IMPERIAL

Imperial College London

Department of Mathematics

Weighted multilevel Monte Carlo method for VIX option pricing

Author: Rafael KYRIAKOU (CID: 02493212)

A thesis submitted for the degree of

MSc in Mathematics and Finance, 2023-2024

Declaration

The work contained in this thesis is my own work unless otherwise stated.

Acknowledgements

Firstly, I would like to thank Dr Ofelia Bonesini for her insightful feedback, and patience as I progressed through the challenges of this thesis. Your passion for the subject and commitment to high standards have greatly inspired me and pushed me to achieve my best work.

I would also like to thank Prof. Antoine Jacquier, whose practical advice have been instrumental in shaping the direction and scope of this thesis. Your ability to clarify complex concepts, and your willingness to discuss ideas at any stage of the process were crucial in helping me overcome the obstacles I encountered.

Abstract

This thesis considers the computational challenges involved in pricing options on the Volatility Index (VIX). The VIX is a crucial financial measure of the market's expectation of future volatility. It is commonly modelled using the rough Bergomi model, which captures the roughness and long memory properties observed in real market data. However, the practical application of the Rough Bergomi model is computationally intensive when using standard Monte Carlo methods due to the need for high accuracy in simulations. To mitigate these challenges, this thesis investigates the use of the Weighted Multilevel Monte Carlo (WMLMC) method, which builds upon the traditional Multilevel Monte Carlo (MLMC) approach by introducing a weighting scheme designed to reduce computational costs. The WMLMC method achieves this by optimally allocating computational resources across different levels of simulation, reducing the number of samples required at coarser levels where variance reduction is more pronounced. The research presented in this thesis includes both a theoretical and a numerical analysis of the computational complexity of the WMLMC. The results demonstrate that while both methods have the same asymptotic complexity $\mathcal{O}(\varepsilon^{-2}\log^2(\varepsilon))$ for the rectangle scheme and $\mathcal{O}(\varepsilon^{-2})$ for the trapezoidal scheme, the WMLMC consistently requires fewer computational resources due to a lower constant factor.

Contents

List of Figures

List of Tables

Introduction

The accurate pricing of financial derivatives is crucial for risk management, trading strategies, and investment decisions. Among these derivatives, options on the Volatility Index (VIX) hold a special place due to their ability to provide insight into market expectations of future volatility. Since its introduction by the Chicago Board Options Exchange (CBOE) in 1993, the VIX has become integral to financial markets, especially for option pricing.

However, pricing VIX options is challenging due to the complex nature of market volatility. Traditional models, like the Black-Scholes model, assume that volatility is constant, which oversimplifies how markets really behave. In reality, market data show patterns of "rough" and long-lasting volatility [11, 7] that these models can't capture effectively. To better address this, the rough Bergomi model was developed [1] which employs a fractional Brownian motion to more accurately reflect the short-term fluctuations and roughness in market volatility. The introduction of the rough Bergomi model marked a significant advancement in this area, offering a more realistic representation of the market's volatility dynamics. However, the computational cost associated with this model, especially when implemented using traditional Monte Carlo simulations, presents a major obstacle to its practical application.

This thesis investigates the use of the Weighted Multilevel Monte Carlo (WMLMC) [13] method as a solution to this problem. The WMLMC method extends the Multilevel Monte Carlo (MLMC) [8] approach by incorporating a weighting scheme designed to optimise computational efficiency [13], particularly in cases where correlations between different levels are not sufficiently high. By reducing variance without increasing the asymptotic complexity, WMLMC offers a promising approach for efficiently pricing VIX options within the rough Bergomi framework.

The structure of this thesis is as follows: Chapter 1 provides a comprehensive overview of VIX option pricing, focusing on the rough Bergomi model and the derivation of the VIX formula. It also covers the time discretisation techniques necessary for simulating VIX samples and discusses the associated strong and weak errors. Chapter 2 delves into Monte Carlo methods, with an emphasis on standard Monte Carlo, MLMC, and the newly proposed WMLMC. Chapter 3 presents the theoretical results of the application of the WMLMC method, including its computational complexity and a comparison with MLMC. Chapter 4 illustrates the numerical results, validating the effectiveness of the WMLMC method. Finally, we conclude the thesis by summarising the findings and discussing potential future research directions.

Chapter 1

VIX Option Pricing

The VIX stands as a pivotal financial metric providing a real-time measure of market expectations for volatility over the coming 30 days. This index is particularly significant as it captures the anticipated fluctuations in the S&P 500 index. The VIX is particularly useful because it reflects the level of uncertainty or risk in the market. When the VIX is high, it usually means that investors are expecting large swings in the stock market, which can indicate fear or concern about future events. Conversely, a low VIX suggests that the market is expected to remain stable. This makes the VIX a valuable tool for both predicting market trends and managing risk.

The CBOE defines the VIX index using the implied variance of a log-contract [5]. In essence, the VIX value at a specific time T represents the implied volatility of a log-contract that delivers $-\frac{2}{\Delta}\ln\left(\frac{S_{T+\Delta}}{S_T}\right)$ at time $T+\Delta$. Here, Δ is 30 days, and S denotes the S&P 500. The formula for calculating the VIX [4, Equation 1] is given by

$$\mathrm{VIX}_T := \sqrt{\mathrm{price}_T^{\mathrm{mkt}}\left(-\frac{2}{\Delta}\ln\left(\frac{S_{T+\Delta}}{S_T}\right)\right)}$$

This means that the VIX index reflects the market's expectation of future volatility, derived from the prices of options on the S&P 500 index.

To accurately price options based on the VIX, it's important to use models that can capture the complex behaviour of market volatility. One such model is the rough Bergomi model, which has been developed to better reflect the roughness observed in real market data. This model uses advanced mathematical techniques to simulate how volatility behaves over time, offering a more realistic approach to option pricing.

In this chapter, we will explore the pricing of VIX options. We will start by discussing the rough Bergomi model and its application to modelling market volatility. Then, we will go through the steps involved in deriving the VIX and discuss the challenges that arise when discretising time in these models.

1.1 Rough Bergomi Model

The rough Bergomi model, introduced in [1], establishes a comprehensive framework for modelling the dynamics of instantaneous forward variances. This model has been extensively studied and validated for its effectiveness in capturing the volatility dynamics observed in financial markets [7, 11]. It was specifically designed to address the limitations of earlier models in capturing the unique characteristics of market volatility. Traditional models, like the Black-Scholes model, assume that volatility is constant or follows a simple stochastic process. However, real-world data show that volatility exhibits a more complex

behaviour, often displaying "roughness" and long-memory effects that these classical models have trouble capturing.

It consists of two main components: the dynamics of the *stock price* $(S_t)_{t\in[0,T]}$ and the dynamics of the *variance process* $(V_t)_{t\in[0,T]}$. The *stock price* S under the rough Bergomi model follows the stochastic differential equation (SDE)

$$dS_t = S_t\sqrt{V_t}\, dW_t, \tag{1.1.1}$$

where

(i) S_t is the stock price at time t,

(ii) V_t is the variance process at time t,

(iii) W_t is a standard Brownian motion.

The rough rough Bergomi model is an extension of the Bergomi model [2, 3]. The stock price S_t under the Bergomi model follows the same dynamics as in (1.1.1), and the variance process V_t is assumed to be stochastic and driven by another Brownian motion W_t^v, correlated with W_t. Specifically, V_t follows the SDE

$$V_t = V_0 \exp\left(\eta W_t^v - \frac{1}{2}\eta^2 t\right),$$

where V_0 is the initial volatility, $\eta > 0$ is the volatility of volatility which controls how much the process V_t fluctuates over time. W_t^v is a standard Brownian motion under $\mathbb{Q}$, correlated with W_t. The rough Bergomi model extends this framework by replacing the standard Brownian motion W_t^v in the variance process with a fractional Brownian motion W_t^H, introducing the concept of rough volatility.

A *fractional Brownian motion* W_t^H with Hurst parameter $H \in (0,1)$ is a continuous-time Gaussian process that satisfies the following properties [16, section 1.2]:

(i) $W_0^H = 0$.

(ii) $\mathbb{E}[W_t^H] = 0$ for all $t \geq 0$.

(iii) For all $s, t \geq 0$, the covariance function of W_t^H W_s^H and is given by

$$\mathbb{C}\text{ov}(W_t^H, W_s^H) = \frac{1}{2}\left(t^{2H} + s^{2H} - |t-s|^{2H}\right).$$

Fractional Brownian motion differs from standard Brownian motion since it includes the Hurst parameter H which controls the degree of roughness in the volatility process. The covariance structure implies that when $H < 0.5$, the increments are negatively correlated meaning an increase in the value of the process during one time interval is likely to be followed by a decrease in the next interval. This anti-persistence means that any influence from a particular movement is quickly neutralised by subsequent movements. The process does not carry the effect of a previous increment for long, which is why the memory of the process is described as "short-term". Similarly, when $H > 0.5$ the increments are positively correlated and the memory of the process can be described as "long-term". When $H = 0.5$, and assuming that $t > s$, then $\mathbb{C}\text{ov}(W_t^H, W_s^H) = \frac{1}{2}\left(t + s - |t-s|\right) = s$. The covariance of a standard Brownian motion is given by $\mathbb{C}\text{ov}(W_t, W_s) = \mathbb{E}[W_s^2] = s$. This is exactly the covariance function of the fractional Brownian motion. Therefore, when $H = 0.5$, fractional Brownian motion reduces to standard Brownian motion, where the increments are independent and identically distributed, following a normal distribution with mean 0 and variance $t - s$. In other words, the standard Brownian motion is a

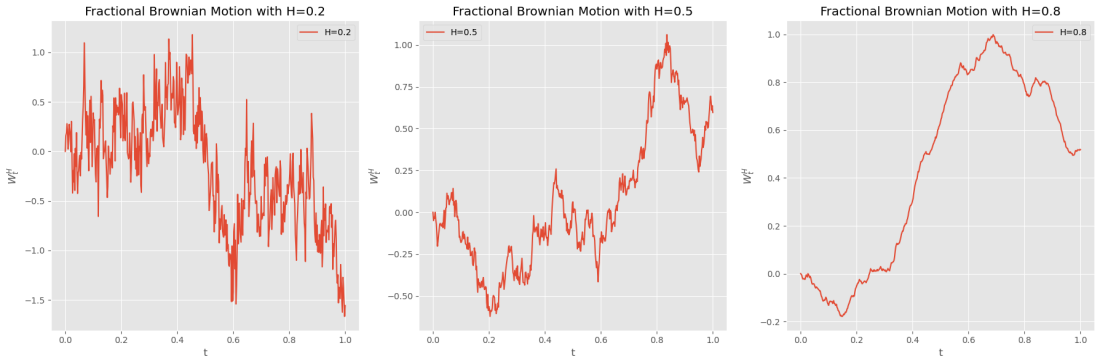

Figure 1.1: Comparison of fractional Brownian motion paths generated with Hurst parameters values: $H = 0.3$, $H = 0.5$, and $H = 0.7$.

particular case of fractional Brownian motion with $H = 0.5$, where the process has no memory.

It is shown in [14] that the fractional Brownian motion satisfying the above properties has the following integral representation

$$W_t^H = \int_{-\infty}^{\infty} h_H(a, b; t, s)\, \mathrm{d}W_s, \quad t \in \mathbb{R},$$

with

$$h_H(a, b; t, s) = a\left[(t-s)_+^{H-1/2} - (-s)_+^{H-1/2}\right] + b\left[(t-s)_-^{H-1/2} - (-s)_-^{H-1/2}\right].$$

Here, a and b are real constants, and x_+, x_- refer to the positive and negative parts of x, respectively. x_+ captures the part of x that is non-negative, while x_- captures the part that is non-positive. For $a = 1$ and $b = 0$ the fractional Brownian motion first introduced by [15] is given by

$$\begin{aligned} W_t^H &= \int_{-\infty}^{\infty} \left[(t-s)_+^{H-1/2} - (-s)_+^{H-1/2}\right] \mathrm{d}W_s \\ &= \int_{-\infty}^{0} \left(|t-s|^{H-1/2} - |s|^{H-1/2}\right) \mathrm{d}B(s) + \int_0^t |t-s|^{H-1/2}\, \mathrm{d}W_s \\ &= \frac{1}{\Gamma\left(H+\frac{1}{2}\right)} \left(\int_{-\infty}^{t} |t-s|^{H-1/2}\, \mathrm{d}B(s) - \int_{-\infty}^{0} |s|^{H-1/2}\, \mathrm{d}W_s\right). \end{aligned}$$

Since we consider $t \geq 0$, we have

$$W_t^H = \frac{1}{\Gamma\left(H+\frac{1}{2}\right)} \int_0^t (t-s)^{H-1/2}\, \mathrm{d}W_s,$$

which is the *Riemann-Liouville* definition of a fractional Brownian motion [14].

The dynamics of the variance process V_t under the rough Bergomi model is expressed as

$$V_t = V_0 \exp\left(\eta W_t^H - \frac{1}{2}\eta^2 t^{2H}\right), \tag{1.1.2}$$

where V_0 is the initial variance, η is the volatility-of-volatility parameter, and W_t^H is a Riemann-Liouville fractional Brownian motion with Hurst parameter H.

In the rough Bergomi model the Hurst parameter ranges between 0 and 0.5, where as seen in figure 1.1, the resulting volatility paths are rougher and exhibit more short-term fluctuations. This characteristic makes the Rough Bergomi model particularly useful

for modelling markets where volatility is highly sensitive to short-term news or events. Empirical studies have validated these parameter choices, showing that they significantly improve the model's accuracy in predicting volatility patterns and market behaviours [1, Chapter 5].

The variance process V_t under the Bergomi model is included in a class of functions of the form

$$X_t^u = X_0^u - \frac{1}{2}\int_0^t K(u,s)^2\,ds + \int_0^t K(u,s)\,dW_s, \quad u \geq t, \tag{1.1.3}$$

where $K(t,s)$ is a kernel function and obtained when you set $K(t,s) = \omega e^{-\kappa(t-s)}$, with $\omega, \kappa > 0$ [4, Equation 2]. According to [1] the kernel $K(t,s)$ corresponding to the rough Bergomi model is

$$K(t,s) = \eta(t-s)^{H-0.5}$$

where $H \in (0, \frac{1}{2})$ and $\eta > 0$. The kernel function plays a pivotal role in the rough Bergomi model. It determines the influence of past volatility on the current variance process. The parameter η controls the magnitude of this influence, while H controls the temporal decay of this influence. The integral form of the variance process V_t, incorporating the kernel function, provides a nuanced and flexible representation of how historical volatility impacts future volatility. The choice of parameters was validated by [11, Chapter 3] for VIX options and by [7, Chapter 3] for reconstructing the observed structure of the volatility skew of the S&P500.

1.2 Derivation of the VIX formula

To further understand the VIX formula, we start with a standard financial model where the price S_t of an asset follows a stochastic process. The asset price S_t under the risk-neutral measure $\mathbb{Q}$ can be represented by the SDE

$$dS_t = \sigma_t S_t dW_t. \tag{1.2.1}$$

Taking the logarithm of the asset price simplifies the analysis. Using Ito's Lemma, we convert the SDE into a form that focuses directly on the logarithm of price changes

$$d(\log S_t) = \sigma_t dW_t - \frac{1}{2}\sigma_t^2 dt. \tag{1.2.2}$$

Integrating (1.2.2) over a specific time interval separates the log returns into stochastic and deterministic components

$$\log S_{T+\Delta} - \log S_T = \int_T^{T+\Delta} \sigma_u\,dW_u - \frac{1}{2}\int_T^{T+\Delta} \sigma_u^2\,du.$$

This step is essential for isolating the random elements from the predictable parts, providing a clearer picture of the underlying processes driving price changes.

