact, the standard numerical pricing methods, such as: PDE discretization schemes, asymptotic expansions and transform methods, being efficient in the case of diffusion, are not easily carried over to the rough setting. Furthermore, due to the lack of Markovianity or affine structure, conventional analytical pricing methods do not apply. To the best of our knowledge, the only prevalent method for pricing options under such models is Monte Carlo (MC) simulation. In particular, recent advances in simulation methods for the rBergomi model and different variants of pricing methods based on MC under such a model have been proposed in [4, 5, 24, 8, 21]. For instance, in [24], the authors employ a novel composition of variance reduction methods. When pricing under the rBergomi model, they got substantial computational gains over the standard MC method. More analytical understanding of option pricing and implied volatility under this model have been achieved in [22, 6, 14]. It is interesting to note that due to the poor behavior of the strong error, that is of order of the Hurst parameter H [27], hierarchical variance reduction methods, such as Multi-level Monte Carlo (MLMC), are inefficient in this context.

Despite the recent advances in MC method, pricing under the rBergomi model is still a time consuming task. To overcome this issue, we design, in this work, a novel fast option pricer, based on a hierarchical adaptive sparse grids quadrature, specifically multi-index stochastic collocation (MISC), coupled with Brownian bridge construction and Richardson extrapolation, for options

whose underlyings follow the rBergomi model. In order to use adaptive sparse grids quadrature for our purposes, we solve two main issues, that constitute the two stages of our new designed method. In the first stage, we smoothen the integrand by using the conditional expectation as was proposed in [28], in the context of Markovian stochastic volatility models. In a second stage, we apply MISC, to solve the integration problem. In this stage, we apply two transformations before using the MISC method, in order to overcome the issue of facing a high dimensional integrand due to the discretization scheme used for simulating the rBergomi dynamics. Given that MISC profits from anisotropy, the first transformation consists of applying a hierarchical path generation method, based on Brownian bridge (Bb) construction, with the aim of reducing the effective dimension. The second transformation consists of applying Richardson extrapolation to reduce the bias, which in turn reduces the needed number of time steps in the coarsest level to achieve a certain error tolerance, and therefore, the maximum number of dimensions needed for the integration problem. We emphasize that we are interested in the pre-asymptotic regime (small number of time steps), and the use of Richardson extrapolation is justified by our observed experimental results in that regime, which, in particular, have an order one of convergence for the weak error. Although, we do not claim that the observed rates will scale well in the asymptotic regime, we observed that the pre-asymptotic regime is enough to get sufficiently accurate estimates for the option prices.

Compared to the works that we mentioned above, mainly [24], our first contribution is that we design a novel alternative approach based on adaptive sparse grid quadrature. Given that the only prevalent option in this context is to use different variants of the MC method, our work opens a new research direction in this field, to investigate the performance of other methods than MC, for pricing and calibrating under the rBergomi model. Our second contribution is that we reduce the computational cost, through variance reduction, by using the conditional expectation as in [24] but also through bias reduction through using Richardson extrapolation. Finally, assuming one targets price estimates with a sufficiently small error tolerance, our proposed method demonstrated substantial computational gains over the standard MC method, even for very small values of the Hurst parameter, H. We show these gains through our numerical experiments for different parameter constellations. We insist that we do not claim that these gains will hold, in the asymptotic regime, that is for higher accuracy requirements. Furthermore, in this work, we limit ourselves to comparing our novel proposed method against the standard MC. A more systematic comparison against the variant of MC proposed in [24] can be carried out but this is left for a future work. Another eventual direction of research may also investigate the performance of quasi-Monte Carlo (QMC) for such problems.

The outline of this paper is as follows: We start in Section 2 by introducing the pricing framework that we consider. We provide some details about the rBergomi model, option pricing under this model and the simulation scheme used to simulate asset prices following the rBergomi dynamics. Then, in Section 3, we explain the different building blocks that constitute our proposed method, which are basically MISC, Brownian bridge construction, and Richardson extrapolation. Finally, in Section 4, we show the results obtained through the different numerical experiments, conducted across different parameter constellations for the rBergomi model. The reported results show the high potential of our proposed method in this context.

2 Problem setting

In this section, we introduce the pricing framework that we are considering in this work. We start by giving some details for the rBergomi model proposed in [4]. We then derive the formula of the price of a European call option under the rBergomi model, in Section 2.2. This section corresponds to the first stage of our approach, that is the analytical smoothing step. Finally, we explain some details about the hybrid scheme that we use to simulate the dynamics of asset prices under the rBergomi model.

2.1 The rBergomi model

We use the rBergomi model for the price process S_t as defined in [4], normalized to $r = 0$, which is defined by

$$\begin{aligned} dS_t &= \sqrt{v_t} S_t dZ_t, \\ v_t &= \xi_0(t) \exp\left(\eta \widetilde{W}_t^H - \frac{1}{2}\eta^2 t^{2H}\right), \end{aligned} \tag{2.1}$$

where the Hurst parameter $0 < H < 1$ and $\eta > 0$. We refer to v_t as the variance process, and $\xi_0(t) = \mathrm{E}[v_t]$ is the forward variance curve. Here, $\widetilde{W}^H$ is a certain Riemann-Liouville fBm process, defined by

$$\widetilde{W}_t^H = \int_0^t K^H(t,s) dW_s^1, \quad t \geq 0, \tag{2.2}$$

where the kernel $K^H : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}_+$ is

$$K^H(t,s) = \sqrt{2H}(t-s)^{H-1/2}, \quad \forall\, 0 \leq s \leq t.$$

$\widetilde{W}^H$ is a centered, locally $(H-\epsilon)$- Hölder continuous, Gaussian process with $\mathrm{Var}\left[\widetilde{W}_t^H\right] = t^{2H}$, and a dependence structure defined by

$$\mathrm{E}\left[\widetilde{W}_u^H \widetilde{W}_v^H\right] = u^{2H} G\left(\frac{v}{u}\right), \quad v > u,$$

where for $x \geq 1$ and $\gamma = \frac{1}{2} - H$

$$G(x) = 2H \int_0^1 \frac{ds}{(1-s)^\gamma (x-s)^\gamma}.$$

In (2.1), W^1, Z denote two *correlated* standard Brownian motions with correlation $\rho \in [-1, 1]$, so that we can write

$$Z = \rho W^1 + \overline{\rho} W^\perp = \rho W^1 + \sqrt{1-\rho^2} W^\perp,$$

where $(W^1, W^\perp)$ are two independent standard Brownian motions. Therefore, the solution to (2.1), with $S(0) = S_0$, can be written as

$$S_t = S_0 \exp\left(\int_0^t \sqrt{v(s)} dZ(s) - \frac{1}{2}\int_0^t v(s) ds\right), \quad S_0 > 0$$

$$v_u = \xi_0(u) \exp\left(\eta \widetilde{W}_u^H - \frac{\eta^2}{2} u^{2H}\right), \quad \xi_0 > 0. \tag{2.3}$$

The filtration $(\mathcal{F}_t)_{t\geq 0}$ can here be taken as the one generated by the two-dimensional Brownian motion $(W^1, W^\perp)$ under the risk neutral measure $\mathbb{Q}$, resulting in a filtered probability space $(\Omega, \mathcal{F}, \mathcal{F}_t, \mathbb{Q})$. The stock price process S is clearly then a local $(\mathcal{F}_t)_{t\geq 0}$-martingale and a supermartingale. We shall henceforth use the notation $\mathrm{E}[.] = E^{\mathbb{Q}}[. \mid \mathcal{F}_0]$ unless we state otherwise.

