Imperial College London

IMPERIAL COLLEGE LONDON

DEPARTMENT OF MATHEMATICS

Dynamically Controlled Kernel Estimation for XVA Pricing and Options Replication

Author: Qingxin Geng (CID: 01205631)

A thesis submitted for the degree of

MSc in Mathematics and Finance, 2019-2020

Declaration

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

Signature: Qingxin Geng
Date: Sep. 8th, 2020

Abstract

Control variates, kernel regression and Gaussian Process Regression are popular tools applied in the field of mathematics. However, by drawing inspiration from Black-Scholes discrete-time hedging and making a combination of these tools together, a powerful pricing and hedging framework, Dynamically Controlled Kernel Estimation (DCKE), is developed. The new algorithm gives robust estimates of the conditional expectation of option values in the future given the information at the current time step, which outperforms the current industry standard Least Squares Monte Carlo (LSM) approach, especially for the prediction of sensitivities and around the 'tails'. DCKE is further enhanced by using quasi-Monte Carlo method with a rate of convergence only $\mathcal{O}(1/n)$ under n paths. The algorithm is independent of the model choice for underlying assets and also valid for more complex path-dependent and multi-dimensional options, improving the current state of the art of XVA computation.

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my industrial supervisor Gordon Lee from UBS for his expert guidance and valuable inspirations throughout this thesis. It is a precious and exciting opportunity to work with him on this project and his constant encouragement and thorough feedback during regular meetings do motivate me to keep exploring in depth and pursuing excellence.

I would also like to show my sincere thanks to my academic supervisor Dr Jack Jacquier for his inspiring suggestions through effective communications as the thesis goes on, which are of value to my report.

In addition, I am really appreciated that Gordon's colleague and friend, other XVA specialists, Dr Emma Jones from UBS and Dr Ignacio Ruiz from MoCaX Intelligence have kindly provided me with helpful advice in regards to my report.

Great thanks to MSc Mathematics and Finance programme and each of my lecturer for the knowledge I gain throughout the year that is helpful for but not limited to this project.

Last but not least, I would like to thank my family, especially my parents for their constant love, patience and support. I am sincerely grateful for the company and encouragement from my close friends during the past few months.

Contents

List of Figures

List of Tables

Introduction

In the financial industry, Credit Valuation Adjustment (CVA) is a change to the market value of derivative instruments to account for counterparty credit risk. It is the most widely known of the valuation adjustments, collectively known as X-Value Adjustment (XVA). The counterparty risk is taken on by an entity that enters an OTC contract with one or more counterparties having relevant default probabilities, and the counterparty credit exposure measures the maximum potential loss of the entity if its counterparties default. Banks use to measure counterparty risk internally with mainly two measures: Potential Future Exposure (PFE), as known as quantiles, which is mainly used internally to monitor when the credit limits with the counterparties are breached, and Expected Positive Exposure (EPE), which is used, when combined with other quantities, for the calculation of Exposure at Default (EAD), i.e., the value an entity is exposed when the counterparty defaults, and the capital requirements due to counterparty risk. This last calculation may combine exposures with default probabilities and recovery estimates, and it produces an approximation to Credit VaR, which is used as a capital requirement.

In terms of calculation, the main interest is to compute the conditional expectation of a derivatives portfolio in the future $\Pi\left(S_{t_{i+1}}, t_{i+1}\right)$ at time t_{i+1} given the information of the underlying spots S_{t_i} under the current state at time t_i, i.e.

$$\mathrm{E}^{\mathrm{Q}}\left[\Pi\left(S_{t_{i+1}}, t_{i+1}\right) | S_{t_i}, t_i\right],$$

from which XVA, Potential Future Exposure (PFE), Expected Positive Exposure (EPE), and Exposure at Default (EAD) are obtained. We try to replicate the portfolio value perfectly and obtain the possibly accurate sensitivities at all the time steps. Also, a proper convergence to the true value within a restricted number of simulation paths is desirable due to the highly-computationally intensive feature of XVA.

For CVA as an example, it is given by the risk-neutral expectation of the discounted loss, formulated as

$$\mathrm{CVA}(t_T) = LGD \int_0^{t_T} \mathrm{E}^{\mathrm{Q}}\left[\frac{B_{t_0}}{B_{t_i}}\Pi(t_i) \mid t_i = \tau\right] d\mathrm{PD}(0, t_i),$$

where t_T is the maturity of the longest transaction in the portfolio, B_{t_i} is the future value of one unit of the base currency invested today at the prevailing interest rate for maturity t_i, LGD is the loss given default, τ is the time of default, $\Pi(t_i)$ is the exposure at time t_i mentioned above, and $\mathrm{PD}(t_s, t_i)$ with $s \leq$ i is the risk neutral probability of counterparty default between times t_s and t_i. These probabilities can be obtained from the term structure of credit default swap (CDS) spreads.

However, the future values of a portfolio are actually tricky to be computed both precisely and efficiently at the same time, and so does XVA. According to the Basel Committee on Banking Supervision (BCBS)'s July 2015 consultation document regarding CVA calculations, if CVA is calculated using 100 timesteps with 10,000 scenarios per timestep, then 1 million simulations are required. Calculating CVA risk would require 250 daily market risk scenarios over the 12-month stress period. CVA has to be calculated for each market risk scenario, resulting in 250 million simulations. These calculations have to be repeated across 6 risk types and 5 liquidity horizons, and as a result, potentially 8.75 billion simulations (Lee (2015)).

The current industry standard for computing the conditional expectation is to use the Least Squares Monte Carlo (LSM) approach. This prominent approximate dynamic programming (ADP) for the evaluation of early and multiple exercise options, firstly American options, was pioneered by Carriere et al. (1996), Longstaff and Schwartz (2001), and Tsitsiklis and Van Roy (2001). Cesari et al. (2009) first introduced how this framework could be applied for pricing credit exposures, as the intermediate valuations over time and scenarios are provided. Also in the literature, Potters

et al. (2001) and Pochart and Bouchaud (2004) proposed Hedged Monte Carlo, which performed regressions on both the optimal hedge and prices, which were however not accepted widely due to their convergence properties: the difficulties in finding the right set of parameters that delivers accurate option prices while computing the prices by regression directly. Instead, Grau (2008) performed regressions to compute the optimal hedge only in order to preserve the convergence of the estimated prices to the true option values and release its dependency on the choice of basis functions.

In this paper, we propose a new algorithm for estimating the sensitivity and replicating portfolios, Dynamically Controlled Kernel Estimation (DCKE), which is an enhanced and improved discrete-time framework compared to all existing approaches in the literature. We concentrate on modelling the conditional expectation of the portfolio value, which is the key throughout this paper. Chapter 1 starts from the most straightforward Least Squares Monte Carlo approach, the results of which highly depends on the chosen basis functions. Then in Chapter 2, we develop a new way of estimating deltas utilising kernel regression to compute the solution to the variance minimisation strategy. This solution is proved to converge to the continuous-time Black-Scholes-Merton (BSM) model, and is in exactly the same form as the optimal control variate parameter. This enables us to add control variates with either the kernel estimator or the pathwise delta as the parameter. Combining these with non-parametric Gaussian process regressions, a powerful discrete-time pricing and hedging framework is built up. The newly proposed DCKE algorithm is proved to outperform the traditional LSM particularly in the estimation of deltas and the 'tails' in examples of the Black-Scholes and the Heston model, and also provides more precise intermediate-value predictions than both Grau's and Potters' methods. In addition, we adopt quasi-Monte Carlo (QMC) method by replacing geometric Brownian motion with quasi-random sequences, Sobol sequence as an example, for the simulation of underlying spots, in the contrast to pseudorandom sequences. Taking advantage of the faster rate of convergence of QMC ($\mathrm{O}(1/n)$ rather than $\mathrm{O}(1/n^{0.5})$ for regular MC, with n the number of simulation paths), the computational efficiency is significantly improved. Later in Chapter 3, we extend DCKE model to a high-dimensional algorithm and price more complex products such as path-dependent barrier options and multi-dimensional basket and rainbow option. The validity of the DCKE algorithm is verified in all these exotic options. We further compare the high-dimensional DCKE algorithm with a high-dimensional direct sampling approach as well as the one-dimensional DCKE algorithm when we compress multiple assets into a single dimension, and reveal the potential curse of dimensionality problems.

Chapter 1

Least Squares Monte Carlo

In this chapter, we introduce the Least Squares Monte Carlo (LSM) approach, together with a demonstration of the algorithm on a European option and its drawbacks. This basic technique for mimicking option values is a cornerstone of our paper, from which we build on new estimation methodologies and frameworks in later chapters to overcome its shortcomings.

1.1 Methodologies and setups

The Least Squares Monte Carlo approach, pioneered by Carriere et al. (1996), Longstaff and Schwartz (2001), and Tsitsiklis and Van Roy (2001), is a prominent approximate dynamic programming (ADP) methodology according to Powell (2011, pg 307), for the valuation and management of early and multiple exercise options. It is regarded as a key method in simplifying the calculation of financial instruments that do not have closed form solutions, and enables the computation of conditional expectations. Different from the standard option pricing problem, our objective is to obtain the option value for each intermediate time steps other than just the starting point. Mathematically, the price of an exercisable security is the discounted expected value of the payoff at the optimal stopping time, which can be formulated as an optimal stopping problem

$$V\left(S_{t_i}, t_i\right) = \sup_{\tau \in \mathcal{T}} \mathrm{E}^{\mathrm{Q}}\left[e^{-r(\tau - t_i)} P\left(S_\tau, \tau\right)\right], \tag{1.1.1}$$

with the option value V, the asset price S_{t_i} at time t_i, the risk-free interest rate r and all set of possible stopping times $\mathcal{T}$ in the risk-neural measure Q. For example, Equation (1.1.1) leads to the idea of LSM for pricing American options, since one only has to estimate an early exercise strategy within the Monte Carlo method for vanilla options.

We first focus onto the LSM for European vanilla options, the steps of which consists of path generation and Least Squares regression with the chosen basis functions.

1.1.1 Path generation

We start with the simplest path generation technique in this section, i.e. assuming the stock price follows the Black–Scholes model, and then the asset price process S_t follows a geometric Brownian motion (GBM). However, GBM is just a model placeholder, and can be replaced with any other path generators, which would be seen in Chapter 2 and 3. Following GBM, S_t evolves in the risk-neutral fashion

$$\mathrm{d}S_t = rS_t\mathrm{d}t + \sigma S_t \mathrm{d}W_t, \tag{1.1.2}$$

where W_t is a Wiener process or Brownian motion and the risk-free interest rate r and the percentage volatility σ are constants.

The easiest sampling technique for Equation (1.1.2) is the Euler method, which samples n trajectories S_t^j, $j = 1, \ldots, n$ at several time steps $t_i, i = 1, \ldots, T$, starting at $S_{t_0}^1 = S_{t_0}^2 = \ldots = S_{t_0}^n =: S_{t_0}$ as

$$S_{t_{i+1}}^j = S_{t_i}^j + rS_{t_i}^j\left(t_{i+1} - t_i\right) + \sigma S_{t_i}^j \theta_{i,j}\sqrt{t_{i+1} - t_i}, \tag{1.1.3}$$

with $\theta_{i,j}$ drawn from a standard Normal distribution.

Then a better discretization is turned out given by

$$S^j_{t_{i+1}} = S^j_{t_i} \exp\left(\left(r - \frac{1}{2}\sigma^2\right)(t_{i+1} - t_i) + \sigma\sqrt{(t_{i+1} - t_i)}\theta_{i,j}\right). \tag{1.1.4}$$

This method is exact in the sense that the distribution of S_{t_T} does not depend on intermediate time steps as in Equation (1.1.3). Consequently, only one time step is required for the valuation of a vanilla European option

$$S^j_{t_T} = S_{t_0} \exp\left(\left(r - \frac{1}{2}\sigma^2\right)(t_T - t_0) + \sigma\sqrt{(t_T - t_0)}\theta_j\right),$$

with θ_j drawn from a standard Normal distribution.

The risk-neutral asset price trajectories can then be generated by Equation (1.1.4) from time 0 through time T starting at the current price S_{t_0}.

1.1.2 Least Squares regression

Regressions are applied in LSM since it enables us to work out the conditional expectations, and hence an approximation for the option prices. Among different regression methods, we focus on Least Squares regressions due to their desirable properties as described below. In the context of option pricing, there are two main applications of the regression: one is the function approximation, and the other is a variance minimisation of a portfolio.

To introduce the function approximation, we first need to make the following assumptions:

1. A data set $(\boldsymbol{X}, \boldsymbol{y}), \boldsymbol{X} \in \boldsymbol{R}^{n,s}, \boldsymbol{y} \in \boldsymbol{R}^n$ is provided.
2. The rows $\boldsymbol{x}^i \in \mathbb{R}^s$ of $\boldsymbol{X} := \left(\boldsymbol{x}^1, \ldots, \boldsymbol{x}^n\right)^T$ are independent and identically distributed (i.i.d.) realizations of a random vector with a probability density function $p(x)$, which is non-zero everywhere on the cube $\mathrm{D} := [\boldsymbol{x}_{\min}, \boldsymbol{x}_{\max}] = \left(\left[x_{j,\min}, x_{j,\max}\right]\right)_{j=1,\ldots,s}$ and zero outside.
3. $\boldsymbol{y} = \left(y^1, \ldots, y^n\right)^T$ are noisy known observations of $\left(\boldsymbol{x}^i\right), i = 1, \ldots, n$, with $y^i = f\left(\boldsymbol{x}^i\right) + \epsilon^i, i = 1 \ldots, n$, where ϵ^i is random with $\mathrm{E}\left[\epsilon^i\right] = 0$ and is independent of x^i.
4. The function $f : \mathbb{R}^s \to \mathbb{R}, f \in \mathcal{B}$ has a representation $f(\boldsymbol{x}) = \sum_{j=1}^{\infty} a_j b_j(\boldsymbol{x}), \boldsymbol{x} \in \mathbb{R}^s$, where $b_j \in \mathcal{B}, j = 1, \ldots, \infty$ are bounded basis functions $b_j : \mathrm{R}^s \to \mathrm{R}$ of a vector space $\mathcal{B} \subset \mathcal{C}^1$ with $\left\|b_j(\boldsymbol{x})\right\|_\infty = c_j < \infty, \exists \boldsymbol{x} \in \boldsymbol{D} : \left|b_j(\boldsymbol{x})\right| > 0, j = 1, \ldots, \infty$.

And then we have the below definition and theorem that describe an approximation:

Definition 1.1.1. Let the four above assumptions be satisfied. A *local basis approximation* $\tilde{f}^m$ of the function f induced by the set of samples $(\boldsymbol{X}, \boldsymbol{y}), \boldsymbol{X} \in \mathbb{R}^{n,s}, \boldsymbol{y} \in \mathbb{R}^n$ with function space $\mathcal{B}^m$ spanned by the basis functions $b_1, \ldots, b_m \in \mathcal{B}$, is given by

$$\tilde{f}^m(\boldsymbol{x}) = \sum_{j=1}^{m} \tilde{a}^n_j b_j(\boldsymbol{x}), \quad \tilde{f}^m \in \mathcal{B}^m \subset \mathcal{B}, \quad \tilde{a}^n_j \in \mathbb{R}, \quad j = 1, \ldots, m$$

with coefficient vector $\tilde{\boldsymbol{a}}^m = A(\boldsymbol{X}, \boldsymbol{y})$, $\tilde{\boldsymbol{a}}^m = \left(\tilde{a}^n_1, \ldots, \tilde{a}^n_m\right)^T$ iff

$$\forall \epsilon \quad \exists N(\epsilon) : \left\| \begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix} - \begin{pmatrix} \tilde{a}^n_1 \\ \vdots \\ \tilde{a}^n_m \end{pmatrix} \right\|_\infty < \epsilon, \quad \forall n \geq N(\epsilon).$$

Theorem 1.1.2. *In the local basis approximation $\tilde{f}^m(\cdot)$ of function $f(\cdot)$ based on a set $(\boldsymbol{X}, \boldsymbol{y})$ of n noisy observations in Definition 1.1.1, the coefficient vector $\tilde{\boldsymbol{a}}^m = \left(\tilde{a}^n_1, \ldots, \tilde{a}^n_m\right)^T$ satisfies*

$$\tilde{\boldsymbol{a}}^m = A(\boldsymbol{X}, \boldsymbol{y})) = \arg\min_{\tilde{\boldsymbol{a}}^m} \left\|\boldsymbol{B}(\boldsymbol{X})\tilde{\boldsymbol{a}}^m - \boldsymbol{y}\right\|_2 \tag{1.1.5}$$

$$= \left(\boldsymbol{B}(\boldsymbol{X})^T \boldsymbol{B}(\boldsymbol{X})\right)^{-1} \boldsymbol{B}(\boldsymbol{X})^T \boldsymbol{y}, \tag{1.1.6}$$

where $\boldsymbol{B}(\boldsymbol{X}) := \begin{pmatrix} b_1(\boldsymbol{x}^1) & \cdots & b_m(\boldsymbol{x}^1) \\ \vdots & \ddots & \vdots \\ b_1(\boldsymbol{x}^n) & \cdots & b_m(\boldsymbol{x}^n) \end{pmatrix}$, *and* $b_j(\boldsymbol{x}) \in \mathcal{B}, j = 1, \ldots, m$ *are the basis functions.*

Longstaff and Schwartz (2001) first proposed LSM for pricing American options in order to estimate the conditional expected payoff to the option holder from continuation, and we use the method for European options here. With no allowance of early exercise, the expected option value under the risk-neutral measure Q at each time step t_i is denoted by $\mathrm{E}_Q\left[V\left(S_{t_{i+1}}, t_{i+1}\right)|S_{t_i}, t_i\right]$. We approximate this value by the LSM approach, written as

$$P\left(S_{t_i}, t_i\right) \approx \mathrm{E}_Q\left[V\left(S_{t_{i+1}}, t_{i+1}\right)|S_{t_i}, t_i\right].$$

The value $P\left(S_{t_i}, t_i\right)$ is computed using a Least Squares regression of multiple path-realizations S^j on some basis functions b_k, i.e. the local basis approximation of $V(S_{t_{i+1}}, t_{i+1})$ given S_{ti} following Theorem 1.1.2. The regressions start from one step before maturity, i.e. the time step t_{T1}, while the approximated values are

$$P\left(S_{t_i}, t_i\right) = \sum_k a_k^i b_k\left(S_{t_i}\right),$$

with some basis functions b_k and unknown coefficients a_k^i minimizing

$$\left\| \left(\sum_k a_k^i b_k\left(S_{t_i}^j\right) - e^{-r(t_{i+1}-t_i)} V_{t_{i+1}}^j \right)_{j=1,\ldots,n} \right\|_2,$$

where $V_{t_T}^j = P(S_{t_T}^j, t_T)$ is the final option payoff at the maturity, given by $\left(K - S_{t_T}^j\right)^+$ for Put option and $\left(S_{t_T}^j - K\right)^+$ for Call option, with the strike price K. And a dynamic program solves for all values $V_{t_i}^j$, starting at time t_T and iterating backwards to t_0.

1.1.3 Basis functions

According to Longstaff and Schwartz (2001), in some cases the choice of the class of basis functions seems to have little effect on the values computed by Least Squares Monte Carlo. And the simplest basis functions to use are polynomials. In this paper, we try the following polynomial interpolations and compare how they would affect our simulation results.

- Monomial basis

  For the simplest polynomial basis, we use the full set B_ℓ^{full} of all s-dimensional monomials up to a certain polynomial degree $\ell = (\ell_1, \ldots, \ell_s) \in \mathbb{N}^s$,

  $$B_\ell^{\text{full}}\ (x_1, \ldots, x_s) := \left\{ \prod_{i=1}^s x_i^{g_i} | g_i \in \mathbb{N}_0 \wedge g_i \leq \ell_i \right\}.$$

  Longstaff and Schwartz suggested that increasing the number of basis functions leads to a better approximation. In the 1-dimensional case in LSM, we choose monomials up to the power of N in x $:= S_{t_T}$ such that $b_k\left(S_{t_i}\right) = S_{t_i}^{k-1}$, where $k = 1, \ldots, N$ and $i = 0, \ldots, T$. Consequently, the matrix $\boldsymbol{B} := \boldsymbol{B}\left(S_{t_i}\right)$ for time step i in the regression problem is

  $$\boldsymbol{B} = \left(1 \quad S_{t_i}^j \quad \left(S_{t_i}^j\right)^2 \quad \ldots \right)\Big|_{j=1,\ldots,n}.$$

However, Judd (1998) shows that orthogonal polynomials can solve the multicollinearity problem when dealing with multidimensional approximation problems and so that are better than simple monomials.

- Orthogonal polynomial basis

  Four examples of orthogonal polynomials are listed in Table 1.3. For the implementation, we focus on the first two orthogonal polynomials as basis functions.

  Chebyshev (1853) was probably the first mathematician to recognise the general concept of orthogonal polynomials. His work arose out of the theory of least squares approximation and probability, and he applied his results to interpolation, approximate quadrature and other areas. Being defined via orthogonality with respect to the most obvious weight function on a finite interval, it sets up the Legendre polynomials as one of the three classical orthogonal polynomial systems. The other two are Laguerre and Hermite, with weight functions that are the most natural analytic functions that ensure convergence of all integrals.

Polynomial	Weight $\omega(x)$	Interval	Definition
Chebychev	$\left(1-x^2\right)^{-1/2}$	$[-1,1]$	$b_k(x) = \cos(k\cosh(x))$
Legendre	1	$[-1,1]$	$b_k(x) = \frac{(-1)^k}{2^k k!}\frac{d^k}{dx^k}\left(1-x^2\right)^k$
Laguerre	$\exp(-x)$	$[0,\infty)$	$b_k(x) = \frac{\exp(x)}{k!}\frac{d^k}{dx^k}\left(x^k\exp(-x)\right)$
Hermite	$\exp\left(-x^2\right)$	$(-\infty,\infty)$	$b_k(x) = (-1)^k\exp\left(x^2\right)\frac{d^k}{dx^k}\exp\left(-x^2\right)$

Table 1.1: Sample orthogonal polynomials

1.1.4 Algorithm - Least Squares Monte Carlo

Algorithm 1 describes LSM approach in pseudo-code.

1.2 Demonstration of LSM on a European option

After the theoretical considerations, we now focus on a real example for the European Call option under both the Black-Scholes model and the Heston Model.

1.2.1 Black-Scholes model

In the discrete-time setting, details of which will be given in Section 2.1, we expect the simulation results to overlap Black-Scholes benchmark values when the number of paths tends to infinity and the lag between time steps is as small as possible.

Consider a European Call option with parameter settings in Table 1.5, we evaluate it with 10,000 asset paths and 4 time steps. We keep the same 9 basis functions as before, i.e. the highest degree of the polynomials is 8.

General features	
Initial stock price S_{t_0}	100
Strike price K	100
Risk-free rate r	5% p.a.
Volatility σ	40% p.a.
Maturity time t_T	1 year
Terminal value $P(S_{t_T}, t_T)$	$\max(S_{t_T}-K, 0)$

Table 1.2: Example European Call option

As we are interested in the intermediate time steps of the estimation procedure, we plot the three sets of intermediate results under monomial, chebychev and legrendre basis functions respectively.

