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

Christian Bayer* Chiheb Ben Hammouda† Raul Tempone‡

October 27, 2018

Abstract

One of the recent rough volatility models that showed a promising potential in quantitative finance is the rough Bergomi model, introduced in [3]. 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, the prevalent option is to use Monte Carlo (MC) simulation for efficient pricing. We design a novel alternative hierarchical approach, based on adaptive sparse grids quadrature, specifically multi-index stochastic collocation (MISC) as in [18], coupled with Brownian bridge construction and Richardson extrapolation. This hierarchical method demonstrates substantial computational gains with respect to the standard Monte Carlo method, assuming a sufficiently small error tolerance in the price estimates, across different parameter constellations.

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 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 [11, 2], and is one of the major criticisms of such models.

Motivated by the statistical analysis of realized volatility by Gatheral, Jaisson and Rosenbaum [16] and the theoretical results on implied volatility by Fukasawa [14], rough stochastic volatility has emerged as a new paradigm in quantitative finance, overcoming the observed limitations of

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

†Computer, Electrical and Mathematical Sciences and Engineering division, King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia (chiheb.benhammouda@kaust.edu.sa).

‡Computer, Electrical and Mathematical Sciences and Engineering division, King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia (raul.tempone@kaust.edu.sa).

diffusive stochastic volatility models. In these models, the trajectories of the volatility has lower Hölder regularity than those of the standard Brownian motion [16, 3]. In fact, they are based on fractional Brownian motion (fBm), which is a centered Gaussian process, whose covariance structure depends on the Hurst parameter $0 < H < 1/2$ (we refer to [22, 12, 9] for more details regarding the fBm processes). In this case, the fBm has negatively correlated increments and rough sample paths. Gatheral, Jaisson, and Rosenbaum [16] 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 [15]). These results were confirmed by Bennedsen, Lunde and Pakkanen [6], 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 [16, 6, 3] 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 [3]. 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 [8].

Despite the promising features of the rBergomi model, pricing and hedging under such 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: Monte Carlo (MC) estimators, 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 MC simulation. In particular, recent advances in simulation methods for the rBergomi model and different variants of pricing methods based on MC under such model have been proposed in [3, 4, 23, 7, 20]. For instance, in [23], 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 attempts to have analytical understanding of option pricing and implied volatility under this model have been achieved in [21, 5, 13].

In this paper, we design a novel hierarchical fast option pricer, based on a hierarchical adaptive sparse grids quadrature, specifically multi-index stochastic collocation (MISC) as in [18], coupled with Brownian bridge construction and Richardson extrapolation, for options whose underlyings follow the rBergomi model as in [3]. 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 [26], in the context of Markovian stochastic volatility models. In a second stage, we apply MISC as in [18], 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.

Compared to the works that we mentioned above, mainly [23], 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 to solve pricing and calibration problems related to the rBergomi model. Our second contribution is that we reduce the computational cost, through variance reduction, by using the conditional expectation as in [23] 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. We show these gains through our numerical experiments for different parameter constellations. In this paper, we limit ourselves to comparing our novel proposed method against standard MC. A more systematic comparison against the variant of MC proposed in [23] 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 different numerical experiments, across different parameter constellations for the rBergomi model.

2 Problem setting

In this section, we introduce the pricing framework that we are considering in this project. We start by giving some details for the rBergomi model proposed in [3]. We then derive the formula of the price of a European call option under the rBergomi model, in Section 2.2. This Section corresponds basically to the first stage of our approach, that is the analytical smoothing step. Finally, we explain some details about the hybrid scheme that we used 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 [3], normalized to $r = 0$, which is defined by

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

where $0 < H < 1$ (Hurst parameter) 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 Var $\left[\widetilde{W}_t^H\right] = t^{2H}$.

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

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

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

$$\begin{aligned} S_t &= S_0 \exp\left(\int_0^t \sqrt{v(s)} dZ(s) - \frac{1}{2}\int_0^t v(s) ds\right), \quad S_0 > 0 \\ v_u &= \xi_0(u) \exp\left(\eta \widetilde{W}_u^H - \frac{\eta^2}{2} u^{2H}\right), \quad \xi_0 > 0. \end{aligned} \tag{2.3}$$

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, therefore integrable. 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 [26] in the context of Markovian stochastic volatility models), we can show that the call price is given by

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

where $C_{BS}(S_0, k, \sigma^2)$ 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; then we obtain by conditional log-normality

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

where $\mathcal{F}_t^1 = \sigma\{W_s^1 : s \leq t\}$. Therefore, we obtain (2.4).

We point out that the analytical smoothing, based on conditioning, performed in (2.4) enables us to 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.