Diving both sides by $\frac{-2}{\Delta}$, we get

$$-\frac{2}{\Delta}\log\left(\frac{S_{T+\Delta}}{S_T}\right) = -\frac{2}{\Delta}\left(\int_T^{T+\Delta} \sigma_u\,dW_u - \frac{1}{2}\int_T^{T+\Delta} \sigma_u^2\,du\right).$$

The payoff of a forward log-contract is given by

$$\mathbb{E}_t^{\mathbb{Q}}\left[-\frac{2}{\Delta}\log\left(\frac{S_{T+\Delta}}{S_T}\right) \mid \mathcal{F}_t\right] = \mathbb{E}_t^{\mathbb{Q}}\left[-\frac{2}{\Delta}\left(\int_T^{T+\Delta} \sigma_u\,dW_u - \frac{1}{2}\int_T^{T+\Delta} \sigma_u^2\,du\right) \mid \mathcal{F}_t\right].$$

Since the expectation of a stochastic integral term with respect to a Brownian motion is zero, we are left with

$$\mathbb{E}_t^{\mathbb{Q}}\left[-\frac{2}{\Delta}\log\left(\frac{S_{T+\Delta}}{S_T}\right) \mid \mathcal{F}_t\right] = \mathbb{E}_t^{\mathbb{Q}}\left[\frac{1}{\Delta}\int_T^{T+\Delta}\sigma_u^2\,du \mid \mathcal{F}_t\right].$$

Thus, the forward variance $V_{t,T}^{T+\Delta}$ is defined as the market price of the forward log-contract

$$V_{t,T}^{T+\Delta} := \text{price}_T^{\text{mkt}}\left(-\frac{2}{\Delta}\ln\left(\frac{S_{T+\Delta}}{S_T}\right)\right) = \mathbb{E}_t^{\mathbb{Q}}\left[\frac{1}{\Delta}\int_T^{T+\Delta}\sigma_u^2\,du \mid \mathcal{F}_t\right]. \tag{1.2.3}$$

This variance represents the expected average of the instantaneous variances over the specified time interval, encapsulating the market's forecast of future volatility.

To further describe the forward variance $V_{t,T}^{T+\Delta}$ in (1.2.3) we characterise it similar to the approach in [10], such that we have

$$\xi_t := \mathbb{E}[\sigma_T^2 \mid \mathcal{F}_t],$$

which gives

$$V_{t,T}^{T+\Delta} = \frac{1}{\Delta}\int_T^{T+\Delta}\xi_t^u\,du. \tag{1.2.4}$$

Following the framework and notations from [4] we consider volatility processes under the rough bergomi model of the form

$$\xi_t^u = e^{X_T^u} \quad \text{with} \quad X_T^u = X_0^u - \frac{1}{2}\int_0^T K(u,s)^2\,ds + \int_0^T K(u,s)\,dW_s, \quad u \geq t, \tag{1.2.5}$$

with kernel

$$K(u,s) = \eta(u-s)^{H-0.5}. \tag{1.2.6}$$

Finally, the squared VIX index at time T is

$$(\text{VIX}_T)^2 = \frac{1}{\Delta}\int_T^{T+\Delta}e^{X_T^u}\,du. \tag{1.2.7}$$

and the price at time zero of an option on the squared VIX with a payoff function φ, and at maturity T is represented as the expectation

$$\mathbb{E}\left[\varphi\left(\text{VIX}_T^2\right)\right] = \mathbb{E}\left[\varphi\left(\frac{1}{\Delta}\int_T^{T+\Delta}e^{X_T^u}du\right)\right]. \tag{1.2.8}$$

In the analysis conducted in this thesis, we consider the standard payoff functions of call and put options, given by $\varphi_{\text{call}}(x) = (\sqrt{x}-K)^+$ and $\varphi_{\text{put}}(x) = (K-\sqrt{x})^+$, where K is the strike price.

1.3 Time Discretisation

The model used to describe the squared VIX in (1.2.7), as mentioned in [4] cannot be solved analytically. Thus, discrete approximations are necessary. In this section, we explore two common time discretisation schemes, the rectangle scheme and the trapezoidal scheme. These methods provide different ways to approximate the integral (1.2.7), with each having its own advantages and potential drawbacks in terms of accuracy and computational efficiency. Through the thesis, we use $\mathcal{D} = \mathcal{R}$ to denote the rectangle scheme and $\mathcal{D} = \mathcal{T}$ to denote the trapezoidal scheme.

1.3.1 Rectangle Scheme

The right-point rectangle scheme, also known as the right Riemann sum, approximates the integral of a function by summing up the areas of rectangles whose heights are determined by the function's value at the right endpoint of each sub-interval.

Definition 1.3.1 (Right-point rectangle scheme)**.** [6, Section 2.1] Given a function $f : [a, b] \to \mathbb{R}$ defined on the interval $[a, b]$, the right-point rectangle scheme over $n \in \mathbb{N}$ points is defined as

$$\mathcal{R}_n = \sum_{i=1}^{n} f(x_i)\Delta x, \tag{1.3.1}$$

where

(i) $\Delta x = \frac{b-a}{n}$ is the width of each sub-interval.

(ii) $x_i = a + i\Delta x$ are the right endpoints of the sub-intervals, for $i = 1, 2, \ldots, n$.

Given that f is bounded and continuous almost everywhere, then the integral of $f(x)$ from a to b is given as the limit of the Riemann sums as the number of sub-intervals n approaches infinity [6, Section 1.5]. Specifically,

$$\int_a^b f(x)\,dx = \lim_{n\to\infty} \mathcal{R}_n = \lim_{n\to\infty} \sum_{i=1}^{n} f(x_i)\Delta x.$$

For the right-point rectangle scheme, we approximate the integral in (1.2.7) by dividing the interval $[T, T + \Delta]$ into n sub-intervals of equal width $\Delta x = \frac{\Delta}{n}$, and use the function value at the right endpoint of each sub-interval. Thus we define the grid points

$$u_i = T + i\Delta x, \quad \text{for} \quad i = 1, 2, \ldots, n, \tag{1.3.2}$$

where $\Delta x = \frac{\Delta}{n}$.

The right-point rectangle approximation for the integral is

$$\frac{1}{\Delta}\int_T^{T+\Delta} e^{X_T^u}\,du \approx \frac{1}{\Delta}\sum_{i=1}^{n} e^{X_T^{u_i}}\Delta x.$$

Substituting $\Delta x = \frac{\Delta}{n}$ and simplifying

$$\frac{1}{\Delta}\sum_{i=1}^{n} e^{X_T^{u_i}}\Delta x = \frac{1}{\Delta}\left(\frac{\Delta}{n}\sum_{i=1}^{n} e^{X_T^{u_i}}\right).$$

Therefore, the right-point rectangle scheme for VIX_T^2 is:

$$\mathrm{VIX}_T^{2,\mathcal{R}_n} = \frac{1}{n}\sum_{i=1}^{n} e^{X_T^{u_i}}. \tag{1.3.3}$$

1.3.2 Trapezoidal Scheme

The Trapezoidal Scheme approximates the integral of a function by summing the areas of trapezoids rather than rectangles. The bases of each trapezoid are determined by the function values at the endpoints of each sub-interval, and the height is the width of the sub-interval.

Definition 1.3.2. [6, Section 2.1] Given a function $f : [a, b] \to \mathbb{R}$ defined on the interval $[a, b]$, the trapezoidal rule over $n \in \mathbb{N}$ points is defined as

$$\mathcal{T}_n = \sum_{i=1}^{n} \frac{1}{2} \left(f(x_i) + f(x_{i-1}) \right) \Delta x, \tag{1.3.4}$$

where

(i) $\Delta x = \frac{b-a}{n}$ is the width of each sub-interval,

(ii) $x_i = a + i\Delta x$ are the endpoints of the sub-intervals, for $i = 0, 1, 2, \ldots, n$.

For the trapezoidal scheme, we approximate the integral in (1.2.7) similarly as in the rectangle scheme. We use the same grid points defined in (1.3.2), and thus the approximation for the integral is

$$\frac{1}{\Delta} \int_T^{T+\Delta} e^{X_T^u} \, du \approx \frac{1}{\Delta} \sum_{i=1}^{n} \frac{1}{2} \left(e^{X_T^{u_i}} + e^{X_T^{u_{i-1}}} \right) \Delta x.$$

Substituting $\Delta x = \frac{\Delta}{n}$ and simplifying we get that

$$\mathrm{VIX}_T^{2,\mathcal{T}_n} = \frac{1}{2n} \sum_{i=1}^{n} e^{X_T^{u_i}} + e^{X_T^{u_{i-1}}}. \tag{1.3.5}$$

1.4 VIX Simulation

It is shown in [4, Section 2] that $\mathrm{VIX}_T^{2,\mathcal{D}_n}$ can be simulated exactly. In financial modelling, exact simulation means creating sample paths of stochastic processes that accurately reflect the theoretical model's properties, without introducing errors from approximations. In other words, the distribution of the simulated samples match the distribution of the model. For $\mathrm{VIX}_T^{2,\mathcal{D}_n}$, an exact simulation ensures that the generated paths precisely follow the dynamics of the rough volatility model.

Considering $\mathrm{VIX}_T^{2,\mathcal{D}_n}$ is defined with discretisation schemes in equations (1.3.3) and (1.3.5) under the rough Bergomi model given in (1.2.5), the exact simulation involves generating samples from a multivariate Gaussian distribution where the mean and covariance structures are derived directly from the theoretical model. Thus, to sample $\mathrm{VIX}_T^{2,\mathcal{D}_n}$ we need to sample n samples of $X_T^{u_i}$ for $i = 0, 1, 2, \ldots, n$.

Recall that under the rough Bergomi model given in (1.2.5) we have

$$\xi_T^{u_i} = e^{X_T^{u_i}} \quad \text{with} \quad X_T^{u_i} = X_0^{u_i} - \frac{1}{2} \int_0^T K(u_i, s)^2 \, ds + \int_0^T K(u_i, s) \, dW_s, \quad u \geq t$$

Expectation:

$$\begin{aligned} \mathbb{E}\left[X_T^{u_i} \right] &= \mathbb{E}\left[X_0^{u_i} - \frac{1}{2} \int_0^T K(u_i, s)^2 \, ds + \int_0^T K(u_i, s) \, dW_s \right] \\ &= \mathbb{E}\left[X_0^{u_i} - \frac{1}{2} \int_0^T K(u_i, s)^2 \, ds \right]. \end{aligned}$$

Since $K(u_i, s)$ is a deterministic kernel we get

$$\mu_i := \mathbb{E}\left[X_T^{u_i} \right] = X_0^{u_i} - \frac{1}{2} \int_0^T K(u_i, s)^2 \, ds = X_0^{u_i} - \frac{\eta^2}{4H} \left(u_i^{2H} - (u_i - T)^{2H} \right). \tag{1.4.1}$$

Covariance:

$$\begin{aligned}\mathbb{C}\text{ov}\left(X_T^{u_i}, X_T^{u_j}\right) &= \mathbb{E}\left[\left(X_T^{u_i} - \mathbb{E}\left[X_T^{u_i}\right]\right)\left(X_T^{u_j} - \mathbb{E}\left[X_T^{u_j}\right]\right)\right] \\ &= \mathbb{E}\left[\int_0^T K(u_i, s)\, dW_s \int_0^T K(u_j, s)\, dW_s\right].\end{aligned}$$

Using the Ito isometry we can rewrite this as

$$\begin{aligned}\mathbb{E}\left[\int_0^T K(u_i, s)\, dW_s \int_0^T K(u_j, s)\, dW_s\right] &= \mathbb{E}\left[\int_0^T K(u_i, s)K(u_j, s)\, ds\right] \\ &= \int_0^T K(u_i, s)K(u_j, s)\, ds.\end{aligned}$$

Thus, we set

$$\Sigma_{i,j} := \mathbb{C}\text{ov}\left(X_T^{u_i}, X_T^{u_j}\right) = \int_0^T K(u_i, s)K(u_j, s)\, ds. \tag{1.4.2}$$

For later simulations, when $i = j$ the terms can easily be computed as

$$\int_0^T K(u_i, s)K(u_j, s)\, ds = \int_0^T K(u_i, s)^2\, ds = \frac{\eta^2}{2H}\left(u_i^{2H} - (u_i - T)^{2H}\right),$$

and when $i < j$, we use the function `scipy.integrate.quad` from the `SciPy` library [17].

Given that the rough Bergomi model is a multivariate Gaussian distribution with mean and covariance structures in Equations (1.4.1) and (1.4.2) respectively, to generate X_T^u we use

$$X_T^u = \mu + LZ$$

where $Z = (Z_1, \ldots, Z_n)^T$ is a vector where each Z_i is an independent standard random variable ($Z_i \sim \mathcal{N}(0, 1)$), $\mu = (\mu_1, \ldots, \mu_n)^T$, and L is a lower triangular matrix such that $\Sigma = LL^T$.

To obtain L we decompose the covariance matrix Σ using the Cholesky decomposition. The Cholesky decomposition is a method to factorise Σ into the product of a lower triangular matrix L and its transpose, such that $\Sigma = LL^T$. Given a symmetric positive-definite matrix Σ of size $n \times n$, the Cholesky decomposition is computed as follows [9, Chapter 4]:

(i) Start with the matrix L initialised as a zero matrix with the same size as Σ.

(ii) For each i from 1 to n:

$$L_{ii} = \sqrt{\Sigma_{ii} - \sum_{k=1}^{i-1} L_{ik}^2}$$

where the sum term accounts for the contributions from previously computed elements of L.

(iii) For each j from $i + 1$ to n:

$$L_{ji} = \frac{1}{L_{ii}}\left(\Sigma_{ji} - \sum_{k=1}^{i-1} L_{jk}L_{ik}\right).$$

These steps ensure that the lower triangular matrix L satisfies the condition $\Sigma = LL^\top$.

Finally, we discuss the cost of generating a sample of $\text{VIX}_T^{2,\mathcal{D}_n}$. The Cholesky decomposition itself is an $\mathcal{O}(n^3)$ operation in terms of computational complexity, where n is the

size of the matrix. However, since this decomposition is performed offline, its cost is not included in the simulation cost. Instead, the focus is on the cost of using this decomposition to generate each sample. Once L is computed, generating a single sample of the $(n+1)$-dimensional Gaussian vector X involves multiplying the matrix L by the vector Z, which requires $\mathcal{O}(n^2)$ operations. This is because matrix-vector multiplication involves performing a dot product for each row of the matrix, and there are $n+1$ rows in the matrix L. After generating the Gaussian vector $(X_T^{u_i})_{i=0,\dots,n}$, the next step is to discretise the integral that defines VIX_T^2 in (1.2.7). The integral is approximated as a sum over the discretisation points, with the choice of discretisation scheme affecting both the accuracy and computational cost. The rectangle scheme is simple to implement and has a computational cost of $\mathcal{O}(n)$ for evaluating the sum, where n is the number of discretisation points. The trapezoidal scheme generally requires the same $\mathcal{O}(n)$ computational cost for evaluating the sum, but it produces a more accurate approximation due to its use of both endpoints. Thus, the cost per sample of $\mathrm{VIX}_T^{2,\mathcal{D}_n}$ is $\mathcal{O}(n^2)+\mathcal{O}(n)$, where the dominant term is $\mathcal{O}(n^2)$.