Note that the simulated underlying spots are far less denser around both ends at each time step, and to ignore the influence of the extreme values at the 'tails' and effectively reduce the

Algorithm 1 Least Squares Monte Carlo

Input: K: strike; r: risk-free rate; T: number of time steps; Δt: lag between consecutive time steps; n: number of simulated paths; S: spot prices sequences S_{t_i} with each length n for $i = 0, \ldots, T$

Output: X: mesh points of spot price sequence X_{t_i}; $\hat{V}$: option prices sequences $\hat{V}_{t_i}$; $\hat{\Phi}$: deltas sequences $\hat{\phi}_{t_i}$ with each length n for $i = 0, \ldots, T$

Set arrays: $V^j_{t_T} = P(S^j_{t_T}, t_T)$: the payoff function, $j = 1, \ldots, n$; $\hat{V}$, $\hat{\Phi}$ (discounting payoff approach); $\hat{V}'$, $\hat{\Phi}'$ (backward recursion approach)

for all time steps from $i = T-1$ down to $i = 0$ **do**

  $X^j_{t_i}$ = samples drawn between $1\% - 99\%$ percentiles of $S^j_{t_i}$

  if $t_i = T-1$ **then**

    get α_{ik} and α'_{ik} from $\sum_k \alpha_{ik} b_k \left(S^j_{t_i}\right) = \sum_k \alpha'_{ik} b_k \left(S^j_{t_i}\right) \approx e^{-r\Delta t} V^j_{t_{i+1}}$

    $\hat{V}^j_{t_i} = \hat{V}'^j_{t_i} = \sum_k \alpha_{ik} b_k \left(X^j_{t_i}\right) = \sum_k \alpha'_{ik} b_k \left(X^j_{t_i}\right)$

  else

    ⚠ **discounting payoff approach**

    get α_{ik} from $\sum_k \alpha_{ik} b_k \left(S^j_{t_i}\right) \approx e^{-r\Delta t} V^j_{t_{i+1}}$

    $\hat{V}^j_{t_i} = \sum_k \alpha_{ik} b_k \left(X^j_{t_i}\right)$

    ⚠ **backward recursion approach**

    get α'_{ik} from $\sum_k \alpha'_{ik} b_k \left(S^j_{t_i}\right) \approx e^{-r\Delta t} V'^j_{t_{i+1}}$

    $\hat{V}'^j_{t_i} = \sum_k \alpha'_{ik} b_k \left(X^j_{t_i}\right)$

    $V'^j_{t_i} = \text{1D-interpolation}(X^j_{t_i}, \hat{V}'^j_{t_i}; S^j_{t_i})$

  end if

  $V^j_{t_i} = e^{-r\Delta t} V^j_{t_{i+1}}$

  $\hat{\phi}^j_{t_i} = \frac{\partial \sum_k \alpha_{ik} b_k \left(X^j_{t_i}\right)}{\partial X^j_{t_i}}$

  $\hat{\phi}'^j_{t_i} = \frac{\partial \sum_k \alpha'_{ik} b_k \left(X^j_{t_i}\right)}{\partial X^j_{t_i}}$

end for

return X, $\hat{V}$, $\hat{V}'$, $\hat{\Phi}$, $\hat{\Phi}'$

computational cost, we create a new set of 200 underlying spots lying evenly between the restricted 1% to 99% percentiles of the original stock prices at each time step. We try two different approaches here (see Algorithm 1 for pseudo-code) by regressing on either the discounted payoff for each time step (discounting payoff approach; solid line) or the resulting option value we get from the last time step (backward resursion apprach; dashed line) and compare the prediction with the Black-Scholes benchmark as shown in Figure 1.1.

It could be observed that the three different basis functions give almost same results under a same approach, so we can treat their prediction results as the outcomes for general polynomial bases. The fitted price for each time step overlaps the Black-Scholes benchmark when the option is near-the-money, but is far from Black-Scholes at the 'tails'. Furthermore, the solid line and the dashed line overlap when $t = 0.75$ because we just discount the payoff back for one step and apply regression in both cases. For early time steps, the discounting payoff approach deviates slightly more from the benchmark model under a range of underlying asset values, but obviously outperforms the backward recursion technique at the ends because the error around the 'tails' in backward recursion method accumulates by each time step back. These observations are verified in Table 1.3, where errors under different metrics are provided after averaging for 100 repetitions. Other than the overall mean squared errors, which is defined as the average of the squared errors between the estimated option price and the true price, more realistic measures, expected exposures (EE), i.e. the expectation of the maximum of the estimated values and zero, and potential future exposures (PFE), i.e. the quantiles of the estimates, are also measured and compared with the corresponding Black-Scholes values. To especially focus onto the 'tails', we include the errors at both ends, i.e., the 1% and 99% quantiles.

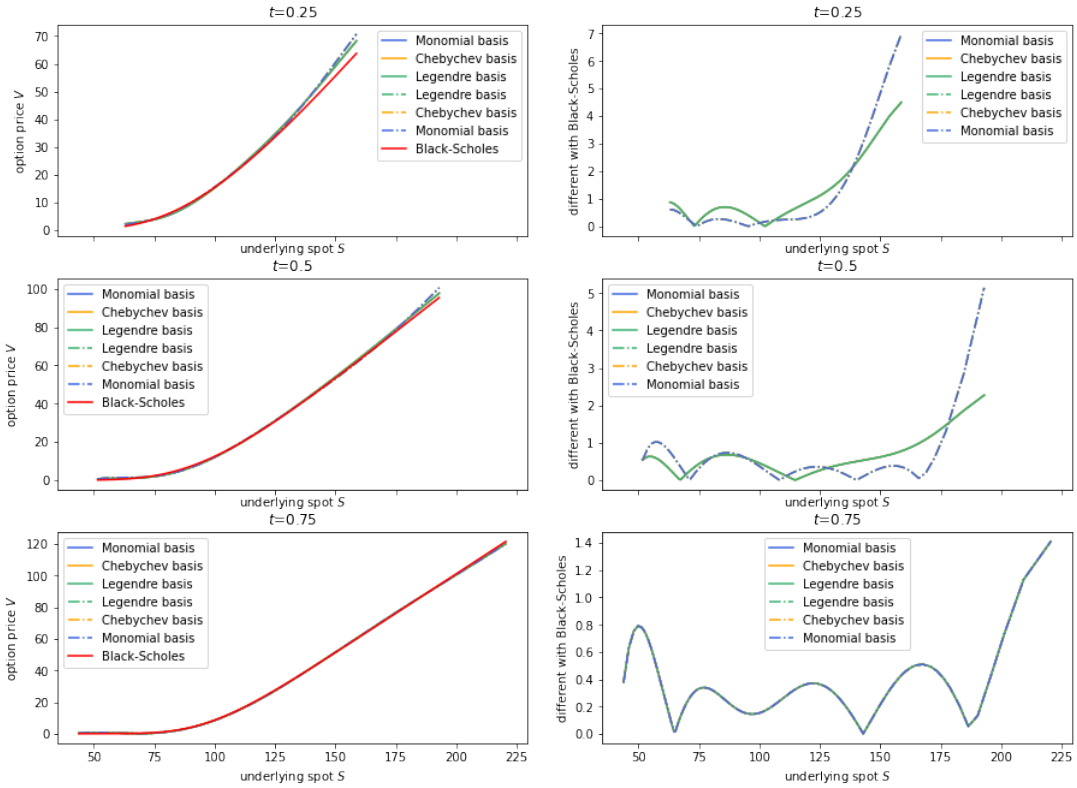

Figure 1.1: LSM for option prices under different basis functions versus Black-scholes pricess (left); difference with Black-Scholes (right): discounting payoff approach ('-'); backward recursion approach ('-·-')

Price	Metric	$t = 0.25$	$t = 0.50$	$t = 0.75$
Discounting payoff	MSE	0.4721	0.4617	0.3527
	mean	0.0118	0.0377	0.0280
	1%	0.0585	0.0977	0.1688
	50%	0.0404	0.0571	0.3063
	99%	0.1380	0.2841	0.0965
Backward recursion	MSE	0.2973	0.3793	0.3527
	mean	0.0251	0.0409	0.0280
	1%	0.0269	0.1653	0.1688
	50%	0.0419	0.0565	0.3063
	99%	0.4275	0.0296	0.0965

Table 1.3: Mean squared errors and absolute errors in EE and PFE of the estimated prices with polynomial bases with 8 degrees compared to Black-Scholes prices (average of 100 repetitions)

We then calculate the fitted derivatives of these polynomial basis functions by $\frac{\partial \sum_k \alpha_k b_k\left(S^j_{t_i}\right)}{\partial S^j_{t_i}}$, where b_k is each term of the polynomial we choose as basis here and α_k is the coefficient, in order to estimate one of the 'Greeks' in option pricing, delta, since it represents the sensitivity of the option relative to the underlying. As shown in Figure 1.2 and Table 1.4, the 'tails' discrepancies are even more obvious for both approach while estimating deltas with LSM. Again, backward recursion approach performs worse in catching the dynamics around the 'tails' because of the error accumulation step by step back in those extreme cases, but this approach again associates with slightly smaller MSE than discounting the payoff directly in the regression. We also note that if we include all simulation points without restricting the range and taking quantiles, backward recursion approach has significantly better performance on dealing with those extreme 'tails'.

So far, we analysed the results based on a randomly chosen number of basis functions, 9.

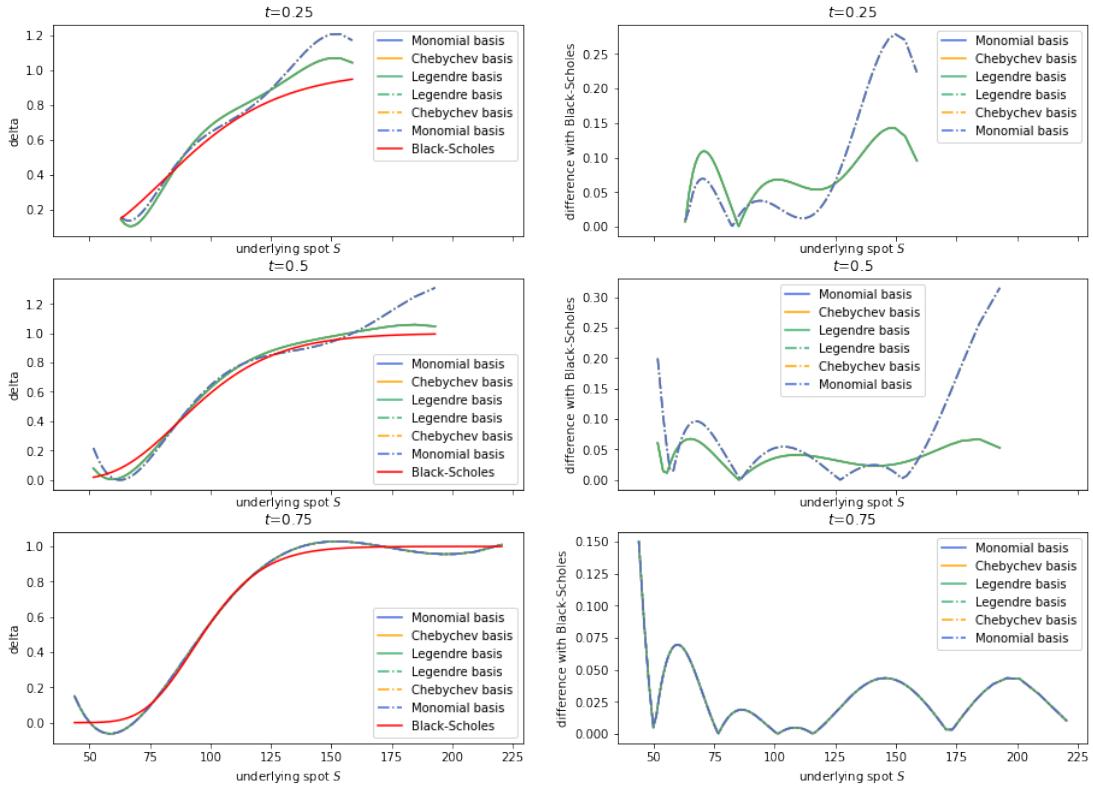

Figure 1.2: Fitted derivatives of the basis functions versus Black-scholes deltas (left); difference with Black-Scholes (right): discounting payoff approach ('-'); backward recursion approach ('-·')

Delta	Metric	$t = 0.25$	$t = 0.50$	$t = 0.75$
Discounting payoff	MSE	0.0036	0.0017	0.0016
	mean	0.0092	0.0069	0.0048
	1%	0.0152	0.0214	0.0718
	50%	0.0006	0.0069	0.0120
	99%	0.0894	0.0540	0.0530
Backward recursion	MSE	0.0018	0.0016	0.0016
	mean	0.0087	0.0067	0.0048
	1%	0.0155	0.0134	0.0718
	50%	0.0079	0.0060	0.0120
	99%	0.0744	0.0577	0.0530

Table 1.4: Mean squared errors and absolute errors in EE and PFE of the fitted derivatives with polynomial bases with 5 degrees compared with Black-Scholes deltas (average of 100 repetitions)

However, how would the estimates generally look like? We now compare the average of mean squared errors and absolute errors in quantiles among a variety of degrees of polynomials used for the estimation, after repeating each for 100 times. We vary the number of bases from 3 until 20 in Figure 1.3 and 1.4. Note that we only plot for the general polynomial bases in Figure 1.4 as we have shown the three polynomial bases we use give indistinguishable results.

First of all, it can be confirmed in Figure 1.3 and 1.4 that the outperformance of the discounting payoff approach is not an occasion. This approach works better than the backward recursion technique under most numbers of bases according to their MSE and absolute errors, especially for small and large number of basis functions due to the better prediction around the 'tails'. When the number of bases is below 5 or greater than 12 for delta, the MSE and absolute errors blow up remarkably. The reason is that when we use a tiny or large degree, either an underfitting or overfittiing problem occurs in the regression and cause prominent errors. It can also be observed that the absolute errors for 1% quantile and 99% quantile are far larger than the 50% quantiles

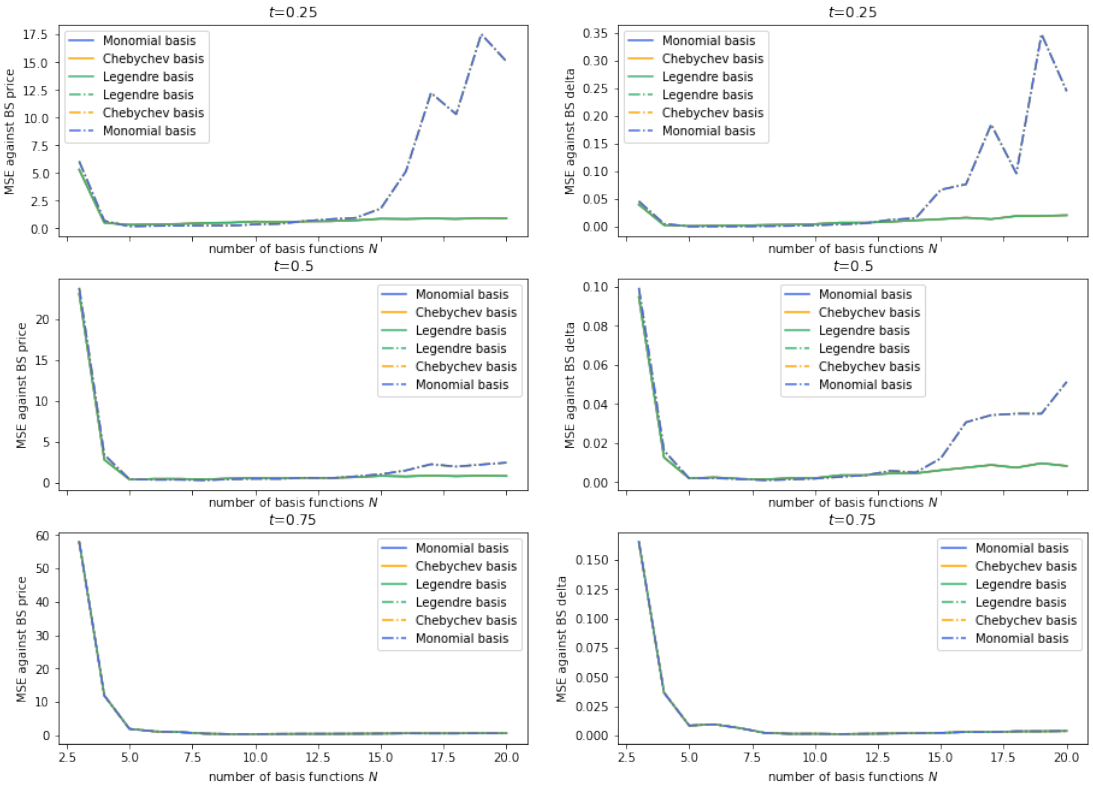

Figure 1.3: Mean squared errors of estimated prices (left); of fitted derivatives (right) against number of basis functions: discounting payoff approach ('-'); backward recursion approach ('-·') (average of 100 repetitions)

ones, which points out the 'tail' deviations in LSM. 7 to 12 bases would be sensible choices according to Figure 1.3 and 1.4, which shows that the example we take, with 9 basis functions, is representative.

In order to improve the fit at both ends, we now replace the regression method before with the piecewise regression. We break the underlyings into three segments, namely the 0%-20%, 20%-80% and 80%-100% quantiles of the underlying at each time step, and perform regression respectively in each case. As less points are used in less points are used in each segment, we reduce the number of basis functions to 7 to avoid overfitting. We again compare the estimated results against Black-Scholes prices and deltas, and the results are shown in Figure 1.5 and 1.6.

We can observe rather than the smooth curves in earlier regressions, the piecewisely-fitted results are more jagged as less data points are used for each segment. In the price estimation, the fit for the left tail, i.e. when the option is out-of-the-money, is significantly improved for each time step than the previous cases observing from the difference with Black-Scholes, while the right tails are slightly better predicted but deviations are still remained. Though the tails are fitted much better for both approaches, the middle parts, especially the linkages between two segments of regressions are far worse predicted than the single regression case. Same could be observed for the resulting delta estimates.

Until this point, we keep the same number of simulation paths, 10,000. An interesting question may be how would the estimates depend on the number of paths in this case. As these three polynomial bases give extremely close results after the regressions, we simply compare the estimates of monomial basis when the number of paths takes 1,000, 10,000, and 1,000,000 for simple regression (Figure C.1 and C.2 in Appendix C.1) and piecewise regression (Figure 1.7 and C.3).

The figures indicate that overall, a larger number of paths do solve the problems of 'tail' deviations to some extent, however only for earlier time steps. As time goes on further, the simulated underlying spot takes a broader range of values since more points are involved so that the paths are more diversified. Consequently, the underlying spots are less concentrated at the extreme 'tails', particularly the right 'tail', than in early time steps, and obvious estimation errors occur compared to Black-Scholes benchmark. The piecewise regression partially solves the inconsis-

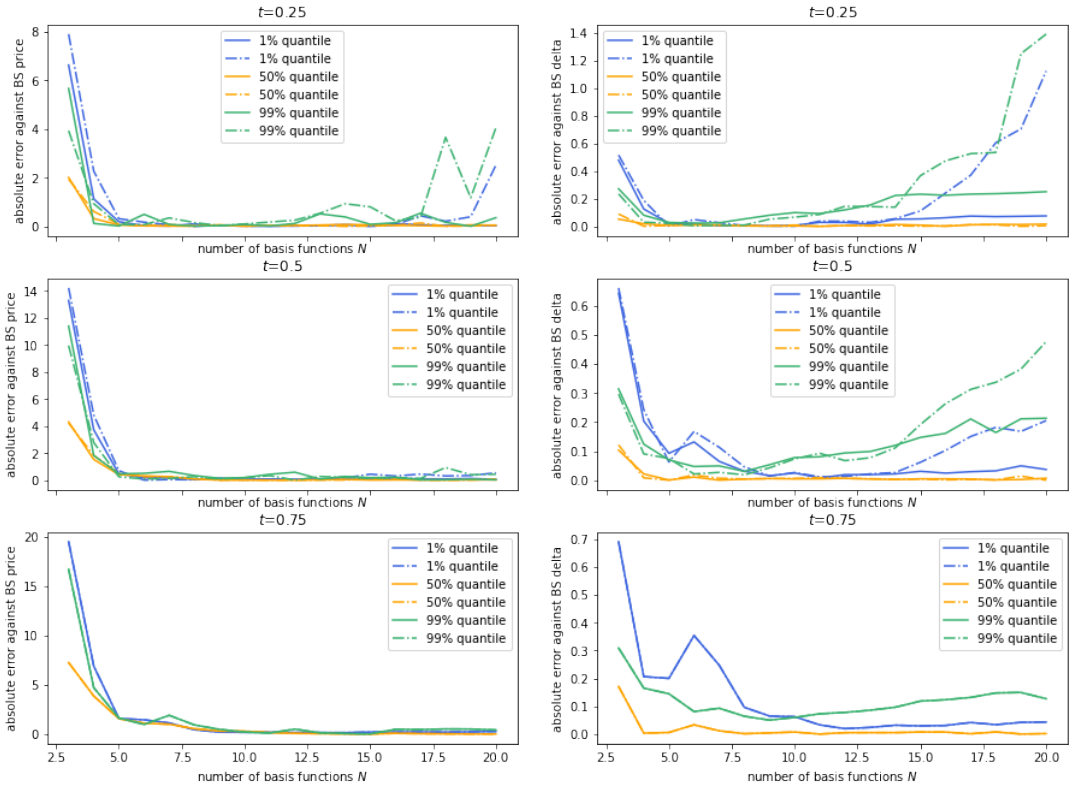

Figure 1.4: Absolute errors in 1%, 50% and 99% quantiles of estimated prices (left); of fitted derivatives (right) against number of polynomial basis functions: discounting payoff approach ('-'); backward recursion approach ('-·') (average of 100 repetitions)

tency between the estimates and Black-Scholes around the 'tails', especially for later time steps, but still doesn't capture the full dynamics of prices and deltas around the linkages of segments even if 1,000,000 paths are generated.

Therefore, there is a trade-off between computation cost and accuracy, and we can find for LSM, a huge number of paths 1,000,000 is needed in order to control the maximum estimation error to be less than 0.5 for prices and 0.05 for deltas.

1.2.2 Heston model

The Black-Scholes model provides a convenient closed-form formula for option prices, but are however unrealistic due partly to its inability to generate the volatility smile and the skewness in the distribution of the return. To capture such properties, Heston (1993) proposed a stochastic volatility model with a closed-form solution for the European call option prices when the underlying assets are correlated with a latent volatility stochastic process.

At time t, the spot asset obeys the following diffusion process under the risk neutral measure, with volatility being treated as a latent stochastic process:

$$dS_t = rS_t dt + S_t \sigma_t \left(\rho W_t^{(v)} + \sqrt{1-\rho} dW_t^{(S)} \right)$$

$$d\sigma_t^2 = \kappa \left(\theta - \sigma_t^2 \right) dt + v \sigma_t W_t^{(v)},$$

where $W_t^{(S)}$ and $W_t^{(v)}$ are both standard Wiener processes with a correlation coefficient $\rho > 0$:

$$dW_1(t) dW_2(t) = \rho dt.$$

The following parameters are extended from the initial Black-Scholes model, and we should ensure $2\kappa\theta > \nu^2$ for the origin, where the square-root function is not smooth, to be unattainable:

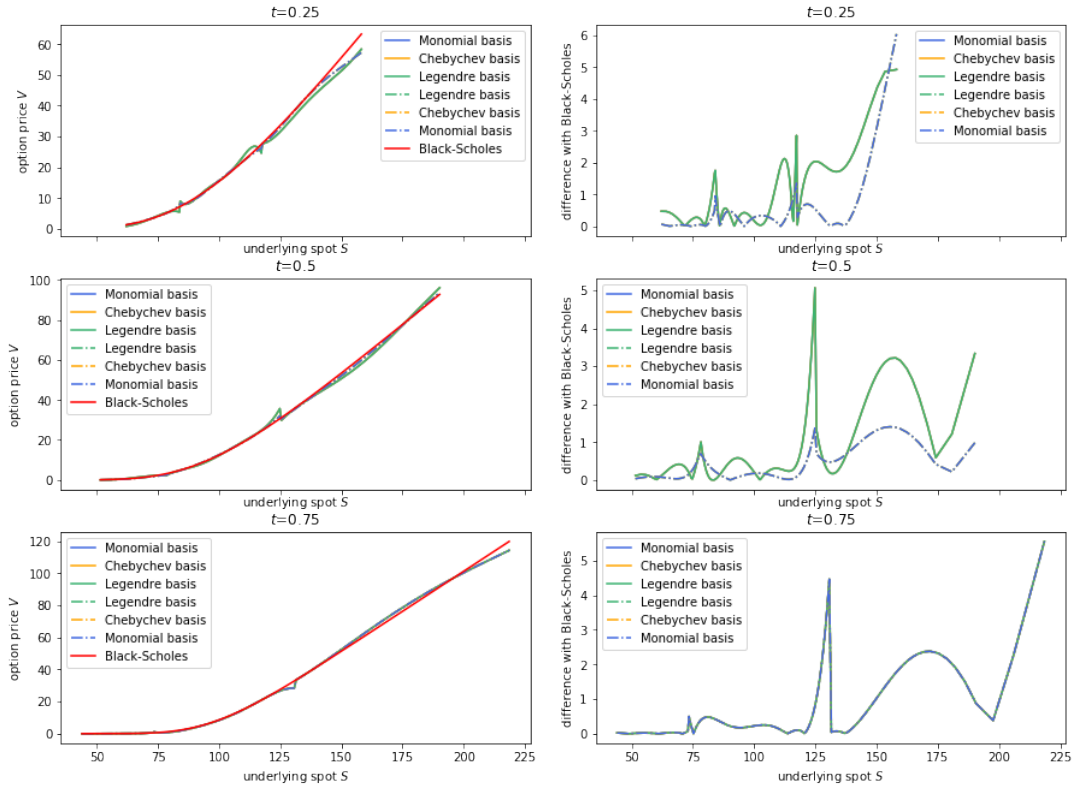

Figure 1.5: LSM for option prices with piecewise polynomial regressions versus Black-Scholes prices (left); difference with Black-Scholes (right): discounting payoff approach ('-'); backward recursion approach ('-·')

General features	
Instantaneous interest rate r	5% p.a.
Mean-reverting rate κ	1
Long-term variance θ	6% p.a.
Volatility of volatility ν	0.1
Initial variance $\sigma_0{}^2$	5% p.a.
Correlation coefficient ρ	-0.5

Table 1.5: Example European Call option - Heston model

We now look at how LSM applies to the estimation of prices for the Heston model. The analytical solution to the Heston model is not easily implemented due to numerical issues. We therefore resort to one of the most commonly used approximation formulae by Lewis (2009) (see Appendix B.1) as the Heston benchmark call prices. We follow Piterbarg (2003)'s version for deltas, in which the initial variance was replaced by the Black-Scholes volatility.