2.3 Simulation of the rBergomi model

One of the numerical challenges encountered in the simulation of the rBergomi dynamics is the computation of the terms $\int_0^T \sqrt{v_t} dW_t^1$ and $V = \int_0^T v_t dt$ as 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 $W_{t_1}^1, \ldots, W_{t_N}$ 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) 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) Given that $W_{t_1}^1, \ldots, W_{t_N}^1, \widetilde{W}_{t_1}^H, \ldots, \widetilde{W}_{t_N}$ together forms 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 $2 \times N$-dimensional Gaussian random vector as input. This method is exact but slow (See [3] and Section 4 in [5] for more details about this scheme). The simulation requires $\mathcal{O}\left(N^3\right)$ flops.

iii) The hybrid scheme of [7] uses a different approach, which is essentially based on Euler discretization as the first way 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 $2 \times N$-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 project, we adopt approach (iii) for the simulation of the rBergomi asset price. We utilize the first order variant ($\kappa = 1$) of the hybrid scheme [7], which is based on the approximation

$$\widetilde{W}^H_{\frac{i}{N}} \approx \overline{W}_{\frac{i}{N}} := \sqrt{2H}\left(\int_{\frac{i-1}{N}}^{\frac{i}{N}} \left(\frac{i}{N}-s\right)^{H-\frac{1}{2}} dW^1_u + \sum_{k=2}^{i}\left(\frac{b_k}{N}\right)^{H-\frac{1}{2}}\left(W^1_{\frac{i-(k-1)}{N}} - W^1_{\frac{i-k}{N}}\right)\right), \tag{2.5}$$

where N is the number of time steps and

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

Employing the fast Fourier transform to evaluate the sum in (2.5), which is a discrete convolution, a skeleton $\overline{W}^H_0, \overline{W}^H_1, \ldots, \overline{W}^H_{\frac{[Nt]}{N}}$ can be generated in $\mathcal{O}(N \log N)$ floating point operations.

The variates $\overline{W}^H_0, \overline{W}^H_1, \ldots, \overline{W}^H_{\frac{[Nt]}{N}}$ 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 (W^1, W^2).

3 Details of our approach

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

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

Thus, we can rewrite (2.4) as

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

where G maps $2N$ independent standard Gaussian random inputs to the parameters fed to Black-Scholes formula as in (3.1), 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 multi-index stochastic collocation (MISC), proposed in [18]. We describe the MISC solver 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 $\mathcal{E}_{\text{tot}}$ the total error of computing the expectation in (2.4) using MISC solver, then we have a natural error decomposition, giving by

$$\mathcal{E} \leq \mathcal{E}_Q(\text{TOL}_{\text{MISC}}, N) + \mathcal{E}_B(N), \tag{3.2}$$

where $\mathcal{E}_Q$ is the quadrature error, a function of the MISC tolerance: TOL_{MISC} and N: the number of time steps, and $\mathcal{E}_B$ is the bias, function of N.

3.1 The MISC solver

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 formula over Γ.

To introduce simplified notations, we start with the one-dimensional case. Let us denote by $\beta \geq 1$ an 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, and following (3.1), we have a multi-variate integration problem and we have, given the previous notations, $f := C_{\text{BS}} \circ G$, $\mathbf{Y} = (\mathbf{W}^1, \mathbf{W}^2)$, and $\Gamma = \mathbb{R}^{2N}$. 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 considering the tensor 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

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

where $\widehat{y}_j \in \mathcal{T}^{m(\boldsymbol{\beta})}$ and $\overline{\omega}_j$ are products of weights of the univariate quadrature rules.

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 $\mathrm{E}[f[\mathbf{Y}]] \approx Q^{m(\boldsymbol{\beta})}[f]$ is not an appropriate option due to the well-known "curse of dimensionality" effect. We use MISC as in [18], which is a hierarchical adaptive sparse grids quadrature strategy that uses stochastic discretizations and classic sparsification approach to obtain an effective approximation scheme for $\mathrm{E}[f]$.