1.5 Strong and Weak Errors

Definition 1.5.1 (Strong convergence of order α)**.** [12, Equation 6.3] Strong convergence refers to the convergence of a numerical approximation. A discrete time approximation $\hat{X}_n$ with step-size n converges strongly with an order $\alpha > 0$ if there exists $c > 0$ such that

$$\mathbb{E}\left[\left|X - \hat{X}_n\right|^p\right] = O\left(n^{-\alpha}\right)$$

for some $p \geq 1$, and $\mathcal{O}\left(n^{-\alpha}\right) \leq cn^{-\alpha}$.

For the rectangle scheme, it has been shown in [10, Proposition 2] that the strong convergence rate is $\alpha = 1$, for $p = 1$, and later extended by [4, Propositon 2] for $p \geq 1$. This indicates that the strong error of the rectangle scheme decreases linearly as the number of discretisation steps increases. Formally, this is expressed as

$$\left(\mathbb{E}\left[\left|\mathrm{VIX}_T^2 - \mathrm{VIX}_T^{2,R_n}\right|^p\right]\right)^{\frac{1}{p}} = \mathcal{O}\left(\frac{1}{n}\right). \tag{1.5.1}$$

While this provides a basic level of accuracy, the convergence rate is relatively slow, which may necessitate a large number of discretisation steps to achieve acceptable accuracy. As shown in [4, Proposition 2] the trapezoidal scheme improves upon the rectangle scheme by achieving a strong convergence rate of $\alpha = 1 + H$, where H is the Hurst parameter associated with the rough volatility model. This improvement is represented as

$$\left(\mathbb{E}\left[\left|\mathrm{VIX}_T^2 - \mathrm{VIX}_T^{2,T_n}\right|^p\right]\right)^{\frac{1}{p}} = \mathcal{O}\left(\frac{1}{n^{1+H}}\right), \tag{1.5.2}$$

as $n \to \infty$. This improved convergence rate allows for more accurate approximations with fewer discretisation steps, making the trapezoidal scheme more effective in models where high precision is required.

Definition 1.5.2 (Weak convergence of order β)**.** [12, Equation 7.4] Weak convergence refers to the convergence of the distribution of the numerical approximation. A discrete time approximation $\hat{X}_n$ with step-size n converges weakly of order $\beta > 0$ if for all $f \in C^{2\beta+2}$, there exists a $c > 0$ such that

$$\left|\mathbb{E}\left[f(X)\right] - \mathbb{E}[f(\hat{X}_n)\right| = \mathcal{O}(n^{-\beta})$$

where $\mathcal{O}(n^{-\beta}) \leq cn^{-\beta}$. Here $C^{2\beta+2}$ denotes the number $2\beta+2$ of times $f : \mathbb{R}^d \to \mathbb{R}$ is differentiable.

A direct consequence of [4, Proposition 2] yields for the rectangle scheme, a weak convergence rate of $\beta = 1$. This implies that as we increase the number of discretisation points n, the error in the expected value of the function $f(\mathrm{VIX}_T^2)$, where VIX_T^2 is simulated using the rectangle scheme, decreases proportionally to $\frac{1}{n}$.

$$\left|\mathbb{E}\left[\varphi\left(\mathrm{VIX}_T^2\right) - \varphi\left(\mathrm{VIX}_T^{2,R_n}\right)\right]\right| = \mathcal{O}\left(\frac{1}{n}\right), \tag{1.5.3}$$

This means that for each doubling of the number of discretisation points, the weak error is halved. While this rate of convergence is sufficient for many practical purposes, it can be limiting when high precision is necessary. This is because it would require a very large number of discretisation points to achieve a small weak error.

Similarly, the trapezoidal scheme offers a more refined approximation by averaging the function values at both endpoints of each discretisation interval. This leads to an improved weak convergence rate of $\beta = 1 + H$.

$$\left|\mathbb{E}\left[\varphi\left(\mathrm{VIX}_T^2\right) - \varphi\left(\mathrm{VIX}_T^{2,T_n}\right)\right]\right| = \mathcal{O}\left(\frac{1}{n^{1+H}}\right). \tag{1.5.4}$$

The trapezoidal scheme's provides a more balanced and accurate representation of the process over each discretisation interval. This balanced approach reduces the overall bias in the approximation, leading to faster convergence.

Chapter 2

Monte Carlo Methods

2.1 Standard Monte Carlo

The fundamental idea behind the Monte Carlo method is to use randomness to solve problems that might be deterministic in principle. By repeatedly sampling random variables and averaging the results, the method approximates the desired quantities with high accuracy, given sufficient samples.

Definition 2.1.1. Let X be a random variable with probability distribution P. The goal is to estimate $\mu = \mathbb{E}[f(X)]$, the expected value of some function f of X. The Monte Carlo estimator for μ using M samples is given by

$$\hat{\mu}_M = \frac{1}{M} \sum_{i=1}^{M} f(X_i),$$

where X_i are i.i.d. samples drawn from P.

Thus, the Monte Carlo estimator for the expectation $\mathbb{E}\left[\varphi(\mathrm{VIX}_T^2)\right]$ is defined as

$$\widehat{P}^{D_n} = \frac{1}{M} \sum_{m=1}^{M} \varphi\left(\mathrm{VIX}_{T,m}^{2,D_n}\right), \tag{2.1.1}$$

where $\left(\mathrm{VIX}_{T,m}^{2,D_n}\right)_{1\leq m\leq M}$ are i.i.d samples of VIX_T^{2,D_n}.

Now, we will examine the mean squared error (MSE) of the Monte Carlo estimator $\widehat{P}^{D_n}$. The MSE is an essential metric used to measure the accuracy of an estimator. It is defined as the expected value of the square of the difference between the estimated value and the true value, and it is a combination of both bias and variance of the estimator. More precisely, it is given by

$$\begin{aligned} \mathrm{MSE} &= \mathbb{E}\left[\left(\mathbb{E}\left[\varphi\left(\mathrm{VIX}_T^2\right)\right] - \widehat{P}^{D_n}\right)^2\right] \\ &= \underbrace{\mathbb{E}\left[\left(\varphi\left(\mathrm{VIX}_T^2\right) - \varphi\left(\mathrm{VIX}_T^{2,D_n}\right)\right)\right]^2}_{(\mathrm{Bias})^2} + \underbrace{\mathbb{V}\mathrm{ar}\left(\widehat{P}^{D_n}\right)}_{\mathrm{Variance}}, \end{aligned} \tag{2.1.2}$$

where the variance of estimator is given by

$$\mathbb{V}\mathrm{ar}\left(\widehat{P}^{D_n}\right) = \frac{1}{M}\mathbb{V}\mathrm{ar}\left(\varphi\left(\mathrm{VIX}_T^{2,D_n}\right)\right)$$

As in Equation (2.1.2) the MSE can be decomposed into two parts:

(i) *Bias Error*: The error due to the difference between the exact value $\varphi\left(\mathrm{VIX}_T^2\right)$ and the approximation $\varphi\left(\mathrm{VIX}_T^{2,D_n}\right)$.

(ii) *Statistical Error*: The error due to the variance of the estimator.

For the estimator $\widehat{P}^{D_n}$ to be optimal, we want the MSE to smaller than ε^2, where $\varepsilon > 0$ is a given threshold. For an optimal bias-variance trade-off we impose

$$\begin{aligned} \mathbb{V}\mathrm{ar}\left(\widehat{P}^{D_n}\right) &\leq \frac{\varepsilon^2}{2}, \quad \text{which will give us the optimal samples } M, \\ \mathbb{E}\left[\left(\varphi\left(\mathrm{VIX}_T^2\right) - \varphi\left(\mathrm{VIX}_T^{2,D_n}\right)\right)\right]^2 &\leq \frac{\varepsilon^2}{2}, \quad \text{which will give us the steps } n. \end{aligned}$$

It was shown in [4, Section 3.1] that using the weak errors given in equations (1.5.3) and (1.5.4), we obtain the optimal discretisation steps n. For the rectangle scheme we set $n = \mathcal{O}\left(\varepsilon^{-1}\right)$, and for the trapezoidal scheme we set $n = \mathcal{O}\left(\varepsilon^{-\frac{1}{1+H}}\right)$. It was also shown that the optimal samples needed to achieve an MSE smaller than ε are $M = \mathcal{O}\left(\varepsilon^{-2}\right)$, for both discretisation schemes.

Finally, the computational complexity of the standard Monte Carlo method is given by the cost of generating a sample of VIX_T^{2,D_n} multiplied by the number of samples M. Thus for the rectangle scheme we have

$$\mathrm{Cost}_{\mathrm{MC}}^{\mathcal{R}} = \mathcal{O}(\varepsilon^{-2}) \times \mathcal{O}(n^2) = \mathcal{O}(\varepsilon^{-2}) \times \mathcal{O}(\varepsilon^{-2}) = \mathcal{O}(\varepsilon^{-4}).$$

This indicates that as the desired accuracy ε becomes smaller, the computational cost increases rapidly. Specifically, halving the error tolerance, results in the computational cost increasing by a factor of 16. This is a significant computational burden, especially for very small ε, which is often required in financial simulations where high precision is necessary. While the rectangle scheme is straightforward to implement, its inefficiency makes it less attractive for scenarios where high accuracy is needed. Therefore, this result is generally not good for practical applications unless computational resources are abundant or the required accuracy is relatively low. For the trapezoidal scheme the total cost is given by

$$\mathrm{Cost}_{\mathrm{MC}}^{\mathcal{T}} = \mathcal{O}(\varepsilon^{-2}) \times \mathcal{O}(n^2) = \mathcal{O}(\varepsilon^{-2}) \times \mathcal{O}\left(\varepsilon^{-2\frac{1}{1+H}}\right) = \mathcal{O}\left(\varepsilon^{-2(1+\frac{1}{1+H})}\right).$$

This shows a significant improvement over the rectangle scheme. For typical values of H (e.g., $H \approx 0.1$), the cost scaling is better than $\mathcal{O}(\epsilon^{-4})$, often closer to $\mathcal{O}(\epsilon^{-3})$. This means that the computational cost increases less rapidly with a decreasing ϵ, making it a more efficient choice. Although, this complexity is still far from being optimal.

2.2 Multilevel Monte Carlo

The Multilevel Monte Carlo (MLMC) method represents a significant advancement in computational finance and numerical simulations, offering a more efficient alternative to traditional Monte Carlo methods. The core idea behind MLMC is to decompose the problem across multiple levels of resolution. By doing so, it capitalises on the fact that coarse simulations are cheaper to compute but less accurate, while fine simulations are more accurate but computationally expensive. By combining these different levels, MLMC achieves a balance, reducing the overall computational cost while maintaining high accuracy.

2.2.1 MLMC Simulation

We first adopt the notation from [4]. We consider $L \in \mathbb{N}^*$ and define the sequences $\mathbf{n} = (n_0, \ldots, n_L)$ and $\mathbf{M} = (M_0, \ldots, M_L)$, where $\mathbf{n}$ represents a series of time steps and $\mathbf{M}$ represents the corresponding Monte Carlo sample sizes. These sequences are structured so that the number of time steps and sample sizes increase across levels. For convenience, we define

$$P_\ell^{\mathcal{D}} := \varphi\left(\mathrm{VIX}_T^{2,\mathcal{D}_{n_\ell}}\right). \tag{2.2.1}$$

for each $\ell = 0, \ldots, L$. In this context, ℓ denotes the discretisation level, and $P_\ell^{\mathcal{D}}$ serves as an approximation of the VIX option payoff $\varphi(\mathrm{VIX}_T^2)$ at the specified level ℓ and discretisation scheme $\mathcal{D}$. Starting with an initial value $n_0 \in \mathbb{N}^*$, we set

$$n_\ell = n_0 2^\ell, \quad \ell = 0, \ldots, L$$

This doubling pattern implies that the number of time steps is doubled at each successive level. We now combine what is above with the notation from [13]. We define for $\ell = 0, \ldots, L$

$$Y_\ell := P_\ell^{\mathcal{D}} - P_{\ell-1}^{\mathcal{D}} \quad \text{with} \quad Y_0 := P_0^{\mathcal{D}}, \tag{2.2.2}$$

$$\eta_\ell^2 := \mathrm{Cost}(Y_\ell) = n_\ell^2, \tag{2.2.3}$$

$$\Delta_\ell^2 := \mathbb{V}\mathrm{ar}(Y_\ell). \tag{2.2.4}$$

where $\mathrm{Cost}(Y_\ell)$ is the cost of generating Y_ℓ. The MLMC method was first presented in [8] and later applied for $P_\ell^{\mathcal{D}}$ in [4]. The expectation of the finest level $P_L^{\mathcal{D}}$ is given by

$$\mathbb{E}\left[P_L^{\mathcal{D}}\right] = \mathbb{E}\left[P_0^{\mathcal{D}}\right] + \sum_{\ell=1}^{L} \mathbb{E}\left[P_\ell^{\mathcal{D}} - P_{\ell-1}^{\mathcal{D}}\right]. \tag{2.2.5}$$

Using standard Monte Carlo to approximate $\mathbb{E}\left[P_0^{\mathcal{D}}\right]$ and each term $\mathbb{E}\left[P_\ell^{\mathcal{D}} - P_{\ell-1}^{\mathcal{D}}\right]$, results in the multilevel estimator for the expectation $\mathbb{E}\left[\varphi(\mathrm{VIX}_T^2)\right]$ given by

$$\widehat{P}_L^{\mathcal{D}} := \frac{1}{M_0^{\mathcal{D}}} \sum_{m=1}^{M_0^{\mathcal{D}}} P_0^{\mathcal{D},m} + \sum_{\ell=1}^{L} \left(\frac{1}{M_\ell^{\mathcal{D}}} \sum_{m=1}^{M_\ell^{\mathcal{D}}} \left(P_\ell^{\mathcal{D},m} - P_{\ell-1}^{\mathcal{D},m} \right) \right), \tag{2.2.6}$$

Now, we will examine the mean squared error (MSE) of the MLMC, which is given by

$$\mathrm{MSE} = \mathbb{E}\left[\left(\widehat{P}_L^{\mathcal{D}} - E[P]\right)^2\right] = \underbrace{\left(E[P_L^{\mathcal{D}}] - E[P]\right)^2}_{(\mathrm{Bias})^2} + \underbrace{\mathrm{Var}\left(\widehat{P}_L^{\mathcal{D}}\right)}_{\mathrm{Variance}} \tag{2.2.7}$$

where the variance of estimator is given by

$$\mathrm{Var}\left(\widehat{P}_L^{\mathcal{D}}\right) = \sum_{\ell=0}^{L} \frac{\mathrm{Var}\left(Y_\ell\right)}{M_\ell} = \sum_{\ell=0}^{L} \frac{\Delta_\ell^2}{M_\ell}$$

For the estimator $\hat{P}_L^{\mathcal{D}}$ to be optimal, we want the MSE to smaller than ε^2, where $\varepsilon > 0$ is a given threshold. For an optimal bias-variance trade-off we impose

(i) $\mathrm{Var}\left(\widehat{P}_L^{\mathcal{D}}\right) \leq \frac{\varepsilon^2}{2}$, which will give us the optimal samples $M_\ell^{\mathcal{D}}$, $\ell = 0, \ldots, L_{\mathcal{D}}$,

(ii) $\left(E[P_L^{\mathcal{D}}] - E[P]\right)^2 \leq \frac{\varepsilon^2}{2}$, which will give us the optimal levels $L_{\mathcal{D}}$.