2.2 Option pricing under the rBergomi model

We are interested in pricing European call options under the rBergomi model. Assuming $S_0 = 1$, and using the conditioning argument on the σ-algebra generated by W^1 (an argument first used by [28] in the context of Markovian stochastic volatility models), we can show that the call price is given by

$$\begin{aligned} C_{\mathrm{RB}}(T,K) &= \mathrm{E}\left[(S_T - K)^+\right] \\ &= \mathrm{E}\left[\mathrm{E}\left[(S_T - K)^+ \mid \sigma(W^1(t), t \leq T)\right]\right] \\ &= \mathrm{E}\left[C_{\mathrm{BS}}\left(S_0 = \exp\left(\rho \int_0^T \sqrt{v_t} dW_t^1 - \frac{1}{2}\rho^2 \int_0^T v_t dt\right),\ k = K,\ \sigma^2 = (1-\rho^2)\int_0^T v_t dt\right)\right], \end{aligned} \tag{2.4}$$

where $C_{\mathrm{BS}}(S_0, k, \sigma^2)$ denotes the Black-Scholes call price, for initial spot price S_0, strike price k and volatility σ^2.

To show (2.4), we use the orthogonal decomposition of S_t into S_t^1 and S_t^2, where

$$S_t^1 = \mathcal{E}\{\rho \int_0^t \sqrt{v_s} dW_s^1\},\ S_t^2 = \mathcal{E}\{\sqrt{1-\rho^2} \int_0^t \sqrt{v_s} dW_s^\perp\},$$

and $\mathcal{E}(.)$ denotes the stochastic exponential [1]; then, if we define $\mathcal{F}_t^1 = \sigma\{W_s^1 : s \leq t\}$, we obtain by conditional log-normality

$$\log S_t \mid \mathcal{F}_t^1 \sim \mathcal{N}\left(\log S_t^1 - \frac{1}{2}(1-\rho^2)\int_0^t v_s ds, (1-\rho^2)\int_0^t v_s ds\right),$$

implying (2.4).

We point out that the analytical smoothing, based on conditioning, performed in (2.4) enables us to uncover the available regularity, and hence get a smooth, analytic integrand inside the expectation. Therefore, applying sparse quadrature techniques becomes an adequate option for computing the call price as we will investigate later. A similar conditioning was used in [24] but for variance reduction purposes.

[1] For a continuous semimartingale Z, the stochastic exponential is defined as $\mathcal{E}(Z)_t = \exp\left(Z_t - Z_0 - \frac{1}{2}[Z]_t\right)$, where $[Z]_t$ denotes the the quadratic variation of Z_t.

2.3 Simulation of the rBergomi model

One of the numerical challenges encountered in the simulation of the rBergomi dynamics is the computation of $\int_0^T \sqrt{v_t} dW_t^1$ and $V = \int_0^T v_t dt$ in (2.4), mainly because of the singularity of the Volterra kernel $K^H(s,t)$ at the diagonal $s = t$. In fact, one needs to jointly simulate the two-dimensional Gaussian process $(W_t^1, \widetilde{W}_t^H : 0 \leq t \leq T)$, resulting in $W_{t_1}^1, \ldots, W_{t_N}^1$ and $\widetilde{W}_{t_1}^H, \ldots, \widetilde{W}_{t_N}^H$ along a given time grid $t_1 < \cdots < t_N$. In the literature, there are essentially three possible ways to achieve this:

i) **Simple Euler discretization**: Euler discretization of the integral (2.2), defining $\widetilde{W}^H$, together with classical simulation of increments of W^1. This is inefficient because the integral is singular and adaptivity probably does not help, as the singularity moves with time. For this method, we need an N-dimensional random Gaussian input vector to produce one (approximate, inaccurate) sample of $W_{t_1}^1, \ldots, W_{t_N}^1, \widetilde{W}_{t_1}^H, \ldots, \widetilde{W}_{t_N}$.

ii) **Covariance based approach**: Given that $W_{t_1}^1, \ldots, W_{t_N}^1, \widetilde{W}_{t_1}^H, \ldots, \widetilde{W}_{t_N}$ together form a $(2N)$-dimensional Gaussian random vector with computable covariance matrix, one can use Cholesky decomposition of the covariance matrix to produce exact samples of $W_{t_1}^1, \ldots, W_{t_N}^1, \widetilde{W}_{t_1}^H, \ldots, \widetilde{W}_{t_N}$, but unlike the first way, we need a $2N$-dimensional Gaussian random vector as input. This method is exact but slow (see [4] and Section 4 in [6] for more details about this scheme). The simulation requires $\mathcal{O}\left(N^2\right)$ flops.

iii) **The hybrid scheme of [8]**: This scheme uses a different approach, which is essentially based on Euler discretization (i) but crucially improved by moment matching for the singular term in the left point rule. It is also inexact in the sense that samples produced here do not exactly have the distribution of $W_{t_1}^1, \ldots, W_{t_N}^1, \widetilde{W}_{t_1}^H, \ldots, \widetilde{W}_{t_N}$, however they are much more accurate then samples produced from method (i), but much faster than method (ii). As in method (ii), in this case, we need a $2N$-dimensional Gaussian random input vector to produce one sample of $W_{t_1}^1, \ldots, W_{t_N}^1, \widetilde{W}_{t_1}^H, \ldots, \widetilde{W}_{t_N}$.

In this work, we adopt approach (iii) for the simulation of the rBergomi asset price. As explained in [8], the scheme discretize $\widetilde{W}^H$ process into Wiener integrals of power functions and a Riemann sum, appearing from approximating the kernel by power functions near the origin and step functions elsewhere (see (2.5)). We utilize the first order variant ($\kappa = 1$) of the hybrid scheme, which is based on the following approximation

$$\begin{aligned} \widetilde{W}_{\frac{i}{N}}^H \approx \overline{W}_{\frac{i}{N}}^H &= \sqrt{2H}\left(\int_{\frac{i-1}{N}}^{\frac{i}{N}} \left(\frac{i}{N} - s\right)^{H-\frac{1}{2}} dW_u^1 + \sum_{k=2}^{i} \left(\frac{b_k}{N}\right)^{H-\frac{1}{2}} \left(W_{\frac{i-(k-1)}{N}}^1 - W_{\frac{i-k}{N}}^1\right)\right) \\ &= \sqrt{2H}\left(W_i^2 + \sum_{k=2}^{i} \left(\frac{b_k}{N}\right)^{H-\frac{1}{2}} \left(W_{\frac{i-(k-1)}{N}}^1 - W_{\frac{i-k}{N}}^1\right)\right), \end{aligned} \tag{2.5}$$

where N is the number of time steps and

$$b_k = \left(\frac{k^{H+\frac{1}{2}} - (k-1)^{H+\frac{1}{2}}}{H+\frac{1}{2}}\right)^{\frac{1}{H-\frac{1}{2}}}.$$

The sum in (2.5) requires the most computational effort in the simulation. Given that (2.5) can be seen as discrete convolution (see [8]), we employ the fast Fourier transform to evaluate it, which results in $\mathcal{O}(N \log N)$ floating point operations.

We note that the variates $\overline{W}_0^H, \overline{W}_1^H, \ldots, \overline{W}_{\frac{[Nt]}{N}}^H$ are generated by sampling $[Nt]$ i.i.d draws from a $(\kappa+1)$-dimensional Gaussian distribution and computing a discrete convolution. We denote these pairs of Gaussian random variables from now on by $(\mathbf{W}^{(1)}, \mathbf{W}^{(2)})$.

3 Details of our hierarchical method

We recall that our goal is to compute the expectation in (2.4). In fact, as seen in Section 2.3, we need $2N$-dimensional Gaussian inputs for the used hybrid scheme (N is the number of time steps in the time grid), namely

- $\mathbf{W}^{(1)} = \{W_i^{(1)}\}_{i=1}^N$: The N Gaussian random variables that are defined in Section 2.1.
- $\mathbf{W}^{(2)} = \{W_j^{(2)}\}_{j=1}^N$: An artificially introduced N Gaussian random variables that are used for left-rule points in the hybrid scheme, as explained in Section 2.3.

