TU Delft

Delft University of Technology

Numerical Finance with Backward Stochastic Differential Equations

An Exploration of Three Schemes

Chau, Ki Wai

DOI
10.4233/uuid:2347b8d5-77b9-46f3-a054-33fdf007e906

Publication date
2020

Document Version
Final published version

Citation (APA)
Chau, K. W. (2020). *Numerical Finance with Backward Stochastic Differential Equations: An Exploration of Three Schemes*. [Dissertation (TU Delft), Delft University of Technology].
https://doi.org/10.4233/uuid:2347b8d5-77b9-46f3-a054-33fdf007e906

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

NUMERICAL FINANCE WITH BACKWARD STOCHASTIC DIFFERENTIAL EQUATIONS

AN EXPLORATION OF THREE SCHEMES

NUMERICAL FINANCE WITH BACKWARD STOCHASTIC DIFFERENTIAL EQUATIONS

AN EXPLORATION OF THREE SCHEMES

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus Prof. dr. ir. T.H.J.J. van der Hagen,
Chair of the Board for Doctorates
to be defended publicly on Thursday 16 January 2020 at 10:00 o'clock

by

Ki Wai CHAU

Master of Philosophy in Financial Mathematics,
The University of Hong Kong, Hong Kong,
born in Hong Kong.

This dissertation has been approved by the

promotor: Prof. dr. ir. C. W. Oosterlee

Composition of the doctoral committee:

Rector Magnificus,	chairperson
Prof. dr. ir. C. W. Oosterlee,	Delft University of Technology, promotor
Independent members:	
Prof. dr. S. C. P. Yam,	Chinese University of Hong Kong
Prof. dr. J.-F. Chassagneux,	Paris Diderot University
Prof. dr. C. Vázquez Cendon,	The University of A Coruña
Prof. dr. R. P. Stevenson,	The University of Amsterdam
Prof. dr. J. E. Frank,	Utrecht University
Prof. dr. J. M. A. M. van Neerven,	Delft University of Technology
Prof. dr. ir. A. W. Heemink	Delft University of Technology, reserve member

This research was funded by the European Commission through European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 643045.

Keywords: Backward Stochastic Differential Equations, Fourier expansion methods, Stochastic Grid Bundling Method, Branching Methods

Printed by: Ipskamp Printing

Front & Back: Asia-Pacific Images Studio.

Copyright © 2019 by K.W. Chau

ISBN 978-94-028-1886-4

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

Research means that you don't know, but are willing to find out.

Charles F. Kettering

CONTENTS

SUMMARY

Backward Stochastic Differential Equations (BSDEs) are interesting mathematical objects with lots of promising applications. Within mathematical finance, they can be seen as an extension of the classical replicating portfolio scheme. They are linked to Partial Differential Equations (PDEs) through Feynman–Kac type formulas. They also find applications in optimal control theory with the Hamilton–Jacobi–Bellman (HJB) equation.

In light of the previous global financial crisis, financial industries are required to consider more and more risk factors in their business. Therefore, there is a renewed interest in advanced quantitative tools in mathematical finance, where BSDEs are promising formulations. However, one of the main obstacles in putting BSDEs into industrial applications is the difficulty to solve BSDEs analytically or numerically.

The main aims of this research are to study various numerical schemes in the approximation of the occurring expectations and their applications in numerically solving BSDEs. We focus on numerical expectation/finite measure integration, since the majority of the BSDE solvers consists of two parts, conditional expectations computations and deterministic functions to map these expectations to target approximations. By simply changing the approximation for conditional expectations, we can effectively generate various schemes for BSDEs that can suit different requirements. Furthermore, our results carry implications in numerical integration too.

In this thesis, we focus on the mathematical properties of these approximations. We will discuss the fundamental assumptions for them, give complete descriptions, derive error bounds and conduct numerical experiments. The main goal is to analyse these approximations. We will also touch upon the financial applications of BSDEs, in financial derivatives pricing and value adjustments computation, and software engineering, in Python and GPU computing.

The main subjects of this thesis are Fourier expansion methods and the Stochastic Grid Bundling Method (SGBM). The Fourier expansion method makes use of the fact that we can use trigonometric functions to form a complete basis of bounded domain/periodic functions. This method generates integration approximations based on the sum of Fourier transforms of the integrated measure. We add localization elements to them in this thesis. SGBM is a least-squares Monte Carlo scheme that combines regress-later and stochastic partition techniques. By regress-later, we mean that the integrand and the regression basis are defined on the same variable (adapted to a filtration at a later time step). By stochastic partition, we mean that we separate the domain of random variables which we condition into separated sets (bundles) based on Monte-Carlo simulations. We define approximations on distinct bundles separately. In this thesis, we derive a baseline error bound for SGBM and discuss the limitation of not being able to perform truncation with regress-later schemes. Finally, we briefly study the branching method, a forward simulation method for BSDEs and also its limitations.

In Chapter 2, we study a localized Fourier expansion method, the quick SWIFT method, and provide an alternative derivation based on the periodic wavelets literature. This method combines the effectiveness of Fourier-based methods and the simplicity of a wavelet-based formula, resulting in an algorithm that is both accurate and easy to implement. Furthermore, we mitigate the problem of inaccurate approximations near the computational boundaries by means of an antireflective boundary technique. To extend localized Fourier expansions to higher dimensions, we adapt results from lattice sequences in Chapter 3 to design the cosine expansion lattice scheme. We also compare localization based on lattices and on wavelets.

In Chapter 4, the basic idea of the SGBM will be introduced and an upper error bound is established for the simplest two-step version of SGBM. A full error analysis is also conducted for the explicit version of the numerical BSDE algorithm based on time discretization. Building on top of the SGBM for BSDE algorithm, we develop a Python demonstrator for pricing total valuation adjustment (XVA) in Chapter 5. This chapter shows the potential of using SGBM on a real-world risk management problem by focusing on XVA, an advanced risk management concept with increasing relevance. We also test the potential of developing a simple yet highly efficient code with SGBM by incorporating CUDA Python on the software engineering side.

In Chapter 6, we study two numerical schemes inspired by the branching method to solve BSDEs with discontinuous drivers. The numerical experiments reveal that the complex local polynomials based approximation is not efficient while a simple randomization procedure provides very good results. In order to motivate the study of this type of BSDEs, we extend a viscosity solution characterization to the case of an American option with a general payoff function in a multi-dimensional setting and link the viscosity solution of a semilinear PDE to a BSDE with a discontinuous driver.

The schemes for the expectation approximations in this thesis are well-justified and can be implemented immediately. Based on the results of this thesis, one can further study and solve the industrial application of BSDEs by identifying suitable BSDE models, developing the discretization scheme for BSDEs under different conditions and producing industrial level software.

SAMENVATTING

Backward Stochastic Differential Equations (BSDE's) zijn interessante wiskundige objecten met een groot aantal veelbelovende toepassingen. Binnen de financiële wiskunde kunnen ze worden gezien als een uitbreiding van de klassieke "replicating portfolio" methodologie. Ze kunnen worden gelinkt aan partiële differentiaalvergelijkingen (PDE's) via formules van het Feynman-Kac type. BSDE's worden ook toegepast in optimale besturingstheorie door middel van de Hamilton-Jacobi-Bellman-vergelijking (HJB).

Als gevolg van de vorige wereldwijde financiële crisis worden financiële instanties steeds meer verplicht om rekening te houden met verschillende risico's. Hierdoor is er veel vraag naar geavanceerde kwantitatieve methoden om deze risico's te bepalen en BSDE formuleringen zijn hier veelbelovend voor. Een van de belangrijkste obstakels bij het gebruiken van BSDE's in industriële toepassingen is de moeilijkheid om BSDE's analytisch of numeriek op te lossen.

De belangrijkste doelstellingen van dit onderzoek zijn het analyseren van verschillende numerieke methoden in de benadering van de verwachtingen die optreden bij het berekenen van deze risico's en het toepassen van deze methoden bij het numeriek oplossen van BSDE's. We richten ons op numerieke integratie voor verwachtingen/eindige maten, aangezien het merendeel van de BSDE-oplossers bestaat uit het berekenen van voorwaardelijke verwachtingen en deze te gebruiken voor het bepalen van een doelfunctie. Door middel van verschillende methoden voor het benaderen van de voorwaardelijke verwachtingen kunnen we meerdere methoden definiëren voor het oplossen van BSDE's die aan verschillende eisen voldoen. Bovendien kunnen deze resultaten ook gebruikt worden voor numerieke integratie.

In dit proefschrift richten we ons op de wiskundige eigenschappen van deze benaderingen. We zullen de fundamentele aannames bespreken, foutgrenzen afleiden en numerieke analyses uitvoeren. Het hoofddoel is om deze benaderingen te analyseren. We zullen ook ingaan op de financiële toepassingen van BSDE's, zoals in het prijzen van financiële derivaten en de berekening van XVA, en het programmeren ervan in Python en met GPU's.

De focus van deze scriptie ligt op Fourier expansie methoden en de Stochastic Grid Bundling Method (SGBM). De Fourier expansie methode maakt gebruik van het feit dat trigonometrische functies een basis vormen voor periodieke functies en functies met een begrensd domein. Deze methode maakt een benadering van integralen gebaseerd op een som van Fourier transformaties van de geïntegreerde maat. In deze scriptie voegen we daar ook elementen van lokalisatie aan toe. SGBM is een least-squares Monte Carlo methode die de zogenaamde regress-later en stochastische-partitietechnieken combineert. Met regress-later bedoelen we dat de integrand en de regressiebasis zijn gedefinieerd op dezelfde variabele (aangepast aan een filtratie in een latere tijdstap). Met een stochastische partitie bedoelen we dat we het domein van de conditionele stochastische variabelen verdelen in sets (de zogenaamde bundels) met behulp van Monte-Carlo-

simulaties. We definieëren vervolgens afzonderlijk voor elke bundel de benaderingen. In dit proefschrift leiden we een foutgrens af voor SGBM en bespreken we wanneer het niet mogelijk is om truncatie uit te voeren met regress-later-schema's. Tot slot bestuderen we kort de branching methode, een voorwaartse-simulatiemethode voor BSDE's, en de beperkingen van deze methode.

In Hoofdstuk 2 bestuderen we een gelokaliseerde Fourier-expansie method, de quick-SWIFT-methode, en geven we een alternatieve afleiding op basis van de periodieke wavelets. Deze methode combineert de effectiviteit van Fourier-gebaseerde methoden en de eenvoud van wavelet-gebaseerde methoden, wat leidt tot een algoritme dat zowel nauwkeurig als eenvoudig te implementeren is. Verder gebruiken we een anti-reflectieve randtechniek om het probleem van onnauwkeurige benaderingen rond de randen te verminderen. Om gelokaliseerde Fourier-expansie methoden uit te breiden naar hogere dimensies passen we in Hoofdstuk 3 roosterreeksen toe om een cosinus-expansie roostermethode te definieren. We vergelijken de lokalisatie op basis van roosters en op basis van wavelets.

In Hoofdstuk 4 wordt SGBM geïntroduceerd en wordt een bovengrens voor de fout afgeleid voor de eenvoudigste twee-stap versie van SGBM. Verder maken we ook een volledige foutenanalyse van de expliciete versie van BSDEs gebaseerd op tijdsdiscretisatie. Voortbouwend op SGBM voor BSDEs ontwikkelen we een Python-tool voor het prijzen van de total valuation adjustment (XVA) in Hoofdstuk 5. Dit hoofdstuk laat het potentiaal zien van SGBM in praktische risico-management problemen door het toe te passen voor XVA, iets wat in risico management met steeds hogere noodzaak berekend moet worden. Verder ontwikkelen we ook een eenvoudige maar zeer effectieve variant van SGBM door het te implementeren in CUDA Python.

In Hoofdstuk 6 bestuderen we twee numerieke schema's geïnspireerd door de branching methode om BSDE's met discontinue drivers op te lossen. De numerieke experimenten laten zien dat de benadering gebaseerd op complexe lokale polynomen niet efficiënt is, hoewel een simpele randomisatie-methode tot hele goede resultaten leidt. Verder dragen we ook bij aan de theorie van dit type BSDEs en breiden we een karakterisering van de viscositeitsoplossing uit naar een Amerikaanse optie met een algemene uitbetalingsfunctie in een hoog-dimensionale setting. We laten ook zien dat de viscositeitsoplossing van een semi-lineaire PDE gelinkt is aan een BSDE met een discontinue driver.

De methoden die in dit proefschrift worden besproken voor het benaderen van verwachtingen zijn allen goed onderbouwd en kunnen eenvoudig worden geïmplementeerd. Op basis van de resultaten van dit proefschrift kan men BSDE's gebruiken in en verder uitbreiden naar verschillende industriële toepassingen door het definiëren van geschikte BSDE-modellen en het ontwikkelen van discretisatie-methoden en het produceren van efficiënte en effectieve software.

1

INTRODUCTION

1.1. BACKGROUND

This thesis focuses on the numerical analysis of backward stochastic differential equations (BSDEs), especially on their application in finance.

In the wake of the previous global financial crisis, there has been a renewed focus on possible risks in financial markets and new regulations have been introduced requiring financial institutes to take new measures into their daily practices. For example, when trading financial derivatives, each party involved must take into account the risk of the counterparty default and post or collect collateral accordingly. This leads to the extra factors of so-called valuation adjustments in option pricing. The increasing complexity of financial model dynamics and the increasing number of risk factors rise the need to advance the available mathematical tools.

We believe that the system of BSDEs (see Equation (1.1)) is one of the tools that should be included in the context of industrial quantitative finance. In the context of finance, BSDEs can be seen as the natural consequence of the replicating portfolio pricing, a common technique in mathematical finance, see Section 1.2.3. Thus, it is only logical to include BSDEs for solving financial risk management problems.

Moreover, BSDEs have been a popular research subject within the academic circle ever since its general notion was introduced in [1] and [2]. Apart from all the research interests, which it has attracted as a mathematical object in its own right [3–5], there is a lot of effort in studying BSDEs' possible applications in economics, finance and game theory. There are works in the context of total valuation adjustment (XVA) [6], indifference pricing and risk measures [7], mean-field games [8], and more. A BSDE is a capable tool to model the new developments in risk management with all these support from academia.

Despite the above, we face a major obstacle when considering BSDEs in actual industrial applications. While BSDEs can cleanly and effectively model dynamics in finance, solving them to retrieve the necessary information is far from trivial. Finding an analytic solution for such equations is often difficult or even impossible, therefore, numerical

1

methods that can efficiently solve BSDEs within limited computational budget are an important component of applying BSDEs in the industry. This thesis thus is devoted to study whether we can design efficient algorithms to approximate BSDEs numerically and implement these algorithms in finance-inspired test cases.

In the next section, we introduce the system of BSDEs and the basic assumptions and settings we adopt in this thesis and describe the general discretization and approximation schemes for BSDEs. Moreover, we will further elaborate on the applications of BSDEs we consider in this thesis.

1.2. BACKWARD STOCHASTIC DIFFERENTIAL EQUATIONS

1.2.1. SETTING

We begin with the definition of BSDEs we consider in this thesis. Given a filtered complete probability space $(\Omega, \mathfrak{F}, \mathbb{F}, \mathbb{P})$, with $\mathbb{F} := (\mathfrak{F}_t)_{0\le t\le T}$ a filtration satisfying the usual conditions of being right-complete and $\mathbb{P}$-complete for a fixed terminal time $T > 0$, the process $W_t = (W_{1,t}, \dots, W_{d,t})^\top$ is a d-dimensional standard Brownian motion adapted to the filtration $\mathbb{F}$ and we are interested in solving the following systems of BSDEs, or so-called decoupled forward-backward stochastic differential equations (FBSDEs).

$$\begin{cases} dX_t = \mu(t, X_t)dt + \sigma(t, X_t)dW_t,\ X_0 = x_0, \\ dY_t = -f(t, X_t, Y_t, Z_t)dt + Z_t^\top dW_t,\ Y_T = g(X_T), \end{cases} \tag{1.1}$$

where $0 \le t \le T$. The functions $\mu : [0, T] \times \mathbb{R}^q \to \mathbb{R}^q$ and $\sigma : [0, T] \times \mathbb{R}^q \to \mathbb{R}^{q\times d}$ refer to the drift and the diffusion coefficients of the forward stochastic process, X, and $x_0 \in \mathfrak{F}_0$ is the initial condition for X. The function $f : [0, T] \times \mathbb{R}^q \times \mathbb{R} \times \mathbb{R}^d$ is called the driver function of the backward process and the terminal condition Y_T is given by $g(X_T)$ for a function $g : \mathbb{R} \to \mathbb{R}$. All stochastic integrals with W_t are of the Itô type.

Remark 1.1. This system is called a system of forward-backward equations since it contains both a forward process X, where the initial condition is adapted to $\mathfrak{F}_0$: x_0 is known, and a backward process with only the final condition adapted to $\mathfrak{F}_T$: $g(X_T)$ is known. The significance of this difference will be further elaborated later in this section. This is a decoupled system since the processes Y and Z are not involved in μ and σ and the backward dynamics have no influence on the forward dynamics.

While this is not the simplest setting for BSDEs, this is the most commonly used one and this choice ensures our numerical algorithms can be applied to most cases immediately. Further comments on how results from this thesis could be applied to other forms of BSDEs can be found in Chapter 7.

It is assumed that both $\mu(t, x)$ and $\sigma(t, x)$ are measurable functions that are uniformly Lipschitz in x and satisfy a linear growth condition in x. Therefore, there exists a unique strong solution for the forward stochastic differential equation,

$$X_t = x_0 + \int_0^t \mu(\tau, X_\tau)d\tau + \int_0^t \sigma(\tau, X_\tau)dW_\tau. \tag{1.2}$$

This process also satisfies the Markov property, namely $\mathbb{E}[X_\tau|\mathfrak{F}_t] = \mathbb{E}[X_\tau|X_t]$ for $\tau \ge t$, where $\mathbb{E}[\cdot]$ denotes the expectation with respect to probability measure $\mathbb{P}$.

Remark 1.2. While the above assumptions are enforced throughout the whole thesis, additional assumptions are required to ensure our numerical schemes are well-defined. These additional assumptions will be presented in the later chapters along with the numerical schemes as these assumptions have a direct impact on the derivations and errors of these schemes.

Given that a solution exists for the forward equation, a pair of adapted processes (Y_t, Z_t) is said to be the solution of the BSDE (1.1) if Y is a continuous real-valued adapted process such that $\mathbb{E}[\sup_{[0,T]} \| Y_t|^2] < \infty$, Z is a real-valued predictable process such that $\int_0^T |Z_t|^2 dt < \infty$ almost surely in $\mathbb{P}$ and the pair satisfies Equation (1.1).

In an application, the processes Y and Z normally are the values of interest that we are trying to calculate. In a replicating portfolio financial model for instance, Y would be the price process of the target financial derivative and Z would be related to the hedging process, see Section 1.2.3. We would like to calculate Y_0 to get the derivative price at the initial time in this case.

As stated before, it is unlikely for us to identify elementary expressions for the processes Y and Z. Therefore, we have to rely on numerical solutions.

1.2.2. DISCRETIZATION

While alternatives exist, one example being [9], the majority of numerical schemes for solving BSDEs is based on the ideas of discretization and backward recursion. For this type of schemes, we construct a time grid $\pi = \{0 = t_0 < \ldots < t_P = T\}$ on the interval $[0, T]$ and for $p = P-1, P-2, \ldots 0$, we use the known information (Y_{p+1}, Z_{p+1}) and an approximation function $\mathfrak{q}$ to approximate $(Y_p, Z_p) = \mathfrak{q}(Y_{p+1}, Z_{p+1})$.

In particular, we denote and let $\Delta_p := t_{p+1} - t_k$, $\Delta W_{l,p} := W_{l,t_{p+1}} - W_{l,t_p}$, and $\Delta W_p := (\Delta W_{1,p}, \ldots, \Delta W_{d,p})^\top$ be the time-step, the Brownian motion increment along the l-th dimension and the Brownian motion increment, respectively, for $p \in \{0, \ldots, P-1\}$. Note that $\Delta W_{l,p} \sim \mathcal{N}(0, \Delta_p)$ is a normally distributed process for all l and p. The backward dynamics in Equation (1.1) would then be discretized along the time grid π, so that we can derive the approximation dynamics.

The basic discretization of the forward dynamics (1.2) is well studied. For example, one can use the classic Euler-Maruyama scheme to define a discretized forward process X^π:

$$X^\pi_{t_0} := x_0,\ X^\pi_{t_{p+1}} := X^\pi_{t_p} + \mu(t_k, X^\pi_{t_p})\Delta t + \sigma(t_p, X^\pi_{t_p})\Delta W_p, \quad p = 0, \ldots, P-1.$$

However, the discretization for the backward dynamics is not as simple. One key difficulty in solving a BSDE is that the pair (Y_t, Z_t) must be adapted to the underlying filtration. The terminal condition Y_T is given by $g(X_T)$, where g is a deterministic function. Therefore, Y_T is adapted to the filtration $\mathfrak{F}_T$ and a naive Euler discretization on the backward equation fails to produce an adapted solution.

Adopting the notation $\mathbf{X} = (X, Y, Z)$, we can observe from the backward equation,

$$Y_{t_p} = Y_{t_{p+1}} + \int_{t_p}^{t_{p+1}} f(\tau, \mathbf{X}_\tau) d\tau - \int_{t_p}^{t_{p+1}} Z_\tau^\top dW_\tau, \tag{1.3}$$

that a simple discretization is not sufficient to produce an approximation. It is because we would require the value of $Y_{t_{p+1}}$ to approximate Y_{t_p}, but $Y_{t_{p+1}}$ is not $\mathfrak{F}_{t_p}$ adapted. For

further discussion on this, the reader may refer to the introduction in [10].

To tackle this problem, we follow the standard methods in the literature. By taking conditional expectations on both sides of Equation (1.3) and approximating the time integral by a θ-time discretization, as in [11] and [12], we get

$$\begin{aligned} Y_{t_p} &= \mathbb{E}_p[Y_{t_{p+1}}] + \int_{t_p}^{t_{p+1}} \mathbb{E}_p[f(\tau, \mathbf{X}_\tau)]d\tau \\ &\approx \mathbb{E}_p[Y_{t_{p+1}}] + \Delta t\theta_1 f(t_p, \mathbf{X}_{t_p}) + \Delta t(1-\theta_1)\mathbb{E}_p[f(t_{p+1}, \mathbf{X}_{t_{p+1}})], \theta_1 \in [0,1]. \end{aligned}$$

The notation $\mathbb{E}_p$ and $\mathbb{E}_p^x$ are defined as

$$\mathbb{E}_p[\cdot] := \mathbb{E}[\cdot|X_{t_p}] = \mathbb{E}[\cdot|\mathfrak{F}_{t_p}], \text{ and } \mathbb{E}_p^x[\cdot] = \mathbb{E}[\cdot|X_{t_p} = x].$$

For the process Z, we derive a recursive approximation formula by multiplying by ΔW_p to both sides of Equation (1.3) and taking conditional expectations,

$$\begin{aligned} 0 =& \mathbb{E}_p[Y_{t_{p+1}}\Delta W_{p+1}] + \int_{t_p}^{t_{p+1}} \mathbb{E}_p[f(\tau, \mathbf{X}_\tau)\Delta W_{p+1}]d\tau - \int_{t_p}^{t_{p+1}} \mathbb{E}_p[Z_\tau]d\tau \\ \approx& \mathbb{E}_p[Y_{t_{p+1}}\Delta W_{p+1}] + \Delta t(1-\theta_2)\mathbb{E}_p[f(t_{p+1}, \mathbf{X}_{t_{p+1}})\Delta W_{p+1}] \\ & - \Delta t\theta_2 Z_{t_p} - \Delta t(1-\theta_2)\mathbb{E}_p[Z_{t_{p+1}}], \theta_2 \in (0,1]. \end{aligned}$$

Again, we applied the θ-method to the time integral. However, the two parameters for the θ-method, θ_1 and θ_2, need not necessarily be the same. We define a discrete-time approximation (Y^π, Z^π) for (Y, Z):

$$Y_{t_p}^\pi := g(X_{t_p}^\pi), \quad Z_{t_p}^\pi = (\nabla g(X_{t_p}^\pi)\sigma(t_p, X_{t_p}^\pi))^\top, \tag{1.4a}$$

for $p = P-1, \dots, 0$,

$$Z_{t_p}^\pi := -\frac{1-\theta_2}{\theta_2}\mathbb{E}_p[Z_{t_{p+1}}^\pi] + \frac{1}{\theta_2 \Delta t}\mathbb{E}_p[Y_{t_{p+1}}^\pi \Delta W_p] + \frac{1-\theta_2}{\theta_2}\mathbb{E}_p[f(t_{p+1}, \mathbf{X}_{t_p}^\pi)\Delta W_{p+1}], \tag{1.4b}$$

$$Y_{t_p}^\pi := \mathbb{E}_p[Y_{t_{p+1}}^\pi] + \Delta t\theta_1 f(t_p, \mathbf{X}_{t_p}^\pi) + \Delta t(1-\theta_1)\mathbb{E}_p[f(t_{p+1}, \mathbf{X}_{t_{p+1}}^\pi)], \tag{1.4c}$$

Again, we used the simplifying notation $\mathbf{X}^\pi = (X^\pi, Y^\pi, Z^\pi)$ [1]. Note that various combinations of θ_1 and θ_2 give different approximation schemes. We have an explicit scheme for Y^π if $\theta_1 = 0$, and an implicit scheme otherwise. The variable $Z_{t_p}^\pi$ depends on $\mathbb{E}_p[Z_{t_{p+1}}^\pi]$ only if $\theta_2 \neq 1$.

Also, since the terminal processes $Y_{t_p}^\pi$ and $Z_{t_p}^\pi$ are deterministic with respect to $X_{t_p}^\pi$ and X^π is a Markov process, one can show by induction that $Y_{t_p}^\pi = y_p^\pi(X_{t_p}^\pi)$, $Z_{t_p}^\pi = z_p^\pi(X_{t_p}^\pi)$, where z_p^π and y_p^π are deterministic functions related to the discretization scheme. We shall use the notation $(y_p^\pi(x), z_p^\pi(x))$ when we wish to emphasize the dependence of our approximation in this thesis.

From Equation (1.4), one can observe that there are two components in each time step in this type of BSDE approximations, the conditional expectations in the form of

[1] In most applications, the forward dynamics are also discretized and approximated by a Markov process. Therefore, we use X^π here instead of X.

$\mathbb{E}_p[Y_{t_{p+1}}], \mathbb{E}_p[Y_{t_{p+1}}\Delta W_p]$, etc. and deterministic functions to calculate Y_{t_p} and Z_{t_p}. With the function given in Equation (1.4), all we need is an efficient way to approximate multiple conditional expectations at each time-step.

The majority of so-called probabilistic methods for solving BSDEs relies on this time discretization idea (even if the exact functions are different), they differ by the methods for calculating the appearing conditional expectations. Techniques used include least-squares Monte Carlo regression in [13–17], chaos decomposition formulas in [18], cubature methods in [19], Fourier expansion technique in [12], among others.

In Chapter 2 and Chapter 4, we will follow the same line of reasoning. We will derive complete numerical algorithms for approximating BSDEs based on the approximating dynamics in Equation (1.4) and different approximation methods for the conditional expectations. We will use an alternative formulation in Chapter 6, but it is still a combination of conditional expectation approximation and deterministic functions.

Therefore, the result in this thesis is not just applicable to BSDEs, but it also contributes to the study of finite measure integration approximation in general. In Section 1.3, we introduce the three different schemes we use to approximate expectations in this thesis. But first, we discuss two applications of BSDEs that are related to this thesis. These applications give an insight into the usefulness of BSDEs.

1.2.3. BSDEs AND REPLICATING PORTFOLIOS

One of the particularly interesting interpretations of BSDEs is to consider them as a natural product of the replicating portfolio pricing method, by which we mean that if we apply the famous replicating portfolio pricing argument in mathematical finance, we will deduce a relevant BSDE and the pricing information is the solution of the BSDE.

In this subsection, we review a simple replicating portfolio argument in a heuristic manner and derive a system of BSDEs. The argument in this section serves as an example of the above connection and we may skip over some details. For further information about financial mathematics and replicating portfolios, one refers to [20]. For the mathematical setting of BSDEs in this context, one may consult other sections of this thesis.

Consider a market with one risky asset X, following the Black and Scholes dynamics and one riskless asset B with riskfree rate $\bar{r}$, and they progress according to the following stochastic dynamics.

$$\begin{aligned} dB_t &= \bar{r}B_t dt \\ dX_t &= \bar{\mu}X_t dt + \bar{\sigma}X_t dW_t. \end{aligned} \tag{1.5}$$

Assume that there is a financial product that will provide a payoff based on a deterministic function g acting on the value of the risky asset X at time T: $g(X_T)$. A common pricing/hedging technique is to construct a self-financing portfolio V consisting of only the above two assets and its value should match $g(X_T)$ at time T. Based on the no arbitrage principle, the value of this portfolio should equal the target financial product at all time before T, especially at the initial time 0.

With the self-financing assumption and denoting the amount of riskless assets and risky assets in portfolio V at time t by $(\mathfrak{a}_t, \mathfrak{b}_t)$, we have

$$\begin{aligned} V_t &= \mathfrak{a}_t B_t + \mathfrak{b}_t X_t \\ dV_t &= \mathfrak{a}_t dB_t + \mathfrak{b}_t dX_t \end{aligned}$$

$$
\begin{aligned}
&= (\bar{r}\mathfrak{a}_t B_t + \bar{\mu} X_t \mathfrak{b}_t)dt + \bar{\sigma} X_t \mathfrak{b}_t dW_t \\
&= \left[\bar{r} V_t + \frac{\bar{\mu}-\bar{r}}{\bar{\sigma}} \bar{\sigma} X_t \mathfrak{b}_t X_t\right] dt + \bar{\sigma} X_t \mathfrak{b}_t dW_t. \qquad (1.6)
\end{aligned}
$$

Combining Equation (1.5), Equation (1.6) and the replicating requirement, we have

$$
\begin{cases}
dX_t = \bar{\mu} X_t dt + \bar{\sigma} X_t dW_t; \\
dV_t = \left[\bar{r} V_t + \frac{\bar{\mu}-\bar{r}}{\bar{\sigma}} \bar{\sigma} \mathfrak{b}_t X_t\right] dt + \bar{\sigma} \mathfrak{b}_t X_t dW_t,\ V_T = g(X_T).
\end{cases}
$$

This is indeed a BSDE system with $(V, \bar{\sigma}\mathfrak{b}_t X_t)$ being the solution of the system. From this point on, there are many different methods to calculate or approximate the product's price at the beginning, V_0, for example, we can deduce the Black-Scholes partial differential equation (PDE) after applying Itô's formula on V and use a numerical algorithm for the PDE to get the price.

However, when the asset models in Equation (1.5) or the market dynamics in Equation (1.6) change, the PDE method or other schemes may not be feasible anymore but the fundamental idea above is still valid and in general we can still deduce a system of BSDEs. That is one of the reasons why the BSDE framework is such a strong tool in mathematical finance research.

In this thesis, the readers will also see some applications of BSDEs of this type. In Section 2.5.2, the same argument as above is used to price a European option. In Section 2.5.3, the market dynamics are changed such that one cannot finance by the riskless interest rate but with a higher interest rate and we price through an alternative BSDE. Finally, following the same idea but using market dynamics involving counterparty risk and margin requirement, Lesniewski and Richer [21] derived a system of BSDEs to price financial products. In Chapter 5, we use a special case of Lesniewski and Richer's BSDE to design an XVA approximation demonstrator.

1.2.4. BSDEs AND PARTIAL DIFFERENTIAL EQUATIONS

In the previous subsection, we briefly mentioned that after we have deduced a replicating portfolio BSDE, one of the ways to perform the calculation is to transform the BSDE into a PDE. In fact, the connection between BSDEs and PDEs is a key factor for the BSDE's popularity ever since its introduction.

Let's recall the function $\mu : \mathbb{R}^+ \times \mathbb{R}^d \to \mathbb{R}^d$ and $\sigma : \mathbb{R}^+ \times \mathbb{R}^d \to \mathbb{R}^{d\times d}$ from Equation (1.2) which satisfy the Lipschitz and linear growth conditions. For each $(t,x) \in [0,T) \times \mathbb{R}^d$, we denote by $\{X_s^{t,x}, t \le s \le T\}$ the unique strong solution of the following stochastic differential equation in this subsection:

$$
\begin{cases}
dX_s^{t,x} &= \mu(s, X_s^{t,x})ds + \sigma(s, X_s^{t,x})dW_s,\ t \le s \le T \\
X_s^{t,x} &= x.
\end{cases}
$$

In [2], the authors have proven the following two theorems.

Theorem 1.1 (Adapted from [2], Theorem 3.1)**.** *If $u : [0,T] \times \mathbb{R}^d$ is differentiable with respect to the first variable and twice differentiable with respect to the second one, and it solves*

$$
\begin{cases}
\frac{\partial u}{\partial t}(t,x) + \mathscr{L}u(t,x) + f(t,x,u(t,x),(\nabla u\sigma)(t,x)) &= 0 \\
u(T,x) &= g(x),
\end{cases} \qquad (1.7)
$$

where

$$\mathscr{L} = \frac{1}{2}\sum_{i,j=1}^{d}(\sigma\sigma^{\top})_{ij}(t,x)\frac{\partial^2}{\partial x_i \partial x_j} + \sum_{i=1}^{d} b_i(t,x)\frac{\partial}{\partial x_i}.$$

Then $u(t,x) = Y_t^{t,x}$, $t \geq 0$, $x \in \mathbb{R}^d$, *where* $\{(Y_s^{t,x}, Z_s^{t,x}); t \leq s \leq T\}_{t\geq 0, x\in\mathbb{R}^d}$ *is the unique solution of the BSDE*

$$Y_s^{t,x} = g(X_T^{t,x}) + \int_s^T f(r, X_r^{t,x}, Y_r^{t,x}, Z_r^{t,x})dr - \int_s^T Z_r^{t,x} dW_r, \ t \leq s \leq T. \tag{1.8}$$

Theorem 1.2 (Adapted from [2], Theorem 4.3)**.** *Assume that* $f(t,x,y,z)$ *is globally Lipschitz with respect to* (x,y,z) *uniformly in* t *and* $g(x)$ *is Lipschitz in* x*, the function* $u(t,x) := Y_t^{t,x}$ *(see Equation* (1.8)*) is the unique viscosity solution of the backward parabolic PDE* (1.7)*.*

The above results and followup works established a close link between BSDEs and PDEs. In particular, this connection provides the opportunity of solving PDEs (in high dimensions) with stochastic methods, which we are interested in.

In Chapter 6, we work on the connection between a particular type of PDE inspired by American option pricing and BSDEs and use the result to derive a Monte-Carlo pricing method. For further information on viscosity solutions and the PDE-BSDE connection in general, readers are referred to that chapter.

1.3. NUMERICAL METHODS

In this section, we briefly discuss the three schemes for approximating expectations $\mathbb{E}[\mathfrak{f}(X)]$ we use in this thesis.

1.3.1. FOURIER EXPANSION METHODS

The first scheme that we are interested in, is based on Fourier series. Fourier based approximation has been one of the most popular ways to approximate expectations as the Fourier transform of a probability measure is usually more widely known than its density. In this particular version of Fourier method, the Fourier expansion method, we project the integrand $\mathfrak{f}$ to a function space spanned by trigonometric functions and calculate the expectation of the projection as our approximant. There are two main advantages of this scheme. First, this scheme does not involve inverse Fourier transforms, which are expensive to calculate. Second, the main approximant is in the form of a simple finite sum, therefore it can be computed efficiently. We will provide further details on this method later, but we review its application history in mathematical finance here.

While approximation in similar form can be found in other contexts, for example [22], it was introduced to European option pricing with the COS method in [23]. This article used the framework that the fair value of a financial option can be expressed as the expectation of the discounted payoff function of such option under the risk neutral measure and applied a cosine series approximation to this expectation.

Ever since, there has been numerous extensions of the COS method, including pricing Bermudan options [24], finding ruin probability [25], solving backward stochastic differential equations [12] and computation of valuation adjustment [26].

1

The original COS method works as follows. It is based on the fact that any function defined in the bounded interval $[a,b]$, it can be represented by a cosine series:

$$\mathfrak{f}(x) = \sum_{k=0}^{\infty}{}' A_k \cos\left(k\pi\frac{x-a}{b-a}\right),$$

with A_k being the cosine coefficient[2]

$$A_k = \frac{2}{b-a}\int_a^b \mathfrak{f}(x)\cos\left(k\pi\frac{x-a}{b-a}\right)dx.$$

Therefore, by replacing the original function with a truncated cosine series, we may express the expectation as a sum of cosine transforms:

$$\mathbb{E}[\mathfrak{f}(X)] \approx \sum_{k=0}^{N-1}{}' A_k \mathbb{E}\left[\cos\left(k\pi\frac{X-a}{b-a}\right)\right],$$

and the cosine transform can be simply calculated from the Fourier transform.

There are two disadvantages when applying the original COS method. The cosine coefficients A_k may be difficult to calculate, especially in a time recursion situation, as in Equation (1.4). Also, while the derivation can be extended easily to higher dimensions with a tensor argument, the COS method suffers from the curse of dimensionality.

In Chapter 2, we address the first point by further studying the SWIFT method introduced in [27] and its quick variant from [28]. The SWIFT methods were derived in a similar way as the original COS method. Instead of using the cosine series expansion, it is based on the following expansion formula.

$$\mathfrak{f}(x) = \frac{1}{N}\sum_{r=1-N}^{N} B_r \varphi_{N,r}(x), \tag{1.9}$$

where

$$\varphi_{N,r}(x) = \sum_{k=1}^{N}\cos\left(\frac{2k-1}{2N}\pi(2^{\vartheta}x-r)\right).$$

Since the basis $\varphi_{N,r}$ is constructed from cosine functions, its expectation $\mathbb{E}\left[\varphi_{N,r}(X)\right]$ can again be calculated by Fourier transforms. Thus, the approximation

$$\mathbb{E}[\mathfrak{f}(X)] \approx \frac{1}{N}\sum_{r=1-N}^{N} B_r \mathbb{E}[\varphi_{N,r}(X)]$$

preserves all the advantages of the Fourier expansion methods.

More importantly, we have

$$B_r = \frac{2^{\vartheta}}{N}\int_{-2^{-\vartheta}N}^{2^{-\vartheta}N}\mathfrak{f}(x)\varphi_{N,r}(x)dx \approx \mathfrak{f}\left(\frac{r}{2^{\vartheta}}\right), \tag{1.10}$$

which means that the expansion coefficient can be read from the integrand immediately. This simplifies finding expansion coefficients and addresses the first disadvantage of the

[2] The notation $\sum'$ means a summation with the first term weighted by half.

COS method. The SWIFT method is our starting point in Chapter 2. We rigorously derive the Equations 1.9 and 1.10, beginning with Theorem 2.1, on top of deriving a numerical scheme for BSDEs.

In Chapter 3, we address the problem of the curse of dimensionality with the results from lattice sequences.

1.3.2. STOCHASTIC GRID BUNDLING METHOD

As stated in the previous subsection, the curse of dimensionality is one of the main concerns in designing a numerical scheme. Notably, higher-dimensional PDEs are common in the financial context. When facing higher-dimensional problems, the Monte-Carlo scheme is the most popular choice as it does not suffer from the curse of dimensionality and it is general as it can be applied to many different integration problems. Therefore, we would also consider one particular version of the Monte-Carlo method, the Stochastic Grid Bundling Method (SGBM) and its application to BSDEs.

SGBM was first invented to price Bermudan options ([29, 30]). It is designed for expectations calculation with a high-dimensional random variable in a time-recurring setting and to avoid applying so-called nested Monte-Carlo schemes. By nested Monte-Carlo, we mean that in order to calculate the conditional expectation: $\mathbb{E}[\mathfrak{f}(X_{t+\Delta t})|X_t = x]$, we simulate N samples of the dynamics X: $\{X^k\}_{1\leq k\leq N}$, starting at time t with $X_t^n = x$ and use the approximation $\mathbb{E}[\mathfrak{f}(X_{t+\Delta t})|X_t = x] \approx \frac{1}{N}\sum_{k=1}^{N}\mathfrak{f}(X_{t+\Delta t}^k)$. Nested Monte-Carlo is computationally expensive, especially for Equation (1.4), where we must progress backwards on a time grid.

SGBM is a non-nested Monte Carlo scheme and only performs one simulation step with the process X starting at time 0 till terminal time T. This scheme achieves accurate approximations for $\mathbb{E}[\mathfrak{f}(X_{t+\Delta t})|X_t]$, where $0 < t < T$ and $\Delta t > 0$ by applying *bundling* and *regress-later* techniques, two new advanced techniques for least-squares Monte-Carlo methods.

First is the bundling approach at time t, where we partition the whole simulation cloud into non-overlapping subsets based on the realized values $\{X_t^k\}_{1\leq k\leq N}$ at time t, then we perform our regression separately within these bundles. Namely, we approximate $\mathbb{E}[\mathfrak{f}(X_{t+\Delta t})|X_t \in \mathscr{B}]$ for some subset $\mathscr{B}$ instead of $\mathbb{E}[\mathfrak{f}(X_{t+\Delta t})|X_t]$ as a whole function defined on $\mathbb{R}^d$. The idea is that for a smaller domain $\mathscr{B}$, the processes $\{X_{t+\Delta t}^k|X_t^k \in \mathscr{B}\}$ would share similar "characteristics" such that we can use a simpler approximation scheme and have accurate results. A non-overlapping partition also facilitates parallel computing.

Within each bundle, we perform a least-squares regression. However, unlike usual least-squares regression for dynamic programming problems, where the target function is defined at the end of a time interval while the basis functions are measured at the beginning of the time interval[3], we use basis functions defined at the end of the interval with regress-later methodology. In this case, the conditional expectation is approximated with the analytic expectation of the basis functions.

It has been shown in the literature that regress-later schemes remove the statistical error from the regress-now approach and theoretically can have better results. Therefore, it is used in the SGBM algorithm. For further reference to regress-later schemes,

[3] It is called the regress-now approach.

one might check out [31].

To facilitate a better understanding of our material, we present SGBM in a simplified algorithmic context here. The goal of SGBM is to find simulated approximations for the conditional expectation $\mathbb{E}[\mathfrak{f}(X_{t+\Delta_t})|X_t]$ defined on the Markovian stochastic process X and adapted to the filtration at time t: $\mathfrak{F}_t$.

We first simulation $N \times B$ samples for process X: $(X^n)_{n=1,2,\ldots,N\times B}$ starting at a given initial condition at time 0, through time t, and ending at time $t+\Delta t$. This is the simulation step.

Next, we rank the samples X^n according to a real-valued ranking function $\mathfrak{S}$ and the samples' realized values at time t. So, we line up the evaluated results as

$$\mathfrak{S}(X_t^{\#1}) \leq \mathfrak{S}(X_t^{\#2}) \leq \ldots \mathfrak{S}(X_t^{\#N\times B}).$$

Afterwards, we designate the first B samples with smallest ranking values in the first bundle $\mathscr{B}_1$, denoted as $(X^{1,n})_{1\leq n\leq N}$, the next B samples into the second bundle, denoted as $(X^{2,n})_{1\leq n\leq N}$, so on and so forth. This is the partition step.

Within each bundle, we perform the regression step. We have a predefined set of real-valued basis functions $(\eta_1,\ldots,\eta_Q)$ for each bundle. We develop a localized approximation by solving the matrix equation

$$\begin{pmatrix} \eta_1(X_{t+\Delta t}^{b,1}) & & \eta_Q(X_{t+\Delta t}^{b,1}) \\ & \ddots & \\ \eta_1(X_{t+\Delta t}^{b,n}) & & \eta_Q(X_{t+\Delta t}^{b,n}) \end{pmatrix} \begin{pmatrix} \alpha_1^b \\ \vdots \\ \alpha_Q^b \end{pmatrix} = \begin{pmatrix} \mathfrak{f}(X_{t+\Delta_t}^{b,1}) \\ \vdots \\ \mathfrak{f}(X_{t+\Delta_t}^{b,n}) \end{pmatrix}$$

to identify the regression coefficients $(\alpha_q^b)_{1\leq q\leq Q}$ for each bundle. Denote the basis functions matrix on the left hand side of the above equation by $\mathscr{I}$, we solve the above equation by first multiplying both sides of the equation by $\mathscr{I}$ on the left. The matrix $\mathscr{I}^\top\mathscr{I}$ is invertible and the regression coefficient is given by

$$\left(\alpha_1^b,\ldots,\alpha_Q^b\right)^\top = (\mathscr{I}^\top\mathscr{I})^{-1}\mathscr{I}^\top\left(\mathfrak{f}(X_{t+\Delta_t}^{b,1}),\ldots,\mathfrak{f}(X_{t+\Delta_t}^{b,n})\right)^\top$$

This is the regression step.

Finally, the approximation samples are given by the combination of results from all bundles. For each sample X_t^n, the approximated samples for the conditional expectation are given by

$$\mathbb{E}[\mathfrak{f}(X_{t+\Delta t})|X_t = X_t^n] \approx \sum_{b=1}^{B} \mathbf{1}_{\mathscr{B}_b}(X_t^n) \sum_{q=1}^{Q} \alpha_q^b \mathbb{E}[\eta_q(X_{t+\Delta t})|X_t = X_t^n].$$

We determine which bundle the sample belongs to, and use the localized regression for that bundle to construct an approximation for the target function. In particular, the resulting samples can be used in the regression step in a backward-in-time recurring setting.

Equipped with the basic notion of SGBM, we investigate its properties further in this thesis. Chapters 4 and 5 concern the application of SGBM with the time discretization scheme (1.4) to solving BSDEs, both from the theoretical and practical point of view.

1.3.3. BRANCHING METHODS

Alternatively, we can derive a Monte-Carlo scheme to solve BSDEs based on directly taking expectations on the backward dynamics. Recall the notation in Section 1.2.4, we here consider a special BSDE of the form

$$Y_s^{x,t} = g(X_T^{x,t}) + \int_s^T \left(\sum_l a_l (Y_r^{x,t})^l \right) dr - \int_s^T Z_r^{t,x} dW_r, \ t \le s \le T,$$

the function u as defined in Theorem 1.2 has the following expression:

$$u(t,x) = \mathbb{E}_t^x \left[\frac{g(X_T)}{\bar{F}(T-t)} \mathbf{1}_{\tau \ge T-t} + \mathbf{1}_{\tau < T-t} \sum_l \frac{a_l}{\rho(\tau)} u(t+\tau, X_{t+\tau}^{t,x})^l \right]^{4}.$$

Here, τ is a random time, independent of the Brownian motion W_t and

$$\bar{F}(t) := \mathbb{P}[\tau > t] = \int_t^\infty \rho(s) ds, \ t \ge 0.$$

Heuristically, the above expression can be approximated by multiple recurring forward simulations of a killing time τ and the corresponding forward process X_t up to the killing time τ. This pure forward simulation scheme for a BSDE, based on a so-called branching process, was first introduced in [32] and [33].

In practice, instead of conducting the branching simulations on the whole predefined time domain $[s,T]$, we can also work with the time grid $\pi = \{0 = t_0 \le t_1 \le \dots t_N = T\}$ and use the following recurring formula:

$$u(t_n,x) = \mathbb{E}_{t_n}^x \left[\frac{u(t_{n+1}, X_{t_{n+1}}^{t_n,x})}{\bar{F}(t_{n+1}-t_n)} \mathbf{1}_{\tau > t_{n+1}-t_n} + \mathbf{1}_{\tau \le t_{n+1}-t_n} \sum_l \frac{\alpha_l}{\rho(\tau)} (u(t_n+\tau, X_{t_n+\tau}^{t_n,x}))^l \right].$$

A backward iteration is then used to complete the whole approximation. At any time period, we compute the branching algorithm with the above formula at predefined space grid points and an interpolation is performed to recover the function u at that time points across the whole domain. Then, we move on to the previous time point. By restricting ourselves to small time intervals, the algorithm will be stable. Again, one can see that the approximation of the BSDE is a combination of a deterministic recurring scheme and a conditional expectation calculation.

There has been work on generalizing the application of the branching processes-based scheme. In [34], the authors proposed a new numerical scheme based on branching processes for a more general class of BSDEs:

$$Y_s^{t,x} = g(X_T^{t,x}) + \int_s^T f(X_r^{t,s}, Y_r^{t,s}) dr - \int_s^T Z_r^{t,x} dW_r, \ t \le s \le T,$$

In Chapter 6, we extend the branching process to American option inspired BSDEs.

[4] The expression is an alternative formulation of a branching process used by Prof. Xavier Warin. This formulation has the advantage of reducing the variance in the Monte Carlo simulation.

1.4. OUTLINE OF THE THESIS

The remainder of the thesis is organized as follows.

In Chapter 2 we rigorously derive the SWIFT method, mentioned in the previous subsection, for expectation approximation using trigonometric wavelets. We propose a numerical algorithm for BSDEs based on time discretization and the SWIFT method. Furthermore, we mitigate the problem of errors occurring near the computational boundaries by means of an antireflective boundary technique, giving an improved approximation. In Chapter 3, we extend the localized Fourier expansion method to higher dimensions by using a lattice sequence from Quasi Monte-Carlo rules. We study the error of this scheme and compare the results in Chapter 2 and 3.

Chapter 4 contains the theoretical justification of a simple two-step SGBM regression and a full explicit time discretization SGBM for a BSDE algorithm. We will also establish an upper error bound for the local regression and a full error analysis for the complete scheme. Numerical experiments on more general time discretization schemes are also included in Chapter 4. These SGBM algorithms for BSDE schemes are applied to an XVA inspired BSDE in Chapter 5. We present a Python demonstrator based on the SGBM algorithm and make use of GPU computing in this chapter and show results from applying this code to pricing problems up to 40 dimensions.

Chapter 6 is based on a joint work in the CEMRACS 2017 research school, in which we prove that the price of a general American option can be seen as a unique viscosity solution of a non-linear parabolic PDE. Using the connection between PDEs and BSDEs and the branching processes scheme, we derive stochastic methods for approximating this viscosity solution.

In Chapter 7 we draw conclusions for each numerical scheme and for the complete thesis work. An outlook and further research from this thesis will also be discussed.

While we aim for consistent notations and unifying definitions in this thesis, since this thesis covers a broad range of topics with various conventions, the meaning of the notations may change from chapter to chapter. We ask the readers' understanding.

REFERENCES

[1] E. Pardoux and S. G. Peng, *Adapted solution of a backward stochastic differential equation,* Systems & Control Letters **14**, 55 (1990).

[2] E. Pardoux and S. G. Peng, *Backward stochastic differential equations and quasilinear parabolic partial differential equations,* in *Stochastic Partial Differential Equations and Their Applications: Proceedings of IFIP WG 7/1 International Conference University of North Carolina at Charlotte, NC June 6–8, 1991*, edited by B. L. Rozovskii and R. B. Sowers (Springer Berlin Heidelberg, Berlin, Heidelberg, 1992) pp. 200–217.

[3] P. Briand and R. Carmona, *BSDEs with polynomial growth generators,* Journal of Applied Mathematics and Stochastic Analysis **13**, 207 (2000).

[4] I. Kharroubi, J. Ma, H. Pham, and J. Zhang, *Backward SDEs with constrained jumps and quasi-variational inequalities,* The Annals of Probability **38**, 794 (2010).

[5] S. N. Cohen, *Undiscounted Markov chain BSDEs to stopping times,* Journal of Applied Probability **51**, 262 (2014).

[6] S. Crépey, R. Élie, W. Sabbagh, and S. Song, *When capital is a funding source: The XVA anticipated BSDEs,* Retrieved from "https://math.maths.univ-evry.fr/crepey/papers/FVA-ABSDE.pdf" (2017).

[7] W. F. Chong, Y. Hu, G. Liang, and T. Zariphopoulou, *An ergodic bsde approach to forward entropic risk measures: representation and large-maturity behavior,* Finance and Stochastics **23**, 239 (2019).

[8] R. Buckdahn, B. Djehiche, J. Li, and S. G. Peng, *Mean-field backward stochastic differential equations: A limit approach,* The Annals of Probability **37**, 1524 (2009).

[9] W. E, M. Hutzenthaler, A. Jentzen, and T. Kruse, *Multilevel Picard iterations for solving smooth semilinear parabolic heat equations,* arXiv e-prints , arXiv:1607.03295 (2016).

[10] B. Bouchard and N. Touzi, *Discrete-time approximation and Monte-Carlo simulation of backward stochastic differential equations,* Stochastic Processes and their Applications **111**, 175 (2004).

[11] A. Iserles, *A First Course in the Numerical Analysis of Differential Equations*, 2nd ed. (Cambridge University Press, 2008) Cambridge Books Online.

[12] M. J. Ruijter and C. W. Oosterlee, *A Fourier cosine method for an efficient computation of solutions to BSDEs,* SIAM Journal on Scientific Computing **37**, A859 (2015).

[13] C. Bender and J. Steiner, *Least-squares Monte Carlo for backward SDEs,* in *Numerical Methods in Finance: Bordeaux, June 2010,* edited by R. A. Carmona, P. Del Moral, P. Hu, and N. Oudjane (Springer Berlin Heidelberg, Berlin, Heidelberg, 2012) pp. 257–289.

[14] D. Crisan, K. Manolarakis, and N. Touzi, *On the Monte Carlo simulation of BSDEs: An improvement on the Malliavin weights,* Stochastic Processes and their Applications **120**, 1133 (2010).

[15] J.-P. Lemor, E. Gobet, and X. Warin, *Rate of convergence of an empirical regression method for solving generalized backward stochastic differential equations,* Bernoulli **12**, 889 (2006).

[16] E. Gobet, J. G. López-Salas, P. Turkedjiev, and C. Vázquez, *Stratified regression Monte-Carlo scheme for semilinear PDEs and BSDEs with large scale parallelization on GPUs,* SIAM Journal on Scientific Computing **38**, C652 (2016).

[17] D. Ding, X. Li, and Y. Liu, *A regression-based numerical scheme for backward stochastic differential equations,* Computational Statistics **32**, 1357 (2017).

[18] P. Briand and C. Labart, *Simulation of BSDEs by Wiener chaos expansion,* The Annals of Applied Probability **24**, 1129 (2014).

[19] D. Crisan and K. Manolarakis, *Solving backward stochastic differential equations using the cubature method: Application to nonlinear pricing,* SIAM Journal on Financial Mathematics **3**, 534 (2012).

[20] S. Shreve, *Stochastic Calculus for Finance II: Continuous-Time Models*, Springer Finance Textbooks No. v. 11 (Springer, 2004).

[21] A. Lesniewski and A. Richter, *Managing counterparty credit risk via BSDEs,* ArXiv e-prints , arXiv:1608.03237 (2016).

[22] J. Abate and W. Whitt, *The Fourier-series method for inverting transforms of probability distributions,* Queueing Systems **10**, 5 (1992).

[23] F. Fang and C. W. Oosterlee, *A novel pricing method for European options based on Fourier-cosine series expansions,* SIAM Journal on Scientific Computing **31**, 826 (2009).

[24] F. Fang and C. W. Oosterlee, *Pricing early-exercise and discrete barrier options by Fourier-cosine series expansions,* Numerische Mathematik **114**, 27 (2009).

[25] K. W. Chau, S. C. P. Yam, and H. Yang, *Fourier-cosine method for ruin probabilities,* Journal of Computational and Applied Mathematics **281**, 94 (2015).

[26] A. Borovykh, A. Pascucci, and C. W. Oosterlee, *Efficient computation of various valuation adjustments under local Lévy models,* SIAM Journal on Financial Mathematics **9**, 251 (2018).

[27] L. Ortiz-Gracia and C. W. Oosterlee, *A highly efficient Shannon wavelet inverse Fourier technique for pricing european options,* SIAM Journal on Scientific Computing **38**, B118 (2016).

[28] S. C. Maree, L. Ortiz-Gracia, and C. W. Oosterlee, *Pricing early-exercise and discrete barrier options by Shannon wavelet expansions,* Numerische Mathematik , 1 (2017).

[29] S. Jain and C. W. Oosterlee, *The Stochastic Grid Bundling Method: efficient pricing of Bermudan options and their greeks,* Applied Mathematics and Computation **269**, 412 (2015).

[30] F. Cong and C. W. Oosterlee, *Pricing Bermudan options under Merton jump-diffusion asset dynamics,* International Journal of Computer Mathematics **92**, 2406 (2015).

[31] P. Glasserman and B. Yu, *Simulation for American options: Regression now or regression later?* in *Monte Carlo and Quasi-Monte Carlo Methods 2002: Proceedings of a Conference held at the National University of Singapore, Republic of Singapore, November 25–28, 2002*, edited by H. Niederreiter (Springer Berlin Heidelberg, Berlin, Heidelberg, 2004) pp. 213–226.

[32] P. Henry-Labordère, *Cutting CVA's complexity,* Risk **25**, 67 (2012).

[33] P. Henry-Labordère, X. Tan, and N. Touzi, *A numerical algorithm for a class of BSDEs via the branching process,* Stochastic Processes and their Applications **124**, 1112 (2014).

[34] B. Bouchard, X. Tan, X. Warin, and Y. Zou, *Numerical approximation of BSDEs using local polynomial drivers and branching processes,* Monte Carlo Methods and Applications (to appear).

2

THE WAVELETS-BASED SWIFT METHOD

In this chapter, we introduce a Fourier expansion based wavelet algorithm in the BSDE context. The Shannon Wavelet Inverse Fourier Technique (SWIFT method) was proposed in [2] for pricing European options and a so-called *quick SWIFT variant* was developed in [3] for pricing American and barrier options. The quick SWIFT method, while also based on Shannon wavelets, has the additional benefit of simplifying the algorithm and the error formula. Moreover, it is much easier to adjust individual approximation values because wavelets form a localized basis. We propose a new approach to solving BSDEs by combining a general θ-method for time-integration, as used in [4] and [5], with the SWIFT method. We also improve on previous work on SWIFT by providing an alternative derivation that takes into account the computational range.

This chapter is organized as follows. In Section 2.1, the class of BSDEs under our consideration along with some notations and standing assumptions will be introduced. Section 2.2 contains the derivation of the SWIFT formula and our numerical algorithm for the BSDEs, while Section 2.3 is related to the error and computational complexity of our algorithm. We further improve our algorithm along the computational boundary in Section 2.4. Various numerical experiments are performed in Section 2.5 and concluding remarks are given in Section 2.6.

2.1. SETTINGS AND ASSUMPTIONS

In this chapter, we focus on the case of one-dimensional BSDEs. Therefore, we have:

- $\mu : [0,T] \times \mathbb{R} \to \mathbb{R}$;
- $\sigma : [0,T] \times \mathbb{R} \to \mathbb{R}$;

This chapter is based on the article 'On the wavelets-based SWIFT method for backward stochastic differential equations', published in IMA Journal of Numerical Analysis [1].

- $f : [0,T] \times \mathbb{R} \times \mathbb{R} \times \mathbb{R}$; and
- $g : \mathbb{R} \to \mathbb{R}$.

Furthermore, we assume that we have a fixed uniform time-step $\Delta t = t_{p+1} - t_p$, $\forall p$ and define $\Delta W_{p+1} := W_{t_{p+1}} - W_{t_p} \sim \mathcal{N}(0, \Delta t)$, a normally distributed process in this chapter.

The discretized forward process X^π is defined by

$$X^\pi_{t_0} := x_0,\ X^\pi_{t_{p+1}} := X^\pi_{t_p} + \mu(t_p, X^\pi_{t_p})\Delta t + \sigma(t_p, X^\pi_{t_p})\Delta W_{p+1}, \quad p = 0, \dots, P-1,$$

which is derived from the classical Euler discretization. Note that we only defined the discretized process at the discrete time points here. While it is possible to extend the definition to $[0,T]$, it is not necessary for our presentation.

Throughout this chapter, in addition to the conditions for μ and σ from Section 1.2.1, we assume the following to be true:

(A1) The function $f(t,x,y,z)$ is continuous with respect to (x,y,z) and all one-sided derivatives exist.

(A2) The function $g(x)$ is continuous in x and all left- and right-side derivatives exist.

When dealing with the discretization scheme with $\theta_1 \neq 1$, we add one more assumption:

(A3) The function f is Lipschitz in (y,z), namely,

$$|f(t,x,y_1,z_1) - f(t,x,y_2,z_2)| \leq \mathscr{C}(|y_1 - y_2| + |z_1 - z_2|);\ x, y_1, y_2, z_1, z_2 \in \mathbb{R}, t \in [0,T],$$

for some constant $\mathscr{C}$.

Under assumptions (A1)-(A3) ((A1)-(A2) if $\theta_1 = 1$), the numerical algorithm for the FBSDE, which will be given in Section 2.2, is well-defined. Although, $D_x g = \nabla g$ in Equation (1.4a) may be undefined at countable many distinctive points, it can just be replaced by a one-sided derivative at these points. The conditions above can also ensure satisfactory performance of our algorithms in general, with more details coming in Section 2.3. However, the above conditions are not sufficient to assure the existence of the pair of adapted processes (Y,Z), which is the foundation of any numerical algorithm. We introduce an extra assumption to ensure the existence and uniqueness of the solution (Y,Z) to Equation (1.1).

(A4) There exists a constant $\mathscr{C}$ such that

$$|f(t,x,y,z)| + |g(x)| \leq \mathscr{C}(1 + |x|^k + |y| + |z|),\ \forall x, y, z \in \mathbb{R}, t \in [0,T], k \geq \frac{1}{2}.$$

For further results on the existence and uniqueness of the solution of BSDEs, readers are referred to [6] and further research extending this result. The last point we would like to raise is that the convergence rate of the discretized process to the original process also depends on the functions μ, σ, f and g. We shall discuss these requirements in Section 2.3.1; these conditions are not included in the standing assumptions.

2.2. SWIFT METHOD

For the computation of the expectations appearing in the discrete FBSDEs (1.4), we will use the wavelet-based SWIFT method. In this section, we first provide an alternative derivation for the SWIFT formula used in [2] and [3]. Instead of using an approximation space based on Shannon wavelets on the whole real line, we construct a Shannon wavelet scaling function on a *finite domain* and derive our formula with this scaling function. This approach is beneficial since a truncation of the integration range is required when calculating the wavelet coefficients for the SWIFT formula. Incorporating the truncated range in the scaling function simplifies the formula for the approximation error. Next, we apply the SWIFT method to compute the conditional expectations in the discrete-time approximation of the FBSDE in Equation (1.4) and produce an algorithm for solving FBSDEs recursively, backwards in time. In Sections 2.2.1 and 2.2.2, we derive the SWIFT formula with the finite range approach and compute the relevant expectations for the FBSDE algorithm. Section 2.2.3 and 2.2.4 discuss the approximations of the functions $z_p^\pi(x)$ and $y_p^\pi(x)$.

2.2.1. SCALING FUNCTIONS

We begin our discussion with some preliminary definitions and results. For any fixed real number ϑ and integer $N \neq 0$, we define an inner product and a norm:

$$< v, w >:= \frac{2^\vartheta}{N} \int_{-2^{-\vartheta}N}^{2^{-\vartheta}N} v(x)\, w(x)\, dx, \quad ||v||_2 := \sqrt{< v, v >}.$$

A function v is said to be in the $L^2((-2^{-\vartheta}N, 2^{-\vartheta}N])$ space if $||v||_2$ is a finite number. It can be shown that the set

$$\Gamma_{\vartheta,N} := \left\{ \cos\left(\left(\frac{2n-1}{2N}\pi\right)2^\vartheta x\right), \sin\left(\left(\frac{2n-1}{2N}\pi\right)2^\vartheta x\right) \middle| \, n = 1, 2, \ldots \right\},$$

is orthonormal with respect to this inner product and is dense in $L^2((-2^{-\vartheta}N, 2^{-\vartheta}N])$.

Equipped with the above definitions, we construct an approximation space together with a localized basis, which are the foundations of the truncated SWIFT approximation method. Consider $2N$ distinctive functions $\varphi_{N,r} : \mathbb{R} \to \mathbb{R}$,

$$\begin{aligned}
\varphi_{N,r}(x) &:= \sum_{k=1}^{N} \left(\cos\left(\left(\frac{2k-1}{2N}\pi\right)2^\vartheta x\right) \cos\left(\left(\frac{2k-1}{2N}\pi\right)2^\vartheta \left(\frac{r}{2^\vartheta}\right)\right) \right. \\
&\qquad \left. + \sin\left(\left(\frac{2k-1}{2N}\pi\right)2^\vartheta x\right) \sin\left(\left(\frac{2k-1}{2N}\pi\right)2^\vartheta \left(\frac{r}{2^\vartheta}\right)\right)\right) \\
&= \sum_{k=1}^{N} \cos\left(\frac{2k-1}{2N}\pi\left(2^\vartheta x - r\right)\right) \qquad (2.1) \\
&= \begin{cases} N & \text{if } x = \frac{2N}{2^\vartheta} l + \frac{r}{2^\vartheta} \text{ for } l \text{ an even integer,} \\ -N & \text{if } x = \frac{2N}{2^\vartheta} l + \frac{r}{2^\vartheta} \text{ for } l \text{ an odd integer,} \\ \frac{\sin(\pi(2^\vartheta x - r))}{2\sin\left(\frac{\pi}{2N}(2^\vartheta x - r)\right)} & \text{otherwise,} \end{cases}
\end{aligned}$$

where r belongs to the integer set $\{1-N, 2-N, \ldots, N\}$. This definition is a special case of the scaling functions given in Equation (2.13) of [7], in which the authors presented a uniform approach for the construction of wavelets based on orthogonal polynomials. The properties of $\varphi_{N,r}$ have been extensively studied in [7] and those that are relevant to our numerical method are listed in the next theorem along with their proof.