2.2.2 Computational Complexity

The computational complexity of the MLMC under the rectangle and trapezoidal scheme was derived in [4, Theorem 8]. The theorem is as follows.

Theorem 2.2.1. *Assume that the function φ is Lipschitz, and let n_0 be a positive natural number. For any given tolerance $\varepsilon > 0$, there exists an initial sample size $M_0^{\mathcal{R}}$ and a number of levels $L_{\mathcal{R}}$ for the rectangle scheme ($M_0^{\mathcal{T}}$ and $L_{\mathcal{T}}$ for the trapezoidal scheme) such that the corresponding multilevel estimator $\widehat{P}_L^{\mathcal{R}}$ (and $\widehat{P}_L^{\mathcal{T}}$), defined in equation (2.2.6), achieves an $MSE < \epsilon^2$. with computational complexity $\mathcal{O}(\ln(\varepsilon)\varepsilon^{-2})$ (and $\mathcal{O}(\varepsilon^{-2})$ respectively). For the rectangle scheme, we set*

$$n_\ell = n_0 2^\ell, \quad M_\ell^{\mathcal{R}} = M_0^{\mathcal{R}} 2^{-2\ell}, \quad \textit{for all} \quad \ell = 0, \dots, L_{\mathcal{R}}, \tag{2.2.8}$$

and for the trapezoidal scheme, we set

$$n_\ell = n_0 2^\ell, \quad M_\ell^{\mathcal{T}} = M_0^{\mathcal{T}} 2^{-(2+H)\ell}, \quad \textit{for all} \quad \ell = 0, \dots, L_{\mathcal{T}}. \tag{2.2.9}$$

The theorem was proved using the MLMC complexity theorem in [8, Theorem 1]. Adopting the notation in [13] as before, we present the theorem.

Theorem 2.2.2 (MLMC Complexity theorem)**.** *Suppose that there are positive constants $c_1, c_2, c_3, \alpha, \beta, \gamma$ with $2\alpha \geq \min(\beta, \gamma)$ such that, for $\ell = 0, 1, 2, \dots L$,*

(i) $\left|\mathbb{E}[P_\ell^{\mathcal{D}} - P]\right| \leq c_1 2^{-\alpha\ell}$,

(ii) $\Delta_\ell^2 \leq c_2 2^{-\beta\ell}$,

(iii) $\eta_\ell^2 \leq c_3 2^{\gamma\ell}$.

Then there exists a positive constant c_4 such that, for any $\varepsilon < 1/e$ there exists an integer L such that $\widehat{P}_L^{\mathcal{D}}$ (as defined by (2.2.6)) has mean squared error with bound

$$\mathbb{E}[(\widehat{P}_L^{\mathcal{D}} - \mathbb{E}[P])^2] < \varepsilon^2,$$

with a computational complexity C with bound

$$C \leq \begin{cases} c_4 \varepsilon^{-2}, & \beta > \gamma, \\ c_4 \varepsilon^{-2} (\log \varepsilon)^2, & \beta = \gamma, \\ c_4 \varepsilon^{-2 - \frac{\gamma - \beta}{\alpha}}, & \beta < \gamma. \end{cases}$$

The parameter α reflects the rate at which the bias, which is the error due to approximation decreases as the level of discretisation becomes finer. Specifically, α indicates how fast the bias converges to zero as ℓ increases. A higher value of α means that the bias decreases rapidly, allowing for fewer levels or a coarser discretisation to achieve a given accuracy, thus reducing computational cost. The parameter β describes the rate of decay of the variance of the difference between successive levels of discretisation as the level ℓ increases. A larger β results in faster variance reduction, which enables fewer samples to be required at finer levels, thereby lowering the overall computational effort. The parameter γ is associated with the computational cost of generating a sample at each level ℓ. Specifically, γ indicates how the cost of computation scales as ℓ gets larger. Higher values of γ imply that the cost increases significantly with each finer level. These three parameters collectively influence the efficiency of the MLMC method by controlling the trade-off between bias, variance, and computational cost across different levels of approximation.

For the rectangle scheme, it is shown in [4] that $\alpha = 1$ derived using the weak error in (1.5.3), $\beta = 2$ derived using the strong error in (1.5.1), and $\gamma = 2$ since the cost of generating a sample of Y_ℓ is $\mathcal{O}\left(n^2\right)$. Thus, we observe that $2\alpha \geq \min(\beta, \gamma)$ and $\beta = \gamma$, which gives us a computational complexity of $\mathcal{O}\left(\ln(\epsilon)^2\epsilon^{-2}\right)$.

Similarly for the trapezoidal scheme, it is shown in [4] that $\alpha = 1+H$ derived using the weak error in (1.5.4), $\beta = 2(1+H)$ derived using the strong error in (1.5.2), and $\gamma = 2$ since the cost of generating a sample of Y_ℓ is the same for both discretisation schemes. Thus, we observe that $2\alpha \geq \min(\beta, \gamma)$ and $\beta > \gamma$, which gives us a computational complexity of $\mathcal{O}\left(\epsilon^{-2}\right)$.

To better understand how equations (2.2.8) and (2.2.9) are derived we first introduce E_L^2 as the total cost of computing $\hat{P}_L^{\mathcal{D}}$ where

$$E_L^2 = \sum_{\ell=0}^{L} M_\ell \, \eta_\ell^2. \tag{2.2.10}$$

To find the optimal samples we use Lagrange multipliers to solve the constrained optimisation problem

$$\min_{M_0,\ldots,M_L} E_L^2 \quad \text{such that} \quad \text{Var}\left(\widehat{P}_L^{\mathcal{D}}\right) \leq \frac{\varepsilon^2}{2},$$

where we attempt to minimise the cost for a given variance. Thus, the Lagrangian can be expressed as

$$\mathcal{L} = \sum_{\ell=0}^{L} M_\ell^{\mathcal{D}} \eta_\ell^2 + \lambda \left(\sum_{\ell=0}^{L} \frac{\Delta_\ell^2}{M_\ell^{\mathcal{D}}} - \frac{\varepsilon^2}{2} \right). \tag{2.2.11}$$

We have

$$\frac{\partial \mathcal{L}}{\partial M_\ell^{\mathcal{D}}} = 0 \iff \eta_\ell^2 - \lambda \frac{\Delta_\ell^2}{\left(M_\ell^{\theta,\mathcal{D}}\right)^2} = 0$$

Solving for $M_\ell^{\mathcal{D}}$ we get

$$M_\ell^{\mathcal{D}} = \sqrt{\lambda} \sqrt{\frac{\Delta_\ell^2}{\eta_\ell^2}} = \sqrt{\lambda} \frac{\Delta_\ell}{\eta_\ell} \tag{2.2.12}$$

Substituting this to the variance term of (2.2.11) gives us the Lagrange multiplier λ. Specifically we get,

$$\sum_{\ell=0}^{L} \frac{\Delta_\ell^2}{M_\ell^{\mathcal{D}}} = \frac{\varepsilon^2}{2} \Rightarrow \sum_{\ell=0}^{L} \frac{\Delta_\ell^2}{\sqrt{\lambda}\sqrt{\frac{\Delta_\ell^2}{\eta_\ell^2}}} = \frac{\varepsilon^2}{2}.$$

Rearranging the terms, gives us

$$\sqrt{\lambda} = \frac{2}{\varepsilon^2} \sum_{\ell=0}^{L} \sqrt{\Delta_\ell^2 \eta_\ell^2} = \frac{2}{\varepsilon^2} \sum_{\ell=0}^{L} \Delta_\ell \eta_\ell.$$

Substituting the Lagrange multiplier λ into equation (2.2.12), we get

$$M_\ell^{\mathcal{D}} = \left(\frac{2}{\varepsilon^2} \sum_{\ell=0}^{L} \Delta_\ell \eta_\ell \right) \frac{\Delta_\ell}{\eta_\ell}.$$

Denote $E_L = \left(\frac{2}{\varepsilon^2} \sum_{\ell=0}^{L} \Delta_\ell \eta_\ell \right)$, then we write

$$M_\ell^{\mathcal{D}} = \frac{E_L \Delta_\ell}{\eta_\ell}.$$

Rectangle scheme: Using conditions (ii) and (iii) in Theorem 2.2.2 for the rectangle scheme we get

$$M_\ell^{\mathcal{R}} \leq \left(\frac{2\sqrt{c_2}\sqrt{c_3}}{\varepsilon^2} \sum_{\ell=0}^{L} 2^{-\ell} 2^{\ell} \right) \frac{\sqrt{c_2}}{\sqrt{c_3}} 2^{-\ell} 2^{-\ell} = \left(\frac{2c_2}{\varepsilon^2} \sum_{\ell=0}^{L} 1 \right) 2^{-2\ell} = \frac{2c_2}{\varepsilon^2}(L+1)2^{-2\ell}.$$

We achieve $M_\ell^{\mathcal{R}} = M_0^{\mathcal{R}} 2^{-2\ell}$, where $M_0^{\mathcal{R}} = \left\lceil \frac{2c_2}{\varepsilon^2}(L+1) \right\rceil$.
Finally, to find the optimal number of levels $L_{\mathcal{R}}$, we set the square bias to be less than or equal to $\frac{\varepsilon^2}{2}$ (which corresponds to condition (1) of theorem (2.2.2)), with $\alpha = 1$ and we get

$$c_1^2 2^{-2L_{\mathcal{R}}} \leq \frac{\varepsilon^2}{2} \iff L_{\mathcal{R}} \geq \frac{\ln\left(\sqrt{2}c_1\varepsilon^{-1}\right)}{\ln(2)}.$$

Thus, we set $L_{\mathcal{R}} = \left\lceil \frac{\ln\left(\sqrt{2}c_1\varepsilon^{-1}\right)}{\ln(2)} \right\rceil$.

Trapezoidal scheme: Using conditions (2) and (3) from theorem (2.2.2) for the trapezoidal scheme we get that

$$\begin{aligned} M_\ell^{\mathcal{R}} \leq \left(\frac{2\sqrt{c_2}\sqrt{c_3}}{\varepsilon^2} \sum_{\ell=0}^{L} 2^{-(1+H)\ell} 2^{\ell} \right) \frac{\sqrt{c_2}}{\sqrt{c_3}} 2^{-(1+H)\ell} 2^{-\ell} &= \left(\frac{2c_2}{\varepsilon^2} \sum_{\ell=0}^{L} 2^{-H\ell} \right) 2^{-(2+H)\ell} \\ &= \frac{2c_2}{\varepsilon^2} \left(\frac{1-2^{-H(L+1)}}{1-2^{-H}} \right) 2^{-(2+H)\ell}. \end{aligned}$$

and we achieve $M_\ell^{\mathcal{R}} = M_0^{\mathcal{R}} 2^{-(2+H)\ell}$ where $M_0^{\mathcal{T}} = \left\lceil \frac{2c_2}{\varepsilon^2} \left(\frac{1-2^{-H(L+1)}}{1-2^{-H}} \right) \right\rceil$.
Finally, to find the optimal number of levels $L_{\mathcal{T}}$, we set the square bias to be less than or equal to $\frac{\varepsilon^2}{2}$, with $\alpha = 1 + H$ and we get

$$c_1^2 2^{-2(1+H)L_{\mathcal{T}}} \leq \frac{\varepsilon^2}{2} \iff L_{\mathcal{T}} \geq \frac{\ln\left(\left(\sqrt{2}c_1\varepsilon^{-1}\right)^{\frac{1}{1+H}} \right)}{\ln(2)}.$$

Thus, we set $L_{\mathcal{T}} = \left\lceil \frac{\ln\left(\left(\sqrt{2}c_1\varepsilon^{-1}\right)^{\frac{1}{1+H}} \right)}{\ln(2)} \right\rceil$.

2.3 Weighted Multilevel Monte Carlo

In this section, we extend the MLMC method by incorporating a weighting scheme that optimises computational efficiency, particularly when correlations between different levels are not sufficiently high. The WMLMC method offers a refined approach that adjusts for these correlations, reducing variance without altering the asymptotic complexity of the original MLMC method.

2.3.1 Recursive formulation of the MLMC

The *control variate method* is a technique used in Monte Carlo simulations to increase the accuracy of an estimate by reducing its variance. It involves using a secondary variable, called the control variate, that is correlated with the main variable of interest and whose expected value is already known.

Definition 2.3.1 (Control variate)**.** Suppose $(X_i, Y_i)_{i=1}^n$ are independent and identically distributed samples of (X, Y) and $\mathbb{E}[Y]$ is known. The control variate with parameter c is given by

$$\overline{X}_n(c) = \overline{X}_n + c\left(\overline{Y}_n - \mathbb{E}[Y]\right) = \frac{1}{n}\sum_{i=1}^{n} X_i + c\left(\frac{1}{n}\sum_{i=1}^{n} Y_i - \mathbb{E}[Y]\right) \tag{2.3.1}$$

We can find the optimal c by minimising the variance of the control variate with respect to c, and by using the optimal c we can show that the variance of the control variate is less than the variance of the standard monte carlo estimator. The variance of the control variate is given by

$$\mathbb{V}\mathrm{ar}\left[\overline{X}_n(c)\right] = \frac{1}{n}\mathbb{V}\mathrm{ar}\left[X + c\left(Y - \mathbb{E}[Y]\right)\right] = \frac{1}{n}\left(\mathbb{V}\mathrm{ar}\left[X\right] + 2c\mathbb{C}\mathrm{ov}\left[X, Y\right] + c^2\mathbb{V}\mathrm{ar}\left[Y\right]\right) \tag{2.3.2}$$

Minimising the variance with respect to c yields the optimal choice

$$\widetilde{c} = -\frac{\mathbb{C}\mathrm{ov}\left[X, Y\right]}{\mathbb{V}\mathrm{ar}\left[Y\right]}.$$

Substituting $\widetilde{c}$ back into (2.3.2) we get

$$\mathbb{V}\mathrm{ar}\left[\overline{X}_n(c)\right] = \frac{1}{n}\left(\mathbb{V}\mathrm{ar}\left[X\right] - \frac{\mathbb{C}\mathrm{ov}\left[X, Y\right]^2}{\mathbb{V}\mathrm{ar}\left[Y\right]}\right) \leq \mathbb{V}\mathrm{ar}\left[\overline{X}_n\right]$$

As shown in [8, Section 1.1] the optimal parameter $\widetilde{c}$ can also be expressed as

$$\widetilde{c} = -\rho\sqrt{\frac{\mathbb{V}\mathrm{ar}(X)}{\mathbb{V}\mathrm{ar}(Y)}}$$

where ρ is the correlation between X and Y, and the variance of the estimator is reduced by $1 - \rho^2$.