Figure 1.8 reflects the same problems of LSM for the Heston model as the previous Black-Scholes model. We get terrible fits for deltas, especially at the 'tails' for both prices and deltas.

1.3 Conclusions of LSM

The results above fully expose the drawbacks of LSM:

Remark 1.3.1. The estimates fail to converge properly to true values even with large numbers of paths, and the convergence is dependent on the degree of basis functions. The algorithm is neither computationally-efficiently nor feasible to obtain a set of accurate estimates. Also, without a sensible degree choice, the danger of underfitting or overfitting occurs.

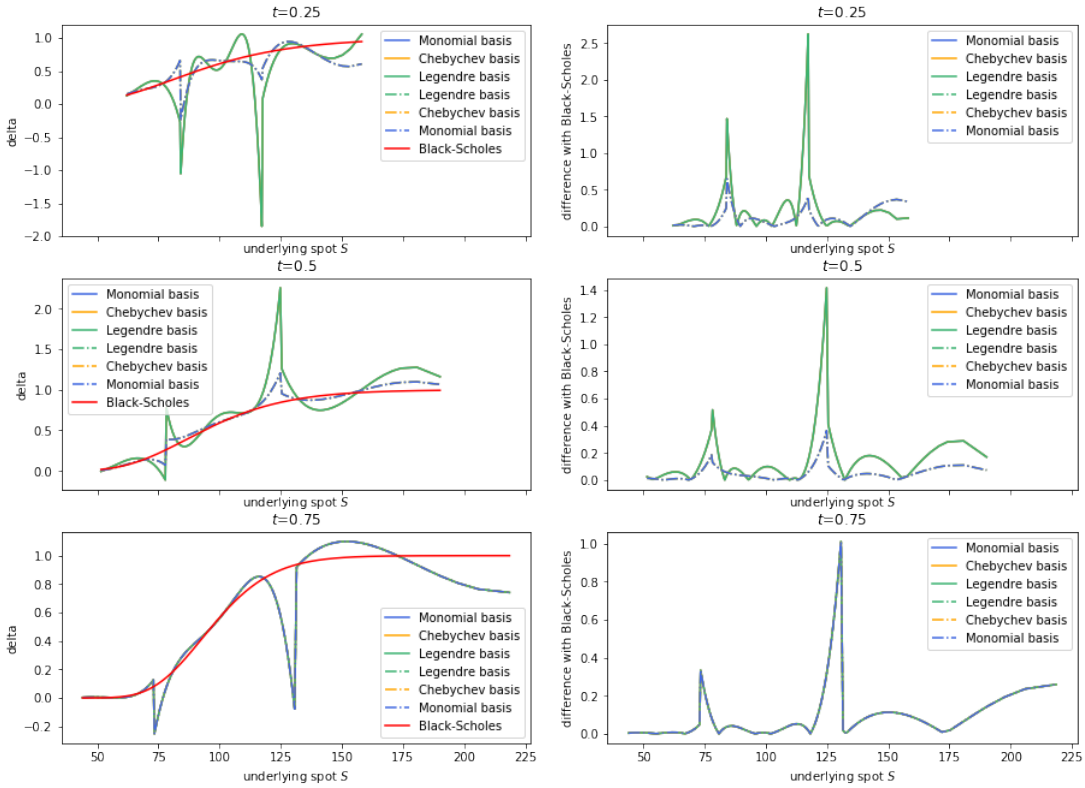

Figure 1.6: LSM for deltas with piecewise polynomial regressions versus Black-Scholes (left); difference with Black-Scholes (right): discounting payoff approach ('-'); backward recursion approach ('-·')

Remark 1.3.2. We are unable to rely on the polynomial bases to compute accurate deltas, which is problematic for the computation of quantities like Margin Value Adjustments (MVA). The fitted derivatives deviate significantly from the true values especially around the 'tails' due to the restriction of the polynomial bases, which contributes to the lack of convergence mentioned above.

Remark 1.3.3. Large discrepancies appear at both ends of the underlying. The results are poorly fitted around the 'tails', even after the refinement via piecewise regression.

To tackle these problems, we introduce the detailed discrete-time pricing and hedging framework in the next chapter, and estimate delta by a novel and far more accurate kernel regression approach, which also contribute to the price estimation while combining with Gaussian process regression and control variates.

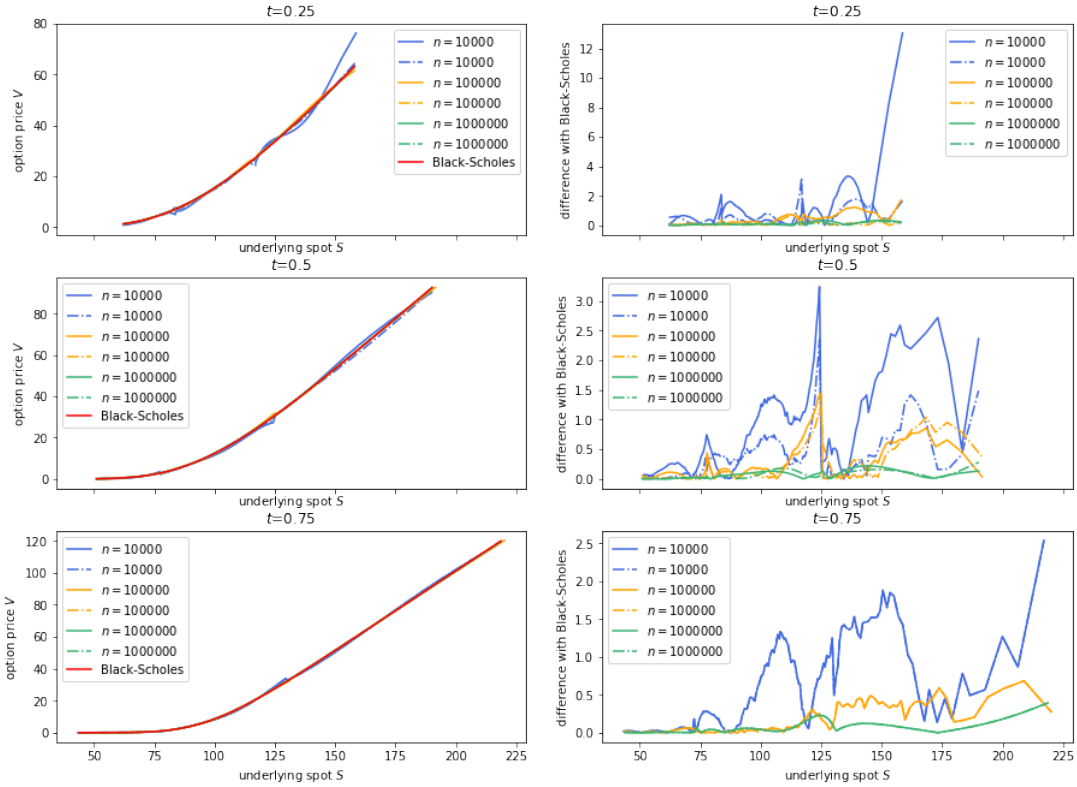

Figure 1.7: LSM for prices under different number of paths with piecewise polynomial regressions versus Black-Scholes prices (left); difference with Black-Scholes (right): discounting payoff approach ('-'); backward recursion approach ('-·')

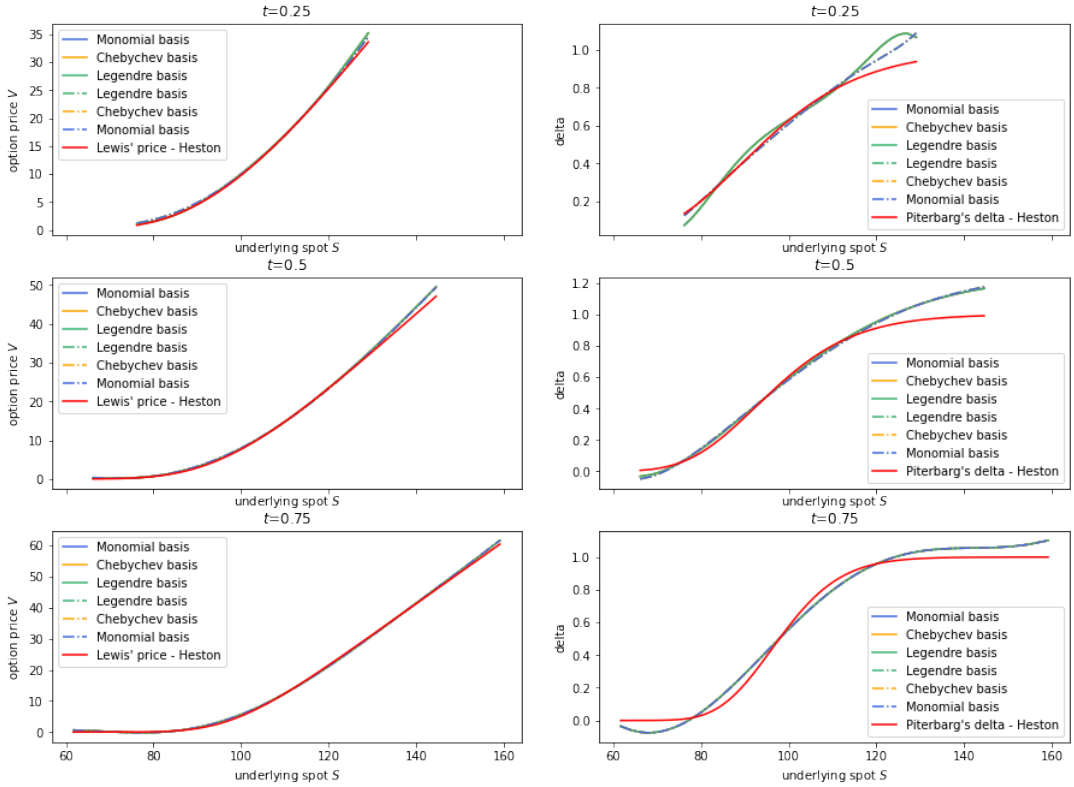

Figure 1.8: LSM for option prices under different basis functions versus Heston prices - Lewis' approximation (left); predicted deltas versus Heston deltas - Piterbarg's approximation (right): discounting payoff approach ('-'); backward recursion approach ('-·')

Chapter 2

Dynamically Controlled Kernel Estimation

We propose a discrete-time pricing and hedging strategy, Dynamically Controlled Kernel Estimation (DCKE), as an improvement of all similar methods in the literature, which is shown to deliver significantly more accurate optimal hedge and option prices via comparisons. The uniqueness of our developed algorithm are as follows:

1. Instead of computing the optimal hedge through direct regressions in both Grau's and Potters' approach (as introduced in next few paragraphs), we propose to work out the hedge according to the variance minimisation solution via pointwise kernal estimation and pathwise derivative estimation, which releases the restriction of basis functions in their algorithms. This perfectly tackles the problem as mentioned as Remark 1.3.2 of LSM, and hence Remark 1.3.1 and 1.3.3.

2. We apply Gaussian Process Regressions to option prices and use the computed optimal hedge as a control variate parameter, which removes the challenges of parameters choosing in Potters' approach. This also improves LSM in terms of Remark 1.3.1 and 1.3.3.

3. We employ quasi-Monte Carlo (QMC) method by replacing geometric Brownian motion with Sobol (quasi-random) sequence, which significantly reduce the number of paths needed for generating reliable sets of solutions, and thus contributes to the computational efficiency and tackles the problem around the edges in Remark 1.3.3 .

The core idea of the Black-Scholes-Merton (BSM) model is that options or other financial derivatives can be priced using the relative value approach to asset pricing. Such approach for options is known as dynamic option replication. It is based on the observation that an option payoff depends only on the price of a stock at expiry of the option. Therefore, if we neglect other sources of uncertainty such as stochastic volatility, the option values at arbitrary times before the expiry should only depend on the stock price. This makes it possible to mimic the option using a simple portfolio made of the underlying stock and cash, which is called the hedge portfolio, dynamically managed by continuously rebalancing its wealth between the stock and cash in a self-financing way. The objective of dynamic replication is to mimic the option using the hedge portfolio as closely as possible, because the portfolio prices are the option prices we aim to get.

Grau (2008)'s Simulation-Based Hedging algorithm relies on these ideas and computes optimal portfolios explicitly in order to obtain prices which have a foundation on a hedging strategy in the physical or real-world measure. The framework can be seen as an extension to the Least Squares Monte Carlo, and the underlying principles for the valuation of exotic options can easily be adapted to this algorithm. It is similar to mean-variance option pricing in incomplete markets, which corresponds to the maximization of a quadratic utility function according to Schweizer (1995), but here we adjust the prices to account for the remaining risk of the hedged position which delivers prices a bank could directly trade on. A similar numerical method to the Simulation-Based Hedging is the Hedged Monte Carlo, which was presented by Potters et al. (2001) and its extension version by Pochart and Bouchaud (2004). However, such methods were not accepted widely due to their convergence properties: the difficulties in finding the right set of

parameters that delivers accurate option prices while computing the prices by regression directly. Instead, Grau (2008) performed regressions to compute the optimal hedge only to preserve the convergence of the estimated prices to the true option values and release its dependency on the choice of basis functions.

Different from these approaches above, we now introduce our unique pricing and hedging framework step by step as follows.

2.1 Discrete-time Black-Scholes-Merton model

To improve LSM, we will revisit the discrete time BSM model to gain further insights into the pricing and hedging framework.

We first restrict our setting to a derivative with a value V and a payoff function depending on a single underlying in order to consider a pricing and hedging framework. The pricing model should be discrete in time because a hedging party can only buy and sell the underlying at discrete times in order to follow a hedging strategy.

2.1.1 Hedge portfolio evaluation

A portfolio at time t_i can be expressed in the form

$$\Pi_{t_i} = B_{t_i} + \phi_{t_i} S_{t_i}, \tag{2.1.1}$$

where B_{t_i} is a bank account value, ϕ_{t_i} represents a position in the option underlying S_{t_i}.

The exact amount of money in the bank account B and thus the portfolio value Π are unknown at the initialisation time t_0 of the derivatives trade, and their distributions will be computed. These values are only known at maturity time t_T, which makes the processes B and Π measurable at t_T. However, as the hedging strategy should be feasible in a real environment, the strategy itself must be measurable at each time step t_i.

The issuer has to pay the payoff of the derivative at maturity, i.e. the trader sells the hedge

$$\phi_{t_T} = 0,$$

and the bank account compensates for the payoff paid to the option holder,

$$B_{t_T} = V_{t_T}.$$

We can find at the maturity, Π is measurable with

$$\Pi_{t_T} = B_{t_T} + \phi_{t_T} S_{t_T} = V_{t_T}.$$

In order to obtain a hedging strategy for the whole life time of the option, we proceed by an induction backward step by step. The bank account at time t_i should be able to compensate for the money required at time $t_i + 1$, i.e.

$$e^{r(t_{i+1}-t_i)} B_{t_i} + \phi_{t_i} S_{t_{i+1}} = B_{t_{i+1}} + \phi_{t_{i+1}} S_{t_{i+1}} \tag{2.1.2}$$

$$\Leftrightarrow B_{t_i} = e^{-r(t_{i+1}-t_i)} \left(B_{t_{i+1}} + \left(\phi_{t_{i+1}} - \phi_{t_i} \right) S_{t_{i+1}} \right), \tag{2.1.3}$$

where $\left(\phi_{t_{i+1}} - \phi_{t_i} \right) S_{t_{i+1}}$ denotes the profit from the position in the underlying and r is the interest rate the issuer receives on the bank account. We can therefore see that B_{t_i} is only measurable at maturity time $t_i = t_T$. Equation (2.1.3) describes a self-financing hedging strategy, and we substitute Equation (2.1.3) into Equation (2.1.1) to derive the value of the hedge portfolio

$$\Pi_{t_i} = e^{-r(t_{i+1}-t_i)} \left(B_{t_{i+1}} + \left(\phi_{t_{i+1}} - \phi_{t_i} \right) S_{t_{i+1}} \right) + \phi_{t_i} S_{t_i} \tag{2.1.4}$$

$$= e^{-r(t_{i+1}-t_i)} \left(\Pi_{t_{i+1}} - \phi_{t_i} \left(S_{t_{i+1}} - e^{r(t_{i+1}-t_i)} S_{t_i} \right) \right) \tag{2.1.5}$$

$$= e^{-r(t_{i+1}-t_i)} \left(\Pi_{t_{i+1}} - \phi_{t_i} \Delta S_{t_i} \right), \tag{2.1.6}$$

with $\Delta S_{t_i} = S_{t_{i+1}} - e^{r(t_{i+1}-t_i)} S_{t_i}$, which would be helpful in formulating the numerical evaluation. Note that the hedge portfolio in Equation (2.1.3) is not deterministic at time t_0 because the bank account B_{t_0} denotes the money required to compensate for the hedging costs including the hedging error. Since the exact path of the underlying is not known at any time t_i, $i < T$, the required amount in the bank account is stochastic.

We then focus on which trading strategy $\phi_{t_i}, i \in \{0, \ldots, T\}$ the issuer should follow. The objective of the hedging party of a derivatives trade is to minimize the risk incurred in the derivative, and the risk can be measured in several different ways and hence lead to various strategies.

2.1.2 Hedging strategies

To determine a specific hedging strategy, there are several objectives of an issuer. The issuer is usually a bank constituting of a derivatives trading department and an investment department. The investment department actively takes risks and usually balances the mean and variance of its returns while the objective of a derivatives hedge should be to minimize a symmetric risk measure.

The key strategy we are going to introduce, in Section 2.1.3, is variance minimisation, which has a property that the more often a hedge is conducted, the less should be the influence of the drift on the option's price, being proved in Proposition 2.1.1. This is the hedging strategy that normally applied under the risk-neutral condition. Due to the difficulties in identifying the drift of the underlying, given by

$$\mu = \frac{1}{(t_{i+1} - t_i)} \cdot \log \frac{\mathrm{E}\left[f_{t_{i+1}}\left(S_{t_i}\right)\right]}{S_{t_i}},$$

with $f_{t_{i+1}}$ the transition function such that $S_{t_{i+1}} = f_{t_{i+1}}\left(S_{t_i}\right)$, $i \in \{0, \ldots, T\}$, we try to minimise the dependency of our strategy on it. And the pricing equations of the traditional Black-Scholes model do not contain the drift term and hence is also a qualified candidate model.

Moreover, it is important to look at the optimisation procedure itself. For an issuer who have already followed a risk-minimizing strategy and lost money due to hedging errors during that period, should he change the hedging portfolio in order to compensate for the errors? The answer is no, because the derivatives department should not invest in order to make gains on the position in the underlying, but to minimize future risks. Consequently, only future risk should be hedged, i.e., a hedge should be local in time rather than global.

2.1.3 Variance minimisation

Variance is a very simple risk measure, which corresponds to a quadratic utility of the issuer. We should pursue a forward global risk minimisation rather than a purely global one according to our discussion at the end of Section 2.1.2, since the purely global version correct errors from the past by investing. The forward global risk minimisation only reduces future hedging errors, based on the assumption that the future strategy does the same. In this approach, the minimisation decomposes into a minimisation of the risky capital costs for each time step. By substituting in Equation (2.1.6), the optimal fraction ϕ_{t_i} of the hedge instrument S is given by

$$\{\phi_{t_i}\} = \arg\min_{\phi_{t_i}} \left(\mathrm{Var}\left[\Pi_{t_i} \mid S_{t_i}, t_i\right]\right) \tag{2.1.7}$$

$$= \arg\min_{\phi_{t_i}} \left(\mathrm{Var}\left[\Pi_{t_{i+1}} - \phi_{t_i} \Delta S_{t_i} \mid S_{t_i}, t_i\right]\right), \tag{2.1.8}$$

with $i = 0, \ldots, T-1$, which is similar to the quadratic hedging or local variance minimisation in discrete time as described by Follmer and Schweizer[47] and Schweizer[104]. Here, basically all future values of the stochastic variables S and V are reduced to one stochastic variable $\Pi_{t_{i+1}}$. Then, Equation (2.1.8) can be solved by setting $\frac{\partial \mathrm{Var}\left[\Pi_{t_{i+1}} - \phi_{t_i} \Delta S_{t_i} | S_{t_i}, t_i\right]}{\partial \phi_{t_i}} = 0$ and obtain the optimal hedging strategy

$$\phi^*_{t_i} = \frac{\operatorname{cov}\left[\Pi_{t_{i+1}}, \Delta S_{t_i} \mid S_{t_i}, t_i\right]}{\operatorname{Var}\left[\Delta S_{t_i} \mid S_{t_i}, t_i\right]}, \tag{2.1.9}$$

$$= \frac{\operatorname{cov}\left[\Pi_{t_{i+1}}, S_{t_{i+1}} \mid S_{t_i}, t_i\right]}{\operatorname{Var}\left[S_{t_{i+1}} \mid S_{t_i}, t_i\right]}, \tag{2.1.10}$$

where the last equality follows in the risk-neutral world. The solution of the optimal hedging strategy involves one-step expectations of quantities at time $t+1$, conditional on time t, and we are first going to show when the lag between two time steps is small enough, this estimator actually converges to the continuous-time BSM delta according to Halperin (2020), and then several ways are provided in Section 2.2 to compute these conditional expectations.