Theorem 2.1. *The scaling function $\varphi_{N,r}$, which is defined in Equation (2.1), satisfies the following properties:*

(a) The inner product between two scaling functions is given by the following equation:

$$< \varphi_{N,r}, \varphi_{N,s} >= \varphi_{N,r}\left(\frac{s}{2^{\vartheta}}\right), \qquad r,s \in \{1-N, 2-N, \ldots, N\}.$$

Thus, $\{\varphi_{N,r} | r = 1-N, 2-N, \ldots, N\}$ form an orthogonal set.

(b) The scaling function $\varphi_{N,r}$ is localized around $\frac{r}{2^{\vartheta}}$. By this we mean that for the subspace

$$V_N := span\left\{\cos\left(\frac{(2k-1)\pi}{2N}2^{\vartheta}x\right), \sin\left(\frac{(2k-1)\pi}{2N}2^{\vartheta}x\right) \middle| k = 1,2,\ldots,N\right\},$$

we have

$$\left|\left|\frac{\varphi_{N,r}}{\varphi_{N,r}(2^{-\vartheta}r)}\right|\right|_2 = \min\left\{||\chi||_2 : \chi \in V_N, \chi\left(\frac{r}{2^{\vartheta}}\right) = 1\right\}.$$

(c) $\{\varphi_{N,r} | r = 1-N, 2-N, \ldots, N\}$ is a basis for V_N.

(d) The scaling function $\varphi_{N,r}$ is also a kernel polynomial in the sense that for any function v in V_j, we have

$$< v, \varphi_{N,r} >= v\left(\frac{r}{2^{\vartheta}}\right).$$

Proof. We can demonstrate (a) by direct computation and applying the orthonormality of the set $\Gamma_{\vartheta,N}$, such that

$$\begin{aligned}
< \varphi_{N,r}, \varphi_{N,s} >= & \sum_{k=1}^{N}\left(\cos\left(\left(\frac{2k-1}{2N}\pi\right)2^{\vartheta}\left(\frac{r}{2^{\vartheta}}\right)\right)\cos\left(\left(\frac{2k-1}{2N}\pi\right)2^{\vartheta}\left(\frac{s}{2^{\vartheta}}\right)\right)\right. \\
& \left. + \sin\left(\left(\frac{2k-1}{2N}\pi\right)2^{\vartheta}\left(\frac{r}{2^{\vartheta}}\right)\right)\sin\left(\left(\frac{2k-1}{2N}\pi\right)2^{\vartheta}\left(\frac{s}{2^{\vartheta}}\right)\right)\right) \\
= & \varphi_{N,r}\left(\frac{s}{2^{\vartheta}}\right) \\
= & \begin{cases} N, & \text{if } s = r, \\ \frac{\sin((s-r)\pi)}{2\sin\left(\frac{(s-r)\pi}{2N}\right)} = 0, & \text{otherwise.} \end{cases}
\end{aligned}$$

Next, let

$$\chi(x) = \sum_{k=1}^{N}\left(c_k\cos\left(\frac{(2k-1)\pi}{2N}2^{\vartheta}x\right) + d_k\sin\left(\frac{(2k-1)\pi}{2N}2^{\vartheta}x\right)\right),$$

and $\chi\left(\frac{r}{2^{\vartheta}}\right) = 1$ for some constants c_k and d_k. By a simple application of the Cauchy-Schwarz inequality, we get

$$\begin{aligned}1 = \chi^2\left(\frac{r}{2^{\vartheta}}\right) &= \left(\sum_{k=1}^{N}\left(c_k\cos\left(\frac{(2k-1)\pi}{2N}r\right) + d_k\sin\left(\frac{(2k-1)\pi}{2N}r\right)\right)\right)^2 \\ &\leq \left(\sum_{k=1}^{N}(c_k^2+d_k^2)\right)\left(\sum_{k=1}^{N}\left(\left(\cos\left(\frac{(2k-1)\pi}{2N}r\right)\right)^2 + \left(\sin\left(\frac{(2k-1)\pi}{2N}r\right)\right)^2\right)\right) \\ &= N||\chi||_2^2.\end{aligned}$$

The last equality follows from the orthonormality of set $\Gamma_{\vartheta,N}$ and since c_k and d_k are arbitrary, $||\chi||_2^2 \geq \frac{1}{N}$ for any $\chi \in V_N$, such that $\chi\left(\frac{r}{2^{\vartheta}}\right) = 1$. On the other hand, as

$$\varphi_{N,r}\left(\frac{r}{2^{\vartheta}}\right) = \sum_{k=1}^{N}\left(\left(\cos\left(\frac{(2k-1)\pi}{2N}2^{\vartheta}\frac{r}{2^{\vartheta}}\right)\right)^2 + \left(\sin\left(\frac{(2k-1)\pi}{2N}2^{\vartheta}\frac{r}{2^{\vartheta}}\right)\right)^2\right) = N,$$

We know that

$$\left|\left|\frac{\varphi_{N,r}}{\varphi_{N,r}(2^{-\vartheta}r)}\right|\right|_2^2 = \frac{<\varphi_{N,r},\varphi_{N,r}>}{N^2} = \frac{1}{N},$$

which concludes the proof of (b). Statement (c) is true since $\{\varphi_{N,r}|r = 1-N, 2-N, \ldots, N\}$ has the same number of elements as the spanning set of V_N; its elements are orthogonal and therefore independent of each other.

Part (d) follows from parts (a) and (c). For any $v \in V_N$, $v(\cdot) = \sum_{s=1-N}^{N}\mathcal{V}_s\varphi_{N,s}(\cdot)$ by (c), and from part (a), we have

$$<v,\varphi_{N,r}> = \sum_{s=1-N}^{N}\mathcal{V}_s<\varphi_{N,s},\varphi_{N,r}> = \sum_{s=1-N}^{N}\mathcal{V}_s\varphi_{N,s}\left(\frac{r}{2^m}\right) = v\left(\frac{r}{2^m}\right).$$

□

The space V_N and the scaling functions $\{\varphi_{N,r}\}_{r=1-N,\ldots,N}$ are our approximation space and our basis functions, respectively. As a result, for any function v in the $L^2((-2^{-\vartheta}N, 2^{-\vartheta}N])$ space, its projection on V_N, denoted as $H_{V_N}v$, can be written in the form

$$\frac{1}{N}\sum_{r=1-N}^{N}<H_{V_N}v,\varphi_{N,r}>\varphi_{N,r} = \frac{1}{N}\sum_{r=1-N}^{N}<v,\varphi_{N,r}>\varphi_{N,r}.$$

2.2.2. QUICK SWIFT FORMULA AND COEFFICIENTS

Assume we wish to approximate a finite integral $\int_{\mathbb{R}} v(\varsigma)q(\varsigma)d\varsigma$, where v is within $L^2((-2^{-\vartheta}N, 2^{-\vartheta}N])$ and we have $\int_{\mathbb{R}} q(\varsigma)d\varsigma < \infty$. We shall approach this problem by replacing v by $H_{V_N}v$. This gives us the following approximation:

$$\begin{aligned}&\int_{\mathbb{R}} v(\varsigma)q(\varsigma)d\varsigma \\ \approx &\int_{\mathbb{R}} q(\varsigma)\frac{1}{N}\sum_{r=1-N}^{N}<v,\varphi_{N,r}>\varphi_{N,r}(\varsigma)d\varsigma\end{aligned}$$

$$= \int_{\mathbb{R}} q(\varsigma) \frac{1}{N} \sum_{r=1-N}^{N} \frac{2^{\vartheta}}{N} \int_{-2^{-\vartheta}N}^{2^{-\vartheta}N} v(\varrho) \sum_{k_1=1}^{N} \cos\left(C_{k_1}(2^{\vartheta}\varrho - r)\right) d\varrho \sum_{k_2=1}^{N} \cos\left(C_{k_2}(2^{\vartheta}\varsigma - r)\right) d\varsigma$$

$$= \sum_{r=1-N}^{N} \int_{\mathbb{R}} q(\varsigma) \frac{2^{\frac{\vartheta}{2}}}{N} \sum_{k_2=1}^{N} \cos\left(C_{k_2}(2^{\vartheta}\varsigma - r)\right) d\varsigma \int_{-2^{-\vartheta}N}^{2^{-\vartheta}N} v(\varrho) \frac{2^{\frac{\vartheta}{2}}}{N} \sum_{k_2=1}^{N} \cos\left(C_{k_2}(2^{\vartheta}\varrho - r)\right) d\varrho,$$

which is the SWIFT formula, proposed in [2], with $C_k := \frac{2k-1}{2N}\pi$. In the above derivation, we only listed the dependency to dummy variable ς of the functions v and q. In practice, v and q will depend on other variables, like for example, time. We will put the additional dependency in our notation without further notice in the remainder of this chapter whenever it is necessary for the presentation.

Remark 2.1. While the accuracy of the approximation depends on other properties of the functions v and q and shall be studied in the rest of this chapter, $v \in L^2((-2^{-\vartheta}N, 2^{-\vartheta}N])$ and q being integrable are the only conditions needed for the above approximation to be well-defined.

Remark 2.2. We only define the approximation on the set $(-2^{-\vartheta}N, 2^{-\vartheta}N]$ that centers around zero. For any function v such that $\int_a^b (v(\varsigma))^2 d\varsigma < \infty$ for a finite range $(a,b]$, we need to perform a change of variables $\varsigma' = \varsigma - \frac{a+b}{2}$, for $\varsigma' \in (a - \frac{a+b}{2}, b - \frac{a+b}{2}]$, let $v'(\varsigma') = v(\varsigma' + \frac{a+b}{2}) = v(\varsigma)$. Then, we can pick N and ϑ accordingly and perform the approximation on v'. With a slight abuse of notation, we assume that we perform this tedious step implicitly and apply the SWIFT formula with any finite range $[a,b]$.

In this work, we shall adopt the quick variant of the SWIFT formula proposed in [3]. Instead of replacing v with $H_{V_N} v$, we approximate v by

$$v(x) \approx \frac{1}{N} \sum_{r=1-N}^{N} v\left(\frac{r}{2^{\vartheta}}\right) \varphi_{N,r}(x).$$

The reason behind this is left for the error section, but this gives an approximation for $\mathbb{E}_p^x[v(t_{p+1}, X_{t_{p+1}}^{\pi})]$, which we see in the discrete-time approximation of FBSDE,

$$\begin{aligned} \mathbb{E}_p^x[v(t_{p+1}, X_{t_{p+1}}^{\pi})] \approx & \mathbb{E}_p^x\left[\frac{1}{N} \sum_{r=1-N}^{N} v\left(t_{p+1}, \frac{r}{2^{\vartheta}}\right) \varphi_{N,r}(X_{t_{p+1}}^{\pi})\right] \\ = & \frac{1}{N} \sum_{r=1-N}^{N} v\left(t_{p+1}, \frac{r}{2^{\vartheta}}\right) \mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^{\pi})]. \end{aligned} \tag{2.2}$$

The expectation $\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^{\pi})]$ can be computed by

$$\begin{aligned} \mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^{\pi})] &= \mathbb{E}_p^x\left[\sum_{k=1}^{N} \cos\left(2^{\vartheta} C_k X_{t_{p+1}}^{\pi} - C_k r\right)\right] \\ &= \Re\left\{\sum_{k=1}^{N} \mathbb{E}_p^x\left[\exp\left(\imath 2^{\vartheta} C_k (x + \mu(t_p, x)\Delta t + \sigma(t_p, x)\Delta W_{p+1})\right) \exp\left(-\imath C_k r\right)\right]\right\} \\ &= \Re\left\{\sum_{k=1}^{N} \exp\left(\imath 2^{\vartheta} C_k x\right) \mathscr{F}\left(t_p, x, 2^{\vartheta} C_k\right) \exp\left(-\imath C_k r\right)\right\}, \end{aligned} \tag{2.3}$$

where the real part of a complex number is denoted by $\Re\{\}$ and $\mathscr{F}$ is the characteristic function of $X^{\pi}_{t_{p+1}} - X^{\pi}_{t_p}$, i.e.

$$\mathscr{F}(\tau,\varsigma,\varrho) := \exp\left(\imath\varrho\mu(\tau,\varsigma)\Delta t - \frac{1}{2}\varrho^2\sigma^2(\tau,\varsigma)\Delta t\right).$$

In this thesis, the notation $\mathscr{F}$ will be used exclusively to denote characteristic functions of probability measures/Fourier transforms of measures.

The authors of [3] demonstrated how to calculate the vector $(\mathbb{E}^x_p[\varphi_{N,r}(X^{\pi}_{t_{p+1}})])_{r=(1-N,\dots,N)}$ with a Fast Fourier Transform (FFT) algorithm. The computations induced by Equation (2.3) in our algorithm have the computational complexity of $O(N\log(N))$.

Expectations in the form $\mathbb{E}^x_p[v(X^{\pi}_{t_{p+1}})\Delta W_{p+1}]$ also occur in the discrete-time approximation of the FBSDE and they can be computed by:

$$\begin{aligned}\mathbb{E}^x_p[v(t_{p+1},X^{\pi}_{t_{p+1}})\Delta W_{p+1}] \approx& \mathbb{E}^x_p\left[\frac{1}{N}\sum_{r=1-N}^{N} v\left(t_{p+1},\frac{r}{2^{\vartheta}}\right)\varphi_{N,r}(X^{\pi}_{t_{p+1}})\Delta W_{p+1}\right]\\ =&\frac{1}{N}\sum_{r=1-N}^{N} v\left(t_{p+1},\frac{r}{2^{\vartheta}}\right)\mathbb{E}^x_p[\varphi_{N,r}(X^{\pi}_{t_{p+1}})\Delta W_{p+1}],\end{aligned}$$

with $\mathbb{E}^x_p[\varphi_{N,r}(X^{\pi}_{t_{p+1}})\Delta W_{p+1}]$ given by the formula:

$$\begin{aligned}&\mathbb{E}^x_p[\varphi_{N,r}(X^{\pi}_{t_{p+1}})\Delta W_{p+1}] = \sigma(t_p,x)\Delta t\mathbb{E}^x_p[D_x\varphi_{N,r}(X^{\pi}_{t_{p+1}})]\\ =&\Re\left\{\imath\sigma(t_p,x)\Delta t\sum_{k=1}^{N}2^{\vartheta}C_k\mathbb{E}^x_p\left[\exp\left(\imath 2^{\vartheta}C_k(x+\mu(t_p,x)\Delta t+\sigma(t_p,x)\Delta W_{p+1})\right)\exp(-\imath C_k r)\right]\right\}\\ =&\Re\left\{\imath\sigma(t_p,x)\Delta t\sum_{k=1}^{N}2^{\vartheta}C_k\exp\left(\imath 2^{\vartheta}C_k x\right)\mathscr{F}\left(t_p,x,2^{\vartheta}C_k\right)\exp(-\imath C_k r)\right\},\end{aligned}\tag{2.4}$$

where the first equality sign follows from an integration by parts argument and we also note that $D_x\varphi_{N,r}(x) = -\sum_{k=1}^{N}2^{\vartheta}C_k\sin\left(2^{\vartheta}C_k x - C_k r\right)$. Once again, we can use the FFT to compute these expectations.

In the next two sections, we will combine the quick variant of the SWIFT formula and the discrete-time approximation of the FBSDE in Equation (1.4).

2.2.3. QUICK SWIFT APPROXIMATION OF FUNCTION $z^{\pi}_p(x)$

There are three different expectations,

$$\mathbb{E}^x_p[Z^{\pi}_{t_{p+1}}], \mathbb{E}^x_p[Y^{\pi}_{p+1}\Delta W_{t_{p+1}}], \text{ and } \mathbb{E}^x_p[f(t_{p+1},\mathbf{X}^{\pi}_{t_{p+1}})\Delta W_{p+1}],$$

that need to be approximated in Equation (1.4b). Applying the quick SWIFT formula, we have:

$$\mathbb{E}^x_p[Z^{\pi}_{t_{p+1}}] \approx \frac{1}{N}\sum_{r=1-N}^{N} z^{\pi}_{p+1}\left(\frac{r}{2^{\vartheta}}\right)\Re\left\{\sum_{k=1}^{N} e^{\imath 2^{\vartheta}C_k x}\mathscr{F}\left(t_p,x,2^{\vartheta}C_k\right)e^{-\imath C_k r}\right\};$$

$$\mathbb{E}^x_p[Y^{\pi}_{t_{p+1}}\Delta W_{p+1}] \approx \frac{1}{N}\sum_{r=1-N}^{N} y^{\pi}_{p+1}\left(\frac{r}{2^{\vartheta}}\right)\times$$

$$\Re\left\{\iota\sigma(t_p,x)\Delta t\sum_{k=1}^{N}2^{\vartheta}C_k e^{\iota 2^{\vartheta}C_k x}\mathscr{F}\left(t_p,x,2^{\vartheta}C_k\right)e^{-\iota C_k r}\right\};$$

$$\mathbb{E}_p^x[f(t_{p+1},\mathbf{X}_{t_{p+1}}^{\pi})\Delta W_{p+1}]\approx\frac{1}{N}\sum_{r=1-N}^{N}f\left(t_{p+1},\frac{r}{2^{\vartheta}},y_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right),z_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right)\right)\times$$

$$\Re\left\{\iota\sigma(t_p,x)\Delta t\sum_{k=1}^{N}2^{\vartheta}C_k e^{\iota 2^{\vartheta}C_k x}\mathscr{F}\left(t_p,x,2^{\vartheta}C_k\right)e^{-\iota C_k r}\right\}.$$

We denote the approximation of the FBSDE by a SWIFT-type formula combined with the Euler discretization at point (t_p,x) by $(\hat{y}_p^{\pi}(x),\hat{z}_p^{\pi}(x))$, then $\hat{z}_p^{\pi}$ satisfies the following recursive relation:

$$\begin{aligned}\hat{z}_p^{\pi}(x)=&\Re\left\{\frac{1}{N}\sum_{k=1}^{N}e^{\iota 2^{\vartheta}C_k x}\left[\mathscr{F}\left(t_p,x,2^{\vartheta}C_k\right)\sum_{r=1-N}^{N}\left(-\frac{1-\theta_2}{\theta_2}\hat{z}_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right)e^{-\iota C_k r}\right)\right]\right\}\\&+\Re\left\{\frac{\iota}{N}\sigma(t_p,x)\sum_{k=1}^{N}e^{\iota 2^{\vartheta}C_k x}2^{\vartheta}C_k\mathscr{F}\left(t_p,x,2^{\vartheta}C_k\right)\times\right.\\&\left.\left[\sum_{r=1-N}^{N}\left(\frac{1}{\theta_2}\hat{y}_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right)+\frac{(1-\theta_2)\Delta t}{\theta_2}f\left(t_{k+1},\frac{r}{2^{\vartheta}},\hat{y}_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right),\hat{z}_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right)\right)e^{-\iota C_k r}\right)\right]\right\},\end{aligned}\tag{2.5}$$

for $p=0,1,\ldots,P-1$.

2.2.4. QUICK SWIFT APPROXIMATION OF FUNCTION $y_p^{\pi}(x)$

Equation (1.4c) for $Y_{t_p}^{\pi}$ contains an explicit and an implicit part if $\theta_1>0$. The explicit part is denoted by:

$$h(t_p,x):=\mathbb{E}_p^x[Y_{t_{p+1}}^{\pi}]+\Delta t(1-\theta_1)\mathbb{E}_p^x[f(t_{p+1},\mathbf{X}_{t_{p+1}}^{\pi})].\tag{2.6}$$

The function h is a linear combination of two expectations, $\mathbb{E}_p^x[Y_{t_{p+1}}^{\pi}]$ and $\mathbb{E}_p^x[f(t_{p+1},\mathbf{X}_{t_{p+1}}^{\pi})]$, and they can be approximated by the following quick SWIFT formulas:

$$\mathbb{E}_p^x[Y_{t_{p+1}}^{\pi}]\approx\frac{1}{N}\sum_{r=1-N}^{N}y_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right)\Re\left\{\sum_{k=1}^{N}e^{\iota 2^{\vartheta}C_k x}\mathscr{F}\left(t_p,x,2^{\vartheta}C_k\right)e^{-\iota C_k r}\right\};\tag{2.7a}$$

$$\begin{aligned}\mathbb{E}_p^x[f(t_{p+1},\mathbf{X}_{t_{p+1}}^{\pi})]\approx&\frac{1}{N}\sum_{r=1-N}^{N}f\left(t_{p+1},\frac{r}{2^{\vartheta}},y_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right),z_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right)\right)\times\\&\Re\left\{\sum_{k=1}^{N}e^{\iota 2^{m}C_k x}\mathscr{F}\left(t_p,x,2^{\vartheta}C_k\right)e^{-\iota C_k r}\right\}.\end{aligned}\tag{2.7b}$$

Therefore, we have an approximation for h:

$$\begin{aligned}\hat{h}(t_p,x):=&\mathbb{E}_p^x[\hat{y}_{p+1}^{\pi}(X_{t_{p+1}}^{\pi})]+\Delta t(1-\theta_1)\mathbb{E}_p^x[f(t_{p+1},X_{t_{p+1}}^{\pi},\hat{y}_{p+1}^{\pi}(X_{t_{p+1}}^{\pi}),\hat{z}_{p+1}^{\pi}(X_{t_{p+1}}^{\pi}))]\\=&\Re\left\{\frac{1}{N}\sum_{k=1}^{N}e^{\iota 2^{\vartheta}C_k x}\mathscr{F}\left(t_p,x,2^{\vartheta}C_k\right)\cdot\right.\end{aligned}$$

$$\sum_{r=1-N}^{N}\left[\hat{y}^{\pi}_{p+1}\left(\frac{r}{2^{\vartheta}}\right)+\Delta t(1-\theta_1)f\left(t_{p+1},\frac{r}{2^{\vartheta}},\hat{y}^{\pi}_{p+1}\left(\frac{r}{2^{\vartheta}}\right),\hat{z}^{\pi}_{p+1}\left(\frac{r}{2^{\vartheta}}\right)\right)\right]e^{-\iota C_k r}\Bigg\}, \tag{2.8}$$

and the function $\hat{y}^{\pi}_{p}$ is defined implicitly by:

$$\hat{y}^{\pi}_{p}(x)=\Delta t\theta_1 f(t_p,x,\hat{y}^{\pi}_{p}(x),\hat{z}^{\pi}_{p}(x))+\hat{h}(t_p,x). \tag{2.9}$$

Whenever $\theta_1\neq 0$, *Picard iterations* are performed I times to recover $\hat{y}^{\pi}_{p}(x)$, which is the same procedure used in both [8] and [5]. The initial guess for the iterations is defined as the approximation of $\mathbb{E}^{x}_{p}[Y^{\pi}_{t_{p+1}}]$ as in Equation (2.7a). The conditions for convergent iterations and an extra error induced will be discussed in Section 2.3.3.

The overall algorithm to generate an approximation $(\hat{y}^{\pi}_{0}(x),\hat{z}^{\pi}_{0}(x))$ for (Y_0,Z_0) has been summarized in Algorithm 1.

For $r=1-N$ to N **do**,
  $\hat{y}^{\pi}_{P}(2^{-\vartheta}r)=g(2^{-\vartheta}r)$, $\hat{z}^{\pi}_{P}(2^{-\vartheta}r)=\sigma D_x g(2^{-\vartheta}r)$ and
  $\hat{f}^{\pi}_{P}(2^{-\vartheta}r)=f(T,2^{-\vartheta}r,\hat{y}^{\pi}_{P}(2^{-\vartheta}r),\hat{z}^{\pi}_{P}(2^{-\vartheta}r))$.
Compute $(\mathbb{E}^{2^{-\vartheta}r}_{P-1}[\varphi_{N,k}(X^{\pi}_{t_P})])_{r,k=1-N,\dots,N}$ and $(\mathbb{E}^{2^{-\vartheta}r}_{P-1}[\varphi_{N,k}(X^{\pi}_{t_P})\Delta W_P])_{r,k=1-J,\dots,J}$ with (2.3) and (2.4).
For $p=P-1$ to 1 **do**,
  Compute the function $(\hat{z}^{\pi}_{p}(2^{-\vartheta}r))_{r=1-N,\dots,N}$ with (2.5).
  Compute the function $(\hat{y}^{\pi}_{p}(2^{-\vartheta}r))_{r=1-N,\dots,N}$ with (2.9) and Picard iterations if necessary.
  Compute the functions $(f(t_p,2^{-\vartheta}r,\hat{y}^{\pi}_{p}(2^{-\vartheta}r),\hat{z}^{\pi}_{p}(2^{-\vartheta}r)))_{r=1-N,\dots,N}$.
  Compute $(\mathbb{E}^{2^{-\vartheta}r}_{p-1}[\varphi_{N,k}(X^{\pi}_{t_p})])_{r,k=1-N,\dots,N}$ and $(\mathbb{E}^{2^{-\vartheta}r}_{p-1}[\varphi_{N,k}(X^{\pi}_{t_p})\Delta W_p])_{r,k=1-N,\dots,N}$ with (2.3) and (2.4) if the distribution of $(X^{\pi}_{t_p}-X^{\pi}_{t_{p-1}})$ is time-dependent.
Compute $\hat{z}^{\pi}_{0}(x_0)$ and $\hat{y}^{\pi}_{0}(x_0)$ at time zero.

Algorithm 1: Quick SWIFT method.

2.3. ERRORS AND COMPUTATIONAL COMPLEXITY

In this section, we shall discuss the major components of the error when solving an FBSDE with a SWIFT-type method. They are the discretization error of the FBSDE, the error of approximation with the SWIFT formula and the error introduced by the Picard iteration. We will also discuss the computational complexity of the SWIFT method. For notational simplicity, we shall use $\mathscr{C}$ to denote a generic constant whose value and dependency may change from line to line.

2.3.1. DISCRETIZATION ERROR OF THE FBSDE

The error due to the discrete-time approximation of the stochastic process depends on the parameters θ_1 and θ_2, the drift μ, the volatility σ, the driver function f and the terminal condition g. It is difficult to provide a universal result for all FBSDEs for which our

method can be applied. However, under some specific assumptions, we can derive an error bound for the global error due to time discretization. Adopting the following error notation:

$$\varepsilon_p^y(X_p) := y_p(X_{t_p}) - y_p^\pi(X_{t_p}^\pi),\ \varepsilon_p^z := z_p(X_{t_p}) - z_p^\pi(X_{t_p}^\pi),\ \varepsilon_p^f(X_{t_p}) := f(t_p, \mathbf{X}_{t_p}) - f(t_p, \mathbf{X}_{t_p}^\pi),$$

one of the results for the discretization error is in the following theorem.

Theorem 2.2 ([5],Theorem 1.)**.** *Assume that the forward process has constant coefficients μ and σ and the discretization scheme with $\theta_1 = \theta_2 = \frac{1}{2}$. If*

$$\mathbb{E}_{P-1}^x[|\varepsilon_P^z|] \sim \mathrm{O}((\Delta t)^3),\ \mathbb{E}_{P-1}^x[|\varepsilon_P^y|] \sim \mathrm{O}((\Delta t)^3),$$

then

$$\mathbb{E}_0^x\left[|\varepsilon_p^y| + \sqrt{\Delta t}|\varepsilon_p^z|\right] \le \mathscr{C}(\Delta t)^2,\ \textit{for } 1 \le p \le P,$$

where the constant $\mathscr{C}$ depends on numbers T, μ and σ and functions g and f.

For general drift and diffusion coefficients, we may only have *first-order weak convergence* for the Euler discretization of forward process X and it may become the dominating error of our algorithm. For the proof of Theorem 2.2 and other convergence results, readers are referred to [5] and the references therein.

2.3.2. ERROR IN SWIFT FORMULAS

In this subsection, we shall discuss the error of the numerical approximation by the SWIFT-type formulas. In order for our formula to be applicable for both the one-step approximation and the recursive case, we adopt the following setting.

Consider an expectation $\mathbb{E}[v(X_{t+\Delta t}^\pi)|X_t^\pi = x]$ for a function v defined on $\mathbb{R}$ and assume that v is continuous with all its left- and right-side derivatives well-defined, we define our expectation approximation as

$$\hat{\mathbb{E}}[v(X_{t+\Delta t}^\pi)|X_t^\pi = x] := \frac{1}{N}\sum_{r=1-N}^{N} \rho_v\left(\frac{r}{2^\vartheta}\right)\mathbb{E}[\varphi_{N,r}(X_{t+\Delta t}^\pi)|X_t^\pi = x], \tag{2.10}$$

where $\{\rho_v(2^{-\vartheta}r)\}_{r=1-N,\ldots,N}$ is an approximation vector related to function v, to be defined. For any function $v : \mathbb{R} \to \mathbb{R}$ and given range $(-2^{-\vartheta}N, 2^{-\vartheta}N]$, we associate v with *an alternating extension* defined below.

Definition 2.3. An *alternating extension of a function v*, denoted by $\tilde{v}$, is a function defined on $\mathbb{R}$ such that it satisfies:

(a) $\tilde{v}(x) = v(x) \quad \forall x \in (-2^{-\vartheta}N, 2^{-\vartheta}N]$;

(b) $\tilde{v}(x + 2^{1-\vartheta}N) = -\tilde{v}(x) \quad \forall x \in \mathbb{R}$.

The difference between the approximation value and the true value is given by

$$\begin{aligned}&\mathbb{E}[v(X_{t+\Delta t}^\pi)|X_t^\pi = x] - \hat{\mathbb{E}}[v(X_{t+\Delta t}^\pi)|X_t^\pi = x]\\ =&\mathbb{E}[v(X_{t+\Delta t}^\pi)|X_t^\pi = x] - \mathbb{E}[\tilde{v}(X_{t+\Delta t}^\pi)|X_t^\pi = x]\end{aligned}$$

$$
\begin{aligned}
&+\mathbb{E}[\tilde{v}(X^{\pi}_{t+\Delta t})|X^{\pi}_t=x]-\mathbb{E}[H_{V_N}v(X^{\pi}_{t+\Delta t})|X^{\pi}_t=x]\\
&+\mathbb{E}\left[\frac{1}{N}\sum_{r=1-N}^{N}<H_{V_N}v,\varphi_{N,r}>\varphi_{N,r}(X^{\pi}_{t+\Delta t})\middle|X^{\pi}_t=x\right]\\
&-\frac{1}{N}\sum_{r=1-N}^{N}\rho_v\left(\frac{r}{2^{\vartheta}}\right)\mathbb{E}[\varphi_{N,r}(X^{\pi}_{t+\Delta t})|X^{\pi}_t=x]\\
=&\mathbb{E}[v(X^{\pi}_{t+\Delta t})\mathbf{1}_{\{X^{\pi}_{t+\Delta t}\notin(-2^{-\vartheta}N,2^{-\vartheta}N]\}}|X^{\pi}_t=x]-\mathbb{E}[\tilde{v}(X^{\pi}_{t+\Delta t})\mathbf{1}_{\{X^{\pi}_{t+\Delta t}\notin(-2^{-\vartheta}N,2^{-\vartheta}N]\}}|X^{\pi}_t=x]\\
&+\mathbb{E}[\tilde{v}(X^{\pi}_{t+\Delta t})-H_{V_N}v(X^{\pi}_{t+\Delta t})|X^{\pi}_t=x]\\
&+\frac{1}{N}\sum_{r=1-N}^{N}\left(H_{V_N}v\left(\frac{r}{2^{\vartheta}}\right)-\rho_v\left(\frac{r}{2^{\vartheta}}\right)\right)\mathbb{E}[\varphi_{N,r}(X^{\pi}_{t+\Delta t})|X^{\pi}_t=x] \qquad (2.11)\\
=&\mathbb{E}[v(X^{\pi}_{t+\Delta t})\mathbf{1}_{\{X^{\pi}_{t+\Delta t}\notin(-2^{-\vartheta}N,2^{-\vartheta}N]\}}|X^{\pi}_t=x]-\mathbb{E}[\tilde{v}(X^{\pi}_{t+\Delta t})\mathbf{1}_{\{X^{\pi}_{t+\Delta t}\notin(-2^{-\vartheta}N,2^{-\vartheta}N]\}}|X^{\pi}_t=x]\\
&+\mathbb{E}[\tilde{v}(X^{\pi}_{t+\Delta t})-H_{V_N}v(X^{\pi}_{t+\Delta t})|X^{\pi}_t=x]\\
&+\frac{1}{N}\sum_{r=1-N}^{N}\left(H_{V_N}v\left(\frac{r}{2^{\vartheta}}\right)-v\left(\frac{r}{2^{\vartheta}}\right)\right)\mathbb{E}[\varphi_{N,r}(X^{\pi}_{t+\Delta t})|X^{\pi}_t=x]\\
&+\frac{1}{N}\sum_{r=1-N}^{N}\left(v\left(\frac{r}{2^{\vartheta}}\right)-\rho_v\left(\frac{r}{2^{\vartheta}}\right)\right)\mathbb{E}[\varphi_{N,r}(X^{\pi}_{t+\Delta t})|X^{\pi}_t=x]. \qquad (2.12)
\end{aligned}
$$

We derive the above formula by telescoping and using the properties of the scaling function. By taking absolute values on both sides of Equation (2.12), we get a simple bound for the approximation error:

$$
\begin{aligned}
&\left|\mathbb{E}[v(X^{\pi}_{t+\Delta t})|X^{\pi}_t=x]-\hat{\mathbb{E}}[v(X^{\pi}_{t+\Delta t})|X^{\pi}_t=x]\right|\\
\leq&|\mathbb{E}[v(X^{\pi}_{t+\Delta t})\mathbf{1}_{\{X^{\pi}_{t+\Delta t}\notin(-2^{-\vartheta}N,2^{-\vartheta}N]\}}|X^{\pi}_t=x]|+\mathbb{E}[|\tilde{v}(X^{\pi}_{t+\Delta t})|\mathbf{1}_{\{X^{\pi}_{t+\Delta t}\notin(-2^{-\vartheta}N,2^{-\vartheta}N]\}}|X^{\pi}_t=x]\\
&+\mathbb{E}[|\tilde{v}(X^{\pi}_{t+\Delta t})-H_{V_N}v(X^{\pi}_{t+\Delta t})||X^{\pi}_t=x]\\
&+\frac{1}{N}\sum_{r=1-N}^{N}\left|H_{V_N}v\left(\frac{r}{2^{\vartheta}}\right)-\tilde{v}\left(\frac{r}{2^{\vartheta}}\right)\right||\mathbb{E}[\varphi_{N,r}(X^{\pi}_{t+\Delta t})|X^{\pi}_t=x]|\\
&+\frac{1}{N}\sum_{r=1-N}^{N}\left|v\left(\frac{r}{2^{\vartheta}}\right)-\rho_v\left(\frac{r}{2^{\vartheta}}\right)\right||\mathbb{E}[\varphi_{N,r}(X^{\pi}_{t+\Delta t})|X^{\pi}_t=x]|.
\end{aligned}
$$

Errors of a SWIFT-type method can be separated into four (in Equation (2.11)) or five (in Equation (2.12)) parts and they will be discussed one by one. Note that the first three terms in Equation (2.11) and (2.12) are the same.

The first error term is related to the tail behaviour of the probability measure and function v. It is finite, as otherwise the original expectation would be infinite. Also, its value should decrease (heuristically speaking, not in strict mathematical sense) when a wider computational domain is used. Assuming v to be uniformly bounded by a number $\mathscr{C}$, this term is bounded by $\mathscr{C}\cdot\mathbb{P}(X^{\pi}_{t+\Delta t}\notin(-2^{-\vartheta}N,2^{-\vartheta}N]|X^{\pi}_t=x)$. Similarly, the second term is related to the tail probability. Because of the continuity of v and the periodicity of $\tilde{v}$, $\tilde{v}$ is uniformly bounded by some number $\mathscr{C}'$ and the second error term is bounded by $\mathscr{C}'\cdot\mathbb{P}(X^{\pi}_{t+\Delta t}\notin(-2^{-\vartheta}N,2^{-\vartheta}N]|X^{\pi}_t=x)$.

The third part is related to the projection error on V_N. From the assumption that v is continuous with all left- and right-side derivatives of v existing, $\tilde{v}=\lim_{N\to\infty}H_{V_N}v$,

a.e.. This can be shown by adopting the classical Dirichlet's kernel argument, which is included in the appendix. Applying the dominated convergence theorem,

$$\mathbb{E}[\tilde{v}(X^{\pi}_{t+\Delta t}) - H_{V_N} v(X^{\pi}_{t+\Delta t}) | X^{\pi}_t = x] = \sum_{k=N+1}^{\infty} < v, \cos(2^{\vartheta} C_k \cdot) > \mathbb{E}[\cos(2^{\vartheta} C_k X^{\pi}_{t+\Delta t}) | X^{\pi}_t = x] + \sum_{k=N+1}^{\infty} < v, \sin(2^{\vartheta} C_k \cdot) > \mathbb{E}[\sin(2^{\vartheta} C_k X^{\pi}_{t+\Delta t}) | X^{\pi}_t = x].$$

Note that in this part, we only require $\tilde{v} = \lim_{N\to\infty} H_{V_N} v$, a.e., so that we can relax the requirement on v from being continuous to being piecewise continuous. Using an integration by parts argument, if the forward process has a smooth density, this error converges exponentially with respect to N but increases with the computational range.

Remark 2.3. In fact, the projection error in the SWIFT formula can be controlled under alternative conditions. Assume that the probability density function $\mathfrak{q}$ of $X^{\pi}_{t+\pi t} | X^{\pi}_t = x$, is in $L^2(\mathbb{R})$, then,

$$\begin{aligned}
&|\mathbb{E}[\tilde{v}(X^{\pi}_{t+\Delta t}) - H_{V_N} v(X^{\pi}_{t+\Delta t}) | X^{\pi}_t = x] - \mathbb{E}[\tilde{v}(X^{\pi}_{t+\Delta t}) \mathbf{1}_{\{X^{\pi}_{t+\Delta t} \notin (-2^{-\vartheta} N, 2^{-\vartheta} N]\}} | X^{\pi}_t = x]| \\
=&|\mathbb{E}[\tilde{v}(X^{\pi}_{t+\Delta t}) - H_{V_N} v(X^{\pi}_{t+\Delta t}) \mathbf{1}_{\{X^{\pi}_{t+\Delta t} \in (-2^{-\vartheta} N, 2^{-\vartheta} N]\}} | X^{\pi}_t = x] \\
&-\mathbb{E}[H_{V_N} v(X^{\pi}_{t+\Delta t}) \mathbf{1}_{\{X^{\pi}_{t+\Delta t} \notin (-2^{-\vartheta} N, 2^{-\vartheta} N]\}} | X^{\pi}_t = x]| \\
\leq& ||\tilde{v} - H_{V_N} v||_2 \left(\int_{\mathbb{R}} \mathfrak{q}^2(\varsigma | x) d\varsigma \right)^{\frac{1}{2}} + \mathscr{C} \cdot \mathbb{P}(X^{\pi}_{t+\Delta t} \notin (-2^{-\vartheta} N, 2^{\vartheta} N] | X^{\pi}_t = x),
\end{aligned}$$

with $\mathscr{C}$ depending on the function $H_{V_N} v$. While we do not use this alternative proof in this chapter, it implies that the SWIFT formula can be used in a more general setting and is suitable for other applications as well.

In the usual variant of the SWIFT formula, we set $\rho_v(2^{-\vartheta} r) = < v, \varphi_{N,r} >$, so the fourth term of Equation (2.11) is by definition zero and the error of applying the SWIFT formula will only consist of the first three terms. However, the calculation of $< v, \varphi_{N,r} >$ is difficult and time-consuming, if not impossible in practice, especially in the recursive case. Therefore, we propose picking $\rho_v(2^{-\vartheta} r) = \hat{v}(2^{-\vartheta} r)$, *an approximation of the original function* v. While it will introduce an extra error, we shall demonstrate that this error can be controlled and that the computation is much simpler.

For the sum in the fourth term of Equation (2.12), we need to consider two cases. When $r \neq N$, the pointwise convergence of $H_{V_N} v$ guarantees that $\left| H_{V_N} v\left(\frac{r}{2^{\vartheta}}\right) - \tilde{v}\left(\frac{r}{2^{\vartheta}}\right) \right| \to 0$. Therefore, these terms are bounded when N is large enough. When $r = N$, it is likely that $\tilde{v}$ is discontinuous at $2^{-\vartheta} N$ and the above argument does not hold. However, we note that this error term is also a weighted sum of $H_{V_N} v(2^{-\vartheta} r) - v(2^{-\vartheta} r)$, with the weight given by $\frac{1}{N}\mathbb{E}[\varphi_{N,r}(X^{\pi}_{t+\Delta t}) | X^{\pi}_t = x]$. Assume that $\mathbb{P}(X^{\pi}_{t+\Delta t} \notin (\lambda - b, \lambda + b)) < \epsilon_1$ and $\frac{1}{N}\varphi_{N,N}(x) < \epsilon_2$, when $x \in (\lambda - b, \lambda + b)$, for some positive numbers b, ϵ_1 and ϵ_2 and some number λ, then

$$\frac{1}{N}\mathbb{E}[\varphi_{N,N}(X^{\pi}_{t+\Delta t}) | X^{\pi}_t = x] < \mathbb{P}(X^{\pi}_{t+\Delta t} \notin (\lambda - b, \lambda + b)) + \epsilon_2 \mathbb{P}(X^{\pi}_{t+\Delta t} \in (\lambda - b, \lambda + b)) < \epsilon_1 + \epsilon_2.$$

These assumptions are satisfied when the distribution of X^π is centered around a point λ, which is true with a diffusion process whose diffusion coefficient is small, the computational range is sufficiently large so that λ is far away from the boundary, and the wavelet order is sufficiently high. If these conditions are met, the weight for the term $\left|H_{V_N} v\left(\frac{N}{2^\vartheta}\right) - \tilde{v}\left(\frac{N}{2^\vartheta}\right)\right|$ is small and the weighted term can be bounded. By combining the two arguments above, we can bound this error term when the computational range is sufficiently large and the wavelet order is sufficiently high.

In the one-step case, we pick $\rho_v = v$. Equation (2.11) along with the above analysis covers all approximation errors.

In the backward recursive case, we can use Equation (2.12) to study the error propagation at each time step. For example, in our BSDE algorithm, we let $v = y^\pi_{p+1}$ or z^π_{p+1} and $\rho_v = \hat{y}^\pi_{p+1}$ or $\hat{z}^\pi_{p+1}$. In these cases, the fifth term of Equation (2.12) is comparing our approximation with the true value at the next time step. The error accumulates in a recursive way by applying this error analysis from time step t_0 to t_{p-1}. Further discussion of the error propagation will be given in Section 2.3.4.

The error for approximating $\mathbb{E}[v(X^\pi_{t+\Delta t})(W_{t+\Delta_t} - W_t)|X^\pi_t = x]$ with

$$\hat{\mathbb{E}}[v(X^\pi_{t+\Delta t})(W_{t+\Delta_t} - W_t)|X^\pi_t = x] := \frac{1}{N}\sum_{r=1-N}^{N} \rho_v\left(\frac{r}{2^\vartheta}\right)\mathbb{E}[\varphi_{N,r}(X^\pi_{t+\Delta t})(W_{t+\Delta} - W_t)|X^\pi_t = x], \tag{2.13}$$

can be studied in a similar way.

Remark 2.4. It is clear from the derivation that the assumption of the function v being continuous with left- and right-side derivatives is crucial in applying the quick version of the SWIFT formula. In our FBSDE algorithm, the functions y and z at intermediate time points, $p = 1, \ldots, P-1$, satisfy the above conditions. This can be observed from Equations (2.5) and (2.9). However, we may still face an issue at the terminal time, as $D_x g$ may contain discontinuities. *We propose a mixed algorithm to deal with this situation.* At the terminal time, the usual SWIFT formulas are used and then the algorithm switches to the quick version in all subsequent time steps, see Algorithm 2.

2.3.3. PICARD ITERATION ERROR

When $\theta_1 \neq 0$, a Picard iteration must be performed with the equation:

$$y = \Delta t \theta_1 f(t_p, x, y, \hat{z}^\pi_p(x)) + \hat{h}(t_p, x),$$

to find the fixed point y. It is well-known that this iterative algorithm will converge to the unique fixed point if the function $\Delta t \theta_1 f$ is a contraction map of y, namely,

$$|\Delta t \theta_1 f(t_p, x, y_1, \hat{z}^\pi_p(x)) - \Delta t \theta_1 f(t_p, x, y_2, \hat{z}^\pi_p(x))| \leq \xi |y_1 - y_2|,$$

with $\xi \in [0, 1)$ for all $x \in (-2^{-\vartheta}N, 2^{-\vartheta}N]$. This condition is satisfied when the driver function is Lipschitz in y and Δt is small enough.

For $r = 1-N$ to N **do**,
$\hat{y}_P^{\pi}(2^{-\vartheta}r) =< g, \varphi_{N,r} >$, $\hat{z}_P^{\pi}(2^{-\vartheta}r) =< \sigma D_x g, \varphi_{N,r} >$ and
$\hat{f}_P^{\pi}(2^{-\vartheta}r) =< f(T,\cdot,g(\cdot),\sigma D_x(\cdot)), \varphi_{N,r} >$.
Compute $(\mathbb{E}_{P-1}^{2^{-\vartheta}r}[\varphi_{N,k}(X_{t_P}^{\pi})])_{r,k=1-N,\dots,N}$ and $(\mathbb{E}_{P-1}^{2^{-\vartheta}r}[\varphi_{N,k}(X_{t_P}^{\pi})\Delta W_P])_{r,k=1-N,\dots,N}$ with (2.3) and (2.4).
For $p = P-1$ to 1 **do**,
Compute the function $(\hat{z}_p^{\pi}(2^{-\vartheta}r))_{r=1-N,\dots,N}$ with (2.5).
Compute the function $(\hat{y}_p^{\pi}(2^{-\vartheta}r))_{r=1-N,\dots,N}$ with (2.9) and Picard iterations if necessary.
Compute the functions $(f(t_p, 2^{-\vartheta}r, \hat{y}_p^{\pi}(2^{-\vartheta}r), \hat{z}_p^{\pi}(2^{-\vartheta}r)))_{r=1-N,\dots,N}$.
Compute $(\mathbb{E}_{p-1}^{2^{-\vartheta}r}[\varphi_{N,k}(X_{t_p}^{\pi})])_{r,k=1-N,\dots,N}$ and $(\mathbb{E}_{p-1}^{2^{-\vartheta}r}[\varphi_{N,k}(X_{t_p}^{\pi})\Delta W_p])_{r,k=1-N,\dots,N}$ with (2.3) and (2.4) if the distribution of $(X_{t_p}^{\pi} - X_{t_{p-1}}^{\pi})$ is time-dependent.
Compute $\hat{z}_0^{\pi}(x_0)$ and $\hat{y}_0^{\pi}(x_0)$ at time zero.

Algorithm 2: Mixed SWIFT method

We adopt the following notation:

$$\begin{cases} \hat{y}_P^{\pi,I}(x) := g(x); \\ \hat{y}_p^{\pi,0}(x) := \dfrac{1}{N}\displaystyle\sum_{r=1-N}^{N} \hat{y}_{p+1}^{\pi,I}\left(\dfrac{r}{2^{\vartheta}}\right) \Re\left\{\sum_{k=1}^{N} e^{\imath 2^{\vartheta} C_k x} \mathscr{F}(t_p, x, 2^{\vartheta} C_k) e^{-\imath C_k r}\right\}; \\ \hat{y}_p^{\pi,i+1}(x) := \Delta t \theta_1 f(t_p, x, \hat{y}_p^{\pi,i}(x), \hat{z}_p^{\pi}(x)) + \hat{h}(t_p, x), \end{cases}$$

for $p = 0,\dots,P-1$ and $i = 0,\dots,I-1$. It is clear that $\hat{y}_p^{\pi,I}(x) = \hat{h}(t_p,x)$ when $\theta_1 = 0$ and $I \geq 1$, which is the explicit scheme. The above notations are consistent with the notations in Section 2.2.4 except we should replace the $\hat{y}_{p+1}^{\pi}$ with $\hat{y}_{p+1}^{\pi,I}$ in equation (2.6). Furthermore, for any given x, we know by definition that $y_p^{\pi}(x)$ is the unique fixed point that satisfies

$$y = \Delta t \theta_1 f(t_p, x, y, z_p^{\pi}(x)) + h(t_p, x).$$

Note that the notation $\hat{y}_p^{\pi}$ was defined by Equation (2.9).

With the above notations and given the extra information that f is Lipschitz with respect to z, we can derive the one-step approximation error for function y^{π}:

$$\begin{aligned} &|\hat{y}_p^{\pi,I}(x) - y_p^{\pi}(x)| \leq |\hat{y}_p^{\pi,I}(x) - \hat{y}_p^{\pi}(x)| + |\hat{y}_p^{\pi}(x) - y_p^{\pi}(x)| \\ \leq &\varepsilon_p^{Picard} + \Delta t \theta_1 |f(t_p, x, \hat{y}_p^{\pi}(x), \hat{z}_p^{\pi}(x)) - f(t_p, x, y_p^{\pi}(x), z_p^{\pi}(x))| + |\hat{h}(t_p,x) - h(t_p,x)| \\ \leq &\varepsilon_p^{Picard} + \xi |\hat{y}_p^{\pi}(x) - y_p^{\pi}(x)| + \xi |\hat{z}_p^{\pi}(x) - z_p^{\pi}(x)| + |\hat{h}(t_p,x) - h(t_p,x)| \\ \leq &(1+\xi)\varepsilon_p^{Picard} + \xi |\hat{y}_p^{\pi,I}(x) - y_p^{\pi}(x)| + \xi |\hat{z}_p^{\pi}(x) - z_p^{\pi}(x)| + |\hat{h}(t_p,x) - h(t_p,x)|. \end{aligned}$$

The term $\varepsilon_p^{Picard} := |\hat{y}_p^{\pi,I}(x) - \hat{y}_p^{\pi}(x)|$ is the error of applying Picard iterations, which depends on Δt and the Lipschitz coefficient of f with respect to y, as stated before in this section. The constant $\mathscr{C}$ is related to the Lipschitz coefficient of f from standing assumption (A3) and $\xi := \mathscr{C}\Delta t \theta_1 \leq 1$. The last inequality is due to a telescoping argument

of the term $|\hat{y}_p^\pi(x) - y_p^\pi(x)|$. Rearranging the terms gives us the following error bound:

$$|\hat{y}_p^{\pi,I}(x) - y_p^\pi(x)| \leq \frac{1+\xi}{1-\xi}\varepsilon_p^{Picard} + \frac{1}{1-\xi}(\xi|\hat{z}_p^\pi(x) - z_p^\pi(x)| + |\hat{h}(t_p,x) - h(t_p,x)|). \quad (2.14)$$

2.3.4. THE ERRORS OF THE FBSDE RECURSIVE SCHEME

Given an FBSDE system, a time partition and a discretization scheme, we may apply the result in Section 2.3.2 to derive a local approximation error for the related expectations. When we are approximating expectations with the functions y_{p+1}^π and z_{p+1}^π in the BSDE scheme, the approximation vector is given by

$$\begin{cases} \rho_{y_{p+1}^\pi} = \{\hat{y}_{p+1}^{\pi,I}(2^{-\vartheta}r)\}_{r=1-N,\dots,N}; \\ \rho_{z_{p+1}^\pi} = \{\hat{z}_{p+1}^{\pi}(2^{-\vartheta}r)\}_{r=1-N,\dots,N}, \end{cases}$$

for $p = 0,\dots,P-2$. At the terminal time $t_P = T$, they are defined as

$$\begin{cases} \rho_{y_p^\pi} = \{Y_T^\pi | X_T^\pi = 2^{-\vartheta}r\}_{r=1-N,\dots,N}; \\ \rho_{z_p^\pi} = \{Z_T^\pi | X_T^\pi = 2^{-\vartheta}r\}_{r=1-N,\dots,N}, \end{cases}$$

or

$$\begin{cases} \rho_{y_P^\pi} = \{< Y_T^\pi, \varphi_{N,r} >\}_{r=1-N,\dots,N}; \\ \rho_{z_P^\pi} = \{< Z_T^\pi, \varphi_{N,r} >\}_{r=1-N,\dots,N}, \end{cases}$$

depending on the scheme we used. When approximating expectations involving $f(t_{p+1}, x, y_{p+1}^\pi(x), z_{p+1}^\pi(x))$, the approximation vector

$$\rho_{f_{p+1}} = \{f(t_{p+1}, 2^{-\vartheta}r, \rho_{y_{p+1}^\pi}(2^{-\vartheta}r), \rho_{z_{p+1}^\pi}(2^{-\vartheta}r)\}_{r=1-N,\dots,N},$$

for $p = 0,\dots P-1$.

From Equation (2.12), we know that the approximation error for the SWIFT formula consists of four parts. Therefore, the local approximation errors at point (t,x) by the SWIFT formula for any function v, denoted as $\zeta_v^{\vartheta,N}(t_p,x)$ or $\zeta_v^{\vartheta,N,W}(t_p,x)$ for the two types of expectations, are given by

$$\begin{aligned} \zeta_v^{\vartheta,N}(t_p,x) :=& \mathbb{E}_p^x[v(X_{t_{p+1}}^\pi)\mathbf{1}_{\{X_{t_{p+1}}^\pi \notin (-2^{-\vartheta}N, 2^{-\vartheta}N]\}}] - \mathbb{E}_p^x[\tilde{v}(X_{t_{p+1}}^\pi)\mathbf{1}_{\{X_{t_{p+1}}^\pi \notin (-2^{-\vartheta}N, 2^{-\vartheta}N]\}}] \\ &+ \mathbb{E}_p^x[\tilde{v}(X_{t_{p+1}}^\pi) - H_{V_N} v(X_{t_{p+1}}^\pi)] \\ &+ \frac{1}{N}\sum_{r=1-N}^{N}\left(H_{V_N} v\left(\frac{r}{2^\vartheta}\right) - v\left(\frac{r}{2^\vartheta}\right)\right)\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^\pi)]; \\ \zeta_v^{\vartheta,N,W}(t_p,x) :=& \mathbb{E}_p^x[v(X_{t_{p+1}}^\pi)\mathbf{1}_{\{X_{t_{p+1}}^\pi \notin (-2^{-\vartheta}N, 2^{-\vartheta}N]\}}\Delta W_{p+1}] \\ &- \mathbb{E}_p^x[\tilde{v}(X_{t_{p+1}}^\pi)\mathbf{1}_{\{X_{t_{p+1}}^\pi \notin (-2^{-\vartheta}N, 2^{-\vartheta}N]\}}\Delta W_{p+1}] \\ &+ \mathbb{E}_p^x[(\tilde{v}(X_{t_{p+1}}^\pi) - H_{V_N} v(X_{t_{p+1}}^\pi))\Delta W_{p+1}] \\ &+ \frac{1}{N}\sum_{r=1-N}^{N}\left(H_{V_N} v\left(\frac{r}{2^\vartheta}\right) - v\left(\frac{r}{2^\vartheta}\right)\right)\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^\pi)\Delta W_{p+1}]. \end{aligned}$$

Applying all the results above and the standing assumptions, we can derive the recurring error formulas of our SWIFT BSDE scheme:

$$\begin{aligned}
&\frac{1}{\mathscr{C}_1}|z_p^\pi(x)-\hat{z}_p^\pi(x)|\\
\leq&|\zeta^{\vartheta,N}_{z^\pi_{p+1}}(t_p,x)|+|\zeta^{\vartheta,N,W}_{y^\pi_{p+1}}(t_p,x)|+|\zeta^{\vartheta,N,W}_{f_{p+1}}(t_p,x)|\\
&+\frac{1}{N}\sum_{r=1-N}^{N}\left|z^\pi_{p+1}\left(\frac{r}{2^\vartheta}\right)-\rho_{z^\pi_{p+1}}\left(\frac{r}{2^\vartheta}\right)\right|(|\mathbb{E}_p^x[\varphi_{N,r}(X^\pi_{t_{p+1}})]|+\mathbb{E}_p^x[\varphi_{N,r}(X^\pi_{t_{p+1}})\Delta W_{p+1}]|)\\
&+\frac{1}{N}\sum_{r=1-N}^{N}\left|y^\pi_{p+1}\left(\frac{r}{2^\vartheta}\right)-\rho_{y^\pi_{p+1}}\left(\frac{r}{2^m}\right)\right||\mathbb{E}_p^x[\varphi_{J,r}(X^\pi_{t_{p+1}})\Delta W_{p+1}]|,
\end{aligned}\tag{2.15}$$

and

$$\begin{aligned}
&\frac{1}{\mathscr{C}_2}|y_p^\pi(x)-\hat{y}_p^{\pi,I}(x)|\\
\leq&\mathscr{C}_3+|\zeta^{\vartheta,N}_{z^\pi_{p+1}}(t_p,x)|+|\zeta^{\vartheta,N,W}_{y^\pi_{p+1}}(t_p,x)|+|\zeta^{\vartheta,N,W}_{f_{p+1}}(t_p,x)|+|\zeta^{\vartheta,N}_{y^\pi_{p+1}}(t_p,x)|+|\zeta^{\vartheta,N}_{f_{p+1}}(t_p,x)|\\
&+\frac{1}{N}\sum_{r=1-N}^{N}\left|z^\pi_{p+1}\left(\frac{r}{2^\vartheta}\right)-\hat{z}^\pi_{p+1}\left(\frac{r}{2^\vartheta}\right)\right|(|\mathbb{E}_p^x[\varphi_{N,r}(X^\pi_{t_{p+1}})]|+\mathbb{E}_p^x[\varphi_{N,r}(X^\pi_{t_{p+1}})\Delta W_{p+1}]|)\\
&+\frac{1}{N}\sum_{r=1-N}^{N}\left|y^\pi_{p+1}\left(\frac{r}{2^\vartheta}\right)-\hat{y}^\pi_{p+1}\left(\frac{r}{2^\vartheta}\right)\right|(|\mathbb{E}_p^x[\varphi_{N,r}(X^\pi_{t_{p+1}})]|+\mathbb{E}_p^x[\varphi_{N,r}(X^\pi_{t_{p+1}})\Delta W_{p+1}]|),
\end{aligned}\tag{2.16}$$

with constants $\mathscr{C}_1$, $\mathscr{C}_2$ and $\mathscr{C}_3$ depending on the underlying FBSDE, the discretization scheme and the Picard iteration errors, but not depending on ϑ and M. Their proof is left for the appendix.

ERROR BOUND AND CHOICE OF PARAMETERS

An error bound at $(0,x_0)$ for applying the SWIFT scheme to a FBSDE system can be found by repeatedly applying Equations (2.15) and (2.16). This bound is given by the weighted sum of the local approximation errors for each point in the grid $\{(0,x_0)\}\cup\{(t_p,2^{-\vartheta}r)|p=1,\dots,P \text{ and } r=1-N,\dots,N\}$, with the weight being 1 for $(0,x_0)$ and the weight being

$$\sum_{u\in\upsilon_p}\frac{1}{N^p}\prod_{l=1}^{p}(|\mathbb{E}_l^{u_{l-1}}[\varphi_{N,u_l}(X^\pi_{t_l})]|+|\mathbb{E}_l^{u_{l-1}}[\varphi_{N,u_l}(X^\pi_{t_l})\Delta W_{l+1}]|),\tag{2.17}$$

for $\{(t_p,2^{-\vartheta}r)|p=1,\dots,P \text{ and } r=1-N,\dots,N\}$, where υ_p is the set containing length $p+1$ vectors $u=(u_0,u_1,\dots,u_p)$, with the first element $u_0=x_0$ and other elements equal to $2^{-\vartheta}r$ for $r=1-N,\dots,N$. A simple example of deriving such an error bound is included in the appendix.

However, as this error bound uses local approximation errors from multiple points in a grid, actually calculating the error bound would be costly. The weight and the local error behave differently at different grid points. Whenever $|r|$ is small, the local error converges exponentially in numerical tests with fixed $2^{-\vartheta}N$ and increasing N, but it may fail to converge to zero when $|r|$ is close to N. On the other hand, when $|r|$ is close to N,

the weight in Equation (2.17) tends to zero quickly and reduces the error. We do not have a simple formula to describe this balance. Last but not least, parameter P, the number of time points, affects the total number of terms in the error bound, the value of the local error and the value of the weight in Equation (2.17). It is highly non-trivial to quantify the effect of P on the overall error from this error bound.

In practice, we would recommend a three-step approach to select the parameters for the scheme and get a global picture of what the error may be. First of all, pick parameter P based on the discretization error for (X, Y, Z). This can be done either through the error bound in Section 2.3.1 or other existing error bounds in the literature. The reason is that ϑ and N have no effect on the discretization error while P affects each part of the approximation error. Next, we should choose our parameters N and ϑ according to error formula (2.12). The interest is in the third term in the equation, which increases with the truncation range but decreases with the wavelet order N. Therefore we should first fix the truncation range $2^{-\vartheta}N$ to a constant value a in our scheme, the tail probability outside our computational range is below a fixed tolerance level. This can be done heuristically by considering the cumulants of X_T, see [9]. Finally, we pick an N value such that the overall error converges and adjust ϑ accordingly so that the truncation range remains the same. This approach is very useful for applications (compared to the error bound itself).

2.3.5. COMPUTATIONAL COMPLEXITY

At each time-step t_p, the following operations have to be performed:

- *Computation* of $\mathbb{E}_p^x[\varphi_{N,k}(X^\pi_{t_{p+1}})]$ and $\mathbb{E}_p^x[\varphi_{N,k}(X^\pi_{t_{p+1}})\Delta W_{p+1}]$ by the FFT algorithm, in $\mathrm{O}(N\log(N))$ operations. It is calculated only once at the terminal time-step if the characteristic function of X^π does not depend on the time point;
- *Computation* of $\hat{z}_p^\pi(x)$, $\hat{h}(t_p, x)$ and $\hat{y}_p^{\pi,0}(x)$ by matrix-vector multiplications, in $\mathrm{O}(N^2)$ operations;
- *Computation* of $\hat{y}^{\pi,I}$ by I Picard iterations on an x-grid, in $\mathrm{O}(IN)$ operations;
- *Evaluation* of $f(t_p, x, \hat{y}_p^{\pi,I}(x), \hat{z}_p^\pi(x))$ in $\mathrm{O}(N)$ operations.

The most time-consuming part in our algorithm is the matrix-vector multiplication. The proposed algorithms have linear computational complexity with respect to the time-steps P and the starting evaluation at terminal time is of order $\mathrm{O}(N)$. In total, the complexity of the SWIFT-type methods is $\mathrm{O}(N + P(N^2 + IN + N\log(N) + N))$.

2.4. ANTIREFLECTIVE BOUNDARY

Recalling Equation (2.12), the approximation of $\mathbb{E}[v(X^\pi_{t+\Delta t})|X^\pi_t = x]$ by the SWIFT formula may have a significant local error when two conditions are satisfied. The first condition being that the alternating extension $\tilde{v}$ diverges from v in the range $[-\lambda - 2^{-\vartheta}N, -2^{-\vartheta}N]$ or $[2^\vartheta N, \lambda + 2^{-\vartheta}N]$, for some number $\lambda > 0$ and the second condition being that the probability of $X^\pi_{t+\Delta t}|X^\pi_t = x$ in the aforementioned ranges is large. While the first condition is almost always true, given that X^π is a diffusion process, the second

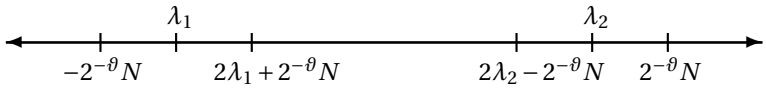


Figure 2.1: The approximation range $(-2^{-m}J, 2^{-m}J]$ and the accuracy range $[\lambda_1, \lambda_2]$.

condition is true only when the starting point x is close to the boundary $-2^{-\vartheta}N$ or $2^{-\vartheta}N$. Therefore, there may be intervals $(-2^{-\vartheta}N, \lambda_1)$ and $(\lambda_2, 2^{-\vartheta}N]$ where the SWIFT formula is inaccurate.

We propose using an antireflective boundary technique to deal with this issue. Antireflective boundary conditions form a popular technique for extrapolation in image deblurring methods. For its applications in image deblurring, the reader may refer to [10, 11] and the references therein.

In practice, assume that we approximate a function $\mathfrak{f}(x)$ by $\hat{\mathfrak{f}}(x)$ on $(-2^{-\vartheta}N, 2^{-\vartheta}N]$ and we know that the approximation is accurate for $x \in [\lambda_1, \lambda_2]$, namely, $|\mathfrak{f}(x) - \hat{\mathfrak{f}}(x)| < \epsilon$, for some small positive real number ϵ. Given the numbers $\lambda_1 > -2^{-\vartheta}N$ and $\lambda_2 < 2^{-\vartheta}N$, so that there is some inaccuracy near the boundary but $(\lambda_1, 2\lambda_1 + 2^{-\vartheta}N), (2\lambda_2 - 2^{-\vartheta}N, \lambda_2) \subset [\lambda_1, \lambda_2]$ (see Figure 2.1). We would extrapolate an approximation of $\mathfrak{f}(x)$ for $x \in (-2^{-\vartheta}N, \lambda_1)$ and $x \in (\lambda_2, 2^{-\vartheta}N]$ by applying antireflective conditions with the accurate approximation. For $x \in (-2^{-\vartheta}N, 2^{-\vartheta}N]$, we define

$$\begin{cases} \hat{\mathfrak{f}}^a(x) := 2\hat{\mathfrak{f}}(\lambda_1) - \hat{\mathfrak{f}}(2\lambda_1 - x) & \text{for } x \in (-2^{-\vartheta}N, \lambda_1); \\ \hat{\mathfrak{f}}^a(x) := \hat{\mathfrak{f}}(x) & \text{for } x \in [\lambda_1, \lambda_2]; \\ \hat{\mathfrak{f}}^a(x) := 2\hat{\mathfrak{f}}(\lambda_2) - \hat{\mathfrak{f}}(2\lambda_2 - x) & \text{for } x \in (\lambda_2, 2^{-\vartheta}N), \end{cases} \tag{2.18}$$

and use $\hat{\mathfrak{f}}^a$ instead of $\hat{\mathfrak{f}}$ as our approximation.