For the sake of concreteness, in our setting, we are left with a $2N$-dimensional Gaussian random inputs, which are chosen independently, resulting in $2N$ numerical parameters for MISC, which we use as the basis of the multi-index construction, reflecting the fact that W_i^1 and W_j^2 can vary independently of each other regardless of $i \neq j$ or $i = j$. Let $l \in \{1, \ldots, 2N\}$ and set

$$p_l \coloneqq \begin{cases} W_l^1, & 1 \leq l \leq N, \\ W_{l-N}^2, & N+1 \leq l \leq 2N. \end{cases} \tag{3.4}$$

For a multi-index $\ell = (l_i)_{i=1}^{2N} \in \mathbb{N}^{2N}$, we denote by $Q_N^\ell \coloneqq Q^{N,m(\ell)}(p_\ell)$ the result of a discretized integral (we emphasize that the "integral" here is the time integral, not the expectation integral) using N time steps and with parameters $p_\ell \coloneqq (p_{l_i})_{i=1}^{2N}$, and with a number of quadrature points $m(l_i)$ in the dimension p_{l_i}. We further define the set of differences ΔQ_ℓ^N as follows: for a single index $1 \leq i \leq 2N$, let

$$\Delta_i Q_N^\ell \coloneqq \begin{cases} Q_N^\ell - Q_N^{\ell'} \text{ with } \ell' = \ell - e_i, \text{ if } \ell_i > 0 \\ Q_N^\ell, \quad \text{otherwise} \end{cases} \tag{3.5}$$

where e_i denotes the ith $2N$-dimensional unit vector. Then, ΔQ_N^ℓ is defined as

$$\Delta Q_N^\ell \coloneqq \left(\prod_{i=1}^{2N} \Delta_i \right) Q_N^\ell. \tag{3.6}$$

For instance, when $N = 1$, then

$$\begin{aligned} \Delta Q_1^\ell = \Delta_2 \Delta_1 Q_1^{(l_1,l_2)} = \Delta_2 \left(Q_1^{(l_1,l_2)} - Q_1^{(l_1-1,l_2)} \right) &= \Delta_2 Q_1^{(l_1,l_2)} - \Delta_2 Q_1^{(l_1-1,l_2)} \\ &= Q_1^{(l_1,l_2)} - Q_1^{(l_1,l_2-1)} - Q_1^{(l_1-1,l_2)} + Q_1^{(l_1-1,l_2-1)}. \end{aligned}$$

We have the telescoping property

$$Q_N^\infty = \sum_{l_1=0}^{\infty} \cdots \sum_{l_{2N}=0}^{\infty} \Delta Q_N^{(l_1,\ldots,l_{2N})} = \sum_{\ell \in \mathbb{N}^{2N}} \Delta Q_N^\ell, \tag{3.7}$$

where Q_N^∞ is the biased option price computed with N time steps.

As stated before, our goal is to approximate Q_N^∞. The MISC estimator computed using a set of multi-indices $\mathcal{I} \subset \mathbb{N}^{2N}$ is given by

$$Q_N^{\mathcal{I}} \coloneqq \sum_{\ell \in \mathcal{I}} \Delta Q_N^\ell.$$

The quadrature error in this case is given by

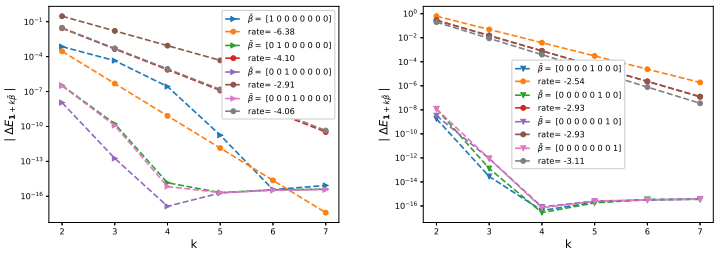
$$\mathcal{E}_Q(\mathrm{TOL}_{\mathrm{MISC}}, N) = \left|Q_N^{\infty} - Q_N^{\mathcal{I}}\right| \leq \sum_{\ell \in \mathbb{N}^{2N} \setminus \mathcal{I}} \left|\Delta Q_N^{\ell}\right|. \tag{3.8}$$

If we denote the computational work at level $\ell = (l_1, \ldots, l_{2N})$ for adding an increment ΔQ_N^{ℓ} in the telescoping sum by $\mathcal{W}_N^{\ell}$, then the construction of the optimal $\mathcal{I}$ will be done by profit thresholding, i.e., for a certain threshold value $\overline{T}$, we add a multi-index ℓ to $\mathcal{I}$ provided that

$$\log\left(\frac{\left|\Delta Q_N^{\ell}\right|}{\mathcal{W}_N^{\ell}}\right) \leq \overline{T}.$$

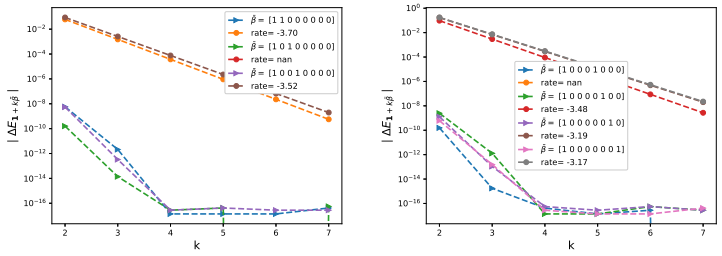
Remark 3.2. The choice of the hierarchy of quadrature points, $m(\ell)$, is flexible in the MISC solver and can be fixed by the user, depending on the convergence properties of the problem at hand. For instance, for the sake of reproducibility, in our numerical experiments we used a linear hierarchy: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

Remark 3.3. As emphasized in [18], one important requirement to get the optimal performance of MISC solver is to check the convergence of first and mixed differences operators (See [18] for details). We checked this requirement in all our numerical experiments, and for illustration, we show in figures (3.1, 3.2), the convergence of first and second order differences for the case of parameter set 2 in table 4.1.