To fully understand the idea behind the addition of weights to MLMC estimator we first need to express the MLMC recursively. By doing this we can show that, at its core, the MLMC can be viewed as a sequence of control variates, each one designed to reduce the variance of the estimator at the next finer level. The idea is that, by carefully selecting these control variates, we can minimise the overall variance of the estimator, thereby achieving more accurate results with fewer computational resources. From equation (2.2.6) we have that

$$\mathbb{E}\left[P_L^{\mathcal{D}}\right] = \mathbb{E}\left[P_0^{\mathcal{D}}\right] + \sum_{\ell=1}^{L}\mathbb{E}\left[P_\ell^{\mathcal{D}} - P_{\ell-1}^{\mathcal{D}}\right] = \sum_{\ell=0}^{L}\mathbb{E}\left[Y_\ell\right].$$

Expanding the last term of the summation across the discretisation levels we get

$$\mathbb{E}\left[P_L^{\mathcal{D}}\right] = \mathbb{E}\left[Y_L\right] + \sum_{\ell=0}^{L-1}\mathbb{E}\left[Y_\ell\right].$$

We observe that the last term is precisely $\mathbb{E}\left[P_{L-1}^{\mathcal{D}}\right]$, so the equation becomes

$$\mathbb{E}\left[P_L^{\mathcal{D}}\right] = \mathbb{E}\left[Y_L\right] + \mathbb{E}\left[P_{L-1}^{\mathcal{D}}\right].$$

Using standard Monte Carlo to approximate $\mathbb{E}[Y_L]$ gives

$$\begin{aligned}\widehat{P}_L^{\mathcal{D}} &= \frac{1}{M_L^{\mathcal{D}}}\sum_{m=1}^{M_L^{\mathcal{D}}}\left(P_L^{\mathcal{D},m} - P_{L-1}^{\mathcal{D},m}\right) + \mathbb{E}\left[P_{L-1}^{\mathcal{D}}\right] \\ &= \frac{1}{M_L^{\mathcal{D}}}\sum_{m=1}^{M_L^{\mathcal{D}}} P_L^{\mathcal{D},m} - \left(\frac{1}{M_L^{\mathcal{D}}}\sum_{m=1}^{M_L^{\mathcal{D}}} P_{L-1}^{\mathcal{D},m} - \mathbb{E}\left[P_{L-1}^{\mathcal{D}}\right]\right).\end{aligned} \tag{2.3.3}$$

Comparing equations (2.3.3) and (2.3.1), we notice that the MLMC estimator can be seen as a nested series of control variates, also shown in [13, Section 2.1], that are used to lower the variance of the higher level estimator. The WMLMC method extends this idea by introducing weights that further optimise the variance reduction process.

2.3.2 WMLMC Simulation

From the recursive formulation of the MLMC we have that

$$\widehat{P}_L^{\mathcal{D}} = \frac{1}{M_L^{\mathcal{D}}} \sum_{m=1}^{M_L^{\mathcal{D}}} P_L^{\mathcal{D},m} - \left(\frac{1}{M_L^{\mathcal{D}}} \sum_{m=1}^{M_L^{\mathcal{D}}} P_{L-1}^{\mathcal{D},m} - \mathbb{E}\left[P_{L-1}^{\mathcal{D}}\right] \right)$$

Given a set of weights $\theta = (\theta_0, \ldots, \theta_L)$, with $\theta_0 = 0$, we define the WMLMC estimator for the expectation $\mathbb{E}\left[\varphi(\mathrm{VIX}_T^2)\right]$ recursively. The weights θ are applied similarly to the way parameter c is applied in equation (2.3.1). Thus, we have

$$\mathbb{E}\left[P_L^{\theta,\mathcal{D}}\right] = \frac{1}{M_L^{\mathcal{D}}} \sum_{m=1}^{M_L^{\mathcal{D}}} P_L^{\mathcal{D},m} - \theta_L \left(\frac{1}{M_L^{\mathcal{D}}} \sum_{m=1}^{M_L^{\mathcal{D}}} P_{L-1}^{\mathcal{D},m} - \mathbb{E}\left[P_{L-1}^{\mathcal{D}}\right] \right).$$

Given this recursive form of the WMLMC, we simulate it as with MLMC in (2.2.6), such that

$$\begin{aligned}
\widehat{P}_L^{\theta,\mathcal{D}} &= \frac{1}{M_L^{\mathcal{D}}} \sum_{m=1}^{M_L^{\mathcal{D}}} P_L^{\mathcal{D},m} - \theta_L \frac{1}{M_L^{\mathcal{D}}} \sum_{m=1}^{M_L^{\mathcal{D}}} P_{L-1}^{\mathcal{D},m} + \theta_L \mathbb{E}\left[P_{L-1}^{\mathcal{D}}\right] \\
&= \frac{1}{M_L^{\mathcal{D}}} \sum_{m=1}^{M_L^{\mathcal{D}}} P_L^{\mathcal{D},m} - \theta_L P_{L-1}^{\mathcal{D},m} + \theta_L \left(\frac{1}{M_{L-1}^{\mathcal{D}}} \sum_{m=1}^{M_{L-1}^{\mathcal{D}}} \left(P_{L-1}^{\mathcal{D},m} - \theta_{L-1} P_{L-2}^{\mathcal{D},m} \right) + \theta_{L-1} \mathbb{E}\left[P_{L-2}^{\mathcal{D}}\right] \right).
\end{aligned}$$

To simplify the calculations, we now define the equivalent of Equation (2.2.2) for the weighted scheme, such that, for $\ell = 0, \ldots, L$, we have

$$Y_\ell^\theta := P_\ell^{\mathcal{D}} - \theta_\ell P_\ell^{\mathcal{D}} \quad \text{with} \quad Y_0^\theta = P_0^{\mathcal{D}}, \tag{2.3.4}$$

$$\eta_\ell^2 := \mathrm{Cost}(Y_\ell^\theta) = \mathrm{Cost}(Y_\ell) = n^2, \tag{2.3.5}$$

$$(\Delta_\ell^\theta)^2 := \mathbb{V}\mathrm{ar}(Y_\ell^\theta). \tag{2.3.6}$$

Substituting this into $\widehat{P}_\ell^{\theta,\mathcal{D}}$ above, yields

$$\widehat{P}_L^{\theta,\mathcal{D}} = \frac{1}{M_L^{\mathcal{D}}} \sum_{m=1}^{M_L^{\mathcal{D}}} Y_L^\theta + \theta_L \frac{1}{M_{L-1}^{\mathcal{D}}} \sum_{m=1}^{M_{L-1}^{\mathcal{D}}} Y_{L-1}^\theta + \theta_L \theta_{L-1} \mathbb{E}\left[P_{L-2}^{\mathcal{D}}\right].$$

If we keep expanding until for all $\ell = 0, \ldots, L$ we arrive to the formula

$$\widehat{P}_L^{\theta,\mathcal{D}} = \sum_{\ell=0}^{L} \left(\prod_{k=\ell+1}^{\ell} \theta_k \right) \frac{1}{M_\ell^{\mathcal{D}}} \sum_{m=1}^{M_\ell^{\mathcal{D}}} Y_\ell^\theta, \quad \text{for} \quad \ell = 0, \ldots, L.$$

Finally, the resulting WMLMC estimator for the expectation $\mathbb{E}\left[\varphi(\mathrm{VIX}_T^2)\right]$ is defined as

$$\widehat{P}_L^{\theta,\mathcal{D}} := \sum_{\ell=0}^{L} \Theta_\ell^L \frac{1}{M_\ell^{\theta,\mathcal{D}}} \sum_{m=1}^{M_\ell^{\theta,\mathcal{D}}} \left(P_\ell^{\mathcal{D},m} - \theta_\ell P_{\ell-1}^{\mathcal{D},m} \right), \tag{2.3.7}$$

where the added weights are given by

$$\Theta_\ell^\ell = 1, \quad \text{and} \quad \Theta_\ell^L = \prod_{k=\ell+1}^{L} \theta_k, \quad 0 \le \ell \le L.$$

Similarly to the MLMC the total cost of computing $\widehat{P}_L^{\theta,\mathcal{D}}$ denoted by $(E_L^\theta)^2$ is given by the following expression:

$$(E_L^\theta)^2 = \sum_{\ell=0}^{L} M_\ell^{\theta,\mathcal{D}} \eta_\ell^2, \tag{2.3.8}$$

It is shown in [13, Lemma 2.5] that the computational (square root) effort needed to compute $\widehat{P}_L^{\theta,\mathcal{D}}$ with variance $\frac{\varepsilon^2}{2}$ is

$$\widetilde{E}_L \le \frac{\sigma_L \sqrt{2}}{\varepsilon} \sum_{l=1}^{L} \eta_l \sqrt{1-\rho_l^2}, \tag{2.3.9}$$

and in [13, Proposition 2.2, Corollary 2.3] it is shown that to achieve this we need to set

$$\widetilde{\theta}_\ell = \frac{\rho_\ell \sigma_\ell}{\sigma_{\ell-1}} - \operatorname{sgn}(\rho_\ell) \frac{\varepsilon \widetilde{\Delta}_\ell \widetilde{E}_{\ell-1}}{\sqrt{2}\sigma_{\ell-1}^2 \eta_\ell}. \tag{2.3.10}$$

Above we denoted the optimal weights by $\widetilde{\theta}_\ell$ and the corresponding optimal values of E_ℓ^θ and Δ_ℓ^θ by $\widetilde{E}_\ell$ and $\widetilde{\Delta}_\ell$, respectively. Finally, we denoted $\rho_\ell := \mathbb{C}\mathrm{or}\left(P_\ell^{\mathcal{D}}, P_{\ell-1}^{\mathcal{D}}\right)$, and $\sigma_\ell^2 := \mathbb{V}\mathrm{ar}\left(P_\ell^{\mathcal{D}}\right)$.

Now, we examine the MSE of the WMLMC, used to measure the accuracy of an estimator. It is given by

$$\mathrm{MSE} = \mathbb{E}\left[\left(\widehat{P}_L^{\theta,\mathcal{D}} - E[P]\right)^2\right] = \underbrace{\left(E[P_L^{\mathcal{D}}] - E[P]\right)^2}_{(\mathrm{Bias})^2} + \underbrace{\mathrm{Var}\left(\widehat{P}_L^{\theta,\mathcal{D}}\right)}_{\mathrm{Variance}} \tag{2.3.11}$$

where the variance of the estimator is given by

$$\mathrm{Var}\left(\widehat{P}_L^{\theta,\mathcal{D}}\right) = \mathrm{Var}\left(\sum_{\ell=0}^{L} \widetilde{\Theta}_\ell^L \frac{1}{M_\ell^{\theta,\mathcal{D}}} \sum_{m=1}^{M_\ell^{\theta,\mathcal{D}}} Y_\ell^\theta\right) = \sum_{\ell=0}^{L} \mathrm{Var}\left(\widetilde{\Theta}_\ell^L \frac{1}{M_\ell^{\theta,\mathcal{D}}} \sum_{m=1}^{M_\ell^{\theta,\mathcal{D}}} Y_\ell^\theta\right),$$

and by independence we get

$$\mathrm{Var}\left(\widehat{P}_L^{\theta,\mathcal{D}}\right) = \sum_{\ell=0}^{L} \frac{\left(\widetilde{\Theta}_\ell^L\right)^2 \widetilde{\Delta}_\ell^2}{M_\ell^{\theta,\mathcal{D}}}.$$

For the estimator $\hat{P}_L^{\mathcal{D}}$ to be optimal, we want the MSE to smaller than ε^2, where $\varepsilon > 0$ is a given threshold. For an optimal bias-variance trade-off we impose

(i) $\mathrm{Var}\left(\widehat{P}_L^{\theta,\mathcal{D}}\right) \le \frac{\varepsilon^2}{2}$, which will give us the optimal samples $M_\ell^{\theta,\mathcal{D}}$, $\ell = 0, \ldots, L_{\mathcal{D}}$.

(ii) $\left(E[P_L^{\mathcal{D}}] - E[P]\right)^2 \le \frac{\varepsilon^2}{2}$, which will give us the optimal levels $L_{\mathcal{D}}$.

Chapter 3

Theoretical Results

This chapter focuses on the theoretical aspects of the Weighted Multilevel Monte Carlo method, building on the concepts and methods introduced in earlier sections. The primary goal of this chapter is to establish the computational complexity of the WMLMC method and compare it with the traditional MLMC approach.

3.1 WMLMC Computational Complexity

To establish the computational complexity of WMLMC for pricing VIX options, we begin by presenting the WMLMC complexity theorem given in [13, Theorem 2.6]. This theorem is the equivalent of Theorem 2.2.2 but for the WMLMC. It is foundational, as it sets the stage for proving the more specific results that follow. It provides the general conditions under which the WMLMC method achieves an MSE within a specified tolerance ε, while maintaining an optimal computational complexity.

Theorem 3.1.1 (WMLMC complexity theorem)**.** *Suppose that there are positive constants* $\widetilde{c}_1, \widetilde{c}_2, \widetilde{c}_3, \alpha, \beta, \gamma$ *with* $2\alpha \geq \min(\beta, \gamma)$ *such that, for* $\ell = 0, 1, 2, \ldots$,

1. $|\mathbb{E}[P_\ell - P]| \leq \widetilde{c}_1 2^{-\alpha\ell}$,
2. $\sqrt{1-\rho_\ell^2} \leq \widetilde{c}_2 2^{-\beta\ell}$,
3. $\eta_\ell \leq \widetilde{c}_3 2^{\gamma\ell}$.

Then there exists a positive constant $\widetilde{c}_4$ *such that, for any* $\varepsilon < 1/e$ *there exists an integer* L *such that* $\widehat{P}_L^{\theta,\mathcal{D}}$ *(as defined by* (2.3.7)*) has mean squared error with bound*

$$\mathbb{E}[(\widehat{P}_L^{\theta,\mathcal{D}} - \mathbb{E}[P])^2] < \varepsilon^2,$$

with a computational complexity C *with bound*

$$C \leq \begin{cases} \widetilde{c}_4 \varepsilon^{-2}, & \beta > \gamma, \\ \widetilde{c}_4 \varepsilon^{-2} (\log \varepsilon)^2, & \beta = \gamma, \\ \widetilde{c}_4 \varepsilon^{-2 - \frac{\gamma - \beta}{\alpha}}, & \beta < \gamma. \end{cases}$$

The conditions outlined in this theorem, including the relationships between the constants α, β, and γ, are crucial to ensure that the WMLMC method performs efficiently. As in the MLMC, α represents the rate at which the bias decreases as the level of discretisation becomes finer. The parameter β affects the rate of which the correlation of successive levels approaches one, as ℓ increases. From the relationship of Δ_ℓ^θ and ρ_ℓ given

in [13, Proposition 2.2], β subsequently affects the rate of decay of the variance of the difference between successive levels of discretisation as the levels increase. The parameter γ (same as in the MLMC) indicates the rate of which the cost of generating a sample increases with level ℓ.