If $\mathfrak{f}$ is two times continuously differentiable on $\mathbb{R}$, then, by a simple application of Taylor's theorem, we have:

$$\begin{aligned} \mathfrak{f}(x) =& \mathfrak{f}(\lambda_1) + \frac{d\mathfrak{f}}{dx}(\lambda_1)(x-\lambda_1) + \frac{1}{2}\frac{d^2\mathfrak{f}}{dx^2}(\varsigma_1)(x-\lambda_1)^2; \\ \mathfrak{f}(2\lambda_1 - x) =& \mathfrak{f}(\lambda_1) - \frac{d\mathfrak{f}}{dx}(\lambda_1)(x-\lambda_1) + \frac{1}{2}\frac{d^2\mathfrak{f}}{dx^2}(\varsigma_2)(x-\lambda_1)^2, \end{aligned}$$

where $x \in (2^{-\vartheta}N, \lambda_1)$, $\varsigma_1 \in (x, \lambda_1)$ and $\varsigma_2 \in (\lambda_1, 2\lambda_1 - x)$. The approximation error for $x \in (-2^{-\vartheta}N, \lambda_1)$ is then bounded by

$$\begin{aligned} |\mathfrak{f}(x) - \hat{\mathfrak{f}}^a(x)| \leq& |\mathfrak{f}(x) - 2\mathfrak{f}(\lambda_1) + \mathfrak{f}(2\lambda_1 - x)| + 2|\mathfrak{f}(\lambda_1) - \hat{\mathfrak{f}}(\lambda_1)| + |\mathfrak{f}(2\lambda_1 - x) - \hat{\mathfrak{f}}(2\lambda_1 - x)| \\ \leq& (-2^{-\vartheta}N - \lambda_1)^2 \sup_{\varsigma \in (-2^{-\vartheta}N, 2\lambda_1 + 2^{-\vartheta}N)} \left| \frac{d^2\mathfrak{f}}{dx^2}(\varsigma) \right| + 3\epsilon. \end{aligned}$$

A similar formula can be derived for the set $(2\lambda_2 - 2^{-\vartheta}N, 2^{-\vartheta}N)$. For the recursive scheme, one can just apply Equation (2.18) at each time-step.

Remark 2.5. The range of accurate approximations for the SWIFT formula depends on the distribution of $X^{\pi}_{t+\Delta t}|_{X^{\pi}_t = x}$, and, therefore, it is model dependent. The performance

of applying the antireflective boundary technique with the SWIFT formula depends on the smoothness of the target function with respect to starting point x, the accuracy of the SWIFT formula in the range $[\lambda_1, \lambda_2]$ and the length of the range.

2.5. NUMERICAL EXPERIMENTS

Several numerical studies have been performed in MATLAB 9.0.0. The computer is equipped with Intel(R) Core(TM) i5-2520M CPU @ 2.50GHz and 7.7 GB RAM. Four different discretization schemes have been used to test the effect of various choices of θ on the numerical algorithm. They are:

Scheme A:	$\theta_1 = 0, \theta_2 = 1,$	Scheme C:	$\theta_1 = 1, \theta_2 = 1,$
Scheme B:	$\theta_1 = 0.5, \theta_2 = 1,$	Scheme D:	$\theta_1 = 0.5, \theta_2 = 0.5.$

The function z_p^π is solved explicitly in all schemes while y_p^π is solved explicitly for scheme A and implicitly with 5 Picard iterations for the other schemes.

For each given example, we associate our numerical algorithm with the computational domain $[\kappa_1 - L\sqrt{\kappa_2}, \kappa_1 + L\sqrt{\kappa_2}]$, where cumulants $\kappa_1 = x_0 + \mu(0, x_0)T$ and $\kappa_2 = \sigma^2(0, x_0)T$ and $L = 10$. It is similar to the setting in [5]. The value $2^{-\theta}N = L\sqrt{\kappa_2}$ is a constant for each example. The value of N is assumed to be 2^9.

2.5.1. EXAMPLE 1

This example has been studied in [5] and is originally from [12]. The considered FBSDE is

$$\begin{cases} dX_t = dW_t, \\ dY_t = -(Y_t Z_t - Z_t + 2.5Y_t - \sin(t + X_t)\cos(t + X_t) - 2\sin(t + X_t))dt + Z_t dW_t. \end{cases}$$

We take the initial and terminal conditions $x_0 = 0$ and $Y_T = \sin(X_T + T)$.

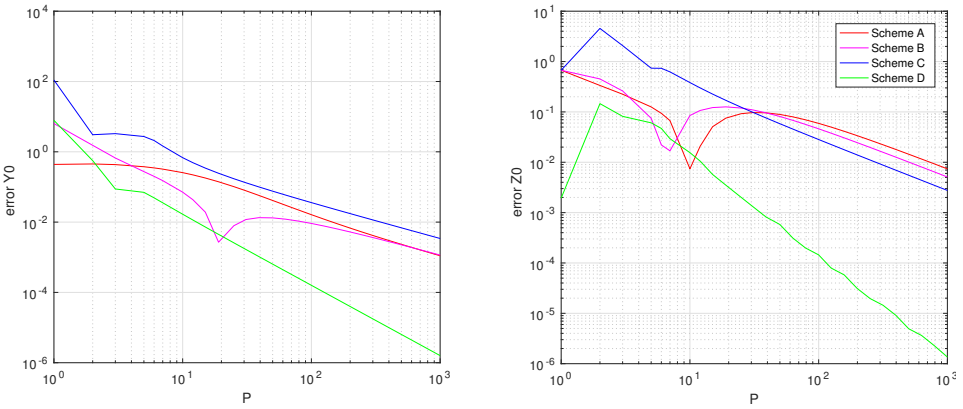
The exact solution is given by

$$(Y_t, Z_t) = (\sin(X_t + t), \cos(X_t + t)).$$

The terminal time is set to be $T = 1$ and $(Y_0, Z_0) = (0, 1)$. The driver function f depends on both time t and current state X_t. The results for the quick SWIFT method are presented in Figure 2.2a while the results for the *mixed SWIFT method* are presented in Figure 2.2b. We observe that there are no significant differences between the quick SWIFT and mixed SWIFT method. For schemes A, B and C, both approximation errors for $Y_0(x_0)$ and $Z_0(x_0)$ are of $O(\Delta t)$ order, while the errors converge with $O((\Delta t)^2)$ for scheme D.

Remark 2.6. The driver functions for some examples in this section are not universally Lipschitz. However, one should notice that for the contraction argument in Section 2.3.3 to be valid, for any fixed z, the driver function should be Lipschitz with respect to y. All the driver functions in this section satisfy this weaker condition. We aim for clear presentation rather than general applicability when we designed the assumptions and conditions and our method can be applied to a broader class of BSDEs than we described here. For a more in-depth analysis of the application of Picard iterations in the numerical treatment of BSDEs, the reader is referred to [8].

(a) Quick SWIFT with $N = 2^9$



(b) Mixed SWIFT with $N = 2^9$

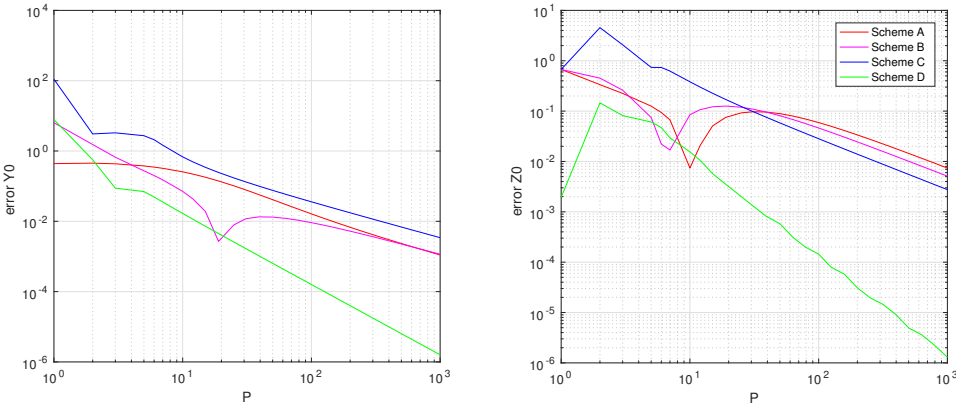


Figure 2.2: Results example 1, left: error in $y(0,x_0)$, right: error in $z(0,x_0)$

2.5.2. EXAMPLE 2: EUROPEAN CALL OPTION

Next, we calculate the price $v(t,S_T)$ of a call option under the Black-Scholes model by solving an FBSDE. The underlying process satisfies:

$$dS_t = \bar{\mu} S_t dt + \bar{\sigma} S_t dW_t.$$

Along the line of reasoning in [5], we assume that the financial market is complete, there are no trading restrictions and the call option can be perfectly hedged. Then $(v(t,S_t), \bar{\sigma} S_t D_s v(t,S_t))$ solves the FBSDE,

$$\begin{cases} dS_t = \bar{\mu} S_t dt + \bar{\sigma} S_t dW_t, \\ dY_t = -(-\bar{r} Y_t - \frac{\bar{\mu}-\bar{r}}{\bar{\sigma}} Z_t) dt + Z_t dW_t, \end{cases}$$

with terminal condition $Y_T = \max(S_T - K, 0)$. The driver function is continuous and linear with respect to y and z. We use the following parameter values in our test:

$$S_0 = 100, K = 100, \bar{r} = 0.1, \bar{\mu} = 0.2, \bar{\sigma} = 0.25, T = 0.1.$$

The exact solutions $Y_0 = 3.65997$ and $Z_0 = 14.14823$ are given by the Black-Scholes formula first shown in [13]. We switch to the log-asset domain $X_t = \log(S_t)$ and solve

$$\begin{cases} dX_t = \left(\bar{\mu} - \frac{1}{2}\bar{\sigma}^2\right) dt + \bar{\sigma} dW_t, \\ dY_t = -(-\bar{r} Y_t - \frac{\bar{\mu} - \bar{r}}{\bar{\sigma}} Z_t) dt + Z_t dW_t, \end{cases} \tag{2.19}$$

where $Y_T = \max(\exp(X_T) - K, 0)$.

For the result of the quick SWIFT method, we refer to Figure 2.3a. Immediately, we notice that the result of scheme D does not improve when increasing the number of time-steps. This is due to the discontinuity of the terminal value of Z which creates a significant error. For the mixed SWIFT method, shown in Figure 2.3b, the error from the discontinuity has been removed. The approximate values $\hat{y}_0^\pi(x_0)$ and $\hat{z}_0^\pi(x_0)$ converge with approximately order one with respect to Δt for schemes A, B, and C and about order two for scheme D.

Since the driver function in Equation (2.19) depends on $\bar{\mu}$, the approximation error also depends on $\bar{\mu}$, even though the final result $v(0, x_0)$ is unrelated to the drift. For the same number of time-steps P, the error increased with the increase of $\bar{\mu}$, as shown in Figure 2.4.

The approximation algorithm can be further improved by applying antireflective boundary conditions in the recursive time-steps. In Figure 2.5 we see results of adding an antireflective step in a mixed SWIFT algorithm. The approximations near the middle of computational range are almost identical with the reference value, but the approximations with antireflective adjustment near both ends of the interval appear to be much better than the ones without.

2.5.3. EXAMPLE 3: BID-ASK SPREAD FOR INTEREST RATE

We next consider a financial model introduced in [14], where we have distinct borrowing and lending interest rates. The resulting market is imperfect and the driver function is non-linear.

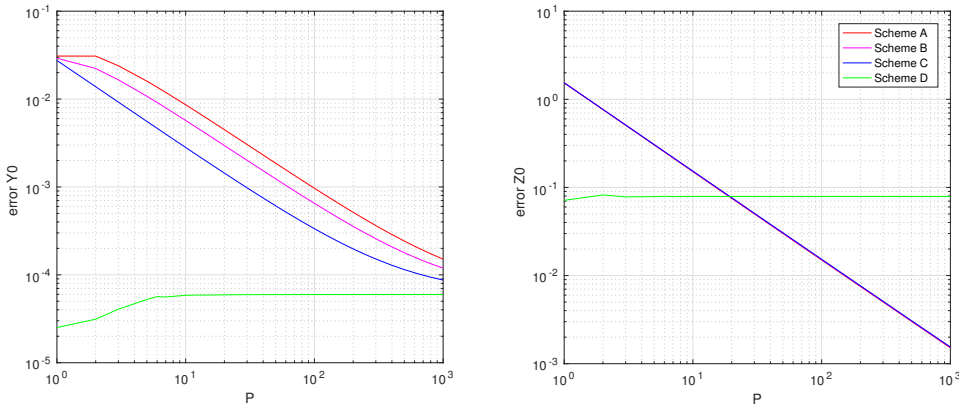
Suppose that an agent can invest in bonds with risk-free return rate r_r and borrow money at rate $r_b > r_r$. For any European-type derivative with payoff $g(X_T)$ at time T, where the underlying asset $S_t = \log(X_t)$ follows a geometric Brownian motion, its price at time 0 can be obtained by solving the FBSDE:

$$\begin{cases} dX_t = \left(\bar{\mu} - \frac{1}{2}\bar{\sigma}^2\right) dt + \bar{\sigma} dW_t, \\ dY_t = -\left(-r_r Y_t - \frac{\bar{\mu} - r_r}{\bar{\sigma}} Z_t - (r_b - r_r) \min\left(Y_t - \frac{Z_t}{\bar{\sigma}}, 0\right)\right) dt + Z_t dW_t, \end{cases}$$

with the payoff as the terminal condition. We use the example studied in both [15] and [5]. The payoff function is given by

$$g(X_T) = (e^{X_T} - K_1)^+ - 2(e^{X_T} - K_2)^+,$$

(a) Quick SWIFT with $N = 2^9$



(b) Mixed SWIFT with $N = 2^9$

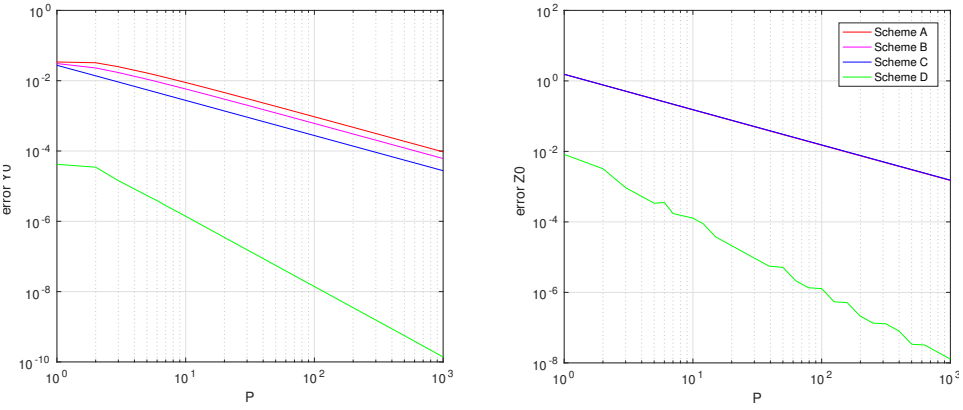


Figure 2.3: Results example 2, left: error in $y(0,x_0)$, right: error in $z(0,x_0)$

which equals a combination of a long call with strike $K_1 = 95$ and two short calls with strike $K_2 = 105$. We use the parameter values

$$S_0 = 100, r_r = 0.01, \bar{\mu} = 0.05, \bar{\sigma} = 0.2, T = 0.25, K_1 = 95, K_2 = 105, r_b = 0.06.$$

We notice that $\hat{z}_0^{\pi}(x_0)$ fails to converge to the reference value with scheme D in Figure 2.6a. The reference values, $Y_0 = 2.9584544$ and $Z_0 = 0.55319$, are obtained by the BCOS method with a large number of time-steps P. Switching to the mixed SWIFT method, whose results are shown in Figure 2.6b, the approximated error for Y converges to zero with order of about one for schemes A, B and C and converges with order $\frac{3}{2}$ for scheme D. For schemes B and C, we also have a first-order convergence for Z but the convergence order is higher for schemes A and D.

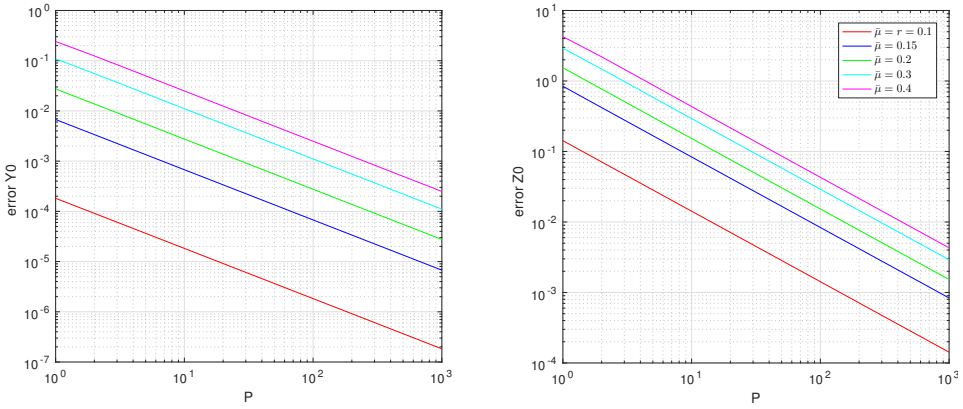


Figure 2.4: Results example 2 for different values of $\bar{\mu}$ (Scheme C)

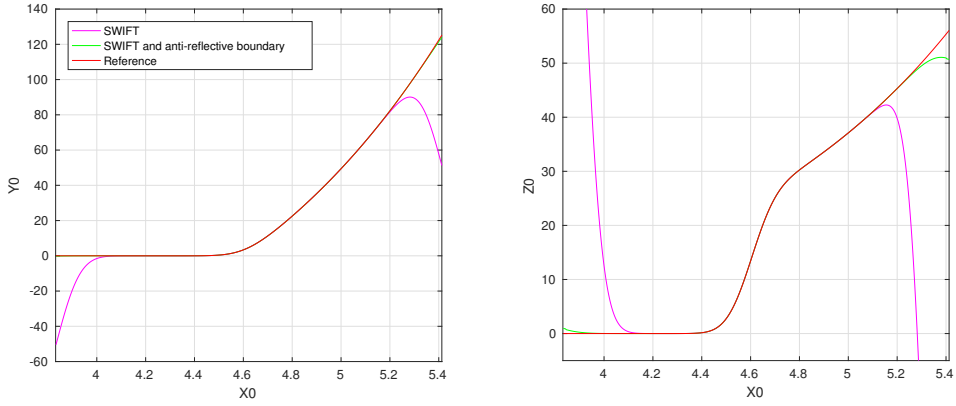


Figure 2.5: Results example 2 with and without applying antireflective boundary technique (Scheme C, $P = 1000$ and $N = 2^9$)

2.5.4. EXAMPLE 4

This example is taken from [16]. For the forward process, the drift and diffusion coefficients are time- and state-dependent. We aim to solve the following FBSDE:

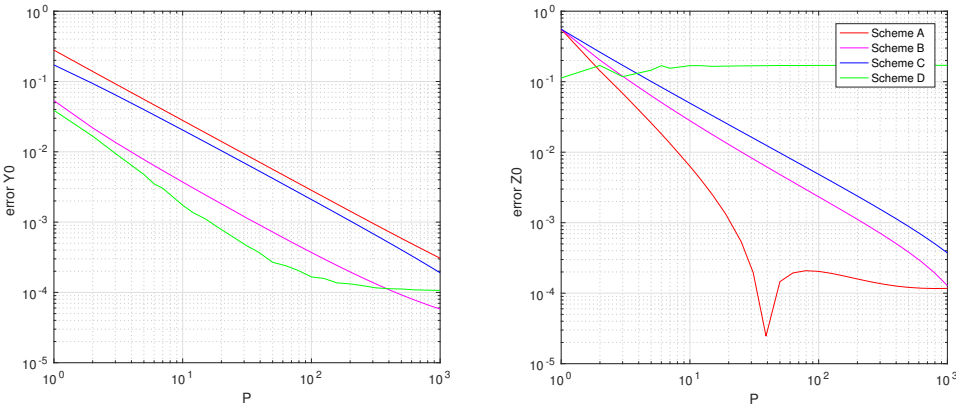
$$\begin{cases} dX_t = \frac{1}{1+2\exp(t+X_t)}\,dt + \frac{\exp(t+X_t)}{1+\exp(t+X_t)}\,dW_t, \\ dY_t = -\left(-\frac{2Y_t}{1+2\exp(t+X_t)} - \frac{1}{2}\left(\frac{Y_t Z_t}{1+\exp(t+X_t)} - Y_t^2 Z_t\right)\right)dt + Z_t\,dW_t, \end{cases}$$

with the terminal condition $Y_T = g(X_T) = \frac{\exp(T+X_T)}{1+\exp(T+X_T)}$.

The exact solutions are given by

$$(Y_t, Z_t) = \left(\frac{\exp(t+X_t)}{1+\exp(t+X_t)}, \frac{(\exp(t+X_t))^2}{(1+\exp(t+X_t))^3}\right).$$

(a) Quick SWIFT with $N = 2^9$



(b) Mixed SWIFT with $N = 2^9$

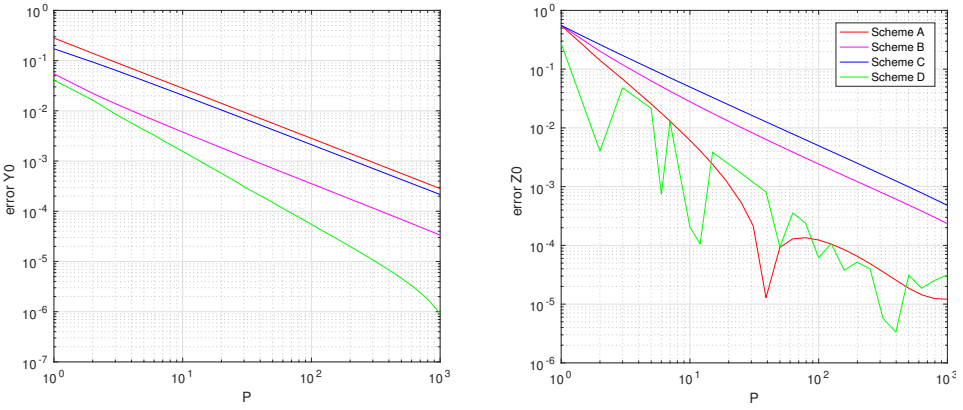


Figure 2.6: Results example 3, left: error in $y(0, x_0)$, right: error in $z(0, x_0)$

We choose terminal time $T = 1$ and initial condition $x_0 = 1$.

For the results of the quick SWIFT method, we refer to Figure 2.7. While the total error is different for each scheme, the approximated values $\hat{y}_0^{\pi}(x_0)$ and $\hat{z}_0^{\pi}(x_0)$ converge with $O(\Delta t)$ for all schemes, as expected. Here the weak order of the Euler scheme plays a prominent role as the drift and volatility are state- and time- dependent.

2.5.5. DISCUSSION

Compared with the BCOS method, the computational time for the SWIFT-type method is slightly lower when the number of basis functions used is the same and the forward process is independent of time. The most time-consuming portion is the matrix-vector multiplication used to calculate the value of $\hat{z}_p^{\pi}$ and $\hat{h}$. We acknowledge that for the same error range, the BCOS and SWIFT-type methods may require different numbers of basis

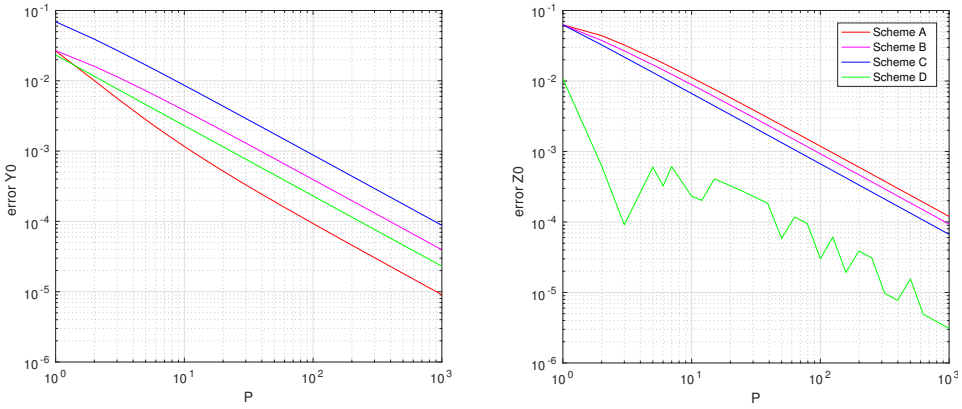


Figure 2.7: Results example 4, ($N = 2^9$), left: error in $y(0, x_0)$, right: error in $z(0, x_0)$

functions.

From the numerical experiments, we conclude that the computation with scheme D often fails to converge with Δt when the time-step is small when using the quick SWIFT method. This is due to the discontinuity of $z_P^{\pi}(x)$ in x. Schemes A, B and C behave similarly for the quick SWIFT and mixed SWIFT methods in our examples. However, scheme D often has the best performance in the mixed SWIFT method. This means that damping the discontinuity in our scheme is beneficial. The graphs also demonstrate that with a proper choice of SWIFT parameters, the approximation error itself will be dominated by the discretization error and decreases with respect to the increase of parameter P. It implies that the error of calculating expectations with SWIFT is relatively small.

2.6. CONCLUSION

A new probabilistic method for solving FBSDEs numerically has been proposed in this chapter. It is derived from a time-discretization of the forward and backward stochastic differential equations, taking conditional expectations to get an $\mathfrak{F}_{t_p}$-adapted approximation and calculating the conditional expectations with the quick variant of the SWIFT formula.

We have shown that in order to apply the quick variant of the SWIFT formula, the continuity of the target function has to be ensured. While applying the quick variant of the SWIFT formula instead of the original version introduces an extra error, it is of the same order as the original version when the target function is continuous and drops quickly with respect to N, due to the exponential convergence of the characteristic function for a smooth density. The error of applying the SWIFT method is relatively minor compared to the discretization error for the stochastic process. However, the quick variant of the SWIFT formula can greatly reduce the difficulties of our algorithm and increase the computational speed. So, we believe that the mixed SWIFT method provides a good balance between efficiency and accuracy.

We have discussed the different approximation errors in detail in this work. Addi-

tional attention is needed for the error of the SWIFT formula near the computational boundary, as we explained in Section 2.4. We also demonstrated how to improve our algorithm with the antireflective boundary conditions. Finally, the applicability and the effectiveness of our numerical algorithm have been tested with various FBSDEs, which all give positive results with the mixed SWIFT method.

Overall, applying the SWIFT method to solve discretized BSDEs retains the high accuracy of Fourier inversion techniques, although the computations involved are greatly simplified. We also gain additional freedom in adjusting the approximation values at each time point.

APPENDIX

POINTWISE CONVERGENCE OF ORTHOGONAL PROJECTION

We shall demonstrate that under some mild assumptions on a square integrable function in $(-2^{-\vartheta}N, 2^{-\vartheta}N]$, its orthogonal projection on V_N converges to the original function in a pointwise fashion, therefore bounding our approximation error. It is an adaptation of the standard Dirichlet kernel argument to our setting, a similar proof can be found in standard Fourier series textbook, like [17].

Theorem 2.4. *Let $\mathfrak{g}$ be a square integrable function defined on the set $(-2^{-\vartheta}N, 2^{-\vartheta}N]$ and the left- and right-side derivatives exist everywhere for its alternating extension $\tilde{\mathfrak{g}}$. If $\tilde{\mathfrak{g}}$ is continuous in a neighborhood around point x, the following result holds:*

$$\lim_{N\to\infty} H_{V_N}\mathfrak{g}(x) = \tilde{\mathfrak{g}}(x).$$

Proof. By direct calculation,

$$\begin{aligned}
&H_{V_N}\mathfrak{g}(x)\\
=&\frac{2^{\vartheta}}{N}\sum_{k=1}^{N}\left(\int_{-2^{-\vartheta}N}^{2^{-\vartheta}N}\mathfrak{g}(\varsigma)\cos\left(2^{\vartheta}C_k\varsigma\right)d\varsigma\cos\left(2^{\vartheta}C_kx\right)+\int_{-2^{-\vartheta}N}^{2^{-\vartheta}N}\mathfrak{g}(\varsigma)\sin\left(2^{\vartheta}C_k\varsigma\right)d\varsigma\sin\left(2^{\vartheta}C_kx\right)\right)\\
=&\frac{2^{\vartheta}}{N}\sum_{k=1}^{N}\left(\int_{-2^{-\vartheta}N}^{2^{-\vartheta}N}\tilde{\mathfrak{g}}(\varsigma)\cos\left(2^{\vartheta}C_k(\varsigma-x)\right)d\varsigma\right)=\frac{2^{\vartheta}}{N}\sum_{k=1}^{N}\left(\int_{-2^{-\vartheta}N}^{2^{-\vartheta}N}\tilde{\mathfrak{g}}(\omega+x)\cos\left(2^{\vartheta}C_k\omega\right)d\omega\right)\\
=&\frac{2^{\vartheta}}{N}\int_{-2^{-\vartheta}N}^{2^{-\vartheta}N}\tilde{\mathfrak{g}}(\omega+x)\Upsilon_N(\omega)d\omega,
\end{aligned}$$

where

$$\Upsilon_N(x) := \sum_{k=1}^{N}\cos(2^{\vartheta}C_kx) = \frac{\sin(2^{\vartheta-1}\pi x)}{\sin(2^{\vartheta-1}\pi x/N)}.$$

It can be shown that

$$\frac{2^{\vartheta}}{N}\int_{0}^{2^{-\vartheta}N}\Upsilon_N(\varsigma)d\varsigma = \frac{2^{\vartheta}}{N}\int_{-2^{-\vartheta}N}^{0}\Upsilon_N(\varsigma)d\varsigma = \frac{2}{\pi}\sum_{k=1}^{N}\frac{(-1)^{k+1}}{2k-1} =: \frac{2}{\pi}G_N.$$

In fact, G_N is the famous Gregory-Leibniz series and we have $\lim_{N\to\infty}G_N = \frac{\pi}{4}$.

Based on our assumption, at point x, $\lim_{h\to 0+}\tilde{\mathfrak{g}}(x+h)=\lim_{h\to 0-}\tilde{\mathfrak{g}}(x+h)=\tilde{\mathfrak{g}}(x)$. Also, $\frac{\mathfrak{g}(s+x)-\mathfrak{g}(x)}{s}\mathbf{1}_{\{s\in(0,2^{-\vartheta}N)\}}$ and $\frac{\mathfrak{g}(s+x)-\mathfrak{g}(x)}{s}\mathbf{1}_{\{s\in(-2^{-\vartheta}N,0)\}}$ are integrable functions on $(2^{-\vartheta}N,2^{-\vartheta}N]$. Note that in our construction, $2^{-\vartheta}N$ is a positive constant (a) and ϑ is a function of N.

Therefore,

$$\begin{aligned}
&H_{V_N}\mathfrak{g}(x)-\frac{4}{\pi}G_N\tilde{\mathfrak{g}}(x)\\
=&\frac{2^\vartheta}{N}\int_0^a(\tilde{\mathfrak{g}}(\omega+x)-\tilde{\mathfrak{g}}(x))\Upsilon_N(\omega)d\omega+\frac{2^\vartheta}{N}\int_{-a}^0(\tilde{\mathfrak{g}}(\omega+x)-\tilde{\mathfrak{g}}(x))\Upsilon_N(\omega)d\omega\\
=&\frac{2}{\pi}\int_0^a\frac{\tilde{\mathfrak{g}}(\omega+x)-\tilde{\mathfrak{g}}(x)}{\omega}\frac{\pi\omega/2a}{\sin(\pi\omega/2a)}\sin(2^{\vartheta-1}\pi\omega)d\omega\\
&+\frac{2}{\pi}\int_{-a}^0\frac{\tilde{\mathfrak{g}}(\omega+x)-\tilde{\mathfrak{g}}(x)}{\omega}\frac{\pi\omega/2a}{\sin(\pi\omega/2a)}\sin(2^{\vartheta-1}\pi\omega)d\omega.
\end{aligned}$$

Since $\frac{x}{\sin(x)}$ is integrable on $[-\frac{\pi}{2},\frac{\pi}{2}]$ and ϑ tends to infinity whenever N tends to infinity, the last two terms go to 0, when N tends to infinity. This is due to the Riemann-Lebesgue Lemma. So, we have

$$\lim_{N\to\infty}H_{V_N}\mathfrak{g}(x)=\lim_{N\to\infty}\frac{4}{\pi}G_N\tilde{\mathfrak{g}}(x)=\tilde{\mathfrak{g}}(x),$$

and complete the proof. □

PROOF OF EQUATIONS (2.15) AND (2.16)

Proof. Using the discretization scheme Equations (1.4b), we have

$$\begin{aligned}
|\hat{z}_p^\pi(x)-z_p^\pi(x)|\le&\frac{1-\theta_2}{\theta_2}|\mathbb{E}_p^x[z_{p+1}^\pi(X_{t_{p+1}}^\pi)]-\hat{\mathbb{E}}[z_{p+1}^\pi(X_{t_{p+1}}^\pi)|X_{t_p}^\pi=x]|\\
&+\frac{1}{\theta_2\Delta t}|\mathbb{E}_p^x[y_{p+1}^\pi(X_{t_{p+1}}^\pi)\Delta W_{p+1}]-\hat{\mathbb{E}}[y_{p+1}^\pi(X_{t_{p+1}}^\pi)\Delta W_{p+1}|X_{t_p}^\pi=x]|\\
&+\frac{1-\theta_2}{\theta_2}|\mathbb{E}_p^x[f(t_{p+1},X_{t_{p+1}}^\pi,y_{p+1}^\pi(X_{t_{p+1}}^\pi),z_{p+1}^\pi(X_{t_{p+1}}^\pi))\Delta W_{p+1}]\\
&\qquad-\hat{\mathbb{E}}[f(t_{p+1},X_{t_{p+1}}^\pi,y_{p+1}^\pi(X_{t_{p+1}}^\pi),z_{p+1}^\pi(X_{t_{p+1}}^\pi))\Delta W_{p+1}|X_{t_p}^\pi=x]|
\end{aligned}$$

Since we enforced the standing assumption (A3), we may apply the error bounds Equation (2.10), (2.13) and (2.12).

$$\begin{aligned}
&|\hat{z}_p^\pi(x)-z_p^\pi(x)|\\
\le&\frac{1-\theta_2}{\theta_2}|\zeta_{z_{p+1}^\pi}^{\vartheta,N}(t_p,x)|+\frac{1}{\theta_2\Delta t}|\zeta_{y_{p+1}^\pi}^{\vartheta,N,W}(t_p,x)|+\frac{1-\theta_2}{\theta_2}|\zeta_{f_{p+1}}^{\vartheta,N,W}(t_p,x)|\\
&+\frac{1-\theta_2}{\theta_2}\frac{1}{N}\sum_{r=1-N}^{N}\left|z_{p+1}^\pi\left(\frac{r}{2^\vartheta}\right)-\rho_{z_{p+1}^\pi}\left(\frac{r}{2^\vartheta}\right)\right||\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^\pi)]|\\
&+\frac{1}{\theta_2\Delta t}\frac{1}{N}\sum_{r=1-N}^{N}\left|y_{p+1}^\pi\left(\frac{r}{2^\vartheta}\right)-\rho_{y_{p+1}^\pi}\left(\frac{r}{2^\vartheta}\right)\right||\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^\pi)\Delta W_{p+1}]|
\end{aligned}$$

$$+\frac{1-\theta_2}{\theta_2}\frac{1}{N}\sum_{r=1-N}^{N}\left|f\left(t_{p+1},\frac{r}{2^\vartheta},y^\pi_{p+1}\left(\frac{r}{2^\vartheta}\right),z^\pi_{p+1}\left(\frac{r}{2^\vartheta}\right)\right)-\rho_{f_{p+1}}\left(\frac{r}{2^\vartheta}\right)\right||\mathbb{E}^x_p[\varphi_{N,r}(X^\pi_{t_{p+1}})\Delta W_{p+1}]|$$

$$\leq\frac{1-\theta_2}{\theta_2}|\zeta^{\vartheta,N}_{z^\pi_{p+1}}(t_p,x)|+\frac{1}{\theta_2\Delta t}|\zeta^{\vartheta,N,W}_{y^\pi_{p+1}}(t_p,x)|+\frac{1-\theta_2}{\theta_2}|\zeta^{\vartheta,N,W}_{f_{p+1}}(t_p,x)|$$

$$+\frac{(1-\theta_2)(1+\mathscr{C})}{\theta_2}\frac{1}{N}\sum_{r=1-N}^{N}\left|z^\pi_{p+1}\left(\frac{r}{2^\vartheta}\right)-\rho_{z^\pi_{p+1}}\left(\frac{r}{2^\vartheta}\right)\right|\times$$

$$(|\mathbb{E}^x_p[\varphi_{N,r}(X^\pi_{t_{p+1}})]|+\mathbb{E}^x_p[\varphi_{N,r}(X^\pi_{t_{p+1}})\Delta W_{p+1}]|)$$

$$+\left(\frac{1}{\theta_2\Delta t}+\frac{(1-\theta_2)\mathscr{C}}{\theta_2}\right)\frac{1}{N}\sum_{r=1-N}^{N}\left|y^\pi_{p+1}\left(\frac{r}{2^\vartheta}\right)-\rho_{y^\pi_{p+1}}\left(\frac{r}{2^\vartheta}\right)\right||\mathbb{E}^x_p[\varphi_{N,r}(X^\pi_{t_{p+1}})\Delta W_{p+1}]|$$

Note that we have added some extra terms in the last inequality to simplify the expression. Taking the maximum over $\left\{\frac{(1-\theta_2)(1+\mathscr{C})}{\theta_2},\frac{1}{\theta_2\Delta t}+\frac{(1-\theta_2)\mathscr{C}}{\theta_2}\right\}$ finishes the proof for Equation (2.15).

Using Equations (2.14), (2.15), (2.6), we have

$$|\hat{y}^{\pi,I}_p-y^\pi_p(x)|$$

$$\leq\frac{1+\xi}{1-\xi}\varepsilon^{Picard}_p+\frac{\xi}{1-\xi}|\hat{z}^\pi_p(x)-z^\pi_p(x)|+\frac{1}{1-\xi}|\hat{h}(t_p,x)-h(t_p,x)|$$

$$\leq\frac{1+\xi}{1-\xi}\varepsilon^{Picard}_p+\frac{\mathscr{C}_1\cdot\xi}{1-\xi}(|\zeta^{\vartheta,N}_{z^\pi_{p+1}}(t_p,x)|+|\zeta^{\vartheta,N,W}_{y^\pi_{p+1}}(t_p,x)|+|\zeta^{\vartheta,N,W}_{f_{p+1}}(t_p,x)|)$$

$$+\frac{\mathscr{C}_1\cdot\xi}{1-\xi}\frac{1}{N}\sum_{r=1-N}^{N}\left|z^\pi_{p+1}\left(\frac{r}{2^\vartheta}\right)-\rho_{z^\pi_{p+1}}\left(\frac{r}{2^\vartheta}\right)\right|(|\mathbb{E}^x_p[\varphi_{N,r}(X^\pi_{t_{p+1}})]|+\mathbb{E}^x_p[\varphi_{N,r}(X^\pi_{t_{p+1}})\Delta W_{p+1}]|)$$

$$+\frac{\mathscr{C}_1\cdot\xi}{1-\xi}\frac{1}{N}\sum_{r=1-N}^{N}\left|y^\pi_{p+1}\left(\frac{r}{2^\vartheta}\right)-\rho_{y^\pi_{p+1}}\left(\frac{r}{2^\vartheta}\right)\right||\mathbb{E}^x_p[\varphi_{N,r}(X^\pi_{t_{p+1}})\Delta W_{p+1}]|$$

$$+\frac{1}{1-\xi}|\mathbb{E}^x_p[y^\pi_{p+1}(X^\pi_{t_{p+1}})]-\hat{\mathbb{E}}[y^\pi_{p+1}(X^\pi_{t_{p+1}})|X^\pi_{t_p}=x]|$$

$$+\frac{\Delta t(1-\theta_1)}{1-\xi}\times|\mathbb{E}^x_p[f(t_{p+1},X^\pi_{t_{p+1}},y^\pi_{p+1}(X^\pi_{t_{p+1}}),z^\pi_{p+1}(X^\pi_{t_{p+1}})]$$

$$-\hat{\mathbb{E}}[f(t_{p+1},X^\pi_{t_{p+1}},y^\pi_{p+1}(X^\pi_{t_{p+1}}),z^\pi_{p+1}(X^\pi_{t_{p+1}}))|X^\pi_{t_p}=x]|$$

With the standing assumption (A3), we may apply the error bounds in Equation (2.8), (2.10) and (2.12), and we derive:

$$|\hat{y}^{\pi,I}_p-y^\pi_p(x)|$$

$$\leq\frac{1+\xi}{1-\xi}\varepsilon^{Picard}_p+\frac{\mathscr{C}_1\cdot\xi}{1-\xi}(|\zeta^{\vartheta,N}_{z^\pi_{p+1}}(t_p,x)|+|\zeta^{\vartheta,N,W}_{y^\pi_{p+1}}(t_p,x)|+|\zeta^{\vartheta,N,W}_{f_{p+1}}(t_p,x)|)$$

$$+\frac{1}{1-\xi}|\zeta^{\vartheta,N}_{y^\pi_{p+1}}(t_p,x)|+\frac{\Delta t(1-\theta_1)}{1-\xi}|\zeta^{\vartheta,N}_{f_{p+1}}(t_p,x)|$$

$$+\frac{\mathscr{C}_1\cdot\xi}{1-\xi}\frac{1}{N}\sum_{r=1-N}^{N}\left|z^\pi_{p+1}\left(\frac{r}{2^\vartheta}\right)-\rho_{z^\pi_{p+1}}\left(\frac{r}{2^\vartheta}\right)\right|(|\mathbb{E}^x_p[\varphi_{N,r}(X^\pi_{t_{p+1}})]|+\mathbb{E}^x_p[\varphi_{N,r}(X^\pi_{t_{p+1}})\Delta W_{p+1}]|)$$

$$
\begin{aligned}
&+\frac{\mathscr{C}_1\cdot\xi}{1-\xi}\frac{1}{N}\sum_{r=1-N}^{N}\left|y_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right)-\rho_{y_{p+1}^{\pi}}\left(\frac{r}{2^{\vartheta}}\right)\right||\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^{\pi})\Delta W_{p+1}]|\\
&+\frac{1}{1-\xi}\frac{1}{N}\sum_{r=1-N}^{J}\left|y_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right)-\rho_{y_{p+1}^{\pi}}\left(\frac{r}{2^{\vartheta}}\right)\right||\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^{\pi})]|\\
&+\frac{\Delta t(1-\theta_1)}{1-\xi}\frac{1}{N}\sum_{r=1-N}^{J}\left|f\left(t_{p+1},\frac{r}{2^{\vartheta}},y_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right),z_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right)\right)-\rho_{f_{p+1}}\left(\frac{r}{2^{\vartheta}}\right)\right||\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^{\pi})]|\\
&\leq\frac{1+\xi}{1-\xi}\varepsilon_p^{Picard}+\frac{\mathscr{C}_1\cdot\xi}{1-\xi}(|\zeta_{z_{p+1}^{\pi}}^{\vartheta,N}(t_p,x)|+|\zeta_{y_{p+1}^{\pi}}^{\vartheta,N,W}(t_p,x)|+|\zeta_{f_{p+1}}^{\vartheta,N,W}(t_p,x)|)\\
&+\frac{1}{1-\xi}|\zeta_{y_{p+1}^{\pi}}^{\vartheta,N}(t_p,x)|+\frac{\Delta t(1-\theta_1)}{1-\xi}|\zeta_{f_{p+1}}^{\vartheta,N}(t_p,x)|\\
&+\frac{\mathscr{C}_1\cdot\xi+\mathscr{C}\Delta t(1-\theta_1))}{1-\xi}\frac{1}{N}\sum_{r=1-N}^{N}\left|z_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right)-\rho_{z_{p+1}^{\pi}}\left(\frac{r}{2^{\vartheta}}\right)\right|\times\\
&\qquad(|\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^{\pi})]|+\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^{\pi})\Delta W_{p+1}]|)\\
&+\frac{\mathscr{C}_1\cdot\xi+1+\mathscr{C}\Delta t(1-\theta_1)}{1-\xi}\frac{1}{N}\sum_{r=1-N}^{N}\left|y_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right)-\rho_{y_{p+1}^{\pi}}\left(\frac{r}{2^{\vartheta}}\right)\right|\times\\
&\qquad(|\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^{\pi})]|+\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^{\pi})\Delta W_{p+1}]|).
\end{aligned}
$$

This concludes the proof if $\mathscr{C}_2:=\frac{\mathscr{C}_1\cdot\xi+1+\mathscr{C}\Delta t(1-\theta_1)}{1-\xi}$ and $\mathscr{C}_3:=\frac{1+\xi}{1-\xi}\frac{\varepsilon_p^{Picard}}{\mathscr{C}_2}$. Again, extra terms are included in the expression to simplify the formula. □

A SIMPLE EXAMPLE FOR DERIVING THE ERROR FORMULA

In this subsection, we would use the result in Section 2.3.4 to derive an error formula for the approximation of Example 2 in Section 2.5.2. In addition to the parameters provided in Section 2.5.2, we let $\theta_1=0$, $\theta_2=1$ and $P=2$. Note that $P=2$ is merely used here to simplify our expression.

It is clear that driver function f and terminal function g satisfy all standing assumptions with the Lipschitz coefficient of f with respect to y and z being 0.4. We have $\Delta t=0.05$, $\xi=\varepsilon_p^{Picard}=0$ for $p=0,1$ in our setting. Using the derivation in Appendix 2.6, Equations (2.15) and (2.16) can be simplified as follows:

$$
\begin{aligned}
&\frac{1}{20}|\hat{z}_p^{\pi}(x)-z_p^{\pi}(x)|\\
\leq&|\zeta_{y_{p+1}^{\pi}}^{\vartheta,N,W}(t_p,x)|+\frac{1}{N}\sum_{r=1-N}^{N}\left|y_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right)-\rho_{y_{p+1}^{\pi}}\left(\frac{r}{2^{\vartheta}}\right)\right||\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^{\pi})\Delta W_{p+1}]|\\
&\frac{1}{1.02}|\hat{y}_p^{\pi,I}(x)-y_p^{\pi}(x)|\\
\leq&|\zeta_{y_{p+1}^{\pi}}^{\vartheta,N}(t_p,x)|+|\zeta_{f_{p+1}}^{\vartheta,N}(t_p,x)|\\
&+\frac{1}{N}\sum_{r=1-N}^{N}\left|z_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right)-\rho_{z_{p+1}^{\pi}}\left(\frac{r}{2^{\vartheta}}\right)\right|(|\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^{\pi})]|+\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^{\pi})\Delta W_{p+1}]|)
\end{aligned}
$$

$$+\frac{1}{N}\sum_{r=1-N}^{N}\left|y_{p+1}^{\pi}\left(\frac{r}{2^{\vartheta}}\right)-\rho_{y_{p+1}^{\pi}}\left(\frac{r}{2^{\vartheta}}\right)\right|(|\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^{\pi})]|+\mathbb{E}_p^x[\varphi_{N,r}(X_{t_{p+1}}^{\pi})\Delta W_{p+1}]|),$$

for $p=0,1$. Therefore, the error of applying the quick SWIFT scheme to the discretized system reads:

$$\begin{aligned}
&|\hat{y}_0^{\pi,I}(x_0)-y_1^{\pi}(x_0)|\\
\leq&1.02|\zeta_{y_1^{\pi}}^{\vartheta,N}(0,x_0)|+1.02|\zeta_{f_1}^{\vartheta,N}(0,x_0)|\\
&+\frac{1.02}{N}\sum_{r=1-N}^{N}\left|z_1^{\pi}\left(\frac{r}{2^{\vartheta}}\right)-\hat{z}_1^{\pi}\left(\frac{r}{2^{\vartheta}}\right)\right|(|\mathbb{E}_0^{x_0}[\varphi_{N,r}(X_{t_1}^{\pi})]|+\mathbb{E}_0^{x_0}[\varphi_{N,r}(X_{t_1}^{\pi})\Delta W_1]|)\\
&+\frac{1.02}{N}\sum_{r=1-N}^{N}\left|y_1^{\pi}\left(\frac{r}{2^{\vartheta}}\right)-\hat{y}_1^{\pi,I}\left(\frac{r}{2^{\vartheta}}\right)\right|(|\mathbb{E}_0^{x_0}[\varphi_{N,r}(X_{t_1}^{\pi})]|+\mathbb{E}_0^{x_0}[\varphi_{N,r}(X_{t_1}^{\pi})\Delta W_1]|)\\
\leq&1.02|\zeta_{y_1^{\pi}}^{\vartheta,N}(0,x_0)|+1.02|\zeta_{f_1}^{\vartheta,N}(0,x_0)|\\
&+\frac{1}{N}\sum_{r=1-N}^{N}\left(20.4\left|\zeta_{y_2^{\pi}}^{\vartheta,N,W}\left(t_1,\frac{r}{2^{\vartheta}}\right)\right|+1.02^2\left|\zeta_{y_2^{\pi}}^{\vartheta,N}\left(t_1,\frac{r}{2^{\vartheta}}\right)\right|+1.02^2\left|\zeta_{f_2}^{\vartheta,N}\left(t_1,\frac{r}{2^{\vartheta}}\right)\right|\right)\\
&\qquad(|\mathbb{E}_0^{x_0}[\varphi_{N,r}(X_{t_1}^{\pi})]|+\mathbb{E}_0^{x_0}[\varphi_{N,r}(X_{t_1}^{\pi})\Delta W_1]|).
\end{aligned}$$

Note that there is no recurring error at t_2 as we know the exact terminal condition.

REFERENCES

[1] K. W. Chau and C. W. Oosterlee, *On the wavelet-based SWIFT method for backward stochastic differential equations,* IMA Journal of Numerical Analysis **38**, 1051 (2018).

[2] L. Ortiz-Gracia and C. W. Oosterlee, *A highly efficient Shannon wavelet inverse Fourier technique for pricing european options,* SIAM Journal on Scientific Computing **38**, B118 (2016).

[3] S. C. Maree, L. Ortiz-Gracia, and C. W. Oosterlee, *Pricing early-exercise and discrete barrier options by shannon wavelet expansions,* Numerische Mathematik , 1 (2017).

[4] A. Iserles, *A First Course in the Numerical Analysis of Differential Equations*, 2nd ed. (Cambridge University Press, 2008) Cambridge Books Online.

[5] M. J. Ruijter and C. W. Oosterlee, *A Fourier cosine method for an efficient computation of solutions to BSDEs,* SIAM Journal on Scientific Computing **37**, A859 (2015).

[6] E. Pardoux and S. G. Peng, *Backward stochastic differential equations and quasilinear parabolic partial differential equations,* in *Stochastic Partial Differential Equations and Their Applications: Proceedings of IFIP WG 7/1 International Conference University of North Carolina at Charlotte, NC June 6–8, 1991*, edited by B. L. Rozovskii and R. B. Sowers (Springer Berlin Heidelberg, Berlin, Heidelberg, 1992) pp. 200–217.

[7] B. Fischer and J. Prestin, *Wavelets based on orthogonal polynomials,* Mathematics of Computation **66**, 1593 (1997).

[8] E. Gobet, J.-P. Lemor, and X. Warin, *A regression-based Monte Carlo method to solve backward stochastic differential equations,* The Annals of Applied Probability **15**, 2172 (2005).

[9] F. Fang and C. W. Oosterlee, *A novel pricing method for european options based on fourier-cosine series expansions,* SIAM Journal on Scientific Computing **31**, 826 (2009).

[10] S. Serra-Capizzano, *A note on antireflective boundary conditions and fast deblurring models,* SIAM Journal on Scientific Computing **25**, 1307 (2004).

[11] M. Donatelli and S. Serra-Capizzano, *Antireflective boundary conditions for deblurring problems,* Journal of Electrical and Computer Engineering **2010** (2010).

[12] W. Zhao, Y. Li, and G. Zhang, *A generalized θ-scheme for solving backward stochastic differential equations,* Discrete and Continuous Dynamical Systems - Series B **17**, 1585 (2012).

[13] F. Black and M. Scholes, *The pricing of options and corporate liabilities,* Journal of Political Economy **81**, 637 (1973).

[14] Y. Z. Bergman, *Option pricing with differential interest rates,* Review of Financial Studies **8**, 475 (1995).

[15] C. Bender and J. Steiner, *Least-squares Monte Carlo for backward SDEs,* in *Numerical Methods in Finance: Bordeaux, June 2010*, edited by R. A. Carmona, P. Del Moral, P. Hu, and N. Oudjane (Springer Berlin Heidelberg, Berlin, Heidelberg, 2012) pp. 257–289.

[16] W. Zhao, Y. Fu, and T. Zhou, *New kinds of high-order multistep schemes for coupled forward backward stochastic differential equations,* SIAM Journal on Scientific Computing **36**, A1731 (2014).

[17] A. Pinkus and S. Zafrany, *Fourier Series and Integral Transforms* (Cambridge University Press, 1997) Cambridge Books Online.

3

EXPLORATION OF A COSINE EXPANSION LATTICE SCHEME

In Chapter 2, we developed a Fourier-cosine expansion based wavelet scheme for one-dimensional BSDEs. Although we have shown that the SWIFT method is accurate and efficient in the previous chapter, it cannot be simply extended to a higher-dimensional problem with a tensor product extension. This trivial extension of the expectation computation in Equation (2.2) will lead to the curse of dimensionality. In this chapter, we design a higher-dimensional approximation scheme for expectation computation by combining a lattice sequence from Quasi-Monte Carlo rules with the philosophy of the Fourier-cosine method. The goal is to develop a Fourier-cosine based higher-dimensional scheme for expectations which can be applied to Equation (1.4).

3.1. INTRODUCTION

In this chapter, we derive a numerical integration formula for a function $f : \mathbb{R}^s \to \mathbb{R}$ with respect to a probability measure ν. Namely, we aim to approximate the following quantity:

$$\mathbb{E}[f(\mathbf{Y})] = \int_{\mathbb{R}^s} f(\mathbf{y})\nu(d\mathbf{y}). \tag{3.1}$$

The inspiration of this work comes from two types of promising methods in numerical integration, Fourier-cosine based methods and lattice rules.

There have been some attempts to extend the Fourier-cosine scheme beyond the one-dimensional situation, for example, in [1]. However, the tensor extension used in previous studies suffers from the curse of dimensionality, with the number of summation terms increasing exponentially when the number of dimensions increases. Therefore, further input is required for such extensions to be feasible in practice.

This chapter is based on the article 'Exploration of a cosine expansion lattice scheme', working paper.

Thus, we take the insight from Quasi-Monte Carlo (QMC) rules, which approximate integrals of the form

$$\int_{[0,1]^s} \mathfrak{f}(\mathbf{y})d\mathbf{y}$$

with equal weight quadrature rules

$$\frac{1}{N}\sum_{n=0}^{N-1} \mathfrak{f}(\mathbf{y}_n).$$

Readers are referred to [2], [3] and references therein for further details.

In particular, we are interested in the rank-1 lattice rule construction method for quadrature points, where for a given positive number N under dimension s, one chooses a vector $\mathbf{g} \in \{1,\dots,N-1\}^s$ called the generating vector and generates a point set:

$$\left(\left\{\frac{n\mathbf{g}}{N}\right\}\right)_{0\le n<N-1},$$

where the notation $\{\}$ denotes the fractional part of the real number in each dimension. The quadrature points are the result of applying some fixed function on this point set.

There have been numerous research papers supporting the implementation of lattice rule QMC, in identifying a suitable generating vector [4–7], in efficient algorithms to generate such vectors [8, 9], and in extensible lattice rules [10–13]. In [14], the authors derived an error bound for QMC with tent transformed lattice rules for functions within the half-period cosine space. Noting the connection between the half-period cosine space and the tensor extension of the cosine series, we are inspired to combine the two approaches and transfer the rich results from the lattice literature to Fourier expansion schemes.

This work is organized as follows. In Section 3.2, we present the two components of the cosine expansion lattice scheme, the half-period cosine expansion and the tent transformed lattice. We also present the full scheme in this section. In Section 3.3, we give further details regarding our current results by providing an alternative formation of its error bound and connecting the scheme to the periodic wavelets introduced in Chapter 2. Numerical experiments are presented in Section 3.4 and we conclude our findings in Section 3.5.

Before we begin, we mention some conventions used in this work. We assume all the integrals in the computation to be finite, therefore Fubini's theorem can be applied and we exchange the order of integration without notice. The operations $\times$ and $/$ act component-wise when used on vectors.

Finally, some notation we use in this chapter:

- The natural number set $\mathbb{N} := \{0,1,2,\dots\}$;
- The positive integer set $\mathbb{Z}^+ := \{i \in \mathbb{Z} | i > 0\}$, similarly for $\mathbb{R}^+$;
- The truncated integer set, for $s \in \mathbb{Z}^+$ we write $[s] := \{1,\dots,s\}$;
- The indicator function $\mathbf{1}_{\mathscr{D}} : \mathbb{R}^s \to \{0,1\}$

$$\mathbf{1}_{\mathscr{D}}(\mathbf{y}) = \begin{cases} 1, & \text{if } \mathbf{y} \in \mathscr{D}; \\ 0, & \text{otherwise;} \end{cases}$$

- The ceil function $\lceil\cdot\rceil : \mathbb{R} \to \mathbb{Z}$, $\lceil y \rceil = \min\{i \in \mathbb{Z} | i \geq y\}$.

3.2. COSINE EXPANSION LATTICE SCHEME

In this section, we introduce the cosine expansion lattice scheme, whose construction consists of two parts: a projection of the original integrand on a reproducing kernel space and a numerical integration technique based on a tent-transformed lattice rule. We will briefly describe the intuition behind our derivation.

3.2.1. THE HALF-PERIOD COSINE SPACE

In a similar framework as the Fourier-cosine method from [15], we would like to define a cosine based periodic expansion of function $\mathfrak{f}$ in $\mathbb{R}^s$. This allows us to connect the expectation problem (3.1) to a Fourier transform.

The common practice to transform Equation (3.1) into a finite problem for computational purposes is restricting the domain of integration to a predefined box $\mathscr{D} := [a_1, b_1] \times \cdots \times [a_s, b_s]$. This step is justified as long as $\mathscr{D}$ contains the majority of the mass of the measure. In fact, this is equivalent to replacing the original integrand $\mathfrak{f}(\mathbf{y})$ by $\mathfrak{f}(\mathbf{y})\mathbf{1}_{\mathscr{D}}(\mathbf{y})$, or setting the function values outside $\mathscr{D}$ to zero.

The COS method, a well-known example of the Fourier-cosine schemes, uses this concept to its advantage by replacing an originally non-periodic one-dimensional function $\mathfrak{f}$ by a cosine series projection that coincides with $\mathfrak{f}$ on the domain $[a, b]$, but is periodic throughout the whole real line.

To be precise, the integrand $\mathfrak{f}$ is replaced by[1]

$$\bar{\mathfrak{f}}(y) := {\sum_{k=1}^{N-1}}' \widetilde{\mathfrak{f}}_{\cos}(k) \cos\left(k\pi \frac{y-a}{b-a}\right),$$

where

$$\widetilde{\mathfrak{f}}_{\cos}(k) := \frac{2}{b-a} \int_a^b \mathfrak{f}(y) \cos\left(k\pi \frac{y-a}{b-a}\right) dx.$$

In this case, the original 1-D expectation is approximated by

$$\begin{aligned}
\mathbb{E}[\mathfrak{f}(Y)] &\approx \int_{\mathbb{R}} {\sum_{k=1}^{N-1}}' \widetilde{\mathfrak{f}}_{\cos}(k) \cos\left(k\pi \frac{y-a}{b-a}\right) \nu(dy) \\
&= {\sum_{k=1}^{N-1}}' \widetilde{\mathfrak{f}}_{\cos}(k) \int_{\mathbb{R}} \cos\left(k\pi \frac{y-a}{b-a}\right) \nu(dy) \\
&= {\sum_{k=1}^{N-1}}' \widetilde{\mathfrak{f}}_{\cos}(k) \Re\left\{ \mathscr{F}\left(\frac{k\pi}{b-a}\right) \exp\left(-\imath k\pi \frac{a}{b-a}\right) \right\}.
\end{aligned}$$

Note that within $[a, b]$, $\bar{\mathfrak{f}}$ is a projection of $\mathfrak{f}$ onto a finite cosine series space and $\bar{\mathfrak{f}}$ converges to $\mathfrak{f}$ in the $L^2([a, b])$ norm, when N tends to infinity. However, when considering the function $\bar{\mathfrak{f}}$ on the whole domain $\mathbb{R}$, it is a periodic function and it thus deviates from

[1] The notation $\sum'$ denotes the first term of the summation is weighted by one half.

f itself. Once again, the error of this transformation is kept under control as the probability mass outside $[a,b]$ is small and we make use of the Fourier transform $\mathscr{F}$, which is typically available, in the approximation.

The main goal of this work is to apply the same idea in a higher-dimensional setting. Thus, we will consider a functional space built on cosine functions and use the series projection of f onto such space as our replacement integrand. The space we pick is a modification to the half-period cosine space introduced in Section 2.3 of [14] and their article serves as an inspiration of the current work.

We define an inner product $<\cdot,\cdot>_{\mathscr{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}$ as

$$<\mathfrak{f},\mathfrak{g}>_{\mathscr{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}=\sum_{\mathbf{k}\in\mathbb{N}^s}\widetilde{\mathfrak{f}}_{\cos}(\mathbf{k})\widetilde{\mathfrak{g}}_{\cos}(\mathbf{k})\mathfrak{d}^{-1}_{\alpha,\boldsymbol{\gamma},s}(\mathbf{k}),$$

for some real number $\alpha>1/2$ and vector $\boldsymbol{\gamma}\in(\mathbb{R}^+)^s$, where the multi-dimensional cosine coefficients are given by

$$\widetilde{\mathfrak{f}}_{\cos}(\mathbf{k}):=\int_{[0,1]^s}\mathfrak{f}(\mathbf{y}\times(\mathbf{b}-\mathbf{a})+\mathbf{a})2^{|\mathbf{k}|_0/2}\prod_{j=1}^{s}\cos(\pi k_j y_j)d\mathbf{y}, \tag{3.2}$$

for $\mathbf{k}=(k_1,\cdots,k_s)\in\mathbb{N}^s$. Here we define $|\mathbf{k}|_0:=\#\{j\in[s]:k_j\neq 0\}$ to be the number of non-zero components in $\mathbf{k}$. Note that only the portion of f within the predefined domain $\mathscr{D}$ is used here.

For $\alpha>1/2, k\in\mathbb{Z}$ and $\gamma>0$, the one-dimensional $\mathfrak{d}$ function is defined as:

$$\mathfrak{d}_{\alpha,\gamma}(k):=\begin{cases}1 & \text{if } k=0;\\ \gamma|k|^{-2\alpha} & \text{if } k\neq 0,\end{cases}$$

and the multidimensional $\mathfrak{d}$ function is set as,

$$\mathfrak{d}_{\alpha,\boldsymbol{\gamma},s}(\mathbf{k}):=\prod_{j=1}^{s}\mathfrak{d}_{\alpha,\gamma_j}(k_j),$$

for $\mathbf{k}=(k_1,\ldots,k_s)\in\mathbb{Z}^s$ and $\boldsymbol{\gamma}=(\gamma_1,\ldots,\gamma_s)\in(\mathbb{R}^+)^s$. The $\mathfrak{d}$ function is introduced to the norm here to assess the decay rate of the cosine coefficients, as it is closely related to the approximation error.

We define the corresponding norm as $\sqrt{<\mathfrak{f},\mathfrak{f}>_{\mathscr{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}}$ and denote it by $||\mathfrak{f}||_{\mathscr{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}$. In particular, we have

$$||\mathfrak{f}||^2_{\mathscr{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}=\sum_{\mathbf{k}\in\mathbb{N}^s}\frac{|\widetilde{\mathfrak{f}}_{\cos}(\mathbf{k})|^2}{\mathfrak{d}_{\alpha,\gamma,s}(\mathbf{k})}=\sum_{\mathbf{h}\in\mathbb{Z}^s}2^{-|\mathbf{h}|_0}\frac{|\widetilde{\mathfrak{f}}_{\cos}(\mathbf{h})|^2}{\mathfrak{d}_{\alpha,\gamma,s}(\mathbf{h})}.$$

The dummy variable is changed from $\mathbf{k}$ to $\mathbf{h}$ here in preparation for future computations. We denote any function f such that $||\mathfrak{f}||_{\mathscr{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}<\infty$ as $\mathfrak{f}\in\mathscr{H}_{\mathscr{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}$.