We can rewrite (2.4) as

$$\begin{aligned} C_{\mathrm{RB}}(T,K) &= \mathrm{E}\left[C_{\mathrm{BS}}\left(S_0 = \exp\left(\rho\int_0^T \sqrt{v_t}dW_t^1 - \frac{1}{2}\rho^2\int_0^T v_t dt\right),\ k=K,\ \sigma^2 = (1-\rho^2)\int_0^T v_t dt\right)\right] \\ &\approx \int_{\mathbb{R}^{2N}} C_{BS}\left(G(\mathbf{W}^{(1)},\mathbf{W}^{(2)})\right)\rho_N(\mathbf{W}^{(1)})\rho_N(\mathbf{W}^{(2)})d\mathbf{W}^{(1)}d\mathbf{W}^{(2)} \\ &= C_{RB}^N, \end{aligned} \tag{3.1}$$

where G maps $2N$ independent standard Gaussian random inputs to the parameters fed to Black-Scholes formula, and ρ_N is the multivariate Gaussian density, given by

$$\rho_N(\mathbf{z}) = \frac{1}{(2\pi)^{N/2}} e^{-\frac{1}{2}\mathbf{z}^T\mathbf{z}}.$$

Therefore, the initial integration problem that we are solving lives in $2N$-dimensional space, which becomes very large as the number of time steps N, used in the hybrid scheme, increases.

Our approach of approximating the expectation in (3.1) is based on MISC, proposed in [19]. We describe the MISC method in our context in Section 3.1. To make an effective use of MISC, we first apply two transformations to overcome the issue of facing a high dimensional integrand due to the discretization scheme used for simulating the rBergomi dynamics. The first transformation consists of applying a hierarchical path generation method, based on Brownian bridge (Bb) construction, with the aim of reducing the effective dimension as described in Section 3.2. The second transformation consists of applying Richardson extrapolation to reduce the bias, resulting in reducing the maximum number of dimensions needed for the integration problem. Details about Richardson extrapolation are provided in Section 3.3.

If we denote by $\mathcal{E}_{\mathrm{tot}}$ the total error of approximating the expectation in (2.4) using the MISC estimator, Q_N, then we have a natural error decomposition

$$\begin{aligned} \mathcal{E}_{\mathrm{tot}} &\leq \left|C_{\mathrm{RB}} - C_{\mathrm{RB}}^N\right| + \left|C_{\mathrm{RB}}^N - Q_N\right| \\ &\leq \mathcal{E}_B(N) + \mathcal{E}_Q(\mathrm{TOL}_{\mathrm{MISC}}, N), \end{aligned} \tag{3.2}$$

where $\mathcal{E}_Q$ is the quadrature error, $\mathcal{E}_B$ is the bias, and C_{RB}^N is the biased price computed with N time steps as given by (3.1).

3.1 The MISC method

We assume that we want to approximate the expected value $\mathrm{E}[f(Y)]$ of an analytic function $f\colon \Gamma \to \mathbb{R}$ using a tensorization of quadrature formulas over Γ.

To introduce simplified notations, we start with the one-dimensional case. Let us denote by β a non negative integer, referred to as a "stochastic discretization level", and by $m : \mathbb{N} \to \mathbb{N}$ a strictly increasing function with $m(0) = 0$ and $m(1) = 1$, that we call "level-to-nodes function". At level β, we consider a set of $m(\beta)$ distinct quadrature points in $\mathbb{R}$, $\mathcal{H}^{m(\beta)} = \{y_\beta^1, y_\beta^2, \ldots, y_\beta^{m(\beta)}\} \subset \mathbb{R}$, and a set of quadrature weights, $\boldsymbol{\omega}^{m(\beta)} = \{\omega_\beta^1, \omega_\beta^2, \ldots, \omega_\beta^{m(\beta)}\}$. We also let $C^0(\mathbb{R})$ be the set of real-valued continuous functions over $\mathbb{R}$. We then define the quadrature operator as

$$Q^{m(\beta)} : C^0(\mathbb{R}) \to \mathbb{R}, \quad Q^{m(\beta)}[f] = \sum_{j=1}^{m(\beta)} f(y_\beta^j)\omega_\beta^j. \tag{3.3}$$

In our case, we have in (3.1) a multi-variate integration problem with, $f = C_{\mathrm{BS}} \circ G$, $\mathbf{Y} = (\mathbf{W}^{(1)}, \mathbf{W}^{(2)})$, and $\Gamma = \mathbb{R}^{2N}$, in the previous notations. Therefore, we define for any multi-index $\boldsymbol{\beta} \in \mathbb{N}^{2N}$

$$Q^{m(\boldsymbol{\beta})} : \Gamma \to \mathbb{R}, \quad Q^{m(\boldsymbol{\beta})} = \bigotimes_{n=1}^{2N} Q^{m(\beta_n)},$$

where the n-th quadrature operator is understood to act only on the n-th variable of f. Practically, we obtain the value of $Q^{m(\boldsymbol{\beta})}[f]$ by using the grid $\mathcal{T}^{m(\boldsymbol{\beta})} = \prod_{n=1}^{2N} \mathcal{H}^{m(\beta_n)}$, with cardinality $\#\mathcal{T}^{m(\boldsymbol{\beta})} = \prod_{n=1}^{2N} m(\beta_n)$, and computing

$$Q^{m(\boldsymbol{\beta})}[f] = \sum_{j=1}^{\#\mathcal{T}^{m(\boldsymbol{\beta})}} f(\widehat{y}_j)\overline{\omega}_j,$$

where $\widehat{y}_j \in \mathcal{T}^{m(\boldsymbol{\beta})}$ and $\overline{\omega}_j$ are products of weights of the univariate quadrature rules. To simplify notation, hereafter, we replace $Q^{m(\boldsymbol{\beta})}$ by $Q^{\boldsymbol{\beta}}$.

Remark 3.1. We note that the quadrature points are chosen to optimize the convergence properties of the quadrature error. For instance, in our context, since we are dealing with Gaussian densities, using Gauss-Hermite quadrature points is the appropriate choice.

A direct approximation $\mathrm{E}\left[f[\mathbf{Y}]\right] \approx Q^{\boldsymbol{\beta}}[f]$ is not an appropriate option due to the well-known "curse of dimensionality". We use MISC, which is a hierarchical adaptive sparse grids quadrature strategy that uses stochastic discretizations and classic sparsification approach to obtain an effective approximation scheme for $\mathrm{E}\left[f\right]$.

For the sake of concreteness, in our setting, we are left with a $2N$-dimensional Gaussian random input, which is chosen independently, resulting in $2N$ numerical parameters for MISC, which we use as the basis of the multi-index construction. For a multi-index $\boldsymbol{\beta} = (\beta_n)_{n=1}^{2N} \in \mathbb{N}^{2N}$, we denote by $Q_N^{\boldsymbol{\beta}}$, the result of approximating (3.1) with a number of quadrature points in the i-th

dimension equal to $m(\beta_i)$. We further define the set of differences $\Delta Q_N^{\boldsymbol{\beta}}$ as follows: for a single index $1 \leq i \leq 2N$, let

$$\Delta_i Q_N^{\boldsymbol{\beta}} = \begin{cases} Q_N^{\boldsymbol{\beta}} - Q_N^{\boldsymbol{\beta}'}, \text{ with } \boldsymbol{\beta}' = \boldsymbol{\beta} - e_i, \text{ if } \boldsymbol{\beta}_i > 0 \\ Q_N^{\boldsymbol{\beta}}, \quad \text{otherwise} \end{cases} \tag{3.4}$$

where e_i denotes the ith $2N$-dimensional unit vector. Then, $\Delta Q_N^{\boldsymbol{\beta}}$ is defined as

$$\Delta Q_N^{\boldsymbol{\beta}} = \left(\prod_{i=1}^{2N} \Delta_i \right) Q_N^{\boldsymbol{\beta}}. \tag{3.5}$$

For instance, when $N = 1$, then

$$\begin{aligned} \Delta Q_1^{\boldsymbol{\beta}} = \Delta_2 \Delta_1 Q_1^{(\beta_1,\beta_2)} = \Delta_2 \left(Q_1^{(\beta_1,\beta_2)} - Q_1^{(\beta_1-1,\beta_2)} \right) &= \Delta_2 Q_1^{(\beta_1,\beta_2)} - \Delta_2 Q_1^{(\beta_1-1,\beta_2)} \\ &= Q_1^{(\beta_1,\beta_2)} - Q_1^{(\beta_1,\beta_2-1)} - Q_1^{(\beta_1-1,\beta_2)} + Q_1^{(\beta_1-1,\beta_2-1)}. \end{aligned}$$