Proposition 2.1.1. *The newly proposed estimator for optimal hedge and hedge portfolio converges to Black-Scholes deltas and option prices respectively when the lag between two time steps tends to zero.*

Proof. We start with the notion of a fair option price $\hat{C}_{t_i}$ defined as the expected value of the hedge portfolio Π_t at time t:

$$\hat{C}_{t_i} = \mathrm{E}\left[\Pi_{t_i} \mid S_{t_i}, t_i\right].$$

Substituting Equation (2.1.6) and by Tower Property, we have

$$\hat{C}_{t_i} = \mathrm{E}\left[e^{-r(t_{i+1}-t_i)}\Pi_{t_{i+1}} \mid S_{t_i}, t_i\right] - \phi_{t_i}\left(S_{t_i}\right)\mathrm{E}\left[\Delta S_t \mid S_{t_i}, t_i\right] \tag{2.1.11}$$

$$= \mathrm{E}\left[e^{-r(t_{i+1}-t_i)}\mathrm{E}\left[\Pi_{t_{i+1}} \mid S_{t_{i+1}}, t_{i+1}\right] \mid \Pi_{t_i}, t_i\right] - \phi_{t_i}\left(S_{t_i}\right)\mathrm{E}\left[\Delta S_{t_i} \mid S_{t_i}, t_i\right] \tag{2.1.12}$$

$$= \mathrm{E}\left[e^{-r(t_{i+1}-t_i)}\hat{C}_{t_{i+1}} \mid S_{t_i}, t_i\right] - \phi_{t_i}\left(S_{t_i}\right)\mathrm{E}\left[\Delta S_{t_i} \mid S_{t_i}, t_i\right], \tag{2.1.13}$$

where $t = 1, \ldots, T-1$.

Note that we can similarly use the tower law of conditional expectations to express the optimal hedge in terms of $\hat{C}_{t+1}$ instead of Π_{t+1}, i.e. Equation (2.1.9) is equivalent to

$$\phi^\star_{t_i}\left(S_{t_i}\right) = \frac{\operatorname{Cov}\left(\hat{C}_{t_{i+1}}, \Delta S_{t_i} \mid S_{t_i}, t_i\right)}{\operatorname{Var}\left(\Delta S_{t_i} \mid S_{t_i}, t_i\right)}. \tag{2.1.14}$$

Recall that the BSM model dynamics under the physical measure P is described by a continuous-time Geometric Brownian motion with a drift μ and volatility σ:

$$\frac{dS_{t_i}}{S_{t_i}} = \mu dt + \sigma dW_{t_i},$$

with W_{t_i} a standard Brownian motion.

By applying the first-order Taylor expansion

$$\hat{C}_{t_{i+1}} = C_{t_i} + \frac{\partial C_{t_i}}{\partial S_{t_i}}\Delta S_{t_i} + \mathcal{O}(\Delta t_i)$$

to Equation (2.1.14), we get

$$\phi^{BS}_{t_i}\left(S_{t_i}\right) = \lim_{\Delta t_i \to 0} \phi^\star_{t_i}\left(S_{t_i}\right) = \frac{\partial C_{t_i}}{\partial S_{t_i}},$$

which is the optimal hedge in the continuous-time BSM model.

To further verify the formula for hedge portfolio also converge to the continuous-time model, we first take the limit of the second term in Equation (2.1.13):

$$\lim_{\Delta t_i \to 0} \phi_{t_i}\left(S_{t_i}\right)\mathrm{E}\left[\Delta S_{t_i} \mid S_{t_i}, t_i\right] = \lim_{dt_i \to 0} \phi^{BS}_t S_{t_i}(\mu - r)dt = \lim_{dt \to 0}(\mu - r)S_{t_i}\frac{\partial C_{t_i}}{\partial S_{t_i}}dt.$$

We then apply the second-order Taylor expansion to Equation (2.1.13), and get

$$\hat{C}_{t_{i+1}} = C_{t_i} + \frac{\partial C_{t_i}}{\partial t_i} dt_i + \frac{\partial C_{t_i}}{\partial S_{t_i}} dS_{t_i} + \frac{1}{2} \frac{\partial^2 C_{t_i}}{\partial S_{t_i}^2} (dS_{t_i})^2 + \dots \tag{2.1.15}$$

$$= C_{t_i} + \frac{\partial C_{t_i}}{\partial t_i} dt_i + \frac{\partial C_{t_i}}{\partial S_{t_i}} S_t (\mu dt_i + \sigma dW_{t_i}) + \frac{1}{2} \frac{\partial^2 C_{t_i}}{\partial S_{t_i}^2} S_{t_i}^2 \left(\sigma^2 dW_{t_i}^2 + 2\mu\sigma dW_{t_i} dt_i \right) + \mathcal{O}\left(dt_i^2 \right) \tag{2.1.16}$$

We substitute the expansions of these two terms back to Equation (2.1.13), and based on the fact that $\mathrm{E}\left[dW_{t_i}\right] = 0$ and $\mathrm{E}\left[dW_{t_i}^2\right] = dt_i$, we find Equation (2.1.13) finally becomes the celebrated Black-Scholes-Merton equation in the limit $dt_i \to 0$:

$$\frac{\partial C_{t_i}}{\partial t_i} + r S_{t_i} \frac{\partial C_{t_i}}{\partial S_{t_i}} + \frac{1}{2} \sigma^2 S_{t_i}^2 \frac{\partial^2 C_{t_i}}{\partial S_{t_i}^2} - r C_{t_i} = 0$$

□

2.2 Dynamically Controlled Kernel Estimation

Different from either Potters et al. (2001), who operated Least Squares regression as described in Section 1 on the optimal hedge and the hedge portfolio right afterward, or Grau (2008), who also regressed directly onto the optimal hedge and then obtained portfolio values by backward recursion according to Equation (2.1.1), we adopt a new approach here.

First of all, we estimate the optimal hedge for each time step by two methods, namely pathwise derivative estimation and variance minimisation solution as shown in Equation (2.1.9), both with kernel estimation, which gives conditional expectations. Then, we discount the payoff back once per time step and again smooth the price by the kernels. Additionally, we replace the parametric models with polynomial bases in Potters and Grau's approaches by the non-parametric Gaussian process regression, which is still a form of supervised learning, but the training data are harnessed in a subtler way. The key to the regression in our algorithm is the inclusion of control variates, the optimal parameter of which could be shown identically to the optimal hedge in discrete-time BSM model. Therefore, the two methods for estimating delta not only give the optimal hedge, but are also vital to the price estimation.

2.2.1 Kernel regression

We apply a non-parametric technique, kernel regression, here because it gives pointwise estimates of the conditional expectations. Its objective is to find a non-linear relation between each pair of random variables. To compute the discrete version of the optimal hedge, Equation (2.1.9) could be written as

$$\phi_{t_i} = \frac{\mathrm{cov}\left[\Pi_{t_{i+1}}, S_{t_{i+1}} \mid S_{t_i}, t_i\right]}{\mathrm{Var}\left[S_{t_{i+1}} \mid S_{t_i}, t_i\right]} \tag{2.2.1}$$

$$= \frac{\mathrm{E}\left[(\Pi_{t_{i+1}} - \mathrm{E}[\Pi_{t_{i+1}}])(S_{t_{i+1}} - \mathrm{E}[S_{t_{i+1}}]) \mid S_{t_i}, t_i\right]}{\mathrm{E}\left[(S_{t_{i+1}} - \mathrm{E}[S_{t_{i+1}}])^2 \mid S_{t_i}, t_i\right]}, \tag{2.2.2}$$

which consists of several terms of conditional expectations. And the expectation in formula 2.2.10 can just be treated as being conditional on S_{t_i} and t_i implicitly. The calculation of conditional expectations as follows applies.

Definition 2.2.1. In any non-parametric regression, the conditional expectation of a variable $\boldsymbol{Y}$ relative to a variable $\boldsymbol{X}$ could be written as:

$$\mathrm{E}[\boldsymbol{Y} \mid \boldsymbol{X}] = m(\boldsymbol{X}; \boldsymbol{Y}),$$

where m is some function representing a kernel estimator as below.

2.2.1.1 Nadaraya-Watson versus Locally Linear estimator

Two estimators, namely Nadaraya-Watson and Locally Linear estimator are commonly used kernel estimators.

Nadaraya (1964) and Watson (1964) proposed to estimate the function m as a locally weighted average, using a kernel as a weighting function. The Nadaraya-Watson estimator is given by:

$$\widehat{m}_h^{NW}(\boldsymbol{x};\boldsymbol{y}) = \frac{\sum_{j=1}^{n}\left\{K_h\left(\boldsymbol{x}-\boldsymbol{x}_j\right)\right\}\boldsymbol{y}_j}{\sum_{j=1}^{n}\left\{K_h\left(\boldsymbol{x}-\boldsymbol{x}_j\right)\right\}}, \tag{2.2.3}$$

where K_h is the skewed kernel function depending on a smoothing parameter, bandwidth h with $K_h(\boldsymbol{x}) = \frac{1}{h^z}K\left(\frac{\boldsymbol{x}}{h}\right)$ and d is the dimensionality of $\boldsymbol{x}$. The kernel function K is normally chosen as the radial basis function $K(\boldsymbol{x}) = e^{-\frac{\|\boldsymbol{x}\|^2}{2h^2}}$, which is also known as Gaussian kernel.

The derivation of Equation (2.2.3) is shown in Appendix A.2, and more details regarding the kernel function and the bandwidth parameter could be found in Section 2.2.1.3, where we formally introduce kernel density estimation.

Hence, in an example of a single underlying, Equation (2.2.2) can be further written as

$$\hat{\phi}_{t_i}^{k} = \frac{\frac{\sum_{j=1}^{n}\left\{K_h\left(x_k-S_{t_i}^{j}\right)\right\}(\Pi_{t_{i+1}}^{j}-\mathrm{E}[\Pi_{t_{i+1}}^{j}|S_{t_i},t_i])(S_{t_{i+1}}^{j}-\mathrm{E}[S_{t_{i+1}}^{j}|S_{t_i},t_i])}{\sum_{j=1}^{n}\left\{K_h\left(x_k-S_{t_i}^{j}\right)\right\}}}{\frac{\sum_{j=1}^{n}\left\{K_h\left(x_k-S_{t_i}^{j}\right)\right\}(\Pi_{t_{i+1}}^{j}-\mathrm{E}[\Pi_{t_{i+1}}^{j}|S_{t_i},t_i]))^2}{\sum_{j=1}^{n}\left\{K_h\left(x_k-S_{t_i}^{j}\right)\right\}}} \tag{2.2.4}$$

$$= \frac{\hat{m}_h(x_k,S_{t_i};(\Pi_{t_{i+1}}-\mathrm{E}[\Pi_{t_{i+1}}\mid S_{t_i},t_i])(S_{t_{i+1}}-\mathrm{E}[S_{t_{i+1}}\mid S_{t_i},t_i]))}{\hat{m}_h(x_k,S_{t_i};(\Pi_{t_{i+1}}-\mathrm{E}[\Pi_{t_{i+1}}\mid S_{t_i},t_i]))^2)} \tag{2.2.5}$$

$$= \frac{\hat{m}_h(x_k,S_{t_i};(\Pi_{t_{i+1}}-\hat{m}_h(x_k,S_{t_i};\Pi_{t_{i+1}})(S_{t_{i+1}}-\hat{m}_h(x_k,S_{t_i};S_{t_{i+1}})))}{\hat{m}_h(x_k,S_{t_i};(\Pi_{t_{i+1}}-\hat{m}_h(x_k,S_{t_i};\Pi_{t_{i+1}})))^2)}, \tag{2.2.6}$$

where $j = 1,2,\ldots,n$ is each simulated path and $k = 1,2,\ldots,u$ is each meshed point of underlying at a certain time step t_i.

Another popular estimator is the locally linear kernel weighted estimator (known as Locally Linear estimator) proposed by Fan (1992) and Ruppert and Wand (1994), which simultaneously corrects for both boundary-based and curvature-based finite-sample bias. This estimator is also a kernel weighted sum, given by

$$\widehat{m}_h^{LL}(\boldsymbol{x};\boldsymbol{y}) = \frac{1}{n}\frac{\hat{s}_2(\boldsymbol{x};h)-\hat{s}_1(\boldsymbol{x};h)\left(\boldsymbol{x}_j-\boldsymbol{x}\right)\boldsymbol{y}_j}{\hat{s}_2(\boldsymbol{x};h)\,\hat{s}_0(\boldsymbol{x};h)-\hat{s}_1(\boldsymbol{x};h)^2}K_h\left(\boldsymbol{x}-\boldsymbol{x}_j\right), \tag{2.2.7}$$

where $\hat{s}_r(\boldsymbol{x};h) := \frac{1}{n}\sum_{j=1}^{n}\left(\boldsymbol{x}_j-\boldsymbol{x}\right)^r K_h\left(\boldsymbol{x}-\boldsymbol{x}_j\right)$ with $r = 0,1,2$. Similarly, Equation (2.2.2) can be expressed in terms of $\widehat{m}_h^{LL}$.

We note that given standard conditions regarding the kernel, bandwidth and data generating process, both estimators are consistent according to Härdle (1990, pg 29) and Fan (1992). For the approximate bias and variance of each estimator for second-order kernels, we refer to Table 2.1.

Estimator	**Bias**	**Variance**
Nadaraya-Watson	$h^2\left(\frac{1}{2}m_h{''}(\boldsymbol{x})+\frac{m_h{'}\boldsymbol{x})f'(\boldsymbol{x})}{f(\boldsymbol{x})}\right)\int z^2k_h(\boldsymbol{z})d\boldsymbol{z}$	$\frac{\sigma^2(\boldsymbol{x})}{f(\boldsymbol{x})nh}\int {k_h}^2(\boldsymbol{z})d\boldsymbol{z}$
Locally Linear	$h^2\left(\frac{1}{2}m_h{''}(\boldsymbol{x})\right)\int z^2k_h(\boldsymbol{z})d\boldsymbol{z}$	$\frac{\sigma^2(\boldsymbol{x})}{f(\boldsymbol{x})nh}\int k_h^2(\boldsymbol{z})d\boldsymbol{z}$

Table 2.1: Pointwise bias up to $\mathcal{O}(h^2)$ and variance of bivariate NW and LL estimators with second-order kernels

We can observe both estimators are biased in finite samples. They both suffer from 'curvature effects' reflecting by the term $m_h{''}(x)$. The additional bias term of Nadaraya-Watson estimator leads to 'tail' problems because the denominator $f(\boldsymbol{x})$ can vanish at the boundary of support for

the regressor, and the whole term would blow up. Nevertheless, two estimators give the same variance, and consequently, Nadaraya-Watson estimator associates with greater mean squared errors compared to Locally Linear estimator.

2.2.1.2 Kernelised pathwise derivative estimation

Pathwise estimation is an alternative approach for the sensitivity analysis. To provide the derivative of the payoff, an automatic differentiating algorithm is required. And pathwise derivative estimation generally gives less estimation errors in estimating deltas than the others such as likelihood ratio estimation, and especially useful in multi-underlying settings as we will see in Chapter 3.

Assume $Y(\theta)$ is differentiable with respect to θ. An unbiased estimator of $\alpha'(\theta)$ is given by

$$\alpha'(\theta) = \frac{\partial}{\partial\theta}\mathrm{E}[Y(\theta)] = \mathrm{E}\left[\frac{\partial Y(\theta)}{\partial\theta}\right]$$

given the interchangeability of differentiation and expectation. We then approximate $\mathrm{E}\left[Y'(\theta)\right]$ by $\alpha'(\theta)$, which is called the pathwise derivative estimation.

This technique is applicable for most types of options with continuous derivatives, and we find out the Black-Scholes delta via pathwise estimation as an example. In the case of European Call option, the Call price at time t_i is given by $C_{t_i} = \mathrm{E}[Y^j_{t_i}]$, where $Y^j_{t_i} = e^{-r(t_T-t_i)}\left(S^j_{t_T} - K\right)^+$ and $S^j_{t_T} = S^j_{t_i} e^{\left(r-\frac{1}{2}\sigma^2\right)(t_T-t_i)+\sigma\sqrt{t_T-t_i}Z}$ with $j = 1, 2, \ldots, n$ and $Z \sim N(0,1)$. Black-Scholes Call option delta is represented as $\frac{dC_{t_i}}{dS_{t_i}}$, and we obtain

$$\frac{dC_{t_i}}{dS_{t_i}} = \mathrm{E}\left[\frac{dY^j_{t_i}}{dS_{t_i}}\right] = \mathrm{E}\left[\frac{dY^j_{t_i}}{dS^j_{t_T}}\frac{dS^j_{t_T}}{dS_{t_i}}\right] \tag{2.2.8}$$

$$= \mathrm{E}\left[e^{-r(t_T-t_i)}\frac{d}{dS^j_{t_T}}(S^j_{t_T}-K)^+\frac{dS^j_{t_T}}{dS_{t_i}}\right] \tag{2.2.9}$$

$$= \mathrm{E}\left[e^{-r(t_T-t_i)}\mathbb{1}_{\left\{S^j_{t_T}>K\right\}}\frac{S^j_{t_T}}{S_{t_i}}\right]. \tag{2.2.10}$$

Applying kernel regression, we get the delta at each meshed point k with $k = 1, 2, \ldots, u$ of underlying at time step t_i:

$$\hat{\phi}^k_{t_i} = \hat{m}_h\left(x_k, S_{t_i}; e^{-r(t_T-t_i)}\mathbb{1}_{\left\{S_{t_T}>K\right\}}\frac{S_{t_T}}{S_{t_i}}\right), \tag{2.2.11}$$

with $\hat{m}_h$ defined in the same form as Equation (2.2.3) or (2.2.7).

From Proposition 2.1.1, we can get while the time lag is small enough, Equation (2.1.9) converges to Black-Scholes-Merton deltas, which is what we just approximate using Equation (2.2.10). Therefore, both Equation (2.2.10) and 2.1.9 give predictions to Black-Scholes deltas, and we have estimated both of them via kernel regression, a way of computing the conditional expectations we are mainly interested in.

2.2.1.3 Kernel density estimation

Kernel density estimation (KDE), as mentioned in the derivation of Equation (2.2.3) in Appendix A.2, is a non-parametric estimation method which constructs an estimate, based on observed data, of an unobservable underlying probability density function via a kernel. It refines the simplest density estimation histogram by endowing with properties such as smoothness or continuity and removing the dependency of density on the end points of the bins.

Let $(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N)$ be a sequence of independent and identically distributed sample drawn from some distribution with an unknown density f, the contribution of data point x_i to the estimate at some point x_j depends on how apart these two points are. The extent of this contribution

is dependent upon the shape of the kernel function adopted and the width (bandwidth) accorded to it. Cacoullos (1966) and Epanechnikov (1969) proposed that in the multivariate case, the estimated density at any point $\boldsymbol{x}$ is given by

$$\widehat{f}(\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N}\left\{K_h\left(\boldsymbol{x}-\boldsymbol{x}_i\right)\right\} \tag{2.2.12}$$

$$= \frac{1}{Nh^d}\sum_{i=1}^{N}\left\{K\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right)\right\}, \tag{2.2.13}$$

where d is the dimensionality of $\boldsymbol{x}$. For any point $\boldsymbol{x}$ in the variable space, each value in the training sequence $\boldsymbol{x}_i$, $i = 1,\ldots,N$ contributes to the estimate, and the contribution is weighted by its distance from $\boldsymbol{x}$, given by $\boldsymbol{x}-\boldsymbol{x}_i$.

- Kernel function

  A kernel function is symmetric, i.e. satisfies $K(\boldsymbol{x}) = K(-\boldsymbol{x})$, and $\int_{\boldsymbol{x}} K(\boldsymbol{x})d\boldsymbol{x} = 1$ (or $\boldsymbol{\Sigma}_{\boldsymbol{x}}K(\boldsymbol{x}) = 1$ if $\boldsymbol{x}$ is from a discrete space). Commonly used kernel functions are uniform, triangular, biweight and Gaussian.

As mentioned in Section 2.2.1.1, we choose Gaussian kernel function, and we now derive the kernel estimator Equation (2.2.12) in the multivariate setting.

In this case, the Gaussian kernel is written as $K \sim N(0,\boldsymbol{\Sigma})$, and the kernel estimator is in the form

$$\hat{f}(\boldsymbol{x}) = \frac{1}{N(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}}\sum_{i=1}^{N}\exp\left(-\frac{1}{2}\left(\boldsymbol{x}-\boldsymbol{x}_i\right)^T\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{x}-\boldsymbol{x}_i\right)\right).$$

To separate the 'size' and the 'orientation' of $\boldsymbol{\Sigma}$, we write $\boldsymbol{\Sigma} = h^2\boldsymbol{A}$, where $|\boldsymbol{A}| = 1$. Thus, the size of $\boldsymbol{\Sigma}$ is $\left|h^2\boldsymbol{A}\right| = h^{2d}$, and the Gaussian kernel estimate becomes

$$\begin{aligned}\hat{f}(\boldsymbol{x}) &= \frac{1}{N(2\pi)^{d/2}h^d}\sum_{i=1}^{N}\exp\left(-\frac{1}{2}\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right)^T\boldsymbol{A}^{-1}\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right)\right)\\ &= \frac{1}{N(2\pi)^{d/2}h^d}\sum_{i=1}^{N}\exp\left(-\frac{1}{2}\frac{\left(\boldsymbol{A}^{-1/2}\left(\boldsymbol{x}-\boldsymbol{x}_i\right)\right)^T}{h}\frac{\left(\boldsymbol{A}^{-1/2}\left(\boldsymbol{x}-\boldsymbol{x}_i\right)\right)}{h}\right).\end{aligned}$$

The last step follows for the reason that since $\boldsymbol{A}$ is a symmetric and positive-definite matrix, the symmetric and positive-definite square-root matrix $\boldsymbol{A}^{-1/2}$ exists. Now we find Equation 2.2.14 is equivalent to rotate the data by the transformation $\boldsymbol{A}^{-1/2}$ and apply the kernel that follows $N\left(0,\boldsymbol{I}_d\right)$. In this transformed space, the kernel estimate becomes

$$\hat{f}(\boldsymbol{x}) = \frac{1}{N(2\pi)^{d/2}h^d}\sum_{i=1}^{N}\exp\left(-\frac{1}{2}\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right)^T\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right)\right) \tag{2.2.14}$$

$$= \frac{1}{N}\sum_{i=1}^{N}K_h\left(\boldsymbol{x}-\boldsymbol{x}_i\right), \tag{2.2.15}$$

which is in the same form as Equation (2.2.12), with $K_h(\boldsymbol{x}) = \frac{1}{h^d}K(\boldsymbol{x}h) = \prod_{k=1}^{d}\phi\left(x^{(k)} \mid 0,h\right)$.

Hence, it is easier to with transformed data and using either the normal kernel or, more generally, a product kernel, possibly with different smoothing parameter, h_k, in the kth direction:

$$\hat{f}(\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N}\left(\prod_{k=1}^{d}K_{h_k}\left(x^{(k)}-x_i^{(k)}\right)\right). \tag{2.2.16}$$

- Bandwidth parameter

  The bandwidth of the kernel exhibits a strong influence on the resulting estimate, and there is always a variance-bias trade-off. In the limit $h \to 0$, the estimate is a sum of N delta functions centered at the coordinates of analyzed samples with no smoothing (reduce the bias but increase variance), while in the other extreme case $h \to \infty$, the estimate retains the shape of the used kernel, centered on the mean of the samples and is smoothed completely (decrease variance but introduces bias).

  According to Silverman (1986), the optimal h for Gaussian distributed data by Silverman's rule of thumb in the univariate case is

$$h^* = 0.9 * \min\left(\hat{\sigma}, \frac{IQR}{1.34}\right) * n^{-\frac{1}{5}}, \tag{2.2.17}$$

  where $\hat{\sigma}$ is the standard deviation of the samples, IQR is the interquartile range and n is the sample size.

  The multivariate version of h is surprisingly simple. Assuming a normal product kernel and a true normal density with $\Sigma = I_z$, then $h^* = n^{-\frac{1}{z+4}}$ for the normal kernel or

$$h_k^* = \hat{\sigma}_k n^{-\frac{1}{z+4}}$$

  for the general normal product estimator in Equation (2.2.16). For other choices of kernel, we can refer to Terrell and Scott (1992).

In practice, a single bandwidth may not be optimal for all regions of the domain. For instance, it may over-smooth features in regions with high density and simultaneously under-smooth regions in the 'tails' of the distribution. We therefore introduce the *balloon estimator*, which attempts to address this issue by allowing the bandwidth to change across the domain of the PDF based on the evaluation point $\boldsymbol{x}$, and adopt it in later experiment. The balloon estimator was introduced in the form of the kth nearest neighbor estimator, and can be expressed as Equation (2.2.12) by setting $h(\boldsymbol{x}) = d_k(\boldsymbol{x})$, where $d_k(\boldsymbol{x})$ returns the distance to the kth nearest data point to $\boldsymbol{x}$.