(a) (b)

Figure 3.1: The rate of convergence of first order differences $|\Delta \mathrm{E}_\beta|$ ($\beta = \mathbf{1} + k\overline{\beta}$) with respect to W^1 (a) and with respect to 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 convergence of second order differences $|\Delta \mathrm{E}_\beta|$ ($\beta = \mathbf{1} + k\overline{\beta}$) with respect to W^1 (a) and with respect to 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 [24, 25, 10], the principal component analysis (PCA) [1] and the linear transformation (LT) [19].

In our context, the Brownian motion can be constructed either sequentially using a standard random walk construction or hierarchically using other hierarchical PGM 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 dimension 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 [17] 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 the formula:

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

where $\rho = \frac{j-i}{k-i}$. In particular, if N is a power of 2, then given $B_0 = 0$, Bb generates the Brownian motion at times $T, T/2, T/4, 3T/4, \ldots$ according

$$\begin{aligned} B_T &= \sqrt{T} z_1 \\ B_{T/2} &= \frac{1}{2}(B_0 + B_T) + \sqrt{T/4} z_2 = \frac{\sqrt{T}}{2} z_1 + \frac{\sqrt{T}}{2} z_2 \\ B_{T/4} &= \frac{1}{2}(B_0 + B_{T/2}) + \sqrt{T/8} z_3 = \frac{\sqrt{T}}{4} z_1 + \frac{\sqrt{T}}{4} z_2 + \sqrt{T/8} z_3 \\ &\vdots \end{aligned} \tag{3.10}$$

where $\{z_j\}_{j=1}^N$ are independent standard normal variables. In Bb construction scheme given by

(3.10), the most important values that determine the large scale structure of Brownian motion are the first components of $\mathbf{z} = (z_1, \ldots, z_N)$.

3.3 Richardson extrapolation

Another transformation that we coupled with MISC is Richardson extrapolation [27]. In fact, applying level K_{R} of Richardson extrapolation reduces dramatically the bias and as a consequence reduces the needed number of time steps N used in the coarsest level to achieve a certain error tolerance. This means basically that Richardson extrapolation reduces directly the total dimension of the integration problem for achieving some error tolerance.

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

$$\left|\mathrm{E}\left[f(\widehat{X}_T^h)\right] - \mathrm{E}\left[f(X_T)\right]\right| \leq Ch \tag{3.11}$$

for some constant C, all sufficiently small h and suitably smooth f. It can be easily shown that (3.11) can be improved to

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

where c depends on f.

Applying (3.12) with discretization step $2h$, we obtain

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

implying

$$2\mathrm{E}\left[f(\widehat{X}_T^{2h})\right] - \mathrm{E}\left[f(\widehat{X}_T^h)\right] = \mathrm{E}\left[f(X_T)\right] + \mathcal{O}\left(h^2\right), \tag{3.14}$$

For higher levels 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

$$I(J, K_{\mathrm{R}}) = \frac{2^{K_{\mathrm{R}}}\left[I(J, K_{\mathrm{R}} - 1) - I(J-1, K_{\mathrm{R}} - 1)\right]}{2^{K_{\mathrm{R}}} - 1}, \quad J = 1, 2, \ldots, K_{\mathrm{R}} = 1, 2, \ldots \tag{3.15}$$

4 Numerical experiments

4.1 Weak error plots

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 $N \in \{2, 4, 8, 16\}$. The options are priced in terms of the moneyness K, where K is

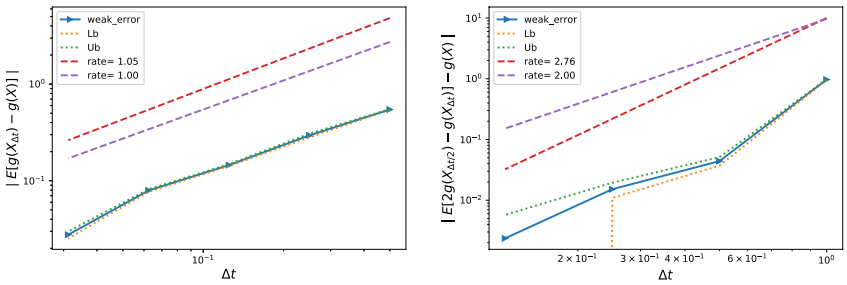
the strike price. The reference solution was computed with $N = 500$ time steps (reported in table 4.1). We note that the weak errors considered here correspond to relative errors. We show in table 4.1 the different parameter constellations that we consider to report our results for MC and MISC, along with the corresponding reference solution.