Given the definition of C_{RB}^N by (3.1), we have the telescoping property

$$C_{RB}^N = Q_N^\infty = \sum_{\beta_1=0}^{\infty} \cdots \sum_{\beta_{2N}=0}^{\infty} \Delta Q_N^{(\beta_1,\ldots,\beta_{2N})} = \sum_{\boldsymbol{\beta} \in \mathbb{N}^{2N}} \Delta Q_N^{\boldsymbol{\beta}}. \tag{3.6}$$

The MISC estimator used for approximating (3.1), and using a set of multi-indices $\mathcal{I} \subset \mathbb{N}^{2N}$ is given by

$$Q_N^{\mathcal{I}} = \sum_{\boldsymbol{\beta} \in \mathcal{I}} \Delta Q_N^{\boldsymbol{\beta}}.$$

The quadrature error in this case is given by

$$\mathcal{E}_Q(\mathrm{TOL}_{\mathrm{MISC}}, N) = \left| Q_N^\infty - Q_N^{\mathcal{I}} \right| \leq \sum_{\boldsymbol{\beta} \in \mathbb{N}^{2N} \setminus \mathcal{I}} \left| \Delta Q_N^{\boldsymbol{\beta}} \right|. \tag{3.7}$$

We define the work contribution, $\Delta \mathcal{W}_{\boldsymbol{\beta}}$, to be the computational cost required to add $\Delta Q_N^{\boldsymbol{\beta}}$ to $Q_N^{\mathcal{I}}$, and the error contribution, $\Delta E_{\boldsymbol{\beta}}$, to be a measure of how much the quadrature error, defined in (3.7), would decrease once $\Delta Q_N^{\boldsymbol{\beta}}$ has been added to $Q_N^{\mathcal{I}}$, that is

$$\begin{aligned} \Delta \mathcal{W}_{\boldsymbol{\beta}} &= \mathrm{Work}[Q_N^{\mathcal{I} \cup \{\boldsymbol{\beta}\}}] - \mathrm{Work}[Q_N^{\mathcal{I}}] \\ \Delta E_{\boldsymbol{\beta}} &= \left| Q_N^{\mathcal{I} \cup \{\boldsymbol{\beta}\}} - Q_N^{\mathcal{I}} \right|. \end{aligned} \tag{3.8}$$

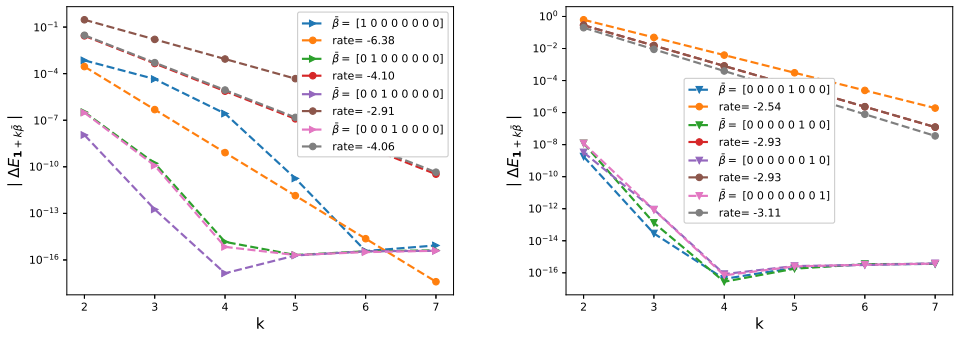
The construction of the optimal $\mathcal{I}$ will be done by profit thresholding, that is, for a certain threshold value $\overline{T}$, and a profit of a hierarchical surplus defined by

$$P_{\boldsymbol{\beta}} = \frac{|\Delta E_{\boldsymbol{\beta}}|}{\Delta \mathcal{W}_{\boldsymbol{\beta}}},$$

the optimal index set $\mathcal{I}$ for MISC is given by $\mathcal{I} = \{\boldsymbol{\beta} : P_{\boldsymbol{\beta}} \geq \overline{T}\}$.

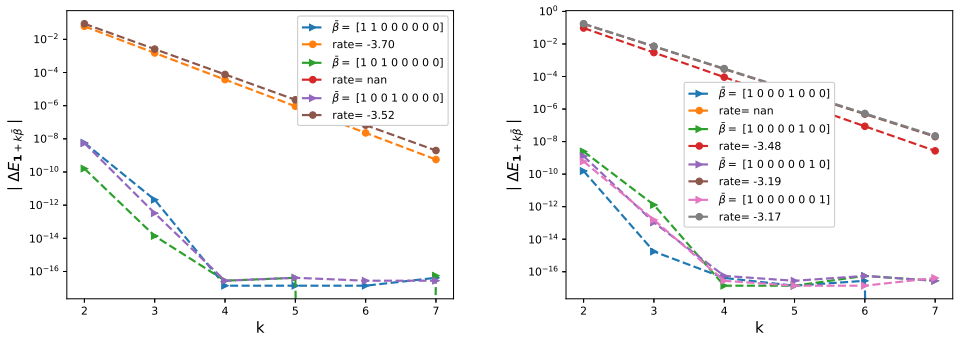
Remark 3.2. The choice of the hierarchy of quadrature points, $m(\boldsymbol{\beta})$, is flexible in the MISC algorithm and can be fixed by the user, depending on the convergence properties of the problem at hand. For instance, for the sake of reproducibility, in our numerical experiments we used a linear hierarchy: $m(\beta) = 4(\beta - 1) + 1$, $1 \leq \beta$, for results of parameter set 1 in Table 4.1. For the remaining parameter sets in Table 4.1, we used a geometric hierarchy: $m(\beta) = 2^{\beta-1} + 1$, $1 \leq \beta$.

Remark 3.3. As emphasized in [19], one important requirement to get the optimal performance of MISC method is to check the error convergence, defined by (3.8), of first and mixed differences operators. We checked this requirement in all our numerical experiments, and for illustration, we show in Figures 3.1 and 3.2, the error convergence of first and second order differences for the case of parameter set 2 in Table 4.1. These plots show that: i) $\Delta \mathrm{E}_{\boldsymbol{\beta}}$ decreases exponentially fast with respect to β_i, and ii) $\Delta \mathrm{E}_{\boldsymbol{\beta}}$ has a product structure since we observe a faster error decay for second differences compared to corresponding first differences operators.



(a) (b)

Figure 3.1: The rate of error convergence of first order differences $|\Delta \mathrm{E}_{\boldsymbol{\beta}}|$, defined by (3.8), ($\boldsymbol{\beta} = \mathbf{1} + k\overline{\boldsymbol{\beta}}$) with respect to $\mathbf{W}^{(1)}$ (a) and with respect to $\mathbf{W}^{(2)}$ (b), for parameter set 2 in Table 4.1. The number of quadrature points used in the i-th dimension is $N_i = 2^{\beta_i - 1} + 1$.



(a) (b)

Figure 3.2: The rate of error convergence of second order differences $|\Delta \mathrm{E}_{\boldsymbol{\beta}}|$, defined by (3.8), ($\boldsymbol{\beta} = \mathbf{1} + k\overline{\boldsymbol{\beta}}$) with respect to $\mathbf{W}^{(1)}$ (a) and with respect to $\mathbf{W}^{(2)}$ (b), for parameter set 2 in Table 4.1. The number of quadrature points used in the i-th dimension is $N_i = 2^{\beta_i - 1} + 1$.

3.2 Brownian bridge construction

In the literature of adaptive sparse grids and QMC, several hierarchical path generation methods (PGMs) or transformation methods have been proposed to reduce the effective dimension. Among these transformations, we cite the Brownian bridge (Bb) construction [25, 26, 11], the principal component analysis (PCA) [1] and the linear transformation (LT) [20].