The half-period cosine space is an example of a reproducing kernel Hilbert space with the corresponding reproducing kernel,

$$\mathscr{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D},\mathbf{x},\mathbf{y}):=\prod_{j=1}^{s}\left(\sum_{k_j\in\mathbb{Z}}\mathfrak{d}_{\alpha,\gamma_j}(k_j)\cos\left(\pi k_j\frac{x_j-a_j}{b_j-a_j}\right)e^{\imath\pi k_j\frac{y_j-a_j}{b_j-a_j}}\right)$$

$$= \sum_{\mathbf{k}\in\mathbb{Z}^s} \mathfrak{d}_{\alpha,\boldsymbol{\gamma},s}(\mathbf{k}) \left(\prod_{j=1}^{s} \cos\left(\pi k_j \frac{x_j - a_j}{b_j - a_j}\right) \right) e^{\imath\pi\mathbf{k}\cdot\frac{\mathbf{y}-\mathbf{a}}{\mathbf{b}-\mathbf{a}}},$$

and for the one-dimensional version,

$$\begin{aligned}
\mathcal{K}^{\cos}_{\alpha,\gamma}(\mathscr{D}, x, y) :=& 1 + \sum_{k=1}^{\infty} \mathfrak{d}_{\alpha,\gamma}(k)\sqrt{2}\cos\left(\pi k\frac{x-a}{b-a}\right)\sqrt{2}\cos\left(\pi k\frac{y-a}{b-a}\right) \\
=& 1 + \sum_{k=1}^{\infty} \mathfrak{d}_{\alpha,\gamma}(k)\cos\left(\pi k\frac{x-a}{b-a}\right)\left(e^{\imath\pi k\frac{y-a}{b-a}} + e^{-\imath\pi k\frac{y-a}{b-a}}\right) \\
=& \sum_{k\in\mathbb{Z}} \mathfrak{d}_{\alpha,\gamma}(k)\cos\left(\pi k\frac{x-a}{b-a}\right) e^{\imath\pi k\frac{y-a}{b-a}}.
\end{aligned}$$

For any $\mathbf{y} \in \mathscr{D}$ and $\mathfrak{f} \in \mathscr{H}_{\mathcal{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}$, we have the reproducing property,

$$\mathfrak{f}(\mathbf{y}) = < \mathfrak{f}, \mathcal{K}^{\cos}_{\mathscr{D},\alpha,\boldsymbol{\gamma},s}(\cdot, y) >_{\mathcal{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}.$$

Readers are referred to [16] and [2] for further information on reproducing kernel Hilbert spaces.

In this work, we use an alternative kernel which drops the $\mathfrak{d}$ function. It is defined as

$$\mathcal{K}^{\cos}_{s}(\mathbf{x},\mathbf{y}) := \prod_{j=1}^{s}\left(\sum_{k_j\in\mathbb{Z}} \cos\left(\pi k_j\frac{x_j-a_j}{b_j-a_j}\right) e^{\imath\pi k_j\frac{y_j-a_j}{b_j-a_j}}\right) = \sum_{\mathbf{k}\in\mathbb{Z}^s}\left(\prod_{j=1}^{s}\cos\left(\pi k_j\frac{x_j-a_j}{b_j-a_j}\right)\right) e^{\imath\pi\mathbf{k}\cdot\frac{\mathbf{y}-\mathbf{a}}{\mathbf{b}-\mathbf{a}}}.$$

We suppress the $\mathscr{D}$ part here to simplify our notation.

Using the reproducing property, we have the following equation:

$$\mathfrak{f}(\mathbf{y}) = \sum_{\mathbf{k}\in\mathbb{N}^s} \widetilde{\mathfrak{f}}_{\cos}(\mathbf{k})(\sqrt{2})^{|\mathbf{k}|_0} \prod_{j=1}^{s}\cos\left(\pi k_i\frac{y_i-a_i}{b_i-a_i}\right), \tag{3.3}$$

for any $\mathfrak{f} \in \mathscr{H}_{\mathcal{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}$ and $\mathbf{y} \in \mathscr{D}$. We use the expansion at the right-hand side of (3.3) as the replacement integrand in Equation (3.1), denoted by $\bar{\mathfrak{f}}$.

Definition 3.1 (Half-period Cosine Expansion)**.** For any given function $\mathfrak{f} : \mathbb{R}^s \to \mathbb{R}$ and given domain $\mathscr{D} \subset \mathbb{R}^s$, the half-period cosine expansion, $\bar{\mathfrak{f}} : \mathbb{R}^s \to \mathbb{R}$, is defined as

$$\bar{\mathfrak{f}}(\mathbf{y}) := \sum_{\mathbf{k}\in\mathbb{N}^s} \widetilde{\mathfrak{f}}_{\cos}(\mathbf{k})(\sqrt{2})^{|\mathbf{k}|_0} \prod_{j=1}^{s}\cos\left(\pi k_i\frac{y_i-a_i}{b_i-a_i}\right),$$

where the cosine coefficients are defined as in Equation 3.2.

3.2.2. LATTICE RULE APPROXIMATIONS

There are essentially two drawbacks when extending the COS method to higher dimensions. First, the cosine coefficient $\widetilde{\mathfrak{f}}_{\cos}$ may be difficult to calculate, especially in a recurring situation. In Chapter 2, we aimed to remedy this shortcoming by adopting a wavelet

basis such that we can approximate Equation (3.1) as a weighted sum of local values, in the form of

$$\mathbb{E}[\mathfrak{f}(Y)] \approx \frac{1}{N} \sum_{n=1-N}^{N} \mathfrak{f}\left(\frac{r}{2^{\vartheta}}\right) \mathbb{E}[\varphi_{N,n}(Y)],$$

with φ being the wavelet basis functions. We will compare our current scheme to that work in Section 3.3.2.

The second point of concern is that if we simply apply the tensor product of cosine spaces in a higher-dimensional case, we will face the curse of dimensionality.

Here, we aim to address the above two issues by constructing quadrature rules for the integration of

$$\int_{\mathbb{R}^s} \bar{\mathfrak{f}}(\mathbf{y}) \nu(d\mathbf{y}). \tag{3.4}$$

In particular, the approximant takes the specific form,

$$\int_{\mathbb{R}^s} \frac{1}{N} \sum_{n=0}^{N-1} \mathfrak{f}(\mathrm{p}_n) \mathscr{K}_s^{\cos}(\mathrm{p}_n, \mathbf{y}) \nu(d\mathbf{y}), \tag{3.5}$$

where $\mathscr{P} := \{\mathrm{p}_0, \ldots, \mathrm{p}_{N-1}\}$ is some predetermined quadrature points set.

For the half-period cosine space that we adopted in the previous section, Dick et. al. showed in [14] that a combination of rank-1 lattice rules and tent transformations converges well in the context of quasi-Monte Carlo methods. We will now introduce the quadrature points proposed in [14] for the half-period cosine space, which we should adapt for the approximation of Equation (3.5).

A lattice point set with $N \geq 2$ points and generating vector $\mathbf{g} \in [N-1]^s$ is given by

$$\mathscr{P}(\mathbf{g}, N) := \left\{ \left\{ \frac{n\mathbf{g}}{N} \right\} : 0 \leq n < N \right\}.$$

The *tent transformation*, $\phi : [0,1] \to [0,1]$ is defined as

$$\phi(x) := 1 - |2x - 1|,$$

and the higher-dimensional version $\boldsymbol{\phi} : [0,1]^s \to [0,1]^s$ is obtained by applying the function component-wise. The tent-transformed lattice point set in [14] is given by

$$\mathscr{P}_{\boldsymbol{\phi}}(\mathbf{g}, N) := \left\{ \boldsymbol{\phi}\left(\left\{ \frac{n\mathbf{g}}{N} \right\} \right) : 0 \leq n < N \right\}.$$

However, since we have to apply the quadrature points on a general box $\mathscr{D}$, instead of just on $[0,1]^s$, we have to transform the lattice rules:

$$\mathscr{P}_{\boldsymbol{\phi}}(\mathbf{g}, N, \mathscr{D}) = \{\mathrm{p}_{\boldsymbol{\phi},0}, \cdots, \mathrm{p}_{\boldsymbol{\phi},N-1}\} := \left\{ \boldsymbol{\phi}\left(\left\{ \frac{n\mathbf{g}}{N} \right\} \right) \times (\mathbf{b} - \mathbf{a}) + \mathbf{a} : 0 \leq n < N \right\}.$$

We wish to quantify the integration error of applying Equation (3.5) to approximate the expectation in the form of equation (3.4) for functions in the half-period cosine space $\mathscr{H}_{\mathscr{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}$. In addition to the above condition, we also enforce some convergence requirements on the cosine transform of the measure ν.

Theorem 3.2. *Consider any function* $\mathfrak{f} \in \mathcal{H}_{\mathcal{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}$ *and any probability measure* ν *such that*

$$\left(\int_{\mathbb{R}^s}\left(\prod_{j=1}^{s}\cos\left(\pi k_j\frac{y_j-a_j}{b_j-a_j}\right)\right)\nu(d\mathbf{y})\right)^2 \le \mathfrak{d}_{\beta+0.5+\delta,\boldsymbol{\rho},s}(\mathbf{k}),$$

for some real number $\beta > s$*, small positive number* δ *and vector* $\boldsymbol{\rho} \in (\mathbb{R}^+)^s$*. Therefore,*

$$\sum_{\mathbf{k}\in\mathbb{N}^s}\left(\int_{\mathbb{R}^s}\left(\prod_{j=1}^{s}\cos\left(\pi k_j\frac{y_j-a_j}{b_j-a_j}\right)\right)\nu(d\mathbf{y})\right)^2/\mathfrak{d}_{\beta,\boldsymbol{\rho},s}(\mathbf{k}) \le \mathscr{M}^2,$$

for some positive constant $\mathscr{M} \le \infty$*. Furthermore, we assume that* $\beta - \alpha > 1/2$*, namely, the cosine transform of the measure* ν *decays at least algebraically and quicker than the cosine coefficients of* $\mathfrak{f}$*. The error for the numerical integration using the tent-transformed lattice rule on* $\mathscr{D}$ *is then bounded by*

$$\begin{aligned}\epsilon_2(\mathfrak{f},\nu,\mathscr{P}_{\boldsymbol{\phi}}(\mathbf{g},N,\mathscr{D})) &:= \left|\int_{\mathbb{R}^s}\frac{1}{N}\sum_{n=0}^{N-1}\tilde{\mathfrak{f}}\left(\mathrm{p}_{\boldsymbol{\phi},n}\right)\mathcal{K}_s^{\cos}\left(\mathrm{p}_{\boldsymbol{\phi},n},\mathbf{y}\right)\nu(d\mathbf{y}) - \int_{\mathbb{R}^s}\tilde{\mathfrak{f}}(\mathbf{y})\nu(d\mathbf{y})\right| \\ &\le \left(\sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}}\mathfrak{d}_{\alpha,\boldsymbol{\gamma},s}(\mathbf{h})\right)^{\frac{1}{2}}\left(\prod_{i=1}^{s}\mathscr{C}_i\right)^{\frac{1}{2}}\mathscr{M}\|\mathfrak{f}\|_{\mathcal{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}\end{aligned}$$

for some constant $\mathscr{C}_i$*, where* $L^\perp := \{\mathbf{h}\in\mathbb{Z}^s : \mathbf{h}\cdot\mathbf{g} \equiv 0 \pmod N\}$ *is the dual lattice.*

Proof. The general flow of this proof follows the proof of Theorem 2 in [14]. Let $\mathfrak{f} \in \mathcal{H}_{\mathcal{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}$ and $\tilde{\mathfrak{f}}$ be its half-period cosine expansion.

For any $k \in \mathbb{N}$, we have

$$\cos(\pi k\phi(y)) = \cos(2\pi k y) \text{ for all } y \in [0,1],$$

and hence

$$\begin{aligned}\tilde{\mathfrak{f}}\left(\mathrm{p}_{\boldsymbol{\phi},n}\right) &= \sum_{\mathbf{k}\in\mathbb{N}^s}\widetilde{\mathfrak{f}}_{\cos}(\mathbf{k})(\sqrt{2})^{|\mathbf{k}|_0}\prod_{j=1}^{s}\cos\left(\pi k_j\phi\left(\left\{\frac{ng_j}{N}\right\}\right)\right) \\ &= \sum_{\mathbf{k}\in\mathbb{N}^s}\widetilde{\mathfrak{f}}_{\cos}(\mathbf{k})(\sqrt{2})^{|\mathbf{k}|_0}\prod_{j=1}^{s}\cos\left(2\pi k_j\frac{ng_j}{N}\right) \\ &= \sum_{\mathbf{h}\in\mathbb{Z}^s}(\sqrt{2})^{-|\mathbf{h}|_0}\widetilde{\mathfrak{f}}_{\cos}(\mathbf{h})e^{2\pi\imath n(\mathbf{h}\cdot\mathbf{g})/N}.\end{aligned}$$

The reproducing kernel can be rewritten as

$$\mathcal{K}_s^{\cos}\left(\mathrm{p}_{\boldsymbol{\phi},n},\mathbf{y}\right) = \sum_{\mathbf{k}\in\mathbb{Z}^s}\left(\prod_{j=1}^{s}\cos\left(\pi k_j\frac{y_j-a_j}{b_j-a_j}\right)\right)e^{2\pi\imath n(\mathbf{k}\cdot\mathbf{g})/N}.$$

Therefore, we obtain

$$\int_{\mathbb{R}^s}\frac{1}{N}\sum_{n=0}^{N-1}\tilde{\mathfrak{f}}\left(\mathrm{p}_{\boldsymbol{\phi},n}\right)\mathcal{K}_s^{\cos}\left(\mathrm{p}_{\boldsymbol{\phi},n},\mathbf{y}\right)\nu(d\mathbf{y})$$

$$= \int_{\mathbb{R}^s} \frac{1}{N} \sum_{n=0}^{N-1} \sum_{\mathbf{h}\in\mathbb{Z}^s} (\sqrt{2})^{-|\mathbf{h}|_0} \widetilde{\mathfrak{f}}_{\cos}(\mathbf{h}) e^{2\pi\imath n(\mathbf{h}\cdot\mathbf{g})/N} \sum_{\mathbf{k}\in\mathbb{Z}^s} \left(\prod_{j=1}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right) e^{2\pi\imath n(\mathbf{k}\cdot\mathbf{g})/N} \nu(d\mathbf{y})$$

$$= \int_{\mathbb{R}^s} \sum_{\mathbf{h}\in\mathbb{Z}^s} \sum_{\mathbf{k}\in\mathbb{Z}^s} (\sqrt{2})^{-|\mathbf{h}|_0} \widetilde{\mathfrak{f}}_{\cos}(\mathbf{h}) \left(\prod_{j=1}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right) \frac{1}{N} \sum_{n=0}^{N-1} e^{2\pi\imath n[(\mathbf{h}+\mathbf{k})\cdot\mathbf{g}]/N} \nu(d\mathbf{y}).$$

The sum $\frac{1}{N}\sum_{n=0}^{N-1} e^{2\pi\imath n[(\mathbf{h}+\mathbf{k})\cdot\mathbf{g}]/N}$ is a character sum over the group $\mathbb{Z}/N\mathbb{Z}$, which is equal to one if $(\mathbf{h}+\mathbf{k})\cdot\mathbf{g}$ is a multiple of N and zero otherwise. From this, we get

$$\begin{aligned} &\int_{\mathbb{R}^s} \frac{1}{N} \sum_{n=0}^{N-1} \bar{\mathfrak{f}}\left(\mathrm{p}_{\boldsymbol{\phi},n}\right) \mathscr{K}_s^{\cos}\left(\mathrm{p}_{\boldsymbol{\phi},n}, \mathbf{y}\right) \nu(d\mathbf{y}) - \int_{\mathbb{R}^s} \bar{\mathfrak{f}}(\mathbf{y}) \nu(d\mathbf{y}) \\ &= \int_{\mathbb{R}^s} \sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}} \sum_{\mathbf{k}\in\mathbb{Z}^s} (\sqrt{2})^{-|\mathbf{h}-\mathbf{k}|_0} \widetilde{\mathfrak{f}}_{\cos}(\mathbf{h}-\mathbf{k}) \left(\prod_{j=1}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right) \nu(d\mathbf{y}). \end{aligned} \tag{3.6}$$

Based on this formula and an application of the Cauchy-Schwarz inequality, we obtain

$$\begin{aligned} &\left| \int_{\mathbb{R}^s} \frac{1}{N} \sum_{n=0}^{N-1} \bar{\mathfrak{f}}\left(\mathrm{p}_{\boldsymbol{\phi},n}\right) \mathscr{K}_s^{\cos}\left(\mathrm{p}_{\boldsymbol{\phi},n}, \mathbf{y}\right) \nu(d\mathbf{y}) - \int_{\mathbb{R}^s} \bar{\mathfrak{f}}(\mathbf{y}) \nu(d\mathbf{y}) \right| \\ &= \left| \sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}} \sum_{\mathbf{k}\in\mathbb{Z}^s} (\sqrt{2})^{-|\mathbf{h}-\mathbf{k}|_0} \widetilde{\mathfrak{f}}_{\cos}(\mathbf{h}-\mathbf{k}) \int_{\mathbb{R}^s} \left(\prod_{j=1}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right) \nu(d\mathbf{y}) \right| \\ &\le \left(\sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}} \sum_{\mathbf{k}\in\mathbb{Z}^s} \frac{2^{-|\mathbf{h}-\mathbf{k}|_0} |\widetilde{\mathfrak{f}}_{\cos}(\mathbf{h}-\mathbf{k})|^2}{\mathfrak{d}_{\alpha,\boldsymbol{\gamma},s}(\mathbf{h}-\mathbf{k})} \frac{\left(\int_{\mathbb{R}^s} \left(\prod_{j=1}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right) \nu(d\mathbf{y})\right)^2}{\mathfrak{d}_{\beta,\boldsymbol{\rho},s}(\mathbf{k})} \right)^{\frac{1}{2}} \times \\ &\quad \left(\sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}} \sum_{\mathbf{k}\in\mathbb{Z}^s} \mathfrak{d}_{\alpha,\boldsymbol{\gamma},s}(\mathbf{h}-\mathbf{k}) \mathfrak{d}_{\beta,\boldsymbol{\rho},s}(\mathbf{k}) \right)^{\frac{1}{2}} \\ &\le \left(\sum_{\mathbf{k}\in\mathbb{Z}^s} \left(\sum_{\mathbf{h}\in\mathbb{Z}^s} \frac{2^{-|\mathbf{h}-\mathbf{k}|_0} |\widetilde{\mathfrak{f}}_{\cos}(\mathbf{h}-\mathbf{k})|^2}{\mathfrak{d}_{\alpha,\boldsymbol{\gamma},s}(\mathbf{h}-\mathbf{k})} \right) \frac{\left(\int_{\mathbb{R}^s} \left(\prod_{j=1}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right) \nu(d\mathbf{y})\right)^2}{\mathfrak{d}_{\beta,\boldsymbol{\rho},s}(\mathbf{k})} \right)^{\frac{1}{2}} \times \\ &\quad \left(\sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}} \sum_{\mathbf{k}\in\mathbb{Z}^s} \mathfrak{d}_{\alpha,\boldsymbol{\gamma},s}(\mathbf{h}-\mathbf{k}) \mathfrak{d}_{\beta,\boldsymbol{\rho},s}(\mathbf{k}) \right)^{\frac{1}{2}} \\ &\le \|\mathfrak{f}\|_{\mathscr{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})} \left(\sum_{\mathbf{k}\in\mathbb{Z}^s} \frac{\left(\int_{\mathbb{R}^s} \left(\prod_{j=1}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right) \nu(d\mathbf{y})\right)^2}{\mathfrak{d}_{\beta,\boldsymbol{\rho},s}(\mathbf{k})} \right)^{\frac{1}{2}} \times \\ &\quad \left(\sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}} \sum_{\mathbf{k}\in\mathbb{Z}^s} \mathfrak{d}_{\alpha,\boldsymbol{\gamma},s}(\mathbf{h}-\mathbf{k}) \mathfrak{d}_{\beta,\boldsymbol{\rho},s}(\mathbf{k}) \right)^{\frac{1}{2}} \\ &\le \|\mathfrak{f}\|_{\mathscr{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})} \mathscr{M} \left(\sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}} \sum_{\mathbf{k}\in\mathbb{Z}^s} \mathfrak{d}_{\alpha,\boldsymbol{\gamma},s}(\mathbf{h}-\mathbf{k}) \mathfrak{d}_{\beta,\boldsymbol{\rho},s}(\mathbf{k}) \right)^{\frac{1}{2}} \end{aligned}$$

$$=:\|f\|_{\mathcal{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}\left(\sum_{\mathbf{h}\in L^{\perp}\setminus\{\mathbf{0}\}}\mathscr{I}(\mathbf{h})\right)^{\frac{1}{2}}. \tag{3.7}$$

The calculation above also makes use of the smoothness assumption on the probability measure.

Next, from direct calculation, we have

$$\begin{aligned}\mathscr{I}(\mathbf{h}) &= \sum_{\mathbf{k}\in\mathbb{Z}^s}\mathfrak{d}_{\alpha,\boldsymbol{\gamma},s}(\mathbf{h}-\mathbf{k})\mathfrak{d}_{\beta,\boldsymbol{\rho},s}(\mathbf{k})\\ &= \sum_{\mathbf{k}\in\mathbb{Z}^s}\prod_{j=1}^{s}\mathfrak{d}_{\alpha,\gamma_j}(h_j-k_j)\mathfrak{d}_{\beta,\rho_j}(k_j)\\ &= \prod_{j=1}^{s}\sum_{k_j=-\infty}^{\infty}\mathfrak{d}_{\alpha,\gamma_j}(h_j-k_j)\mathfrak{d}_{\beta,\rho_j}(k_j) =: \prod_{j=1}^{s}\mathscr{I}_j(h_j).\end{aligned} \tag{3.8}$$

In order to control the error, we need to control the sum $\mathscr{I}_j$ in Equation (3.8) for all $h_j\in\mathbb{Z}$. We have three different cases to consider.

Case 1: $h_j=0$. In this case,

$$\mathscr{I}_j(0)=\sum_{k_j=-\infty}^{\infty}\mathfrak{d}_{\alpha,\gamma_j}(-k_j)\mathfrak{d}_{\beta,\rho_j}(k_j)=1+2\gamma_j\rho_j\sum_{k_j=1}^{\infty}\frac{1}{(k_j)^{2\alpha+2\beta}}=1+2\gamma_j\rho_j\zeta(2\alpha+2\beta),$$

in which ζ denotes the Riemann zeta function.

Case 2: $h_j>0$. In this case,

$$\begin{aligned}\mathscr{I}_j(h_j) &= \sum_{k_j=-\infty}^{\infty}\mathfrak{d}_{\alpha,\gamma_j}(h_j-k_j)\mathfrak{d}_{\beta,\rho_j}(k_j)\\ &= \sum_{k_j=1}^{\infty}\mathfrak{d}_{\alpha,\gamma_j}(h_j+k_j)\mathfrak{d}_{\beta,\rho_j}(k_j)+\mathfrak{d}_{\alpha,\gamma_j}(h_j)+\sum_{k_j=1}^{h_j-1}\mathfrak{d}_{\alpha,\gamma_j}(h_j-k_j)\mathfrak{d}_{\beta,\rho_j}(k_j)+\mathfrak{d}_{\beta,\rho_j}(h_j)\\ &\quad+\sum_{k_j=1}^{\infty}\mathfrak{d}_{\alpha,\gamma_j}(k_j)\mathfrak{d}_{\beta,\rho_j}(h_j+k_j)\\ &\le \gamma_j|h_j|^{-2\alpha}\sum_{k_j=1}^{\infty}\rho_j|k_j|^{-2\beta}+\mathfrak{d}_{\alpha,\gamma_j}(h_j)+\sum_{k_j=1}^{h_j-1}\gamma_j|h_j-k_j|^{-2\alpha}\rho_j|k_j|^{-2\beta}+\frac{\rho_j}{\gamma_j}\mathfrak{d}_{\alpha,\gamma_j}(h_j)\\ &\quad+\frac{\rho_j}{\gamma_j}\mathfrak{d}_{\alpha,\gamma_j}(h_j)\sum_{k_j=1}^{\infty}\mathfrak{d}_{\alpha,\gamma_j}(k_j)\\ &\le \rho_j\zeta(2\beta)\mathfrak{d}_{\alpha,\gamma_j}(h_j)+\mathfrak{d}_{\alpha,\gamma_j}(h_j)+\gamma_j|h_j|^{-2\alpha}2^{2\alpha}\sum_{k_j=1}^{h_j-1}\rho_j|k_j|^{-2(\beta-\alpha)}+\frac{\rho_j}{\gamma_j}\mathfrak{d}_{\alpha,\gamma_j}(h_j)\\ &\quad+\rho_j\mathfrak{d}_{\alpha,\gamma_j}(h_j)\sum_{k_j=1}^{\infty}|k_j|^{-2\alpha}\end{aligned}$$

$$\leq \rho_j \zeta(2\beta)\mathfrak{d}_{\alpha,\gamma_j}(h_j) + \mathfrak{d}_{\alpha,\gamma_j}(h_j) + 2^{2\alpha}\rho_j\zeta(2(\beta-\alpha))\mathfrak{d}_{\alpha,\gamma_j}(h_j) + \frac{\rho_j}{\gamma_j}\mathfrak{d}_{\alpha,\gamma_j}(h_j) + \rho_j\zeta(2\alpha)\mathfrak{d}_{\alpha,\gamma_j}(h_j).$$

The above inequality simply follows from the definition and the orderings $|h_j+k_j|^{-1} < |h_j|^{-1}$, $|h_j|^{-\beta} < |h_j|^{-\alpha}$ and $(h_j-1)\times 1 < (h_j-2)\times 2 < \cdots \left(\frac{h_j}{2}\right)$. Note that we use the convention $\sum_{k_j=1}^{0} = 0$ here, so the inequality also holds for $h_j = 1$.

3

Case 3: $h_j < 0$. The derivation of Case 3 is similar to Case 2. The only difference is that we have the following orderings: $|-h_j+k_j| > |h_j|$ for $k_j > 0$ and $(-h_j-1)\times 1 < (-h_j-2)\times 2 < \cdots < \left(\frac{h_j}{2}\right)^2$.
Therefore,

$$\begin{aligned}
\mathscr{I}_j(h_j) &= \sum_{k_j=-\infty}^{\infty} \mathfrak{d}_{\alpha,\gamma_j}(h_j-k_j)\mathfrak{d}_{\beta,\rho_j}(k_j) \\
&= \sum_{k_j=1}^{\infty} \mathfrak{d}_{\alpha,\gamma_j}(k_j)\mathfrak{d}_{\beta,\rho_j}(k_j-h_j) + \mathfrak{d}_{\beta,\rho_j}(h_j) + \sum_{k_j=1}^{-h_j-1} \mathfrak{d}_{\alpha,\gamma_j}(h_j+k_j)\mathfrak{d}_{\beta,\rho_j}(k_j) + \mathfrak{d}_{\alpha,\gamma_j}(h_j) \\
&\quad + \sum_{k_j=1}^{\infty} \mathfrak{d}_{\alpha,\gamma_j}(h_j-k_j)\mathfrak{d}_{\beta,\rho_j}(k_j) \\
&\leq \rho_j\zeta(2\alpha)\mathfrak{d}_{\alpha,\gamma_j}(h_j) + \frac{\rho_j}{\gamma_j}\mathfrak{d}_{\gamma_j,\alpha}(h_j) \\
&\quad + \sum_{k_j=1}^{-h_j-1} \gamma_j|h_j+k_j|^{-2\alpha}\rho_j|k_j|^{-2\beta} + \mathfrak{d}_{\alpha,\gamma_j}(h_j) + \rho_j\zeta(2\beta)\mathfrak{d}_{\alpha,\gamma_j}(h_j) \\
&\leq \rho_j(\zeta(2\alpha)+\zeta(2\beta))\mathfrak{d}_{\alpha,\gamma_j}(h_j) + \frac{\rho_j}{\gamma_j}\mathfrak{d}_{\gamma_j,\alpha}(h_j) \\
&\quad + \mathfrak{d}_{\alpha,\gamma_j}(h_j) + 2^{2\alpha}\mathfrak{d}_{\alpha,\gamma_j}(h_j)\rho_j\zeta(2(\beta-\alpha)).
\end{aligned}$$

Finally, let $\mathscr{C}_j = \max\{1+2\gamma_j\rho_j\zeta(2\alpha+2\beta), 1+\rho_j\left(\zeta(2\alpha)+\zeta(2\beta)+\frac{1}{\gamma_j}+2^{2\alpha}\zeta(2(\beta-\alpha))\right)\}$, which is independent of h_j. Then,

$$\left|\int_{\mathbb{R}^s} \sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}} \sum_{\mathbf{k}\in\mathbb{Z}^s} (\sqrt{2})^{-|\mathbf{h}-\mathbf{k}|_0} \widetilde{f}_{\cos}(\mathbf{h}-\mathbf{k}) \left(\prod_{j=1}^{s} \cos\left(\pi k_j \frac{y_j-a_j}{b_j-a_j}\right)\right) \nu(d\mathbf{y})\right|$$

$$\leq \left(\sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}} \prod_{j=1}^{s} \mathscr{C}_j\mathfrak{d}_{\alpha,\gamma_j}(h_j)\right)^{\frac{1}{2}} \mathscr{M}\|f\|_{\mathscr{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})} = \left(\sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}} \mathfrak{d}_{\alpha,\boldsymbol{\gamma},s}(\mathbf{h})\right)^{\frac{1}{2}} \left(\prod_{j=1}^{s}\mathscr{C}_j\right)^{\frac{1}{2}} \mathscr{M}\|f\|_{\mathscr{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}.$$

In this proof, we break down the sum in Equation (3.7) and derive a bound for each dual lattice point $\mathbf{h}$ along each direction, namely, the $\mathscr{I}_j(h_j)$ terms in (3.8). By considering the three possible cases for h_d, positive, negative and zero, we provide a bound in each case. It is necessary to separate the three cases as we have to identify the section

where both $|h_j - k_j|$ and $|k_j|$ are smaller than $|h_j|$ when k_j is moving along the real line and apply a separated bound on these sections.

Taking the maximum out of the three cases and putting $\mathscr{I}_j(h_j)$ back into the sum in Equation (3.5) finishes the proof. □

Remark 3.1. The integrals of the form, $\int_{[0,1]^s} \mathfrak{f}(\mathbf{y})d\mathbf{y}$, can be seen as special cases of Equation (3.4) where we take the probability measure as a uniform distribution over the whole box $\mathscr{D} = [0,1]^s$. In this case,

$$\begin{aligned}\int_{\mathbb{R}^s}\left(\prod_{j=1}^{s}\cos\left(\pi k_j\frac{y_j-a_j}{b_j-a_j}\right)\right)\nu(d\mathbf{y}) &= \int_{[0,1]^s}\left(\prod_{j=1}^{s}\cos\left(\pi k_j y_j\right)\right)d\mathbf{y}\\ &= \begin{cases}1 & \text{if } k\equiv\mathbf{0};\\ 0 & \text{otherwise.}\end{cases}\end{aligned}$$

Therefore, we may simplify the square of the term in Equation (3.7), as follows

$$\begin{aligned}&\sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}}\left(\sum_{\mathbf{k}\in\mathbb{Z}^s}(\sqrt{2})^{-|\mathbf{h}-\mathbf{k}|_0}\widetilde{\mathfrak{f}}_{\cos}(\mathbf{h}-\mathbf{k})\int_{\mathbb{R}^s}\left(\prod_{j=1}^{s}\cos\left(\pi k_j\frac{y_j-a_j}{b_j-a_j}\right)\right)\nu(d\mathbf{y})\right)^2\\ &=\sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}}(\sqrt{2})^{-|\mathbf{h}|_0}\widetilde{\mathfrak{f}}_{\cos}(\mathbf{h}),\end{aligned}$$

which is the same as the right-hand side of Equation (6) in [14]. Therefore, our numerical integration can be seen as an extension of the Quasi-Monte Carlo (QMC) rules in [14].

The above proof is rather rough, note the $2^{2\alpha}$ term in the last two cases. However, it suggests we can still relate the integration error to the smoothness of the target function $\mathfrak{f}$ in terms of the cosine coefficients. The error can be controlled by the sum $\left(\sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}}\mathfrak{d}_{\alpha,\boldsymbol{\gamma},s}(\mathbf{h})\right)^{1/2}$. Thus all the results for the worst-case error for the QMC integration in the half-period cosine space using tent-transformed lattice rules can be used here. This opens the door for applying the generating vector for extensible lattice rules from [13] and other results from the QMC literature to the half-period cosine expansion scheme. We believe this result is a promising starting point for the study of lattice expansions for higher-dimension numerical integration with general finite measure.

3.2.3. FULL APPROXIMATION SCHEMES AND ERRORS

Now, we can introduce the full approximation scheme. The final obstacle lies in the reproducing kernel, which is defined as an infinite sum. Instead of calculating the full sum, a truncated version is used in the actual algorithm. We define

$$\mathscr{K}^{\cos}_{s,K}(\mathbf{x},\mathbf{y}) := \sum_{\mathbf{k}\in\mathbb{Z}^s,\,\sum|k_i|\le K}\left(\prod_{j=1}^{s}\cos\left(\pi k_j\frac{x_j-a_j}{b_j-a_j}\right)\right)e^{\imath\pi\mathbf{k}\cdot\frac{\mathbf{y}-\mathbf{a}}{\mathbf{b}-\mathbf{a}}},\ K>s. \tag{3.9}$$

For any expectation that satisfies the conditions in the previous section, we have our full approximant:

$$\mathbb{E}[\mathfrak{f}(\mathbf{Y})]$$

$$
\begin{aligned}
&\approx \int_{\mathbb{R}^s} \frac{1}{N}\sum_{n=0}^{N-1} \bar{\mathfrak{f}}\left(\mathrm{p}_{\boldsymbol{\phi},n}\right) \mathscr{K}_{s,K}^{\cos}\left(\mathrm{p}_{\boldsymbol{\phi},n},\mathbf{y}\right) \nu(d\mathbf{y}) \\
&= \frac{1}{N}\sum_{n=0}^{N-1} \bar{\mathfrak{f}}\left(\mathrm{p}_{\boldsymbol{\phi},n}\right) \int_{\mathbb{R}^s} \sum_{\mathbf{k}\in\mathbb{Z}^s,\, \sum|k_i|\le K} \left(\prod_{j=1}^{s} \cos\left(\pi k_j \phi\left(\left\{\frac{n g_j}{N}\right\}\right)\right)\right) e^{\imath\pi\mathbf{k}\cdot\frac{\mathbf{y}-\mathbf{a}}{\mathbf{b}-\mathbf{a}}} \nu(d\mathbf{y}) \\
&= \frac{1}{N}\sum_{n=0}^{N-1} \bar{\mathfrak{f}}\left(\mathrm{p}_{\boldsymbol{\phi},n}\right) \left(\sum_{\mathbf{k}\in\mathbb{Z}^s,\, \sum|k_i|\le K} \left(\prod_{j=1}^{s} \cos\left(\pi k_j \phi\left(\left\{\frac{n g_j}{N}\right\}\right)\right)\right) e^{-\imath\pi\mathbf{k}\cdot\frac{\mathbf{a}}{\mathbf{b}-\mathbf{a}}} \int_{\mathbb{R}^s} e^{\imath\pi\mathbf{k}\cdot\frac{\mathbf{y}}{\mathbf{b}-\mathbf{a}}} \nu(d\mathbf{y})\right) \\
&= \frac{1}{N}\sum_{n=0}^{N-1} \bar{\mathfrak{f}}\left(\mathrm{p}_{\boldsymbol{\phi},n}\right) \left(\sum_{\mathbf{k}\in\mathbb{Z}^s,\, \sum|k_i|\le K} \left(\prod_{j=1}^{s} \cos\left(\pi k_j \phi\left(\left\{\frac{n g_j}{N}\right\}\right)\right)\right) e^{-\imath\pi\mathbf{k}\cdot\frac{\mathbf{a}}{\mathbf{b}-\mathbf{a}}} \mathscr{F}_\nu\left(\frac{\mathbf{k}\pi}{\mathbf{b}-\mathbf{a}}\right)\right), \qquad (3.10)
\end{aligned}
$$

where $\mathscr{F}_\nu$ is the Fourier transform of the measure ν. Recall from Equation (3.3) that $\mathfrak{f}$ and $\bar{\mathfrak{f}}$ coincide in $\mathscr{D}$. We can simply replace $\bar{\mathfrak{f}}$ by $\mathfrak{f}$ in the above scheme.

In practice, one would first calculate the expected reproducing kernel value $\int_{\mathbb{R}^s} \mathscr{K}_{s,K}^{\cos}\left(\mathrm{p}_{\boldsymbol{\phi},n},\mathbf{y}\right)\nu(d\mathbf{y})$ for each lattice point, possibly in an offline setting and then calculate the main sum.

Define the absolute approximation error under this setting as ϵ and we see that

$$
\begin{aligned}
\epsilon &:= \left|\mathbb{E}[\mathfrak{f}(\mathbf{Y})] - \int_{\mathbb{R}^s} \frac{1}{N}\sum_{n=0}^{N-1} \mathfrak{f}\left(\mathrm{p}_{\boldsymbol{\phi},n}\right) \mathscr{K}_{s,K}^{\cos}\left(\mathrm{p}_{\boldsymbol{\phi},n},\mathbf{y}\right) \nu(d\mathbf{y})\right| \\
&\le \left|\mathbb{E}[\mathfrak{f}(\mathbf{Y})] - \mathbb{E}[\bar{\mathfrak{f}}(\mathbf{Y})]\right| \\
&\quad + \left|\mathbb{E}[\bar{\mathfrak{f}}(\mathbf{Y})] - \int_{\mathbb{R}^s} \frac{1}{N}\sum_{n=0}^{N-1} \bar{\mathfrak{f}}\left(\mathrm{p}_{\boldsymbol{\phi},n}\right) \mathscr{K}_{s}^{\cos}\left(\mathrm{p}_{\boldsymbol{\phi},n},\mathbf{y}\right) \nu(d\mathbf{y})\right| \\
&\quad + \left|\int_{\mathbb{R}^s} \frac{1}{N}\sum_{n=0}^{N-1} \bar{\mathfrak{f}}\left(\mathrm{p}_{\boldsymbol{\phi},n}\right) \mathscr{K}_{s}^{\cos}\left(\mathrm{p}_{\boldsymbol{\phi},n},\mathbf{y}\right) \nu(d\mathbf{y}) - \int_{\mathbb{R}^s} \frac{1}{N}\sum_{n=0}^{N-1} \bar{\mathfrak{f}}\left(\mathrm{p}_{\boldsymbol{\phi},n}\right) \mathscr{K}_{s,K}^{\cos}\left(\mathrm{p}_{\boldsymbol{\phi},n},\mathbf{y}\right) \nu(d\mathbf{y})\right| \\
&\le \mathbb{E}[\left|\mathfrak{f}(\mathbf{Y}) - \bar{\mathfrak{f}}(\mathbf{Y})\right| \mathbf{1}_{\{\mathbb{R}^s\setminus\mathscr{D}\}}(\mathbf{Y})] + \left(\sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}} \mathfrak{d}_{\alpha,\boldsymbol{\gamma},s}(\mathbf{h})\right)^{\frac{1}{2}} \left(\prod_{i=1}^{s} \mathscr{C}_i\right)^{\frac{1}{2}} \mathscr{M}\,||\mathfrak{f}||_{\mathscr{K}_{\alpha,\boldsymbol{\gamma},s}^{\cos}(\mathscr{D})} \\
&\quad + \frac{1}{N}\sum_{n=0}^{N-1} \left|\bar{\mathfrak{f}}\left(\mathrm{p}_{\boldsymbol{\phi},n}\right)\right| \left(\sum_{\mathbf{k}\in\mathbb{Z}^s,\, \sum|k_i|>K} \left|\int_{\mathbb{R}^s} \left(\prod_{j=1}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right) \nu(d\mathbf{y})\right|\right).
\end{aligned}
$$

For the third term, a bit more detail is required. Again we have the convergence assumption on the cosine transform of measure ν with the additional condition that $\beta + 0.5 + \delta > s$. With the inequality $(K-1)\times 1 < (K-2)\times 2 < \ldots < \left(\frac{K}{2}\right)^2$, for all K with $K \in \mathbb{N}$, $K > s$ and $K > 2$ and mathematical induction, one can show that if $\sum|k_i| = K$,

$$
\left(\mathfrak{d}_{\beta+0.5+\delta,\boldsymbol{\rho},s}(\mathbf{k})\right)^{\frac{1}{2}} \le |K-s|^{-(\beta+0.5+\delta)} \prod_{j=1}^{s} \max\{1, \sqrt{\rho_j}\}.
$$

There are in total $\frac{1}{(s-1)!}\prod_{j=1}^{s-1}(K+j)$ s-tuples of natural numbers satisfying $\sum_{i=1}^{s} k_i = K$ for $s \ge 2$. (see [17] for further explanation.) Taking into account all the possible combinations of positive and negative signs among all dimensions when we consider all integers, we have

$$
\sum_{\mathbf{k}\in\mathbb{Z}^s,\, \sum|k_i|>K} \left(\mathfrak{d}_{\beta+0.5+\delta,\boldsymbol{\rho},s}(\mathbf{k})\right)^{\frac{1}{2}}
$$

$$\leq \sum_{k=K+1}^{\infty} 2^s \frac{1}{(s-1)!} \prod_{j=1}^{s-1} (k+j) \times |k-s|^{-(\beta+0.5+\delta)} \prod_{j=1}^{s} \max\{1, \sqrt{\rho_j}\}$$
$$\leq \frac{2^s}{(s-1)!} \left(1 + \frac{2s-1}{K}\right)^{s-1} \prod_{j=1}^{s} \max\{1, \sqrt{\rho_j}\} \sum_{k=K+1}^{\infty} |k-s|^{-(\beta+1.5+\delta-s)}.$$

The final inequality also holds when $s = 1$.

Therefore, we find the following error bound for the lattice expansion scheme:

$$\epsilon \leq \mathbb{E}[\left|\hat{f}(\mathbf{Y}) - \bar{\hat{f}}(\mathbf{Y})\right| \mathbf{1}_{\{\mathbb{R}^s \setminus \mathscr{D}\}}(\mathbf{Y})] + \left(\sum_{\mathbf{h} \in L^{\perp} \setminus \{\mathbf{0}\}} \vartheta_{\alpha,\boldsymbol{\gamma},s}(\mathbf{h})\right)^{\frac{1}{2}} \left(\prod_{i=1}^{s} \mathscr{C}_i\right)^{\frac{1}{2}} \mathscr{M} ||\hat{f}||_{\mathscr{K}^{\cos}_{\alpha,\boldsymbol{\gamma},s}(\mathscr{D})}$$
$$+ \frac{1}{N} \sum_{n=0}^{N-1} \left|\bar{\hat{f}}(\mathrm{p}_{\boldsymbol{\phi},n})\right| \frac{2^s}{(s-1)!} \left(1 + \frac{2s-1}{K}\right)^{s-1} \prod_{j=1}^{s} \max\{1, \sqrt{\rho_j}\} |K-s|^{-(\beta+0.5+\delta-s)}. \quad (3.11)$$

To conclude, we can apply the telescoping technique and describe our error by three separate pieces: the projection error from a non-periodic to a periodic function, the lattice integration error and the kernel truncation error. With this proof, we successfully extended the lattice integration in [14] to a more general setting and made use of both the smoothness of the integrand in terms of cosine coefficients and the convergence of the cosine transform with respect to the probability measure. However, with our "number theory based" derivation, the model dimension s still heavily impacts the complete error bound, which may be an obstacle when applying the method to higher-dimensional problems. Moreover, the assumption of the cosine transform's decay is rough. Results from Fourier analysis may be incorporated to further improve the error bound. Further study for the error control is therefore recommended.

3.3. DISCUSSION

In Section 3.2, we have provided a justification for the cosine expansion lattice scheme. We extended the tent transformed lattice rule in the half-cosine space to general probability measures, with its error controlled by the decay rate of the cosine transform of the probability measures as well as the cosine coefficients of the integrands.

However, there are remaining questions. Is there a better way to control the approximation error than simple algebraic manipulation? How does our current scheme connect to previous work on wavelets? Here we aim to present some discussion on the above issues.

3.3.1. ALTERNATIVE ERROR FORMULATION

The first possibility we would like to consider is whether the error of the approximation in Section 3.2.2 can be written in an alternative form. By doing so, we aim to avoid bounding the term $\mathscr{I}(\mathbf{h})$ along each dimension, as this estimation is rough.

We revisit the derivation of ϵ_2 in Section 3.2.2, specifically the right-hand side of Equation (3.6). Note that the innermost sum in the expression can be rewritten as follows:

$$\sum_{\mathbf{k} \in \mathbb{Z}^s} (\sqrt{2})^{-|\mathbf{h}-\mathbf{k}|_0} \widetilde{\hat{f}}_{\cos}(\mathbf{h}-\mathbf{k}) \left(\prod_{j=1}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right)$$

$$= \sum_{\vec{k}\in\mathbb{Z}^s} \left(\prod_{j=1}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right) \int_{[0,1]^s} f(\mathbf{z}\times(\mathbf{b}-\mathbf{a})+\mathbf{a}) \prod_{j=1}^{s} \cos(\pi(h_j - k_j)z_j) d\mathbf{z}$$

$$= \sum_{k_1=1}^{\infty} \sum_{\mathbf{k}_{-1}\in\mathbb{Z}^{s-1}} \cos\left(\pi k_1 \frac{y_1 - a_1}{b_1 - a_1}\right) \left(\prod_{j=2}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right) \times$$

$$\int_{[0,1]^s} f(\mathbf{z}\times(\mathbf{b}-\mathbf{a})+\mathbf{a}) \cos(\pi(h_1 - k_1)z_1) \prod_{j=2}^{s} \cos(\pi(h_j - k_j)z_j) d\mathbf{z}$$

$$+ \sum_{\mathbf{k}_{-1}\in\mathbb{Z}^{s-1}} \left(\prod_{j=2}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right) \times$$

$$\int_{[0,1]^s} f(\mathbf{z}\times(\mathbf{b}-\mathbf{a})+\mathbf{a}) \cos(\pi h_1 z_1) \prod_{j=2}^{s} \cos(\pi(h_j - k_j)z_j) d\mathbf{z}$$

$$+ \sum_{k_1=-\infty}^{-1} \sum_{\mathbf{k}_{-1}\in\mathbb{Z}^{s-1}} \cos\left(\pi k_1 \frac{y_1 - a_1}{b_1 - a_1}\right) \left(\prod_{j=2}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right) \times$$

$$\int_{[0,1]^s} f(\mathbf{z}\times(\mathbf{b}-\mathbf{a})+\mathbf{a}) \cos(\pi(h_1 - k_1)z_1) \prod_{j=2}^{s} \cos(\pi(h_j - k_j)z_j) d\mathbf{z},$$

by using the definition of cosine coefficients. We separate the sum here in three parts according to whenever k_1 is positive, negative or zero. Next, we combine the terms whose $|k_1|$ value is the same.

$$\sum_{\mathbf{k}\in\mathbb{Z}^s} (\sqrt{2})^{-|\mathbf{h}-\mathbf{k}|_0} \widetilde{f}_{\cos}(\mathbf{h}-\mathbf{k}) \left(\prod_{j=1}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right)$$

$$= \sum_{\mathbf{k}_{-1}\in\mathbb{Z}^{s-1}} \left(\prod_{j=2}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right) \int_{[0,1]^s} f(\mathbf{z}\times(\mathbf{b}-\mathbf{a})+\mathbf{a}) \cos(\pi h_1 z_1) \prod_{j=2}^{s} \cos(\pi(h_j - k_j)z_j) d\mathbf{z}$$

$$+ \sum_{k_1=1}^{\infty} \sum_{\mathbf{k}_{-1}\in\mathbb{Z}^{s-1}} \cos\left(\pi k_1 \frac{y_1 - a_1}{b_1 - a_1}\right) \left(\prod_{j=2}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right) \times$$

$$\int_{[0,1]^s} f(\mathbf{z}\times(\mathbf{b}-\mathbf{a})+\mathbf{a}) (\cos(\pi(h_1 + k_1)z_1) + \cos(\pi(h_1 - k_1)z_1)) \prod_{j=2}^{s} \cos(\pi(h_j - k_j)z_j) d\mathbf{z}$$

$$= \sum_{k_1\in\mathbb{N}_0} \sum_{\mathbf{k}_{-1}\in\mathbb{Z}^{s-1}} \left(\prod_{j=1}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right) \times$$

$$2^{|k_1|_0} \int_{[0,1]^s} f(\mathbf{z}\times(\mathbf{b}-\mathbf{a})+\mathbf{a}) \cos(\pi h_1 z_1) \cos(\pi k_1 z_1) \prod_{j=2}^{s} \cos(\pi(h_j - k_j)z_j) d\mathbf{z}.$$

Then we repeat the similar steps of separating terms and combining those with the same absolute value for $k_2, \cdots, k_s$, which leads to:

$$\sum_{\mathbf{k}\in\mathbb{Z}^s} (\sqrt{2})^{-|\mathbf{h}-\mathbf{k}|_0} \widetilde{f}_{\cos}(\mathbf{h}-\mathbf{k}) \left(\prod_{j=1}^{s} \cos\left(\pi k_j \frac{y_j - a_j}{b_j - a_j}\right)\right)$$

$$= \sum_{\mathbf{k}\in\mathbb{N}_0^s} \int_{[0,1]^s} f(\mathbf{z}\times(\mathbf{b}-\mathbf{a})+\mathbf{a}) \left(\prod_{j=1}^{s} \cos(\pi h_j z_j)\right) 2^{|\mathbf{k}|_0/2} \prod_{j=1}^{s} \cos(\pi k_j z_j) d\mathbf{z} \times$$

$$2^{|\mathbf{k}|_0/2}\left(\prod_{j=2}^{s}\cos\left(\pi k_j\frac{y_j-a_j}{b_j-a_j}\right)\right).$$

This can be seen as the half-period cosine expansion of the function

$$\mathfrak{f}_{\mathbf{h}}(\mathbf{y}):=\mathfrak{f}(\mathbf{y})\left(\prod_{j=1}^{s}\cos\left(\pi h_j\frac{y_j-a_j}{b_j-a_j}\right)\right).$$

Therefore, the right-hand side of Equation (3.6) can be rewritten as

$$\int_{\mathbb{R}^s}\sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}}\sum_{\mathbf{k}\in\mathbb{Z}^s}(\sqrt{2})^{-|\mathbf{h}-\mathbf{k}|_0}\widetilde{f}_{\cos}(\mathbf{h}-\mathbf{k})\left(\prod_{j=1}^{s}\cos\left(\pi k_j\frac{y_j-a_j}{b_j-a_j}\right)\right)\nu(d\mathbf{y})$$
$$=\sum_{\mathbf{h}\in L^\perp\setminus\{\mathbf{0}\}}\int_{\mathbb{R}^s}\bar{\mathfrak{f}}_{\mathbf{h}}(\mathbf{y})\nu(d\mathbf{y}).$$

To summarize, the error of this approximation is just the sum of functions $\bar{\mathfrak{f}}_{\mathbf{h}}$ integrated over the probability measure ν over the dual lattice. The approximation error is controlled by a series with individual terms $\int_{\mathbb{R}^s}\bar{\mathfrak{f}}_{\mathbf{h}}(\mathbf{y})\nu(d\mathbf{y})$, a cosine transform with respect to the cosine series of $\mathfrak{f}$ under the probability measure ν. The decay rate of such integral, depending on the combined smoothness of the measure ν and the function $\mathfrak{f}$, is the key to limit the error of our scheme.

3.3.2. COSINE WAVELETS

We place our work in the context of the construction of wavelets in Chapter 2 to see the similarities and differences between the two approaches. Instead of using the full Fourier series for the wavelet construction, as in Chapter 2, we choose a "cosine function only" basis set as our starting point, corresponding to the half-period cosine space we used in Section 3.2.

Under this setting, we start our derivation from the set

$$\Gamma_{a,b}:=\left\{\frac{1}{\sqrt{2}},\cos\left(k\pi\frac{y-a}{b-a}\right)\middle|\,k=1,2,\ldots\right\},$$

with $[a,b]\in\mathbb{R}$ a finite range. We define an inner product

$$<\mathfrak{f},\mathfrak{g}>_{L^2([a,b])}:=\frac{2}{b-a}\int_a^b\mathfrak{f}(y)\mathfrak{g}(y)dy,$$

and the corresponding norm $||\cdot||_{L^2([a,b])}$. We denote any function $\mathfrak{f}$ such that $||\mathfrak{f}||_{L^2([a,b])}<\infty$, as $\mathfrak{f}\in L^2([a,b])$. We may suppress the range part from the notation $L^2([a,b])$ if there is no ambiguity in the context.

Equipped with the above definitions, we construct an approximation space together with a localized basis. Consider the following function $\mathcal{K}_{N'}^{wl}:\mathbb{R}\times[N']\to\mathbb{R}$,

$$\mathcal{K}_{N'}^{wl}(x,r)$$
$$:=\frac{1}{2}+\sum_{k=1}^{N'-1}\cos\left(k\pi\frac{x-a}{b-a}\right)\cos\left(k\pi\frac{2r-1}{2N'}\right)$$

$$= \frac{1}{2} + \frac{1}{2}\sum_{k=1}^{N'-1} \cos\left(k\pi\left(\frac{x-a}{b-a} - \frac{2r-1}{2N'}\right)\right) + \frac{1}{2}\sum_{k=1}^{N'-1} \cos\left(k\pi\left(\frac{x-a}{b-a} + \frac{2r-1}{2N'}\right)\right) \tag{3.12}$$

$$= \begin{cases} \frac{N'}{2} & \text{if } \frac{x-a}{b-a} = 2l \pm \frac{2r-1}{2N'} \text{ for } l \text{ an integer,} \\ \frac{\sin((N'-\frac{1}{2})(\frac{x-a}{b-a} - \frac{2r-1}{2N'})\pi)}{4\sin(\frac{\pi}{2}(\frac{x-a}{b-a} - \frac{2r-1}{2N'}))} + \frac{\sin((N'-\frac{1}{2})(\frac{x-a}{b-a} + \frac{2r-1}{2N'})\pi)}{4\sin(\frac{\pi}{2}(\frac{x-a}{b-a} + \frac{2r-1}{2N'}))} & \text{otherwise,} \end{cases}$$

where $r = 1, 2, \ldots, N'$. This definition is again a special case of the scaling functions given in Equation (2.13) of [18], in which the authors presented a uniform approach for the construction of wavelets based on orthogonal polynomials. The properties of $\mathcal{K}^{wl}_{N'}$, that are relevant to our numerical method are listed in the next proposition.

Proposition 3.3. *The function $\mathcal{K}^{wl}_{N'}$, which is defined in Equation (3.12), satisfies the following properties:*

(a) The inner product of two scaling functions is given by the following equation:

$$< \mathcal{K}^{wl}_{N'}(\cdot, r), \mathcal{K}^{wl}_{N'}(\cdot, q) >= \mathcal{K}^{wl}_{N'}\left(a + \frac{2r-1}{2N'}(b-a), q\right), \qquad r, q = 1, 2, \ldots, N'.$$

Thus, $\{\mathcal{K}^{wl}_{N'}(x, r) | r = 1, \ldots, N'\}$ is an orthogonal set.

(b) The scaling function $\mathcal{K}^{wl}_{N'}(\cdot, r)$ is localized around $a + \frac{2r-1}{2N'}(b-a)$. By this we mean that for the subspace

$$V_{N'} := span\left\{\frac{1}{\sqrt{2}}, \cos\left(k\pi\frac{y-a}{b-a}\right) \middle| \, k = 1, 2, \ldots, N'-1\right\},$$

we have

$$\left\| \frac{\mathcal{K}^{wl}_{N'}(\cdot, r)}{\mathcal{K}^{wl}_{N'}(a + \frac{2r-1}{2N'}(b-a), r)} \right\|_{L^2} = \min\left\{ ||\mathfrak{f}||_{L^2} : \mathfrak{f} \in V_{N'}, \mathfrak{f}\left(a + \frac{2r-1}{2N'}(b-a)\right) = 1 \right\}.$$

(c) $\{\mathcal{K}^{wl}_{N'}(\cdot, r) | r = 1, 2, \ldots, N'\}$ is a basis for $V_{N'}$.

(d) The scaling function $\mathcal{K}^{wl}_{N'}$ is also a kernel polynomial in the sense that, for any function v in $V_{N'}$, we have

$$< v, \mathcal{K}^{wl}_{N'}(\cdot, r) >_{L^2([a,b])} = v\left(a + \frac{2r-1}{2J}(b-a)\right).$$

Readers are referred to [18] for further properties of such functions and for a condensed proof of the above properties one may follow the derivation of Theorem 2.1.

Applying the results obtained above, we may define the wavelet expansion $\mathfrak{f}_{wl} : \mathbb{R} \to \mathbb{R}$ for any function $\mathfrak{f} \in L^2([a, b])$ by

$$\mathfrak{f}_{wl}(y) := \frac{2}{N'}\sum_{r=1}^{N'} \mathfrak{f}\left(a + \frac{2r-1}{2N'}(b-a)\right)\mathcal{K}^{wl}_{N'}(y, r),$$

and approximate the expectation $\mathbb{E}[\mathfrak{f}(Y)]$ by,

$$\begin{aligned}
\mathbb{E}[\mathfrak{f}(Y)] \approx & \mathbb{E}\left[\frac{2}{N'}\sum_{r=1}^{N'}\mathfrak{f}\left(a+\frac{2r-1}{2N'}(b-a)\right)\mathcal{K}_{N'}^{wl}(Y,r)\right] \\
= & \frac{2}{N}\sum_{r=1}^{N'}\mathfrak{f}\left(a+\frac{2r-1}{2N'}(b-a)\right)\mathbb{E}\left[\frac{1}{2}+\sum_{k=1}^{N'-1}\cos\left(k\pi\frac{Y-a}{b-a}\right)\cos\left(k\pi\frac{2r-1}{2N'}\right)\right] \\
= & \frac{1}{N'}\sum_{r=1}^{N'}\mathfrak{f}\left(a+\frac{2r-1}{2N'}(b-a)\right)\mathbb{E}\left[1+\sum_{k=1}^{N'-1}(e^{\imath k\pi\frac{Y-a}{b-a}}+e^{-\imath k\pi\frac{Y-a}{b-a}})\cos\left(k\pi\frac{2r-1}{2N'}\right)\right] \\
= & \frac{1}{N'}\sum_{r=1}^{N'}\mathfrak{f}\left(a+\frac{2r-1}{2N'}(b-a)\right)\sum_{k=1-N'}^{N'-1}\cos\left(k\pi\frac{2r-1}{2N'}\right)e^{-\imath\pi k\frac{a}{b-a}}\mathbb{E}\left[e^{\imath\frac{k\pi}{b-a}Y}\right].
\end{aligned}$$

Comparing this expression with Equation (3.10) when $s=1$, it is clear that these two formulas are of the same form. The main difference is that the wavelet formula can be seen as an expansion scheme using the lattice points

$$\mathscr{P}_{wl}(N',[a,b]) := \left\{a+n\frac{2r-1}{2N'}(b-a) : 1\le r\le N'\right\},$$

with the kernel also bounded at the corresponding value $N'-1$.

Considering a two-dimensional function $\mathfrak{f}:\mathbb{R}^2\to\mathbb{R}$, we can extend the cosine wavelet to two dimensions by applying the expansion to each dimension separately.

$$\begin{aligned}
& \mathbb{E}[\mathfrak{f}(Y_1,Y_2)] \\
= & \mathbb{E}\left[\frac{2}{N'}\sum_{r_1=1}^{N'}\mathfrak{f}\left(a_1+\frac{2r_1-1}{2N'}(b_1-a_1),Y_2\right)\mathcal{K}_{N'}^{wl}(Y_1,r_1)\right] \\
= & \mathbb{E}\left[\frac{4}{N'^2}\sum_{r_1=1}^{N'}\sum_{r_2=1}^{N'}\mathfrak{f}\left(a_1+\frac{2r_1-1}{2N'}(b_1-a_1),a_2+\frac{2r_2-1}{2N'}(b_2-a_2)\right)\mathcal{K}_{N'}^{wl}(Y_1,r_1)\mathcal{K}_{N'}^{wl}(Y_2,r_2)\right] \\
= & \frac{1}{N'^2}\sum_{r_1=1}^{N'}\sum_{r_2=1}^{N'}\mathfrak{f}\left(a_1+\frac{2r_1-1}{2N'}(b_1-a_1),a_2+\frac{2r_2-1}{2N'}(b_2-a_2)\right)\times \\
& \sum_{k_1=1-N'}^{N'-1}\sum_{k_2=1-N'}^{N'-1}\left(\prod_{j=1}^{s}\cos\left(k_j\pi\frac{2r_j-1}{2N'}\right)\right)e^{-\imath\pi\mathbf{k}\cdot\frac{\mathbf{a}}{\mathbf{b}-\mathbf{a}}}\mathbb{E}\left[e^{\imath\frac{\mathbf{k}\pi}{\mathbf{b}-\mathbf{a}}\cdot\mathbf{Y}}\right].
\end{aligned}$$

Again this is in a similar form as in Equation (3.10). However, the total number of terms N'^2 increases with the dimension and there is no clear way to select the more significant ones.

One of the key advantages of the cosine expansion lattice scheme is that we have removed the link between the quadrature points and the number of summation terms N. This allows us to apply an online/offline construction in our algorithm. We can first compute the expectations of reproducing kernels $\mathbb{E}\left[\mathcal{K}_{s,K}^{\cos}(\mathrm{p}_{\boldsymbol{\phi},n},\mathbf{Y})\right]$ for all points in the lattice sequence and use the stored values in the approximation scheme. This construction is not feasible in the cosine wavelets setting and it is one of the advantages from the cosine expansion lattice scheme.

3.4. NUMERICAL EXPERIMENTS

In this section, we perform numerical experiments to test the cosine expansion lattice scheme.

In order to demonstrate that our scheme can inherit the results from the previous literature, we use the lattice sequence from [13] and perform the tent-transformation on them in all our tests. This sequence has been used in the numerical results section of [14] and readers are referred to the references therein for further information.

All figures displaying numerical results in this section present the log absolute error $\log_{10}(|\epsilon|)$ against the log number of lattice points $\log_{10}(N)$.

For the first two experiments, we compute the expectation with the following test function:

$$\mathfrak{f}^1_{s,w}(\mathbf{y}) := \prod_{j=1}^{s}\left(1+\frac{w^j}{21}(-10+42y_j^2-42y_j^5+21y_j^6)\right).$$

3.4.1. UNIFORM DISTRIBUTION

Whenever the reproducing kernel's expectation $\mathbb{E}[\mathscr{K}_s^{\cos}(\mathrm{p}_{\boldsymbol{\phi},n},\mathbf{Y})]$ is known explicitly, we can simplify the algorithm to its original form in Equation (3.5) instead of the full scheme in Equation (3.10). This results in an algorithm that has no difference to the original QMC rule in terms of computational complexity. As stated in Remark 3.1, this is indeed the case for the uniform distribution.

In this subsection, we approximate

$$\mathbb{E}[\mathfrak{f}^1_{s,w}(\mathbf{Y})],$$

where $\mathbf{Y}$ follows a uniform distribution on the domain $[0,1]\times[-1,1]\times[0,1]\times[-1,1]\times\cdots$ for dimension $s\in[8]$. The reference value is $\prod_{j=1}^{s}\left(2+w^j\frac{2}{3}\right)^{(j \mod 2)}$.

Note that we are not establishing a new numerical result or proof of convergence here as this problem can be solved by the original QMC algorithm through a change of variables. These tests serve two purposes. They establish the base line result as a comparison to the results in the next section and allow us to provide comments on the actual implementation of the QMC algorithm.

The result for $w=0.5$ can be seen in Figure 3.1. All tests use 2^{20} evaluations for their final result. The reader should note that instead of having 1 as the common reference value throughout all dimensions, as in [14], the reference values of our setting follow those in Table 3.1. This is by design to demonstrate some key properties of lattice expansions. Under this setting, results from neighboring dimensions form a pair, for example, dimensions 4 and 5 share the same reference value. As demonstrated in Figure 3.1, the absolute errors converge to similar limits for these pairs, while the results worsen when the reference increases. This demonstrates that the QMC rules can be generalized to higher dimensions, but their error depends on the value of the target integration, which fits the result from Theorem 3.2.

Finally, it should be noted that the result for dimension 1 is particular strong. This is because the function evaluation for $y_j\in[0,1)$, the monomial y_j^k would always be around zero which helps stabilizing the function evaluation. When we use a projection domain

Table 3.1: Reference solution for Uniform Distribution

Dimension	1	2	3	4	5	6	7	8
Ref. Values	1	2.167	2.167	4.424	4.424	8.893	8.893	17.81

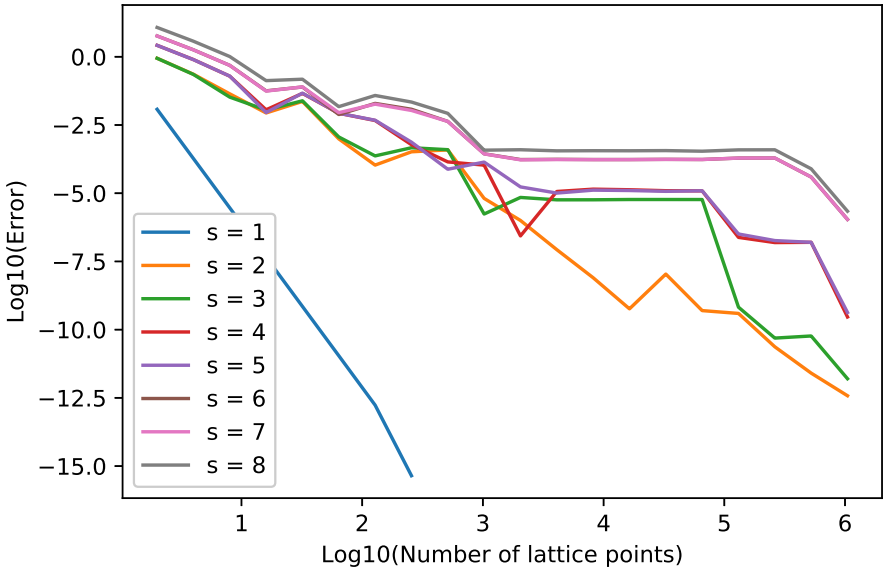


Figure 3.1: Absolute error verses the number of lattice points for uniform distribution.

that is larger than the unit box when approximating a polynomial, as we do in the upcoming examples, the results may seem to be worse than standard QMC result and this is one of the contributing factors.