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 constellation. The numbers between parentheses correspond to the statistical errors.

For illustration purposes, we just show the weak errors related to set 1 parameter 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.

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 on estimating the rates specifically but rather a sufficient precise estimate of the weak error (Bias), $\mathcal{E}_B(N)$, for different time steps N. For a fixed discretization, the corresponding biased solution will be set as a reference solution to MISC method in order to estimate the quadrature error $\mathcal{E}_Q(\text{TOL}_{\text{MISC}}, N)$.



(a) (b)

Figure 4.1: The rate of 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).

4.2 Comparing the different errors and complexity for MC and MISC

In this Section, we show tables and plots reporting the different errors involved in MC method (bias and statistical error), and in MISC (quadrature error). We do this for each parameter set. Given

that both methods, MC and MISC, have the same bias, the computational time of MC and MISC is compared such that the statistical error is almost equal to the stable quadrature error produced by MISC. The results were reported for the different sets of parameters defined in table 4.1. We consider a number of time steps $N \in \{2, 4, 8, 16\}$. The options are priced in terms of the moneyness K, where K is the strike price.

We show in Section 4.2.1 the detailed outputs for the case of set 1 parameter, reflecting the methodology of our proposed method. Then, in the subsequent sections, we show the main results for the other cases, following the same methodology. The conclusions of this section are:

i) For the case of set 1 (See Section 4.2.1), MISC coupled with Richardson extrapolation is 19 times faster than MC coupled with Richardson extrapolation, to achieve a total relative error around 8% and 4 times faster than MC, to achieve total relative error around 2% (see tables (4.6,4.7)). This gain is improved when applying level 2 Richardson extrapolation. In fact, MISC coupled with Richardson extrapolation (level 2) is 17 times faster than MC coupled with Richardson extrapolation (level 2), to achieve a total relative error below 1% (see tables (4.9, 4.10)). Applying Richardson extrapolation brought a significant improvement for MISC (see figure 4.4 and related tables).

ii) For the case of set 2 (See Section 4.2.2), MISC is 43 times faster than MC, to achieve a total relative error below 1% and 8 times faster than MC, to achieve a total relative error around 0.1% (see figure 4.5 and tables (4.12, 4.11)). Using Richardson extrapolation brought an improvement in terms of the complexity constant compared to using simple MISC (See figure 4.6 and tables (4.13, 4.14)).

iii) For the case of set 3 (See Section 4.2.3), MISC is 141 times faster than MC, to achieve a total relative error around 0.6% (see figure 4.7 and tables (4.16, 4.15)).

iv) For the case of set 4 (See Section 4.2.4), MISC is 220 times faster than MC, to achieve a total relative error below 7% and 16 times faster than MC, to achieve a total relative error around 2% (see figure 4.8 and tables (4.18, 4.17)).

4.2.1 Case of set 1 parameters in table 4.1

Without Richardson extrapolation

We show through table 4.2 the bias and the statistical error for MC method, related to Section 4.1. We define the statistical error of MC to be $\frac{\sigma_M}{\sqrt{M}}$, where M is the number of samples and σ_M is the standard deviation of the MC estimator.

Method \Steps	2	4	8	16
MC Bias ($M = 8.10^6$)	**0.5375** (0.0426)	**0.2922** (0.0208)	**0.1542** (0.0122)	**0.0731** (0.0058)
MC Statistical error ($M = 8.10^6$)	**2.5e − 03** (2.0e−04)	**1.3e − 03** (1.0e−04)	**6.3e − 04** (5.0e−05)	**1.33e − 03** (1.1e−04)

Table 4.2: Bias and statistical errors of MC, without Richardson extrapolation, for computing call option price for different number of time steps. The numbers between parentheses are the corresponding absolute errors.

In figure 4.2, we plot the behavior of the relative quadrature error with respect to TOL_{MISC}. The quadrature error (see (3.8)) is computed by subtracting the MISC solution from the biased solution, computed with sufficiently large number of samples.