In our context, the Brownian motion, on a time discretization, can be constructed either sequentially using a standard random walk construction or hierarchically using other hierarchical PGMs as listed above. For our purposes, to make an effective use of MISC, which profits from anisotropy, we use the Bb construction since it produces dimensions with different importance for MISC, contrary to random walk procedure for which all the dimensions of the stochastic space have equal importance. This transformation reduces the effective dimension of the problem and as a consequence accelerates the MISC procedure by reducing the computational cost.

Let us denote $\{t_i\}_{i=0}^{N}$ the grid of time steps. Then the Bb construction [18] consists of the following: given a past value B_{t_i} and a future value B_{t_k}, the value B_{t_j} (with $t_i < t_j < t_k$) can be generated according to

$$B_{t_j} = (1-\rho)B_{t_i} + \rho B_{t_k} + \sqrt{\rho(1-\rho)(k-i)\Delta t}\, z,\ z \sim \mathcal{N}(0,1), \tag{3.9}$$

where $\rho = \frac{j-i}{k-i}$.

3.3 Richardson extrapolation

Another transformation that we coupled with MISC is Richardson extrapolation [29]. In fact, applying level K_{R} (level of extrapolation) of Richardson extrapolation reduces dramatically the bias and as a consequence reduces the number of time steps N needed in the coarsest level to achieve a certain error tolerance. This means basically that Richardson extrapolation directly reduces the total dimension of the integration problem for achieving some error tolerance.

We recall that the Euler scheme has weak order one so that

$$\left|\mathrm{E}\left[f(\widehat{X}_T^h)\right] - \mathrm{E}\left[f(X_T)\right]\right| \leq Ch \tag{3.10}$$

for some constant C, all sufficiently small h and suitably smooth f. It can be easily shown that (3.10) can be improved to

$$\mathrm{E}\left[f(\widehat{X}_T^h)\right] = \mathrm{E}\left[f(X_T)\right] + ch + \mathcal{O}\left(h^2\right), \tag{3.11}$$

where c depends on f.

Applying (3.11) with discretization step $2h$, we obtain

$$\mathrm{E}\left[f(\widehat{X}_T^{2h})\right] = \mathrm{E}\left[f(X_T)\right] + 2ch + \mathcal{O}\left(h^2\right), \tag{3.12}$$

implying

$$2\mathrm{E}\left[f(\widehat{X}_T^{2h})\right] - \mathrm{E}\left[f(\widehat{X}_T^h)\right] = \mathrm{E}\left[f(X_T)\right] + \mathcal{O}\left(h^2\right), \tag{3.13}$$

For higher levels of extrapolations, we use the following: Let us denote by $h_J = h_0 2^{-J}$ the grid sizes (where h_0 is the coarsest grid size), by K_{R} the level of the Richardson extrapolation, and by $I(J, K_{\mathrm{R}})$ the approximation of $\mathrm{E}\left[f((X_T)\right]$ by terms up to level K_{R} (leading to a weak error of order K_{R}), then we have the following recursion

$$I(J, K_{\mathrm{R}}) = \frac{2^{K_{\mathrm{R}}}\left[I(J, K_{\mathrm{R}} - 1) - I(J-1, K_{\mathrm{R}} - 1)\right]}{2^{K_{\mathrm{R}}} - 1}, \quad J = 1, 2, \ldots, K_{\mathrm{R}} = 1, 2, \ldots \tag{3.14}$$

Remark 3.4. We emphasize that through our work, we are interested in the pre-asymptotic regime (small number of time steps), and the use of Richardson extrapolation is justified by our observed experimental results in that regime (see Section 4.1), which in particular, show an order one of convergence for the weak error. Although, we do not claim that the observed rates will scale well in the asymptotic regime, we observed that the pre-asymptotic regime is enough to get sufficiently accurate estimates for the option prices.

4 Numerical experiments

In this section, we show the results obtained through the different numerical experiments, conducted across different parameter constellations for the rBergomi model. Details about these examples are presented in Table 4.1. The first set is the one that is closest to the empirical findings [17, 7], which suggest that $H \approx 0.1$ (The choice of parameters values of $\nu = 1.9$ and $\rho = -0.9$ is justified by [4], where it is shown that these values are remarkably consistent with the SPX market on 4th February 2010). For the remaining 3 sets in Table 4.1, we wanted to test the potential of our method for a very rough case, that is $H = 0.02$, for 3 different scenarios of moneyness. It is interesting to note that due to the poor behavior of the strong error, that is of order of the Hurst parameter H [27], we expect that variance reduction methods, such as MLMC, to be inefficient for very small values of H. We emphasize that we checked the robustness of our method for other parameter sets, but for illustration purposes, we just show results for parameters sets presented in Table 4.1.

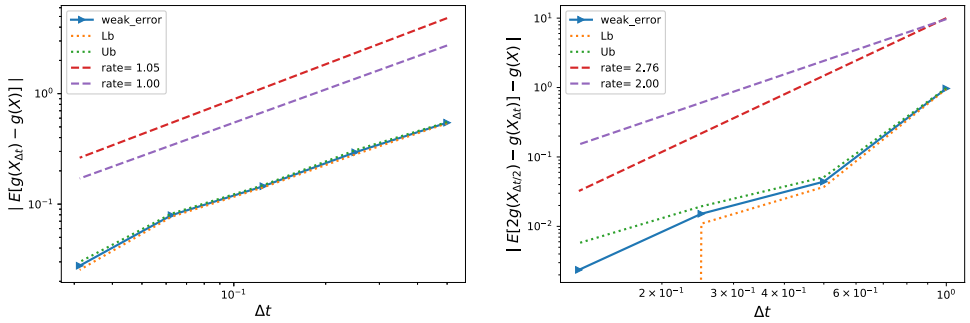
Parameters	Reference solution
Set 1: $H = 0.07, K = 1, S_0 = 1, \rho = -0.9, \eta = 1.9, \xi = 0.235^2$	0.0791 (7.9e−05)
Set 2: $H = 0.02, K = 1, S_0 = 1, \rho = -0.7, \eta = 0.4, \xi = 0.1$	0.1248 (1.3e−04)
Set 3: $H = 0.02, K = 0.8, S_0 = 1, \rho = -0.7, \eta = 0.4, \xi = 0.1$	0.2407 (5.6e−04)
Set 4: $H = 0.02, K = 1.2, S_0 = 1, \rho = -0.7, \eta = 0.4, \xi = 0.1$	0.0568 (2.5e−04)

Table 4.1: Reference solution, using MC with 500 time steps and number of samples, $M = 10^6$, of call option price under the rBergomi model, for different parameter constellations. The numbers between parentheses correspond to the statistical errors estimates.

4.1 Weak error

We start our numerical experiments with estimating accurately the weak error (bias) for the different parameter sets as in Table 4.1, with and without Richardson extrapolation. We consider a number of time steps $N \in \{2, 4, 8, 16\}$. We clarify that the weak errors considered here correspond to relative errors, normalized by the reference solutions provided in Table 4.1

For illustration purposes, we just show the weak errors related to set 1 in Table 4.1 (see Figure 4.1). We note that we observed similar behavior for the other parameter sets, with slightly worse rates for some cases.



(a) (b)

Figure 4.1: The convergence of the weak error, using MC, for set 1 parameter in Table 4.1. The upper and lower bounds are 95% confidence intervals. a) Without Richardson extrapolation. b) With Richardson extrapolation (level 1).

Remark 4.1. We emphasize that the reported weak rates correspond to the pre-asymptotic regime that we are interested in. Our results are purely experimental, and hence we cannot be sure what will happen in the asymptotic regime. We are not interested in estimating the rates specifically but rather obtaining a sufficiently precise estimate of the weak error (bias), $\mathcal{E}_B(N)$, for different number of time steps , N. For a fixed discretization, the corresponding estimated biased solution will be set as a reference solution to MISC method in order to estimate the quadrature error $\mathcal{E}_Q(\mathrm{TOL}_{\mathrm{MISC}}, N)$.