Theorem 3.1.2. *Assume that the function φ is Lipschitz, and let n_0 be a positive natural number. For any given tolerance $\varepsilon > 0$, there exists an initial sample size $M_0^{\theta,\mathcal{R}}$ and a number of levels $L_{\mathcal{R}}$ for the rectangle scheme ($M_0^{\theta,\mathcal{T}}$ and $L_{\mathcal{T}}$ for the trapezoidal scheme) such that the corresponding multilevel estimator $\widehat{P}_L^{\theta,\mathcal{R}}$ (and $\widehat{P}_L^{\theta,\mathcal{T}}$), defined in equation (2.3.7), achieves an $MSE < \epsilon^2$. with computational complexity $\mathcal{O}(\ln(\varepsilon)\varepsilon^{-2})$ (and $\mathcal{O}(\varepsilon^{-2})$ respectively). For the rectangle scheme, we set*

$$n_\ell = n_0 2^\ell, \quad M_\ell^{\theta,\mathcal{R}} = M_0^{\theta,\mathcal{R}} 2^{-2\ell}, \quad \textit{for all} \quad \ell = 0, \dots, L_{\mathcal{R}}, \tag{3.1.1}$$

and for the trapezoidal scheme, we set

$$n_\ell = n_0 2^\ell, \quad M_\ell^{\theta,\mathcal{T}} = M_0^{\theta,\mathcal{T}} 2^{-(2+H)\ell}, \quad \textit{for all} \quad \ell = 0, \dots, L_{\mathcal{T}}. \tag{3.1.2}$$

Proof of Theorem 3.1.2. **Step 1: Finding parameters α, β, γ along with parameters $\widetilde{c_1}, \widetilde{c_2}, \widetilde{c_3}$ for both discretisation schemes.**

For $\mathcal{D} = \mathcal{R}$: Assuming that the cost of generating a sample of Y_ℓ^θ is the same as the cost of generating a single-level sample P_ℓ, we have that $\text{Cost}(Y_\ell^\theta) = \text{Cost}(Y_\ell) = \mathcal{O}(n_\ell^2)$ which gives

$$\eta_\ell \leq \sqrt{c_3} 2^\ell.$$

Thus, we set $\gamma = 1$ and $\widetilde{c_3} = \sqrt{c_3}$.
Using the variance formula for the difference of two correlated variables

$$\Delta_\ell^2 = \sigma_\ell^2 + \sigma_{\ell-1}^2 - 2\rho_\ell \sigma_\ell \sigma_{\ell-1} = \sigma_\ell^2(1-\rho_\ell^2) + (\sigma_{\ell-1} - \rho_\ell \sigma_\ell)^2 \geq \sigma_\ell^2(1-\rho_\ell^2) \geq \sigma_L^2(1-\rho_\ell^2).$$

Thus, we get

$$\sqrt{(1-\rho_\ell^2)} \leq \frac{\Delta_\ell}{\sigma_L} \leq \frac{\sqrt{c_2}}{\sigma_L} 2^{-\ell}, \tag{3.1.3}$$

and we set $\beta = 1$ and $\widetilde{c_2} = \sqrt{c_2}$.
Since condition 1 of Theorem 2.2.2 involves the bias of the discretisation scheme we set, same as in the proof of [4, Theorem 8], $\alpha = 1$ and $\widetilde{c_1} = c_1$.
We observe that $2\alpha > min(\beta, \gamma)$, and $\beta = \gamma$, which means we are in the second case of the complexity theorem (Theorem 3.1.1). We conclude that the computational complexity of the WMLMC under the rectangle scheme is $\mathcal{O}\left(\ln(\epsilon)^2 \epsilon^{-2}\right)$.

For $\mathcal{D} = \mathcal{T}$: For the trapezoidal scheme we proceed similarly as for the rectangle scheme. Since the cost of generating a single-level P_ℓ is the same for both discretisation schemes we have

$$\eta_\ell \leq \sqrt{c_3} 2^\ell.$$

Thus, we set $\gamma = 1$ and $\widetilde{c_3} = \sqrt{c_3}$.
In addition, we have

$$\sqrt{(1-\rho_\ell^2)} \leq \frac{\Delta_\ell}{\sigma_L} \leq \frac{\sqrt{c_2}}{\sigma_L} 2^{-(1+H)\ell},$$

and we set $\beta = 1 + H$ and $\widetilde{c_2} = \sqrt{c_2}$.
The bias of the trapezoidal scheme gives us $\alpha = 1 + H$ and $\widetilde{c_1} = c_1$.
We observe that $2\alpha > min(\beta, \gamma)$, and $\beta > \gamma$, which means we are in the first case of the complexity theorem (Theorem 3.1.1). We conclude that the computational complexity of

the WMLMC under the trapezoidal scheme is $\mathcal{O}(\epsilon^{-2})$.

Step 2: Finding the optimal number of samples $M_\ell^{\theta,\mathcal{R}}$ and $M_\ell^{\theta,\mathcal{T}}$ along with parameters $M_0^{\theta,\mathcal{R}}, M_0^{\theta,\mathcal{R}}$.

For $\mathcal{D} = \mathcal{R}$: To find the optimal samples we use Lagrange multipliers to solve the constrained optimisation problem:

$$\min_{M_0,\ldots,M_L} (E_L^\theta)^2, \quad \text{such that} \quad \text{Var}\left(\widehat{P}_L^{\theta,\mathcal{D}}\right) \leq \frac{\varepsilon^2}{2}.$$

This means that we minimise the cost for a given variance. Thus, the Lagrangian for the problem can be expressed as

$$\mathcal{L} = \sum_{\ell=0}^{L} M_\ell^{\theta,\mathcal{R}} \eta_\ell^2 + \lambda \left(\sum_{\ell=0}^{L} \frac{(\widetilde{\Theta}_\ell^L)^2 \widetilde{\Delta}_\ell^2}{M_\ell^{\theta,\mathcal{R}}} - \frac{\varepsilon^2}{2} \right),$$

where we have

$$\frac{\partial \mathcal{L}}{\partial M_\ell^{\theta,\mathcal{R}}} = 0 \Rightarrow \eta_\ell^2 - \lambda \frac{\left(\widetilde{\Theta}_\ell^L\right)^2 \widetilde{\Delta}_\ell^2}{\left(M_\ell^{\theta,\mathcal{R}}\right)^2} = 0.$$

Solving for $M_\ell^{\theta,\mathcal{R}}$, we get

$$M_\ell^{\theta,\mathcal{R}} = \sqrt{\lambda} \sqrt{\frac{\left(\widetilde{\Theta}_\ell^L\right)^2 \widetilde{\Delta}_\ell^2}{\eta_\ell^2}} = \sqrt{\lambda} \frac{\left|\widetilde{\Theta}_\ell^L\right| \widetilde{\Delta}_\ell}{\eta_\ell}. \tag{3.1.4}$$

Substituting this to the variance formula gives us the Lagrange multiplier λ. Specifically, we have

$$\sum_{\ell=0}^{L} \frac{(\widetilde{\Theta}_\ell^L)^2 \widetilde{\Delta}_\ell^2}{M_\ell^{\theta,\mathcal{R}}} = \frac{\varepsilon^2}{2} \Rightarrow \sum_{\ell=0}^{L} \frac{(\widetilde{\Theta}_\ell^L)^2 \widetilde{\Delta}_\ell^2}{\sqrt{\lambda}\sqrt{\frac{\left(\widetilde{\Theta}_\ell^L\right)^2 \widetilde{\Delta}_\ell^2}{\eta_\ell^2}}} = \frac{\varepsilon^2}{2}.$$

Rearranging terms, we obtain

$$\sqrt{\lambda} = \frac{2}{\varepsilon^2} \sum_{\ell=0}^{L} \sqrt{(\widetilde{\Theta}_\ell^L)^2 \widetilde{\Delta}_\ell^2 \eta_\ell^2} = \frac{2}{\varepsilon^2} \sum_{\ell=0}^{L} \left|\widetilde{\Theta}_\ell^L\right| \widetilde{\Delta}_\ell \eta_\ell.$$

Substituting the Lagrange multiplier into equation (3.1.4), we get

$$M_\ell^{\theta,\mathcal{R}} = \left(\frac{2}{\varepsilon^2} \sum_{\ell=0}^{L} \left|\widetilde{\Theta}_\ell^L\right| \widetilde{\Delta}_\ell \eta_\ell \right) \frac{\left|\widetilde{\Theta}_\ell^L\right| \widetilde{\Delta}_\ell}{\eta_\ell}.$$

From [13, Lemma 2.5], we have the relationship

$$\frac{2}{\varepsilon^2} \sum_{\ell=0}^{L} \left|\widetilde{\Theta}_\ell^L\right| \widetilde{\Delta}_\ell \eta_\ell \leq \frac{2\sigma_L}{\varepsilon^2} \sum_{\ell=1}^{L} \eta_\ell \sqrt{1-\rho_\ell^2} \quad \text{and} \quad \left|\widetilde{\Theta}_\ell^L\right| \widetilde{\Delta}_\ell \leq \sigma_L \sqrt{1-\rho_\ell^2} \leq \Delta_\ell.$$

Thus, we write

$$\begin{aligned} M_\ell^{\theta,\mathcal{R}} \leq \left(\frac{2\sigma_L}{\varepsilon^2} \sum_{\ell=1}^{L} \eta_\ell \sqrt{1-\rho_\ell^2} \right) \frac{\Delta_\ell}{\eta_\ell} &= \frac{2\sqrt{c_2}\sqrt{c_3}}{\varepsilon^2} \left(\sum_{\ell=1}^{L} 2^\ell 2^{-\ell} \right) \frac{\sqrt{c_2} 2^{-\ell}}{\sqrt{c_3} 2^\ell} \\ &= \frac{2c_2}{\varepsilon^2} \left(\sum_{\ell=1}^{L} 1 \right) 2^{-2\ell} \\ &= \frac{2c_2 L}{\varepsilon^2} 2^{-2\ell}. \end{aligned}$$

Finally, we achieve $M_\ell^{\theta,\mathcal{R}} = M_0^{\theta,\mathcal{R}} 2^{-2\ell}$ where $M_0^{\theta,\mathcal{R}} = \lceil \frac{2c_2 L}{\varepsilon^2} \rceil$.

For $\mathcal{D} = \mathcal{T}$: For the trapezoidal scheme we proceed similarly, and we see

$$\begin{aligned} M_\ell^{\theta,\mathcal{T}} \leq \left(\frac{2\sigma_L}{\varepsilon^2} \sum_{\ell=1}^{L} \eta_\ell \sqrt{1-\rho_\ell^2} \right) \frac{\Delta_\ell}{\eta_\ell} &= \frac{2\sqrt{c_2}\sqrt{c_3}}{\varepsilon^2} \left(\sum_{\ell=1}^{L} 2^\ell 2^{-(1+H)\ell} \right) \frac{\sqrt{c_2} 2^{-(1+H)\ell}}{\sqrt{c_3} 2^\ell} \\ &= \frac{2c_2}{\varepsilon^2} \left(\sum_{\ell=1}^{L} 2^{-H\ell} \right) 2^{-(2+H)\ell} \\ &= \frac{2c_2}{\varepsilon^2} \left(\frac{1-2^{-H(L+1)}}{1-2^{-H}} - 1 \right) 2^{-(2+H)\ell} \end{aligned}$$

Finally, we achieve $M_\ell^{\theta,\mathcal{T}} = M_0^{\theta,\mathcal{T}} 2^{-(2+H)\ell}$ where $M_0^{\theta,\mathcal{T}} = \left\lceil \frac{2c_2}{\varepsilon^2} \left(\frac{1-2^{-H(L+1)}}{1-2^{-H}} - 1 \right) \right\rceil$.

Step 3: Finding the optimal number of levels $L_\mathcal{R}$ and $L_\mathcal{T}$.

For $\mathcal{D} = \mathcal{R}$: To find the optimal number of levels $L_\mathcal{R}$, we set the square bias to be less than or equal to $\frac{\varepsilon^2}{2}$ (which corresponds to condition (i) of Theorem 3.1.1) with $\alpha = 1$ and we get

$$c_1^2 2^{-2L_\mathcal{R}} \leq \frac{\varepsilon^2}{2} \iff L_\mathcal{R} \geq \frac{\ln\left(\sqrt{2} c_1 \varepsilon^{-1}\right)}{\ln(2)}.$$

Thus, we set $L_\mathcal{R} = \left\lceil \frac{\ln\left(\sqrt{2} c_1 \varepsilon^{-1}\right)}{\ln(2)} \right\rceil$.

For $\mathcal{D} = \mathcal{T}$: To find the optimal number of levels $L_\mathcal{T}$, we set the square bias to be less than or equal to $\frac{\varepsilon^2}{2}$ with $\alpha = 1 + H$ and we get

$$c_1^2 2^{-2(1+H)L_\mathcal{T}} \leq \frac{\varepsilon^2}{2} \iff L_\mathcal{T} \geq \frac{\ln\left(\left(\sqrt{2} c_1 \varepsilon^{-1}\right)^{\frac{1}{1+H}}\right)}{\ln(2)}.$$

Thus, we set $L_\mathcal{T} = \left\lceil \frac{\ln\left(\left(\sqrt{2} c_1 \varepsilon^{-1}\right)^{\frac{1}{1+H}}\right)}{\ln(2)} \right\rceil$. □

3.2 Comparison with MLMC

3.2.1 Variance reduction

Recall that under the MLMC the optimal samples are given by $M_\ell^\mathcal{R} = M_0^\mathcal{R} 2^{-2\ell}$ using the rectangle scheme and by $M_\ell^\mathcal{T} = M_0^\mathcal{T} 2^{-(2+H)\ell}$ when using the trapezoidal scheme. Under the WMLMC the optimal samples are given similarly by $M_\ell^{\theta,\mathcal{R}} = M_0^{\theta,\mathcal{R}} 2^{-2\ell}$ when using the rectangle scheme and by $M_\ell^{\theta,\mathcal{T}} = M_0^{\theta,\mathcal{T}} 2^{-(2+H)\ell}$ when using the trapezoidal scheme. We assume that

(i) $L_\mathcal{R} = L_\mathcal{T} = 3$,

(ii) $H = 0.1$, $c_2 = 0.01$, $\varepsilon = 0.001$,

(iii) $M_0^\mathcal{R} = \lceil \frac{2}{\varepsilon^2}(L_\mathcal{R} + 1) \rceil$, $M_0^\mathcal{T} = \left\lceil \frac{2}{\varepsilon^2} \left(\frac{1-2^{-H(L_\mathcal{T}+1)}}{1-2^{-H}} \right) \right\rceil$,

(iv) $M_0^{\theta,\mathcal{R}} = \left\lceil \frac{2L_\mathcal{R}}{\varepsilon^2} \right\rceil$, $M_0^{\theta,\mathcal{T}} = \left\lceil \frac{2}{\varepsilon^2} \left(\frac{1-2^{-H(L_\mathcal{T}+1)}}{1-2^{-H}} - 1 \right) \right\rceil$.

Number of samples needed to achieve $\mathbb{V}\mathrm{ar}(\widehat{P}_L^{\mathcal{R}}) \leq \frac{\varepsilon^2}{2}$ **(resp.** $\mathbb{V}\mathrm{ar}(\widehat{P}_L^{\theta,\mathcal{T}}) \leq \frac{\varepsilon^2}{2}$ **)**				
	Rectangle Scheme		**Trapezoidal Scheme**	
Discretisation Level	**MLMC**	**WLMC**	**MLMC**	**WLMC**
$\ell = 0$	80000	60000	72316	52316
$\ell = 1$	20000	15000	16868.3	12203.1
$\ell = 2$	5000	3750	3934.6	2846.4
$\ell = 3$	1250	937.5	917.7	663.9

Table 3.1: Number of samples required for MLMC and WMLMC to achieve a target variance of $\varepsilon/2$ under rectangle and trapezoidal discretisation schemes for $L_{\mathcal{R}} = L_{\mathcal{T}} = 3$.