3.4.2. NORMAL DISTRIBUTION

In this subsection, we approximate

$$\mathbb{E}[f^1_{s,0.9}(\mathbf{Y})],$$

where $\mathbf{Y}$ follows a multivariate normal distribution with mean 0 for all dimensions and covariance matrix

$$\begin{pmatrix} 0.5^2 & & \\ & \ddots & \\ & & 0.5^2 \end{pmatrix}.$$

Further, we restrict our lattice to the box $[-4.5, 4.5]^s$ and we study the cases of dimension $s \in [3]$. We applied the full scheme in Equation (3.10) here with $K = 2^7$. The reference values listed in vector form are $(1.2324, 1.4901, 1.7705)$.

The convergence result can be seen in Figure 3.2. All tests use 2^{18} evaluations for their final result. The approximation value converges to the references value in all tests,

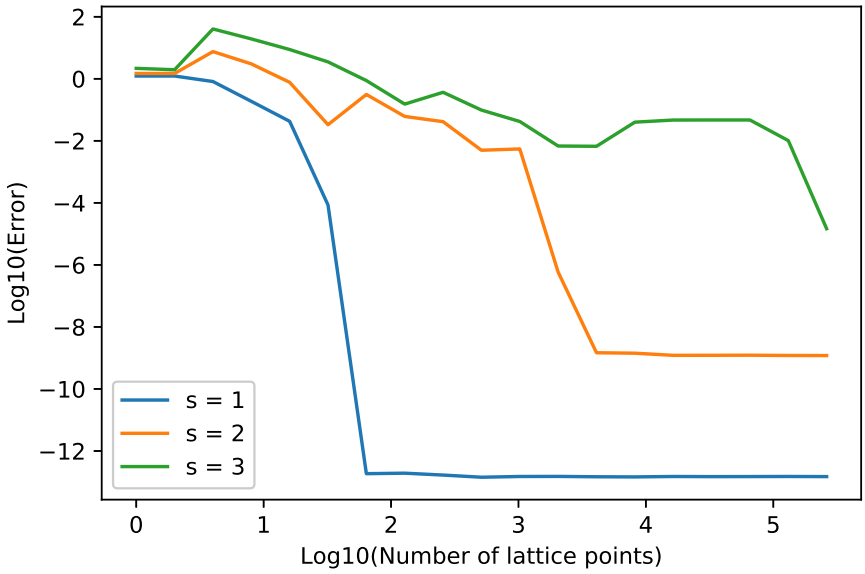


Figure 3.2: Absolute error verses the number of lattice points for normal distribution.

as we would expect. We can observe the structure of the error plateauing for a while with respect to the number of lattice points and suddenly improving when a critical number is reached in Figure 3.2. This seems to be related to the lattice sequence.

In the case of the normal distribution, there is a final level of error for each dimension, where increasing the number of lattice points further would not improve the approximation result. This is related to the error of projecting a non-periodic function to the half-period cosine space and the truncation of reproducing kernel. These errors dominate when the lattice rule error has converged and limit the final result.

FURTHER STUDIES ON PARAMETERS

With the successful implementation of our scheme, we conducted further tests for the selection of the kernel truncation parameter and projection domain. We did not consider the generation and construction of different lattice sequences in this article considering the large amount of previous research and work on this topic. It is difficult to draw a satisfying picture on this topic from a simple test. However, we would further consider the choices of projection domain and kernel truncation limit K and their impact on the approximation error. In the following tests, we focus on the two-dimensional case.

In Figure 3.3, we see the result of the same test we used for the normal distribution but instead of changing the test dimension s, we fixed the test dimension to 2 and varied the projection domain through a variable L. The projection domain is defined as $[-0.5L, 0.5L]^2$ and the kernel truncation K is fixed at 2^7.

From the result, we can see that the convergence behavior is similar among all tests

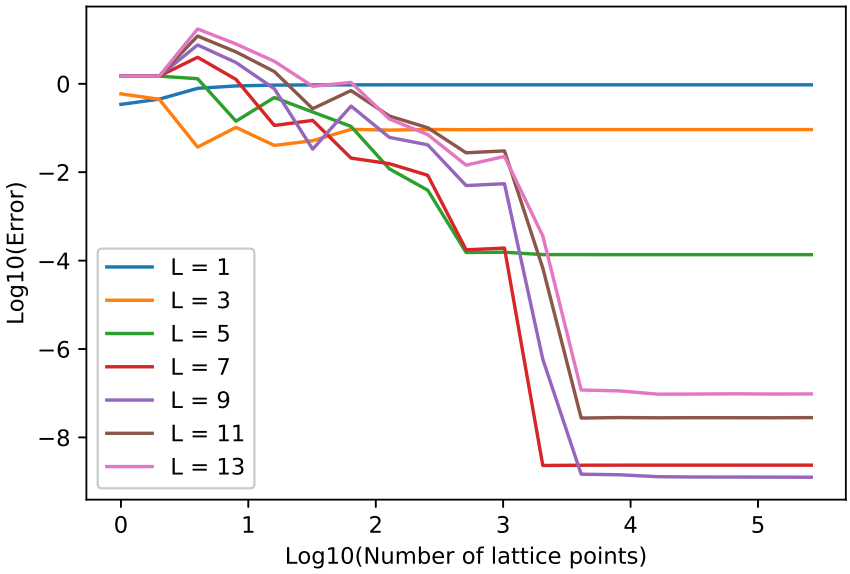


Figure 3.3: Absolute error verses the number of lattice points for the normal distribution for different projection domains.

for $L \geq 5$. Our result implies that the lattice sequence has the most significant influence on the convergence behavior with respect to the number of lattice points of our scheme. When most of the probability mass is included in the projection range ($L \geq 5$ here), the projection range only affects the final error. As one can see, the error is the lowest when $L = 9$, which can possibly be explained by the expression in Equation (3.11). The first term at the right hand side decreases when the probability mass outside of the projection range increases, while the rest of the terms relate to the averages of the integrand and increase with the projection range. We have to find the balance for these two competing effects when we choose the projection range.

Next, we can see the result of changing the kernel truncation factor K in Figure 3.4. The projection domain is fixed as $[-4.5, 4.5]^2$. Once again, the approximation converges to the final limit when the number of lattice points reaches 2^{12}. This again shows that the kernel truncation is only related to the final limit of approximation but not to the convergence behavior.

Another point of interest for us is the exponential decay of the Fourier transforms (the building blocks of the reproducing kernel). When $K \geq 64$, including more terms in the reproducing kernel does not improve the approximation. This was one of the main advantages of the COS method, making use of the smoothness of the probability measure to achieve better convergence results. It seems that we can arrive at similar results for this combined scheme.

However, the construction for the truncated kernel, defined in Equation (3.9), still suffers from the curse of dimensionality. Under our current setting, this issue can be ad-

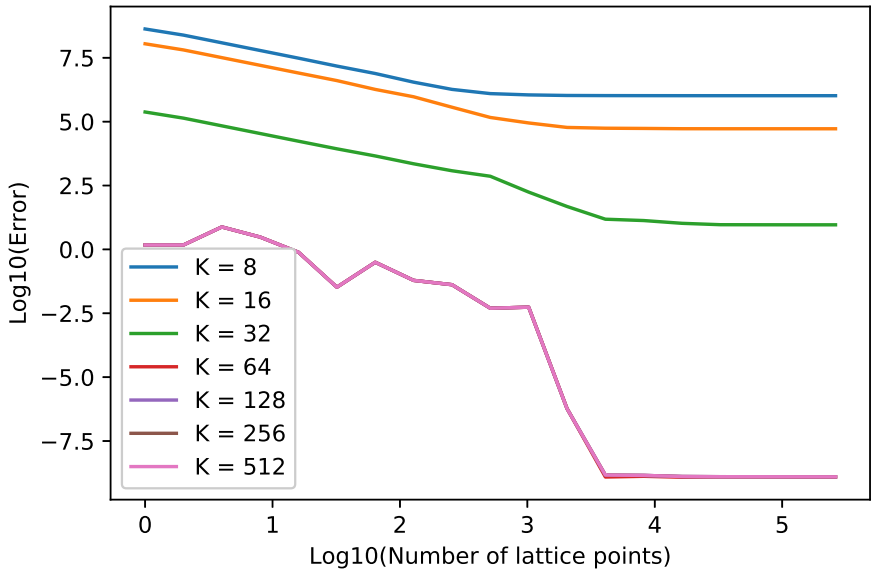


Figure 3.4: Absolute error verses the number of lattice points for the normal distribution with varying K.

dressed through software engineering by making use of an online/offline construction, in which we calculate and store the reproducing kernel in an offline stage and then use the stored results in actual calculation. Parallel computing or GPU computing are also possible means to speed up the computation.

We believe that this scheme is valuable for specifically making use of the smoothness of the probability measure and the clear separation of the measure from the integrand, which implies we may switch one part without affecting the other. Possible remedies to be studied for this scheme may include hyperbolic cross or non-linear basis constructions for the truncated kernel, or alternatively, different means for approximating the reproducing kernel.

To conclude, our additional parameter of kernel truncation and domain projection does not change the convergence behavior but only affects the final approximation error when it is sufficiently large. However, the kernel construction still gives rise to the curse of dimensionality.

3.4.3. ASYMMETRIC MULTIVARIATE LAPLACE DISTRIBUTION

Finally, we implement our scheme to a typical asymmetric 2-dimensional Laplace distribution $\mathbf{Y}$, whose characteristic function is given by

$$\mathscr{F}_{Lap}(\mathbf{k}) = \frac{1}{1 + 0.5\mathbf{k}^\top \boldsymbol{\Sigma}\mathbf{k} - \imath\bar{\boldsymbol{\mu}}^\top \mathbf{k}},$$

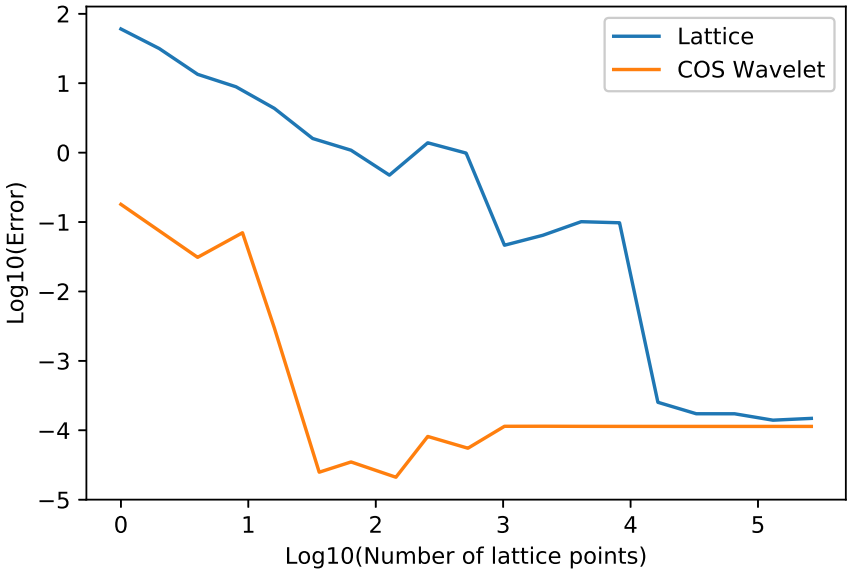


Figure 3.5: Absolute error verses the number of lattice points for Laplace distribution.

in this subsection[2]. In our setting, $\bar{\boldsymbol{\mu}}$ is a 2-dimensional vector given by $(0.3, -0.1)^\top$, while $\boldsymbol{\Sigma}$ is a 2 matrix defined as

$$\begin{pmatrix} 0.25 & -0.15 \\ -0.15 & 0.75 \end{pmatrix}.$$

The mean and covariance of this distribution are given by $\bar{\boldsymbol{\mu}}$ and $\boldsymbol{\Sigma} + \bar{\boldsymbol{\mu}}\bar{\boldsymbol{\mu}}^\top$, respectively. For further information, readers are referred to [19]. In this test, we compare the algorithm in Section 3.2 and the extended COS wavelet scheme in Section 3.3.2.

In this example, the integrand is simply $Y_1 Y_2$ and the analytic solution for the expectation is $\bar{\mu}_1\bar{\mu}_2 + (\boldsymbol{\Sigma} + \bar{\boldsymbol{\mu}}\bar{\boldsymbol{\mu}}^\top)_{1,2} = -0.21$.

We pick the projection domain $[\bar{\mu}_1 - 20\Sigma_{1,1}, \bar{\mu}_1 + 20\Sigma_{1,1}] \times [\bar{\mu}_2 - 20\Sigma_{2,2}, \bar{\mu}_2 + 20\Sigma_{2,2}]$ and set the kernel truncation parameter K as 2^6 for the cosine expansion lattice scheme. For the COS wavelet scheme, the order of the kernel and the number of lattice points are linked. Therefore, when we use N lattice points for the lattice scheme, we pick $N' = \lceil \sqrt{N} \rceil$ for the respective wavelet test. We report in Figure 3.5.

It is clear that both schemes converge to the same limit. The approximation error is dominated by the measure truncation error when the number of lattice points is high enough and therefore they have similar end results. Comparing both schemes, it seems that the COS wavelet performs better with a similar number of lattice points at the early stage. This may suggest that its quadrature point distribution is more efficient. The difference in quadrature points for the two schemes can be seen in Figure 3.6.

However, there are still some shortcomings in the construction of the COS wavelet that makes it a less attractive option comparing to the lattice scheme. For example, the

[2] Again, we use the notation $\mathscr{F}$ to denote characteristic functions in this thesis.

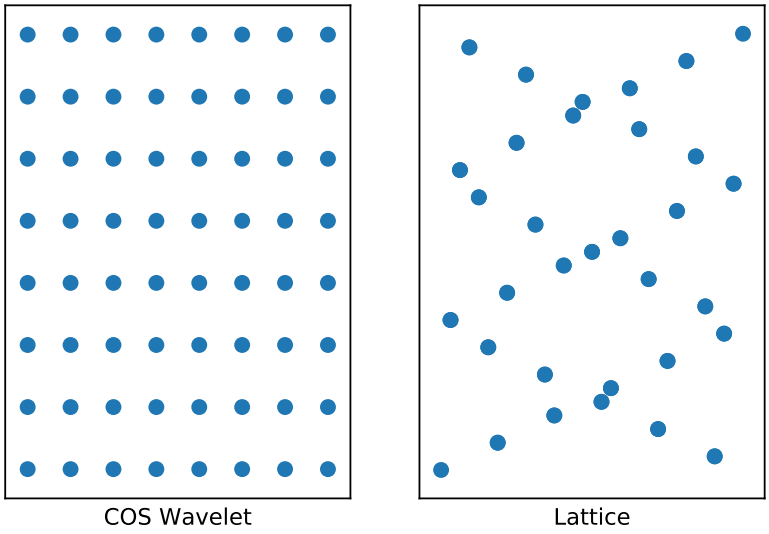


Figure 3.6: The quadrature points distribution for COS wavelet(left) and lattice sequence(right).

COS wavelet is not extendable in the sense that we cannot simply add an extra evaluation point to improve the approximation. A new evaluation from the beginning is required in such case.

Nevertheless, this test suggests that it would be of great interest to implement other quadrature rules with the half-cosine reproducing kernel.

Another point of interest is that in fact the cosine transform decay rate $\beta < 1.5$ for this example, so the rough error bound for the last term in Equation 3.11 fails to converge. However, the success in applying our algorithm clearly shows that we underestimate the decay rate of the cosine transform and a tighter error bound may be derived from advanced results in Fourier analysis. This motivates us to derive better error bounds for our scheme in the future.

3.5. CONCLUSION

Equipped with the Fourier-cosine projection technique, we extended the lattice sequence from [13] to the broader space of probability measures (instead of just the uniform distribution in [14]). The resulting approximation is reproducible and, in theory, this scheme can be generalized to higher-dimensional space by the extension of the results from the previous lattice literature.

The fact that our scheme concentrates all the information from the probability measure into the reproducing kernel and simply evaluates the integrand at some preset quadrature points gives us flexibility. We may use the same set of quadrature points for different probability measures as our derivation is not measure specific. Moreover, our scheme can be adjusted and possibly improved by simply replacing the lattice sequence by another one.

The main remaining issue is the approximation of the reproducing kernel which still suffers from the curse of dimensionality. However, with the rapid decay of the Fourier transform and the specific form of the reproducing kernel, we believe that further research on this topic would be fruitful.

REFERENCES

[1] M. J. Ruijter and C. W. Oosterlee, *Two-dimensional Fourier cosine series expansion method for pricing financial options,* SIAM Journal on Scientific Computing **34**, B642 (2012).

[2] J. Dick and F. Pillichshammer, *Digital Nets and Sequences: Discrepancy Theory and Quasi–Monte Carlo Integration* (Cambridge University Press, 2010).

[3] H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods* (Society for Industrial and Applied Mathematics, 1992).

[4] J. Dick, *On the convergence rate of the component-by-component construction of good lattice rules,* Journal of Complexity **20**, 493 (2004).

[5] F. Y. Kuo and S. Joe, *Component-by-component construction of good lattice rules with a composite number of points,* Journal of Complexity **18**, 943 (2002).

[6] I. H. Sloan and A. V. Reztsov, *Component-by-component construction of good lattice rules,* Mathematics of Computation **71**, 263 (2002).

[7] I. H. Sloan, F. Y. Kuo, and S. Joe, *On the step-by-step construction of quasi-monte carlo integration rules that achieve strong tractability error bounds in weighted Sobolev spaces,* Mathematics of Computation **71**, 1609 (2002).

[8] D. Nuyens and R. Cools, *Fast algorithms for component-by-component construction of rank-1 lattice rules in shift-invariant reproducing kernel Hilbert spaces,* Mathematics of Computation **75**, 903 (2006).

[9] D. Nuyens and R. Cools, *Fast component-by-component construction of rank-1 lattice rules with a non-prime number of points,* Journal of Complexity **22**, 4 (2006), special Issue.

[10] R. Cools, F. Y. Kuo, and D. Nuyens, *Constructing embedded lattice rules for multivariate integration,* SIAM Journal on Scientific Computing **28**, 2162 (2006).

[11] J. Dick, F. Pillichshammer, and B. J. Waterhouse, *The construction of good extensible rank-1 lattices,* Mathematics of Computation **77**, 2345 (2008).

[12] F. J. Hickernell and H. Niederreiter, *The existence of good extensible rank-1 lattices,* Journal of Complexity **19**, 286 (2003), oberwolfach Special Issue.

[13] F. J. Hickernell, P. Kritzer, F. Y. Kuo, and D. Nuyens, *Weighted compound integration rules with higher order convergence for all N,* Numerical Algorithms **59**, 161 (2012).

[14] J. Dick, D. Nuyens, and F. Pillichshammer, *Lattice rules for nonperiodic smooth integrands,* Numerische Mathematik **126**, 259 (2014).

[15] F. Fang and C. W. Oosterlee, *A novel pricing method for European options based on Fourier-cosine series expansions,* SIAM Journal on Scientific Computing **31**, 826 (2009).

[16] N. Aronszajn, *Theory of reproducing kernels,* Transactions of the American Mathematical Society **68**, 337 (1950).

[17] M. Lisi, *Some remarks on the Cantor pairing function,* Le Matematiche **62**, 55 (2007).

[18] B. Fischer and J. Prestin, *Wavelets based on orthogonal polynomials,* Mathematics of Computation **66**, 1593 (1997).

[19] T. J. Kozubowski, K. Podgórski, and I. Rychlik, *Multivariate generalized Laplace distribution and related random fields,* Journal of Multivariate Analysis **113**, 59 (2013), special Issue on Multivariate Distribution Theory in Memory of Samuel Kotz.

4

STOCHASTIC GRID BUNDLING METHOD FOR BSDES

In this chapter, we apply the Stochastic Grid Bundling Method (SGBM) to numerically solve BSDEs. The SGBM algorithm is based on conditional expectations approximation by means of bundling of Monte Carlo sample paths and a local regress-later regression within each bundle. The basic algorithm for solving the backward stochastic differential equations will be introduced and an upper error bound is established for the local regression. A full error analysis is also conducted for the explicit version of our algorithm and numerical experiments are performed to demonstrate various properties of our algorithm.

4.1. INTRODUCTION

The Stochastic Grid Bundling Method (SGBM) is a Monte Carlo based algorithm designed to solve backward dynamic programming problems, with applications in pricing Bermudan options in [2] and [3]. This algorithm has been further extended computationally by the incorporation of GPU acceleration in [4] and generalized to the computation of Credit Valuation Adjustment and Potential Future Exposure in [5]. In this work, we will extend its applicability to the approximation of Backward Stochastic Differential Equations (BSDEs). We shall also study the errors in the SGBM algorithm.

The SGBM algorithm is based on the so-called *regress-later technique* and on an adaptive local basis approach. In usual Monte Carlo regression methods for backward-in-time problems, the values of the target function at the end of a time interval are regressed on certain dependent variables that are measured at the beginning of the time interval (which is called the regress-now approach). This creates a statistical error. Instead, the dependent variable is projected onto a set of basis functions at the end of the interval in a regress-later method, and a conditional expectation across the interval is

This chapter is based on the article 'Stochastic grid bundling method for backward stochastic differential equations', published in International Journal of Computer Mathematics [1].

then computed for each basis function. This difference removes the statistical error in the regression step. Regress-later schemes have been further discussed in [6].

With an adaptive local basis approach, the whole simulation is partitioned into non-overlapping subsets and we perform least-squares regressions separately within these subsets, possibly with a different basis for each subset. The exact partition depends on the simulated samples themselves and its purpose is to gather samples that share similar "characteristics" such that the local regression is more accurate than the global one. For further application of localization in numerical schemes, the reader may check out [7]. Since each partition is non-overlapping, SGBM is easy to scale up in dimensionality and can facilitate parallel computing. We would like to test the SGBM algorithm in a new problem setting such that we can take advantage of its nice properties and also get a better understanding of the underlying principles.

In this work, we aim to construct an approximate scheme for BSDEs by the theta-scheme from [8] and apply the SGBM algorithm. The rest of the chapter is organized as follows. We start in Section 4.2 with the introduction of the SGBM algorithm, along with the necessary time discretization scheme and assumptions. Section 4.3 will present an error analysis of a simplified case of SGBM. The proof in this section forms the foundation for the error bound in any algorithm applying SGBM. Later, in Section 4.4, we derive the full error bound for a specific choice of discretization scheme as an example. The chapter finishes with numerical experiments and a conclusion.

To close off this section, here is some further notation that is used in this chapter.

- For any vector x, $|x|$ denotes its Euclidean norm and x_k denotes its k-th component.
- Similarly, $X_{k,t}$ denotes the k-th component for any random process X_t.
- The gradient ∇g is defined as $\left(\frac{\partial g}{\partial x_1}, \ldots, \frac{\partial g}{\partial x_q}\right)$ for any differentiable function $g : \mathbb{R}^q \to \mathbb{R}$.
- For any set $\mathscr{S}$, the function $\mathbf{1}_S$ is the indicator function which takes value 1 when the input is within set $\mathscr{S}$ and 0 otherwise.
- For any function space H containing functions $\phi : \mathbb{R}^q \to \mathbb{R}$, H^+ is defined as the set $\{\{(x, y) \in \mathbb{R}^q \times \mathbb{R} : \phi(x) \geq y\} : \phi \in H\}$.
- For any function ϕ and compact set $\mathscr{A}$, the control constant $C_{\phi,\mathscr{A}}$ is defined as an extended real number $\sup_{x\in\mathscr{A}} |\phi(x)|$.
- In general, the function set $\{\eta_1, \ldots, \eta_Q\}$ is used to denote a general regression basis in this chapter.

4.2. ASSUMPTIONS AND ALGORITHM

In this section, we shall introduce the SGBM algorithm and its application to the approximation of BSDEs. For the mathematical derivation in Section 4.2 and Section 4.4, we use a simplified backward discretization scheme described in the upcoming subsection. This is done for the interest of a clearer presentation.

4.2.1. DISCRETIZATION SCHEME

For the forward process X_t, we shall apply a Markovian approximation $X^\pi_{t_p}, t_p \in \pi$. The most common choice is the Euler-Maruyama scheme, which will be explained in Section 4.5. However, our algorithm can work with any simulation method where the conditional expectations over one time step are known for some specific functions.

The backward in time discretizations (Y^π, Z^π) come from a special case of the discretization formula of Equation (1.4), by selecting $(\theta_1, \theta_2) = (0, 1)$:

$$
\begin{aligned}
y_P(x) &= g(x), \\
z_p(x) &= \frac{1}{\Delta_p}\mathbb{E}^x_{t_p}\left[y_{p+1}(X^\pi_{t_{p+1}})\Delta W_p\right], \ p = P-1,\dots,0, \\
y_p(x) &= \mathbb{E}^x_{t_p}\left[y_{p+1}(X^\pi_{t_{p+1}})\right] + \Delta_p \mathbb{E}^x_{t_p}\left[f_{p+1}(y_{p+1}(X^\pi_{t_{p+1}}), z_{p+1}(X^\pi_{t_{p+1}}))\right], \ p = P-1,\dots,0, \quad (4.1)
\end{aligned}
$$

where $f_p(y,z) := f(t_p, X^\pi_{t_p}, y, z)$.

4.2.2. STANDING ASSUMPTIONS

To ensure the existence and uniqueness of the solution of the continuous BSDEs, some basic assumptions are required. Moreover, these assumptions will affect the algorithm designed regarding the admissible choice of π and the error bound of the scheme. In this work, we assume the global Lipschitz condition as stated in Assumption 4.1. Note that this assumption will affect the derivation and the result of the error bound for the complete algorithm. Assumption 4.1 is in force here as it is the most common assumption in the BSDE literature. Alternative assumptions can be found, for instance, in [9].

Assumption 4.1 (Globally Lipschitz driver)**.**

($\mathbf{A}_\xi$) i.) g is a measurable function.

ii.) The control constant $C_{\Phi,\mathscr{A}} < \infty$ for any given compact set $\mathscr{A}$.

($\mathbf{A}_F$) i.) $(t,x,y,z) \mapsto f(t,x,y,z)$ is $\mathfrak{B}(\mathbb{R}) \otimes \mathfrak{B}(\mathbb{R}^q) \otimes \mathfrak{B}(\mathbb{R}) \otimes \mathfrak{B}(\mathbb{R}^d)$-measurable.

ii.) For every $p \le N$, $f_p(y,z)$ as defined in Subsection 4.2.1 is $\mathfrak{F}_{t_p} \otimes \mathfrak{B}(\mathbb{R}) \otimes \mathfrak{B}(\mathbb{R}^d)$-measurable and there exists an $L_f \in [0, +\infty)$ such that

$$|f_p(y,z) - f_p(y',z')| \le L_f(|y-y'| + |z-z'|), \qquad \forall k \in \{0,\dots,N\},$$

for any $(y,y',z,z') \in \mathbb{R}\times\mathbb{R}\times\mathbb{R}^d\times\mathbb{R}^d$.

iii.) There exists a $C_f \in [0,\infty)$ such that

$$|f_p(0,0)| \le C_f, \qquad \forall p \in \{0,\dots,P\}.$$

iv.) The time discretization is such that

$$\limsup_{P\to\infty} R_\pi < +\infty, \quad \text{where } R_\pi = \sup_{0\le p\le P-2} \frac{\Delta_p}{\Delta_{p+1}}.$$

Again, the assumption here is for the consistency of our derivation and does not imply that our algorithm can only be applied when these assumptions are satisfied.

4.2.3. STOCHASTIC GRID BUNDLING METHOD

We now introduce SGBM. Recall from Chapter 1 that due to the Markovian setting of $(X^\pi_{t_p}, \mathfrak{F}_{t_p})_{t_p\in\pi}$, there exist functions $y_p(x)$ and $z_p(x)$ such that

$$Y^\pi_{t_p} = y_p(X^\pi_{t_p}),\ Z^\pi_{t_p} = z_p(X^\pi_{t_p}).$$

Our method is based on estimating these functions $(y_p(x), z_p(x))$ recursively backwards in time by a local least-squares regression technique onto a finite function space with basis functions $(\eta_l)_{0\le l\le Q}$.

As a Monte Carlo based algorithm, our program starts with the simulation of M independent samples of $(X^\pi_{t_p})_{0\le p\le P}$, denoted by $(X^{\pi,m}_{t_p})_{1\le m\le M, 0\le p\le P}$. Note that in this basic algorithm, the simulation is only performed once. This scheme is therefore a non-nested Monte Carlo scheme.

The next step is the backward recursion. Denote by y^R_p the SGBM approximation of the function y_p. The function z^R_p similarly means the approximation of z_p.

At initialization, we assign the terminal values to each path for our approximations, i.e.,

$$y^R_P(X^{\pi,m}_{t_P}) = g(X^{\pi,m}_{t_P}),\quad m = 1,\dots,M,$$

The following steps are performed recursively, backwards in time, at t_p, $p = P-1,\dots,0$. First, we bundle all paths into $\mathscr{B}_{t_p}(1),\dots,\mathscr{B}_{t_p}(B)$ non-overlapping partitions based on the result of $(X^{\pi,m}_{t_p})$. Note that our design allows the application of various clustering techniques within the SGBM algorithm. A previous study in [4] compares the k-means clustering with an equal partitioning, and shows that they are similar in accuracy. However, it remains an interesting problem which clustering technique would provide the optimal result. We use the equal partition technique, which will be specified in Section 4.3, for the error analysis and the numerical experiment.

Next, we perform the regress-later approximation separately within each bundle. The regress-later technique we are using combines the least-squares regression with the (analytical) expectations of the basis functions to calculate the target expectations.

Generally speaking, for M Monte Carlo paths, a standard regress-now algorithm for a dynamic programming problem finds a function ι within the space spanned by the regression basis such that it minimizes the value $\frac{1}{M}\sum_{i=1}^M(\mathfrak{g}(X^i_{t+\delta}) - \iota(X^i_t))^2$ and approximates the expectation $\mathbb{E}_t[\mathfrak{g}(X_{t+\delta})]$ by $\mathbb{E}_t[\iota(X_t)] = \iota(X_t)$. As a projection from a function of $X_{t+\delta}$ to a function of X_t is performed then, it would introduce a statistical bias to the approximation.

Instead, the regress-later technique we employ picks out a function κ such that it minimizes $\frac{1}{M}\sum_{i=1}^M(\mathfrak{g}(X^i_{t+\delta}) - \kappa(X^i_{t+\delta}))^2$ and approximates the expectation $\mathbb{E}_t[\mathfrak{g}(X_{t+\delta})]$ by $\mathbb{E}_t[\kappa(X_{t+\delta})]$. By using functions on the same variable in the regression basis, we can remove the statistical bias in the regression. However, the expectation of all basis functions must preferably be known in order to apply the regress-later technique efficiently.

In the context of our algorithm, we define the bundle-wise regression parameters $\alpha_{p+1}(b), \beta_{p+1}(b), \gamma_{p+1}(b)$ as

$$\alpha_{p+1}(b) = \arg\min_{\alpha\in\mathbb{R}^Q} \frac{\sum_{m=1}^M(\eta(X^{\pi,m}_{t_{p+1}})\alpha - y^R_{p+1}(X^{\pi,m}_{p+1}))^2\mathbf{1}_{\mathscr{B}_{t_p}(b)}(X^{\pi,m}_{t_p})}{\sum_{m=1}^M\mathbf{1}_{\mathscr{B}_{t_p}(b)}(X^{\pi.m}_{t_p})},$$

$$\beta_{i,p+1}(b) = \arg\min_{\beta\in\mathbb{R}^Q} \frac{\sum_{m=1}^{M}(\eta(X_{t_{p+1}}^{\pi,m})\beta - z_{i,p+1}^{R}(X_{p+1}^{\pi,m}))^2 \mathbf{1}_{\mathscr{B}_{t_p}(b)}(X_{t_p}^{\pi,m})}{\sum_{m=1}^{M}\mathbf{1}_{\mathscr{B}_{t_p}(b)}(X_{t_p}^{\pi.m})},$$

$$\gamma_{p+1}(b) = \arg\min_{\gamma\in\mathbb{R}^Q} \frac{\sum_{m=1}^{M}(\eta(X_{t_{p+1}}^{\pi,m})\gamma - f_{p+1}(y_{p+1}^{R}(X_{p+1}^{\pi,m}), z_{p+1}^{R}(X_{p+1}^{\pi,m})))^2 \mathbf{1}_{\mathscr{B}_{t_p}(b)}(X_{t_p}^{\pi,m})}{\sum_{m=1}^{M}\mathbf{1}_{\mathscr{B}_{t_p}(b)}(X_{t_p}^{\pi.m})}.$$

The approximate functions within the bundle at time p are defined by the above parameters and the expectations $\mathbb{E}_{t_p}^{x}[\eta(X_{t_{p+1}}^{\pi})]$ and $\mathbb{E}_{t_p}^{x}\left[\eta(X_{t_{p+1}}^{\pi})\frac{\Delta W_{r,p}}{\Delta_p}\right]$:

$$z_{r,p}^{R}(b,x) = \mathbb{E}_{t_p}^{x}\left[\frac{\Delta W_{r,p}}{\Delta_p}\eta(X_{t_{p+1}}^{\pi})\right]\alpha_{p+1}(b), \quad r=1,\dots,d;$$

$$y_{p}^{R}(b,x) = \mathbb{E}_{t_p}^{x}\left[\eta(X_{t_{p+1}}^{\pi})\right](\alpha_{p+1}(b) + \Delta_p\gamma_{p+1}(b)).$$

This comes from putting the regression results back in Equation (4.1).

As the expectations related to the basis functions are the foundation of any regress-later scheme, we assume that the following assumptions are satisfied.

Assumption 4.2. The regression basis $\{\eta_1,\dots,\eta_Q\}$ is assumed to satisfy the following assumptions.

($\mathbf{A}_\eta$) i.) $\mathbb{E}_{t_p}^{x}[\eta_l(X_{t_{p+1}}^{\pi})]$ and $\mathbb{E}_{t_p}^{x}\left[\eta_l(X_{t_{p+1}}^{\pi})\frac{\Delta W_{r,p}}{\Delta_p}\right]$ are known, either analytically or empirically, for all $p=0,\dots P-1$, $l=1,\dots,Q$ and $r=1,\dots,d$.

ii.) For any given compact set $\mathscr{A}$ in $\mathbb{R}^q$, the constant $C_{\eta,\mathscr{A}} := \max_{l=1,\dots,Q} C_{\eta_l,\mathscr{A}}$. Moreover, there exists a constant $C_{M,\mathscr{A}}$ such that

$$\sum_{l=1}^{Q}\left|\mathbb{E}_{t_p}^{x}[\eta_l(X_{t_{p+1}}^{\pi})]\right| \le C_{M,\mathscr{A}}, \qquad \forall x\in\mathscr{A}, \text{ and } p=0,\dots,P-1;$$

and

$$\sum_{l=1}^{Q}\left|\mathbb{E}_{t_p}^{x}\left[\eta_l(X_{t_{p+1}}^{\pi})\frac{\Delta W_{r,p}}{\Delta_p}\right]\right| \le C_{M,\mathscr{A}}, \qquad \forall x\in\mathscr{A}, \text{ and } p=0,\dots,P-1.$$

Next, to ensure the stability of our algorithm, $|\alpha_p(b)|$, $|\beta_{r,p}(b)|$ and $|\gamma_p(b)|$ must be bounded above for all p,b,r. In practice, this means that an error notion should be given by the program when the Euclidean norm of any regression coefficient vector is greater than a predetermined constant L. Further details on this requirement will be described in Section 4.3.

Finally, to simplify notation, we define the notations below for the regression result across the bundles.

$$\tilde{y}_{p+1}^{R}(x_1,x_2) := \sum_{b=1}^{B}\mathbf{1}_{\mathscr{B}_{t_p}(b)}(x_1)\eta(x_2)\alpha_{p+1}(b),$$

$$\tilde{z}_{r,p+1}^{R}(x_1,x_2) := \sum_{b=1}^{B}\mathbf{1}_{\mathscr{B}_{t_p}(b)}(x_1)\eta(x_2)\beta_{r,p+1}(b),$$

$$\tilde{f}_{p+1}^{R}(x_1,x_2) := \sum_{b=1}^{B}\mathbf{1}_{\mathscr{B}_{t_p}(b)}(x_1)\eta(x_2)\gamma_{p+1}(b).$$

4.3. REFINED REGRESSION

In this section, we derive a proof of an error bound for our regress-later strategy. In order to ensure the stability of our algorithm, we have introduced a sample selection step into the algorithm and modified the classical proof for nonparametric regression from [10], which was used in [9], for the derivation of the error bound to SGBM.

In order to simplify expressions, different notations are used in this section. We consider a random vector (X, Y), where X and Y are both $\mathbb{R}^q$, following the probability measure ν. A cloud of simulation paths can be generated by independently simulating M copies, $\{(X^m, Y^m) : m = 1, \dots, M\}$, defined on a probability space $(\hat{\Omega}, \hat{\mathfrak{F}}, \hat{\mathbb{P}})$. In our content, the pair (X, Y) represents the independent and dependent variables under consideration and (X^m, Y^m) are the simulated samples for (X, Y).

Denote by $\mathbb{B}$ a specific partition with $\mathbb{B} := \{\mathscr{B}(1), \dots, \mathscr{B}(B)\}$ and $\bigcup_{b=1}^{B} \mathscr{B}(b) = \mathbb{R}^d$. The partition which is used in the regression estimates is based on the simulation data X^m in our setting and to which bundle a sample belongs solely depends on X^m.

The main goal of SGBM is finding an effective and accurate way to approximate the expectation $\mathbb{E}[\nu(Y)|X]$ in a recurrence setting for some deterministic function $\nu : \mathbb{R}^q \to \mathbb{R}$, and we begin with establishing an estimate $\tilde{\nu} : \mathbb{R}^q \times \mathbb{R}^q$ for ν. Note that although ν solely depends on the dependent variables, the estimate $\tilde{\nu}$ depends on both the independent and dependent variables in preparation for further calculation.

For a given partition and samples, one way to define the estimate $\tilde{\nu}$ is

$$\tilde{\nu}(x, y) := \sum_{b=1}^{B} \mathbf{1}_{\mathscr{B}(b)}(x)\, \tilde{\nu}_b(y) = \sum_{b=1}^{B} \mathbf{1}_{\mathscr{B}(b)}(x) \sum_{k=1}^{Q} \alpha_k(b) \eta_k(y), \tag{4.2}$$

where

$$\tilde{\nu}_b := \arg\min_{\phi \in H} \left\{ \frac{\sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)}(X^m) |\nu(Y^m) - \phi(Y^m)|^2}{\sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)}(X^m)} \right\}.$$

Remark 4.1. It is possible that under some particular clustering scheme for SGBM, there would be empty bundles in the resulting partition.

In practice, one could simply ignore these empty bundles in the algorithm. As there are no samples in these bundles, approximations within these bundles are not needed for the next time step. One point to note is that since least-squares regression requires a sufficient number of samples to be accurate, adopting a bundling scheme that would produce bundles with few samples may not be a good idea.

When generalizing the theoretical proof below to bundling methods other than equal partition, one has to take this into account and define the measurable partition $\mathbb{B}$ in such a way that it is consistent with the practical bundling scheme while it also merges all empty bundle to non-empty ones.

Further discussion on bundles with few samples under the equal partition scheme is placed in Remark 4.4.

Note that functions $\tilde{\nu}_b : \mathbb{R}^q \to \mathbb{R}$ are stochastic with respect to the simulation samples (X^m, Y^m). The linear vector space H is spanned by continuous functions $\{\eta_1, \dots, \eta_Q\}$, with $\eta_l : \mathbb{R}^q \to \mathbb{R}$, $\forall l = 1, \dots, Q$. Thus, the second equality in Equation (4.2) just follows from the definition of H and typical least-squares regression. In fact, if we denote the

total number of samples in a given bundle by $\#\mathscr{B}(b)$ and let

$$\{(X^{b,1}, Y^{b,1}), \dots, (X^{b,\#\mathscr{B}(b)}, Y^{b,\#\mathscr{B}(b)})\}$$

be the samples in this bundle, the coefficients $\alpha(b)$ satisfy

$$\mathscr{I}^\top \mathscr{I} \alpha(b) = \mathscr{I}^\top v(Y^b), \tag{4.3}$$

with

$$\mathscr{I} = (\eta_j(Y^{b,i}))_{1\le i\le \#\mathscr{B}(b), 1\le j\le Q} \text{ and } v(Y^b) = (v(Y^{b,1}), \dots, v(Y^{b,\#\mathscr{B}(b)}))^\top.$$

According to [10], system (4.3) is always solvable, in the next section, we also provide a heuristic argument for its invertibility. Again, the coefficients α in each bundle can be seen as random variables with respect to $(X^m, Y^m)_{m=1,\dots,M}$.

Reversely, we may select the simulation cloud based on the regression coefficients. Let the set S be the set containing all possible collections of $(x_m, y_m)_{1\le m\le M} \in (\mathbb{R}^q \times \mathbb{R}^q)^M$ such that $|\alpha(b)|^2 \le L$ for all b given that $(X^m, Y^m) = (x_m, y_m)_{1\le m\le M}$. We modify the probability of the simulation cloud by only accepting those results that are in S. We denote the modified expectation by $\hat{\mathbb{E}}_S$ and it is related to the original expectation by $\hat{\mathbb{E}}_S[\mathbf{1}_A] = \frac{\hat{\mathbb{E}}[\mathbf{1}_A \mathbf{1}_S]}{\hat{\mathbb{E}}[\mathbf{1}_S]}$. [1]

Remark 4.2. In a regress-now scheme, especially in a recursion scheme, the resulting approximation is truncated such that its value is within a bounded interval $[\mathscr{C}_1, \mathscr{C}_2]$. The truncation guarantees the convergence and the stability of the scheme. However, truncation is not feasible in our regress-later scheme as we have to keep the full function for further operation. Therefore, we must instead control the output by limiting the admissible samples.

Remark 4.3. The introduction of bundling here essentially serves two purposes. First of all, clustering data may act as a localization of function v, thus a more accurate approximation for v can be achieved with a lower order function basis. This is especially beneficial for the high-dimensional case as basis functions in higher dimensions are generally complicated and hard to calculate. We need a method to increase accuracy without adding more basis functions. Secondly, by partitioning data into non-overlapping bundles, we can facilitate the application of parallel computing, which is important when we are in a high-dimensional situation. However, while the above benefit depends on the particular choice of basis, the analysis we do in this section is applicable for a more general setting. So, we would not emphasis these points further in this section.

Using ν to denote the probability measure induced by the random variable (X, Y) and $(X^m, Y^m)_{m=1,\dots,M}$ are independent and identical copies following the same law under a different probability space, the following random norms (depending on the simulation cloud (X^m, Y^m)) are used to quantify the error of approximation.

Definition 4.3. Let $\mathfrak{f} : \hat{\Omega} \times \mathbb{R}^q \times \mathbb{R}^q \to \mathbb{R}$ be measurable. For any set $\mathscr{B} \subset \mathbb{R}^q$, we define the following random norms

$$||\mathfrak{f}||^2_{\mathscr{B},\infty} := \frac{\int_{\mathscr{B}} \int |\mathfrak{f}(x,y)|^2 \nu(dx,dy)}{\int_{\mathscr{B}} \int \nu(dx,dy)}; \qquad ||\mathfrak{f}||^2_{\mathscr{B},\#} := \frac{\sum_{m=1}^M \mathbf{1}_{\mathscr{B}}(X^m) |\mathfrak{f}(X^m, Y^m)|^2}{\sum_{m=1}^M \mathbf{1}_{\mathscr{B}}(X^m)}.$$

[1] The situation of $\frac{0}{0}$ should be understood as 0 and $\frac{K}{0}$ as ∞ in the rest of this chapter.

We derive the following theorem for the estimation of the error. Since we only accept a simulation result that satisfies event S, we should only consider the average error among all these accepted events.

Theorem 4.4. *Assume that we perform an equal partition at the bundling step, namely, we order all samples according to some specific measurable sorting function on X, and separate them into almost-equal size bundles by the ordering. Further, assume a compact set $\mathscr{A} \subset \mathbb{R}^q$ to be given such that $C_{v,\mathscr{A}} \le \infty$ and $\int v^2(y)\nu(dx,dy) \le \infty$, namely, the function v is within the L^2 space with respect to the given probability measure. Then, for any real function v, we have*

$$\begin{aligned}
&\hat{\mathbb{E}}_S\left[\iint |v(y)-\tilde{v}(x,y)|^2\nu(dx,dy)\right]\\
\le&\frac{\vartheta(L')}{\hat{\mathbb{E}}[\mathbf{1}_S]}\hat{\mathbb{E}}\left[\sum_{b=1}^{B}\int_{\mathscr{B}(b)}\int \nu(dx,dy)\frac{(\log(\sum_{m=1}^{M}\mathbf{1}_{\mathscr{B}(b)}(X^m)-1)+1)(Q+1)}{\sum_{m=1}^{M}\mathbf{1}_{\mathscr{B}(b)}(X^m)-1}\right]\\
&+\hat{\mathbb{E}}_S\left[\sum_{b=1}^{B-1}\int_{\mathscr{B}(b)}\int \nu(dx,dy)\frac{24L'}{(\sum_{m=1}^{M}\mathbf{1}_{\mathscr{B}(b)}(X^m))}\right]\\
&+\frac{12}{\hat{\mathbb{E}}[\mathbf{1}_S]}\hat{\mathbb{E}}\left[\sum_{\mathscr{B}\in\mathbb{B}}\int_{\mathscr{B}}\int \nu(dx,dy)(\inf_{\phi\in H}\sup_{x\in\mathscr{B}}\mathbb{E}\left[|v(Y)-\phi(Y)|^2|X=x\right]\wedge L')\right]\\
&+\hat{\mathbb{E}}_S\left[\iint |v(y)-\tilde{v}(x,y)|^2(1-\mathbf{1}_{\mathscr{A}}(y))\nu(dx,dy)\right],
\end{aligned}$$

for $L' := 2LQC_{\eta,A}^2 + 2C_{v,A}^2$, and $\vartheta(L')$ a function depending on L'. Note that the set $\mathscr{A}$ is introduced to avoid the restrictive assumption of v being bounded. It does not play a role in the actual algorithm.

Proof. To prepare for our analysis, a more formal construction of the equal partition technique needs to be introduced.

In practice, for samples $(X^m, Y^m)_{1\le m\le M}$ and a measurable sorting function $\mathfrak{S}:\mathbb{R}^q\to\mathbb{R}$, the M different values can be ordered into,

$$\mathfrak{S}(X^{1^*}) \le \mathfrak{S}(X^{2^*}) \le \cdots \le \mathfrak{S}(X^{M^*}),$$

by simply putting $\{X^{1^*},\cdots,X^{(M/B)^*}\}$ into the first bundle, $\{X^{(M/B+1)^*},\cdots,X^{(2M/B)^*}\}$ into the second one, etc., assuming M can be divided by B for simplicity.

However, in order to conduct meaningful analysis, a measurable partition of $\mathbb{R}^q$ based on the simulation cloud $(X^m)_{1\le m\le M}$ is required. Thus for any simulation $(X^m)_{1\le m\le M}$ with $\{X^{1^*}=x_1^*,\cdots,X^{M^*}=x_M^*\}$, we define

$$\begin{aligned}
\mathscr{B}(1) := \mathfrak{S}^{-1}((-\infty,\mathfrak{S}(x^*_{M/B})]), \mathscr{B}(2) := \mathfrak{S}^{-1}((\mathfrak{S}(x^*_{M/B}),\mathfrak{S}(x^*_{2M/B})]),\cdots,\\
\mathscr{B}(B) := \mathfrak{S}^{-1}(\mathfrak{S}(x^*_{M-M/B}),\infty)
\end{aligned}$$

and $\cup_{b=1}^{B}\mathscr{B}(b)=\mathbb{R}^q$.

Therefore, $\mathbb{B}=\{\mathfrak{S}^{-1}((-\infty,\mathfrak{S}(x_1)]),\mathfrak{S}^{-1}((\mathfrak{S}(x_1),\mathfrak{S}(x_2)]),\cdots,\mathfrak{S}^{-1}((\mathfrak{S}(x_{B-1}),\infty))\}$ if and only if $(X^{1^@},X^{2^@},\cdots X^{M^@}) \in (\mathfrak{S}^{-1}((-\infty,\mathfrak{S}(x_1))))^{M/B-1}\times\{x_1\}\times(\mathfrak{S}^{-1}((x_1,x_2)))^{M/B-1}\times\{x_2\}\times$

$\cdots \times \{x_{B-1}\} \times (\mathfrak{S}^{-1}((\mathfrak{S}(x_{B-1}),\infty)))^{M/B-1}$. The notation @ denotes any permutation of the set $\{1,2,\cdots M\}$, noting that each sample is independent of the others and interchangeable. This is measurable with respect to the sigma algebra generated by the simulation cloud X^m as there are finite permutations for fixed M and $\mathfrak{S}$ is measurable.

Note that this setting is not unique for defining a workable partition and there may be alternative definitions that may improve the analysis result. However, this is an intuitive definition.

Assuming $\sigma(\mathbb{B})$ to be the smallest sigma algebra to determine the partition, we notice that it is smaller than the sigma algebra generated by the random samples X^m, $\sigma(\mathbb{B}) \subset \sigma(X^m)$. This is because multiple realizations of the samples can lead to the same partition. A simple thought experiment is to consider a fixed partition, and subsequently move one interior sample within a bundle. If we conduct a new bundling with this new set of samples, the partition will remain the same. Indeed, the samples within a bundle are independent among each other and have the same distribution.

As for the actual analysis, we start by decomposing the error into different terms for any given partition

$$\mathbb{B} = \{\mathscr{B}(1),\cdots,\mathscr{B}(B)\} = \{\mathfrak{S}^{-1}((-\infty,\mathfrak{S}(x_1)]),\mathfrak{S}^{-1}((\mathfrak{S}(x_1),\mathfrak{S}(x_2)]),\cdots,\mathfrak{S}^{-1}((\mathfrak{S}(x_{B-1}),\infty))\}.$$

In line with the Monte Carlo literature, we assume $X^i \neq X^j$, if $i \neq j$, and

$$\begin{aligned}
&\iint |v(y)-\tilde{v}(x,y)|^2 \nu(dx,dy) \\
\leq &\sum_{\mathscr{B}\in\mathbb{B}} \int_{\mathscr{B}} \int |v(y)-\tilde{v}(x,y)|^2 \mathbf{1}_{\mathscr{A}}(y)\nu(dx,dy) + \iint |v(y)-\tilde{v}(x,y)|^2(1-\mathbf{1}_{\mathscr{A}}(y))\nu(dx,dy) \\
= &\sum_{b=1}^{B} \int_{\mathscr{B}(b)} \int \nu(dx,dy)\Big(||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b),\infty} - 2||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b)\setminus\{x_b\},\#} \\
&\qquad + 2||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b)\setminus\{x_b\},\#} - 2||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b),\#} + 2||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b),\#}\Big)^2 \\
&+ \iint |v(y)-\tilde{v}(x,y)|^2(1-\mathbf{1}_{\mathscr{A}}(y))\nu(dx,dy) \\
\leq &\sum_{b=1}^{B} \int_{\mathscr{B}(b)} \int \nu(dx,dy)\Big(\max\{||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b),\infty} - 2||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b)\setminus\{x_b\},\#}, 0\} \\
&\qquad + 2||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b)\setminus\{x_b\},\#} - 2||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b),\#} + 2||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b),\#}\Big)^2 \\
&+ \iint |v(y)-\tilde{v}(x,y)|^2(1-\mathbf{1}_{\mathscr{A}}(y))\nu(dx,dy) \\
\leq &\sum_{b=1}^{B} \int_{\mathscr{B}(b)} \int \nu(dx,dy) 3\max\{||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b),\infty} - 2||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b)\setminus\{x_b\},\#}, 0\}^2 \\
&+ \sum_{b=1}^{B-1} \int_{\mathscr{B}(b)} \int \nu(dx,dy) 12(||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b)\setminus\{x_b\},\#} - ||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b),\#})^2 \\
&+ \sum_{\mathscr{B}\in\mathbb{B}} 12 \int_{\mathscr{B}} \int \nu(dx,dy)||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||^2_{\mathscr{B},\#} + \iint |v(y)-\tilde{v}(x,y)|^2(1-\mathbf{1}_{\mathscr{A}}(y))\nu(dx,dy)
\end{aligned}$$

$$=: \sum_{b=1}^{B} \int_{\mathscr{B}(b)} \int \nu(dx,dy) T_{1,\mathscr{B}(b)} + \sum_{b=1}^{B-1} \int_{\mathscr{B}(b)} \int \nu(dx,dy) T_{2,\mathscr{B}(b)}$$
$$+ \sum_{\mathscr{B}\in\mathbb{B}} \int_{\mathscr{B}} \int \nu(dx,dy) T_{3,\mathscr{B}} + \iint |v(y)-\tilde{v}(x,y)|^2 (1-\mathbf{1}_{\mathscr{A}}(y)) \nu(dx,dy), \tag{4.4}$$

with a slight abuse of notation, $\{x_B\} := \emptyset$ above.

Note that the easiest way to conceptualize the term $||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b)\setminus\{x_b\},\#}$ is that we simply remove the sample that is used for defining the partition. Thus,

$$\sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)\setminus\{x_b\}}(X^m) = \sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)}(X^m) - 1.$$

This is done to ensure that all samples in the empirical norm $||\cdot||_{\mathscr{B}(b)\setminus\{x_b\},\#}$ are independent of each other.

As the last term cannot be further simplified, we now focus on the first three terms. The meaning of all error terms will be discussed in the next subsection.

The first term we study is $T_{3,\mathscr{B}}$, which represents the best possible approximation from the space H to the target function under the empirical norm within the bundle. To begin, within any bundle $\mathscr{B}(b)$ under any given partition $\mathbb{B}$, it is obvious that

$$||\mathbf{1}_{\mathscr{A}}(v-\tilde{v})||_{\mathscr{B}(b),\#} \le ||v-\tilde{v}_b||_{\mathscr{B}(b),\#} = \min_{\phi\in H} ||v-\phi||_{\mathscr{B}(b),\#},$$

for any $\mathscr{B}(b)$. Only the $\tilde{v}_b$ term in the series of $\tilde{v}$ matters and $\tilde{v}_b$ is the function that minimizes the approximation difference under the empirical norm within the given bundle.

Alternatively, we may consider the following composite norm

$$\sup_{x\in\mathscr{B}(b)} \left(\mathbb{E}\left[(\mathfrak{f}(Y))^2 | X=x\right]\right)^{\frac{1}{2}}, \tag{4.5}$$

for any given bundle $\mathscr{B}(b)$. For the sake of simplicity, we assume there exists an element $\phi_{\mathscr{B}(b)}$ within the space H such that $v(\cdot)-\phi_{\mathscr{B}(b)}(\cdot)$ minimizes the norm, namely,

$$\sup_{x\in\mathscr{B}(b)} \mathbb{E}\left[|v(Y)-\phi_{\mathscr{B}(b)}(Y)|^2 | X=x\right] = \inf_{\phi\in H} \sup_{x\in\mathscr{B}(b)} \mathbb{E}\left[|v(Y)-\phi(Y)|^2 | X=x\right].$$

As $\phi_{\mathscr{B}(b)} \in H$, it is clear that under the empirical norm, we have

$$||\mathbf{1}_{\mathscr{A}}(v-\tilde{v})||^2_{\mathscr{B}(b),\#} \le ||v-\tilde{v}_b||^2_{\mathscr{B}(b),\#} \le ||v-\phi_{\mathscr{B}(b)}||^2_{\mathscr{B}(b),\#} = \frac{\sum_{m=1}^{\#\mathscr{B}(b)} |v(Y^{b,m})-\phi_{\mathscr{B}}(Y^{b,m})|^2}{\#\mathscr{B}(b)}.$$

Without loss of generality, assume $(X^{b,\#\mathscr{B}(b)}, Y^{b,\#\mathscr{B}(b)})$ is the bundle defining the sample as stated in the construction of an equal partition if $b \neq B$. Recalling that samples within a bundle are i.i.d. given the partition, we can take the conditional expectation of the empirical norm with respect to the position of $(X^{b,\#\mathscr{B}(b)}, Y^{b,\#\mathscr{B}(b)})_{1\le m\le \#\mathscr{B}(b)-1}$.

$$\hat{\mathbb{E}}\left[||\mathbf{1}_{\mathscr{A}}(v-\tilde{v})||^2_{\mathscr{B},\#} \middle| \sigma(\mathbb{B}), X^{b,1}, X^{b,2}, \cdots, X^{b,\#\mathscr{B}(b)-1} \right]$$
$$\le \frac{\sum_{m=1}^{\#\mathscr{B}(b)} \hat{\mathbb{E}}\left[|v(Y^{b,m})-\phi_{\mathscr{B}}(Y^{b,m})|^2 \middle| \sigma(\mathbb{B}), X^{b,1}, \cdots, X^{b,\#\mathscr{B}(b)-1} \right]}{\#\mathscr{B}(b)}$$

$$
\begin{aligned}
&= \frac{\sum_{m=1}^{\#\mathscr{B}(b)-1} \mathbb{E}\left[\,|v(Y)-\phi_{\mathscr{B}}(Y)|^2 \,\middle|\, X = X^{b,m} \in \mathscr{B}(b)\backslash\{x_b\}\right] + \mathbb{E}\left[\,|v(Y)-\phi_{\mathscr{B}}(Y)|^2 \,\middle|\, X = x_b\right]}{\#\mathscr{B}(b)} \\
&\leq \frac{\sum_{m=1}^{\#\mathscr{B}(b)} \sup_{x\in\mathscr{B}(b)} \mathbb{E}\left[\,|v(Y)-\phi_{\mathscr{B}}(Y)|^2 \,\middle|\, X = x\right]}{\#\mathscr{B}(b)} \\
&= \inf_{\phi\in H} \sup_{x\in\mathscr{B}(b)} \mathbb{E}\left[\,|v(Y)-\phi(Y)|^2 \,\middle|\, X = x\right].
\end{aligned}
$$

There are some details in the above calculation that require explanation. Note that the boundary point information is included in $\sigma(\mathbb{B})$, therefore we only calculate conditional expectations with the remaining samples. For the last bundle $\mathscr{B}(B)$, the separation is not necessary as no sample is used to define the bundle, but this does not alter the result. We use the fact that each sample is independent within the bundle in the first equality.

Next, if the minimal element $\phi_{\mathscr{B}(b)}$ does not exist, one has to adjust the derivation above with a limiting argument. This means that by the definition of the infimum, one can find a sequence of functions $(v-\phi_{\mathscr{B}(b),n})_{n\in\mathbb{Z}^+}$, such that

$$
\sup_{x\in\mathscr{B}(b)} \mathbb{E}\left[|v(Y)-\phi_{\mathscr{B}(b),n}(Y)|^2 | X = x\right] \leq \inf_{\phi\in H} \sup_{x\in\mathscr{B}(b)} \mathbb{E}\left[|v(Y)-\phi(Y)|^2 | X = x\right] + \frac{1}{n}.
$$

By repeating the above argument for each function in the sequence, replacing the infimum in the proof by the corresponding upper bound and taking n to infinity (with the eventual inequality), we arrive at the same conclusion.

Thereafter, if we consider the expectation of the empirical norm conditioning on $\sigma(\mathbb{B})$, we have

$$
\begin{aligned}
\hat{\mathbb{E}}\left[\,||\mathbf{1}_{\mathscr{A}}(v-\tilde{v})||^2_{\mathscr{B},\#} \,\middle|\, \sigma(\mathbb{B})\right] =& \hat{\mathbb{E}}\left[\,|\hat{\mathbb{E}}\left[\,||\mathbf{1}_{\mathscr{A}}(v-\tilde{v})||^2_{\mathscr{B},\#} \,\middle|\, \sigma(\mathbb{B}), X^{b,1}, X^{b,2}, \cdots, X^{b,\#\mathscr{B}(b)-1}\right] \,\middle|\, \sigma(\mathbb{B})\right] \\
&\leq \inf_{\phi\in H} \sup_{x\in\mathscr{B}(b)} \mathbb{E}\left[\,|v(Y)-\phi(Y)|^2 \,\middle|\, X = x\right].
\end{aligned}
$$

Note that this bound is defined on all given partitions and solely depends on the partition but not on the choice of $\phi_{\mathscr{B}}$ for any bundle $\mathscr{B}$. Our calculation here is purely within a bundle given the partition is known. Therefore, even if the minimum function $\phi_{\mathscr{B}(b)}$ or the sequence $(\phi_{\mathscr{B}(b),n})_{n\in\mathbb{Z}^+}$ is not unique, the actual choice of these functions does not matter as long as they are picked in a consistent and measurable way.

Here, we derive a bound for the expectation of the weight summation $T_{3,\mathscr{B}}$ in (4.4) with respect to the simulation cloud, i.e.,

$$
\begin{aligned}
&\hat{\mathbb{E}}_S\left[\sum_{\mathscr{B}\in\mathbb{B}} \int_{\mathscr{B}}\int v(dx,dy) T_{3,\mathscr{B}}\right] \\
&\leq 12\hat{\mathbb{E}}_S\left[\sum_{\mathscr{B}\in\mathbb{B}} \int_{\mathscr{B}}\int v(dx,dy) ||v-\phi_{\mathscr{B}}||^2_{\mathscr{B},\#}\right] \\
&\leq \frac{12}{\hat{\mathbb{E}}[\mathbf{1}_S]} \hat{\mathbb{E}}\left[\sum_{\mathscr{B}\in\mathbb{B}} \int_{\mathscr{B}}\int v(dx,dy) \hat{\mathbb{E}}\left[\,||v-\phi_{\mathscr{B}}||^2_{\mathscr{B},\#} \,\middle|\, \sigma(\mathbb{B})\right]\right] \qquad (4.6) \\
&\leq \frac{12}{\hat{\mathbb{E}}[\mathbf{1}_S]} \hat{\mathbb{E}}\left[\sum_{\mathscr{B}\in\mathbb{B}} \int_{\mathscr{B}}\int v(dx,dy) \inf_{\phi\in H} \sup_{x\in\mathscr{B}} \mathbb{E}\left[|v(Y)-\phi(Y)|^2 | X = x\right]\right].
\end{aligned}
$$

In this inequality, we expand the denominator of our adjusted probability by also including the rejected cases and applying the results above for each partition.

However, for an unbounded bundle $\mathscr{B}$, it is possible to find an example such that $\sup_{x\in\mathscr{B}}\mathbb{E}\left[|v(Y)-\phi(Y)|^2|X=x\right]=\infty$. An alternative bound is required to ensure that our error bound is not trivial. Note that given the square norm $|\alpha(b)|^2\le L$, we have

$$\forall y,b,|\tilde{v}_{\mathscr{B}(b)}(y)\mathbf{1}_{\mathscr{A}}(y)|^2\le\left(\sum_{l=1}^{Q}|\alpha_l(b)|^2\right)\left(\sum_{l=1}^{Q}|\eta_l(y)\mathbf{1}_{\mathscr{A}}(y)|^2\right)\le LQ\max_{l=1\ldots,Q}\max_{y\in\mathscr{A}}|\eta_l(y)|^2,$$

and

$$\begin{aligned}||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||^2_{\mathscr{B},\#}&=\frac{\sum_{m=1}^{M}\mathbf{1}_{\mathscr{B}}(X^m)|(v(Y^m)-\phi_{\mathscr{B}}(Y^m))\mathbf{1}_{\mathscr{A}}|^2}{\sum_{m=1}^{M}\mathbf{1}_{\mathscr{B}}(X^m)}\\&\le\frac{\sum_{m=1}^{M}\mathbf{1}_{\mathscr{B}}(X^m)(2|(v(Y^m)\mathbf{1}_{\mathscr{A}}|^2+2|\phi_{\mathscr{B}}(Y^m))\mathbf{1}_{\mathscr{A}}|^2}{\sum_{m=1}^{M}\mathbf{1}_{\mathscr{B}}(X^m)}\\&\le\frac{\sum_{m=1}^{M}\mathbf{1}_{\mathscr{B}}(X^m)(2C^2_{v,\mathscr{A}}+2LQC^2_{\eta,\mathscr{A}})}{\sum_{m=1}^{M}\mathbf{1}_{\mathscr{B}}(X^m)}=L'.\end{aligned}$$

We shall use this alternative bound in Equation (4.6) for any bundle $\mathscr{B}$ such that $\sup_{x\in\mathscr{B}}\mathbb{E}\left[|v(Y)-\phi(Y)|^2|X=x\right]>L'$. Combining the two error bounds, we have

$$\hat{\mathbb{E}}_S\left[\sum_{\mathscr{B}\in\mathbb{B}}T_{3,\mathscr{B}}\right]\le\frac{12}{\hat{\mathbb{E}}[\mathbf{1}_S]}\hat{\mathbb{E}}\left[\sum_{\mathscr{B}\in\mathbb{B}}\int_{\mathscr{B}}\int v(dx,dy)(\inf_{\phi\in H}\sup_{x\in\mathscr{B}}\mathbb{E}\left[|v(Y)-\phi(Y)|^2|X=x\right]\wedge L')\right].$$

Next, we consider the term $T_{1,\mathscr{B}}$ in (4.4). This term concerns the difference between the theoretical projection and the empirical regression function within each bundle. Here we restate that S denotes the modified probability, based on the regression coefficients, where $\mathscr{A}$ is a compact set defined with respect to Y only.