4.2 Comparing the different errors and computational time for MC and MISC

In this Section, we conduct a comparison between MC and MISC in terms of errors and computational time. we show tables and plots reporting the different relative errors involved in MC method (bias and statistical error [2] estimates), and in MISC (bias and quadrature error estimates). Fixing a sufficiently small error tolerance in the price estimates, we also compare the computational time needed for both methods to meet the desired error tolerance. The results are reported for the different sets of parameters defined in Table 4.1, and we consider a number of time steps $N \in \{2, 4, 8, 16\}$. The options are priced in terms of the moneyness. We emphasize that all reported errors are relative errors, normalized by the reference solutions provided in Table 4.1. We also note that in all cases the actual work (runtime) is obtained using an Intel(R) Xeon(R) CPU E5-268 architecture.

Through our conducted numerical experiments for each parameter set, we follow these steps to get our reported results:

i) For a fixed number of time steps, N, we compute an accurate estimate, using a large number of samples, M, of the biased MC solution, C_{RB}^N. This step also provide us with the bias error, $\mathcal{E}_B(N)$, defined by (3.2).

ii) The estimated biased solution, C_{RB}^N, is used as a reference solution to MISC to compute the quadrature error, $\mathcal{E}_Q(\mathrm{TOL}_{\mathrm{MISC}}, N)$, defined by (3.7).

We show in Table 4.2 the summary of our numerical findings, highlighting the computational gains achieved by MISC over MC to meet a certain error tolerance. More detailed results for each case of parameter set, as in Table 4.1, are provided in Sections (4.2.1, 4.2.2, 4.2.3, 4.2.4).

Parameter set	Level of Richardson extrapolation	Total relative error	Ratio of CPU time (MC/MISC)
Set 1	level 1	3%	1.6
	level 2	1%	> 9
Set 2	without	0.2%	5
Set 3	without	0.4%	7
Set 4	without	2%	1.3

Table 4.2: Summary of our numerical results.

4.2.1 Case of set 1 parameters in Table 4.1

For the case of set 1, we conduct our numerical experiments for 3 different scenarios: i) without Richardson extrapolation (see Tables 4.3 and 4.4), ii) with (level 1) Richardson extrapolation (see Tables 4.5 and 4.6), and iii) with (level 2) Richardson extrapolation (see Tables 4.7 and 4.8). Our numerical experiments show that MISC coupled with (level 1) Richardson extrapolation is 1.6 times faster than MC coupled with (level 1) Richardson extrapolation, to achieve a total relative error around 3%. This gain is improved when applying level 2 Richardson extrapolation. In fact, MISC coupled with (level 2) Richardson extrapolation is more than 9 times faster than MC coupled with

[2] The statistical error estimate of MC is $\frac{\sigma_M}{\sqrt{M}}$, where M is the number of samples and σ_M is the standard deviation estimate of the MC estimator.

(level 2) Richardson extrapolation, to achieve a total relative error below 1%. Applying Richardson extrapolation brought a significant improvement for MISC (see Figure 4.2 and related tables). We note that the values marked in red correspond to the total relative errors plotted in Figure 4.2, and will be used for computational work comparison against MC.

Method \Steps	2	4	8	16
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	**0.69** (0.54,0.15)	**0.42** (0.29,0.13)	**0.31** (0.15,0.16)	**0.11** (0.07,0.04)
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	**0.66** (0.54,0.12)	**0.29** (0.29,6e−04)	**0.16** (0.15,0.01)	**0.08** (0.07,0.01)
MC	**1.05** (0.54,0.51)	**0.59** (0.295,0.295)	**0.31** (0.155,0.155)	**0.14** (0.07,0.07)
M(# MC samples)	2×10	4×10	10^2	4×10^2

Table 4.3: Total relative error of MISC, without Richardson extrapolation, with different tolerances, and MC to compute call option price for different number of time steps. The values between parentheses correspond to the different errors contributing to the total relative error: for MISC we report the bias and quadrature errors and for MC we report the bias and the statistical errors estimates.

Method \Steps	2	4	8	16
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	0.08	0.13	0.7	163
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	0.2	5	333	1602
MC method	0.001	0.003	0.02	0.2

Table 4.4: Comparison of the computational time (in Seconds) of MC and MISC, to compute call option price of the rBergomi model for different number of time steps. The average MC CPU time is computed over 100 runs.

Method \Steps	$1-2$	$2-4$	$4-8$
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	**1.33** (0.96,0.37)	**0.18** (0.07,0.11)	**0.144** (0.015,0.129)
MISC ($\text{TOL}_{\text{MISC}} = 5.10^{-2}$)	**1.33** (0.96,0.37)	**0.23** (0.07,0.16)	**0.025** (0.015,0.010)
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	**1.08** (0.96,0.12)	**0.08** (0.07,0.01)	**0.025** (0.015,0.010)
MC	**1.88** (0.96,0.92)	**0.14** (0.07,0.07)	**0.03** (0.015,0.015)
M(# MC samples)	10	2×10^3	4×10^4

Table 4.5: Total relative error of MISC, coupled with Richardson extrapolation (level 1), with different tolerances, and MC, coupled with Richardson extrapolation (level 1), to compute call option price for different number of time steps. The values between parentheses correspond to the different errors contributing to the total relative error: for MISC we report the bias and quadrature errors and for MC we report the bias and the statistical errors.

Method \Steps	$1-2$	$2-4$	$4-8$
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	0.1	0.2	1.6
MISC ($\text{TOL}_{\text{MISC}} = 5.10^{-2}$)	0.1	0.6	37
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	1.3	6	2382
MC	0.003	2	60

Table 4.6: Comparison of the computational time (in Seconds) of MC and MISC, using Richardson extrapolation (level 1), to compute call option price of the rBergomi model for different number of time steps. The average MC CPU time is computed over 100 runs.

Method \Steps	$1-2-4$	$2-4-8$
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	**0.54** (0.24,0.30)	**0.113** (0.006,0.107)
MISC ($\text{TOL}_{\text{MISC}} = 5.10^{-2}$)	**0.49** (0.24,0.25)	**0.009** (0.006,0.003)
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	**0.27** (0.24,0.03)	**0.009** (0.006,0.003)
MC	**0.45** (0.24,0.21)	**0.012** (0.006,0.006)
M(# MC samples)	4×10^2	4×10^5

Table 4.7: Total relative error of MISC, coupled with Richardson extrapolation (level 2), with different tolerances, and MC, coupled with Richardson extrapolation (level 2), to compute call option price for different number of time steps. The values between parentheses correspond to the different errors contributing to the total relative error: for MISC we report the bias and quadrature errors and for MC we report the bias and the statistical errors.

Method \Steps	$1-2-4$	$2-4-8$
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	0.2	2
MISC ($\text{TOL}_{\text{MISC}} = 5.10^{-2}$)	0.5	74
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	9	3455
MC	0.2	690

Table 4.8: Comparison of the computational time (in Seconds) of MC and MISC, using Richardson extrapolation (level 2), to compute call option price of the rBergomi model for different number of time steps. The average MC CPU time is computed over 100 runs.

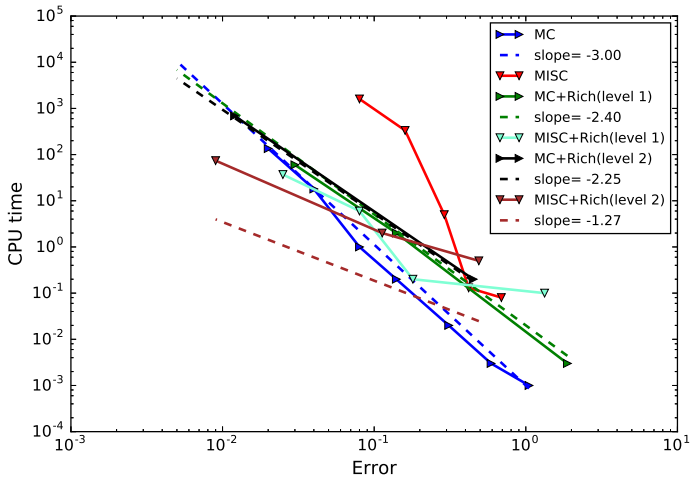


Figure 4.2: Computational work comparison for MISC and MC (with and without) Richardson extrapolation.