In table 3.1 we observe the corresponding number of samples needed for both the MLMC and WMLMC estimators, as defined in (2.2.6) and (2.3.7) respectively, to achieve a variance less than $\frac{\varepsilon^2}{2}$.

At coarser levels, the number of samples are larger and start to decrease as the levels get finer. On the other hand, the number of time steps are smaller at coarser levels and increase as the levels get finer. Thus, at smaller levels the dominant term minimising the MSE is the variance of the estimator. We have shown that at those levels the number of samples needed to achieve the desired variance when using the WMLMC is smaller than the MLMC for both discretisation schemes, achieving a more effective variance reduction.

3.2.2 Overall complexity

Now, we examine the overall computational complexity of both the MLMC and WMLMC methods under both discretisation schemes. The focus is on understanding how the WMLMC, despite not altering the asymptotic complexity of the method, still offers improvements in terms of the total computational cost.

Rectangle scheme: For the MLMC, the total cost is given by Equation (2.2.10), where

$$E_L^2 = \sum_{\ell=0}^{L} M_\ell^{\mathcal{D}}\, \eta_\ell^2.$$

For $\underline{\mathcal{D} = \mathcal{R}}$ it becomes

$$E_L^2 = \sum_{\ell=0}^{L} M_0^{\mathcal{R}} 2^{-2\ell} c_3 2^{2\ell} = c_3 M_0^{\mathcal{R}} \sum_{\ell=0}^{L} 1 = c_3 M_0^{\mathcal{R}} (L+1)$$

Since $M_0^{\mathcal{R}} \leq \frac{2c_2}{\varepsilon^2}\,(L+1)$, in this case the total cost becomes

$$E_L^2 \leq \frac{2c_2c_3}{\varepsilon^2}(L+1)^2$$

From Theorem 2.2.1, we have that $E_L^2 \leq c_4 \varepsilon^{-2} (\ln \varepsilon)^2$, so we set

$$c_4 = \frac{2c_2c_3(L+1)^2}{(\ln \varepsilon)^2}$$

For the WMLMC, the total cost is given by equation (2.3.8), where

$$(E_L^\theta)^2 = \sum_{\ell=0}^{L} M_\ell^{\theta,\mathcal{D}}\, \eta_\ell^2.$$

When $\underline{\mathcal{D} = \mathcal{R}}$ it becomes

$$(E_L^\theta)^2 = \sum_{\ell=0}^{L} M_0^{\theta,\mathcal{R}} 2^{-2\ell} c_3 2^{2\ell} = c_3 M_0^{\theta,\mathcal{R}} \sum_{\ell=0}^{L} 1 = c_3 M_0^{\theta,\mathcal{R}} (L+1).$$

Since $M_0^{\theta,\mathcal{R}} \leq \frac{2c_2 L}{\varepsilon^2}$, the total cost becomes

$$(E_L^\theta)^2 \leq \frac{2c_2 c_3}{\varepsilon^2} L(L+1).$$

From Theorem 3.1.2 we have that $(E_L^\theta)^2 \leq \widetilde{c}_4 \varepsilon^{-2} (\ln \varepsilon)^2$. Thus we set

$$\widetilde{c}_4 = \frac{2c_2 c_3 L(L+1)}{(\ln \varepsilon)^2}.$$

Finally, we have

$$\delta(L) := \frac{\widetilde{c}_4}{c_4} = \frac{L(L+1)}{(L+1)^2} = \frac{L}{L+1} < 1,$$

which implies that $\widetilde{c}_4 < c_4$.

Trapezoidal scheme: For the MLMC, the total cost, given by Equation (2.2.10), becomes

$$E_L^2 \leq \sum_{\ell=0}^{L} M_0^{\mathcal{T}} 2^{-(2+H)\ell} c_3 2^{2\ell} = c_3 M_0^{\mathcal{T}} \sum_{\ell=0}^{L} 2^{-H\ell} = c_3 M_0^{\mathcal{T}} \left(\frac{1 - 2^{-H(L+1)}}{1 - 2^{-H}} \right).$$

Since $M_0^{\mathcal{T}} \leq \frac{2c_2}{\varepsilon^2} \left(\frac{1-2^{-H(L+1)}}{1-2^{-H}} \right)$, the total cost becomes

$$E_L^2 \leq \frac{2c_2 c_3}{\varepsilon^2} \left(\frac{1 - 2^{-H(L+1)}}{1 - 2^{-H}} \right)^2.$$

From Theorem 2.2.1 we have $E_L^2 \leq c_4 \varepsilon^{-2}$. Thus we set

$$c_4 = 2c_2 c_3 \left(\frac{1 - 2^{-H(L+1)}}{1 - 2^{-H}} \right)^2.$$

For the WMLMC, the total cost, given by Equation (2.3.8), becomes

$$(E_L^\theta)^2 = \sum_{\ell=0}^{L} M_0^{\theta,\mathcal{T}} 2^{-(2+H)\ell} c_3 2^{2\ell} = c_3 M_0^{\theta,\mathcal{T}} \sum_{\ell=0}^{L} 2^{-H\ell} = c_3 M_0^{\theta,\mathcal{T}} \left(\frac{1 - 2^{-H(L+1)}}{1 - 2^{-H}} \right).$$

Since $M_0^{\theta,\mathcal{T}} \leq \frac{2c_2}{\varepsilon^2} \left(\frac{1-2^{-H(L+1)}}{1-2^{-H}} - 1 \right)$, the total cost becomes

$$(E_L^\theta)^2 \leq \frac{2c_2 c_3}{\varepsilon^2} \left(\frac{1 - 2^{-H(L+1)}}{1 - 2^{-H}} - 1 \right) \left(\frac{1 - 2^{-H(L+1)}}{1 - 2^{-H}} \right)$$

From Theorem 2.2.1 we have that $(E_L^\theta)^2 \leq \widetilde{c}_4 \varepsilon^{-2}$. Thus we set

$$\widetilde{c}_4 = 2c_2 c_3 \left(\frac{1 - 2^{-H(L+1)}}{1 - 2^{-H}} - 1 \right) \left(\frac{1 - 2^{-H(L+1)}}{1 - 2^{-H}} \right).$$

Finally, we have

$$\delta(L) := \frac{\widetilde{c}_4}{c_4} = \frac{2c_2c_3\left(\frac{1-2^{-H(L+1)}}{1-2^{-H}} - 1\right)\left(\frac{1-2^{-H(L+1)}}{1-2^{-H}}\right)}{2c_2c_3\left(\frac{1-2^{-H(L+1)}}{1-2^{-H}}\right)^2} = \frac{\left(\frac{1-2^{-H(L+1)}}{1-2^{-H}} - 1\right)}{\left(\frac{1-2^{-H(L+1)}}{1-2^{-H}}\right)}.$$

Setting $L' := \frac{1-2^{-H(L+1)}}{1-2^{-H}} - 1$, we see that, similarly to the rectangle scheme,

$$\frac{\widetilde{c}_4}{c_4} = \frac{L'}{L'+1} < 1,$$

which implies that $\widetilde{c}_4 < c_4$.

Figure 3.1 shows the ratio $\delta(L)$, which is an indicator of how much the WMLMC method improves on the MLMC method for a given discretisation scheme. Specifically, $\delta(L)$ reflects the efficiency gain achieved by WMLMC over MLMC at each level L. The

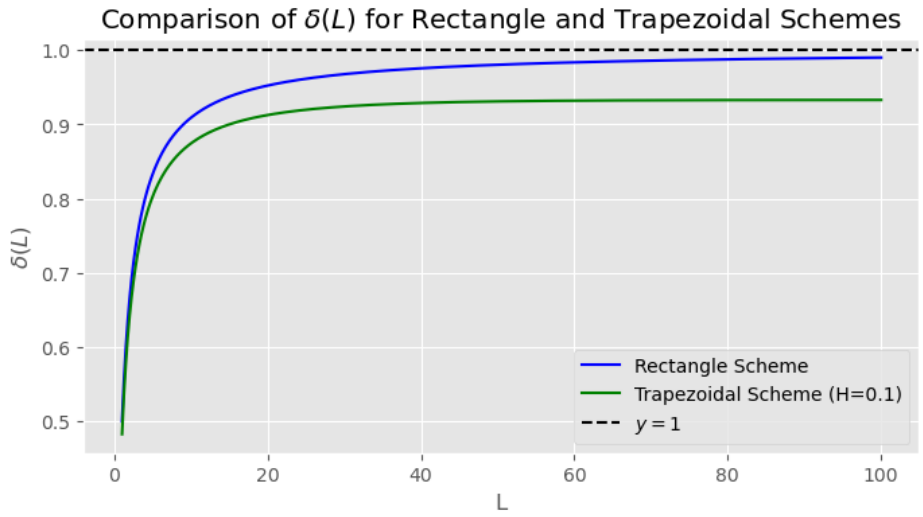

Figure 3.1: Comparison of the ration $\delta(L)$ for the rectangle and trapezoidal discretisation schemes. The graph illustrates how the ratio $\delta(L)$ approaches 1 as the finer level of the estimators increase.

$\delta(L)$ ratio for the rectangle scheme starts low and rapidly increases as the number of levels L increases, indicating a significant improvement in efficiency when using WMLMC compared to MLMC, particularly at lower levels. As L increases further, the ratio approaches but does not reach 1, implying that WMLMC consistently outperforms MLMC but with diminishing returns at higher levels. The $\delta(L)$ ratio for the trapezoidal scheme with $H = 0.1$ starts lower than the rectangle scheme but follows a similar pattern of rapid improvement at lower levels L. Over time, the Trapezoidal Scheme ratio also approaches 1, though it stays slightly below the Rectangle Scheme, indicating that while WMLMC improves upon MLMC, the improvement is more pronounced compared to the rectangle scheme. The dashed line at $y = 1$ represents the point where WMLMC and MLMC have equal efficiency. The fact that both curves approach this line but do not cross it suggests that WMLMC generally provides better performance across all levels.

Chapter 4

Numerical Results

In this chapter, we present the numerical results that serve to validate the theoretical findings discussed in the previous chapters. Specifically, we aim to demonstrate the computational efficiency and accuracy of the WMLMC method in the context of pricing VIX options.

To effectively illustrate the theoretical results, we employ log-log plots, which provide a clear and visual representation of the relationship between the MSE and computational cost of the estimators. These plots are particularly useful because they allow us to confirm whether the expected power-law relationship between MSE and the cost holds true in practice.

A log-log plot is a graph where both the x-axis (representing the computational cost) and the y-axis (representing the MSE) are on logarithmic scales. This type of plot is advantageous for analysing data that follows a power-law relationship, as it transforms the power-law function into a linear function. In our context, if the relationship between the computational cost of the estimators and the MSE follows a power law of the form

$$\text{MSE} \propto \text{Cost}^{-\alpha} \iff \text{MSE} = K \cdot \text{Cost}^{-\alpha}$$

for a constant K, then by taking the logarithm on both sides, we obtain

$$\ln(\text{MSE}) = -\alpha \ln(\text{Cost}) + \ln(K).$$

This equation is linear with a slope of $-\alpha$, which should manifest as a straight line on a log-log plot. The value of the slope provides a direct verification of the theoretical predictions. If the slope matches the expected value, it confirms that the WMLMC method performs as predicted, both in terms of reducing the MSE and in terms of computational efficiency.

From theorem 3.1.2 the computational complexity of the WMLMC under the rectangle scheme is

$$\text{Cost}^{\mathcal{R}}_{\text{WMLMC}} = \mathcal{O}\left(\varepsilon^{-2} \ln^2(\varepsilon)\right)$$

To achieve an MSE less than ε^2, we consider the log-log relationship

$$\ln(\varepsilon^2) = -\alpha \ln\left(\varepsilon^{-2} \ln^2(\varepsilon)\right) + \ln(K).$$

Simplifying, we get

$$-2\ln(\varepsilon) = -\alpha(-2\ln(\varepsilon) + 2\ln(\ln(\varepsilon))) + \ln(K).$$

Matching terms, the dominant part $-2\ln(\varepsilon)$ gives us $\alpha = 1$, with a minor correction due to the logarithmic term $2\ln(\ln(\varepsilon))$. This correction does not significantly alter the slope,

which means it is still expected to be close to -1. Similarly for the WMLMC under the trapezoidal scheme, from theorem 3.1.2, the computational complexity of the estimator is

$$\text{Cost}^{\mathcal{T}}_{\text{WMLMC}} = \mathcal{O}(\varepsilon^{-2}).$$

In this case, to achieve an MSE less than ε^2, we consider the log-log relationship

$$\ln(\varepsilon^2) = -\alpha \ln\left(\varepsilon^{-2}\right) + \ln(K).$$

Therefore, matching terms gives us $\alpha = 1$. Thus, the slope of the log-log plot for the trapezoidal scheme is also expected to be -1.

For all four estimators, we consider an at-the-money VIX call option, using parameters $T = 0.5, H = 0.1, \eta = 0.5, X_0 = \ln\left(0.235^2\right), \Delta = \frac{1}{12}$, and $n_0 = 6$. For different values of ε, the total coat of the MLMC estimator under the rectangle scheme is calculated using equation (2.2.10), such that

$$\text{Cost}^{\mathcal{R}}_{\text{MLMC}} = \sum_{\ell=0}^{L_{\mathcal{R}}} M_\ell^{\mathcal{D}}\, \eta_\ell^2 = c_3 M_0^{\mathcal{R}}(L_{\mathcal{R}} + 1)$$

where $c_3 = n_0^2$, $L_{\mathcal{R}} = \left\lceil \frac{\ln\left(\sqrt{2}c_1\varepsilon^{-1}\right)}{\ln(2)} \right\rceil$ and $M_0^{\mathcal{R}} = \left\lceil \frac{2c_2}{\varepsilon^2}(L_{\mathcal{R}} + 1) \right\rceil$. Under the trapezoidal scheme the total cost is calculated using

$$\text{Cost}^{\mathcal{T}}_{\text{MLMC}} = \sum_{\ell=0}^{L_{\mathcal{T}}} M_\ell^{\mathcal{T}}\, \eta_\ell^2 = c_3 M_0^{\mathcal{T}} \left(\frac{1 - 2^{-H(L_{\mathcal{T}}+1)}}{1 - 2^{-H}} \right)$$

where $L_{\mathcal{T}} = \left\lceil \frac{\ln\left(\left(\sqrt{2}c_1\varepsilon^{-1}\right)^{\frac{1}{1+H}}\right)}{\ln(2)} \right\rceil$ and $M_0^{\mathcal{T}} = \left\lceil \frac{2c_2}{\varepsilon^2} \left(\frac{1-2^{-H(L+1)}}{1-2^{-H}} \right) \right\rceil$. The total cost of the WMLMC estimator under the rectangle scheme is calculated using equation (2.3.8) such that

$$\text{Cost}^{\mathcal{R}}_{\text{WMLMC}} = \sum_{\ell=0}^{L} M_\ell^{\theta,\mathcal{R}}\, \eta_\ell^2 = c_3 M_0^{\theta,\mathcal{R}}(L + 1)$$

where $L_{\mathcal{R}} = \left\lceil \frac{\ln\left(\sqrt{2}c_1\varepsilon^{-1}\right)}{\ln(2)} \right\rceil$ and $M_0^{\theta,\mathcal{R}} = \left\lceil \frac{2c_2 L}{\varepsilon^2} \right\rceil$. Under the trapezoidal scheme the total cost is calculated using

$$\text{Cost}^{\mathcal{T}}_{\text{WMLMC}} = \sum_{\ell=0}^{L} M_\ell^{\theta,\mathcal{T}}\, \eta_\ell^2 = c_3 M_0^{\theta,\mathcal{T}} \left(\frac{1 - 2^{-H(L+1)}}{1 - 2^{-H}} \right)$$

where $L_{\mathcal{T}} = \left\lceil \frac{\ln\left(\left(\sqrt{2}c_1\varepsilon^{-1}\right)^{\frac{1}{1+H}}\right)}{\ln(2)} \right\rceil$ and $M_0^{\theta,\mathcal{T}} = \left\lceil \frac{2c_2}{\varepsilon^2} \left(\frac{1-2^{-H(L+1)}}{1-2^{-H}} - 1 \right) \right\rceil$. The parameters c_1 and c_2 are calculated in [4, Proof of Theorem 8] and are given by $c_1 = \mathcal{L}_\varphi \Lambda(X_0, T, \Delta, H) n_0^{-1}$, and $c_2 = 10\mathcal{L}_\varphi^2 \Lambda(X_0, T, \Delta, H)^2 n_0^{-2}$. $\mathcal{L}_\varphi$ is the Lipschitz constant of the function $\varphi_{\text{call}}(x) = (\sqrt{x} - K)^+$ and is given by $\mathcal{L}_\varphi = \frac{1}{2K}$ [4, Remark 6]. $\Lambda(X_0, T, \Delta, H)$ was calculated in [4, Theorem 5] and is given by

$$\Lambda(X_0, T, \Delta, H) = \frac{e^{X_0}}{2} \left(e^{\eta^2 \frac{T^{2H}}{2H}} + e^{\eta^2 \frac{(T+\Delta)^{2H} - \Delta^{2H}}{2H}} - 2e^{\eta^2 \int_0^T t^{H-\frac{1}{2}} (\Delta+t)^{H-\frac{1}{2}}\, dt} \right)^{1/2}$$

and the integral is calculated using the `scipy.integrate.quad` function.