2.2.1.4 Practical experiments

We keep the same general features as in Table 1.5, and we use the same meshed points among 1% to 99% percentiles of the original underlying at each time step as in Chapter 1. Referring to the last Chapter, we first focus on the outperformed discounting payoff approach in the experiment, and will later compare it with the backward recursion approach in Section 2.2.5.

We employ kernel estimation and firstly investigate whether the raw weighted average price in Equation (2.2.3) or Equation (2.2.7) could be used to estimate the true option prices directly.

Under both kernel estimators $\hat{m}_h^{NW}$ and $\hat{m}_h^{LL}$, we adopt both a fixed bandwidth and variable bandwidths inspired by the balloon estimator. We skew the bandwidth at each meshed underlying by a ratio of the maximal kernel size to the local kernel size under the fixed kernel, with pseudo-code in Algorithm 2. Note that such ratios are capped from below by a constant, which controls the extent of smoothing at the 'tails', and the choice of such cap depends on the number of paths used for simulation, where large numbers of paths are associated with smaller caps. Usually, due to the variance-bias trade-off, a small cap, which makes a variable kernel more similar to a fixed kernel, would give accurate estimates at the 'tails', but are highly biased with violent fluctuations, while a large cap smooths the estimates at the 'tails' perfectly but introduces more biases. Therefore, we select different caps according to different number of paths and the trade-off between variance and biased. As a result, we end up with different form of kernel in each scenario, even for pricing and hedging at the same time step.

We notice that different from utilising polynomial bases, we get results converging to exact Black-Scholes solutions when the number of paths is adjusted to be greater. Figure 2.1 shows when 50,000 paths are simulated, the raw kernel estimates under both kernels, either fixed or variable, all show promising convergence. NW and LL with fixed kernel give indistinguishable results, but LL with variable kernel clearly outperforms the others, especially NW estimator with variable kernel, and shows particular great performance at the 'tails'.

Compared to fixed bandwidths, variable bandwidths shows greater flexibility in capturing the curvatures in low-density regions. It is worth noted that due to variance-bias trade-off, when

we select a higher order bandwidth, the estimates around the 'tails' tend to be smoother but with more deviations. However, LL does not suffer from this problem. As in Figure 2.1, even if the large enough bandwidth leads to significant estimation errors, even exceed the error by fixed kernels, around the ends of underlying spots using NW variable kernel, LL variable kernel successfully controls the risk of blowing up at the 'tails' and are hence preferred.

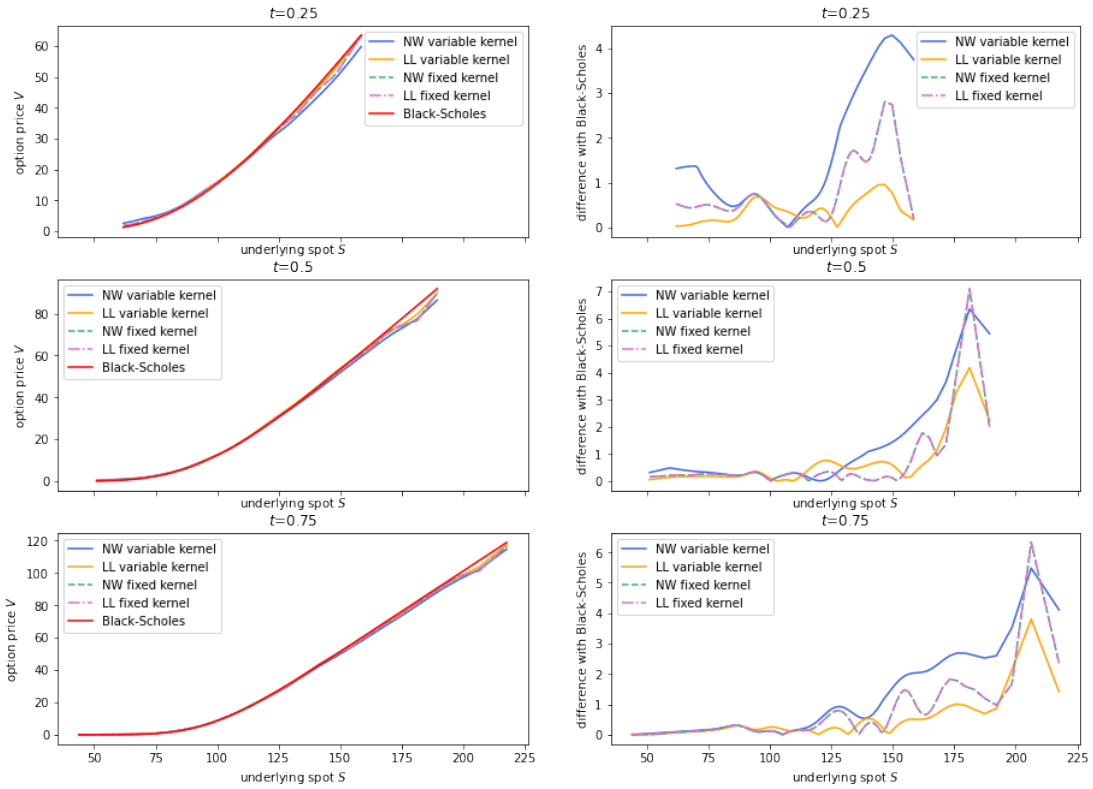

Figure 2.1: Raw kernel estimates of option prices without control variates using different variable and fixed kernel estimators versus Black-Scholes prices (left); difference with Black-Scholes (right)

We now focus onto Figure 2.2 and 2.3, the estimates of deltas by kernel regression on two different approaches, the variance minimisation strategy in Equation (2.1.9) and pathwise estimation, respectively. First of all, we obtain exactly the same conclusions as the raw kernel estimates above, LL estimator deals with estimations around the 'tails' excellently, especially combining with balloon estimators with the allowance of varying bandwidths. Additionally, pathwise deltas estimates tend to associate with less errors than the deltas obtained by variance minimisation solution compared with Black-Scholes deltas. And we can also find Local Linear estimator is particularly robust in estimating the pathwise delta, with smaller errors than any NW estimator throughout almost the whole range of underlying spots in all time steps.

2.2.2 Control variates

We now apply the control variate (CV) technique to our algorithm because it provides a effective variance reduction in a simple theoretical framework, and we benefit from CV since the optimal hedge in Equation (2.1.9) is in the same form as the optimal CV parameter. The method, first introduced to option pricing by Boyle (1977), takes advantage of random variables with known expected value that are positively correlated with the variable under consideration.

Let Y be a random variable whose mean is to be determined through simulation and X a random variable with known mean $\mathrm{E}[X]$. For each trial the outcome of X_i is calculated along with the output of Y_i, and we further suppose the pairs $(X_i, Y_i), i = 1, \ldots, n$ are i.i.d.. Then for a fixed parameter β, the control variate estimator $\bar{Y}_{CV}$ of $E[Y]$ is defined by

$$\bar{Y}_{CV}(\beta) = \bar{Y} - \beta\,(\bar{X} - \mathrm{E}[X]) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \beta(X_i - \mathrm{E}[X])), \tag{2.2.18}$$

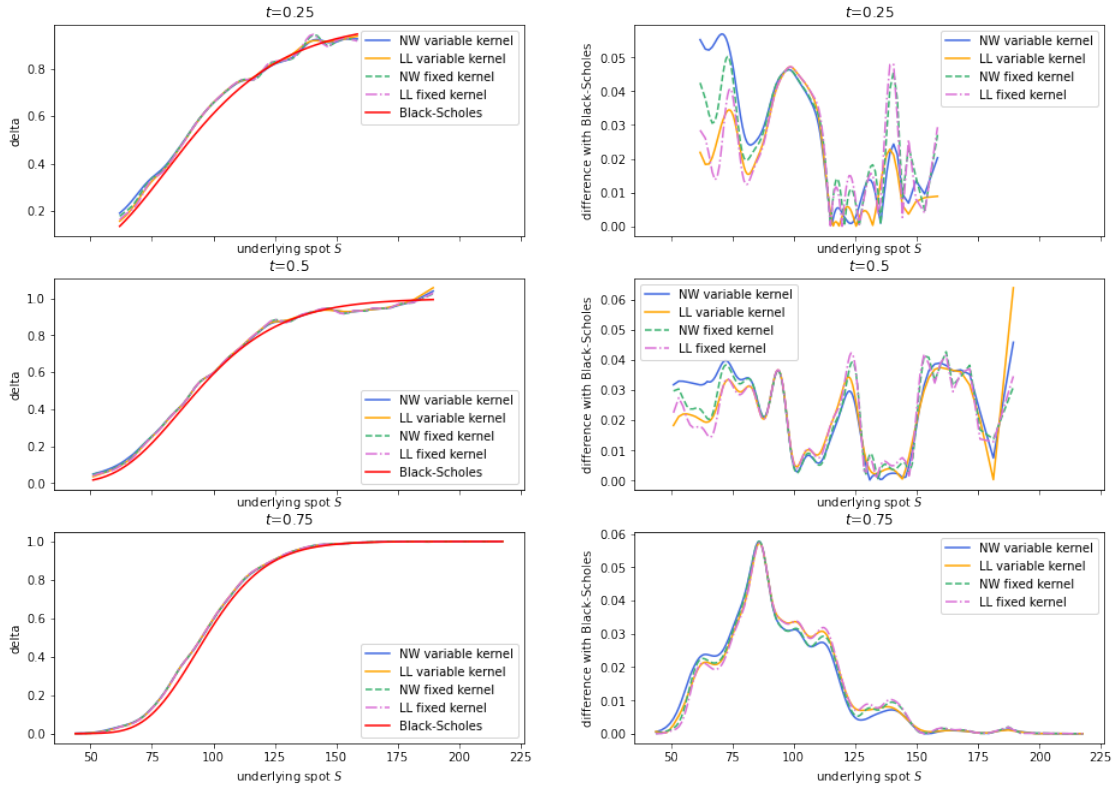

Figure 2.2: Variance minimisation delta using different variable and fixed kernel estimators versus Black-Scholes prices (left); difference with Black-Scholes (right)

where the observed error $\bar{X} - \mathrm{E}[X]$ serves as a control in estimating $\mathbb{Y}$.

Lemma 2.2.2. *The control variate estimator is unbiased and consistent.*

Proof. The unbiasedness and consistency are given as follows:

$$\mathrm{E}\left[\bar{Y}_{CV}\right] = \mathrm{E}[\bar{Y} - \beta\left(\bar{X} - \mathrm{E}[X]\right)] = \mathrm{E}\left[\frac{1}{n}\sum_{i=1}^{n} Y_i\right] = \mathrm{E}[Y],$$

and with probability 1,

$$\lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} Y_{CV}(i) = \lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} \left(Y_i - \beta\left(X_i - \mathrm{E}[X]\right)\right) = \mathrm{E}\left[Y - \beta(X - \mathrm{E}[X])\right] = \mathrm{E}[Y].$$

□

The resulting variance for the control variate estimator is

$$\mathrm{Var}\left(\bar{Y}_{CV}(\beta)\right) = \frac{1}{n}\mathrm{Var}(\bar{Y}) + \beta^2 \mathrm{Var}\left(\bar{X} - \mathrm{E}[X]\right) - 2\beta\,\mathrm{cov}(\bar{Y}, \bar{X} - \mathrm{E}[X]) \tag{2.2.19}$$

$$= \frac{1}{n}\left(\sigma_Y^2 + \beta^2\sigma_X^2 - 2\beta\rho_{XY}\sigma_X\sigma_Y\right), \tag{2.2.20}$$

which indicates that the control variate estimator $\bar{Y}_{CV}$ has a lower variance than $\bar{Y}$ if $\beta^2\sigma_X < 2\rho_{XY}\sigma_Y$.

Minimizing the variance with respect to β yields

$$\beta^* = \frac{\sigma_Y}{\sigma_X}\rho_{XY} = \frac{\mathrm{Cov}(X,Y)}{\mathrm{Var}(X)}, \tag{2.2.21}$$

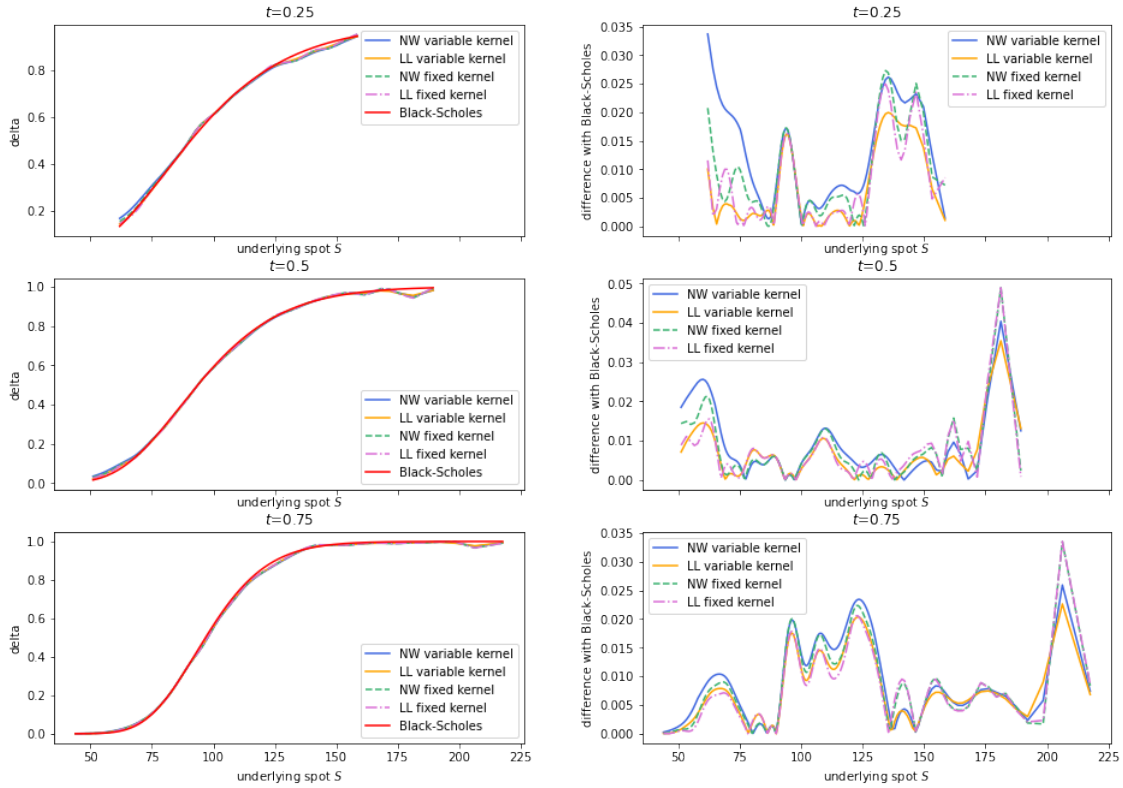

Figure 2.3: Pathwise delta using different variable and fixed kernel estimators versus Black-Scholes prices (left); difference with Black-Scholes (right)

which is exactly the optimal hedge we obtain in Equation (2.1.9) while considering the underlying as the control variate and the portfolio value as estimated variable. Substitute Equation (2.2.21) into Equation (2.2.20), we get the minimum variance of the control variate

$$\mathrm{Var}\left(\bar{Y}_{CV}\left(\beta^*\right)\right) = \left(1-\rho_{XY}^2\right)\frac{\sigma_Y^2}{n}. \tag{2.2.22}$$

It is obvious that the variance reduction depends heavily on the correlation coefficient between the two variables. ρ_{XY} approaching to 1 associates with a sharp reduction in the variance of the estimated random variable.

However, Equation (2.2.22) only applies if β^* is known. In practice, if $\mathrm{E}[Y]$ is unknown, it is unlikely that ρ_Y or ρ_{XY} would be known. However, we may still get most of the benefit of a control variate using an estimate of β^*. For example, replacing the population parameters in Equation (2.2.21) with their sample counterparts yields the estimate

$$\hat{\beta}^* = \frac{\sum_{i=1}^n \left(X_i-\bar{X}\right)\left(Y_i-\bar{Y}\right)}{\sum_{i=1}^n \left(X_i-\bar{X}\right)^2}, \tag{2.2.23}$$

same as how we apply kernel regression to estimate the optimal hedge on the underlying with n meshed points as Equaition (2.2.6). Dividing numerator and denominator by n and applying the strong law of large numbers shows that $\hat{\beta}^*$ converges to β^* with probability 1. However, a bias exists in the estimation E_Y arises.

While β is no longer fixed, the control variate estimator $\bar{Y}(\beta^*)$ need not to be unbiased, since β^* and $\bar{X}$ are dependent. A way to remove such dependency is to use n_1 replications to compute an estimate β_1^*, and then apply this coefficient with the remaining $n-n_1$ replications of (X_i, Y_i). However, the bias produced by estimating β^* by $\hat{\beta}^*$ is small as shown below, so the cost of separating replications is unattractive.

Proposition 2.2.3. *The estimation error in $\hat{\beta}^*$ is only $O(1/\sqrt{n})$.*

We give a detailed proof of this proposition in Appendix A.1.

2.2.2.1 Multiple Controls

We now generalize the method of control variates to the case of multiple controls as it will help when we deal with more complex payoffs like basket options and rainbow options.

Suppose each replication i of a simulation produces outputs Y_i and $X_i = \left(X_i^{(1)}, \ldots, X_i^{(z)}\right)^\top$, and suppose the expectation E[X] is known. We assume that the pairs (Xi, Yi), $i = 1, \ldots, n$, are i.i.d. with covariance matrix

$$\begin{pmatrix} \Sigma_X & \Sigma_{XY} \\ \Sigma_{XY}^\top & \sigma_Y^2 \end{pmatrix},$$

where Σ_X is $z \times z$ and assumed to be non-singular, Σ_{XY} is $z \times 1$, and scalar σ_Y^2 is the variance of the Y_i.

Let $\bar{X}$ denote the vector of sample means of the controls. For fixed $\beta \in \mathbb{R}^z$,

$$\bar{Y}_{CV}(\beta) = \bar{Y} - b^\top(\bar{X} - \mathrm{E}[X]).$$

Then the variance per replication is

$$\mathrm{Var}\left[Y_i - \beta^\top (X_i - \mathrm{E}[X])\right] = \sigma_Y^2 - 2\beta^\top \Sigma_{XY} + \beta^\top \Sigma_{XX} \beta, \tag{2.2.24}$$

which is minimised at

$$\beta^* = \Sigma_X^{-1} \Sigma_{XY}. \tag{2.2.25}$$

Similar to the single control variate case, this is the vector of coefficients in a regression of Y against X.

In statistics, the squared correlation coefficient between scalar X and Y is defined as

$$R^2 = \Sigma_{XY}^\top \Sigma_X^{-1} \Sigma_{XY} / \sigma_Y^2.$$

We substitute Equation (2.2.25) into (2.2.24) per replication, and find the minimal variance of $Y_i(\beta^*)$ is given by

$$\sigma_Y^2 - \Sigma_{XY}^\top \Sigma_X^{-1} \Sigma_{XY} = \left(1 - R^2\right) \sigma_Y^2.$$

Thus, R^2 measures the fraction of the variance of Y that is removed in optimally using X as a control.

In practice, the optimal vector of coefficients β is unknown but may be estimated. The standard estimator replaces Σ_X and Σ_{XY} in Equation (2.2.24) with their sample counterparts to get

$$\hat{\beta}_n = S_X^{-1} S_{XY},$$

where S_X is the $z \times z$ matrix with entry j, k, given as

$$\frac{1}{n-1}\left(\sum_{i=1}^n X_i^{(j)} X_i^{(k)} - n\bar{X}^{(j)}\bar{X}^{(k)}\right),$$

and S_{XY} is a z-dimensinal vector with entry j:

$$\frac{1}{n-1}\left(\sum_{i=1}^n X_i^{(j)} Y_i - n\bar{X}^{(j)}\bar{Y}\right).$$

2.2.3 Connections between control variates and option pricing

To put the ideas into action in derivative pricing, underlying assets provide a virtually universal source of control variates. We know that the absence of arbitrage is essentially equivalent to the requirement that appropriately discounted asset prices be martingales. To be concrete, suppose we are working in the risk-neutral measure and suppose the interest rate is a constant r. If S_{t_i} is an asset price, then $e^{-r(t_{i+1}-t_i)} S_{t_{i+1}}$ is a martingale and $\mathrm{E}^{\mathbb{Q}}[e^{-r(t_{i+1}-t_i)} S_{t_{i+1}}] = S_{t_i}$.

We can therefore form the control variate estimator for the portfolio value at each time step t_i:

$$\Pi_{t_i} = e^{-r(t_{i+1}-t_i)} \left(\Pi_{t_{i+1}} - \hat{\beta}^*_{t_i} \left(S_{t_{i+1}} - e^{r(t_{i+1}-t_i)} S_{t_i} \right) \right), \tag{2.2.26}$$

with $\Pi_{t_{i+1}}$ the option value at time t_{i+1}, and this estimator has exactly the same form as Equation (2.1.6). This enables us to estimate the optimal control variate parameters using the same ways that we predict deltas, i.e. by applying kernel regression onto both the pathwise estimator and the variance minimisation solution.

For a standard call option, the correlation between the discounted payoff and the underlying are thus the effectiveness of the control variate depends on the strike K. At $K = 0$ we would have perfect correlation; for an option that is deep out-of-the-money, i.e., with large K, the correlation could be quite low, where $\hat{\beta}^*_{t_i}$ would tend to zero.

Remark 2.2.4. In the risk-neutral world, the stock price always equals to the expectation of the discounted forward price, so that $\mathrm{E}[S_{t_{i+1}} - e^{r(t_{i+1}-t_i)} S_{t_i}] = 0$ always holds, which leads the control variate estimator to be unbiased all the time. Hence, the control variate would work well even if $\hat{\beta}$ does not equal to $\hat{\beta}^*$ exactly.

Remark 2.2.5. For the non risk-neutral cases where the conditional expectation is no longer zero, it could still be calculated via kernel estimation

$$\hat{m}_h \left(x_k, S_{t_i}; S_{t_{i+1}} - e^{r(t_{i+1}-t_i)} S_{t_i} \right), \tag{2.2.27}$$

where $\hat{m}_h$ is defined in Equation (2.2.3) and (2.2.7), and and $k = 1, 2, \ldots, u$ is each meshed point of underlying at time t_i.

2.2.3.1 Practical experiments

By including a control variate with pathwise delta and the variance minimisation solution (Equation (2.1.9)) as a CV parameter respectively, we get the left plot in Figure 2.4. Comparing with the raw kernel estimates as shown in Figure 2.1, the inclusion of control variates significantly reduces the variance of the estimator, and in all the four cases, the estimates overlap Black-Scholes benchmark with tiny errors. Furthermore, same as what we observed above, LL variable kernel estimators give outperforming fits than keeping the same bandwidth, especially around the 'tails'.

An important notice is that though we find the estimated deltas are not exactly lying on the true values from Figure 2.2 and 2.3, substituting them into the price estimates as control variate parameters gives nearly perfect results. The reason is that, again as we mentioned in Section 2.2.3, in the risk-neutral case as long as the control variate estimator is unbiased as in Lemma 2.2.2, even if the control variate parameter is not chosen perfectly same as Equation (2.2.23), the variance reduction still works well.

To explicitly compare these four models in Figure 2.4, we refer to Table 2.2. Variable kernel associates with smaller overall mean squared errors in each step under comparisons, no matter which control variate parameter is chosen. And more importantly, variable kernel estimator does improve the fit around the 'tails', since less absolute errors occur in 1% and 99% quantiles. Another interesting finding is that although according to Figure 2.2 and 2.3, pathwise delta estimates are closer to Black-Scholes deltas, when we include them as control variate parameters in order to estimate option prices, they gives slightly less precise outcomes than the variance minimisation deltas except the estimated expected exposure, which suggests selecting the variance minimisation deltas seems to be a recommended choice.