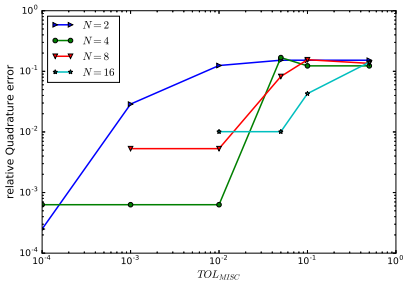


Figure 4.2: Quadrature error of MISC, without Richardson extrapolation, with different tolerances, to compute call option price for different number of time steps.

We report in table 4.3 the total relative error which is the sum of the bias and statistical error for MC, and the quadrature error for MISC. We note that we used a number of samples for MC such that the statistical error is almost equal to the stable quadrature error, in order to have a fair complexity comparison between the two methods.

Method \Steps	2	4	8
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	**0.6900**	**0.4153**	**0.3097**
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	**0.6622**	**0.2928**	**0.1595**
MISC ($\text{TOL}_{\text{MISC}} = 10^{-3}$)	**0.6371**	**0.2928**	**0.1595**
MISC ($\text{TOL}_{\text{MISC}} = 10^{-4}$)	**0.5378**	**0.2928**	–
MC	**0.5400** (M=8.10^6)	**0.2935** (M=8.10^6)	0**0.1593** (M=10^5)

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 marked in red correspond to the total relative errors associated with stable quadrature errors for MISC, and will be used for complexity comparison against MC. The values between parentheses correspond to the needed number of samples for MC to achieve the desired error.

In table 4.4, we show the CPU time needed for each method. We note that in all cases the actual work (runtime) was obtained using a 40-core Intel(R) Xeon(R) CPU E5-268 architecture.

Method \Steps	2	4	8
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	0.08	0.13	0.7
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	0.2	5	333
MISC ($\text{TOL}_{\text{MISC}} = 10^{-3}$)	2	73	3650
MISC ($\text{TOL}_{\text{MISC}} = 10^{-4}$)	43	1240	–
MC method	220	358	9
Ratio of (MC/MISC)	5	72	0.03

Table 4.4: Comparison of the computational time (in Seconds) of MC and MISC, used to compute call option price of the rBergomi model for different number of time steps. The average MC CPU time is computed over 10 runs.

With Richardson extrapolation (level 1)

Method \Steps	$1-2$	$2-4$	$4-8$	$8-16$
MC Bias ($M = 10^6$)	**0.9594** (0.0760)	**0.0686** (0.0054)	**0.0149** (0.0012)	**0.0024** (0.0004)
MC Statistical error ($M = 10^6$)	**1.3e − 02** (1e−03)	**4.1e − 03** (3.2e−04)	**2.1e − 03** (1.7e−04)	**1.6e − 03** (1.3e−04)

Table 4.5: Bias and statistical errors of MC, with Richardson extrapolation (level 1), for computing call option price for different number of time steps. The numbers between parentheses are the corresponding absolute errors.

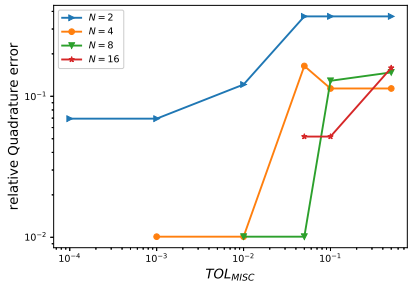


Figure 4.3: Quadrature error of MISC,with Richardson extrapolation (level 1), with different tolerances, to compute call option price for different number of time steps.

Method \Steps	$1-2$	$2-4$	$4-8$
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	**1.3281**	**0.1822**	**0.1437**
MISC ($\text{TOL}_{\text{MISC}} = 5.10^{-2}$)	**1.3281**	**0.2327**	**0.0250**
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	**1.0806**	**0.0787**	**0.0250**
MISC ($\text{TOL}_{\text{MISC}} = 10^{-3}$)	**1.0288**	**0.0787**	–
MC	**1.0288** (M=4.10^4)	**0.0787** (M=2.10^5)	**0.0250** (M=5.10^4)

Table 4.6: 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 marked in red correspond to the total relative errors associated with stable quadrature errors for MISC, and will be used for complexity comparison against MC. The values between parentheses correspond to the needed number of samples for MC to achieve the desired error.

Method \Steps	$1-2$	$2-4$	$4-8$
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	0.1	0.18	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
MISC ($\text{TOL}_{\text{MISC}} = 10^{-3}$)	3.5	244	–
MC	12	113	130
Ratio of (MC/MISC)	3.4	18.8	3.5

Table 4.7: Comparison of the computational time (in Seconds) of MC and MISC, using Richardson extrapolation (level 1), used to compute call option price of the rBergomi model for different number of time steps. The average MC CPU time is computed over 10 runs.