To estimate the MSE for each method, we consider the same values of ε and it is calculated as the average of the squared differences between the estimated prices and a reference price. Specifically, the MSE is computed using

$$\text{MSE} = \frac{1}{N_{\text{MSE}}} \sum_{j=1}^{N_{\text{MSE}}} (\hat{p}_j - p)^2$$

Here, p represents the reference price, which is typically obtained from a highly accurate or analytical solution, serving as a benchmark for evaluating the accuracy of the estimated prices. The quantities $\hat{p}_j$ are the estimated prices generated by the simulation method under consideration, either the MLMC method or the WMLMC method. We choose $N_{MSE} = 10^3$ and $p = 0.0298840$ which was obtained in [4, Section 3.3] using an intensive standard Monte Carlo simulation with a control variate using 500 discretisation points and 2×10^7 samples M.

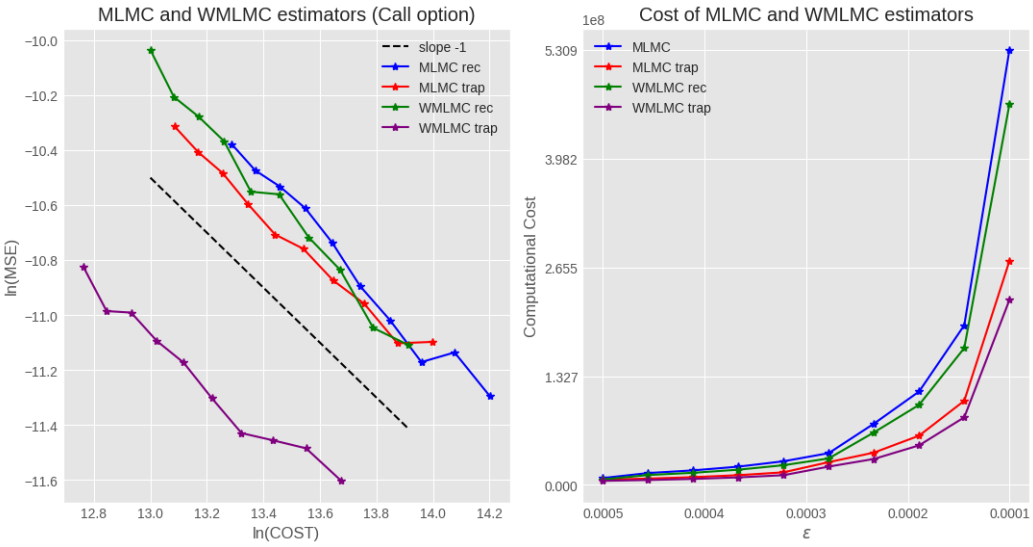

Figure 4.1: Comparison of MSE and computational cost between MLMC and WMLMC estimators for a VIX call option. The *left* graph shows the log-log plot of MSE with the Cost, using $T = 0.5, H = 0.1, \eta = 0.5, X_0 = \ln\left(0.235^2\right), \Delta = \frac{1}{12}$, and $n_0 = 6$. The *right* graph illustrates the computational cost as a function of the desired accuracy ε.

The results of the log-log plot are presented in the left graph of Figure 4.1. The straight-line relationship observed on the log-log plot is a strong validation of the theoretical predictions. The slope matching the expected value of -1 indicates that the asymptotic complexities discussed in Theorem 3.1.2 are indeed reflected in a practical setting.

For the rectangle scheme, the WMLMC method shows a slight improvement over the standard MLMC method. This improvement would manifest as the WMLMC line being positioned lower on the plot compared to the MLMC line, although the MSE of the WMLMC appears to be higher than the MSE of MLMC. This could be attributed to comparing relatively close numbers of ε, which may affect the accuracy of these experiments.

The trapezoidal scheme generally provides better accuracy than the rectangle scheme for the same number of discretisation points. The WMLMC line for the trapezoidal scheme lies below the corresponding MLMC line, indicating that the WMLMC method improves the efficiency of the trapezoidal scheme by further reducing the required computational resources and by achieving a lower MSE. This allows for faster computations or more accurate results within the same computational budget, making the WMLMC method a more efficient option. This is further confirmed by the right graph of Figure 4.1, where it shows

that the the WMLMC under the trapezoidal scheme has the lowest overall computational cost.

Conclusion

In this thesis we explored the application of advanced computational techniques to the challenging problem of pricing VIX options. The VIX is a critical measure of market expectations for future volatility, and traditional pricing models fail to provide an accurate model, necessitating the use of more sophisticated models such as the Rough Bergomi model. We have seen that the rough Bergomi model, while offering a more realistic representation of volatility dynamics through fractional Brownian motion, it introduces significant computational challenges due to the high cost of simulating its rough volatility paths. The primary contribution of this thesis is the investigation and application of the Weighted Multilevel Monte Carlo method as a solution to these challenges.

Through a detailed theoretical analysis and comprehensive numerical experiments, we demonstrated that the WMLMC method significantly reduces the computational cost associated with VIX option pricing under the Rough Bergomi model. The WMLMC method has been shown to maintain the same asymptotic complexity as the MLMC method, with a complexity of $\mathcal{O}(\epsilon^{-2}\log^2(\epsilon))$ for the rectangle scheme and $\mathcal{O}(\epsilon^{-2})$ for the trapezoidal scheme. However, the WMLMC achieves a lower total computational cost due to a reduced constant factor, $\widetilde{c}_4$, which arises from the optimised allocation of computational resources across different levels. The reduced computational cost associated with WMLMC does not compromise the accuracy of the simulations. Instead, it allows for the same level of precision to be achieved with fewer resources, making WMLMC a more efficient and practical choice for financial modelling. This efficiency is particularly pronounced under the trapezoidal scheme, where the WMLMC method significantly outperforms the MLMC method.

In conclusion, this thesis has provided a comprehensive examination of the WMLMC method and its application to a critical problem in quantitative finance. The findings suggest that the WMLMC method is not only theoretically sound but also practically advantageous, offering significant computational savings without sacrificing accuracy. This work contributes to the broader field of financial mathematics by providing a more efficient method for pricing complex financial derivatives, such as VIX options, under models that capture the rough nature of market volatility.

The success of the WMLMC method in this context opens several avenues for future research. Potential directions include: Investigating the application of WMLMC to other complex financial models, particularly those that exhibit rough volatility or other challenging stochastic characteristics. Further refining the weighting scheme within the WMLMC to enhance efficiency, especially in scenarios with extremely high computational demands. Developing real-time pricing tools based on WMLMC, which could be integrated into trading platforms to provide real time pricing for VIX options and other derivatives.

The continued development of efficient computational methods like the WMLMC will be crucial as financial markets evolve and the demand for high-speed, high-accuracy simulations increases.

Appendix A

Python Code

A.1 MLMC Simulation

```
def mlmc_simulation(H, eta, T, delta, L, M0, K, precomputed_values):

    payoff_diffs = np.zeros(L + 1)

    for ell in range(L + 1):
        M = int(M0 * 2**(-2*ell))
        mu, L_matrix, n = precomputed_values[ell]

        VIX_fine_samples = np.zeros(M)
        VIX_coarse_samples = np.zeros(M)

        for m in range(M):
            Z = np.random.normal(size=n + 1)
            VIX_fine_samples[m] = calculate_vix(precomputed_values, ell, Z)

            if ell > 0:
                VIX_coarse_samples[m] = calculate_vix(precomputed_values,
   ell - 1, Z[::2])

        if ell > 0:
            payoff_diff = np.mean(call_option_payoff(VIX_fine_samples, K) -
    call_option_payoff(VIX_coarse_samples, K))
        else:
            payoff_diff = np.mean(call_option_payoff(VIX_fine_samples, K))

        payoff_diffs[ell] = payoff_diff

    mlmc_estimator = np.sum(payoff_diffs)

    return mlmc_estimator
```

A.2 WMLMC Simulation

```
def calculate_weights(L, M0_theta, strike, epsilon, precomputed_values):
    v = epsilon/np.sqrt(2)
    E = []
    weights = []
    vix_samples = []

    for ell in range(L + 1):
        mu, L_matrix, n = precomputed_values[ell]
        M = int(M0_theta * 2**(-2*ell))
```

```
        VIX_fine_samples = np.zeros(M)
        VIX_coarse_samples = np.zeros(M)
        Eta = n

        for m in range(M):
            Z = np.random.normal(size=n + 1)
            VIX_fine_samples[m] = calculate_vix(precomputed_values, ell, Z)

            if ell > 0:
                VIX_coarse_samples[m] = calculate_vix(precomputed_values,
    ell - 1, Z[::2])
            else:
                VIX_coarse_samples[m] = 0

        vix_samples.append((VIX_fine_samples, VIX_coarse_samples))

        if ell == 0:
            weights.append(0)  # No weight needed for the coarsest level
            sigma_l = np.std(call_option_payoff(VIX_fine_samples, strike))
            E.append((sigma_l*Eta)/v)
        else:
            Delta_l = np.std(call_option_payoff(VIX_fine_samples, strike) -
     call_option_payoff(VIX_coarse_samples, strike))
            sigma_l = np.std(call_option_payoff(VIX_fine_samples, strike))
            sigma_l_minus_1 = np.std(call_option_payoff(VIX_coarse_samples,
     strike))
            rho_l = np.corrcoef(call_option_payoff(VIX_fine_samples, strike
    ), call_option_payoff(VIX_coarse_samples, strike))[0, 1]
            theta = (rho_l * sigma_l / sigma_l_minus_1) - (np.sign(rho_l) *
     (Delta_l*v*E[ell-1])/(sigma_l_minus_1**2 * Eta) )
            weights.append(theta)
            E.append((1/v)*(Delta_l*Eta + np.abs(theta)*E[ell-1]*v))

    return weights, vix_samples

def calculate_Theta_l(weights, l, L):
    if l == L:
        return 1
    else:
        return np.prod(weights[l+1:])

def wmlmc_simulation(H, eta, T, delta, L, M0_theta, K, precomputed_values,
    weights, vix_samples):

    payoff_diffs = np.zeros(L + 1)

    for ell in range(L + 1):
        mu, L_matrix, n = precomputed_values[ell]
        M = int(M0_theta * 2**(-2*ell))

        VIX_fine_samples, VIX_coarse_samples = vix_samples[ell]

        Y_l = call_option_payoff(VIX_fine_samples, K) - weights[ell] *
    call_option_payoff(VIX_coarse_samples, K)
        Theta_l = calculate_Theta_l(weights, ell, L)

        payoff_diff = Theta_l * np.mean(Y_l)

        payoff_diffs[ell] = payoff_diff

    wmlmc_estimator = np.sum(payoff_diffs)

    return wmlmc_estimator
```

Bibliography

[1] C. Bayer, P. K. Friz, and J. Gatheral, *Pricing under rough volatility*, Quantitative Finance, 16 (2016), pp. 887–904.

[2] L. Bergomi, *Smile dynamics ii*, Risk, 18 (2005), pp. 67–73.

[3] ——, *Stochastic volatility modeling*, CRC Press, 2016.

[4] F. Bourgey and S. De Marco, *Multilevel monte carlo simulation for vix options in the rough bergomi model*, arXiv preprint arXiv:2105.05356, (2021).

[5] Chicago Board Options Exchange, *The cboe volatility index - vix*. `http://www.cboe.com/micro/vix/vixwhite.pdf`, 2009.

[6] P. J. Davis and P. Rabinowitz, *Methods of numerical integration*, Courier Corporation, 2007.

[7] M. Fukasawa, *Rough volatility and fractional calculus*, in Advanced Modelling in Mathematical Finance, Springer, 2017, pp. 237–260.

[8] M. B. Giles, *Multilevel monte carlo methods*, Acta numerica, 24 (2015), pp. 259–328.

[9] G. H. Golub and C. F. van Loan, *Matrix Computations*, JHU Press, fourth ed., 2013.

[10] B. Horvath, A. Jacquier, and P. Tankov, *Volatility options in rough volatility models*, SIAM Journal on Financial Mathematics, 11 (2020), pp. 437–469.

[11] A. Jacquier, C. Martini, and A. Muguruza, *On vix futures in the rough bergomi model*, Quantitative Finance, 18 (2018), pp. 45–61.

[12] P. E. Kloeden, E. Platen, P. E. Kloeden, and E. Platen, *Stochastic differential equations*, Springer, 1992.

[13] Y. Li and A. Ware, *A weighted multilevel monte carlo method*, arXiv preprint arXiv:2405.03453, (2024).

[14] S. Lim and V. Sithi, *Asymptotic properties of the fractional brownian motion of riemann-liouville type*, Physics Letters A, 206 (1995), pp. 311–317.

[15] B. B. Mandelbrot and J. W. Van Ness, *Fractional brownian motions, fractional noises and applications*, SIAM Review, 10 (1968), pp. 422–437.

[16] Y. Mishura and I. Mishura, *Stochastic Calculus for Fractional Brownian Motion and Related Processes*, no. no. 1929 in Lecture Notes in Mathematics, Springer, 2008.

[17] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, et al., *Scipy 1.0: fundamental algorithms for scientific computing in python*, Nature methods, 17 (2020), pp. 261–272.