2.2.4 Gaussian process regression

We mentioned in Section 1 that regression enables us to calculate the conditional expectations to approximate the option prices. Due to the restrictions of the parametric models with basis functions using in LSM, we now replace it with the non-parametric Gaussian process regression (GPR), since it gives a more generalised kernel based regression instead of relying on a specific choice of basis. GPR brings plenty of benefits. Firstly, GP directly gives a distribution for the prediction value rather than a single value in the regression so that is good at identifying model

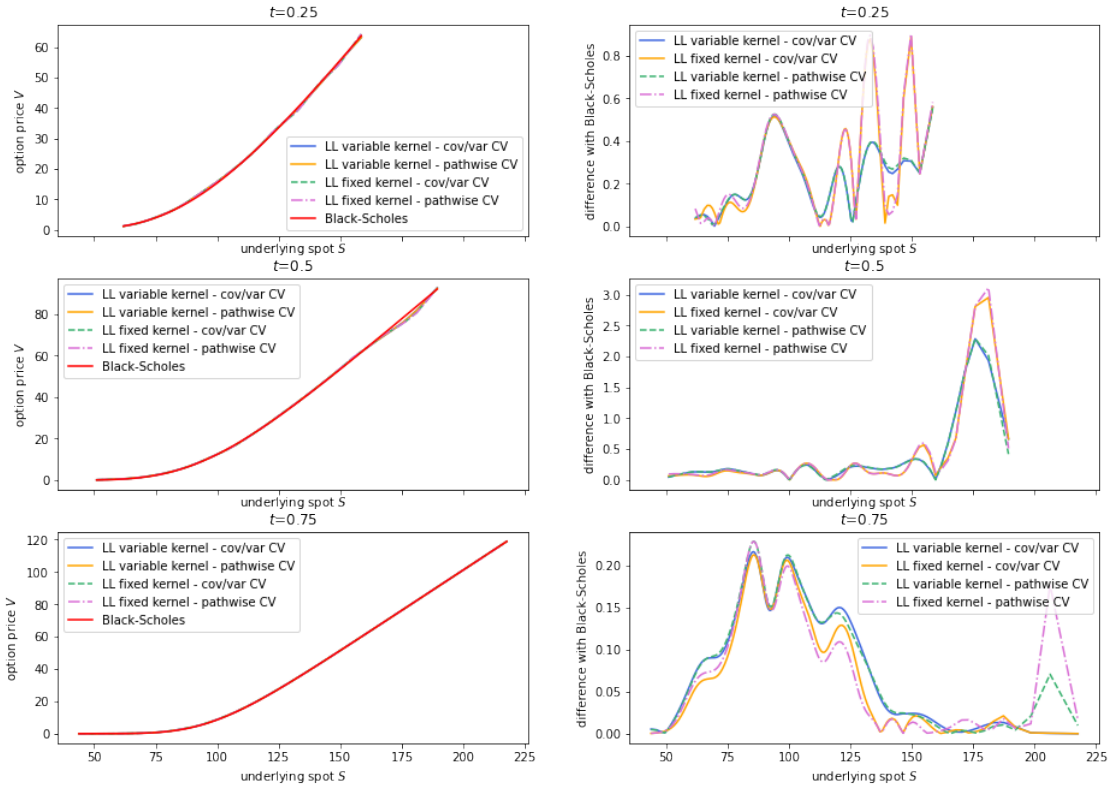

Figure 2.4: Raw kernel estimates of option prices with control variates using Local Linear kernel estimator with variable kernel versus Black-Scholes prices (left); difference with Black-Scholes (right)

		Variance minimisation CV			**Pathwise delta CV**		
Kernel	Metric	$t = 0.25$	$t = 0.50$	$t = 0.75$	$t = 0.25$	$t = 0.50$	$t = 0.75$
Variable LL	MSE	**0.0863**	**0.1173**	0.0074	**0.0888**	**0.1189**	0.0078
	mean	0.1871	0.0416	0.1151	0.1907	0.0455	0.1158
	1%	0.0398	0.1040	0.0006	0.0289	0.1051	0.0006
	50%	0.3824	0.0663	0.1490	0.3914	0.0674	0.1485
	99%	**0.4985**	**2.6760**	0.0005	**0.5021**	**2.6802**	0.0181
Fixed LL	MSE	**0.1507**	**0.1530**	0.0065	**0.1525**	**0.1606**	0.0070
	mean	0.1802	0.0178	0.0999	0.1856	0.0243	0.0976
	1%	0.1393	0.0755	0.0008	0.0903	0.0909	0.0010
	50%	0.3902	0.0735	0.1480	0.3952	0.0735	0.1417
	99%	**1.3051**	**2.9289**	0.0017	**1.3156**	**2.9392**	0.0059

Table 2.2: Mean squared errors and absolute errors in mean, 1%, 50% and 99% quantile of estimated prices with control variates compared to Black-Scholes results

uncertainties. Also, prior knowledge and specifications about the shape of the model are able to be added by selecting different kernel functions, which further enhances its robustness. Rather than claiming the value function relates to some specific models, a Gaussian process can represent the objective function obliquely, but rigorously.

For detailed descriptions of Gaussian process regression, please refer to Appendix B.2.

2.2.4.1 Practical experiments

When we replace the previous polynomial basis functions with GPR, we choose RBF (Radial-basis function) as introduced in Appendix B.2 as the kernel function. Making use of *sklearn.gaussian_process* in Python, we display the results in Figure 2.5 for six different GPR models, regressing purely on

the raw variable kernel estimates, adding a control variate with either pathwise deltas or variance minimisation deltas as the parameter with NW or LL kernel estimator, respectively.

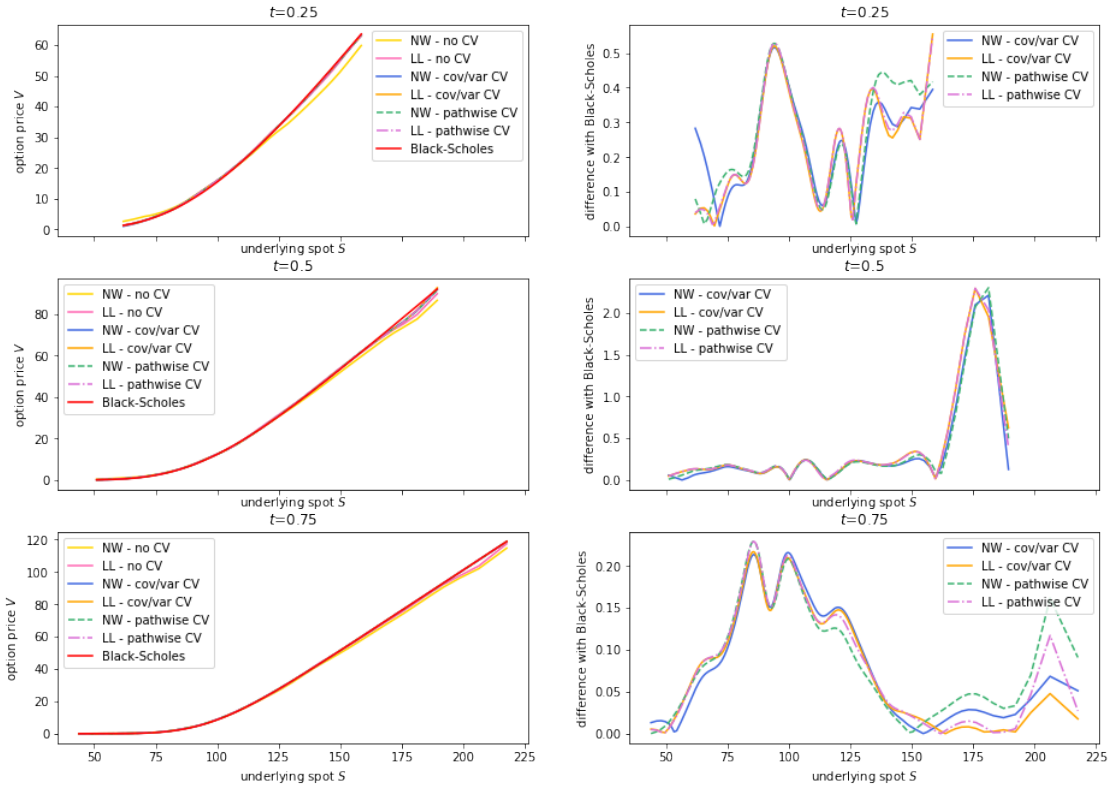

Figure 2.5: Price estimates by Gaussian Process Regression with(out) control variates using different variable kernel estimators versus Black-Scholes prices (left); difference with Black-Scholes (right)

We can observe that if no control variate is added for GPR, the deviations at the 'tails' compared to Black-Scholes cannot be eliminated even if 50,000 paths are already simulated using NW estimator, due to the high estimation variance. And similar to the comparisons from Figure 2.1 and 2.4, we again verify that in order to get a reliable estimation result, control variates need to do far less work for LL estimator than NW, i.e. we still get excellent enough estimates without the need of estimating covariance and variance in Equation (2.2.1) by kernel regression or estimating Equation (2.2.9) by differentiating the payoff. Since it is difficult to differentiate the payoff for a generic framework, LL estimator effectively reduces the computational cost.

All four models with control variates under GPR overlap the Black-Scholes benchmark, and the model using LL estimator with variance minimisation solutions as control variate parameter seems to make slightly more robust predictions around the 'tails'.

So far, we find out several outperformed models in estimating deltas as well as the option prices. Now, we gather all these models and aim to evaluate their performance with the following robustness check, and compare the model that gives the best performance under our new approach with the algorithms in the literature.

2.2.5 Robustness check

There are mainly three parts in this robustness check section. To begin with, we select the best-performed model among those we discuss above under the newly proposed DCKE algorithm through comparisons, taking into account both discounting payoff and backward recursion approach. Next, we compare the error statistics of the outperformed model with that obtained from the traditional LSM algorithm. Finally, we compare the new algorithm with those famous ones in the literature by plotting the resulting estimates. These procedures all verify the robustness of our new algorithm.

2.2.5.1 Model selection

We now reduce the number of simulation paths and compare the four outperformed model for price estimation throughout the experiments above, namely 'raw kernel estimates with control variates' and 'GPR on raw kernel estimates with control variates' with either pathwise estimation or the variance minimisation solution as the control parameter using LL kernel estimator, which are also the estimates of deltas. We use 10,000 simulated paths to make our improved approach comparable with the LSM in Chapter 1.

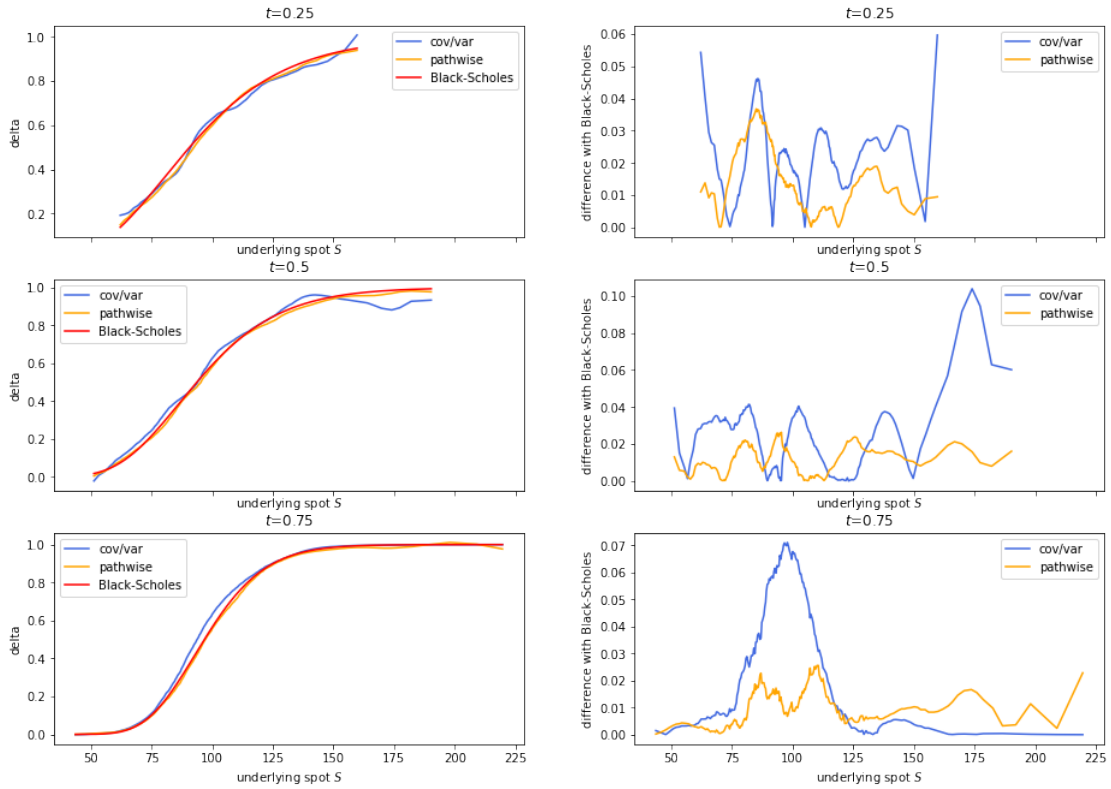

Figure 2.6: Discounting payoff approach: pathwise estimates and variance minimisation solution of delta against Black-Scholes deltas (left); difference with Black-Scholes (right)

From Figure 2.6, the reduction of number of paths indeed gives worse estimates of deltas in discounting payoff approach compared with the results in Figure 2.2 and 2.3, especially as time goes backward. However, we have similar observations. Compared with the covariance-variance form estimator, pathwise deltas are closer to Black-Scholes benchmarks particularly around the 'tails' at earlier time steps and along with less estimation variance.

Then, we estimate deltas using backward recursion approach as shown in Figure 2.7. According to Algorithm 2, we still have the same pathwise deltas as in discounting payoff approach, since pathwise deltas do not depend on the intermediate option prices. However, the deltas obtaining from variance minimisation solution are different. We get two different sets of deltas in this form while applying recursion on the prices interpolated from GPR with pathwise deltas and variance minimisation deltas as control variates respectively. It is evident that the pathwise deltas are more precise than any other form of the estimates, particularly around the 'tails', which indicates pathwise estimators are indeed more robust and backward recursion approach does not help with the prediction of deltas.

As for the price estimates, we come back to the original discounting payoff approach first. According to Figure 2.8, it is still difficult to directly distinguish which model for price estimation stands out even if the number of simulation paths is reduced to 10,000, which verifies the robustness of our DCKE algorithm when control variates apply. To compare these models, we finally reduce the number of paths to 3,000 and plot the difference with Black-Scholes. The right plot in Figure 2.8 indicates the inclusion of Gaussian Process Regression smooths extreme prediction errors especially at later time steps, and choosing pathwise deltas as the control variates parameters

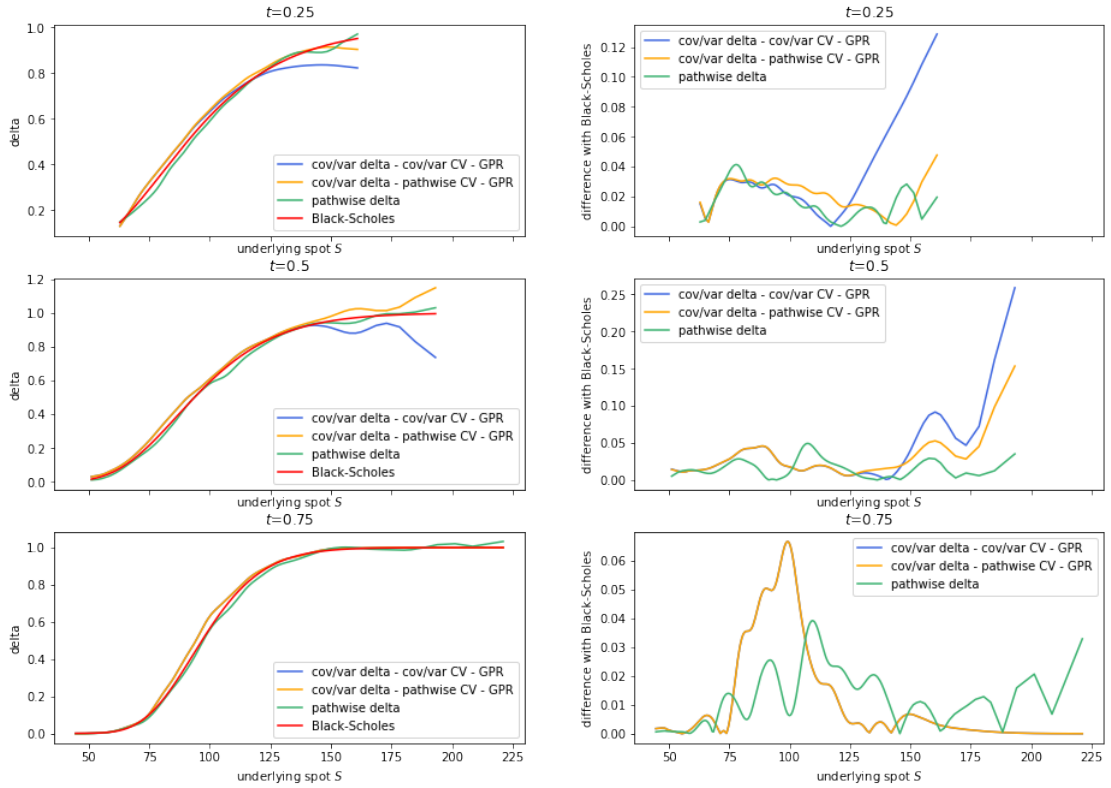

Figure 2.7: Backward recursion approach: pathwise estimates and variance minimisation solution of delta against Black-Scholes deltas (left); difference with Black-Scholes (right)

effectively improves the precision in the middle and at the ends of the underlying spots.

Hence, among these models, pathwise estimation gives the best estimates of deltas, and kernel regression with pathwise delta as the control variate parameter along with GPR outperform all other models in estimating option prices. We further compare the price estimates obtained from the best model using discounting payoff approach and backward recursion approach (see Algorithm 2 for pseudo-code). Figure 2.9 shows that similar to the conclusion in Chapter 1, backward recursion approach performs terribly while estimating the 'tails', and the error accumulates as time goes backwards. The error statistics from Table C.1 in Appendix C.3 verifies that neither the overall MSE nor absolute errors in PFE for backward recursion approach outperforms discounting payoff approach, and therefore we pursue discounting payoff approach in later experiment.

After repeating the process for 100 times, we report the error statistics of the best-performed DCKE model and compute the error ratios of LSM in Table 1.3 and 1.4 to our preferred model, and put them into the brackets side by side to the error statistics of the new model in Table 2.3.

We observe that errors under almost all metrics have been reduced with our new algorithm, and the most significant improvements happen in MSE and the absolute errors around the 'tails' of underlying spots. The proposed DCKE model associates with only seventh of the MSE and even fiftieth of the absolute error in LSM at 1% quantile while approaching the maturity. Such sharp contrasts verifies the success of our improved algorithm.

2.2.5.2 Validity under Heston model

Similarly, by applying our selected model to the Heston framework, remarkably improvements are also observed in Figure 2.10 compared to the results in Figure 1.8 while using LSM. The DCKE algorithm enables a perfect fit for both prices and deltas.

2.2.5.3 Improvements on LSM

Table 2.3 proves that the DCKE algorithm successfully tackles the shortcomings of LSM we list in Section 1.3:

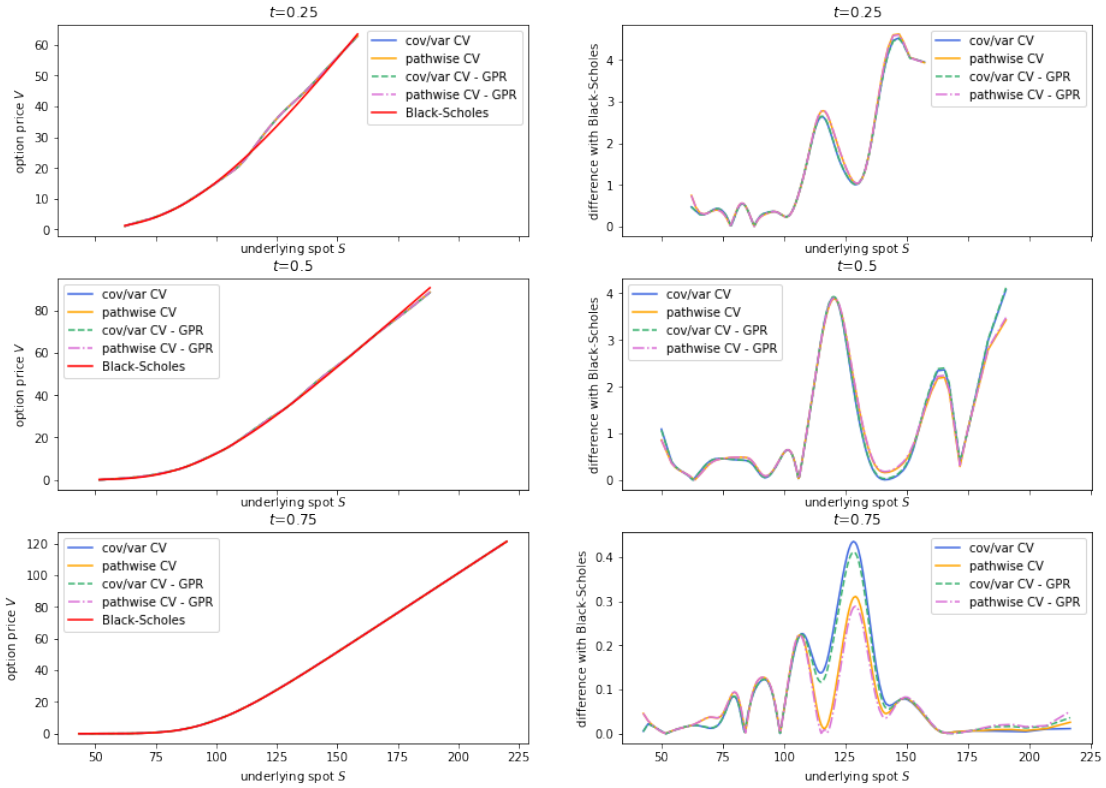

Figure 2.8: Models comparison for price estimates against Black-Scholes prices (left); difference with Black-Scholes (right)

	Option price			Delta		
Metric	$t = 0.25$	$t = 0.50$	$t = 0.75$	$t = 0.25$	$t = 0.50$	$t = 0.75$
MSE	0.3967	0.4081	0.0560	0.0003	0.0003	0.0002
	(**1.19**)	(**1.13**)	(**6.30**)	(**12.31**)	(**5.62**)	(**8.11**)
mean	0.1808	0.1989	0.1857	0.0073	0.0069	0.0070
	(0.07)	(0.17)	(0.15)	(**1.26**)	(0.99)	(0.69)
1%	0.2010	0.0895	0.0252	0.0060	0.0020	0.0002
	(0.29)	(**1.14**)	(**6.70**)	(**2.54**)	(**10.68**)	(**377.89**)
50%	0.2485	0.2670	0.3208	0.0064	0.0114	0.0153
	(0.16)	(0.21)	(0.95)	(0.09)	(0.61)	(0.78)
99%	0.1750	0.2809	0.0473	0.0032	0.0015	0.0061
	(0.79)	(**1.01**)	(**2.04**)	(**27.94**)	(**36.22**)	(**8.69**)

Table 2.3: Mean squared errors and absolute errors in mean and 1%, 50% and 99% quantile of option price and delta of DCKE model compared to Black-Scholes results (average of 100 repetitions; in backets: error ratios of LSM model to DCKE model reported in brackets)

Remark 2.2.6. The estimates for both prices and deltas converge properly to true results with less overall MSE, and the convergence is independent of the choice of basis functions.