By taking conditional expectations with respect to $\sigma(\mathbb{B})$, we have,

$$\begin{aligned}&\hat{\mathbb{E}}_S\left[\sum_{b=1}^{B}\int_{\mathscr{B}(b)}\int v(dx,dy)T_{1,\mathscr{B}(b)}\right]\\&=\frac{1}{\hat{\mathbb{E}}[\mathbf{1}_S]}\hat{\mathbb{E}}\left[\sum_{b=1}^{B}\int_{\mathscr{B}(b)}\int v(dx,dy)\hat{\mathbb{E}}[T_{1,\mathscr{B}(b)}\mathbf{1}_S|\sigma(\mathbb{B})]\right]\\&=\frac{1}{\hat{\mathbb{E}}[\mathbf{1}_S]}\hat{\mathbb{E}}\left[\sum_{b=1}^{B}\int_{\mathscr{B}(b)}\int v(dx,dy)\times\right.\\&\qquad\left.\hat{\mathbb{E}}_{\mathscr{B}(b)}[\mathbf{1}_S3\max\{||\mathbf{1}_{\mathscr{A}}(v-\tilde{v})||_{\mathscr{B}(b),\infty}-2||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b)\setminus\{x_b\},\#},0\}^2]\right].\end{aligned}$$

It is important that we condition on the smaller sigma algebra such that all samples have an identical conditional distribution. If we can condition on the whole sigma algebra generated by $(X^m)_{1\le m\le M}$, each Y^m will have a different distribution depending on the position of X^m. Within each (given) bundle $\mathscr{B}(b)$, we may consider the two norms

$||\cdot||_{\mathscr{B}(b),\infty}$ and $||\cdot||_{\mathscr{B}(b)\setminus\{x_b\},\#}$ as the theoretical and empirical L^2 norms of a random process satisfying the probability distribution $\mathbb{P}_{\mathscr{B}(b)} := \frac{\int_{\mathscr{B}(b)} \nu(dx,\cdot)}{\int_{\mathscr{B}(b)}\int \nu(dx,dy)}$ and extend our notation for expectations to this measure. In other words, $\mathbb{P}_{\mathscr{B}(b)}$ is the conditional probability of Y^m given that X^m is within the bundle. As only the samples within bundle $\mathscr{B}(b)$ are considered in $T_{1,\mathscr{B}(b)}$, we only have to consider the identically distributed samples following $\mathbb{P}_{\mathscr{B}(b)}$. Thus, we simplify the notation with this measure.

Assume that $\sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)\setminus\{x_b\}}(X^m) = N-1$ and let $u > 864L'/(N-1)$ be arbitrary, by Theorem 11.2 in [10], we find

$$\begin{aligned}
&\hat{\mathbb{P}}_{\mathscr{B}(b)}\{3\max\{||\mathbf{1}_{\mathscr{A}}(\nu-\tilde{\nu})||_{\mathscr{B}(b),\infty} - 2||(\nu-\tilde{\nu})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b)\setminus\{x_b\},\#}, 0\}^2 > u \text{ and event } S \text{ is true}\}\\
\leq&\hat{\mathbb{P}}_{\mathscr{B}(b)}\{\exists \phi \in H_L : ||\mathbf{1}_{\mathscr{A}}(\nu-\phi)||_{\mathscr{B}(b),\infty} - 2||(\nu-\phi)\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b)\setminus\{x_b\},\#} > \sqrt{u/3} \text{ and } S \text{ is true}\}\\
\leq&\hat{\mathbb{P}}_{\mathscr{B}(b)}\{\exists \phi \in H_L : ||\mathbf{1}_{\mathscr{A}}(\nu-\phi)||_{\mathscr{B}(b),\infty} - 2||(\nu-\phi)\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b)\setminus\{x_b\},\#} > \sqrt{u/3}\}\\
\leq&3\hat{\mathbb{E}}_{\mathscr{B}(b)}[\mathscr{N}_2(\sqrt{2/3}\sqrt{u}/24, H_{L,\mathscr{A}}, Y^{2(N-1)}_{\mathscr{B}(b)})]\exp\left(-\frac{(N-1)u}{864L'}\right)\\
\leq&3\hat{\mathbb{E}}_{\mathscr{B}(b)}[\mathscr{N}_2(\sqrt{L'}/\sqrt{N-1}, H_{L,\mathscr{A}}, Y^{2(N-1)}_{\mathscr{B}(b)})]\exp\left(-\frac{(N-1)u}{864L'}\right),
\end{aligned} \tag{4.7}$$

where H_L is the set of all functions in H whose coordinates with respect to the basis $(\eta_l)_{1\leq l\leq Q}$, have a Euclidean norm no greater than L and $H_{L,\mathscr{A}}$ the set containing all functions of the form $\mathbf{1}_{\mathscr{A}}(\phi-\nu)$, where ϕ belongs to H_L. Again, since we condition only on the partition, all samples within a bundle are i.i.d. and the condition for Theorem 11.2 in [10] is satisfied. In fact, this proof should work for all partitions for which the samples remain i.i.d. within a bundle. Note that the indicator for the event S is kept in the first line to keep our regression function bounded, then we drop the indicator in the second inequality to take advantage of the independent samples. Finally, $Y_{\mathscr{B}(b)}$ is a sample following the conditional probability $\mathbb{P}_{\mathscr{B}}$ here.

Constant $\mathscr{N}_2$ in (4.7) is called the covering number and it is bounded by Lemma 9.2 and Theorem 9.4 of [10]:

$$\begin{aligned}
&\mathscr{N}_2(\sqrt{L'}/\sqrt{N-1}, H_{L,\mathscr{A}}, Y^{2(N-1)})\\
\leq&3\left(\frac{2eL'}{L'/(N-1)}\log(\frac{3eL'}{L'/(N-1)})\right)^{V_{H^+_{L,\mathscr{A}}}} \leq 3\left[3e(N-1)\right]^{2V_{H^+_{L,\mathscr{A}}}},
\end{aligned}$$

where V denotes the Vapnik-Chervonenkis dimension, which represents the number of elements in the largest set that can be shattered by a class of subsets in $\mathbb{R}^q$. As the details for the Vapnik-Chervonenkis dimension are beyond the scope of this thesis, here we simply mention the results for its bound from [10] and show the necessary justification for these results to be applied here. The reader is referred to section 9.4 of [10] for further information on $\mathscr{N}_2$ and V, the meaning of shattering in this context and more.

Recalling the definition H^+ from Section 4.1, we notice that $V_{H^+_{L,\mathscr{A}}} \leq V_{H^+_L}$, which can be shown by the following argument. Let $(y,z) \in \mathbb{R}^q \times \mathbb{R}$, if $y \notin \mathscr{A}$ and $z \geq 0$, then (y,z) is contained in none of the sets in $H^+_{L,\mathscr{A}}$ and if $y \notin \mathscr{A}$ and $z \leq 0$, then (y,z) is contained in each set of $H^+_{L,\mathscr{A}}$. Hence, if $H^+_{L,\mathscr{A}}$ shatters a set of points, then the x-coordinates of these points must lie in $\mathscr{A}$ and H^+_L also shatters this set of points.

In addition, we have the fact that $H_L \subset H$ and observe that

$$H^+ \subseteq \{\{(x,t) : \phi(x) + a_0 t \geq 0\} : \phi \in H, a_0 \in \mathbb{R}\},$$

which is a linear vector space of dimension less than or equal to $Q+1$, thus Theorem 9.5 of [10] implies

$$V_{H_L^+} \leq Q+1.$$

It follows that, for any $u > 864L'/(N-1)$, the probability under consideration is bounded by $9[3e(N-1)]^{2(Q+1)} \exp\left(-\frac{(N-1)u}{864L'}\right)$, and for any $w > 864L'/(N-1)$,

$$\begin{aligned}
&\hat{\mathbb{E}}[T_{1,\mathscr{B}(b)} \mathbf{1}_S | \sigma(\mathbb{B}), \sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)\setminus\{x_b\}}(X^m) = N-1] \\
\leq & w + 9[3e(N-1)]^{2(Q+1)} \int_w^\infty \exp\left(-\frac{(N-1)t}{864L'}\right) dt \\
= & w + 9[3e(N-1)]^{2(Q+1)} \frac{864L'}{N-1} \exp\left(-\frac{(N-1)w}{864L'}\right).
\end{aligned}$$

By setting,

$$w = \frac{864L'}{N-1} \log\left(9[3e(N-1)]^{2(Q+1)}\right),$$

and taking expectations with respect to $\sigma(\mathbb{B})$, we find

$$\begin{aligned}
&\hat{\mathbb{E}}_S \left[\sum_{b=1}^{B} \int_{\mathscr{B}(b)} \int \nu(dx,dy) T_{1,\mathscr{B}(b)} \right] \\
\leq & \frac{\vartheta(L')}{\hat{\mathbb{E}}[\mathbf{1}_S]} \hat{\mathbb{E}} \left[\sum_{b=1}^{B} \int_{\mathscr{B}(b)} \int \nu(dx,dy) \frac{(\log(\sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)}(X^m) - 1) + 1)(Q+1)}{\sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)}(X^m) - 1} \right],
\end{aligned}$$

where one possible choice of ϑ is $\vartheta(L') := 1728(\log(27e)+1)L'$. This can be checked by simple algebra. Note that ϑ is independent of the number of samples in a bundle and only depends on L'.

Finally, $T_{2,\mathscr{B}(b)}$ is the technical term introduced by the definition of the partition $\mathbb{B}$. Consider any realization of the simulation cloud (X^m, Y^m) and partition $\mathbb{B}$, in particular, there exist boundary defining samples $(x_b, y_b) \in (X^m, Y^m)$ for $b = 1, 2, \cdots, B-1$. Using the inequality $(\sqrt{a} - \sqrt{b})^2 \leq |a-b|$ and the definition of L', we have

$$\begin{aligned}
&12(||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b)\setminus\{x_b\},\#} - ||(v-\tilde{v})\mathbf{1}_{\mathscr{A}}||_{\mathscr{B}(b),\#})^2 \\
\leq & 12 \left| \frac{\sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)\setminus\{x_b\}}(X^m) |(v(Y^m) - \tilde{v}(X^m, Y^m)\mathbf{1}_{\mathscr{A}}(Y^m)|^2}{\sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)}(X^m) - 1} \right. \\
& \left. - \frac{\sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)}(X^m) |(v(Y^m) - \tilde{v}(X^m, Y^m)\mathbf{1}_{\mathscr{A}}(Y^m)|^2}{\sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)}(X^m)} \right| \\
\leq & 12 \times \\
& \left| \frac{\sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)\setminus\{x_b\}}(X^m) (|(v(Y^m) - \tilde{v}(X^m, Y^m)\mathbf{1}_{\mathscr{A}}(Y^m)|^2 - |(v(y_m) - \tilde{v}(x_m, y_m)\mathbf{1}_{\mathscr{A}}(y_m)|^2)}{(\sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)}(X^m) - 1)(\sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)}(X^m))} \right|
\end{aligned}$$

$$\leq \frac{24L'}{(\sum_{m=1}^{M} \mathbf{1}_{\mathscr{B}(b)}(X^m))},$$

for $b = 1, 2, \cdots, B-1$.

By substituting all the results above into Equation (4.4) we conclude the proof.

□

Remark 4.4. Implicitly, it is assumed in our proof that the sorting function used behaves nicely such that there is no empty bundle or bundle with a few samples in our partition. If this is not the case, these bundles need to be merged with other bundles in a consistent way and the number of bundles needs to be adjusted accordingly. We omit this extra complexity in favor of the presentation.

4.3.1. DISCUSSION ON THE ERROR BOUND

We shall discuss the meaning of all error terms in Theorem 4.4. Note that most discussions here are in heuristic sense instead of rigorous analysis.

The last term in the sum $\hat{\mathbb{E}}_S\left[\iint |v(y) - \tilde{v}(x,y)|^2 (1 - \mathbf{1}_{\mathscr{A}}(y)) v(dx, dy)\right]$ represents an expectation with respect to an integration of the approximation error using the probability measure of (X, Y) outside a given compact set $\mathscr{A}$ for Y. In theory, for an increasing series of compact sets $\mathscr{A}_1 \subset \mathscr{A}_2 \subset \ldots \subset \mathscr{A}_A \subset \mathbb{R}^q$,

$$\lim_{A\to\infty} \iint |v(y) - \tilde{v}(x,y)|^2 (1 - \mathbf{1}_{\mathscr{A}_A}(y)) v(dx, dy) \to 0,$$

since the original function and the approximant are both in $L^2(v)$ and the dominating convergent theorem. Again, the set $\mathscr{A}$ plays no role in the algorithm. This term is introduced to reflect that only the area with high probability measure has strong impact on a Monte Carlo approximation, thus, we can only consider the behavior of function v in this area and do not impose strong conditions on v over the whole domain. In practice, we can find a big enough set $\mathscr{A}$ such that the last term is smaller than any preset tolerated level. Otherwise, the probability distribution is too much spread out so that a Monte-Carlo technique may not be a suitable approximation method.

The term $\frac{12}{\hat{\mathbb{E}}[\mathbf{1}_S]} \hat{\mathbb{E}}\left[\sum_{\mathscr{B}\in\mathbb{B}} \int_{\mathscr{B}} \int v(dx, dy) (\inf_{\phi\in H} \sup_{x\in\mathscr{B}} \mathbb{E}\left[|v(Y) - \phi(Y)|^2 | X = x\right] \wedge L')\right]$ can be seen as the average of the best projection error among bundles and upper bounded by L'. It concerns how close the original function and its projection onto the space spanned by our basis are. This term should be controlled by increasing the number of bundles. As the number of bundles increases, all the bundles converge to a point and the error term becomes $\frac{12}{\hat{\mathbb{E}}[\mathbf{1}_S]} \hat{\mathbb{E}}\left[(\inf_{\phi\in H} \mathbb{E}\left[|v(Y) - \phi(Y)|^2 | X\right] \wedge L')\right]$, which is the average projection error over the whole range of X. It is clear that $\hat{\mathbb{E}}[\mathbf{1}_S]$ might change when we increase the number of bundles, thus the above analysis is by no mean rigorous. We will provide comments on S at the end of this section.

The first error term is the bound for the estimation error based on the empirical norm instead of the theoretical norm. This term can be controlled by simply increasing the number of samples within each bundle such that this term is below a certain threshold. Because low sample numbers imply high error bounds, if there are bundles in any partition which contain very few samples, they should be merged with other bundles or the sorting function must be adapted.

The second error term is just a technical term for constructing a measurable partition and can be controlled by the number of samples in each bundle.

Therefore, the best way to set the parameters for the SGBM algorithm is to first fix the number of samples in each bundle such that the first two terms in the error bound are under a given threshold, then increase the number of bundles to control the projection error. However, we cannot write a simple convergence rate for the combined error under the current conditions.

4.3.2. DISCUSSION ON EVENT S

There is one final question remaining. The above argument heavily depends on L, which is a user defined quantity, and the probability of S, the event that L^2 norms of the regression coefficients are below threshold L. It is natural to ask if we can actually find a number L such that $\mathbb{P}_S$ is bounded below, as our bound becomes trivial when $\frac{1}{\hat{\mathbb{E}}[\mathbf{1}_S]}$ tends to infinity and it would be incredibly expensive to apply the algorithm if we reject most of the simulations. In the following, we shall provide a heuristic argument for the convergence of the regression coefficients within the bundle, therefore, there is a natural choice of L depending on the target function itself and a hard cut off may be unnecessary. The numerical experiments in the next section also back up this argument.

Once again, we consider Equation (4.3), where the regression coefficients within any bundle $\mathscr{B}(b)$ satisfy

$$\mathscr{I}^\top \mathscr{I}\, \alpha(b) = \mathscr{I}^\top v(Y^b),$$

with

$$\mathscr{I}^\top \mathscr{I} = \begin{pmatrix} \sum_{i=1}^{\#\mathscr{B}(b)} (\eta_1(Y^{b,i}))^2 & & \sum_{i=1}^{\#\mathscr{B}(b)} \eta_1(Y^{b,i})\eta_Q(Y^{b,i}) \\ & \ddots & \\ \sum_{i=1}^{\#\mathscr{B}(b)} \eta_Q(Y^{b,i})\eta_1(Y^{b,i}) & & \sum_{i=1}^{\#\mathscr{B}(b)} (\eta_Q(Y^{b,i}))^2 \end{pmatrix}$$

and

$$\mathscr{I}^\top v(Y^b) = \begin{pmatrix} \sum_{i=1}^{\#\mathscr{B}(b)} \eta_1(Y^{b,i}) v(Y^{b,i}) \\ \vdots \\ \sum_{i=1}^{\#\mathscr{B}(b)} \eta_Q(Y^{b,i}) v(Y^{b,i}) \end{pmatrix}.$$

When the number of samples within a bundle tends to infinity, it is easy to see that

$$\frac{1}{\#\mathscr{B}(b)} \mathscr{I}^\top \mathscr{I} \to \begin{pmatrix} \mathbb{E}[\eta_1(Y))^2 | X \in \mathscr{B}(b)] & & \mathbb{E}[\eta_1(Y)\eta_Q(Y) | X \in \mathscr{B}(b)] \\ & \ddots & \\ \mathbb{E}[\eta_Q(Y)\eta_1(Y) | X \in \mathscr{B}(b)] & & \mathbb{E}[\eta_Q(Y))^2 | X \in \mathscr{B}(b)] \end{pmatrix}$$

and

$$\frac{1}{\#\mathscr{B}(b)} \mathscr{I}^\top v(Y^b) \to \begin{pmatrix} \mathbb{E}[\eta_1(Y) v(Y) | X \in \mathscr{B}(b)] \\ \vdots \\ \mathbb{E}[\eta_Q(Y) v(Y) | X \in \mathscr{B}(b)] \end{pmatrix}.$$

So, the empirical system of equations "converges" to the system of equations of a projection. Therefore, as long as our basis is properly defined such that they remain linearly independent for all bundles, this system of equations should be solvable with enough

simulation paths. Moreover, since the regression coefficients should "converge" to the theoretical projection coefficients, we could pick L depending on the L^2 norm of v itself, like, for example, two times its theoretical norm. Alternatively, when there are enough samples within each bundle, we suspect that the regression coefficients simply "converge" to the theoretical value and satisfy the bounded condition of regression in a natural way. Therefore, no actual rejection step in the algorithm is needed when there are enough samples within the bundles. This proposition appears to be supported by our numerical experiments.

However, there are multiple difficulties to incorporate the above argument into Theorem 4.4. First, since equal partitioning is not a recursive partitioning scheme as defined in [11], we cannot use a martingale argument on equal partitioning, limiting the available tools. Secondly, as the partition changes when increasing the overall number of samples, the above "convergence" does not seem to be properly defined. Finally, we would have to introduce a measure of convergence with respect to a matrix inverse, which is beyond the scope of this work.

On the other hand, there is a possibility that when the number of samples within a bundle is too small, the algorithm as a whole will fail to converge. Thus, it is beneficial to remind a user of such possibility and put a safety check in place. Therefore, we keep the derivation of Theorem 4.4 as a complete justification for SGBM. In practice, one can either make sure that there are enough samples within each bundle and let L be infinity. In this case, Theorem 4.4 no longer applies but we believe that the overall error will satisfy a bound of similar form. Alternatively, one starts from a small L when running the algorithm and increases L's value until most tests are accepted. By these techniques, the error bound from Theorem 4.4 remains valid.

4.4. ERROR ANALYSIS

A complete error description of the algorithm with respect to the application of SGBM towards BSDEs will be derived in this section.

We wish to apply the theorem from the last section to establish an error bound for the expectation of our approximation with respect to the selected simulation cloud. We need to check that after rejecting the simulations that generate regression coefficients that are "too large", our approximation functions are bounded in the recursion. We notice that for any $p \le P$,

$$|y_p^R(x)| \le \max\{C_{M,A} L\sqrt{2(1+C_\pi^2)}, C_{g,\mathscr{A}}\} =: C_{Y,\mathscr{A}}$$

and

$$|z_p^R(x)| \le C_{M,\mathscr{A}} L\sqrt{2(1+C_\pi^2)} =: C_{Z,\mathscr{A}}$$

for all x in a compact set $\mathscr{A}$. The constant C_π is defined as $\max_{k=0,\dots,N-1}\Delta_k$. These bounds can be proven by Assumption 4.1 and some simple inequalities. Furthermore, we have $\forall x \in \mathscr{A}$,

$$f_p^R(x) := f_p(y_p^R(x), z_p^R(x)) \le C_f + L_f(C_{Y,\mathscr{A}} + C_{Z,\mathscr{A}}) =: C_{f,\mathscr{A}},$$

which follows from the Lipschitz assumptions of f. Therefore, Theorem 4.4 applies.

We denote by S the set of all simulation cloud values $(X^{\pi,m}_{t_p})_{\substack{1\le m\le M\\ 0\le p\le P}}$ such that the Euclidean norm of the regression coefficients at each time step in each bundle is bounded by L, and the expectation is adjusted accordingly. With the application of Theorem 4.4, we know that for any given compact set $\mathscr{A}$,

$$\begin{aligned}
&\hat{\mathbb{E}}^x_{t_p,S}\left[\mathbb{E}^x_{t_p}\left[|y^R_{i+1}(X^\pi_{t_{i+1}})-\tilde{y}^R_{i+1}(X^\pi_{t_i},X^\pi_{t_{i+1}})|^2\right]\right]\\
&\le\frac{\vartheta(L'_y)}{\hat{\mathbb{E}}^x_{t_p}[\mathbf{1}_S]}\hat{\mathbb{E}}^x_{t_p}\left[\sum_{b=1}^{B}\int_{\mathscr{B}(b)}\int\nu(dx,dy)\frac{(\log(\sum_{m=1}^M\mathbf{1}_{\mathscr{B}(b)}(X^m)-1)+1)(Q+1)}{\sum_{m=1}^M\mathbf{1}_{\mathscr{B}(b)}(X^m)-1}\right]\\
&\quad+\hat{\mathbb{E}}^x_{t_p,S}\left[\sum_{b=1}^{B-1}\int_{\mathscr{B}(b)}\int\nu(dx,dy)\frac{24L'_y}{(\sum_{m=1}^M\mathbf{1}_{\mathscr{B}(b)}(X^m))}\right]\\
&\quad+\frac{12}{\hat{\mathbb{E}}^x_{t_p}[\mathbf{1}_S]}\hat{\mathbb{E}}^x_{t_p}\left[\sum_{\mathscr{B}\in\mathbb{B}}\int_{\mathscr{B}}\int\nu(dx,dy)(\inf_{\phi\in H}\sup_{\theta\in\mathscr{B}}\mathbb{E}^\theta_{t_i}\left[|y^R_{t_{i+1}}(X^\pi_{t_{i+1}})-\phi(X^\pi_{t_{i+1}})|^2\right]\wedge L'_y)\right]\\
&\quad+\hat{\mathbb{E}}^x_{t_p,S}\left[\mathbb{E}^x_{t_p}\left[|y^R_{i+1}(X^\pi_{t_{i+1}})-\tilde{y}^R_{i+1}(X^\pi_{t_i},X^\pi_{t_{i+1}})|^2(1-\mathbf{1}_{\mathscr{A}}(X^\pi_{t_{i+1}}))\right]\right]=:\Xi^x_{t_p}(i,y).
\end{aligned}$$

and

$$\begin{aligned}
&\hat{\mathbb{E}}^x_{t_p,S}\left[\mathbb{E}^x_{t_p}\left[|f^R_{i+1}(X^\pi_{t_{i+1}})-\tilde{f}^R_{i+1}(X^\pi_{t_i},X^\pi_{t_{i+1}})|^2\right]\right]\\
&\le\frac{\vartheta(L'_f)}{\hat{\mathbb{E}}^x_{t_p}[\mathbf{1}_S]}\hat{\mathbb{E}}^x_{t_p}\left[\sum_{b=1}^{B}\int_{\mathscr{B}(b)}\int\nu(dx,dy)\frac{(\log(\sum_{m=1}^M\mathbf{1}_{\mathscr{B}(b)}(X^m)-1)+1)(Q+1)}{\sum_{m=1}^M\mathbf{1}_{\mathscr{B}(b)}(X^m)-1}\right]\\
&\quad+\hat{\mathbb{E}}^x_{t_p,S}\left[\sum_{b=1}^{B-1}\int_{\mathscr{B}(b)}\int\nu(dx,dy)\frac{24L'_f}{(\sum_{m=1}^M\mathbf{1}_{\mathscr{B}(b)}(X^m))}\right]\\
&\quad+\frac{12}{\hat{\mathbb{E}}^x_{t_p}[\mathbf{1}_S]}\hat{\mathbb{E}}^x_{t_p}\left[\sum_{\mathscr{B}\in\mathbb{B}}\int_{\mathscr{B}}\int\nu(dx,dy)(\inf_{\phi\in H}\sup_{\theta\in\mathscr{B}}\mathbb{E}^\theta_{t_i}\left[|f^R_{t_{i+1}}(X^\pi_{t_{i+1}})-\phi(X^\pi_{t_{i+1}})|^2\right]\wedge L'_f)\right]\\
&\quad+\hat{\mathbb{E}}^x_{t_p,S}\left[\mathbb{E}^x_{t_p}\left[|f^R_{i+1}(X^\pi_{t_{i+1}})-\tilde{f}^R_{i+1}(X^\pi_{t_i},X^\pi_{t_{i+1}})|^2(1-\mathbf{1}_{\mathscr{A}}(X^\pi_{t_{i+1}}))\right]\right]=:\Xi^x_{t_p}(i,f),
\end{aligned}$$

where $L'_y=2LQC^2_{\eta,\mathscr{A}}+2C^2_{Y,\mathscr{A}}$ and $L'_f=2LQC^2_{\eta,\mathscr{A}}+2C^2_{f,A}$. Note that although the size of $C_\pi:=\max_{p=0,\dots,P-1}\Delta_p$ may affect multiple constants here, like $C_{M,\mathscr{A}}$ due to the probability law, $C_{Y,\mathscr{A}}$ and $C_{f,\mathscr{A}}$ by definition, however, $C_\pi\to 0$ would not make these constants converge to 0. So it may be easier to replace the constant C_π by T and consider these constants independent of the discretization scheme. Therefore, we consider the refined regression error to be independent of the discretization scheme.

The following proposition summarizes the error bound for our scheme:

$$\Delta z_p(x):=z_p(x)-z^R_p(x);\quad \Delta y_p(x):=y_p(x)-y^R_p(x).$$

Proposition 4.5. *Given Assumption 4.1, and the time-grid π and an N-dimensional vector $\gamma\in(0,+\infty)^N$ satisfying $12q(L^2_fR_\pi\vee 1)(\Delta_p+\frac{1}{\gamma_p})\le 1$, for all $p\le P-1$, we have, for $0\le p\le P$,*

$$\hat{\mathbb{E}}^x_{t_p,S}[|\Delta y_p(x)|^2]$$

$$\leq 6qe^{T/4}\sum_{i=p}^{P-2}(\Delta_i+\gamma_i^{-1})\Gamma_i L_f^2\Xi_{t_p}^x(i+1,y)+3e^{T/4}\sum_{i=p}^{P-1}(\Delta_i+\gamma_i^{-1})\Gamma_i\frac{1}{\Delta_i}\Xi_{t_p}^x(i,y)$$
$$+3e^{T/4}\sum_{i=p}^{P-1}(\Delta_i+\gamma_i^{-1})\Gamma_i\Delta_i\Xi_{t_p}^x(i,f), \tag{4.8}$$

where $\Gamma_i := \prod_{i=0}^{p-1}(1+\gamma_i\Delta_i)$, *and*

$$\hat{\mathbb{E}}_{t_p,S}^x\left[\sum_{i=p}^{P-1}\Delta_i\mathbb{E}_{t_p}^x\left[|\Delta z_i(X_{t_i}^\pi)|^2\right]\Gamma_i\right]$$
$$\leq(12q+3Te^{T/4})\sum_{i=p+1}^{P-1}\left(\Delta_i+\gamma_i^{-1}\right)\frac{1}{\Delta_i}\Xi_{t_p}^x(i,y)\Gamma_i+6qTe^{T/4}\sum_{i=p}^{P-2}(\Delta_i+\gamma_i^{-1})\Gamma_i L_f^2\Xi_{t_p}^x(i+1,y)$$
$$+(12q+3Te^{T/4})\sum_{i=p+1}^{P-1}\left(\Delta_i+\gamma_i^{-1}\right)\Delta_i\Xi_{t_p}^x(i,f)\Gamma_i+4\sum_{i=p}^{P-1}q\Xi_{t_p}^x(i,y)\Gamma_i.$$

We will discuss the bound on Δy_k here only as the two bounds are quite similar in structure. Note that the three terms within the sum at the right hand side of Equation (4.8) are also of similar structure. They all sum up the refined regression error multiplied by some constant related to Δ_i. The most problematic term is $3e^{T/4}\sum_{i=p}^{P-1}(\Delta_i+\gamma_i^{-1})\Gamma_i\frac{1}{\Delta_i}\Xi_{t_p}^x(i,y)$ as the coefficient is $\mathcal{O}(1)$. The value $\sum_{i=p}^{P-1}(\Delta_i+\gamma_i^{-1})\Gamma_i\frac{1}{\Delta_i}$ tends to infinity as the number of time steps tends to infinity. Therefore, one must use the parameters M and B to ensure the refined regression term is bounded by $\mathscr{C}\Delta_i^{1+\epsilon}$ for some constant $\mathscr{C}$, such that the sum and the error are bounded by $\mathscr{C}C_\pi^\epsilon$. This error plus the discretization error between the continuous system and the discretized system would be the complete error. So, in practice, one should ensure that $P, M, M/B$ all tend together to infinity.

Proof. The proof is fairly similar to the one used in [9] with the necessary modifications for our present algorithm.

We shall derive an a-priori estimate of the error propagation in the recursion steps and we start with an estimate of $\Delta z_p(x)$. Note that we add an extra term in the formula which is equal to zero due to the expectation of the Brownian motion being equal to zero. This term is added here to facilitate future steps of the proof. We have

$$\begin{aligned}|\Delta_p\Delta z_p(x)|^2 =& \left(\mathbb{E}_{t_p}^x\left[\left(\Delta y_{p+1}(X_{t_{p+1}}^\pi)-\mathbb{E}_{t_p}^x\left[\Delta y_{p+1}(X_{t_{p+1}}^\pi)\right]\right)\Delta W_p^\top\right]\right.\\ &\left.+\,\mathbb{E}_{t_p}^x\left[\left(y_{p+1}^R(X_{t_{p+1}}^\pi)-\tilde{y}_{p+1}^R(X_{t_p}^\pi,X_{t_{p+1}}^\pi)\right)\Delta W_p^\top\right]\right)^2\\ \leq& 2\left(\mathbb{E}_{t_p}^x\left[\left(\Delta y_{p+1}(X_{t_{p+1}}^\pi)-\mathbb{E}_{t_p}^x\left[\Delta y_{p+1}(X_{t_{p+1}}^\pi)\right]\right)\Delta W_p^\top\right]\right)^2\\ &+2\left(\mathbb{E}_{t_p}^x\left[\left(y_{p+1}^R(X_{p+1}^\pi)-\tilde{y}_{p+1}^R(X_{t_p}^\pi,X_{t_{p+1}}^\pi)\right)\Delta W_p^\top\right]\right)^2.\end{aligned}$$

The inequality follows from the inequality $(\sum_{p=1}^P a_p)^2 \leq \sum_{n=1}^P P a_p^2$, which will be frequently used in the proof and will not be specified again. By applying the Cauchy-Schwarz inequality, we can derive bounds for the two terms separately, where

$$\left|\mathbb{E}_{t_p}^x\left[\left(\Delta y_{p+1}(X_{t_{p+1}}^\pi)-\mathbb{E}_{t_p}^x\left[\Delta y_{p+1}(X_{t_{p+1}}^\pi)\right]\right)\Delta W_p^\top\right]\right|^2$$

$$\leq q\Delta_p\left(\mathbb{E}^x_{t_p}\left[(\Delta y_{p+1}(X^\pi_{t_{p+1}}))^2\right]-\left(\mathbb{E}^x_{t_p}\left[\Delta y_{p+1}(X^\pi_{t_{p+1}})\right]\right)^2\right),$$

and

$$\begin{aligned}&\left|\mathbb{E}^x_{t_p}\left[\left(y^R_{p+1}(X^\pi_{t_{p+1}})-\tilde{y}^R_{p+1}(X^\pi_{t_p},X^\pi_{t_{p+1}})\right)\Delta W_p^\top\right]\right|^2\\ \leq& q\Delta_p\mathbb{E}^x_{t_p}\left[\left|y^R_{p+1}(X^\pi_{t_{p+1}})-\tilde{y}^R_{p+1}(X^\pi_{t_p},X^\pi_{t_{p+1}})\right|^2\right].\end{aligned}$$

Therefore,

$$\begin{aligned}\Delta_p|\Delta z_p(x)|^2\leq& 2q\left(\mathbb{E}^x_{t_p}\left[(\Delta y_{p+1}(X^\pi_{t_{p+1}}))^2\right]-\left(\mathbb{E}^x_{t_p}\left[\Delta y_{p+1}(X^\pi_{t_{p+1}})\right]\right)^2\right)\\ &+2q\mathbb{E}^x_{t_p}\left[\left|y^R_{p+1}(X^\pi_{t_{p+1}})-\tilde{y}^R_{p+1}(X^\pi_{t_p},X^\pi_{t_{p+1}})\right|^2\right].\end{aligned}\tag{4.9}$$

Combining the fact that $(a+b)^2\leq(1+\gamma_p\Delta_p)a^2+(1+\gamma_k^{-1}\Delta_p^{-1})b^2$ for $(a,b)\in\mathbb{R}^2$, $\gamma_p>0$, and the Lipschitz property of f, one deduces with Equation (4.9) that, for $0\leq p\leq P-2$:

$$\begin{aligned}|\Delta y_p(x)|^2\leq&\Big(\mathbb{E}^x_{t_p}\left[\Delta y_{p+1}(X^\pi_{t_{p+1}})\right]+\mathbb{E}^x_{t_p}\left[y^R_{p+1}(X^\pi_{t_{p+1}})-\tilde{y}^R_{p+1}(X^\pi_{t_p},X^\pi_{t_{p+1}})\right]\\ &+\mathbb{E}^x_{t_p}\left[f_{p+1}(y_{p+1}(X^\pi_{t_{p+1}}),z_{p+1}(X^\pi_{t_{p+1}}))-f^R_{p+1}(X^\pi_{t_{p+1}})\right]\Delta_p\\ &+\mathbb{E}^x_{t_p}\left[f^R_{p+1}(X^\pi_{t_{p+1}})-\tilde{f}^R_{p+1}(X^\pi_{t_p},X^\pi_{t_{p+1}})\right]\Delta_p\Big)^2\\ \leq&(1+\gamma_p\Delta_p)\left(\mathbb{E}^x_{t_p}\left[\Delta y_{p+1}(X^\pi_{t_{p+1}})\right]\right)^2\\ &+3\left(\Delta_p+\gamma_p^{-1}\right)\Delta_p\Big[L_f^2\mathbb{E}^x_{t_p}\left[(\Delta y_{p+1}(X^\pi_{t_{p+1}}))^2\right]+L_f^2\mathbb{E}^x_{t_p}\left[(\Delta z_{p+1}(X^\pi_{t_{p+1}}))^2\right]\\ &\qquad+\frac{1}{\Delta_p^2}\mathbb{E}^x_{t_p}\left[|y_{p+1}(X^\pi_{t_{p+1}})-\tilde{y}_{p+1}(X^\pi_{t_p},X^\pi_{t_{p+1}})|^2\right]\\ &\qquad+\mathbb{E}^x_{t_p}\left[|f^R_{p+1}(X^\pi_{t_{p+1}})-\tilde{f}^R_{p+1}(X^\pi_{t_p},X^\pi_{t_{p+1}})|^2\right]\Big]\\ \leq&(1+\gamma_p\Delta_p)\left(\mathbb{E}^x_{t_p}\left[\Delta y_{p+1}(X^\pi_{t_{p+1}})\right]\right)^2\\ &+3(\Delta_p+\gamma_p^{-1})\Delta_pL_f^2\mathbb{E}^x_{t_p}\left[(\Delta y_{p+1}(X^\pi_{t_{p+1}}))^2\right]\\ &+6q(\Delta_p+\gamma_p^{-1})L_f^2R_\pi\left(\mathbb{E}^x_{t_p}\left[(\Delta y_{p+2}(X^\pi_{t_{p+2}}))^2\right]-\mathbb{E}^x_{t_p}\left[\left(\mathbb{E}_{t_{p+1}}\left[\Delta y_{p+2}(X^\pi_{t_{p+2}})\right]\right)^2\right]\right)\\ &+6q(\Delta_p+\gamma_p^{-1})L_f^2\mathbb{E}^x_{t_p}\left[\left|y^R_{p+2}(X^\pi_{t_{p+2}})-\tilde{y}^R_{p+2}(X^\pi_{t_{p+1}},X^\pi_{t_{p+2}})\right|^2\right]\\ &+3(\Delta_p+\gamma_p^{-1})\Delta_p\frac{1}{\Delta_k^2}\mathbb{E}^x_{t_p}\left[|y^R_{p+1}(X^\pi_{t_{p+1}})-\tilde{y}^R_{p+1}(X^\pi_{t_p},X^\pi_{t_{p+1}})|^2\right]\\ &+3(\Delta_p+\gamma_p^{-1})\Delta_p\mathbb{E}^x_{t_p}\left[|f^R_{p+1}(X^\pi_{t_{p+1}})-\tilde{f}^R_{p+1}(X^\pi_{t_p},X^\pi_{t_{p+1}})|^2\right],\end{aligned}\tag{4.10}$$

while

$$|\Delta y_{P-1}(x)|^2\leq 3\left(\Delta_p+\gamma_p^{-1}\right)\Delta_p\left[\frac{1}{\Delta_p^2}\mathbb{E}^x_{t_p}\left[|y^R_{p+1}(X^\pi_{t_{p+1}})-\tilde{y}^R_{p+1}(X^\pi_{t_p},X^\pi_{t_{p+1}})|^2\right]\right.$$

$$+\mathbb{E}^x_{t_p}\left[|f^R_{p+1}(X^\pi_{t_{p+1}}) - \tilde{f}^R_{p+1}(X^\pi_{t_p}, X^\pi_{t_{p+1}})|^2\right]\Big]. \quad (4.11)$$

Next, we define the following sequence

$$\lambda_p := \left[1 + \left(\gamma_{p-1} + \frac{1}{4}\right)\Delta_{p-1}\right]\lambda_{p-1}, \text{ where } \lambda_0 := 1,$$

consider the sum of $|\Delta y_i(X^\pi_{t_i})|\lambda_i$, from $i = 1$ to $P-1$, and take conditional expectations with respect to $\mathfrak{F}_p$. Applying Equation (4.11) for the case $p = P-1$ and Equation (4.10) otherwise, we have:

$$\begin{aligned}
\sum_{i=p}^{P-1}\mathbb{E}^x_{t_p}\left[|\Delta y_i(X^\pi_{t_i})|^2\lambda_i\right] \le &\sum_{i=p}^{P-2}\lambda_{i+1}\mathbb{E}^x_{t_p}\left[\left(\Delta y_{i+1}(X^\pi_{t_{i+1}})\right)^2\right] \\
&+ \sum_{i=p}^{P-2} 6q(\Delta_i + \gamma_i^{-1})L_f^2\lambda_i\mathbb{E}^x_{t_p}\left[\left|y^R_{i+2}(X^\pi_{i+2}) - \tilde{y}^R_{i+2}(X^\pi_{i+1}, X^\pi_{i+2})\right|^2\right] \\
&+ \sum_{i=p}^{P-1} 3(\Delta_i + \gamma_i^{-1})\Delta_i\frac{1}{\Delta_i^2}\lambda_i\mathbb{E}^x_{t_p}\left[|y^R_{i+1}(X^\pi_{t_{i+1}}) - \tilde{y}^R_{i+1}(X^\pi_{t_i}, X^\pi_{t_{i+1}})|^2\right] \\
&+ \sum_{i=p}^{P-1} 3(\Delta_i + \gamma_i^{-1})\Delta_i\lambda_i\mathbb{E}^x_{t_p}\left[|f^R_{i+1}(X^\pi_{t_{i+1}}) - \tilde{f}^R_{i+1}(X^\pi_{t_i}, X^\pi_{t_{i+1}})|^2\right].
\end{aligned}$$

By rearranging the terms, we have:

$$\begin{aligned}
|\Delta y_p(x)|^2\lambda_p \le &\sum_{i=p}^{P-2} 6q(\Delta_i + \gamma_i^{-1})L_f^2\lambda_i\mathbb{E}^x_{t_p}\left[\left|y^R_{i+2}(X^\pi_{t_{i+2}}) - \tilde{y}^R_{i+2}(X^\pi_{t_{i+1}}, X^\pi_{t_{i+2}})\right|^2\right] \\
&+ \sum_{i=p}^{P-1} 3(\Delta_i + \gamma_i^{-1})\Delta_i\frac{1}{\Delta_i^2}\lambda_i\mathbb{E}^x_{t_p}\left[|y^R_{i+1}(X^\pi_{t_{i+1}}) - \tilde{y}^R_{i+1}(X^\pi_{t_i}, X^\pi_{t_{i+1}})|^2\right] \\
&+ \sum_{i=p}^{P-1} 3(\Delta_i + \gamma_i^{-1})\Delta_i\lambda_i\mathbb{E}^x_{t_p}\left[|f^R_{i+1}(X^\pi_{t_{i+1}}) - \tilde{f}^R_{i+1}(X^\pi_{t_i}, X^\pi_{t_{i+1}})|^2\right].
\end{aligned}$$

It follows from the simple inequality $\Gamma_p \le \lambda_p = \exp(\sum_{i=0}^p \log(1 + (\gamma_i + 0.25)\Delta_i) \le e^{T/4}\Gamma_p$ that, for all $p \in \{0, \dots, P\}$,

$$\begin{aligned}
|\Delta y_p(x)|^2 \le &|\Delta y_p(x)|^2\Gamma_p \\
\le &6qe^{T/4}\sum_{i=p}^{P-2}(\Delta_i + \gamma_i^{-1})\Gamma_i L_f^2\mathbb{E}^x_{t_p}\left[\left|y^R_{i+2}(X^\pi_{t_{i+2}}) - \tilde{y}^R_{i+2}(X^\pi_{t_{i+1}}, X^\pi_{t_{i+2}})\right|^2\right] \\
&+ 3e^{T/4}\sum_{i=p}^{P-1}(\Delta_i + \gamma_i^{-1})\Gamma_i\frac{1}{\Delta_i}\mathbb{E}^x_{t_p}\left[|y^R_{i+1}(X^\pi_{t_{i+1}}) - \tilde{y}^R_{i+1}(X^\pi_{t_i}, X^\pi_{t_{i+1}})|^2\right] \\
&+ 3e^{T/4}\sum_{i=p}^{P-1}(\Delta_i + \gamma_i^{-1})\Gamma_i\Delta_i\mathbb{E}^x_{t_p}\left[|f^R_{i+1}(X^\pi_{t_{i+1}}) - \tilde{f}^R_{i+1}(X^\pi_{t_i}, X^\pi_{t_{i+1}})|^2\right]. \quad (4.12)
\end{aligned}$$

We can take expectations with respect to the simulation cloud and apply Theorem 4.4, which finishes the calculation for Δy.

Regarding the error term Δz, $\sum_{i=p}^{P-1} \Delta_i \mathbb{E}^x_{t_p}\left[|\Delta z_i(X^\pi_{t_i})|^2\right]\Gamma_i$ is bounded from above by

$$
\begin{aligned}
&\sum_{i=p}^{P-1} \Delta_i \mathbb{E}^x_{t_p}\left[|\Delta z_i(X^\pi_{t_i})|^2\right]\Gamma_i \\
\leq &\sum_{i=p}^{P-1} 2q\left(\mathbb{E}^x_{t_p}\left[(\Delta y_{i+1}(X^\pi_{t_{i+1}}))^2\right] - \mathbb{E}^x_{t_p}\left[\left(\mathbb{E}_{t_i}\left[\Delta y_{i+1}(X^\pi_{t_{i+1}})\right]\right)^2\right]\right)\Gamma_{i+1} \\
&+ \sum_{i=p}^{P-1} 2q\mathbb{E}^x_{t_p}\left[\left|y^R_{i+1}(X^\pi_{t_{i+1}}) - \tilde{y}^R_{i+1}(X^\pi_{t_i}, X^\pi_{t_{i+1}})\right|^2\right]\Gamma_i \\
\leq &2q\Gamma_P \mathbb{E}^x_{t_p}\left[(\Delta y_P(X^\pi_{t_P}))^2\right] \\
&+ \sum_{i=p+1}^{P-1} 2q\Gamma_i\left(\mathbb{E}^x_{t_p}\left[(\Delta y_i(X^\pi_{t_i}))^2\right] - (1+\gamma_i\Delta_i)\mathbb{E}^x_{t_p}\left[\left(\mathbb{E}_{t_i}\left[\Delta y_{i+1}(X^\pi_{t_{i+1}})\right]\right)^2\right]\right) \\
&+ \sum_{i=p}^{P-1} 2q\mathbb{E}^x_{t_p}\left[\left|y^R_{i+1}(X^\pi_{t_{i+1}}) - \tilde{y}^R_{i+1}(X^\pi_{t_i}, X^\pi_{t_{i+1}})\right|^2\right]\Gamma_i,
\end{aligned}
$$

because of Equation (4.9), and from (4.10), we have

$$
\begin{aligned}
\sum_{i=p}^{P-1} \Delta_i \mathbb{E}^x_{t_p}\left[|\Delta z_i(X^\pi_{t_i})|^2\right]\Gamma_i \leq &6\sum_{i=p+1}^{P-1} q\left(\Delta_i + \gamma_i^{-1}\right)\Delta_i L_f^2 \mathbb{E}^x_{t_p}\left[(\Delta y_{i+1}(X^\pi_{t_{i+1}}))^2\right]\Gamma_i \\
&+ 6\sum_{i=p+1}^{P-1} q\left(\Delta_i + \gamma_i^{-1}\right)\Delta_i L_f^2 \mathbb{E}^x_{t_p}\left[(\Delta z_{i+1}(X^\pi_{t_{i+1}}))^2\right]\Gamma_i \\
&+ 6\sum_{i=p+1}^{P-1} q\left(\Delta_i + \gamma_i^{-1}\right)\frac{1}{\Delta_i}\mathbb{E}^x_{t_p}\left[|y^R_{i+1}(X^\pi_{t_{i+1}}) - \tilde{y}^R_{i+1}(X^\pi_{t_i}, X^\pi_{t_{i+1}})|^2\right]\Gamma_i \\
&+ 6\sum_{i=p+1}^{P-1} q\left(\Delta_i + \gamma_i^{-1}\right)\Delta_i \mathbb{E}^x_{t_p}\left[|f^R_{i+1}(X^\pi_{t_{i+1}}) - \tilde{f}^R_{i+1}(X^\pi_{t_i}, X^\pi_{t_{i+1}})|^2\right]\Gamma_i \\
&+ \sum_{i=p}^{P-1} 2q\mathbb{E}^x_{t_p}\left[\left|y^R_{i+1}(X^\pi_{i+1}) - \tilde{y}^R_{i+1}(X^\pi_{t_i}, X^\pi_{t_{i+1}})\right|^2\right]\Gamma_i.
\end{aligned}
$$

Using the assumptions of the proposition statement, it follows that

$$
\begin{aligned}
&\sum_{i=p}^{P-1} \Delta_i \mathbb{E}^x_{t_p}\left[|\Delta z_i(X^\pi_{t_i})|^2\right]\Gamma_i \\
\leq &12\sum_{i=p+1}^{P-1} q\left(\Delta_i + \gamma_i^{-1}\right)\frac{1}{\Delta_i}\mathbb{E}^x_{t_p}\left[|y^R_{i+1}(X^\pi_{t_{i+1}}) - \tilde{y}^R_{i+1}(X^\pi_{t_i}, X^\pi_{t_{i+1}})|^2\right]\Gamma_i \\
&+ 12\sum_{i=p+1}^{P-1} q\left(\Delta_i + \gamma_i^{-1}\right)\Delta_i \mathbb{E}^x_{t_p}\left[|f^R_{i+1}(X^\pi_{t_{i+1}}) - \tilde{f}^R_{i+1}(X^\pi_{t_i}, X^\pi_{t_{i+1}})|^2\right]\Gamma_i \\
&+ 4\sum_{i=p}^{P-1} q\mathbb{E}^x_{t_p}\left[\left|y^R_{i+1}(X^\pi_{i+1}) - \tilde{y}^R_{i+1}(X^\pi_{t_i}, X^\pi_{i+1})\right|^2\right]\Gamma_i + \sum_{j=p+1}^{P-1} \Delta_j \mathbb{E}^x_{t_p}\left[(\Delta y_{j+1}(X^\pi_{t_{j+1}}))^2\right]\Gamma_{j+1}.
\end{aligned}
$$

Note that we may bound each individual term in the last sum with the estimate from Equation (4.12) and by taking conditional expectations.

$$
\begin{aligned}
&\mathbb{E}^x_{t_p}[(\Delta y_{j+1}(X^\pi_{t_{j+1}}))^2\Gamma_{i+1}]\\
\leq& 6qe^{T/4}\sum_{i=j+1}^{P-2}(\Delta_i+\gamma_i^{-1})\Gamma_i L_f^2\mathbb{E}^x_{t_p}\left[\left|y^R_{i+2}(X^\pi_{t_{i+2}})-\tilde{y}^R_{i+2}(X^\pi_{t_{i+1}},X^\pi_{t_{i+2}})\right|^2\right]\\
&+3e^{T/4}\sum_{i=j+1}^{P-1}(\Delta_i+\gamma_i^{-1})\Gamma_i\frac{1}{\Delta_i}\mathbb{E}^x_{t_p}\left[|y^R_{i+1}(X^\pi_{t_{i+1}})-\tilde{y}^R_{i+1}(X^\pi_{t_i},X^\pi_{t_{i+1}})|^2\right]\\
&+3e^{T/4}\sum_{i=j+1}^{P-1}(\Delta_i+\gamma_i^{-1})\Gamma_i\Delta_i\mathbb{E}^x_{t_p}\left[|f^R_{i+1}(X^\pi_{t_{i+1}})-\tilde{f}^R_{i+1}(X^\pi_{t_i},X^\pi_{t_{i+1}})|^2\right]\\
\leq& 6qe^{T/4}\sum_{i=p+1}^{P-2}(\Delta_i+\gamma_i^{-1})\Gamma_i L_f^2\mathbb{E}^x_{t_p}\left[\left|y^R_{i+2}(X^\pi_{t_{i+2}})-\tilde{y}^R_{i+2}(X^\pi_{t_{i+1}},X^\pi_{t_{i+2}})\right|^2\right]\\
&+3e^{T/4}\sum_{i=p+1}^{P-1}(\Delta_i+\gamma_i^{-1})\Gamma_i\frac{1}{\Delta_i}\mathbb{E}^x_{t_p}\left[|y^R_{i+1}(X^\pi_{t_{i+1}})-\tilde{y}^R_{i+1}(X^\pi_{t_i},X^\pi_{t_{i+1}})|^2\right]\\
&+3e^{T/4}\sum_{i=p+1}^{P-1}(\Delta_i+\gamma_i^{-1})\Gamma_i\Delta_i\mathbb{E}^x_{t_p}\left[|f^R_{i+1}(X^\pi_{t_{i+1}})-\tilde{f}^R_{i+1}(X^\pi_{t_i},X^\pi_{t_{i+1}})|^2\right]
\end{aligned}
$$

This upper bound is independent of j. Summing up the remaining parts, the time increments, results in the full time length T. We have:

$$
\begin{aligned}
&\sum_{i=p}^{P-1}\Delta_i\mathbb{E}^x_{t_p}\left[|\Delta z_i(X^\pi_{t_i})|^2\right]\Gamma_i\\
\leq&(12q+3Te^{T/4})\sum_{i=p+1}^{P-1}\left(\Delta_i+\gamma_i^{-1}\right)\frac{1}{\Delta_i}\mathbb{E}^x_{t_p}\left[|y^R_{i+1}(X^\pi_{t_{i+1}})-\tilde{y}^R_{i+1}(X^\pi_{t_i},X^\pi_{t_{i+1}})|^2\right]\Gamma_i\\
&+6qTe^{T/4}\sum_{i=p}^{P-2}(\Delta_i+\gamma_i^{-1})\Gamma_i L_f^2\mathbb{E}^x_{t_p}\left[\left|y^R_{i+2}(X^\pi_{t_{i+2}})-\tilde{y}^R_{i+2}(X^\pi_{t_{i+1}},X^\pi_{t_{i+2}})\right|^2\right]\\
&+(12q+3Te^{T/4})\sum_{i=p+1}^{P-1}\left(\Delta_i+\gamma_i^{-1}\right)\Delta_i\mathbb{E}^x_{t_p}\left[|f^R_{i+1}(X^\pi_{t_{i+1}})-\tilde{f}^R_{i+1}(X^\pi_{t_i},X^\pi_{t_{i+1}})|^2\right]\Gamma_i\\
&+4\sum_{i=p}^{P-1}q\mathbb{E}^x_{t_p}\left[\left|y^R_{i+1}(X^\pi_{i+1})-\tilde{y}^R_{i+1}(X^\pi_{t_i},X^\pi_{t_{i+1}})\right|^2\right]\Gamma_i,
\end{aligned}
$$

Again, taking expectations with respect to the simulation cloud finishes the proof. □

4.5. NUMERICAL EXPERIMENTS

In this section, numerical experiments are conducted for some selected examples. Before discussing these examples, we would specify the forward and backward discretization scheme used in these experiments. In particular, we use a more general backward scheme to show that our algorithm can be applied under general circumstances.

4.5.1. FORWARD AND BACKWARD SCHEME

In this section, we conduct our numerical experiments with the Euler-Maruyama discretization scheme, which is a common standard in the literature.

Definition 4.6 (Euler-Maruyama scheme)**.** The Euler-Maruyama scheme is defined by

$$X^\pi_{t_{p+1}} = X^\pi_{t_p} + \mu(t_p, X^\pi_{t_p})\Delta_p + \sigma(t_p, X^\pi_{t_p})\Delta W_p =: \mathfrak{e}(X^\pi_{t_p}, \Delta W_p).$$

Note that the conditional expectation $\mathbb{E}^x_{t_p}\left[\frac{\Delta W_{l,p}}{\Delta_p}\eta(X^\pi_{t_{p+1}})\right]$ can be calculated by:

$$\begin{aligned}
\mathbb{E}^x_{t_p}\left[\frac{\Delta W_{l,p}}{\Delta_p}\eta(X^\pi_{t_{p+1}})\right] =& \frac{1}{\sqrt{(2\pi)^q\Delta_p^q}}\int_{\mathbb{R}^q}\eta(\mathfrak{e}(x,y))\frac{\partial}{\partial y_l}\left(-\exp\left(-\frac{1}{2}\sum_{r=1}^q\frac{y_r^2}{\Delta_p}\right)\right)dy\\
=& \frac{1}{\sqrt{(2\pi)^q\Delta_k^q}}\int_{\mathbb{R}^q}\exp\left(-\frac{1}{2}\sum_{r=1}^q\frac{y_r^2}{\Delta_p}\right)\nabla\eta(\mathfrak{e}(x,y))\frac{\partial\mathfrak{e}(x,y)}{\partial y_l}dy\\
=& \mathbb{E}^x_{t_p}\left[\nabla\eta(X^\pi_{t_{p+1}})\right]\sigma_l(t_p,x),
\end{aligned}$$

where σ_l is the l-th column of the matrix σ.

For example, for the one-dimensional monomial x^r, $r\in\mathbb{N}$ and a forward process discretized by the Euler-Maruyama scheme, we have

$$\mathbb{E}^x_{t_p}\left[\frac{\Delta W_p}{\Delta_p}(X^\pi_{t_{p+1}})^r\right] = \mathbb{E}^x_{t_p}\left[r(X^\pi_{t_{p+1}})^{r-1}\right]\sigma(t_p,x).$$

The conditional expectations of polynomials are calculated directly by definition. We have

$$\begin{aligned}
\mathbb{E}^x_{t_p}[(X^\pi_{t_{p+1}})^0] =& 1;\\
\mathbb{E}^x_{t_p}[(X^\pi_{t_{p+1}})^1] =& x+\mu(t_p,x)\Delta_p;\\
\mathbb{E}^x_{t_p}[(X^\pi_{t_{p+1}})^2] =& x^2+2x\mu(t_p,x)\Delta_k+\sigma(t_p,x)^2\Delta_p+\mu(t_p,x)^2\Delta_p^2,
\end{aligned}$$

and so on.

For the backward discretization, we apply the more general theta-scheme from Chapter 1:

$$\begin{aligned}
Y^\pi_{t_p} =& \Phi(X^\pi_{t_p}), \quad Z^\pi_{t_p} = \left(\nabla g(X^\pi_{t_p})\sigma(t_P, X^\pi_{t_p})\right)^\top,\\
Z^\pi_{t_p} =& -\theta_2^{-1}(1-\theta_2)\mathbb{E}_{t_p}\left[Z^\pi_{t_{p+1}}\right] + \frac{1}{\Delta_p}\theta_2^{-1}\mathbb{E}_{t_p}\left[Y^\pi_{t_{p+1}}\Delta W_p\right]\\
&+\theta_2^{-1}(1-\theta_2)\mathbb{E}_{t_p}\left[f_{p+1}(Y^\pi_{t_{p+1}}, Z^\pi_{t_{p+1}})\Delta W_p\right],\ p=P-1,\dots,0,\\
Y^\pi_{t_p} =& \mathbb{E}_{t_p}\left[Y^\pi_{t_{p+1}}\right] + \Delta_p\theta_1 f_p(Y^\pi_{t_p}, Z^\pi_{t_p})\\
&+\Delta_p(1-\theta_1)\mathbb{E}_{t_p}\left[f_{p+1}(Y^\pi_{t_{p+1}}, Z^\pi_{t_{p+1}})\right],\ p=P-1,\dots,0,
\end{aligned}$$

where $0 \le \theta_1 \le 1$ and $0 < \theta_2 \le 1$.

Applying the SGBM algorithm to this general scheme, we have that the approximate functions within the bundle at time p are defined by:

$$z_{r,p}^{(\theta_1,\theta_2),R}(b,x) = -\theta_2^{-1}(1-\theta_2)\mathbb{E}_{t_p}^x\left[\eta(X_{t_{p+1}}^\pi)\right]\beta_{r,p+1}(b) + \theta_2^{-1}\mathbb{E}_{t_p}^x\left[\frac{\Delta W_{r,p}}{\Delta_p}\eta(X_{t_{p+1}}^\pi)\right](\alpha_{p+1}(b) + (1-\theta_2)\Delta_p\gamma_{p+1}(b)), \quad r = 1,\dots,d;$$

$$y_p^{(\theta_1,\theta_2),R,0}(b,x) = \mathbb{E}_{t_k}^x\left[p(X_{t_{k+1}}^\pi)\right]\alpha_{k+1}(b),$$

$$y_k^{(\theta_1,\theta_2),R,i}(b,x) = \Delta_k\theta_1 f(t_k,x,y_k^{(\theta_1,\theta_2),R,i-1}(x),z_k^{(\theta_1,\theta_2),R}(x)) + h_k(b,x),$$

$$h_k(b,x) = \mathbb{E}_{t_k}^x\left[p(X_{t_{k+1}}^\pi)\right](\alpha_{k+1}(b) + \Delta_k(1-\theta_1)\gamma_{k+1}(b)), \quad i = 1,\dots,I.$$

Note that a Picard iteration is performed at each time step for each bundle if the choice of (θ_1,θ_2) results in an implicit scheme.

Different types of backward discretizations will be considered for Example 1.

4.5.2. EXAMPLE 1

This example is originally from [8] and has been used in Chapter 2. The considered FBSDE is given by

$$\begin{cases} dX_t = dW_t, \\ dY_t = -(Y_tZ_t - Z_t + 2.5Y_t - \sin(t+X_t)\cos(t+X_t) - 2\sin(t+X_t))dt + Z_tdW_t. \end{cases}$$

We take the initial and terminal conditions $x_0 = 0$ and $Y_T = \sin(X_T + T)$.

The exact solution is given by

$$(Y_t, Z_t) = (\sin(X_t + t), \cos(X_t + t)).$$

The terminal time is set to be $T = 1$ and $(Y_0, Z_0) = (0,1)$. We use the set $\{1, x, x^2\}$ as the regression base for this example. We apply equal partitioning bundling for all our tests with the sample paths sorted by the value function x. As mentioned in Session 4.2.2, not all assumptions set in this work are necessary for the basic SGBM algorithm to work. For example, Assumption 4.1 is included to ensure the existence and uniqueness of the solution of the BSDE. In this example, even though the driver is not Lipschitz, one can check that the above solution solves the BSDE with Itô's formula, and the SGBM algorithm still applies.

Table 4.1 shows the tests that we have run. Basically, our test cases can be placed into two groups. Test cases 1a, 1b, 1c are tests for the explicit version of our algorithm, while test cases 1d, 1e, 1f are for the Crank-Nicolson version. Within each group, the three tests are run for identical test settings, except for the constant L, i.e., the pre-set limit for the Euclidean norm so that we may check the influence of the factor L. Within each test, the factors M, P and B are linked to a common factor J such that when J tends infinity, P, B and M/B tend to infinity as well. This setting is inspired by our observation on the error bound that all three factors should tend to infinity together to ensure the convergence of the algorithm. However, the exact ratio between the three factors is from empirical experience.

Table 4.1: Test cases for Example 1

Test Case	θ_1	θ_2	I	M	P	B	L
1a	0	1	-	2^{2J}	2^J	2^J	100
1b	0	1	-	2^{2J}	2^J	2^J	10000
1c	0	1	-	2^{2J}	2^J	2^J	–
1d	0.5	0.5	4	2^{2J}	2^J	2^J	100
1e	0.5	0.5	4	2^{2J}	2^J	2^J	10000
1f	0.5	0.5	4	2^{2J}	2^J	2^J	–

4.5.3. EXAMPLE 2: BLACK-SCHOLES EUROPEAN OPTION

The second example under consideration is the calculation of the price $v(t, S_t)$ of a European option under the d-dimensional Black-Scholes model by solving a FBSDE, which has been a classical application of BSDEs. It has been introduced in classical papers, like [12], and we have seen a one-dimensional Black-Scholes BSDE in Section 2.5.2. Here we provide a brief review for the multidimensional Black-Scholes model with BSDEs.

We consider a market where the assets satisfy:

$$dS_{i,t} = \bar{\mu}_i S_{i,t} dt + \bar{\sigma}_i S_{i,t} d\omega_{i,t},\ 1 \le i \le d,$$

where ω_t is a correlated d-dimensional Wiener process, with

$$d\omega_{i,t} d\omega_{j,t} = \rho_{ij} dt.$$

The parameters ρ_{ij} form a symmetric matrix ρ,

$$\rho = \begin{pmatrix} 1 & \rho_{12} & \rho_{13} & \cdots & \rho_{1q} \\ \rho_{21} & 1 & \rho_{23} & \cdots & \rho_{2q} \\ \vdots & \vdots & \vdots & & \vdots \\ \rho_{q1} & \rho_{q2} & \rho_{q3} & \cdots & 1 \end{pmatrix},$$

and we assume it is invertible. By performing a Cholesky decomposition on ρ such that $\mathfrak{C}\mathfrak{C}^\top = \rho$, where $\mathfrak{C}$ is a lower triangular matrix with real and positive diagonal entries, we may relate the correlated and standard Brownian motions, as follows,

$$\omega_t = \mathfrak{C} W_t.$$

Along the line of reasoning in [13] and similar to Section 2.5.2, we assume the financial market is complete, there is no trading restriction and a derivative can be perfectly hedged. To derive the corresponding pricing BSDE for a European option with terminal payoff $g(S_T)$, we construct a replicating portfolio Y_t, containing $\mathfrak{m}_{i,t}$ of asset $S_{i,t}$ and bonds with risk-free return rate $\bar{r}$. Applying the self-financing assumption, the portfolio follows the SDE:

$$dY_t = -(-\bar{r} Y_t - \sum_{i=1}^{d} \mathfrak{m}_{i,t}(\bar{\mu}_i - \bar{r}) S_{i,t}) dt + \sum_{i=1}^{d} \mathfrak{m}_{i,t} \bar{\sigma}_i S_i d\omega_{i,t}.$$

If we set $Z_t = (\mathfrak{m}_{1,t}\bar{\sigma}_1 S_{1,t}, \ldots, \mathfrak{m}_{d,t}\sigma_d S_{d,t})\mathfrak{C}$, then (Y, Z) solves the BSDE,

$$\begin{cases} dY_t = -\left(-\bar{r}Y_t - Z_t\mathfrak{C}^{-1}\left(\frac{\bar{\mu}-\bar{r}}{\bar{\sigma}}\right)\right)dt + Z_t dW_t; \\ Y_T = g(S_T), \end{cases}$$

where $\left(\frac{\bar{\mu}-\bar{r}}{\bar{\sigma}}\right) = \left(\frac{\bar{\mu}_1-\bar{r}}{\bar{\sigma}_1}, \cdots, \frac{\bar{\mu}_q-\bar{r}}{\bar{\sigma}_q}\right)^\top$.

We test our algorithm for the next two cases.

EXAMPLE 2.1: ARITHMETIC BASKET PUT OPTION

In this numerical test, we use the 5-dimensional example from [14], which is designed as a tractable representation for the German stock index DAX at that time. All $\bar{\mu}_i$ are assumed to be $\bar{r}$ here. The volatilities are given by

$$(\bar{\sigma}_1, \bar{\sigma}_2, \bar{\sigma}_3, \bar{\sigma}_4, \bar{\sigma}_5) = (0.518, 0.648, 0.623, 0.570, 0.530),$$

while the correlations ρ are given by

$$\rho = \begin{pmatrix} 1.00 & 0.79 & 0.82 & 0.91 & 0.84 \\ 0.79 & 1.00 & 0.73 & 0.80 & 0.76 \\ 0.82 & 0.73 & 1.00 & 0.77 & 0.72 \\ 0.91 & 0.80 & 0.77 & 1.00 & 0.90 \\ 0.84 & 0.76 & 0.72 & 0.90 & 1.00 \end{pmatrix}.$$

We would consider a European weighted basket put option for $T = 1$ year, with the payoff function g given by

$$g(S) = \left(1 - \sum_{i=1}^{5} w_i S_i\right)^+,$$

where $(w_1, w_2, w_3, w_4, w_5) = (38.1, 6.5, 5.7, 27.0, 22.7)$. The risk free interest rate is $\bar{r} = 0.05$ and all the stocks have starting value 0.01. The reference price is given as 0.175866 in [14].