With Richardson extrapolation (level 2)

Method \Steps	$1-2-4$	$2-4-8$
MC Bias ($M = 3.10^6$)	**0.2411** (1.9e−02)	**0.0058** (4.6e−04)
MC Statistical error ($M = 3.10^6$)	**3.5e − 03** (2.8e−04)	**1.8e − 03** (1.4e−04)

Table 4.8: Bias and statistical errors of MC, with Richardson extrapolation (level 2), for computing call option price for different number of time steps. The numbers between parentheses are the corresponding absolute errors.

Method \Steps	$1-2-4$	$2-4-8$
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	**0.2588**	**0.1131**
MISC ($\text{TOL}_{\text{MISC}} = 5.10^{-2}$)	**0.4936**	**0.0096**
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	**0.2689**	**0.0096**
MC	**0.2689** (M=5.10^4)	**0.0096** (M=7.10^5)

Table 4.9: 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 marked in red correspond to the total relative errors associated with stable quadrature errors for MISC, and will be used for complexity comparison against MC.

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	118	1274
Ratio of (MC/MISC)	13	17

Table 4.10: Comparison of the computational time (in Seconds) of MC and MISC, using Richardson extrapolation (level 2), used to compute call option price of the rBergomi model for different number of time steps. The average MC CPU time is computed over 10 runs.

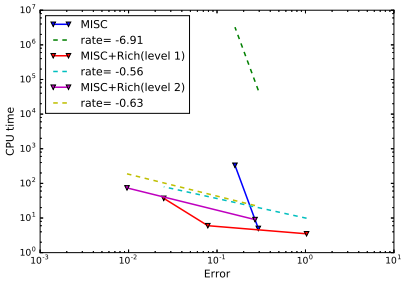


Figure 4.4: Complexity plot for MISC (with and without) Richardson extrapolation.

4.2.2 Case of set 2 parameters in table 4.1

Without Richardson extrapolation

Method \Steps	2	4	8	16
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	**0.0262**	**0.0222**	**0.0218**	**0.0168**
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	**0.0262**	**0.0166**	**0.0082**	**0.0016**
MISC ($\text{TOL}_{\text{MISC}} = 10^{-3}$)	**0.0190**	**0.0094**	**0.0050**	**0.0008**
MISC ($\text{TOL}_{\text{MISC}} = 10^{-4}$)	**0.0182**	**0.0086**	**0.0050**	–
MC	**0.0179** (M=5.10^6)	**0.0083** (M=5.10^6)	**0.0047** (M=5.10^6)	**0.0013** (M=5.10^6)

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	0.2	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	2226
MISC ($\text{TOL}_{\text{MISC}} = 10^{-4}$)	1	6	80	–
MC method	122	260	427	766
Ratio of ($MC/MISC$)	122	43	18	8

Table 4.12: Comparison of the computational time (in Seconds) of MC and MISC, used to compute call option price of the rBergomi model for different number of time steps. The average MC CPU time is computed over 10 runs.

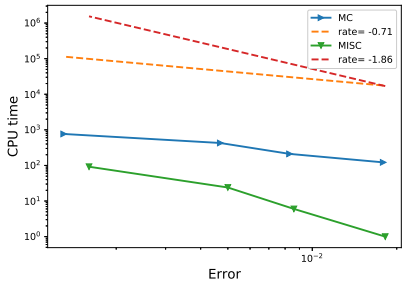


Figure 4.5: Complexity plot for MC and MISC, without Richardson extrapolation.

With Richardson extrapolation (level 1)

Method \Steps	$2-4$	$4-8$	$8-16$
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	0.0273	0.0225	0.0181
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	0.0193	0.0041	0.0013
MISC ($\text{TOL}_{\text{MISC}} = 10^{-3}$)	0.0057	0.0025	0.0013
MC	**0.0073** (M=5.10^4)	**0.0025** (M=4.10^5)	**0.0013** (M=2.10^6)

Table 4.13: 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.

Method \Steps	$2-4$	$4-8$	$8-16$
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	0.15	0.25	1
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	0.6	10	112
MISC ($\text{TOL}_{\text{MISC}} = 10^{-3}$)	3.5	34	3150
MC	45	438	2240
Ratio of (MC/MISC)	13	13	20

Table 4.14: Comparison of the computational time (in Seconds) of MC and MISC, using Richardson extrapolation (level 1), used to compute call option price of the rBergomi model for different number of time steps. The average MC CPU time is computed over 10 runs.