Remark 2.2.7. The fit of deltas is improved remarkably, especially around the edges of the underlying spots, which also enables accurate estimations of quantities like Margin Value Adjustments (MVA).

Remark 2.2.8. Even the 'tails' are now perfectly predicted with the new algorithm.

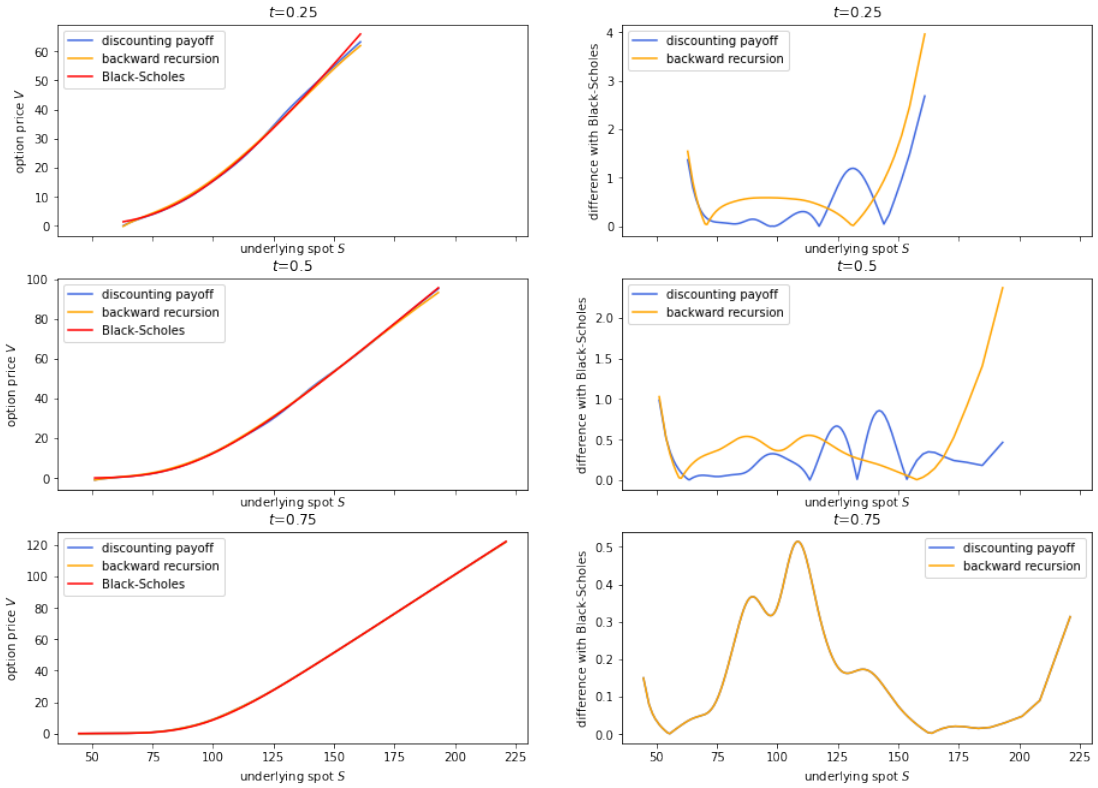

Figure 2.9: DCKE algorithm for discounting payoff approach and backward recursion approach (left); difference with Black-Scholes (right)

2.2.5.4 Comparisons with literature

Now we compare our proposed DCKE approach with both of the approaches from Potters et al. (2001) and Grau (2008, pg 126) as introduced at the beginning of the chapter. They both performed least-squares regressions to determine delta at each time step, after which Grau obtained portfolio values with the backward recursion formula we have shown in Equation (2.1.6). Instead, Potters continued applying regression to predict portfolio values by setting the response according to the same equation.

We first choose the same degree of basis functions as the example in Grau (2008, pg 126) and obtain the left plot in Figure 2.11. It is obvious that the deltas obtained by the DCKE algorithm through kernel regression outperform Grau and Potters' regression method throughout the whole range of underlying spots. To be more precise, we loop through different number of basis functions from 3 to 10 and take the average MSE over 100 repetitions. According to the right plot in Figure 2.11, no matter which polynomial degree we choose in Potters' and Grau's regression method for delta, our DCKE approach consistently gives a better performance. As the DCKE algorithm is independent of the polynomial degree, it improves the delta estimation universally.

Given that we obtain more accurate deltas than Potters and Grau, we now look into the price estimation. As Potters' approach also employ regressions to estimate prices with the inclusion of estimated deltas as in our algorithm, there is no need to check his result, since we already get better deltas at each time step. Grau used backward recursion to estimate prices, in which estimated deltas are also plugged in. We plot his predictions and also the results when we substitute our own predicted meshed delas into the recursion. These predictions along with DCKE prices, and the average MSE are shown in Figure 2.12.

It is evident that our DCKE algorithm again outperforms Grau's model by involving far fewer fluctuations. Grau's method might be feasible to obtain a reliable option price at time zero, since all the predicted prices fluctuate around the actual values. However, for computing XVA, i.e., when we need predictions for intermediate time steps, DCKE is a better algorithm to pursue. For backward recursion with DCKE deltas, we get jagged predictions while only including the meshed underlying values. Furthermore, when we interpolate the estimated deltas to the full

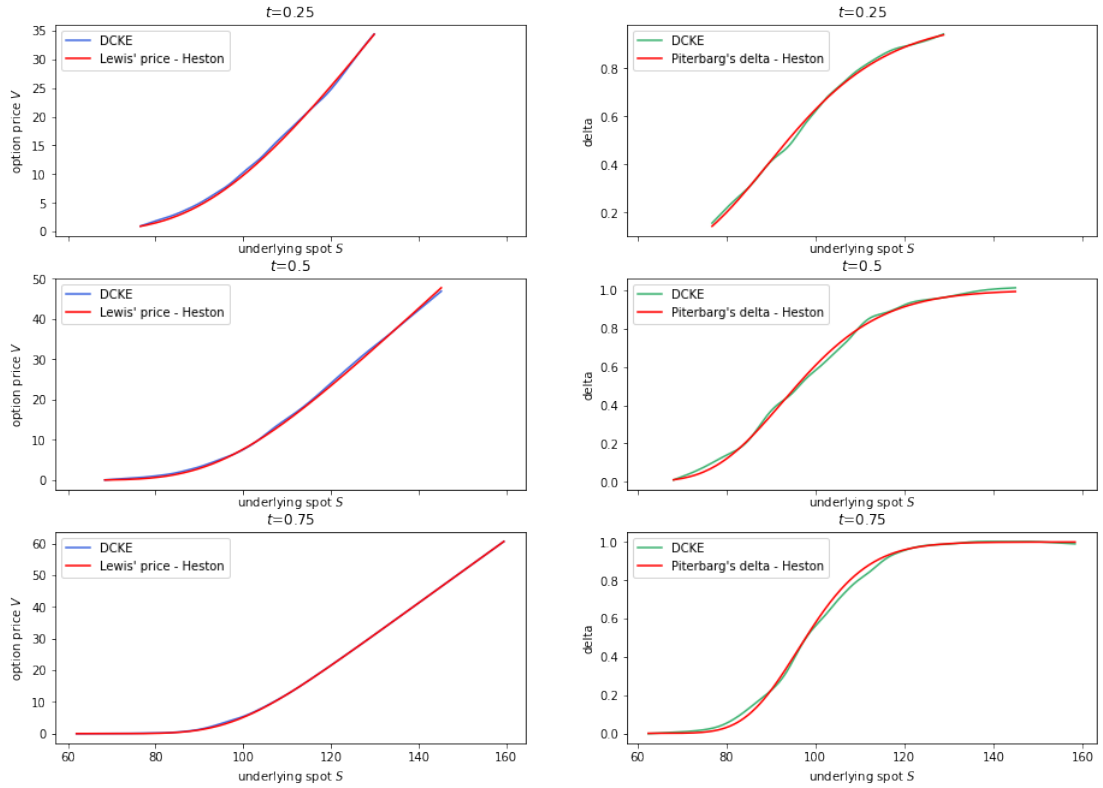

Figure 2.10: DCKE for option prices versus Heston prices - Lewis'approximation (left); delta estimates versus Heston deltas - Piterbarg's approximation (right)

dimension and do the recursion again, the predicted prices are still far from lying on the Black-Scholes benchmark. Therefore, we further verify the robustness of the DCKE framework.

2.2.5.5 Neural network-based DCKE

Given that our best performed model, the DCKE algorithm works robustly in all cases above through comparisons, we are interested to investigate how would the model work if we replace GPR with a neural network (NN) in the algorithm.

Neural network provides an alternative possibility of replacing the necessity of choosing a specific form of basis functions through pricing and hedging, which is rich in the sense that it encompasses almost arbitrary reasonable functional relationship between the outputs and inputs.

In neural networks, functions constructed by composing alternatingly affine and activation functions. Mathematically, a neural network $\boldsymbol{f} : \mathbb{R}^I \to \mathbb{R}^O$ can be expressed as

$$\boldsymbol{f} = \boldsymbol{\varphi}_r \circ \boldsymbol{L}_r \circ \cdots \circ \boldsymbol{\varphi}_1 \circ \boldsymbol{L}_1,$$

where $\boldsymbol{L}_i : \mathbb{R}^{d_{i-1}} \to \mathbb{R}^{d_i}$ with $\boldsymbol{L}_i(\boldsymbol{x}) = \omega_i \boldsymbol{x} + \boldsymbol{b}_i, \boldsymbol{x} \in \mathbb{R}^{d_{i-1}}, \omega_i \in \mathbb{R}^{d_i \times d_{i-1}}$ and $\boldsymbol{b_i} \in \mathbb{R}^{d_i}$, for any $i = 1, \ldots, r$, is an affine function, such that $d_0 = I$ and $d_r = O$. $\varphi_i : \mathbb{R}^{d_i} \to \mathbb{R}^{d_i}$ is component-wise application of a function $\varphi_i : \mathbb{R} \to \mathbb{R}$, which is called an activation function, i.e.,

$$\boldsymbol{\varphi}_i(x_1, \ldots, x_{d_i}) := \left(\varphi_i(x_1), \ldots, \varphi_i\left(x_{d_i}\right)\right).$$

For more details and the concepts of hidden layers in a feedforward neural network can be found in the graph illustrations Figure C.4 in Appendix C.2.

The only difference in NN-based DCKE compared to the original DCKE approach is that after predicting deltas using either pathwise estimation or variance minimisation solution, we interpolate the deltas to full dimension as control variates adding onto the kernel fitted prices, and apply a neural network to make predictions with the underlying spot sequence as input variable.

In our experiment, we find it is especially tricky to obtain a great set of price estimates through NN-based DCKE. We end up with a model with 'SELU' activation function for the first four layer,

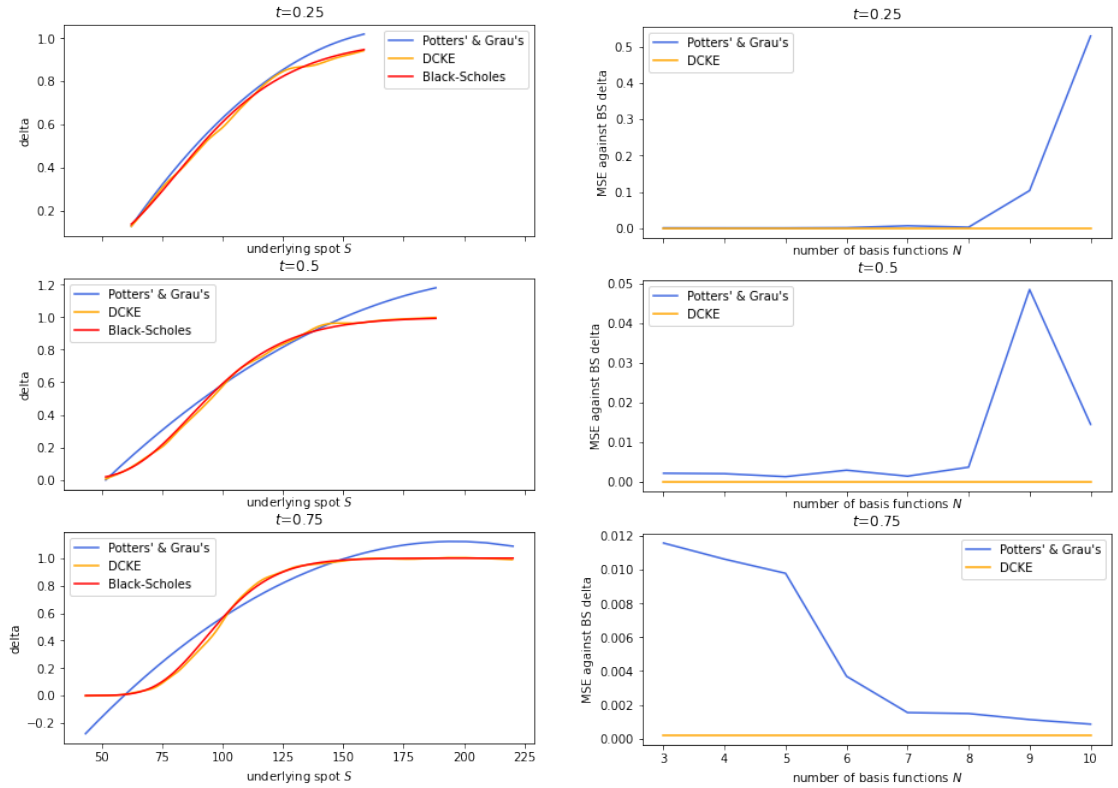

Figure 2.11: Delta estimates by DCKE and Potters & Grau against Black-Scholes prices (left); mean squared errors of estimated deltas against number of basis functions (average of 100 repetitions)

and 'linear' activation for the last. We restrict the minimum learning rate to be 10^{-8} and set batchsize to be 500. It is surprising that even under Sobol sequence (which will be shown to be more efficient than GBM in Section 2.3), 100,000 paths are required to get a set of predictions with MSE comparable to GPR-based DCKE method with 10,000. Table 2.4 directly reflects how computationally-costly this approach is, which indicates we should pursue the original GPR-based DCKE instead.

As we mentioned in Section 2.2.4, GP directly captures the model uncertainty by giving a distribution for the prediction value rather than just a single value in the regression. However, this uncertainty is not directly captured by neural networks according to Gal and Ghahramani (2016), and therefore destroys the model performance.

2.2.6 Algorithm - Dynamically Controlled Kernel Estimation

Algorithm 2 describes the newly proposed algorithm DCKE in pseudo-code.

2.3 Sobol sequence

We introduce Sobol sequence as a new generator of underlying spots in order to remarkably improve the computational efficiency.

2.3.1 Generalisation

A Sobol sequence is a low discrepancy quasi-random sequence used in quasi-Monte Carlo (QMC) method. The discrepancy of a sequence is a measure of its uniformity and is defined as follows:

Definition 2.3.1. Given a set of points $x^1, x^2, \cdots, x^N \in I^S$ and a subset $G \subset I^S$. Define the counting function as the number of points $x^i \in G$. For each $x = (x_1, x_2, \ldots, x_s) \in I^S$, let G_x be the

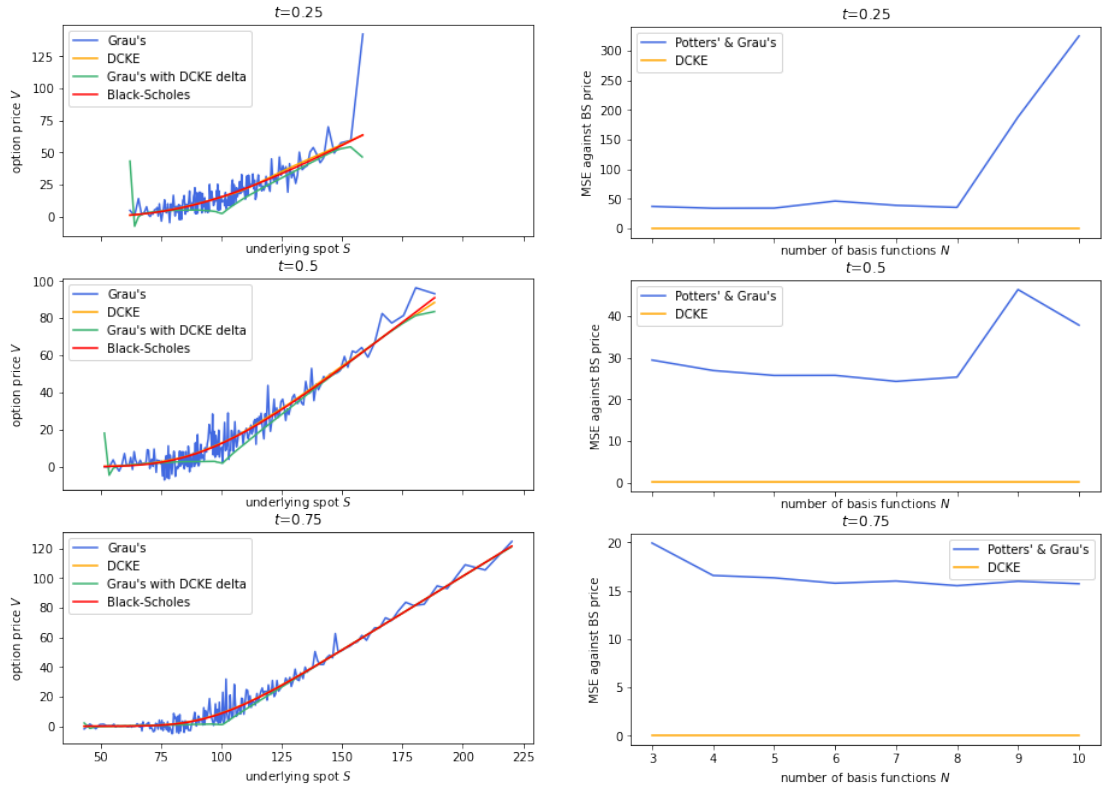

Figure 2.12: Price estimates by DCKE and Grau against Black-Scholes prices (left); mean squared errors of estimated prices against number of basis functions (average of 100 repetitions)

rectangular s-dimensional region $G_x = [0, x_1) \times [0, x_2) \times \cdots \times [0, x_S)$ with volume $x_1, x_2, \ldots, x_N$. Then the discrepancy of the points $x^1, x^2, \ldots, x^N$ is given by:

$$D_N^* \left(x^1, x^2, \ldots, x^N \right) = \sup_{x \in I} \left| S_N \left(G_x \right) - N x_1 x_2, \cdots, x_S \right| .$$

The discrepancy is therefore computed by comparing the actual number of sample points in a given volume of multidimensional space with the number of sample points that should be there assuming a uniform distribution. The discrepancy of the first N terms of quasi-random sequence has the form:

$$D_N^* \left(x^1, x^2, \ldots, x^N \right) \leq C_S (\log N)^S + O \left((\log N)^{S-1} \right),$$

for all $N \geq 2$.

The principal aim in the construction of low-discrepancy sequences is thus to find sequences in which the constant C_S is as small as possible. Various sequences have been constructed to achieve this goal, and we focus on one of the quasi-random sequences introduced by Sobol' (1967), Sobol. Sobol sequence was designed to cover the unit hypercube with lower discrepancy than completely random sampling (e.g. Random Search), shown as Figure 2.13.

2.3.2 Practical experiment

In our experiment, we find that after replacing Geometric Brownian Motion with Sobol sequence in DCKE with pathwise delta as the control variate parameter, the estimation efficiency soars evidently, which is demonstrated through Figure 2.14 and Figure 2.15.

In terms of mean squared errors, Figure 2.14 indicate that for the same number of paths ranging from 1,000 until 10,000, Sobol outperforms GEM consistently while estimating both opyion prices and deltas. Especially at earlier time steps, Sobol already gives much smaller MSE with 1,000 paths than GBM with 10,000 paths. The increase of simulation paths only has tiny reduction in MSE for Sobol compared to GBM. On average, only 2,000 paths are needed for Sobol sequence

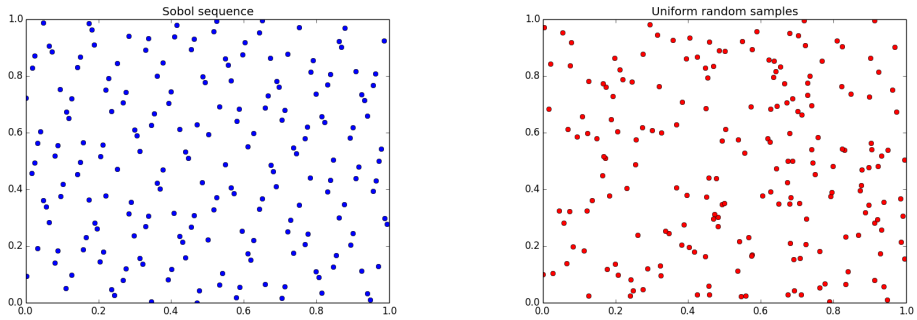

Figure 2.13: 200 points sampled in 2D with Sobol sequence (left); uniformly at random (right)

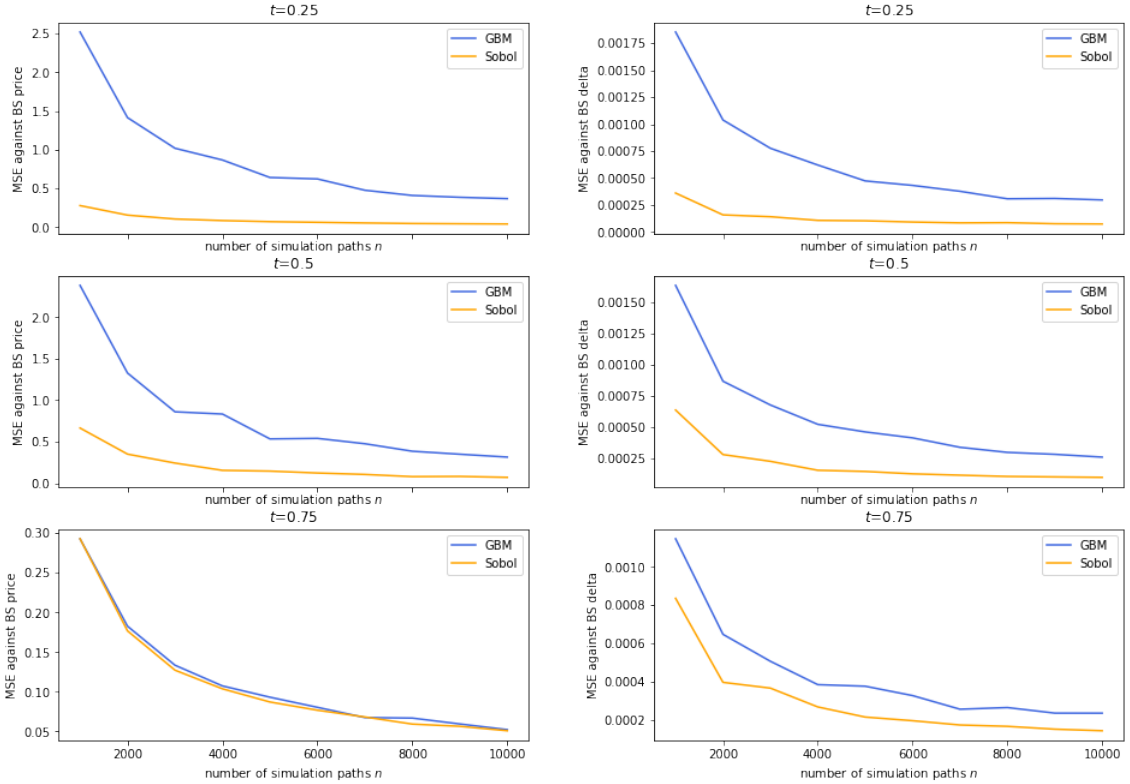

Figure 2.14: Mean squared errors of estimated prices (left); of estimated deltas (right) against number of simulation paths for GBM and Sobol (average of 100 repetitions)

to obtain similar estimation accuracy to GBM with 10,000 paths, and when it is further from the expiration date, even fewer paths are needed.