We perform the equal-partitioning bundling technique and sort the paths in different bundles according to the ordering of the value $\sum_{i=1}^{5} w_i S^m_{i,t_p}$. The regression basis is chosen to be $\eta_l(x) = \left(\sum_{i=1}^{5} w_i x_i\right)^{l-1}$ for $l = 1, \ldots, Q$.

Table 4.2 shows the tests that we have run. In these tests, we keep most of the parameters fixed but vary the number of bundles. We test our algorithm for the explicit scheme with a second-order regression basis and the Crank-Nicolson scheme with a third-order regression basis. The change of basis is made to test the impact of the regression basis to our algorithm. We just keep these two sets of tests to demonstrate the impact of the number of bundles.

Table 4.2: Test cases for Example 2.1

Test Case	θ_1	θ_2	I	M	P	B	L	Q
2.1a	0.5	0.5	4	2^{12}	10	2^{2J}	-	3
2.1b	0	1	-	2^{12}	10	2^{2J}	-	2

EXAMPLE 2.2: GEOMETRIC BASKET PUT OPTION

Here we also consider the problem of pricing q-dimensional geometric basket options with initial state $S_0 = (40,\dots,40) \in \mathbb{R}^q$; strike $K = 40$; risk-free interest rate $\bar{r} = 0.06$; volatility $\bar{\sigma}_i = 0.2, i = 1,\dots,d$; correlation $\rho_{ij} = 0.25, i,j = 1,\dots,d, i \neq j$; and maturity $T = 1.0$. The final payoff function is given by

$$g(S) = \left(K - \left(\prod_{i=1}^{d} S_i \right)^{\frac{1}{d}} \right)^+ .$$

This is the same setting as in [4] but for European options instead of Bermudan options.

We again use the equal-partitioning technique and sort the paths in different bundles according to the ordering of the values $\left(\prod_{i=1}^{d} S^m_{i,t_p}\right)^{\frac{1}{d}}$. The regression basis is chosen to be $\eta_l(x) = \left(\prod_{i=1}^{d} x_i\right)^{\frac{l-1}{d}}$ for $l = 1,2,3$.

Since the geometric product of a geometric Brownian motion remains a geometric Brownian motion, the analytic solution can be found using Black-Scholes formula and any other classical pricing method.

Table 4.3 shows the tests that we have run. In these sets of tests, we fixed all the parameters but change the number of stocks in our test. This example is used to test the *scalability* of our methodology. Tests are performed for both explicit and Crank-Nicolson schemes.

Table 4.3: Test cases for Example 2.2

Test Case	θ_1	θ_2	I	M	P	B	L
2.2a	0	1	-	2^{12}	20	16	-
2.2b	0.5	0.5	4	2^{12}	20	16	-

4.5.4. RESULTS

The results are given as the average values of 10 separated runs of the algorithm.

We first consider the results of the explicit version of our algorithm applied to Example 1, namely test cases 1a, 1b and 1c, in Table 4.4. This test can be seen as a proof of concept. As mentioned, we design the test in such a way that the number of steps P, the number of bundles B and the ratio M/B all tend to infinity. As expected, our algorithm converges under this setting. Moreover, the total variation of the absolute errors among each successful run converges with respect to J too, as the reader can read from the second part of Table 4.4. It is defined as the sum of the individual differences between the Monte Carlo result of each run (which is not rejected) and the analytic solution, divided by the total number of successful runs.

While we have not shown the proof of convergence for the Crank–Nicolson scheme, where $\theta_1 = \theta_2 = 0.5$, our numerical tests for test cases 1d, 1e, 1f, in Table 4.5, suggest that it works well in our framework.

A specific point of interest is the impact of factor L introduced in Section 4.3 for the samples selection. It can be seen in Table 4.5 that when the number of paths or the

Table 4.4: Test result for Example 1 with explicit scheme.

$|Y_0 - y_0^{(\theta_1,\theta_2),R}(x_0)|$

J	2	3	4	5	6	7	8
1a	0.0235	0.204	0.0469	0.0571	0.0266	0.0182	0.0162
1b	0.184	0.178	0.0988	0.0302	0.0288	0.0196	0.00576
1c	0.416	0.144	0.104	0.0466	0.0181	0.0192	0.00984

Total Variation/Successful Run

J	2	3	4	5	6	7	8
1a	0.282	0.204	0.0810	0.0571	0.0273	0.0182	0.0162
1b	0.310	0.178	0.0989	0.0446	0.0288	0.0204	0.00795
1c	0.601	0.157	0.104	0.0547	0.0194	0.0192	0.0118

Table 4.5: Test result for Example 1 with Crank–Nicolson scheme

$|Y_0 - y_0^{(\theta_1,\theta_2),R}(x_0)|$

J	2	3	4	5	6	7	8
1a	0.00534	0.0326	0.181	0.0258	0.00604	0.0206	NA
1b	3.68	0.246	0.349	0.0692	0.0129	0.00137	0.00241
1c	4.68×10^8	3.52×10^{137}	1.08×10^{44}	0.0511	0.00505	0.0117	0.00305

Total Variation/Successful Run

J	2	3	4	5	6	7	8
1a	0.235	0.0326	0.181	0.0258	0.0126	0.0206	NA
1b	4.57	0.376	0.349	0.0756	0.0146	0.0125	0.0109
1c	4.68×10^8	3.52×10^{137}	1.08×10^{44}	0.0583	0.0209	0.0143	0.00789

bundles are few, a smaller value of L preserves the stability of our algorithm. In test case $1d$, where the factor L is relatively small, our algorithm rejected all tests for $J = 8$. One of the explanations is that the regression coefficients converge to the analytic projection coefficients on the basis space but the norm of these analytic coefficient is greater than L. The effect of the factor L actually can be seen in Table 4.4 too. Some runs for test case 1a were rejected when $J = 8$ and the result for $J = 8$ is worse than either 1b or 1c. On the contrary, if we remove the restriction on L altogether, the results are non-satisfactory when the value of J is low but converge when the number of time steps and samples are high enough. Heuristically, the regression coefficients should converge to the actual projection coefficients on the basis space, which results in a function that is bounded in a compact set. This in turns satisfies the conditions of the proof of convergence with respect to the regression. Although it may look like we can adjust L in the same time as other algorithm parameters in order to achieve the optimal result, we should still note that L is model dependent and there is no clear way to figure out the best link of L with the simulation parameters. It remains important to use L as a warning system.

Next, we shall move on to the result for the more practical and higher-dimensional Example 2. The results for Example 2.1 in Table 4.6 show that our method can be easily

applied to a practical problem.

Table 4.6: Test result for Example 2.1

	$\lvert Y_0 - y_0^{(\theta_1,\theta_2),R}(x_0)\rvert$		
J	0	1	2
2.1a	2.03×10^{-3}	2.26×10^{-3}	1.99×10^{-3}
2.1b	2.93×10^{-3}	1.89×10^{-3}	2.22×10^{-4}

With respect to the problem of dimensionality, we can check the results in Table 4.7. Since the analytic solution is known to this problem, we compare our result to the actual value. It can be seen that under our choice of bundling and regression basis, the accuracy of our method is similar across all choices of problem dimensions. This suggested that with appropriate setting, our algorithm can easily scale up to tackle high-dimensional problems.

Table 4.7: Test result for Example 2.2

	$\lvert Y_0 - y_0^{(\theta_1,\theta_2),R}(x_0)\rvert$				
Stock dim.	1	2	3	4	5
2.2a	6.55×10^{-3}	7.30×10^{-3}	6.68×10^{-3}	8.04×10^{-3}	7.13×10^{-3}
2.2b	5.19×10^{-3}	6.95×10^{-3}	6.40×10^{-3}	6.95×10^{-3}	7.49×10^{-3}
Stock dim.	6	7	8	9	10
2.2a	6.99×10^{-3}	7.51×10^{-3}	6.93×10^{-3}	7.00×10^{-3}	7.57×10^{-3}
2.2b	7.20×10^{-3}	7.16×10^{-3}	7.09×10^{-3}	7.20×10^{-3}	6.76×10^{-3}
Stock dim.	11	12	13	14	15
2.2a	6.95×10^{-3}	7.40×10^{-3}	7.53×10^{-3}	7.14×10^{-3}	7.14×10^{-3}
2.2b	8.46×10^{-3}	7.14×10^{-3}	7.63×10^{-3}	7.90×10^{-3}	7.25×10^{-3}

More generally, all the results from Example 2 suggest that linking the bundling criterion and the regression basis to the terminal condition can deliver an accurate algorithm. Adapting our algorithm to a specific problem to improve the performance could be a promising direction of further research. In fact, the choice of basis itself deserves further study. Even in our localised setting, regression with respect to the linear basis scheme fails to converge for Example 1. A more sophisticated way to pick the regression basis may be important to put our algorithm into actual applications.

To sum up, we have developed a new algorithm for approximating BSDEs based on SGBM and our numerical tests showed that this new algorithm can deliver accurate estimation results.

REFERENCES

[1] K. W. Chau and C. W. Oosterlee, *Stochastic grid bundling method for backward stochastic differential equations,* International Journal of Computer Mathematics **96**, 2272 (2019).

[2] S. Jain and C. W. Oosterlee, *The stochastic grid bundling method: Efficient pricing*

of Bermudan options and their greeks, Applied Mathematics and Computation **269**, 412 (2015).

[3] F. Cong and C. W. Oosterlee, *Pricing Bermudan options under Merton jump-diffusion asset dynamics,* International Journal of Computer Mathematics **92**, 2406 (2015).

[4] A. Leitao and C. W. Oosterlee, *GPU acceleration of the stochastic grid bundling method for early-exercise options,* International Journal of Computer Mathematics **92**, 2433 (2015).

[5] C. S. L. De Graaf, Q. Feng, D. Kandhai, and C. W. Oosterlee, *Efficient computation of exposure profiles for counterparty credit risk,* International Journal of Theoretical and Applied Finance **17**, 1450024 (2014).

[6] P. Glasserman and B. Yu, *Simulation for American options: Regression now or regression later?* in *Monte Carlo and Quasi-Monte Carlo Methods 2002: Proceedings of a Conference held at the National University of Singapore, Republic of Singapore, November 25–28, 2002*, edited by H. Niederreiter (Springer Berlin Heidelberg, Berlin, Heidelberg, 2004) pp. 213–226.

[7] B. Bouchard and X. Warin, *Monte-carlo valuation of american options: Facts and new algorithms to improve existing methods,* in *Numerical Methods in Finance: Bordeaux, June 2010*, edited by R. A. Carmona, P. Del Moral, P. Hu, and N. Oudjane (Springer Berlin Heidelberg, Berlin, Heidelberg, 2012) pp. 215–255.

[8] W. Zhao, Y. Li, and G. Zhang, *A generalized θ-scheme for solving backward stochastic differential equations,* Discrete and Continuous Dynamical Systems - Series B **17**, 1585 (2012).

[9] E. Gobet and P. Turkedjiev, *Linear regression MDP scheme for discrete backward stochastic differential equations under general conditions,* Mathematics of Computation **85** (2016).

[10] L. Györfi, M. Kohler, A. Krzyzak, and H. Walk, *A Distribution-Free Theory of Nonparametric Regression*, Springer Series in Statistics (Springer, 2002).

[11] L. Gordon and R. A. Olshen, *Almost surely consistent nonparametric regression from recursive partitioning schemes,* Journal of Multivariate Analysis **15**, 147 (1984).

[12] N. El Karoui, S. G. Peng, and M. C. Quenez, *Backward stochastic differential equations in finance,* Mathematical Finance **7**, 1 (1997).

[13] M. J. Ruijter and C. W. Oosterlee, *A Fourier cosine method for an efficient computation of solutions to BSDEs,* SIAM Journal on Scientific Computing **37**, A859 (2015).

[14] C. Reisinger and G. Wittum, *Efficient hierarchical approximation of high-dimensional option pricing problems,* SIAM Journal on Scientific Computing **29**, 440 (2007).

5

AN SGBM-XVA DEMONSTRATOR: A SCALABLE PYTHON TOOL FOR PRICING XVA

In this chapter, we develop a Python demonstrator for pricing total valuation adjustment (XVA) based on the stochastic grid bundling method (SGBM) developed in Chapter 4. The motivation for this work is basically two-fold. On the application side, by focusing on a particular financial application of BSDEs, we can show the potential of using SGBM on a real-world risk management problem. On the implementation side, we explore the potential of developing a simple yet highly efficient code with SGBM by incorporating CUDA Python into our program.

5.1. INTRODUCTION

This work can be seen as a follow-up of our theoretical research on the stochastic grid bundling method (SGBM) for BSDEs in Chapter 4. Here, we will study the practical side by developing a demonstrator in Python, where we shall also make use of computing on a Graphics Processing Unit (GPU) in order to improve the scalability, and make use of the CUDA Python package. CUDA Python is a recently open to public programming tool, which was developed by Anaconda. It has become freely available at the end of 2017, being previously a commercial software. This programming tool carries the promise of combining fast development time for Python coding with the high efficiency of GPU computing.

Another focus of this work is the application of BSDEs in financial risk management, where we would like to demonstrate the practical opportunities for efficient BSDE solving software. We choose the modeling of the variation margin and the close-out value within risk management, in the form of BSDEs, as the main test problem. In this work,

This chapter is based on the article 'An SGBM-XVA demonstrator: a scalable Python tool for pricing XVA', to appear in Journal of Mathematics in Industry.

we work under a complete market assumption which includes counterparty risk and margin requirements, and develop a numerical solver for XVA.

A Python demonstrator for solving XVA pricing problems with the SGBM algorithm with GPU computing is the main result of this study. The strength of this package is its scalability with respect to the dimensionality of the underlying stock price processes. While the demonstrator is designed for a specific problem setting, because of the general framework with BSDEs and SGBM, the package could easily be transformed into a general solver for BSDEs and used for other financial problems.

This chapter is organized as follows. Section 5.2 describes the programming language, the application of parallel computing with SGBM, the financial setting for our SGBM-XVA demonstrator, and other technical details, while some numerical tests are performed in Section 5.3. Concluding remarks, possible extensions and outlook are given in Section 5.4.

5.2. THE SGBM-XVA DEMONSTRATOR

In order to test and analyze the applicability and practicality of the SGBM algorithm in the BSDE-based financial model framework, we have created an SGBM-XVA demonstrator which computes XVA, i.e. a value of great interest in modern risk management, with the algorithm introduced in the previous chapter. We make use of the Python programming language and the CUDA Python package for the development of this SGBM-XVA demonstrator. In this section, we address the programming tools that we have used, the basic financial setting for this problem and the design of our code.

5.2.1. GPU COMPUTATION: PARALLEL SGBM

In order to improve the efficiency of the SGBM for BSDEs algorithm, we would make use of GPU acceleration in our demonstrator. This idea has been successfully implemented for the SGBM algorithm for early-exercise options in [1]. The framework of parallel computation from [1] can also be used here. In this framework, we divide the SGBM algorithm in two stages, namely, the forward simulation stage and the backward recursion approximation stage. In this subsection, we briefly describe how parallel GPU computing is used in each stage.

In the forward simulation stage, we simulate independent samples of the stock price models [1] from the initial time t_0 to the expiration date T as defined by the problem. Moreover, we can already pre-compute all values of interest for the backward step that are related to the stock prices and store them in the memory. These values of interest include the sorting parameter used for bundling, the basis function values, the expectations of the basis functions and the terminal values. Since the generation of each sample is independent of the other samples and a huge amount of samples may be needed for an accurate Monte Carlo simulation, this stage is particularly suitable for parallel computing. Also note that after the calculation of the values of interest, the actual stock paths $(S_{t_p^\pi})_{0\le p\le P}$ can be discarded in the GPU, as they are not required anymore in the backward step.

[1] These are the forward processes in the terminology of Chapter 4. They are denoted as S instead of X in this chapter.

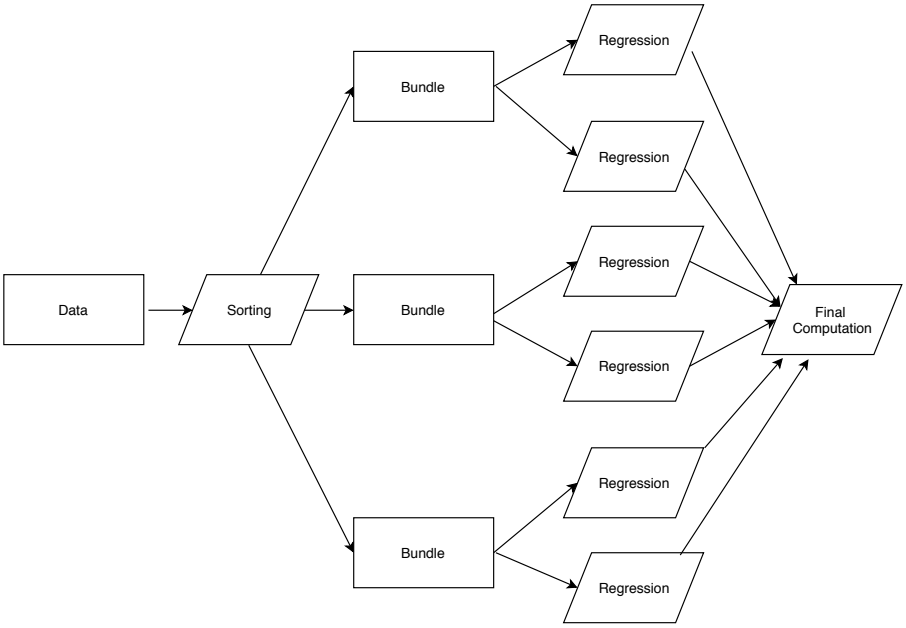


Figure 5.1: Flowchart for the backward calculation within a time step.

The second stage is the backward approximation stage. From the discretization of the BSDE, we notice that the calculation in time-wise direction should process sequentially, i.e., starting from the terminal time and going backward along the time direction. However, within each time step, there are many independent processes that are well suitable for parallel computing. Within each time step, the data (i.e. the values of interest) is separated into different non-overlapping bundles and the computations in the different bundles are independent of each other. Within a bundle, multiple regression steps on different variables (Y, Z, f, ...) need to be completed. For a graphical representation of the computation within each time step, we refer to Figure 5.1. As each regression within the time step is independent, we can also perform these computations simultaneously. Finally, in order to reduce the overall volume of memory transfers, only the information for the current time point is transferred to the GPU.

5.2.2. PROGRAMMING LANGUAGE

Python is the programming language of choice for this project as it is a popular tool in the financial industry nowadays. Being a high-level programming language, Python is easy to develop and particularly useful for scripting because of its easy to write syntax and grammar. These properties are especially useful for the financial industry, as practitioners need to constantly monitor and adapt their models to the changing markets. Moreover, Python has been one of the dominating programming languages in data

science with its widely available packages. Therefore, we develop our algorithm under the Python framework. However, Python is an interpreted programming language, so, its performance is not as strong as a compiled language. One of our focusses of study is therefore the balance between rapid development and actual computation performance which can be achieved by Python.

One of the effective techniques for improving the computational efficiency of Python is to pre-compile parts of the code, for example, with the help of the Numba package. This technique has been adopted in our SGBM-XVA demonstrator. In order to further improve the efficiency of our algorithm, we apply parallel GPU computing as stated in the last subsection. The use of GPUs has been a major development in scientific computing. Along with the computing platform CUDA, the GPU provides a high potential for computational speed-up. With more than hundred threads (the basic operational units within CUDA) in a typical GPU, repetitive function computations can be dedicated to various treads to be run in parallel. It will greatly reduce the computational time.

In this work, we use the CUDA Python packages to incorporate GPU programming into Python. CUDA Python is made out of Python packages from Continuum Analytics, that allow a user to make use of CUDA within the native Python syntax. The tool consists of two main parts: a Numba compiler and the pyculib library. The Numba compiler transforms a native Python code with only supported features into a CUDA kernel or a device function with only a function decorator. This feature enables GPU computing in Python without the need for the programmer to learn a new language. The pyculib library [2] is a collection of several numerical libraries, that provide functions for random number generation, sorting, linear algebra operations and more. This set of tools has been previously included in a commercial software, but it has become open source since 2017.

Another benefit of using CUDA Python is in terms of an automatic memory transfer. In a GPU computing framework, it is often necessary to transfer data between the CPU and GPU memory space. In this tool, a programmer can either manage the transfer of memory for better control of GPU memory usage and the bandwidth of memory transfer between CPU and GPU, or let the platform handle it automatically, again simplifying the code development.

The main down-side of this tool is that so far it only supports certain functions from the native Python and Numpy packages. Some of the well-optimized packages in linear algebra are not available for the GPU, and some of the Python code has to be rewritten into a supported version. However, using the package still requires less adaptation effort as compared to the incorporation of other compiled programming languages, like C, into a Python code. As we will discuss later, this tool delivers a great improvement in efficiency in some areas.

5.2.3. FINANCIAL TEST CASE: TOTAL VALUATION ADJUSTMENT (XVA)

Next, we shall introduce the test problem for our SGBM-XVA demonstrator. The main goal here is to show that BSDEs and SGBM can be used to solve financial total valuation adjustment (XVA) problems in risk management.

In short, we consider a financial market where the bank is selling derivatives to a counterparty, but either the bank or the counterparty may default before expiry. There-

fore, a variation margin has to be posted as collateral which will be used to settle the account when one party defaults. In this market, the funding rate for each financial product may be different, as well as the deposit rate for a risk-free account and the funding rate through bonds. The goal of this model is to compare the price of a financial portfolio with and without counterparty risk. The difference between these two prices is called the total value adjustment.

We use the standard multi-dimensional Black-Scholes model introduced in Chapter 4 for the asset dynamics and we use a simplified version of the dynamics for our financial market, as in [3]. Within this setting, we take into account the possibility that both parties, in an over-the-counter financial contract, may default, also we include the exchange of collateral, a close-out exchange in the case of a default and the usage of a repurchasing agreement (repo) for position funding. While this model is definitely more realistic than the classical financial derivative pricing theory (where one can borrow and lend freely with negligible cost) by taking into account the counterparty risk, this model still leaves out some parts of the financial deals, like regulatory capital or haircuts that apply to collateral. We select this model as a balance between a relatively realistic model and the tractability for the equations involved. The specific SGBM algorithm can be easily generalized to other models, as described in the outlook section.

Next, we introduce the mathematical model for our asset prices and the default events, as well as the notations that we use. Subsequently, we present the fundamental equations that we are solving in our demonstrator. Detailed financial interpretation as well as the model derivation will be left out and readers are referred to [3] for the description of this XVA model.

THE XVA MODEL

In this model, we take on a bank's perspective onto risk management. This perspective focuses on a non-centrally cleared financial derivative on underlying assets S, which are traded between the bank and its client and both parties may default. However, the default will not affect the underlying assets S.

Before we proceed to the BSDE description, which we are going to compute in the SGBM-XVA demonstrator, we introduce the meaning and financial background of the different terms involved.

As in Chapter 4, S_i denotes the underlying i-th asset, which follows the standard Black-Scholes model under our assumptions. Its dynamics are defined by the SDE:

$$dS_{t,i} = \bar{\mu}_i S_{t,i} dt + \bar{\sigma}_i S_{t,i} d\omega_{t,i},\ 1 \le i \le d,$$

where ω_t is a correlated d-dimensional Wiener process, where

$$d\omega_{t,i} d\omega_{t,j} = \rho_{ij} dt.$$

The constant vectors $\bar{\mu} = (\bar{\mu}_1, \dots, \bar{\mu}_d)^\top$ and $(\bar{\sigma}_1, \dots, \bar{\sigma}_d)^\top$ represent respectively the drift rate and the standard deviation of the financial assets. The parameters ρ_{ij} form a sym-

metric correlation matrix ρ,

$$\rho = \begin{pmatrix} 1 & \rho_{12} & \rho_{13} & \cdots & \rho_{1d} \\ \rho_{21} & 1 & \rho_{23} & \cdots & \rho_{2d} \\ \vdots & \vdots & \vdots & & \vdots \\ \rho_{d1} & \rho_{d2} & \rho_{d3} & \cdots & 1 \end{pmatrix},$$

which is invertible under our assumptions. We can relate the correlated Brownian motion ω to a standard, independent d-dimensional Brownian motion W by performing a Cholesky decomposition on ρ. They satisfy the equality

$$B_t = \mathfrak{C} W_t,$$

where $\mathfrak{C}$ is a lower triangular matrix with real and positive diagonal entries, and $\mathfrak{C}\mathfrak{C}^\top = \rho$.

The processes J^B and J^C are used to model the events of default for each party in the transaction. Mathematically, they are defined as counting processes,

$$J_t^B = \mathbf{1}_{\tau^B \leq t},$$

and

$$J_t^C = \mathbf{1}_{\tau^C \leq t},$$

where τ^B and τ^C are stopping times, denoting the random default times of the bank and the counterparty, respectively. The processes are assumed to have stochastic, time-dependent intensities, λ^B, λ^C, i.e.

$$\lambda_t^B \, dt = \mathbb{E}[dJ_t^B | \mathfrak{F}_{t-}]$$

and

$$\lambda_t^C \, dt = \mathbb{E}[dJ_t^C | \mathfrak{F}_{t-}].$$

Next, we discuss the financial derivatives traded within this model, where we use g to denote the terminal payoff for the portfolio. To mitigate counterparty risk, the *variation margins*, X, need to be computed for the two parties. As in [3], the values X are based on the market value of the financial product, and they are computed and re-adjusted frequently. When $X > 0$, the counterparty is posting collateral with the bank.

When one party in the financial contract defaults, the contract position needs to be closed. We denote the portfolio value at default (at time $\tau = \tau^B \wedge \tau^C$) by θ_τ, and it is given by

$$\begin{aligned} \theta_\tau :=& \mathbf{1}_{\tau^C < \tau^B} \theta_\tau^C + \mathbf{1}_{\tau^B < \tau^C} \theta_\tau^B \\ :=& \mathbf{1}_{\tau^C < \tau^B} (X_\tau + R^C (M_\tau - X_\tau)^+ + (M_\tau - X_\tau)^-) \\ &+ \mathbf{1}_{\tau^B < \tau^C} (X_\tau + (M_\tau - X_\tau)^+ + R^B (M_\tau - X_\tau)^-), \end{aligned}$$

where R^C, $R^B \in [0,1]$ are the recovery rates in the case the counterparty or the bank default, respectively. The variable M denotes the close-out value of the portfolio when any party defaults. We will give more details regarding M in a later section.

Next, we introduce the notation for the financial quantities in our model. The adapted stochastic vector processes $q^{\mathscr{S}}$ and $\gamma^{\mathscr{S}}$ are respectively the repo rate and the dividend yield of the underlying assets. The process r is the stochastic risk-less interest rate. The processes r^B and r^C are the yields of the risky zero coupon bonds of the bank and the counterparty, respectively. The process q^C is the repo rate for the bonds of the counterparty. The interest rate for the variation margin is given by r^X, and, finally, r^F is the cost of external funding. In order to simplify the expression of the demonstrator, we assume all the above processes to be constant.

FUNDAMENTAL BSDE AND REDUCED BSDE

In [3], a fundamental BSDE is derived through a hedging argument based on a replicating portfolio for the financial derivative. For the hedging portfolio $\hat{Y}$, including the counterparty credit risk, the dynamics of posting collateral and holding counterparty bonds for hedging, are given by,

$$\begin{aligned} -d\hat{Y}_t &= f(t,S_t,\hat{Y}_t,\hat{Z}_t,U_t^B,U_t^C)dt - \hat{Z}_t dW_t - U_{t-}^B dJ_t^B - U_t^C dJ_t^C, \quad t\in[0,\tau\wedge T], \\ \hat{Y}_{\tau\wedge T} &= \mathbf{1}_{\tau>T} g(S_T) + \mathbf{1}_{\tau\leq T}\theta_\tau, \end{aligned} \tag{5.1}$$

and the driver function is defined as

$$\begin{aligned} f(t,s,\hat{y},\hat{z},u^B,u^C) = &-\hat{z}(\mathrm{diag}(s)\mathrm{diag}(\bar{\sigma})\mathfrak{C})^{-1}(\mathrm{diag}(s)\bar{\mu} \\ &+ \mathrm{diag}(s)(\gamma^{\mathscr{S}}-q^{\mathscr{S}})) + (r^B-r)u^B + (r^C-q^C)u^C \\ &+ (r^X+r)X_t - r\hat{y} - (r^F-r)(\hat{y}-X_t+u^B)^-. \end{aligned}$$

The notation $\mathrm{diag}(\mathfrak{M})$ denotes a diagonal matrix with the terms of vector $\mathfrak{M}$ on the main diagonal. We compute the price of the derivative by approximating the values of the stochastic processes $(\hat{Y},\hat{Z},U^B,U^C)$ that solve the above BSDE. The price of the contract at time 0 is given by $\hat{Y}_0$.

As stated in the introduction chapter, the above BSDE is derived from the replicating portfolio argument. The main differences between the arguments in Section 1.2.3 and the derivation in [3] are found in the final payoff function, which is simply g in the basic, counterparty risk-free case but is θ_τ when considering the possibility of default and the dynamics of the replicating portfolio are then given by

$$d\hat{Y}_t = \delta_t^\top + \alpha_t^{\mathscr{B}} dP_t^{\mathscr{B}} + \alpha_t^{\mathscr{C}} dP_t^{\mathscr{C}} + \varphi_t^\top dB_t. \qquad \text{(Equation (19) in [3])}$$

These dynamics include the financing position of holding bonds of each party involved to hedge against the events of default.

Using the above dynamics, in combination with the stochastic processes of the trading assets, bonds and financing positions (Equations (1-2, 11-18) in [3]) and the general argument in Section 1.2.3, we can derive Equation (5.1). Again, readers may refer to Section 2 of [3] for further datails.

The main difficulty when dealing with this BSDE is that the terminal time is random, as it depends on the time of default. Therefore, this BSDE has to be transformed into a standard BSDE, to reduce the computational complexity, following the techniques used

in [4]. The authors in [4] showed that we may recover the solution of (5.1) by solving the BSDE,

$$\begin{aligned} -d\hat{\mathcal{Y}}_t = &\quad f(t,S_t,\hat{\mathcal{Y}}_t,\hat{\mathcal{Z}}_t,\theta^B_t-\hat{\mathcal{Y}}_t,\theta^C_t-\hat{\mathcal{Y}}_t)\,dt-\hat{\mathcal{Z}}_t\,dW_t,\\ \hat{\mathcal{Y}}_T = &\quad g(S_T), \qquad t\in[0,T], \end{aligned} \tag{5.2}$$

and using

$$\begin{aligned} \hat{Y}_t = &\quad \hat{\mathcal{Y}}_t\mathbf{1}_{t<\tau}+\theta_\tau\mathbf{1}_{t\geq\tau},\\ \hat{Z}_t = &\quad \hat{\mathcal{Z}}_t\mathbf{1}_{t\leq\tau},\\ U^B_t = &\quad (\theta^B_t-\hat{\mathcal{Y}}_t)\mathbf{1}_{t\leq\tau},\\ U^C_t = &\quad (\theta^C_t-\hat{\mathcal{Y}}_t)\mathbf{1}_{t\leq\tau}. \end{aligned} \tag{5.3}$$

Proposition 5.1 (Theorem A.1 in [3])**.** *If the pair of adapted stochastic processes* $(\hat{\mathcal{Y}},\hat{\mathcal{Z}})$ *solves Equation* (5.2)*, then the solution* $(\hat{Y},\hat{Z},U^B,U^C)$ *of Equation* (5.1) *is given by* (5.3)*.*

Idea of the proof. This technique considers the three possibilities of termination, i.e. no default, the bank's default and the counterparty default, and shows that the results in (5.3) solve (5.1) under all three situations. □

5

When we consider the total value adjustment to the financial option values, we compute an alternative price for the same financial contracts, under the assumption of risk-free dynamics (meaning no counterparty default). The value adjustment is defined as the difference between the two portfolio values (risky minus risk-free option values).

The default-free portfolio dynamics, in a repo funding setting, can be expressed as

$$-dY_t = (-Z_t(\mathrm{diag}(S_t)\mathrm{diag}(\bar{\sigma})\mathfrak{C}))^{-1}(\mathrm{diag}(S_t)\bar{\mu}+\mathrm{diag}(S_t)(\gamma^{\mathscr{S}}_t-q^{\mathscr{S}}_t))-r_tY_t)dt-Z_t\,dW_t,$$

and $Y_T=g(S_T)$.

DISCRETE SYSTEM

Here, we explain the setting for the variation margins X and close-out value M. The actual discrete system used in our demonstrator will be introduced as well.

In [5], the variation margin has been mandated to be "the full colleteralised mark-to-market exposure of non-centrally cleared derivatives". Therefore, at any time t, either the mark-to-market risk-free portfolio value Y or the counterparty risk adjusted value $\hat{Y}$ appear to be reasonable choices for the variation margin X. There are also two possible conventions for the portfolio close-out value, namely $M=Y$ and $M=\hat{Y}$. In this demonstration, the variation margin and the close-out value are assumed to be the mark-to-market price Y. In this case, the valuation problem results in a linear BSDE that contains an exogenous process Y. This means that both the variation margin and the close-out value are marked to the market, and are thus not dictated by the bank's internal model. Therefore, there is an additional stochastic process Y in the driver for $\hat{Y}$. If the expression of Y is not readily available, the numerical simulation of the $\hat{Y}$ system poses extra challenges, because we have to simulate Y and $\hat{Y}$ simultaneously.

The driver for the reduced BSDE *with counterparty credit risk* is now given by

$$\bar{f}(t,s,\hat{y},\hat{z},y) := f(t,s,\hat{y},\hat{z},y-\hat{y},y-\hat{y})$$

$$= -\hat{z}(\mathrm{diag}(s)\mathrm{diag}(\bar{\sigma})\mathfrak{C})^{-1}(\mathrm{diag}(s)\bar{\mu} + \mathrm{diag}(s)(\gamma^{\mathscr{S}} - q^{\mathscr{S}}))$$
$$+ (r^B + r^C - q^C + r^X)y - (r^B + r^C - q^C)\hat{y},$$

while the *counterparty risk-free* price Y can be expressed as

$$Y_t(x) = \mathbb{E}_t^x[e^{-r(T-t)}\Gamma_{t,T}g(S_T)],$$

where

$$\Gamma_{t,T} = \exp\left(-\int_t^T \frac{1}{2}\varphi_u^T\varphi_u du - \int_t^T \varphi_u^T dW_u\right),$$

and

$$\varphi_t := \mathfrak{C}^{-1}\mathrm{diag}(\bar{\sigma})^{-1}(\bar{\mu} + \gamma_t^{\mathscr{S}} - q_t^{\mathscr{S}}).$$

Indeed, the elementary expression of Y may be available in closed-form for some basic financial derivatives under the simple 1D Black-Scholes model. However, in order for our demonstrator to cover a wide range of products, we solve Y by SGBM instead.

Recall the discretization scheme in Equation (1.4), we can approximate the reduced BSDE with the following numerical scheme:

$$\begin{aligned}
\hat{Y}_{t_P}^\pi &= g(S_{t_P}^\pi), \quad \hat{\mathcal{Z}}_{t_N}^\pi = \nabla g(S_{t_P}^\pi)\sigma(t_P, S_{t_P}^\pi),\\
\hat{\mathcal{Z}}_{t_p}^\pi &= -\theta_2^{-1}(1-\theta_2)\mathbb{E}_{t_p}\left[\hat{\mathcal{Z}}_{t_{p+1}}^\pi\right] + \frac{1}{\Delta t}\theta_2^{-1}\mathbb{E}_{t_k}\left[\hat{\mathscr{Y}}_{t_{p+1}}^\pi \Delta W_p{}^\top\right]\\
&\quad + \theta_2^{-1}(1-\theta_2)\mathbb{E}_{t_p}\left[\bar{f}(t_{p+1}, S_{t_{p+1}}^\pi, \hat{\mathscr{Y}}_{t_{p+1}}^\pi, \hat{\mathcal{Z}}_{t_{p+1}}^\pi, Y_{t_{p+1}})\Delta W_p^\top\right], p = P-1, \dots, 0,\\
\hat{\mathscr{Y}}_{t_p}^\pi &= \mathbb{E}_{t_p}\left[\hat{\mathscr{Y}}_{t_{p+1}}^\pi\right] + \Delta t\theta_1 \bar{f}(t_p, S_{t_p}^\pi, \hat{\mathscr{Y}}_{t_p}^\pi, \hat{\mathcal{Z}}_{t_p}^\pi, Y_{t_p})\\
&\quad + \Delta t(1-\theta_1)\mathbb{E}_{t_p}\left[\bar{g}(t_{p+1}, S_{t_{p+1}}^\pi, \hat{\mathscr{Y}}_{t_{p+1}}^\pi, \hat{\mathcal{Z}}_{t_{p+1}}^\pi, Y_{t_{p+1}})\right], \ p = P-1, \dots, 0,
\end{aligned}$$

where $0 \le \theta_1 \le 1$ and $0 < \theta_2 \le 1$ and assuming a fixed uniform time-step $\Delta_t = t_{p+1} - t_p, \forall p$.

Furthermore, if we include the simulation of Y and include explicitly the driver functions, we have to solve the following system of discretized BSDEs:

$$\begin{aligned}
v_P^\pi(s) &= \hat{v}_P^\pi(s) = g(s), \quad z_P^\pi(s) = \hat{z}_P^\pi(s) = \nabla g(s)\sigma(t_P, s),\\
z_{p,q}^\pi(s) &= -\frac{1-\theta_2}{\theta_2}\mathbb{E}_{t_p}^s\left[z_{p+1,q}^\pi(S_{t_{p+1}}^\pi)\right]\\
&\quad + \frac{1}{\theta_2}\left[1-(1-\theta_2)r\Delta t\right]\mathbb{E}_{t_p}^s\left[y_{p+1}^\pi(S_{t_{p+1}}^\pi)\frac{\Delta W_{p,q}}{\Delta t}\right]\\
&\quad - \frac{1-\theta_2}{\theta_2}\Delta t\left(\frac{\bar{\mu} + \gamma^{\mathscr{S}} - q^{\mathscr{S}}}{\bar{\sigma}}\right)^\top (\mathfrak{C}^{-1})^\top \mathbb{E}_{t_p}^s\left[z_{p+1}^\pi(S_{t_{p+1}}^\pi)^\top \frac{\Delta W_{p,q}}{\Delta t}\right],\\
&\quad p = P-1, \dots, 0, \quad q = 1, \dots d,\\
y_p^\pi(s) &= \frac{1-(1-\theta_1)r\Delta t}{1+\theta_1 r\Delta t}\mathbb{E}_{t_p}^s\left[y_{p+1}^\pi(S_{t_{p+1}}^\pi)\right]\\
&\quad - \frac{\theta_1\Delta t}{1+\theta_1 r\Delta t}\left(\frac{\bar{\mu} + \gamma^{\mathscr{S}} - q^{\mathscr{S}}}{\bar{\sigma}}\right)^\top (\mathfrak{C}^{-1})^\top (z_p^\pi(s))^\top
\end{aligned}$$

$$
\begin{aligned}
&- \frac{(1-\theta_1)\Delta t}{1+\theta_1 r \Delta t}\left(\frac{\bar{\mu}+\gamma^{\mathscr{S}}-q^{\mathscr{S}}}{\bar{\sigma}}\right)^{\top}(\mathfrak{C}^{-1})^{\top}\mathbb{E}^s_{t_p}\left[(z^{\pi}_{p+1}(S^{\pi}_{t_{p+1}}))^{\top}\right],\\
&p = P-1,\dots,0,\\
\hat{z}^{\pi}_{p,q}(s) =& -\frac{1-\theta_2}{\theta_2}\mathbb{E}^s_{t_p}\left[\hat{z}^{\pi}_{p+1,q}(S^{\pi}_{t_{p+1}})\right]\\
&+\frac{1}{\theta_2}\left[1-(1-\theta_2)\Delta t(r^B+r^C-q^C)\right]\mathbb{E}^s_{t_p}\left[\hat{y}^{\pi}_{p+1}(S^{\pi}_{t_{p+1}})\frac{\Delta W_{p,q}}{\Delta t}\right]\\
&-\frac{1-\theta_2}{\theta_2}\Delta t\left(\frac{\bar{\mu}+\gamma^{\mathscr{S}}-q^{\mathscr{S}}}{\bar{\sigma}}\right)^{\top}(\mathfrak{C}^{-1})^{\top}\mathbb{E}^s_{t_p}\left[(\hat{z}^{\pi}_{p+1}(S^{\pi}_{t_{p+1}}))^{\top}\frac{\Delta W_{p,q}}{\Delta t}\right]\\
&+\frac{1-\theta_2}{\theta_2}\Delta t(r^B+r^C-q^C+r^X)\mathbb{E}^s_{t_p}\left[y^{\pi}_{p+1}(S^{\pi}_{t_{p+1}})\frac{\Delta W_{p,q}}{\Delta t}\right],\\
&p = P-1,\dots,0,\quad q=1,\dots,d,\\
\hat{y}^{\pi}_p(s) =& \frac{1-(1-\theta_1)\Delta t(r^B+r^C-q^C)}{1+\theta_1\Delta t(r^B+r^C-q^C)}\mathbb{E}^s_{t_p}\left[\hat{y}^{\pi}_{p+1}(S^{\pi}_{t_{p+1}})\right]\\
&-\frac{\theta_1\Delta t}{1+\theta_1\Delta t(r^B+r^C-q^C)}\left(\frac{\bar{\mu}+\gamma^{\mathscr{S}}-q^{\mathscr{S}}}{\bar{\sigma}}\right)^{\top}(\mathfrak{C}^{-1})^{\top}(\hat{z}^{\pi}_p(s))^{\top}\\
&-\frac{(1-\theta_1)\Delta t}{1+\theta_1\Delta t(r^B+r^C-q^C)}\left(\frac{\bar{\mu}+\gamma^{\mathscr{S}}-q^{\mathscr{S}}}{\bar{\sigma}}\right)^{\top}(\mathfrak{C}^{-1})^{\top}\mathbb{E}^s_{t_p}\left[(\hat{z}^{\pi}_{p+1}(S^{\pi}_{t_{p+1}}))^{\top}\right]\\
&+\frac{\theta_1\Delta t(r^B+r^C-q^C+r^X)}{1+\theta_1\Delta t(r^B+r^C-q^C)}y_p(s)\\
&+\frac{(1-\theta_1)\Delta t(r^B+r^C-q^C+r^X)}{1+\theta_1\Delta t(r^B+r^C-q^C)}\mathbb{E}^s_{t_p}\left[y^{\pi}_{p+1}(S^{\pi}_{t_{p+1}})\right], p=P-1,\dots,0. \qquad (5.4)
\end{aligned}
$$

This system is the one that we have implemented in our demonstrator. Note that the notation $\left(\frac{\bar{\mu}+\gamma^{\mathscr{S}}-q^{\mathscr{S}}}{\bar{\sigma}}\right)$ is a shorthand for the vector $\left(\frac{\bar{\mu}_i+\gamma^{\mathscr{S}}_i+q^{\mathscr{S}}_i}{\bar{\sigma}_i}\right)_{1\le i\le d}$.

To close-out this section, we would like to mention that there is an analytic expression for the reference price under the current setting, which is given by

$$
\begin{aligned}
\hat{\mathscr{Y}}_t =& \mathbb{E}_t\left[e^{-\int_t^T(r^B+r^C-q^C)du}\Gamma_{t,T}g(S_T)+\int_t^T e^{-\int_t^s(r^B+r^C-q^C)du}\Gamma_{t,s}(r^X+r^B+r^C-q^C)Y_s ds\right]\\
=& e^{-(r^B+r^C-q^C-r)(T-t)}Y_t+(r^X+r^B+r^C-q^C)\mathbb{E}_t\left[\int_t^T e^{-(r^B+r^C-q^C)(s-t)}\Gamma_{t,s}Y_s ds\right]
\end{aligned}
$$

There is however no elementary expression or simple way to evaluate this quantity.

5.2.4. FUNCTION DESCRIPTIONS

There are two parallel versions of our algorithm, both are written in Python. One of them is based on generic Python code and the Numpy, Scipy and Numba packages for performance and efficiency, which we would refer to as Version 1 from now on for simplicity. The second one makes use of the CUDA toolkit, the CUDA Python portion of the Numba package and the pyculib library and shall be referred to as Version 2. A `numerical_test`

script has been provided for running basic comparison tests between the two versions under a predefined test setting.

Next, we list the main functions within our implementation. Since the two versions have similar architecture, we would only present Version 2. We have:

- the main function for the whole algorithm: `cuda_sgbm_xva`;
- the forward simulation function for the basis values and partition ordering: `cuda_jit_montecarlo`
- the main function for the backward approximation stage: `cuda_sgbminnerblock`
- the parallel regression function: `cuda_regression`
- an example class to store all the information related to the test case, including Equation (5.4): `ArithmeticBasketPut`

All of the above functions form the complete demonstrator but each individual component can be used or replaced separately, which gives us flexibility for future development.

Version 1 only depends on Python and the Numba library. In addition to that, Version 2 also requires the CUDA driver and the pyculib library.

The platform independent Python 3 code generated during the current study is available at https://github.com/kwchau/sgbm_xva under the MIT license.

5.3. NUMERICAL EXPERIMENTS

In this section, we present the tests we have implemented in our demonstrator. In these tests we assume that the bank sells a portfolio consisting of an arithmetic put option, whose payoff is given by

$$g(s) = -\left(K - \frac{1}{d}\sum_{i=1}^{d} s_i\right)^+,$$

where K is the strike price. The detailed setting for the numerical test is presented in Table 5.1. This set of parameters is based on the one used in [1] and it has been adopted here as an easy to scale up problem. The main purpose of the tests is to investigate whether the code can be easily extended to solving high-dimensional problems, so we choose a set of parameters whose properties do not change when we change the dimensionality of the problem.

Note that here we have adopted a test case with the borrowing rate and lending rate being equal for the purpose of comparing our default-free prices to a previously known result. Additional tests have been performed to assure that the convergence behavior will not be affected by different sets of parameters.

The numerical test has been conducted on a common desktop computer with an Intel(R) Core(TM) i5-7400 processor and a GeForce GTX 1080 graphic card.

Table 5.1: Model parameters

S_0	$q^{\mathscr{S}}$	$\gamma^{\mathscr{S}}$	$\bar{\mu}$	$\bar{\sigma}$	ρ
$\begin{pmatrix} 40 \\ \vdots \\ 40 \end{pmatrix}$	$\begin{pmatrix} 0.06 \\ \vdots \\ 0.06 \end{pmatrix}$	$\begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0.06 \\ \vdots \\ 0.06 \end{pmatrix}$	$\begin{pmatrix} 0.2 \\ \vdots \\ 0.2 \end{pmatrix}$	$\begin{pmatrix} 1 & 0.25 & & \dots & 0.25 \\ 0.25 & 1 & 0.25 & \dots & 0.25 \\ \vdots & & \ddots & & \vdots \\ 0.25 & & \dots & & 1 \end{pmatrix}$
r	r^B	r^C	q^C	r^X	
0.06	0	0	0	0.1	
K	T	(θ_1, θ_2)			
40	1	(0,1)			

Table 5.2: Testing parameters (paths, steps, bundles)

Parameters	Low dimension (≤ 15)	High dimension (> 15)
1	(2048, 4, 8)	(16384, 4, 8)
2	(4096, 8, 16)	(32768, 8, 16)
3	(8192, 12, 32)	(65536, 12, 32)
4	(16384, 16, 64)	(131072, 16, 64)
5	(32768, 20, 128)	(262144, 20, 128)

5.3.1. PERFORMANCE OF GPU COMPUTING

The first set of tests is performed to check whether there is any benefit of GPU computing for our demonstrator. We perform the same test 10 times separately with the two versions and take the averaged computing time over these 10 runs to compare the efficiency between them. Moreover, in accordance with the work in Chapter 4, we use 5 sets of parameters for testing the consistency of this SGBM algorithm. The parameters are presented in Table 5.2. We have conducted tests with the stock price dimensionality being 1, 5 and 10. The results are shown in the Tables 5.3, 5.4 and 5.5.

As we expected from the theoretical work, the approximate values converge with respect to the progressively increasing grid sizes and the standard deviations decrease as we progress through the parameters sets. More importantly, there is a clear speed-up of Version 2 over Version 1. The biggest improvement comes from the forward simulation stage where we run each independent sample in parallel, and greatly reduce the computational time. There still seems to be room for improvement regarding the backward simulation stage as even if we perform the regression in parallel on the GPU in Version 2 instead of sequentially in Version 1, the speed-up is not obvious. The main reason may be that some part of the code is not performed on the GPU to keep some flexibility of our demonstrator. Some optimization is only available for the basic Python code but not for CUDA Python.

Nevertheless, our tests show that CUDA Python improves the efficiency and may provide a good balance between the speed of execution and the speed of development. In

Table 5.3: Test result for dimension 1

Ver.	Parameters	Risk Free Price (Std. Deviation)	Risk Adjusted Price (Std. Deviation)	Time (s)	Speedup
1	Set 1	−2.060 (0.01)	−2.402 (0.01)	0.42	
2		−2.049 (0.007)	−2.389 (0.008)	0.34	1.24
1	Set 2	−2.066 (0.003)	−2.408 (0.003)	0.93	
2		−2.063 (0.002)	−2.405 (0.002)	0.24	3.88
1	Set 3	−2.065 (0.002)	−2.406 (0.002)	2.58	
2		−2.068 (0.003)	−2.410 (0.004)	0.36	7.17
1	Set 4	−2.066 (0.002)	-2.407 (0.002)	6.81	
2		−2.066 (0.001)	−2.407 (0.001)	0.66	10.32
1	Set 5	−2.066 (0.0006)	−2.407 (0.0007)	16.69	
2		−2.066 (0.0006)	−2.407 (0.0007)	1.21	13.79

particular, as the workload increases due to using more time steps, samples and bundles, the speed-up get higher due to GPU computing.

Finally, whereas the build-in floating point format of the GPU is 32 bits, 64 bits format is needed for ensuring high accuracy with our result.

5.3.2. SCALABILITY

Next, we would like to check if our algorithm can be used for high-dimensional test cases. The setting is the same as in the last section. However, Version 1 is too time-consuming for a common desktop, so the focus is on Version 2. We perform tests up to 40 dimensions, as shown in Table 5.6.

It can be seen that with the help of the GPU, our method can be generalized to higher dimensions. The application of GPU computing definitely expands the applicability of our method. However, we also notice that the time ratio increase is higher than the dimensional ratio. This is probably related to the architecture and the build-in hardware of the GPU. It maybe worthwhile to tune the GPU setting within our demonstrator corresponding to the GPU at hand, but this is hardware-dependent.

To sum up, the application of GPU computing not only speeds up our algorithm, but it also allows us to deal with more demanding problems with about the same hardware.

5.4. CONCLUSION AND OUTLOOK

5.4.1. CONCLUSION

Although we just developed a demonstrator, our study gives us interesting insight in the performance of the SGBM algorithm for BSDEs. We have shown that we can apply the SGBM algorithm to solve high-dimensional BSDEs with the help of GPU computing, which is an important feature for using BSDEs in practice. With a suitable BSDE model, we can deal with complicated financial risk management problems with our solver and since our code is based on a general framework for BSDEs, our demonstrator can be

Table 5.4: Test result for dimension 5

Ver.	Parameters	Risk Free Price (Std. Deviation)	Risk Adjusted Price (Std. Deviation)	Time (s)	Speedup
1	Set 1	−1.007 (0.006)	−1.175 (0.007)	1.89	
2		−1.001 (0.01)	−1.167 (0.01)	0.53	3.57
1	Set 2	−1.011 (0.002)	−1.178 (0.002)	6.99	
2		−1.010 (0.002)	−1.178 (0.002)	0.86	8.13
1	Set 3	−1.011 (0.001)	−1.178 (0.001)	20.59	
2		−1.011 (0.002)	−1.178 (0.002)	2.20	9.36
1	Set 4	−1.012 (0.0008)	−1.179 (0.0010)	54.24	
2		−1.012 (0.0009)	−1.180 (0.001)	5.34	10.16
1	Set 5	−1.013 (0.0005)	−1.180 (0.0005)	134.61	
2		−1.012 (0.0005)	−1.180 (0.0006)	12.28	10.96

adopted to different pricing and valuation situations. We also demonstrated that we can incorporate GPU computing into a native Python code. We believe that this work serves as a basis for developing BSDE-based software for financial applications.

We encourage readers to download and test our demonstrator, which solves the problem which was stated in this work. The aim is to develop this tool further in terms of financial applications and computational efficiency.

Next, we will mention some possible generalizations for our algorithm as a guideline to future use. We will divide the outlook into two parts, a financial and a computational part.

5.4.2. FINANCIAL OUTLOOK

In this work, we used the classical Black-Scholes model for the stock dynamics. It is of interest to generalize the stock model by other diffusion SDEs, of the form:

$$\begin{cases} dS_t &= \mu(t,S_t)dt + \sigma(t,S_t)dW_t, \\ S_0 &= s_0. \end{cases}$$

This would already result in a different implementation regarding the SGBM regressor-later component.

We may also alter the model dynamics to better fit the market, for example, with the inclusion of initial margins and capital requirement. Recall that in Section 5.2.3 we have made some assumptions about the variation margin X and the close-out value M. Simply by adjusting these assumptions, a completely different BSDE driver may result. There are four possible combinations for the variation margin and the close-out value. Each choice gives rise to a different BSDE driver, which can be seen in Table 5.7.

When X and M are the adjusted values $\hat{Y}$, it results in a linear BSDE for $\hat{Y}$. This means that the variation margin and the close-out value are marked to the model. As all the values are marked in $\hat{Y}$ and both values match, the equation reduces to an option pricing equation, with different interest rates. When both of them are the risk-free process Y, a

Table 5.5: Test result for dimension 10

Ver.	Parameters	Risk Free Price (Std. Deviation)	Risk Adjusted Price (Std. Deviation)	Time (s)	Speedup
1	Set 1	−0.834 (0.005)	−0.973 (0.006)	6.38	
2		−0.838 (0.004)	−0.978 (0.004)	1.15	5.55
1	Set 2	−0.839 (0.002)	−0.977 (0.002)	24.74	
2		−0.840 (0.002)	−0.979 (0.002)	2.96	8.36
1	Set 3	−0.841 (0.001)	−0.980 (0.002)	74.12	
2		−0.840 (0.001)	−0.979 (0.001)	8.64	8.58
1	Set 4	−0.841 (0.0008)	−0.980 (0.0009)	198.08	
2		−0.841 (0.0006)	−0.979 (0.0007)	21.58	9.18
1	Set 5	−0.841 (0.0003)	−0.980 (0.0003)	498.71	
2		−0.841 (0.0007)	−0.980 (0.0008)	53.51	9.32

linear BSDE that contains an exogenous process Y results. This means that both the variation margin and the close-out value are marked to the market and are not dictated by the internal model. Then there is an additional stochastic process Y in the driver for $\hat{Y}$, which is the case we have used in this work.

Finally, when there is a mismatch between M and X, the resulting BSDE is no longer linear. Take Case 4 as an example, it implies that the variation margin is collected according to an internal model while the close-out value is given by the market. The mismatch between the variation margin and the close-out value results in non-linear terms in the driver. These non-linear terms come from the cash flow when one party defaults.

We should notice that since we have a general BSDE algorithm in the form of SGBM, we may adapt our code to these new models by simply changing the model setting in the example class without changing the actual function. This is one of the advantages of using BSDEs in pricing and risk management.

5.4.3. COMPUTATIONAL OUTLOOK

As we have mentioned before, one possible way of improvement is to sacrifice some flexibility and move the remaining solver parts of the code also to the GPU. Alternatively, one may further optimize the code within the GPU as the GPU set of routines seems to be still developing.

Furthermore, other features may be included in the software, for example, different payoff functions or a different SGBM regression basis. Finally, to have a fully independent software, a user interface and modular code are recommended.

Table 5.6: Test result for high-dimensional cases with Version 2

Dim.	Parameters	Risk Free Price (Std. Deviation)	Risk Adjusted Price (Std. Deviation)	Time (s)
15	Set 1	−0.777 (0.005)	−0.906 (0.006)	2
	Set 2	−0.780 (0.002)	−0.909 (0.002)	8
	Set 3	−0.779 (0.0008)	−0.908 (0.0009)	22
	Set 4	−0.781 (0.0006)	−0.910 (0.0007)	60
	Set 5	−0.781 (0.0004)	−0.909 (0.0004)	144
20	Set 6	−0.746 (0.002)	−0.870 (0.002)	29
	Set 7	−0.748 (0.0007)	−0.872 (0.0008)	116
	Set 8	−0.749 (0.0004)	−0.872 (0.0005)	345
	Set 9	−0.749 (0.0002)	−0.873 (0.0003)	911
	Set 10	−0.749 (0.0001)	−0.873 (0.0001)	2278
30	Set 6	−0.715 (0.002)	−0.834 (0.002)	84
	Set 7	−0.716 (0.0008)	−0.835 (0.0009)	337
	Set 8	−0.717 (0.0005)	−0.836 (0.0006)	997
	Set 9	−0.718 (0.0002)	−0.836 (0.0002)	2675
	Set 10	−0.718 (0.0001)	−0.837 (0.0002)	6715
40	Set 6	−0.698 (0.001)	−0.814 (0.002)	176
	Set 7	−0.701 (0.0007)	−0.817 (0.0009)	701
	Set 8	−0.701 (0.0005)	−0.817 (0.0006)	2086
	Set 9	−0.702 (0.0002)	−0.818 (0.0003)	5562
	Set 10	-0.702 (0.0002)	−0.818 (0.0002)	13893

REFERENCES

[1] A. Leitao and C. W. Oosterlee, *GPU acceleration of the Stochastic Grid Bundling Method for early-exercise options,* International Journal of Computer Mathematics **92**, 2433 (2015).

[2] *Pyculib website, Retrieved from "https://pyculib.readthedocs.io/en/latest/",* .

[3] A. Lesniewski and A. Richter, *Managing counterparty credit risk via BSDEs,* ArXiv e-prints , arXiv:1608.03237 (2016).

[4] I. Kharroubi, T. Lim, and A. Ngoupeyou, *Mean-variance hedging on uncertain time horizon in a market with a jump,* Applied Mathematics & Optimization **68**, 413 (2013).

[5] Basel Committee on Banking Supervision, *Margin requirements for non-centrally cleared derivatives* (2015).

Table 5.7: Different BSDE drivers for various choices of X and M.

Case	X	M	$f(t,S_t,\hat{\mathcal{Y}}_t,\hat{\mathcal{Z}}_t,\theta_t^B-\hat{\mathcal{Y}}_t,\theta_t^C-\hat{\mathcal{Y}}_t)$
1	$\hat{Y}$	$\hat{Y}$	$-\hat{\mathcal{Z}}_t\sigma(t,S_t)^{-1}(\mu(t,S_t)+\text{diag}(\gamma_t^{\mathcal{S}}-q^{\mathcal{S}})S_t)+r_t^X\hat{Y}_t$
2	Y	Y	$-\hat{\mathcal{Z}}_t\sigma(t,S_t)^{-1}(\mu(t,S_t)+\text{diag}(\gamma_t^{\mathcal{S}}-q^{\mathcal{S}})S_t)+(r_t^B+r_t^C-q_t^C)(Y_t-\hat{Y}_t)+r_t^XY_t$
3	Y	$\hat{Y}$	$-\hat{\mathcal{Z}}_t\sigma(t,S_t)^{-1}(\mu(t,S_t)+\text{diag}(\gamma_t^{\mathcal{S}}-q^{\mathcal{S}})S_t)+(r_t^B+r_t^C-q_t^C)(Y_t-\hat{\mathcal{Y}}_t)+r_t^XY_t+(r_t^B+R^Cr_t^C-r_t-R^Cq_t^C)(\hat{Y}_t-Y_t)^+ +[(r_t^B-r_t^F)R^B+r_t^C-q_t^C](\hat{Y}_t-Y_t)^-$
4	$\hat{Y}$	Y	$-\hat{\mathcal{Z}}_t\sigma(t,S_t)^{-1}(\mu(t,S_t)+\text{diag}(\gamma_t^{\mathcal{S}}-q^{\mathcal{S}})S_t)+r_t^XY_t+[(r_t^B-r_t)+R^C(r_t^C-q_t^C)](Y_t-\hat{Y}_t)^+ +[(r^B-r_t^F)R^B+r_t^C-q_t^C](Y_t-\hat{Y}_t)^-$

6

BRANCHING METHOD FOR THE PRICING OF AMERICAN OPTIONS

In this chapter, we study the connection between BSDEs and partial differential equations (PDEs). We also introduce a numerical method for pricing American options based on the above connection and branching processes. Moreover, based on our numerical results, we show that only appropriately chosen algorithms can be used to approximate the relevant BSDEs.

In particular, we extend the viscosity solution characterization proved in [2] for call/put American option prices to the case of a general payoff function in a multi-dimensional setting: the price satisfies a semilinear reaction/diffusion type equation. Based on this, we propose two new numerical schemes inspired by the branching processes based algorithm of [3]. Our numerical experiments show that approximating the discontinuous driver of the associated reaction/diffusion PDE by local polynomials is not efficient, while a simple randomization procedure provides very good results.

6.1. INTRODUCTION

An American option is a financial contract which can be exercised by its holder at any time until a given future date, called maturity. When it is exercised, the holder receives a payoff that depends on the value of the underlying assets.

Here we introduce some common mathematical tools that have been used to price this kind of option. Let us first consider the case of a single stock (non-dividend paying) market under the famous Black Scholes setting from [4] and used throughout this thesis. Namely, let $(\Omega, \mathbb{F}, (\mathfrak{F}_t)_{t\geq 0}, \mathbb{P})$ be a filtered probability space carrying a standard

This chapter is based on the article 'Monte-Carlo methods for the pricing of American options: a semilinear BSDE point of view', published in ESAIM: Proceedings and Surveys [1]. The majority of research is done during the CEMRACS 2017 research school within a research team consists of me, Arij Manai and Ahmed Sid-Ali, under the supervision of Prof. Bruno Bouchard. All involved parties agreed for the inclusion of this piece in the thesis.

one-dimensional Brownian motion W and let us model the stock price process X as

$$X_s = x \exp\big((\bar{r} - \frac{\bar{\sigma}^2}{2})(s-t) + \bar{\sigma}(W_s - W_t)\big),\ s \geq t,$$

under the risk neutral probability. Here, $x > 0$ is the stock price at time t, $\bar{r} > 0$ is the risk-free interest rate and $\bar{\sigma} > 0$ is the volatility. Then, the arbitrage free value at time t of an American option maturing at $T \geq t$ is given by

$$V(t,x) = \sup_{\tau \in \mathcal{T}_{[t,T]}} \mathbb{E}[e^{-\bar{r}(\tau - t)} g(X_\tau)] \tag{6.1}$$

where $\mathcal{T}_{[t,T]}$ is the collection of $[t,T]$-valued stopping times, and g is the payoff function, say continuous, see e.g. [5] and the references therein. Typical examples are

$$g(x') = \begin{cases} (x' - K)^+, & \text{for a call option;} \\ (K - x')^+, & \text{for a put option.} \end{cases}$$

where $K > 0$ denotes the strike price.

By construction, $V(\cdot, X) \geq g(X)$, and the option should be exercised only when $V(\cdot, X) = g(X)$. This leads us to define the following two regions:

- the continuation region:

$$C = \{(t,x) \in [0,T) \times \mathbb{R}^+ : V(t,x) > g(x)\}$$

- the stopping (or the exercise) region:

$$S = \{(t,x) \in [0,T) \times \mathbb{R}^+ : V(t,x) = g(x)\}.$$

These are the basics of the common formulation of the American option price as a free boundary problem, which already appears in McKean [6]: V solves a heat-equation type linear parabolic problem on C and equals g on S, with the constraint of being always greater than g. Another formulation is based on the quasi-variational approach of Bensoussan and Lions [7]: the price solves (at least in the viscosity solution sense) the quasi-variational partial differential equation

$$\begin{cases} \min\big(\bar{r}\varphi - \mathcal{L}_{BS}\varphi, \varphi - g\big) = 0, & \text{on } [0,T) \times \mathbb{R}^+ \\ \varphi(T,\cdot) = g, & \text{on } \mathbb{R}^+ \end{cases}$$

in which $\mathcal{L}_{BS}$ is the Dynkin operator associated to X:

$$\mathcal{L}_{BS} = \partial_t + \bar{r} x D + \frac{1}{2}\bar{\sigma}^2 x^2 D^2$$

where D and D^2 are the Jacobian and Hessian operators.

In this chapter, we focus on another formulation that can be found in [2], see also [8] and the references therein. We study how can we use a BSDE formation to approximate this type of PDE.

The American option valuation problem can be stated in terms of a semilinear Black and Scholes partial differential equation set on a fixed domain, namely:

$$\begin{cases} \bar{r}\varphi - \mathscr{L}_{BS}\varphi = \mathfrak{y}(\cdot,\varphi), & \text{on } [0,T)\times\mathbb{R}^+, \\ \varphi(T,\cdot) = g, & \text{on } \mathbb{R}^+, \end{cases} \tag{6.2}$$

where $\mathfrak{y}$ is a nonlinear reaction term defined as

$$\mathfrak{y}(x,\varphi(t,x)) = c(x)H(g(x)-\varphi(t,x)) = \begin{cases} 0 & \text{if} \quad g(x) < \varphi(t,x) \\ c(x) & \text{if} \quad g(x) \geq \varphi(t,x), \end{cases}$$

in which c is a certain cash flow function, e.g. $c = \bar{r}K$ for a put option, and H is the Heaviside function.