4.2.2 Case of set 2 parameters in Table 4.1

For the case of set 2, we just conduct our numerical experiments for the case without Richardson extrapolation, since the results show that we meet a small enough error tolerance without the need to apply Richardson extrapolation. Our numerical experiments show that MISC is 5 times faster than MC, to achieve a total relative error around 0.2% (see Tables 4.10 and 4.9). We note that the values marked in red correspond to the total relative errors plotted in Figure 4.3, and will be used for computational work comparison against MC.

Method \Steps	2	4	8	16
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	**0.03** (0.02,0.01)	**0.022** (0.008,0.014)	**0.022** (0.004,0.018)	**0.017** (0.001,0.016)
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	**0.03** (0.02,0.01)	**0.017** (0.008,0.009)	**0.008** (0.004,0.004)	**0.001** (0.001,4e−04)
MISC ($\text{TOL}_{\text{MISC}} = 10^{-3}$)	**0.02** (0.02,8e−04)	**0.009** (0.008,8e−04)	**0.005** (0.004,8e−04)	**0.001** (0.001,4e−04)
MC	**0.04** (0.02,0.02)	**0.016** (0.008,0.008)	**0.007** (0.004,0.003)	**0.002** (0.001,0.001)
M(# MC samples)	4×10^3	2×10^4	10^5	10^6

Table 4.9: Total relative error of MISC, without Richardson extrapolation, with different tolerances, and MC to compute call option price for different number of time steps.

Method \Steps	2	4	8	16
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	0.1	0.1	0.2	0.8
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	0.1	0.5	8	92
MISC ($\text{TOL}_{\text{MISC}} = 10^{-3}$)	0.5	3	24	226
MC method	0.15	1.6	16.5	494

Table 4.10: Comparison of the computational time (in Seconds) of MC and MISC, to compute call option price of the rBergomi model for different number of time steps. The average MC CPU time is computed over 100 runs.

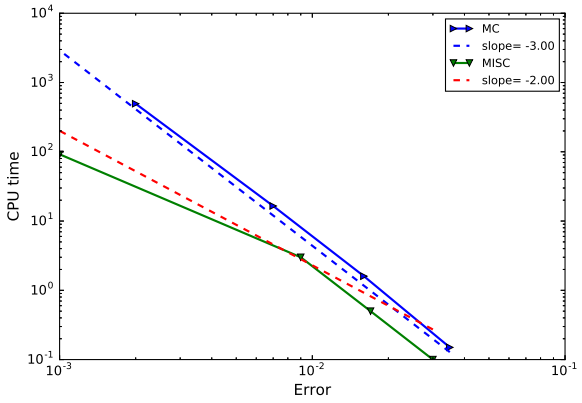


Figure 4.3: Computational work comparison for MISC and MC.

4.2.3 Case of set 3 parameters in Table 4.1

For the case of set 3, we just conduct our numerical experiments for the case without Richardson extrapolation, since the results show that we meet a small enough error tolerance without the need to apply Richardson extrapolation. Our numerical experiments show that MISC is 7 times faster than MC, to achieve a total relative error around 0.4% (see Figure 4.4 and Tables 4.12 and 4.11). We note that the values marked in red correspond to the total relative errors plotted in Figure 4.4, and will be used for computational work comparison against MC.

Method \Steps	2	4	8	16
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	**0.008** (0.006,0.002)	**0.009** (0.004,0.005)	**0.008** (0.003,0.005)	**0.009** (0.002,0.007)
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	**0.008** (0.006,0.002)	**0.009** (0.004,0.005)	**0.005** (0.003,0.002)	**0.002** (0.002,1e−04)
MISC ($\text{TOL}_{\text{MISC}} = 10^{-3}$)	**0.008** (0.006,0.002)	**0.006** (0.004,0.002)	**0.003** (0.003,1e−04)	**0.002** (0.002,1e−04)
MISC ($\text{TOL}_{\text{MISC}} = 10^{-4}$)	**0.006** (0.006,4e−04)	**0.004** (0.004,2e−04)	**0.003** (0.003,1e−04)	–
MC	**0.01** (0.006,0.005)	**0.008** (0.004,0.004)	**0.006** (0.003,0.003)	**0.004** (0.002,0.002)
M(# MC samples)	2×10^4	4×10^4	6×10^4	8×10^4

Table 4.11: Total relative error of MISC, without Richardson extrapolation, with different tolerances, and MC to compute call option price for different number of time steps.

Method \Steps	2	4	8	16
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	0.1	0.1	0.1	1
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	0.1	0.15	9	112
MISC ($\text{TOL}_{\text{MISC}} = 10^{-3}$)	0.2	2	27	2226
MISC ($\text{TOL}_{\text{MISC}} = 10^{-4}$)	1	6	136	–
MC method	1	3	10	40

Table 4.12: Comparison of the computational time (in Seconds) of MC and MISC, to compute call option price of the rBergomi model for different number of time steps. The average MC CPU time is computed over 100 runs.

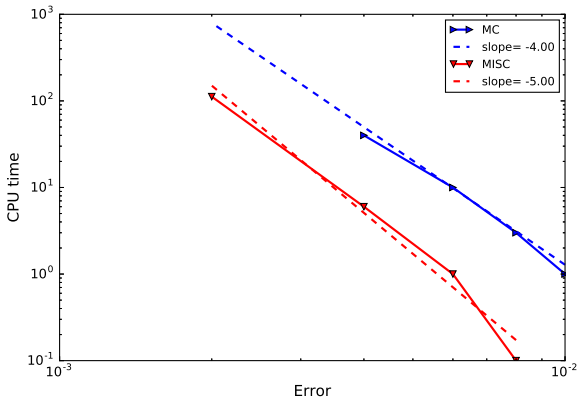


Figure 4.4: Comparison of computational work for MC and MISC.

4.2.4 Case of set 4 parameters in Table 4.1

For the case of set 4, we just conduct our numerical experiments for the case without Richardson extrapolation. Our numerical experiments show that MISC is 1.3 times faster than MC, to achieve

a total relative error around 2% (see Figure 4.5 and Tables 4.14 and 4.13). We note that the values marked in red correspond to the total relative errors plotted in Figure 4.5, and will be used for computational work comparison against MC.

Method \Steps	2	4	8	16
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	**0.09** (0.07,0.05)	**0.07** (0.03,0.04)	**0.07** (0.02,0.05)	**0.06** (0.01,2e−04)
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	**0.09** (0.07,5e−04)	**0.07** (0.03,0.04)	**0.02** (0.02,3e−04)	**0.02** (0.01,2e−04)
MISC ($\text{TOL}_{\text{MISC}} = 10^{-3}$)	**0.07** (0.07,5e−04)	**0.03** (0.03,4e−04)	**0.02** (0.02,3e−04)	**0.01** (0.01,2e−04)
MC	**0.14** (0.07,0.07)	**0.07** (0.03,0.04)	**0.04** (0.02,0.02)	**0.02** (0.01,0.01)
M(# MC samples)	6×10^2	2×10^3	8×10^3	2×10^4

Table 4.13: Total relative error of MISC, without Richardson extrapolation, with different tolerances, and MC to compute call option price for different number of time steps.

Method \Steps	2	4	8	16
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	0.1	0.1	0.2	0.5
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	0.1	0.1	8	97
MISC ($\text{TOL}_{\text{MISC}} = 10^{-3}$)	0.7	4	26	1984
MC method	0.02	0.15	1.4	10

Table 4.14: Comparison of the computational time (In seconds) of MC and MISC, to compute call option price of rBergomi model for different number of time steps. The average MC CPU time is computed over 100 runs.