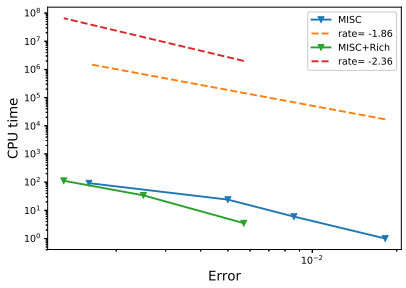


Figure 4.6: Complexity plot for MISC (with and without) Richardson extrapolation.

4.2.3 Case of set 3 parameters in table 4.1

Method \Steps	2	4	8	16
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	**0.0083**	**0.0089**	**0.0075**	**0.0090**
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	**0.0083**	**0.0089**	**0.0050**	**0.0025**
MISC ($\text{TOL}_{\text{MISC}} = 10^{-3}$)	**0.0083**	**0.0056**	**0.0030**	**0.0025**
MISC ($\text{TOL}_{\text{MISC}} = 10^{-4}$)	**0.0058**	**0.0037**	**0.0030**	–
MC	**0.0057** (M=5.10^6)	**0.0038** (M=5.10^6)	**0.0032** (M=5.10^6)	**0.0027** (M=5.10^6)

Table 4.15: 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	141	246	461	820
Ratio of ($MC/MISC$)	141	41	17	7

Table 4.16: Comparison of the computational time (in Seconds) of MC and MISC, used to compute call option price of the rBergomi model for different number of time steps. The average MC CPU time is computed over 10 runs.

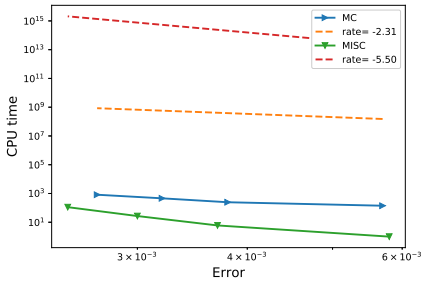


Figure 4.7: Complexity plot for MC and MISC.

4.2.4 Case of set 4 parameters in table 4.1

Method \Steps	2	4	8	16
MISC ($\text{TOL}_{\text{MISC}} = 10^{-1}$)	**0.0914**	**0.0736**	**0.0693**	**0.0654**
MISC ($\text{TOL}_{\text{MISC}} = 10^{-2}$)	**0.0914**	**0.0736**	**0.0223**	**0.0195**
MISC ($\text{TOL}_{\text{MISC}} = 10^{-3}$)	**0.0655**	**0.0334**	**0.0205**	**0.0135**
MC	**0.0657** (M=5.10^6)	**0.0337** (M=5.10^6)	**0.0209** (M=5.10^6)	**0.0136** (M=5.10^6)

Table 4.17: 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	154	229	420	938
Ratio of $(MC/MISC)$	220	57	16	0.5

Table 4.18: Comparison of the computational time (In seconds) of MC and MISC, used to compute call option price of rBergomi model for different number of time steps. The average MC CPU time is computed over 10 runs.

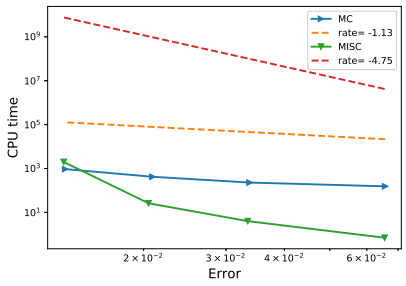


Figure 4.8: Complexity plot for MC and MISC.

5 Conclusions and future work

In this work, we propose a novel hierarchical fast option pricer, based on a hierarchical adaptive sparse grids quadrature, specifically MISC as in [18], coupled with Brownian bridge construction and Richardson extrapolation, for options whose underlyings follow the rBergomi model as in [3].

Given that the only prevalent option, in this context, is to use different variants of MC method, our first contribution is that we 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, to solve pricing and calibration problems related to the rBergomi model. Our second contribution is that we reduce the computational cost, through variance reduction, by using the conditional expectation as in [23] but also through bias reduction through using

Richardson extrapolation. Finally, assuming one targets prices estimates with a sufficiently small error tolerance, our proposed method demonstrated substantial computational gains over standard MC method, when pricing under the rBergomi model. We show these gains through our numerical experiments for different parameters constellations.

In this paper, we limit ourselves to compare our novel proposed method against standard MC. A more systematic comparison against the variant of MC proposed in [23] 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 ... R. Tempone and C. Ben Hammouda are members of the KAUST SRI Center for Uncertainty Quantification at the Computer, Electrical and Mathematical Sciences & Engineering Division at King Abdullah University of Science and Technology (KAUST).

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

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

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

[5] 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.

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

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

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

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

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

[11] 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.

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

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

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

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

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

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

[18] 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.

[19] 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.

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

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

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

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

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

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

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

[27] 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.