Note that this semilinear Black and Scholes equation does not make sense if we consider classical solutions because of the discontinuity of $y \to \mathfrak{y}(x,y)$. It has to be considered in the discontinuous viscosity solution sense, see e.g. Crandall, Ishii and Lions [9]. Namely, even if V is continuous, the supersolution property should be stated in terms of the lower-semicontinuous envelope of $\mathfrak{y}$, the other way round for the subsolution property. This means in particular that the super- and subsolution properties are not defined with respect to the same operator. Still, thanks to the very specific monotonicity of $y \to \mathfrak{y}(x,y)$, it is proved in [2] that, within the Black and Scholes model, the American option price is the unique solution of (6.2) in the appropriate sense.

In this work, we first extend the characterization of [2] in terms of (6.2) to a general payoff function and to a general market model, see Section 6.2. Then, we suggest two numerical schemes based on this formulation. The general idea consists in (formally) identifying the solution V of (6.2) to the solution (Y,Z) of the backward stochastic differential equation

$$Y = e^{-\bar{r}T}g(X_T) + \int_\cdot^T e^{-\bar{r}s}\mathfrak{y}(X_s, e^{\bar{r}s}Y_s)ds - \int_\cdot^T Z_s dW_s$$

by $e^{-\bar{r}\cdot}V(\cdot,X) = Y$.

In the first algorithm, we follow the approach of Bouchard et al. [3] and approximate the nonlinear driver $\mathfrak{y}$ by local polynomials so as to be able to apply an extended version of the pure forward branching processes based the Feynman-Kac representation of the Kolmogorov-Petrovskii-Piskunov equation, see [10, 11]. Unfortunately, our numerical experiments show that this algorithm is quite unstable, see Section 6.3.1.

In the second algorithm, we do not try to approximate $\mathfrak{y}$ by local polynomials but in place regularize it with a noise by replacing $\mathfrak{y}(X, e^{\bar{r}\cdot}Y)$ by $c(X)\mathbf{1}_{\{g(X)+\epsilon \geq e^{\bar{r}\cdot}Y\}}$, in which ϵ is an independent random variable. When the variance of ϵ vanishes, this provides a converging estimator. For ϵ given, the corresponding Y is estimated by using the approach of Bouchard et al. [3] with (random) polynomial $(t,x,y,y') \mapsto c(x)\mathbf{1}_{\{g(x)+\epsilon \geq e^{\bar{r}t}y'\}}$ and particles that can only die (without creating any children). This algorithm turns out to be very precise, see Section 6.3.2.

6.2. NON-LINEAR PARABOLIC EQUATION REPRESENTATION

From now on, we take Ω as the space of $\mathbb{R}^d$-valued continuous maps on $[0,T]$ starting at 0, endowed with the Wiener measure $\mathbb{P}$. We let W denote the canonical process and let

$(\mathfrak{F}_t)_{t\le T}$ be its completed filtration. Given $t\in[0,T]$ and $x\in(\mathbb{R}^+)^d$, we consider a financial market with d stocks whose price process $X^{t,x}$ evolves according to

$$X^{t,x}=x+\int_t^{\cdot}\bar{r}X_s^{t,x}ds+\int_t^{\cdot}\sigma(s,X_s^{t,x})dW_s \tag{6.3}$$

in which $\bar{r}\in\mathbb{R}$ is a constant[1], the risk free interest rate, and $\sigma:[0,T]\times(\mathbb{R}^+)^d\mapsto\mathbb{R}^{d\times d}$ is a matrix valued-function that is assumed to be continuous and uniformly Lipschitz in its second component. We also assume that $\hat{\sigma}:(t',x')\in[0,T]\times(\mathbb{R})^d\mapsto \mathrm{diag}[x']^{-1}\sigma(t',x')$ is uniformly Lipschitz in its second component and bounded, where $\mathrm{diag}[x']$ stands for the diagonal matrix with i-th diagonal entry equal to the i-th component of x'. This implies that $X^{t,x}$ takes values in $(\mathbb{R}^+)^d$ whenever $x\in(\mathbb{R}^+)^d$.

We also assume that $\mathbb{P}$ is the only (equivalent) probability measure under which $e^{-\bar{r}(\cdot-t)}X^{t,x}$ is a (local) martingale, for $(t,x)\in[0,T]\times(\mathbb{R}^+)^d$. Then, given a continuous payoff function $g:(\mathbb{R}^+)^d\to\mathbb{R}$, with polynomial growth, the price of the American option with payoff g is given by

$$V(t,x)=\sup_{\tau\in\mathcal{T}_{[t,T]}}\mathbb{E}[e^{-\bar{r}(\tau-t)}g(X_\tau^{t,x})], \tag{6.4}$$

in which $\mathcal{T}_{[t,T]}$ is the collection of $[t,T]$-valued stopping times. See [5].

6

Remark 6.1. Note that $(t,x)\in[0,T]\times(\mathbb{R}^+)^d\mapsto V(t,x)$ is continuous with polynomial growth follows from standard estimates under the above assumptions. In particular, the set $\{(t,x)\in[0,T]\times(\mathbb{R}^+)^d:V(t,x)=g(x)\}$ is closed.

The aim of this section is to establish the relevant PDE representation for general American option pricing. We prove that V is a viscosity solution of the non-linear parabolic equation

$$\begin{aligned}\bar{r}\varphi-\mathscr{L}\varphi-\mathfrak{y}(\cdot,\varphi)=0 \quad &\text{on } [0,T)\times(\mathbb{R}^+)^d\\ \varphi(T,\cdot)=g \quad &\text{on } (\mathbb{R}^+)^d,\end{aligned} \tag{6.5}$$

for a suitable reaction function $\mathfrak{y}$ on $(\mathbb{R}^+)^d\times\mathbb{R}$. In the above, $\mathscr{L}$ denotes the Dynkin operator associated to (6.3):

$$\mathscr{L}\varphi(t',x')=\partial_t\varphi(t',x')+\langle\bar{r}x',D\varphi(t',x')\rangle+\frac{1}{2}\mathrm{Tr}[\sigma\sigma^\top D^2\varphi](t',x'),$$

for a smooth function φ. To be more precise, we define the function $\mathfrak{y}$ by

$$\mathfrak{y}(x,y)=\begin{cases}0 & \text{if } g(x)<y\\ c(x) & \text{if } g(x)\ge y\end{cases},\ (x,y)\in(\mathbb{R}^+)^d\times\mathbb{R},$$

where c is a measurable map satisfying the following Assumption 6.1.

Assumption 6.1. The map $c:(\mathbb{R}^+)^d\mapsto\mathbb{R}^+$ is continuous with polynomial growth. Moreover, g is a viscosity subsolution of $\bar{r}\varphi-\mathscr{L}\varphi-c=0$ on $\{(t,x)\in[0,T)\times(\mathbb{R}^+)^d:V(t,x)=g(x)\}$.

[1] It should be clear that this assumption is only made for simplicity. Also note that a dividend rate could be added at no cost.

Before providing examples of such a function c, let us make some important observations.

Remark 6.2. First, $\{(t,x) \in [0,T) \times (\mathbb{R}^+)^d : V(t,x) = g(x)\} \subset \{x \in (\mathbb{R}^+)^d : g(x) > 0\}$ if $V > 0$ on $[0,T) \times (\mathbb{R}^+)^d$. This is typically the case in practice because g is non-negative and the probability that $g(X) > 0$ on $[0,T]$ is positive. In particular, if g is C^2 on $\{g > 0\}$ then one can choose $c = [rg - \mathscr{L}g]^+$ on $\{g > 0\}$.

Second, if g is convex, then it can not be touched from above by a C^2 function at a point at which it is not C^1, which implies that one can forget some singularity points in the verification of Assumption 6.1 above.

In Section 6.3, we shall suggest Monte-Carlo based numerical methods for the computation of V. One can then try to minimize the variance of the estimator over the choice of c. However, it seems natural to choose the function c so that g is actually a viscosity solution of $\bar{r}\varphi - \mathscr{L}\varphi - c = 0$ on $\{(t,x) \in [0,T) \times (\mathbb{R}^+)^d : V(t,x) = g(x)\}$. In the numerical study of Section 6.3, this choice coincides with the c with the minimal absolute value, which intuitively should correspond to the one minimizing the variance of the Monte-Carlo estimator. We leave the theoretical study of this variance minimization problem to future research.

Example 6.2. Let us consider the following examples in which $\hat{\sigma}$ is a constant matrix with i-th lines $\hat{\sigma}^i$. Fix $K, K_1, K_2 > 0$ with $K_1 < K_2$.

- For $d = 1$ and a put $g : x \in \mathbb{R}^+ \mapsto (K - x)^+$, the function c is given by the constant $\bar{r}K$. This is one of the cases treated in [2].
- For $d = 1$ and a strangle $g : x \in \mathbb{R}^+ \mapsto (K_1 - x)^+ + (x - K_2)^+$, the function c can be any continuous function equal to $\bar{r}K_1$ on $(0, K_1)$ and equal to $-\bar{r}K_2$ on (K_2, ∞), whenever $V > 0$.
- For $d = 2$ and a put on the arithmetic mean $g : x \in (\mathbb{R}^+)^2 \mapsto (K - \frac{1}{2}\sum_{i=1}^{2} x^i)^+$, we can take $c = \bar{r}K$.
- For $d = 2$ and a put on the geometric mean $g : x \in (\mathbb{R}^+)^2 \mapsto (K - \sqrt{x^1 x^2})^+$, c can be taken as
$$x \in (\mathbb{R}^+)^2 \mapsto (\bar{r}K - \frac{1}{8}(\|\hat{\sigma}^1\|^2 + \|\hat{\sigma}^2\|^2 - 2\langle \hat{\sigma}^1, \hat{\sigma}^2 \rangle)\sqrt{x^1 x^2})^+.$$

Since $\mathfrak{y}$ is discontinuous, we need to consider (6.5) in the sense of viscosity solutions for discontinuous operators. More precisely, let $\mathfrak{y}_*$ and $\mathfrak{y}^*$ denote the lower- and upper-semicontinuous envelopes of $\mathfrak{y}$. We say that a lower-semicontinuous function v is a viscosity supersolution of (6.5) if it is a viscosity supersolution of

$$\begin{aligned} \bar{r}\varphi - \mathscr{L}\varphi - \mathfrak{y}_*(\cdot, \varphi) = 0 \quad & \text{on } [0,T) \times (\mathbb{R}^+)^d \\ \varphi(T, \cdot) = g \quad & \text{on } (\mathbb{R}^+)^d. \end{aligned}$$

Similarly, we say that an upper-semicontinuous function v is a viscosity subsolution of (6.5) if it is a viscosity subsolution of

$$\begin{aligned} \bar{r}\varphi - \mathscr{L}\varphi - \mathfrak{y}^*(\cdot, \varphi) = 0 \quad & \text{on } [0,T) \times (\mathbb{R}^+)^d \\ \varphi(T, \cdot) = g \quad & \text{on } (\mathbb{R}^+)^d. \end{aligned}$$

We say that a continuous function is a viscosity solution of (6.5) if it is both a viscosity super- and subsolution.

Then, we have the following characterization of the American option price, which extends the result of [2] to our context. Recall Remark 6.1.

Theorem 6.3. *Let c be as in Assumption 6.1. Then, V is a viscosity solution of (6.5). It has a polynomial growth.*

Proof. This proof has the same line of arguments as in Section 5 of [2]. We begin with proving that V is a supersolution of Equation (6.5). First note that $V \geq g$, so that $\mathfrak{y}_*(\cdot, V) = 0$[2]. Hence, the supersolution property is equivalent to being a supersolution of

$$\bar{r}\varphi - \mathscr{L}\varphi = 0 \text{ on } [0,T) \times (\mathbb{R}^+)^d \text{ and } \varphi(T,\cdot) = g \text{ on } (\mathbb{R}^+)^d,$$

which is standard by the definition of supersolution, the application of Equation (6.4) and Itô's formula. In which we apply Itô's formula to the function $e^{-\bar{r}(\theta-t)}\varphi(\theta, X(\theta))$, where θ is the exit time for the process $(s, X(s))$ from a strictly positive radius ball centred at (t,x) and we take expectations on the both sides. The required inequality follows accordingly.

For the subsolution part, we fix $(t,x) \in [0,T] \times (\mathbb{R}^+)^d$ and a smooth function φ such that (t,x) achieves a maximum on $[0,T] \times (0,\infty)^d$ of $V - \varphi$ and $(V-\varphi)(t,x) = 0$. If $t = T$, then the required result holds by definition. We now assume that $t < T$. If (t,x) belongs to the open set $C := \{V > g\}$, recall Remark 6.1, then one can find a $[t,T]$-valued stopping time τ such that $(\cdot \wedge \tau, X^{t,x}_{\cdot\wedge\tau}) \in C$, and it follows from the dynamic programming principle, see e.g. [12], that

$$\varphi(t,x) \leq \mathbb{E}\left[e^{-\bar{r}(\tau_\epsilon - t)}\varphi(\tau_\epsilon, X_{\tau_\epsilon})\right]$$

in which $\tau_\epsilon := \tau \wedge (t+\epsilon)$ for $\epsilon > 0$. Then, applying Itô's formula to the right hand side leads to

$$0 \geq \bar{r}\varphi(t,x) - \mathscr{L}\varphi(t,x) = \bar{r}\varphi(t,x) - \mathscr{L}\varphi(t,x) - \mathfrak{y}^*(x, \varphi(t,x)).$$

Let us now assume that $(t,x) \in S := \{V = g\}$. In particular, $\varphi(t,x) = V(t,x) = g(x)$ and therefore $\mathfrak{y}^*(x, \varphi(t,x)) = \mathfrak{y}^*(x, V(t,x)) = c(x)$. Since $V \geq g$, (t,x) is also a maximum of $g - \varphi$ and φ satisfies

$$0 \geq \bar{r}\varphi(t,x) - \mathscr{L}\varphi(t,x) - c(x) = \bar{r}\varphi(t,x) - \mathscr{L}\varphi(t,x) - \mathfrak{y}^*(x, \varphi(t,x)),$$

by Assumption 6.1. □

This viscosity solution property can be complemented with a comparison principle as in [2]. Combined with Theorem 6.3, it shows that V is the unique viscosity solution of (6.5) with polynomial growth.

Proposition 6.4. *Let the conditions of Theorem 6.3 hold. Let v and w be respectively a super- and a subsolution of (6.5), with polynomial growth. Then, $v \geq w$ on $[0,T] \times (\mathbb{R}^+)^d$.*

[2] Note that this is an important consequence of using $\mathfrak{y}_*$ instead of $\mathfrak{y}$.

Proof. The proof is a simple adaptation of the arguments of [2] and [5, Proof of Theorem 4.5]. Therefore, we give a brief outline of the proof here while referring interested readers to the above publications.

As usual, one can assume without loss of generality that $\bar{r} > 0$, upon replacing v by $(t,x) \mapsto e^{-\rho t} v(t,x)$ and w by $(t,x) \mapsto e^{-\rho t} w(t,x)$ for some $\rho > |r|$. Fix $p \geq 1$ and $C > 0$ such that $|v(t,x)| + |w(t,x)| \leq C(1 + \|x\|^p)$ for all $(t,x) \in [0,T] \times (\mathbb{R}^+)^d$. Set $\psi(t,x) := e^{-\kappa t}(1 + \|x\|^{2p})$ for $(t,x) \in [0,T] \times (\mathbb{R}^+)^d$, for some κ large enough so that ψ is a supersolution of $-\mathscr{L}\varphi = 0$ on $[0,T) \times (\mathbb{R}^+)^d$, which is possible since $\hat{\sigma}$ is bounded. Set

$$\phi_n^\epsilon(t,x,y) := w(t,y) - v(t,x) - n\|x-y\|^{2p} - \lambda\psi(t,y) - \frac{\epsilon}{\prod_{i=1}^d x^i} - \frac{\epsilon}{\prod_{i=1}^d y^i}$$

for $n \geq 1$, $\epsilon > 0$, $(t,x,y) \in [0,T] \times (\mathbb{R}^+)^{2d}$, and a given $\lambda > 0$. Assume that $\sup_{[0,T]\times(\mathbb{R}^+)^d}(w - v) > 0$. Then one can find $\epsilon_\circ, \lambda > 0$ and $\delta > 0$ such that

$$\sup_{[0,T]\times(\mathbb{R}^+)^{2d}} \phi_n^\epsilon \geq \delta, \text{ for } \epsilon \in (0,\epsilon_\circ) \text{ and } n \geq 1. \tag{6.6}$$

Clearly, ϕ_n^ϵ admits a maximum point $(t_n^\epsilon, x_n^\epsilon, y_n^\epsilon)$ on $[0,T] \times (0,\infty)^{2d}$ because of the semi-continuity of w, v and ψ and the fact that ϕ_n^ϵ is $-\infty$ at the origin and 0 at infinity. Moreover, it follows from standard arguments that $(t_n^\epsilon, x_n^\epsilon, y_n^\epsilon)$ converges to some $(t_n, x_n, y_n) \in [0,T] \times (\mathbb{R}^+)^d$ as $\epsilon \to 0$, possibly along a subsequence, and that

$$\lim_{\epsilon\to 0}\Big(\frac{\epsilon}{\prod_{i=1}^d (x_n^\epsilon)^i} + \frac{\epsilon}{\prod_{i=1}^d (y_n^\epsilon)^i}\Big) = 0 \ , \ \lim_{n\to\infty} n\|x_n - y_n\|^{2p} = 0, \tag{6.7}$$

$$\lim_{\epsilon\to 0}(w(t_n^\epsilon, y_n^\epsilon), v(t_n^\epsilon, x_n^\epsilon)) = (w(t_n, y_n), v(t_n, x_n)) \tag{6.8}$$

$$\lim_{n\to\infty} y_n = \hat{y}, \text{ for some } \hat{y} \in (\mathbb{R}^+)^d, \tag{6.9}$$

possibly along subsequences, see e.g. [5, Proof of Theorem 4.5] and [9]. Combining Ishii's Lemma, see e.g. [9], with the super- and subsolution properties of v, ψ and w, we obtain

$$\begin{aligned} 0 \geq\ & \bar{r}(w(t_n^\epsilon, y_n^\epsilon) - v(t_n^\epsilon, x_n^\epsilon)) - \mathfrak{y}^*(y_n^\epsilon, w(t_n^\epsilon, y_n^\epsilon)) + \mathfrak{y}_*(x_n^\epsilon, v(t_n^\epsilon, x_n^\epsilon)) \\ & - O(n\|x_n^\epsilon - y_n^\epsilon\|^{2p}) - \eta_\epsilon^n \end{aligned}$$

in which, thanks to the left-hand side of (6.7), the last term $\eta_\epsilon^n \to 0$ as $\epsilon \to 0$, for all $n \geq 1$. By the right-hand side of (6.7), the discussion just above it, and (6.8), sending $\epsilon \to 0$ and then $n \to \infty$ leads to

$$0 \geq \limsup_{n\to\infty}\big\{\bar{r}(w(t_n,y_n) - v(t_n,x_n)) - \mathfrak{y}^*(y_n, w(t_n,y_n)) + \mathfrak{y}_*(x_n, v(t_n,x_n))\big\}$$

and therefore

$$\liminf_{n\to\infty}\{\mathfrak{y}^*(y_n, w(t_n,y_n)) - \mathfrak{y}_*(x_n, v(t_n,x_n))\} \geq r\delta$$

by (6.6). Recall that c is non-negative and that $w(t_n,y_n) - v(t_n,x_n) \geq \delta$ by (6.6). If, along a subsequence, $g(x_n) > v(t_n,x_n)$ for all n, then $\mathfrak{y}^*(y_n, w(t_n,y_n)) - \mathfrak{y}_*(x_n, v(t_n,x_n)) \leq c(y_n) - c(x_n)$ for all n, leading to a contradiction since $c(x_n) - c(y_n) \to 0$ as $n \to \infty$ (recall (6.7) and (6.9)) and $r > 0$. If, along a subsequence, $g(x_n) \leq v(t_n,x_n)$ for all n, then $g(y_n) \leq v(t_n,x_n) + \delta/2 \leq w(t_n,y_n) - \delta/2$ for all n large enough and the above liminf is also non-positive. A contradiction too. □

6.3. MONTE-CARLO ESTIMATION

In this section, we use the connection between PDEs and BSDEs to derives pricing schemes for American options. This demonstrates another important application of BSDEs.

The solution of (6.5) is formally related to the solution $(Y,Z)\in\mathbf{S}_2\times\mathbf{L}_2$ of the backward stochastic differential equation

$$Y=e^{-\bar r T}g(X_T)+\int_\cdot^T e^{-\bar r s}\mathfrak{y}(X_s,e^{\bar r s}Y_s)\,ds-\int_\cdot^T Z_s\,dW_s$$

by $e^{-\bar r\cdot}V(\cdot,X)=Y$. In the above, $\mathbf{S}_2$ denotes the space of adapted processes ξ such that $\mathbb{E}[\sup_{[0,T]}\|\xi\|^2]<\infty$ and $\mathbf{L}_2$ denotes the space of predictable processes ξ such that $\mathbb{E}[\int_0^T\|\xi_t\|^2dt]<\infty$.

Remark 6.3. Note that, if (Y,Z) satisfies the above BSDE, then

$$Y_0=\mathbb{E}\left[e^{-\bar r T}g(X_T)+\int_0^T e^{-\bar r s}\mathfrak{y}(X_s,e^{\bar r s}Y_s)\,ds\right].$$

In the case where $c=\bar r g-\mathscr{L}g$, on $\{(t,x)\in[0,T)\times(\mathbb{R}^+)^d : V(t,x)=g(x)\}$, this corresponds to the early-exercise premium formula. Recall Assumption 6.1 and see [2, Section 6].

In practice, the above BSDE is not well-posed because $\mathfrak{y}$ is not continuous. However, it can be smoothed out for the purpose of numerical approximations. In the following, we write $\mathbb{E}_s[\cdot]$ to denote the expectation given $\mathfrak{F}_s$, $s\le T$.

Proposition 6.5. *Let the condition of Theorem 6.3 hold. Let $(\mathfrak{y}_n)_{n\ge1}$ be a sequence of continuous functions on $(\mathbb{R}^+)^d\times\mathbb{R}$ that are Lipschitz in their last component[3]. Assume that $(\mathfrak{y}_n)_{n\ge1}$ is uniformly bounded by a function with polynomial growth in its first component and linear growth in its last component. Assume further that*

$$\limsup_{\substack{n\to\infty\\(x',y')\to(x,y)}}\mathfrak{y}_n(x',y')\le\mathfrak{y}^*(x,y)\quad\text{and}\quad\liminf_{\substack{n\to\infty\\(x',y')\to(x,y)}}\mathfrak{y}_n(x',y')\ge\mathfrak{y}_*(x,y)\tag{6.10}$$

for all $(x,y)\in(\mathbb{R}^+)^d\times\mathbb{R}$. For $(t,x)\in[0,T]\times(\mathbb{R}^+)^d$, let $(Y^{t,x,n})_{n\ge1}$ be such that

$$Y_s^{t,x,n}=\mathbb{E}_s\left[e^{-\bar r T}g(X_T^{t,x})+\int_s^T e^{-\bar r u}\mathfrak{y}_n(X_u^{t,x},e^{\bar r u}Y_u^{t,x,n})\,du\right],$$

for $s\in[t,T]$, and set $V_n(t,x):=e^{\bar r t}Y_t^{t,x,n}$. Then, $(V_n)_{n\ge1}$ converges pointwise to V as $n\to\infty$.

Proof. Each BSDE associated to $\mathfrak{y}_n$ admits a unique solution $(Y^{t,x,n},Z^{t,x,n})\in\mathbf{S}_2\times\mathbf{L}_2$, and it is standard to show that V_n is a continuous viscosity solution of

$$\bar r\varphi-\mathscr{L}\varphi-\mathfrak{y}_n(\cdot,\varphi)=0\text{ on }[0,T)\times(\mathbb{R}^+)^d\text{ and }\varphi(T,\cdot)=g\text{ on }(\mathbb{R}^+)^d.$$

Moreover, $(V_n)_{n\ge1}$ has (uniformly) polynomial growth, thanks to the uniform polynomial growth assumption on $(\mathfrak{y}_n)_{n\ge1}$. See e.g. [13]. By stability and (6.10), see e.g. [14], it follows that the relaxed limsup V^* and liminf V_* of $(V_n)_{n\ge1}$ are respectively sub- and super-solutions of (6.5). By Proposition 6.4, $V^*\le V\le V_*$ and therefore equality holds among the three functions. □

[3] See below for examples.

Therefore, up to a smoothing procedure, we are back to essentially solving a BSDE. In the next two subsections, we propose two approaches. The first one consists in smoothing $\mathfrak{y}$ into a a smooth function $\mathfrak{y}_n$ to which we apply the local polynomial approximation procedure of [3]. This allows us to use a pure forward Monte-Carlo method for the estimation of V_n, based on branching processes. In the second approach, we only add an independent noise in the definition of $\mathfrak{y}$, which also has the effect of smoothing it out, and then use a very simple version of the algorithm in [3]. As our numerical experiments show, the first approach is quite unstable while the second one is very efficient.

6.3.1. LOCAL POLYNOMIAL APPROXIMATION AND BRANCHING PROCESSES

Given Proposition 6.5, it is tempting to estimate the American option price by using a recently developed Monte-Carlo method for BSDEs based solely on forward simulations, see [15] and the references therein. Here, we propose to use the forward approach suggested by [3], which is based on the use of branching processes coupled (in theory) with Picard iterations.

The first step consists in approximating the Heaviside function $H : z \mapsto \mathbf{1}_{\{z\geq 0\}}$ by a sequence of Lipschitz functions $(H_n)_{n\geq 1}$ and to define $\mathfrak{y}_n$ by

$$\mathfrak{y}_n : (x, y) \mapsto c(x) H_n(g(x) - y).$$

Then, $\mathfrak{y}_n$ is approximated by a map $(x, y) \mapsto \bar{\mathfrak{y}}_n(x, y, y)$ of local polynomial form:

$$\bar{\mathfrak{y}}_n : (x, y, y') \to \sum_{j=1}^{j_0} \sum_{l=0}^{l_0} a_{j,l}(x) y^l \phi_j(y')$$

where $(a_{j,l}, \phi_j)_{l\leq l_0, j\leq j_0}$ is a family of continuous and bounded maps satisfying

$$|a_{j,l}| \leq C_{l_0},\ |\phi_j(y_1') - \phi_j(y_2')| \leq L_\phi |y_1' - y_2'| \text{ and } |\phi_j| \leq 1,$$

for all $y_1', y_2' \in \mathbb{R}$, $j \leq j_0$ and $l \leq l_0$, for some constants C_{l_0}, $L_\phi \geq 0$. The elements of $(a_{j,l}(x))_{l\leq l_0}$ should be interpreted as the coefficients of a polynomial approximation of $\mathfrak{y}_n$ on a subset $\mathscr{A}_j$, in which $(\mathscr{A}_j)_{j\leq j_\circ}$ forms a partition of $\mathbb{R}$ and the ϕ_j's as smoothing kernels that allow one to pass in a Lipschitz way from one part of the partition to another one, see [3].

Then, one can consider the sequence of BSDEs

$$\begin{aligned}\bar{Y}_s^{t,x,n,k+1} =& \mathbb{E}_s[e^{-\bar{r}T} g(X_T^{t,x})] \\ &+ \mathbb{E}\left[\int_s^T e^{-\bar{r}u} \bar{\mathfrak{y}}_n(X_u^{t,x}, e^{\bar{r}u} \bar{Y}_u^{t,x,n,k+1}, e^{\bar{r}u} \bar{Y}_u^{t,x,n,k}) du\right],\ k \geq 1,\end{aligned}$$

with $\bar{Y}^{t,x,n,1}$ given as an initial prior (e.g. $e^{\bar{r}\cdot} g(X^{t,x})$). Given $\bar{Y}^{t,x,n,k}$, $\bar{Y}^{t,x,n,k+1}$ solves a BSDE with polynomial driver that can be estimated by using branching processes as in the Feynman-Kac representation of the Kolmogorov-Petrovskii-Piskunov equation, see [10, 11]. We refer to [3] for more details.

In practice, we use the Method A of [3, Section 3]. We perform a numerical experiment in dimension 1, with a time horizon of one year, and a risk-free interest rate set at

6%. We consider the Black-Scholes model with one single stock whose volatility is 40%. We price a put option whose strike is $K := 40$. At the money, the American option price is around 5.30, while the European option is worth 5.05. In view of Example 6.2, we take $c = \bar{r}K$[4].

We first smooth the driver with a centered Gaussian density with variance κ^{-2}, so as to replace it by $0.5\bar{r}Ke^{-\bar{r}t}\mathrm{erfc}(\kappa * (y - e^{-\bar{r}t}g(x)))$ with $\kappa = 10$, where erfc is the complementary error function. See Figure 6.1. Then, we apply a quadratic spline approximation. In actual computation, as it is impossible to apply spline approximation on the whole half real line, we limited the domain of y for the driver function to $[0, 40(1-e^{-0.06})]$. We partition this bounded domain into 20 intervals with equal-distant points and define a piecewise polynomial on this domain by assigning a quadric polynomial to each interval. Finally, we match the values and derivatives of our piecewise polynomial at each point of the grid to the original function (except at the right-end where the derivative is assumed to be zero). The truncation of domain will not alter the computational result as our limited domain includes the maximum payoff for the put option. The resulting approximation is indistinguishable from the original function displayed on Figure 6.1.

We also partition $[0, T]$ in 10 periods. As for the grid in the x-component, we use a 25-point uniform space-grid on the interval $[e^{-20}, 80]$.

We estimate the early exercise value by first using 1000 Monte-Carlo paths. As can be seen on Figure 6.2, the results are not good and this does not improve much with a higher number of simulations. The algorithm turns out to be quite unstable and not accurate. It remains pretty unstable even for a large number of simulated paths. This is not so surprising. Indeed, as explained in [3], their approach is dedicated to situations where the driver functions are rather smooth, so that the local polynomial's coefficients $(a_{j,l})_{j,l}$ are small, and the supports of the ϕ_j's are large and do not intersect too much. Since we are approximating the Heaviside function, none of these requirements is met.

6.3.2. DRIVER RANDOMIZATION

In this second approach, we enlarge the state space so as to introduce an independent integrable random variable ϵ with density f such that $z \mapsto (1 + |z|)f'(z)$ is integrable. We assume that the interior of the support of f is of the form (m_ϵ, M_ϵ) with $-\infty \le m_\epsilon < M_\epsilon \le \infty$. Then, we define the sequence of random maps

$$\tilde{\mathfrak{y}}_n(x, y) := c(x)\mathbf{1}_{\{g(x)+\frac{\epsilon}{n} \ge y\}}$$

as well as

$$\begin{aligned}\mathfrak{y}_n(x, y) :=& c(x)n\left\{[g(x) + M_\epsilon/n - y]^+ f(M_\epsilon) - [g(x) + m_\epsilon/n - y]^+ f(m_\epsilon)\right\} \\ &- c(x)n\int [g(x) + z/n - y]^+ f'(z)dz\end{aligned}$$

so that

$$\mathfrak{y}_n(x, y) = \mathbb{E}[\tilde{\mathfrak{y}}_n(x, y)]$$

[4] Note that, for this payoff, the constant $\bar{r}K$ is the function with the smallest absolute value among the functions c satisfying the requirements of Assumption 6.1.

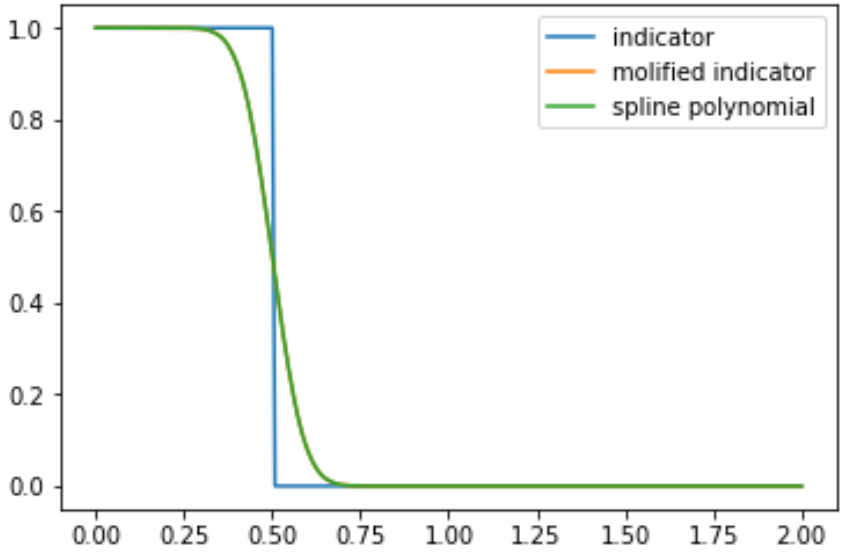


Figure 6.1: Approximation of $y \mapsto \mathbf{1}_{\{y \leq 0.5\}}$.

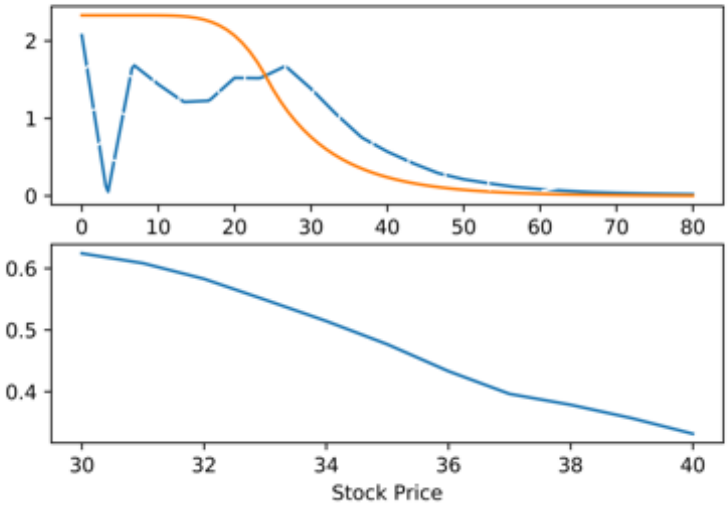


Figure 6.2: Branching with local polynomial approximation. Upper graph: Early exercise premium (plain line obtained by a pde solver, dashed line estimated). Lower graph: Error on the early exercise premium estimation.

for $n \geq 1$. If c is non-negative, continuous and has polynomial growth, then the sequence $(\mathfrak{y}_n)_{n\geq 1}$ matches the requirements of Proposition 6.5.

We now let τ be an independent exponentially distributed random variable with density ρ and cumulative distribution $1-\bar{F}$. Then, $Y^{t,x,n}$ defined as in Proposition 6.5 satisfies

$$Y_s^{t,x,n} = \mathbb{E}_s\left[e^{-rT}\frac{g(X_T^{t,x})}{\bar{F}(T-t)}\mathbf{1}_{\{T-t\leq\tau\}} + \mathbf{1}_{\{T-t>\tau\}}\frac{e^{-r\tau}\tilde{\mathfrak{y}}_n(X_{t+\tau}^{t,x}, e^{r\tau}Y_{t+\tau}^{t,x,n})}{\rho(\tau)}\right].$$

This can be viewed as a branching based representation in which particles die at an exponential time. If a particle dies before T, we give it the (random) mark $\tilde{\mathfrak{y}}_n(X_{t+\tau}^{t,x}, e^{r\tau}Y_{t+\tau}^{t,x,n})$. In terms of the representation of Section 6.3.1, this corresponds to $j_0 = 1$, $l_0 = 0$, to replacing $a_{1,0}(x)\phi_1(y')$ by $\tilde{\mathfrak{y}}_n(x,y')$, and to not using a Picard iteration scheme.

On a finite time grid $\pi \subset [0,T]$ containing $\{0,T\}$, it can be approximated by the sequence v_n^π defined by $v_n^\pi(T,\cdot) = g$ and

$$\begin{aligned} v_n^\pi(t,x) =& \mathbb{E}\left[e^{-rT}\frac{g(X_T^{t,x})}{\bar{F}(T-t)}\mathbf{1}_{\{T-t\leq\tau\}}\right] \\ &+ \mathbb{E}\left[\mathbf{1}_{\{T-t>\tau\}}\frac{e^{-r\tau}\tilde{\mathfrak{y}}_n(X_{\phi_{t+\tau}^\pi}^{t,x}, e^{r\tau}v_n^\pi(\phi_{t+\tau}^\pi, X_{\phi_{t+\tau}^\pi}^{t,x}))}{\rho(\tau)}\right], \end{aligned} \tag{6.11}$$

where $\phi_s^\pi := \inf\{s' \geq s : s' \in \pi\}$ for $s \leq T$. Showing that $v_n^\pi(\phi_t^\pi, x)$ converges point-wise to $Y_t^{t,x,n}$ as the modulus of π vanishes can be done by working along the lines of [16, Section 4.3] or [17]. In view of Proposition 6.5, v_n^π converges point-wise to V as $|\pi| \to 0$ and $n \to \infty$. A similar analysis could be performed when considering a grid in space, which will be necessary in practice.

Then, (6.11) provides a natural backward algorithm: given a space-time grid $\Pi := (t_i, x_j)_{i,j}$, (6.11) can be used to compute $v_n^\pi(t_i, x_j)$ given the already computed values of v_n^π at the later times in the grid, by replacing the expectation by a Monte-Carlo counterpart.

NUMERICAL EXPERIMENTS

Let us now consider a put option pricing problem within the Black-Scholes model as in the previous section. The interest rate is 6%, the volatility is 20% and the strike is 25. The partition π of $[0,T]$ is uniform with 100 time steps. However, we update v_n^π only every 10 time steps (and consider that it is constant in time in between). The fine grid π is therefore only used to approximate $X_\tau^{t,x}$ by $X_{\phi_\tau^\pi}^{t,x}$ accurately. We use a 40-points equidistant space-grid on the interval $[5,50]$. The random variable ϵ/n is exponentially distributed, with mean equal to 10^{-100}, while τ has mean 0.6.

In Figures 6.3, 6.4 and 6.5, we provide the estimated prices, the estimated early exercise premium as well as the corresponding relative errors with different numbers of sample paths. The statistics are based on 50 independent trials. The reference values are computed with an implicit scheme for the associated PDE, with regular grids of 500 points in space and 1.000 points in time (we also provide the European option price in the top-left graph, for comparison). The relative errors are capped to 10% or 40% for

ease of readability. These graphs show that the numerical method is very efficient. The relative error for a stock price higher that 30/35 are not significant since it corresponds to option prices very close to 0. For 10.000 simulated paths, it takes 12 secondes for one estimation of the whole price curve with an R code running on a Macbook 2014, 2.5 GHz Intel Core i7, with 4 physical cores.

The second example we considered is a strangle with strikes 25 and 27, see Example 6.2. The results obtained with 50000 sample paths are displayed in Figures 6.6. The algorithm also performs well in this case with accurate result.

Finally, we state that we do not use any variance reduction technique in these experiments.

6.4. CONCLUSION

In this chapter, we used the American option pricing problem as a case study for the application of the PDE and BSDE's connection. We extended the semilinear PDE formulation for valuating American options in [2] with the Black-Scholes model and vanilla option to more general models and payoff functions. In order to solve the more complex PDEs from this generalization, we link the derived PDE with a BSDE and tested two Monte-Carlo schemes to solve the pricing problem. While the driver randomization scheme worked well, the branching process with local polynomial scheme was proven to be unstable and failed to converge. Thus it shows the importance of picking an appropriate numerical scheme.

REFERENCES

[1] B. Bouchard, K. W. Chau, A. Manai, and A. Sid-Ali, *Monte-carlo methods for the pricing of american options: a semilinear bsde point of view,* ESAIM: Proceedings and Surveys **65**, 294 (2019).

[2] F. E. Benth, K. H. Karlsen, and K. Reikvam, *A semilinear black and scholes partial differential equation for valuing american options,* Finance and Stochastics **7**, 277 (2003).

[3] B. Bouchard, X. Tan, X. Warin, and Y. Zou, *Numerical approximation of BSDEs using local polynomial drivers and branching processes,* Monte Carlo Methods and Applications (to appear).

[4] F. Black and M. Scholes, *The pricing of options and corporate liabilities,* Journal of Political Economy **81**, 637 (1973).

[5] B. Bouchard and J.-F. Chassagneux, *Fundamentals and advanced techniques in derivatives hedging* (Springer, 2016).

[6] H. McKean Jr, *A free boundary problem for the heat equation arising from a problem in mathematical economics,* Industrial Management Review **6**, 32 (1965).

[7] A. Bensoussan and J.-L. Lions, *Applications of variational inequalities in stochastic control*, Vol. 12 (Elsevier, 2011).

[8] F. E. Benth, K. H. Karlsen, and K. Reikvam, *A semilinear Black and Scholes partial differential equation for valuing american options: approximate solutions and convergence,* Interfaces and Free Boundaries **6**, 379 (2004).

[9] M. G. Crandall, H. Ishii, and P.-L. Lions, *User's guide to viscosity solutions of second order partial differential equations,* Bulletin of the American Mathematical Society **27**, 1 (1992).

[10] P. Henry-Labordère, *Cutting CVA's complexity,* Risk **25**, 67 (2012).

[11] P. Henry-Labordere, N. Oudjane, X. Tan, N. Touzi, and X. Warin, *Branching diffusion representation of semilinear pdes and monte carlo approximation,* arXiv preprint arXiv:1603.01727 (2016).

[12] B. Bouchard and N. Touzi, *Weak dynamic programming principle for viscosity solutions,* SIAM Journal on Control and Optimization **49**, 948 (2011).

[13] É. Pardoux, *Backward stochastic differential equations and viscosity solutions of systems of semilinear parabolic and elliptic PDEs of second order,* Stochastic Analysis and Related Topics VI , 79 (1998).

[14] G. Barles, *Solutions de viscosité des équations de Hamilton-Jacobi* (Springer, 1994).

[15] B. Bouchard and X. Warin, *Monte-Carlo valuation of American options: Facts and new algorithms to improve existing methods,* in *Numerical Methods in Finance: Bordeaux, June 2010*, edited by R. A. Carmona, P. Del Moral, P. Hu, and N. Oudjane (Springer Berlin Heidelberg, Berlin, Heidelberg, 2012) pp. 215–255.

[16] N. Baradel, B. Bouchard, and N. Dang, *Optimal control under uncertainty and bayesian parameters adjustments,* SIAM Journal on Control and Optimization **56**, 1038 (2018).

[17] W. H. Fleming and P. E. Souganidis, *On the existence of value functions of two-player, zero-sum stochastic differential games,* Indiana University Mathematics Journal **38**, 293 (1989).

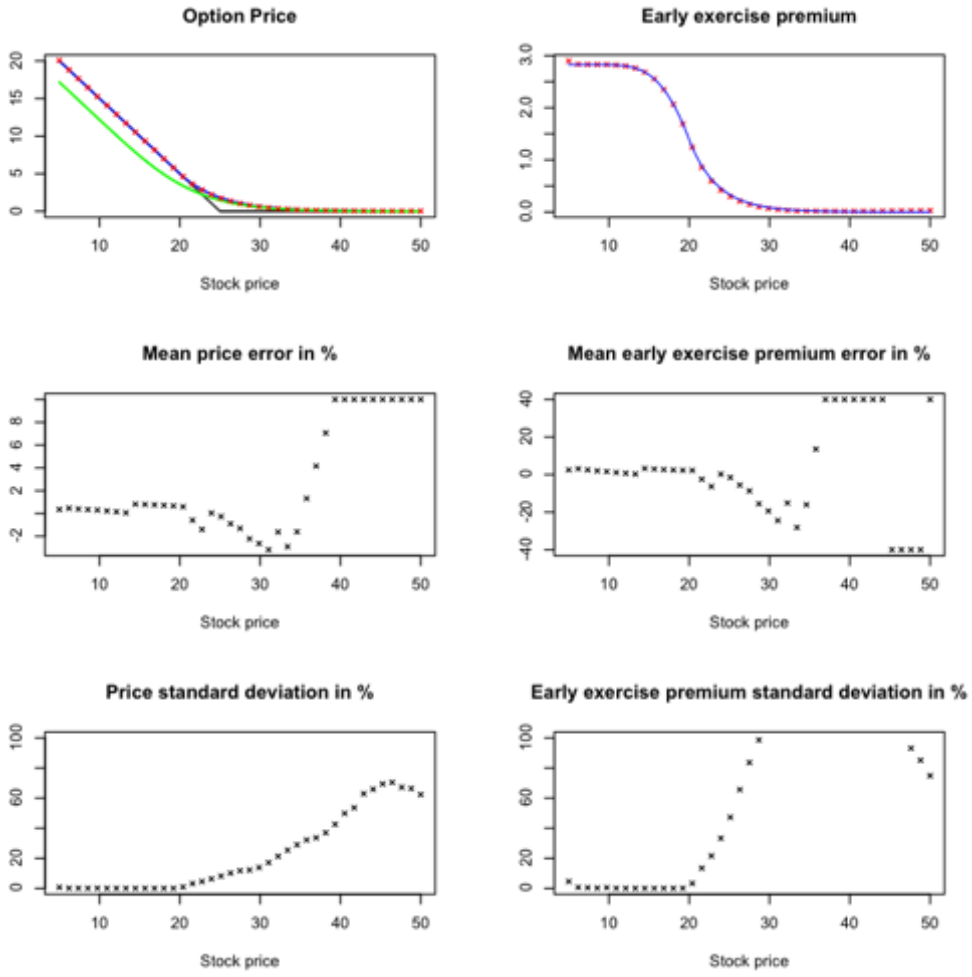


Figure 6.3: Branching with indicator driver. Put option, 1000 sample paths. Plain lines=true values, crosses=estimations.

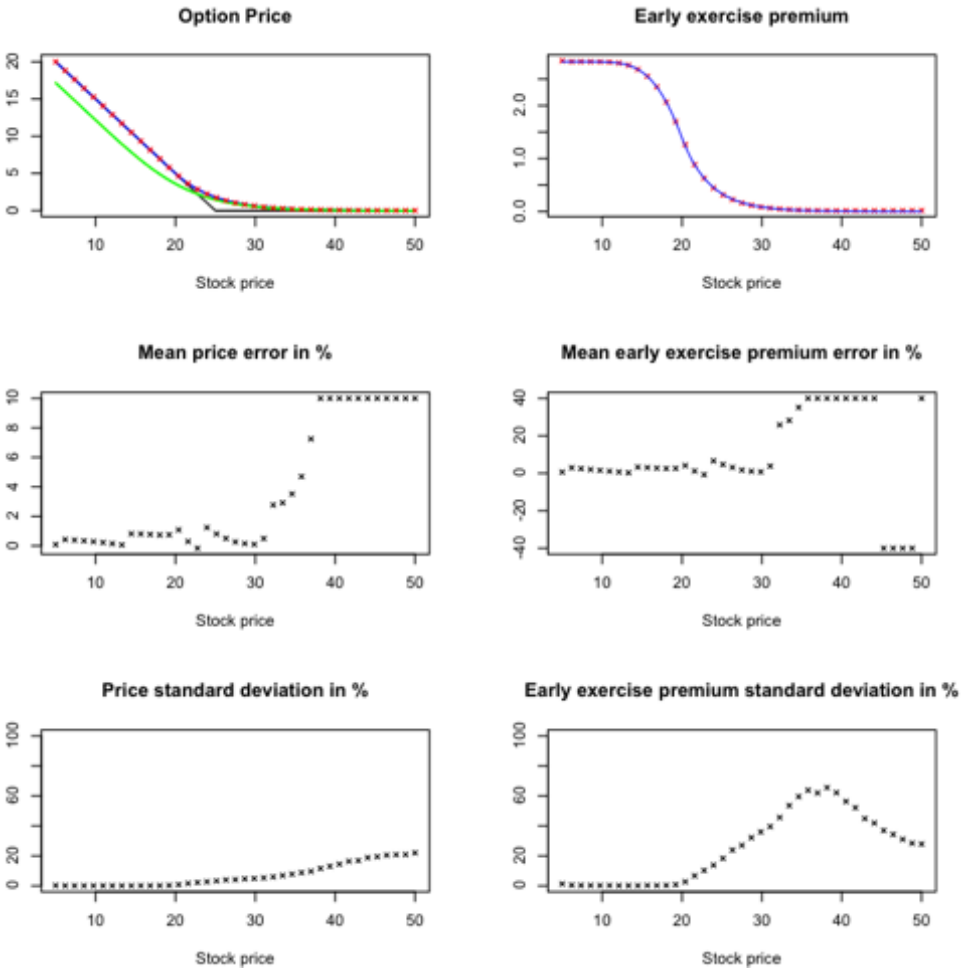


Figure 6.4: Branching with indicator driver. Put option, 10000 sample paths. Plain lines=true values, crosses=estimations.

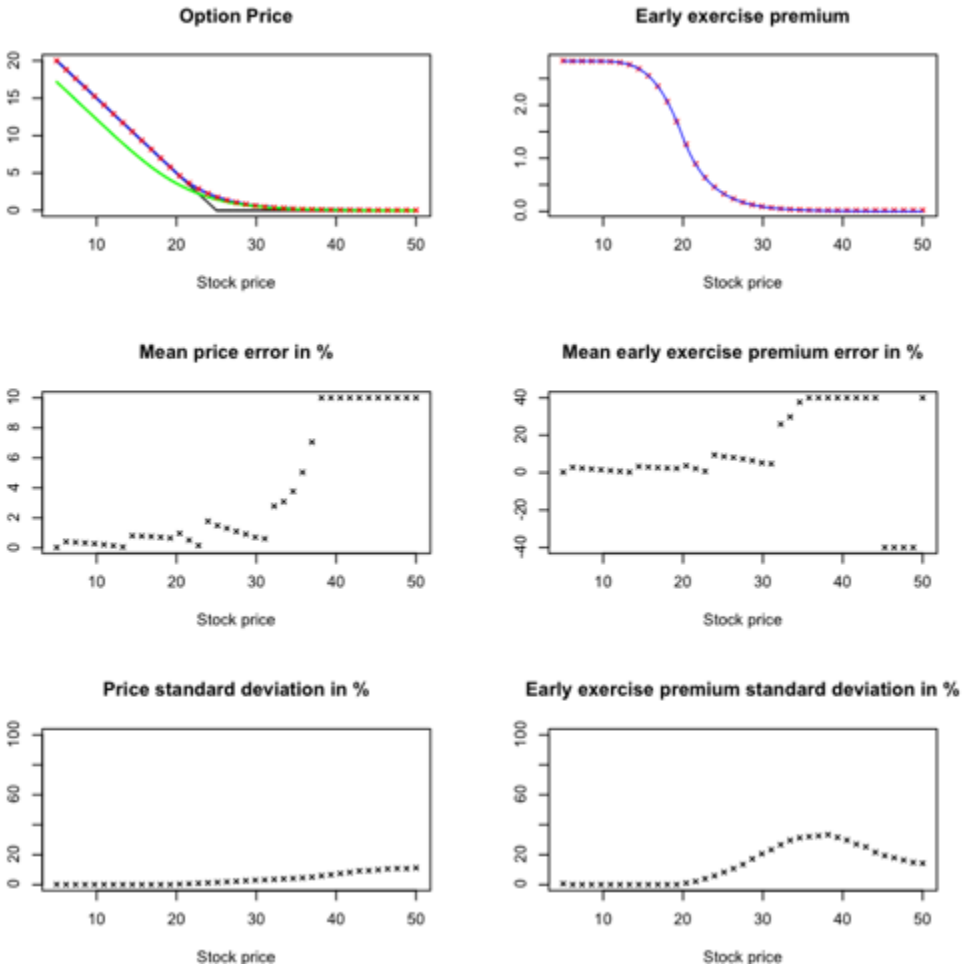


Figure 6.5: Branching with indicator driver. Put option, 50000 sample paths. Plain lines=true values, crosses=estimations.

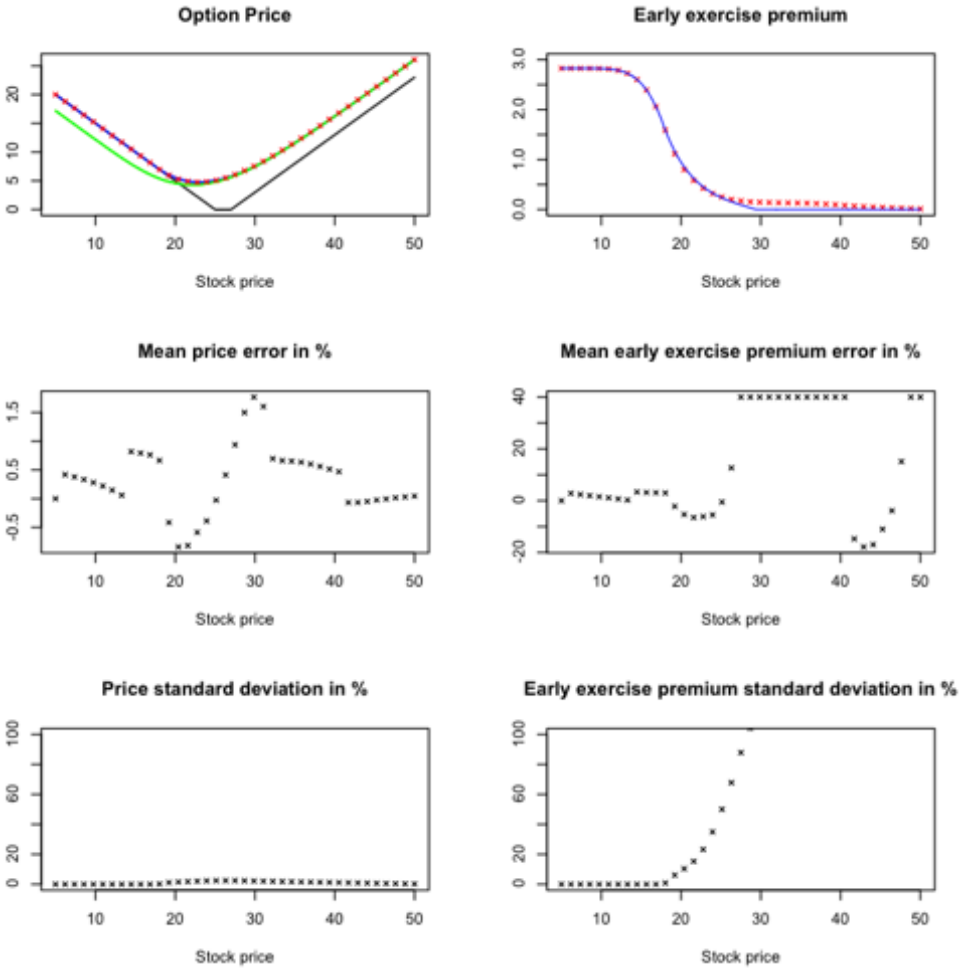


Figure 6.6: Branching with indicator driver. Strangle option, 50000 sample paths. Plain lines=true values, crosses=estimations.

7

CONCLUSION

In this thesis, we studied essentially three different numerical schemes to approximate BSDE solutions. This work has covered the derivation and error bound of algorithms, the implementation of numerical schemes in modern software infrastructure and the application of BSDEs. But the main focus of this work remains understanding the core properties of various numerical algorithms. Here is a brief review of our conclusions from researching these algorithms.

7.1. GENERAL CONCLUSION

7.1.1. FOURIER EXPANSION METHOD

In order to overcome the difficulty of calculating cosine coefficients in the original COS method, a localized variant of the Fourier expansion method was introduced in Chapter 2. We have shown that when the target function is continuous and the integration range is sufficiently large, even if the localized SWIFT formula induces an extra error term in the error bound, this error term decays quickly with respect to the truncation number N. The localized/quick SWIFT method gave a similar approximation result to the full Fourier series version but it could greatly reduce the required computational effort.

Combining the quick SWIFT formula with the classical time-discretization of BSDEs, we proposed a new probabilistic method for solving FBSDEs numerically. The numerical result was satisfactory for one-dimensional BSDEs. The error of applying the SWIFT method is relatively minor compared to the discretization error for the stochastic process. One only has to note the possible discontinuities within the terminal conditions of BSDEs and adjust the numerical scheme accordingly (the mixed SWIFT method).

We generalized the philosophy of localized Fourier expansion methods to higher-dimensional space in Chapter 3. This was achieved by applying results from the lattice literature in [1]. We also demonstrated the similarity between the periodic wavelet construction and the reproducing kernel construction.

7.1.2. STOCHASTIC GRID BUNDLING METHOD

Built on top of previous works on SGBM, we have developed a new error formulation for it based on classic non-parametric regression. This has not been done in the literature as far as we are aware of. We have explored the major difficulty in implementing SGBM, where the classic truncation technique cannot be used and a new sample selection procedure had to be introduced. We had to deal with the troubles when establishing the error bound for SGBM, where we have to properly define the equal partition technique. We have shown that an intuitive numerical scheme does not always lead to an easy numerical analysis.

We also combined SGBM with GPU computing, which had greatly improved the efficiency of our test code. Our numerical tests suggested that SGBM for BSDEs can be applied to higher-dimensional problems with appropriate basis and bundling setting.

7.1.3. PDE AND BRANCHING METHOD

For the final part of this thesis, the Feynman-Kac type representation was extended to a type of non-linear parabolic PDEs and we worked on approximating the resulting BSDEs. Although the local polynomial approximation based branching method seemed to be a suitable advanced numerical scheme for this problem, this scheme was proven to be unstable and failed to converge as the driver of the BSDEs fails to fulfil some basic conditions. On the other hand, the redesigned numerical scheme with drivers randomization worked well. Thus, it is important to apply appropriate numerical schemes to different kinds of BSDEs.

7

7.2. OUTLOOK

We have achieved many results in thesis, especially showing that BSDEs can be solved effectively and therefore it is possible to include BSDEs in industrial applications. Nevertheless, on top of our results in this thesis, many interesting aspects can be further explored.

7.2.1. LOCALIZED FOURIER EXPANSION METHOD

The main remaining issue with the higher-dimensional expectation approximation is that the computation effort required on the scaling function/reproducing kernel increases with the problems' dimension. However, there has been research on the construction of an index set for truncated reproducing kernel such that it does not suffer from the curse of dimensionality. Incorporating these results from Fourier analysis would provide a workable algorithm in higher dimensions.

We can also combine the cosine expansion lattice scheme with time discretization to derive a numerical scheme for BSDEs.

Finally, we believe that our result in combining the Fourier expansion and lattice sequences opens many research opportunities in extending the QMC results to general measures. For example, we may consider developing an adaptive integration scheme based on the half-cosine reproducing kernel.

7.2.2. STOCHASTIC GRID BUNDLING METHOD

The numerical results in this work suggest that truncation maybe unnecessary for SGBM when the number of simulation samples is high enough. This properties should be included in the error bound of SGBM by applying convergence results for matrix equations.

A proof for non-refinement partitioning is still required too. One possibility is to prove convergence with a refinement sub-partition and relate other partitionings to this sequence.

Finally, our proof for the error bound is inspired by the classical result for non-parametric least-squares regression. The resulting convergence rate is therefore in the order of $N^{\frac{1}{2}}$. We may consider including results from importance sampling to improve our error bound.

7.2.3. GENERAL OUTLOOK

In this thesis, we have mainly focused on classical decoupled FBSDEs. However, this is not a fundamental requirement for the expectation algorithm. It would be of great interest to test the above algorithms with BSDEs under various assumptions. Indeed, the authors applied the COS method with coupled FBSDEs in [2]. A similar strategy will also work for our localized Fourier expansion schemes, the SWIFT method and the lattice expansion scheme, and they should be able to solve coupled systems.

There is also a practical reason to extend the applicability of our numerical algorithms. For example, a McKean type BSDE is required to solve initial margin valuation adjustment [3].

All in all, there are plenty of research questions remaining to be tackled. This thesis provides a mathematical foundation for numerical integration schemes and probabilistic schemes for BSDEs. Our results open doors for further basic numerical analysis research, especially with the combination of Fourier and lattice schemes and the rigorious analysis of SGBM. As we stated in the introduction chapter, we believe that numerical analysis researches results from our work will pave the way for better industrial tools for risk management.

REFERENCES

[1] J. Dick, D. Nuyens, and F. Pillichshammer, *Lattice rules for nonperiodic smooth integrands,* Numerische Mathematik **126**, 259 (2014).

[2] T. Huijskens, M. J. Ruijter, and C. W. Oosterlee, *Efficient numerical Fourier methods for coupled forward–backward SDEs,* Journal of Computational and Applied Mathematics **296**, 593 (2016).

[3] A. Agarwal, S. De Marco, E. Gobet, J. G. López-Salas, F. Noubiagain, and A. Zhou, *Numerical approximations of mckean anticipative backward stochastic differential equations arising in initial margin requirements,* ESAIM: Proceedings and Surveys **65**, 1 (2019).

ACKNOWLEDGEMENTS

It may not take a village to finish a doctorate, it is definitely not a one man journey.

First and foremost, I could not have embarked on this adventure without the support of my promotor and supervisor Prof. Kees Oosterlee. This dissertation contains numerous inputs from you through your timely comments and detailed proofreading. More importantly, I cannot work in my preferred way without your trust and faith in me. Thank you.

A part of this project is conducted in VORtech as an industrial secondment. I thank all the programmers and workers in VORtech for their welcoming and helpful attitude. All your code reviews and comments made me a better programmer with a deep appreciation for the art of software engineering. Special thanks go to Jok and Johan for their patient and understanding. It was not always smooth sailing for me to satisfy all parties involved and to conduct research I am proud of but your work made it possible.

With the blessing of luck, I was part of a European consortium with six early stage researchers working toward improving risk management. Enrico, Sidy, Zuzanna and Anastasia, it was a great honour to share all these training sessions, meetings and conferences with you. In particular, I would like to thank Andrea for the uncountable Maths weekends, dim sum lunches and rant sessions. It is always better to share the burden with a team.

It was also not possible to finish this journey without being adequately prepared for it in the first place. I can't count how many times have I recalled what I have learned during my master in the last 4 years. Thank you, Dr Yam, for setting up the foundation.

Besides research, I have received tremendous help for other aspects as well.

My life in CWI would not have been as easy and efficient without Nada's help. Your work saves me from all the administrative troubles. Our office will not be as lively without you, Sangeetika. Thank you for showing me the art of Indian food and whiskey, for sharing countless time with me in the office outside working hours, or simply for being the closest person resembling family to me when I am far from home. Thank you Willem Jan for always sharing your reasonable insight with me. It was a pleasure to work at CWI and share memories with everyone I met there.

There is simply not enough words of gratitude for everyone who played a part in my journey. I would like to show my appreciations for the followings: Nick and Debarati for all the insightful conversations we had; Krzysztof and Kristoffer for all the discussions about maths; Prashant for all your niceness; Alvaro for all the time we pretended to work; Alfred for being a role model and showing me what could have been; Gemma for the salty beer; Bea, Luis and Hema for all the fun time you brought; Rebecca for all your positivenesses; Ghaitrie for always offering to help; my friends from the Hague for showing me other sides of the world; all the Lindy hoppers for giving me a home base;... The list goes on and on.

Last but not least, thanks mum and dad, for letting me run wild since I was young but always giving me a comfort that I can return to.

I cannot thank you all enough for not just helping me get through this journey, but also making me a better person, to be open-minded and ready to learn but hold dear to one's principles.

This thesis is finished at a time of turmoil for the city that shaped me. There are scenes on the street that I could not have imagined 4 years ago and I do not know how things will turn out. Nevertheless, I am hopeful because of the courage, creativity and determination Hong Kongers have shown.

願榮光歸香港!

CURRICULUM VITÆ

Ki Wai CHAU

08-01-1991	Born in Hong Kong.

EDUCATION

2009–2012	Bachelor of Science in Mathematics The University of Hong Kong, Hong Kong
2012–2014	Master of Philosophy in Financial Mathematics The University of Hong Kong, Hong Kong
2015–2018	Marie-Curie Early Stage Researcher Centrum Wiskunde & Informatica, Amsterdam, the Netherlands
2018–2019	PhD. Researcher Delft University of Technology, Delft, the Netherlands
2020	PhD. Applied Mathematics Delft University of Technology, Delft, the Netherlands *Thesis:* Numerical Finance with Backward Stochastic Differential Equations *Promotor:* Prof. dr. ir. C.W. Oosterlee

LIST OF PUBLICATIONS

5. **K. W. Chau** and C. W. Oosterlee, *Exploration of a cosine expansion lattice scheme*, working paper.

4. **K. W. Chau**, J. Tang and C. W. Oosterlee, *An SGBM-XVA demonstrator: a scalable Python tool for pricing XVA*, Journal of Mathematics in Industry (to appear).

3. **K. W. Chau** and C. W. Oosterlee, *Stochastic grid bundling method for backward stochastic differential equations*, International Journal of Computer Mathematics, **96**, 2272 (2019).

2. B. Bouchard, **K. W. Chau**, A. Manai and A. Sid-Ali, *Monte-Carlo methods for the pricing of American options: a semilinear BSDE point of view*, ESAIM: Proceedings and Surveys, **65**, 294 (2019).

1. **K. W. Chau** and C. W. Oosterlee, *On the wavelet-based SWIFT method for backward stochastic differential equations*, IMA Journal of Numerical Analysis, **38(2)**, 1051 (2018).

List of Attended Conferences with Presentations

Talks

5. **9th International Congress on Industrial and Applied Mathematics**, Valencia, Spain, July 2019.
4. **10th World Congress of The Bachelier Finance Society**, Dublin, Ireland, July 2018.
3. **17th Winter school on Mathematical Finance** (invited), Lunteren, The Netherlands, January 2018.
2. **The Quantitative Methods in Finance 2017 Conference**, Sydney, Australia, December 2017.
1. **2nd International Conference on Computational Finance**, Lisbon, Portugal, September 2017.

Posters

3. **CEMRACS 2017**, Marseille, France, July 2017.
2. **International Workshop on BSDEs and SPDEs**, Edinburgh, United Kingdom, July 2017.
1. **10th Actuarial and Financial Mathematics Conference**, Brussels, Belgium, February 2017.