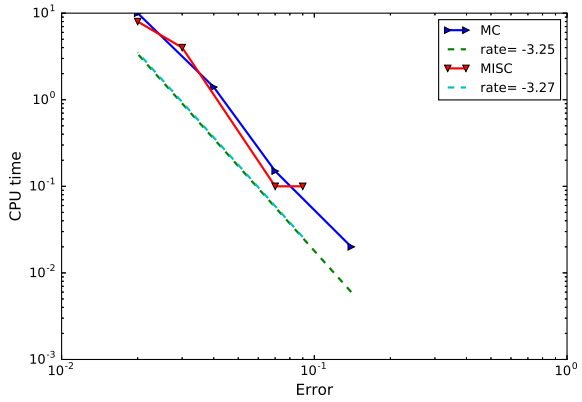


Figure 4.5: Comparison of computational work for MC and MISC.

5 Conclusions and future work

In this work, we propose a novel fast option pricer, based on a hierarchical adaptive sparse grids quadrature, specifically MISC as in [19], coupled with Brownian bridge construction and Richardson extrapolation, for options whose underlyings follow the rBergomi model as in [4].

Given that the only prevalent option, in this context, is to use different variants of MC method, which is computationally expensive, our first contribution is that we uncover the available regularity in the rBergomi model and design a novel alternative approach based on adaptive sparse grid quadrature, which opens a new research direction in this field to investigate the performance of other methods besides MC, for pricing and calibrating under the rBergomi model. Our second contribution is that we reduce the computational cost, through variance reduction, by using the conditional expectation as in [24] but also through bias reduction through using Richardson extrapolation. Finally, assuming one targets prices estimates with a sufficiently small error tolerance, our proposed method demonstrates substantial computational gains over the standard MC method, when pricing under the rBergomi model, even for very small values of the Hurst parameter. We show these gains through our numerical experiments for different parameters constellations. We clarify that we do not claim that these gains will hold, in the asymptotic regime, that is for higher accuracy requirements. Furthermore, the use of Richardson extrapolation is justified in the pre-asymptotic regime, in which our observed experimental results show, in particular, an order one of convergence for the weak error.

In this work, we limit ourselves to compare our novel proposed method against the standard MC. A more systematic comparison against the variant of MC proposed in [24] can be carried out but this is left for a future work. Another eventual direction of research may also investigate the performance of QMC for such problems. Finally, accelerating our novel designed method can be reached by coupling MISC with a more optimal hierarchical path generation method than Brownian bridge construction, such as PCA or LT transformations.

Acknowledgments C. Bayer gratefully acknowledges support from the German Research Foundation (DFG, grant BA5484/1). This work was supported by the KAUST Office of Sponsored Research (OSR) under Award No. URF/1/2584-01-01 and the Alexander von Humboldt Foundation. C. Ben Hammouda and R. Tempone are members of the KAUST SRI Center for Uncertainty Quantification in Computational Science and Engineering.

References Cited

[1] Peter A Acworth, Mark Broadie, and Paul Glasserman. A comparison of some monte carlo and quasi monte carlo techniques for option pricing. In *Monte Carlo and Quasi-Monte Carlo Methods 1996*, pages 1–18. Springer, 1998.

[2] Elisa Alòs, Jorge A León, and Josep Vives. On the short-time behavior of the implied volatility for jump-diffusion models with stochastic volatility. *Finance and Stochastics*, 11(4):571–589, 2007.

[3] Pierre Bajgrowicz, Olivier Scaillet, and Adrien Treccani. Jumps in high-frequency data: Spurious detections, dynamics, and news. *Management Science*, 62(8):2198–2217, 2015.

[4] Christian Bayer, Peter Friz, and Jim Gatheral. Pricing under rough volatility. *Quantitative Finance*, 16(6):887–904, 2016.

[5] Christian Bayer, Peter K Friz, Paul Gassiat, Joerg Martin, and Benjamin Stemper. A regularity structure for rough volatility. *arXiv preprint arXiv:1710.07481*, 2017.

[6] Christian Bayer, Peter K Friz, Archil Gulisashvili, Blanka Horvath, and Benjamin Stemper. Short-time near-the-money skew in rough fractional volatility models. *arXiv preprint arXiv:1703.05132*, 2017.

[7] Mikkel Bennedsen, Asger Lunde, and Mikko S Pakkanen. Decoupling the short-and long-term behavior of stochastic volatility. *arXiv preprint arXiv:1610.00332*, 2016.

[8] Mikkel Bennedsen, Asger Lunde, and Mikko S Pakkanen. Hybrid scheme for brownian semistationary processes. *Finance and Stochastics*, 21(4):931–965, 2017.

[9] Lorenzo Bergomi. Smile dynamics ii. 2005.

[10] F. Biagini, Y. Hu, B. Øksendal, and T. Zhang. *Stochastic Calculus for Fractional Brownian Motion and Applications*. Probability and Its Applications. Springer London, 2008.

[11] Russel E Caflisch, William J Morokoff, and Art B Owen. *Valuation of mortgage backed securities using Brownian bridges to reduce effective dimension*. 1997.

[12] Kim Christensen, Roel CA Oomen, and Mark Podolskij. Fact or friction: Jumps at ultra high frequency. *Journal of Financial Economics*, 114(3):576–599, 2014.

[13] Laure Coutin. An introduction to (stochastic) calculus with respect to fractional brownian motion. In *Séminaire de Probabilités XL*, pages 3–65. Springer, 2007.

[14] Martin Forde and Hongzhong Zhang. Asymptotics for rough stochastic volatility models. *SIAM Journal on Financial Mathematics*, 8(1):114–145, 2017.

[15] Masaaki Fukasawa. Asymptotic analysis for stochastic volatility: martingale expansion. *Finance and Stochastics*, 15(4):635–654, 2011.

[16] Jim Gatheral, Thibault Jaisson, Andrew Lesniewski, and Mathieu Rosenbaum. Volatility is rough, part 2: Pricing.

[17] Jim Gatheral, Thibault Jaisson, and Mathieu Rosenbaum. Volatility is rough. *arXiv preprint arXiv:1410.3394*, 2014.

[18] Paul Glasserman. *Monte Carlo methods in financial engineering*. Springer, New York, 2004.

[19] Abdul-Lateef Haji-Ali, Fabio Nobile, Lorenzo Tamellini, and Raul Tempone. Multi-index stochastic collocation for random pdes. *Computer Methods in Applied Mechanics and Engineering*, 306:95–122, 2016.

[20] Junichi Imai and Ken Seng Tan. Minimizing effective dimension using linear transformation. In *Monte Carlo and Quasi-Monte Carlo Methods 2002*, pages 275–292. Springer, 2004.

[21] Antoine Jacquier, Claude Martini, and Aitor Muguruza. On vix futures in the rough bergomi model. *Quantitative Finance*, 18(1):45–61, 2018.

[22] Antoine Jacquier, Mikko S Pakkanen, and Henry Stone. Pathwise large deviations for the rough bergomi model. *arXiv preprint arXiv:1706.05291*, 2017.

[23] Benoit B Mandelbrot and John W Van Ness. Fractional brownian motions, fractional noises and applications. *SIAM review*, 10(4):422–437, 1968.

[24] Ryan McCrickerd and Mikko S Pakkanen. Turbocharging monte carlo pricing for the rough bergomi model. *arXiv preprint arXiv:1708.02563*, 2017.

[25] William J Morokoff and Russel E Caflisch. Quasi-random sequences and their discrepancies. *SIAM Journal on Scientific Computing*, 15(6):1251–1279, 1994.

[26] Bradley Moskowitz and Russel E Caflisch. Smoothness and dimension reduction in quasi-monte carlo methods. *Mathematical and Computer Modelling*, 23(8):37–54, 1996.

[27] Andreas Neuenkirch and Taras Shalaiko. The order barrier for strong approximation of rough volatility models. *arXiv preprint arXiv:1606.03854*, 2016.

[28] Marc Romano and Nizar Touzi. Contingent claims and market completeness in a stochastic volatility model. *Mathematical Finance*, 7(4):399–412, 1997.

[29] Denis Talay and Luciano Tubaro. Expansion of the global error for numerical schemes solving stochastic differential equations. *Stochastic analysis and applications*, 8(4):483–509, 1990.