For the PFE comparison in Figure 2.15, Sobol again associates with smaller absolute errors in each quantile than GBM on average, and the error plots for Sobol are far smoother with the increasing number of simulation paths than the ones for GBM, particularly for deltas, which also supports the feasibility of pursuing Sobol with far fewer simulation paths for the estimation.

This is an amazing breakthrough since we remarkably accelerate the computaion of prices and sensitivities, which is hence beneficial to the computation of XVA.

We report the precise error statistics in Table C.2 in Appendix C.4. Table 2.4 reflects the improvement of Sobol in computational efficiency more directly by giving the exact average calculation times. In a conclusion, similar results are obtained with Sobol by saving nearly 60% of the computational cost while using GBM.

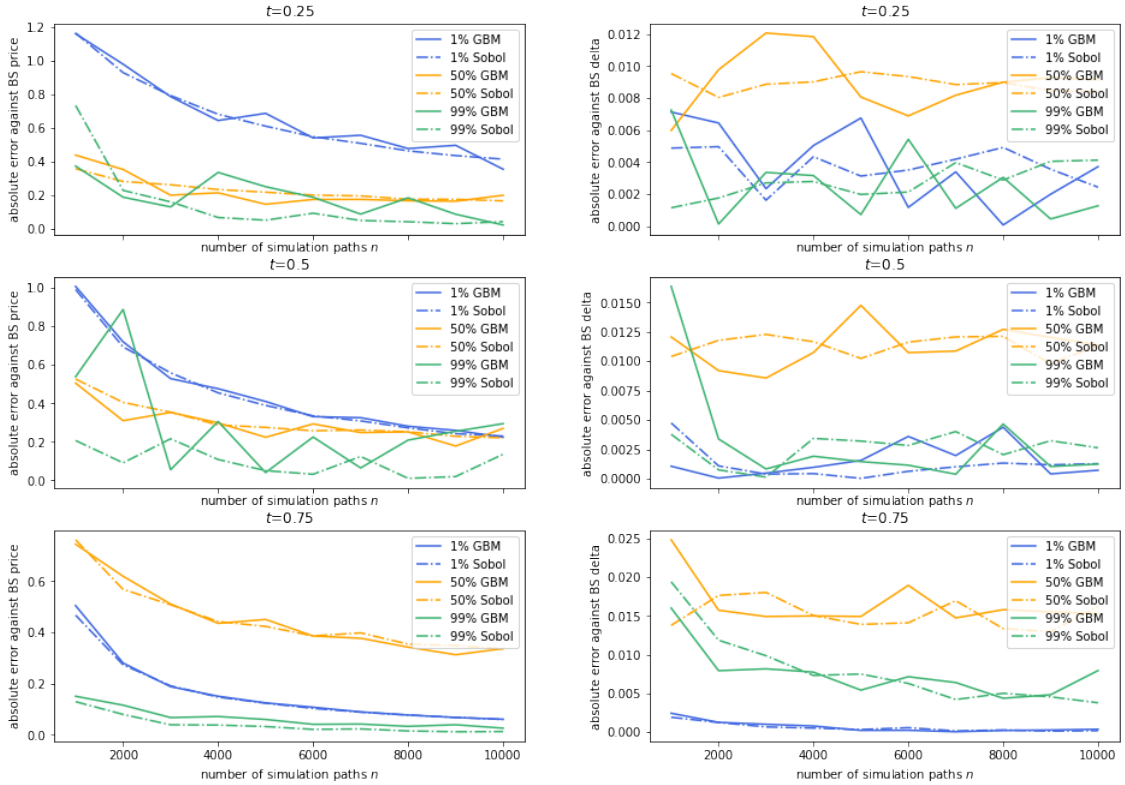

Figure 2.15: Absolute errors in 1%, 50% and 99% quantiles of estimated prices (left); of estimated deltas (right) against number of simulation paths for GBM and Sobol (average of 100 repetitions)

Sequence	**Method**	**Paths**	**Time** (s)
GBM	GPR-based DCKE	10,000	7.1289
Sobol	GPR-based DCKE	2,000	3.0491
	GPR-based DCKE	1,000	2.7203
	NN-based DCKE	100,000	276.6336

Table 2.4: Computational time of GBM and Sobol with different number of paths under methods DCKE and Neural Networks for 3 intermediate time steps (average of 100 repetitions)

Algorithm 2 Dynamically Controlled Kernel Estimation

Input: K: strike; r: risk-free rate; T: number of time steps; Δt: lag between consecutive time steps; n: number of simulated paths; u: number of meshed samples; S: spot prices sequences S_{t_i} with each length n for $i = 0, \ldots, T$; C: cap of variable kernel ratio (usually 1-30)

Output: X: mesh points of spot price sequence X_{t_i}; $\hat{V}$: option prices sequences $\hat{V}_{t_i}$; $\hat{\Phi}$: deltas sequences $\hat{\phi}_{t_i}$ with each length n for $i = 0, \ldots, T$

Set arrays: $V^j_{t_T} = P(\cdot)$: the payoff function, $j = 1, \ldots, n$; $\hat{V}^*$, $\hat{V}'^*$, $\hat{\Phi}$, $\hat{\Phi}'$ (discounting payoff approach); $\hat{V}^\circ$, $\hat{\Phi}^\circ$ (backward recursion approach)

$h = 0.9 * \min\left(\sqrt{\mathrm{Var}(S_{t_T})}, \frac{IQR(S_{t_T})}{1.34}\right) * n^{-\frac{1}{5}}$ (**fixed kernel**)

for all time steps from $i = T - 1$ down to $i = 0$ **do**

X_{t_i} = samples drawn between $1\% - 99\%$ percentiles of S_{t_i}

for each mesh point $k = 1$ up to $k = u$ **do**

$K^k_{t_i} = \sum_{j=1}^{n} e^{-\frac{1}{2h^2}(S^j_{t_i} - X^k_{t_i})^2}$

end for

$K^* = max(K^k_{t_i})$ for $k = 1, 2, \ldots, u$

for each mesh point $k = 1$ up to $k = u$ **do**

$K^k_{ratio} = min(\frac{K^*}{K^k_{t_i}}, C)$

$h_{new} = hK^k_{ratio}$ (**variable kernel**)

$\hat{V}^k_{t_{i+1}} = \hat{m}^{LL}_{h_{new}}(x_k, S_{t_i}; V_{t_{i+1}})$ ($\hat{m}^{LL}_h$ defined as Equation (2.2.7))

$\hat{S}^k_{t_{i+1}} = \hat{m}^{LL}_{h_{new}}(x_k, S_{t_i}; S_{t_{i+1}})$ ($\hat{m}^{LL}_h$ defined as Equation (2.2.7))

$\hat{\phi}^k_{t_i} = \frac{\hat{m}^{LL}_{h_{new}}(x_k, S_{t_i}; (\Pi_{t_{i+1}} - \hat{m}^{LL}_{h_{new}}(x_k, S_{t_i}; \Pi_{t_{i+1}}))(S_{t_{i+1}} - \hat{m}^{LL}_{h_{new}}(x_k, S_{t_i}; S_{t_{i+1}})))}{\hat{m}^{LL}_{h_{new}}(x_k, S_{t_i}; (\Pi_{t_{i+1}} - \hat{m}^{LL}_{h_{new}}(x_k, S_{t_i}; \Pi_{t_{i+1}})))^2)}$ ($\hat{m}^{LL}_h$ defined as Equation (2.2.7))

$\hat{\phi}'^k_{t_i} = \hat{m}^{LL}_{h_{new}}\left(x_k, S_{t_i}; e^{-r(t_T - t_i)}\frac{dP(\cdot)}{dS_{t_i}}\right)$ ($\hat{m}^{LL}_h$ defined as Equation (2.2.7))

end for

⚠ **discounting payoff approach**

$\hat{V}^*_{t_i} = e^{-r\Delta t}\mathrm{GPR}(\mathrm{RBF}, X_{t_i}, \hat{V}_{t_{i+1}} - \hat{\phi}_{t_i}(\hat{S}_{t_{i+1}} - e^{r\Delta t}X_{t_i}))$

(**CV parameter: variance minimisation solution**)

$\hat{V}'^*_{t_i} = e^{-r\Delta t}\mathrm{GPR}(\mathrm{RBF}, X_{t_i}, \hat{V}_{t_{i+1}} - \hat{\phi}'_{t_i}(\hat{S}_{t_{i+1}} - e^{r\Delta t}X_{t_i}))$

(**CV parameter: kernelised pathwise delta**)

$V^j_{t_i} = e^{-r\Delta t}V^j_{t_{i+1}}$

⚠ **backward recursion approach**: $V^\circ_{t_i}$, $\hat{V}^\circ_{t_i}$, $\hat{\phi}^\circ_{t_i}$ replace V_{t_i}, $\hat{V}_{t_i}$, $\hat{\phi}'_{t_i}$ in the loop above

$\hat{V}^\circ_{t_i} = e^{-r\Delta t}\mathrm{GPR}(\mathrm{RBF}, X_{t_i}, \hat{V}^\circ_{t_{i+1}} - \hat{\phi}^\circ_{t_i}(\hat{S}_{t_{i+1}} - e^{r\Delta t}X_{t_i})$

(**CV parameter: kernelised pathwise delta**)

$V^\circ_{t_i} = \text{1D-interpolate}\ (X_{t_i}, \hat{V}^\circ_{t_i}; S_{t_i})$

end for

return $\hat{V}^*$, $\hat{V}'^*$, $\hat{V}^\circ$, $\hat{\Phi}$, $\hat{\Phi}'$, $\hat{\Phi}^\circ$

Chapter 3

Extension to Exotic Options

In the previous sections, we tested the validity of our new algorithm by applying it to the European vanilla option. The reason is that the Black-Scholes formula gives closed-form solutions so that we can compare how well our estimation algorithm works. After demonstrating the breakthrough we make in predicting vanilla option prices and deltas both theoretically and numerically, we extend the DCKE algorithm to more complex products in this section, even including those high-dimensional exotic options without closed-form solutions, in which case nested Monte Carlo are used as the comparison benchmark.

3.1 Barrier option

Barrier option is a derivative product which payoff is contingent on whether the stock price path has reached a barrier level B during the product lifetime.

Denote the running minimum and maximum of a stock price process as

$$L_{t_i} := \inf_{t_0 \le u_i \le t_i} S_{u_i}, \quad H_{t_i} := \sup_{t_0 \le u_i \le t_i} S_{u_i},$$

where $t_i = t_0, \ldots, t_T$. The payoffs of different types of barrier call option are then given by

Type	**Down-and-out**	**Down-and-in**	**Up-and-out**	**Up-and-in**
Payoff	$(S_{t_T} - K)^+ \mathbf{1}_{\{L_{t_T} > B\}}$	$(S_{t_T} - K)^+ \mathbf{1}_{\{L_{t_T} \le B\}}$	$(S_{t_T} - K)^+ \mathbf{1}_{\{H_{t_T} < B\}}$	$(S_{t_T} - K)^+ \mathbf{1}_{\{H_{t_T} \ge B\}}$

Table 3.1: Payoffs of different types of barrier Call options

For barrier put options, $(S_{t_T} - K)^+$ is replaced by $(K - S_{t_T})^+$. This product is a cheaper alternative to the simple European option since the payoff is only activated when the barrier level has (not) been hit.

We focus on Down-and-out Call option and apply the DCKE algorithm. Note that for such kind of path-dependent options, we simulate the paths on a finer mesh in order to determine whether the option has knocked out, and then extract and plot a couple of representative time steps to compare with the closed-form solution.

3.1.1 Continuity correction

As we are modelling discretely in time, in general, there are no easily computed closed-form expressions for the prices of these discrete barrier options. A consequence of our analysis is the obvious conclusion that the discrete price converges to the continuous price as the monitoring frequency increases, suggesting that the continuous price may be used as a naive approximation. The following theorem according to Broadie et al. (1997, pg 327) shows how to adjust the continuous formula to obtain a greater approximation to the discrete price.

Theorem 3.1.1. *Let $V^*(B)$ be the price of a discretely monitored knock-in or knockout down call or up put with barrier B. Let $V(B)$ be the price of the corresponding continuously monitored barrier options. Then*

$$V^*(B) = \begin{cases} V\left(Be^{+\beta\sigma\sqrt{T/m}}\right) + o\left(\frac{1}{\sqrt{m}}\right) & \text{if } B > S_{t_0} \\ V\left(Be^{-\beta\sigma\sqrt{T/m}}\right) + o\left(\frac{1}{\sqrt{m}}\right) & \text{if } B < S_{t_0}, \end{cases} \tag{3.1.1}$$

where $\beta = -\zeta\left(\frac{1}{2}\right)/\sqrt{2\pi} \approx 0.5826$, with ζ the Riemann zeta function, and m is the number of simulated time steps.

The theorem indicates that to use the continuous price as an approximation to the discrete price, we should first shift the barrier away from S_{t_0} by a factor of $e^{\beta\sigma\sqrt{\Delta t}}$. Compared with using the unadjusted continuous price, this correction reduces the pricing error from $O(1/\sqrt{m})$ to $o(1/\sqrt{m})$ as the number of time steps m increases.

Numerical results in Section 3.1.3 suggests a significant improvement in approximation accuracy after we take the barrier adjustment into account.

3.1.2 Brownian bridge

Since we are dealing with path-dependent options, we no longer utilize the exact simulations for underlying spots. In this case, we apply Brownian bridge discretisation (BBD) scheme that outperforms the standard discretisation (SD) scheme of Black-Scholes model for quasi-Monte Carlo (QMC) methods, as described below.

Recall that the discretisation of the Black-Scholes solution leads to

$$S^j_{t_{i+1}} = S^j_{t_i} \exp\left(\left(r - \frac{1}{2}\sigma^2\right)(t_{i+1} - t_i) + \sigma\left(W_{t_{i+1}} - W_{t_i}\right)\right),$$

with $S^j_{t_i}, j = 1, \ldots, n$ and $i = 1, \ldots, n$ starting at $S^j_{t_0}$. Equation (1.1.4) is the SD scheme of the solution. Different from SD which generates the Brownian motion sequentially across time steps, in the alternative BBD scheme we use different orderings. The first value of the Brownian motion is initialised to zero and the terminal value is generated from the first variate, while subsequent variates are used to generate intermediate points, conditioned on the points that are already simulated. We apply bisected recursion on each path to determine the values at each intermediate steps according to the following formula:

$$W^j_{t_0} = 0 \tag{3.1.2}$$

$$W^j_{t_T} = \sqrt{t_T - t_0}\,\theta_{1,j} \tag{3.1.3}$$

$$W^j_{t_i} = \frac{t_{i+k} - t_i}{t_{i+k} - t_{i-k}} W_{i-k} + \frac{t_i - t_{i-k}}{t_{i+k} - t_{i-k}} W_{i+k} + \sqrt{\frac{(t_{i+k} - t_i)(t_i - t_{i-k})}{t_{i+k} - t_{i-k}}}\,\theta_{l,j}, \tag{3.1.4}$$

where $i = 1, \ldots, T$, $l = 2, \ldots, T$, $k = \frac{T}{2^n}$ and $n = 1, \ldots, \log_2 T$.

Due to different orderings of Brownian motion generation, the stochastic part of Equation (3.1.2) associated with smaller variance than the discretisation of Brownian motion in SD scheme under the same number of time steps, and the first few points in BBD scheme contain most of the variance. Both schemes have the same total variance under Geometric Brownian Motion, which leads to the same Monte Carlo convergence rates. Nevertheless, as proved by Bianchetti et al. (2015), QMC sampling gives outperforming efficiencies for BBD than SD, which shows faster and more stable convergence to exact or almost exact results including the highest-dimensional simulations and Greeks calculations.

3.1.3 Practical experiment

Note that since the terminal payoff of barrier options consists of a non-differentiable indicator function, the Dominated Convergence Theorem (DCT) which enables us to interchange the order of differentiation and expectation in Equation (2.2.8) no longer applies. As a consequence, the

kernelised pathwise derivative exists with probability one but is entirely uninformative. Hence, we use the variance minimisation solution as Equation (2.1.9) to estimate deltas, and then plug into the estimation of prices.

We follow the general features listed in Table 3.2 in our numerical experiment and compare our estimates with the closed form solution according to Fei (n.d.). We first set the barrier level to be 95, which is close to but not coincides with the initial stock price 100. Additionally, we set the number of time steps $m = 128$ to control the error term in Equation (3.1.1).

General features	
Initial stock price S_{t_0}	100
Strike price K	100
Barrier level B	95
Risk-free rate r	5% p.a.
Volatility σ	40% p.a.
Maturity time t_T	1 year
Terminal value P	$\mathbb{1}_{\left\{(\inf_{t_0 \le u_i \le t_i} S_{u_i}) > B\right\}} \max\left(S_{t_T} - K, 0\right)$

Table 3.2: Example barrier Call option

From Figure 3.1, after the adjustment of barrier level, the price estimates under all of GBM, Sobol sequence, and Sobol with Brownian bridge discretisation almost perfectly lie on the closed-form solutions to down-and-in barrier options. As we already set up a finer mesh and use small enough lags between time steps, only tiny discrepancies happen when the underlying spot approaches the barrier level after the continuity correction. However, when we zoom in the plot of results without barrier level adjustments, obvious biases occur around the barrier level and all estimates tend to overpredict the option prices.

Figure 3.1 demonstrates the necessity of the adjustment made for the closed-form solution. However, through this adjustment, the closed-form prices around the barrier level move upwards to coincide the estimates, and hence the prices with underlying spots that are just below the barrier increases as well. As a result, the closed-form price is adjusted to be greater than zero in the cases that the options are already knocked out. To avoid such biases occur when the options are already knocked out and to compare the results obtain using GBM, Sobol and Sobol with BBD, as introduced in Section 3.1.2, respectively, we compute the mean squared errors between our estimates and the adjusted closed-form solution only for $S_{t_i} > B$ at each time step in each case, repeat for 100 times and take the average. The barrier level ranges from 81 till 99, and we also investigate how the amount of MSE reduction due to the continuity adjustment in each case changes with the increase of barrier level in Figure 3.2.

As shown in Figure 3.2, Sobol sequence with BBD indeed outperforms Sobol and GBM with standard discretisation as expected, especially in earlier time steps, due to the faster Monte Carlo convergence rate. We again verify that Sobol is a better choice than GBM. As the barrier level approaches the initial stock price 100, the estimation errors increase as well, because although we already applied the continuity correction and kept a large number of time steps, the true gap between discrete and continuous monitoring is difficult to be recovered. In other words, the approximation is accurate enough to correctly price barrier options in all but the most extreme circumstance, i.e., except when the price of the underlying asset nearly coincides with the barrier. We also observe that as the barrier level increases, the barrier adjustment is more effective in reducing the total MSE, except when the barrier and spot price nearly accord with each other. The reason is, when the barrier level is small enough, in almost no cases will the option knock out and the apparent deviations between the estimation and the closed-form solution around the barrier level no longer exist. Moreover, similar to above, the barrier adjustment becomes slightly less effective at the extreme cases while $S_{t_i} \approx B$.

Similarly, we plot the estimated deltas with $S_{t_i} > B$ at each time step together with the deltas we get by applying the central difference to the adjusted closed-form prices in Figure 3.3. We get a better estimate when the option approaches its maturity date. As the adjusted closed-form price is not well-continuously differentiable, the central difference deltas we get are not representative when $S_{t_i} \approx B$, but we could still refer to the values when the spot price is large. Also, we find the evident reduction in MSE after we adjust the true barrier level. With the increase of barrier level, the total MSE both before and after the correction increases, and so does the effect of correction.

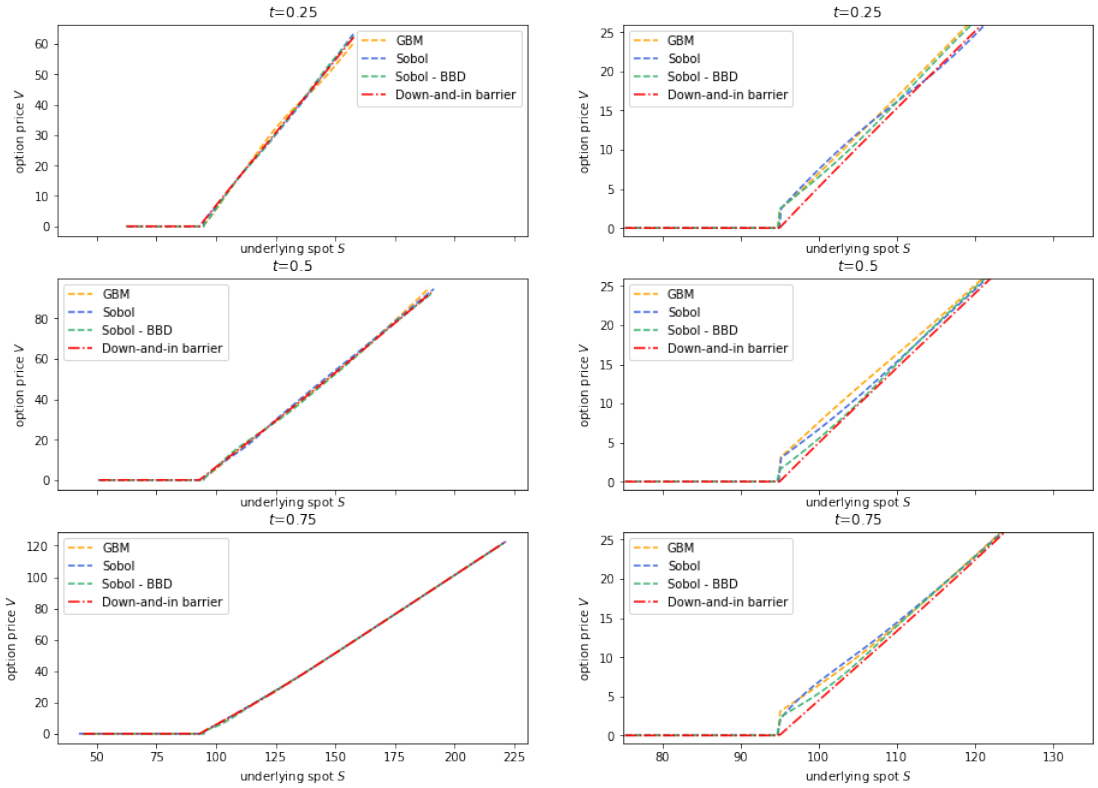

Figure 3.1: Price estimates for barrier options with continuity correction against closed form prices (left); zoomed-in without continuity correction (right)

Table 3.3 reports the improvements in MSE when we apply Sobol sequence with Brownian bridge discretisation instead of GBM, and before versus after continuity with Sobol sequence with Brownian bridge discretisation at each time step. Sobol with BBD does help with the reduction of Monte Carlo error accumulation, especially for earlier time steps and lower barrier levels. We can also find that more importantly, the continuity adjustment for barrier options increases the estimation precision universally.

	Sobol-BBD vs GBM			**Adjusted vs Unadjusted**		
Barrier	$t = 0.25$	$t = 0.50$	$t = 0.75$	$t = 0.25$	$t = 0.50$	$t = 0.75$
81	68.50%	52.28%	26.68%	53.15%	28.65%	42.61%
83	60.00%	53.92%	19.39%	59.32%	33.43%	50.02%
85	54.37%	47.54%	16.09%	63.79%	34.01%	52.62%
87	55.48%	39.82%	8.86%	66.33%	36.29%	57.69%
89	49.71%	41.14%	6.92%	65.36%	42.69%	57.13%
91	45.05%	38.94%	14.64%	65.42%	43.18%	59.50%
93	39.17%	22.79%	3.21%	62.36%	36.66%	54.49%
95	22.00%	-2.07%	-19.74%	61.85%	33.50%	52.63%
97	17.13%	4.37%	4.72%	52.36%	24.88%	45.98%
99	23.06%	-2.97%	2.59%	33.65%	23.26%	27.08%

Table 3.3: Reductions in MSE of estimated priced against closed form solutions under Sobol sequence with Bownian bridge discretisation vs GBM; before vs after continuity correction at different barrier levels with $S_{t_0} = 100$

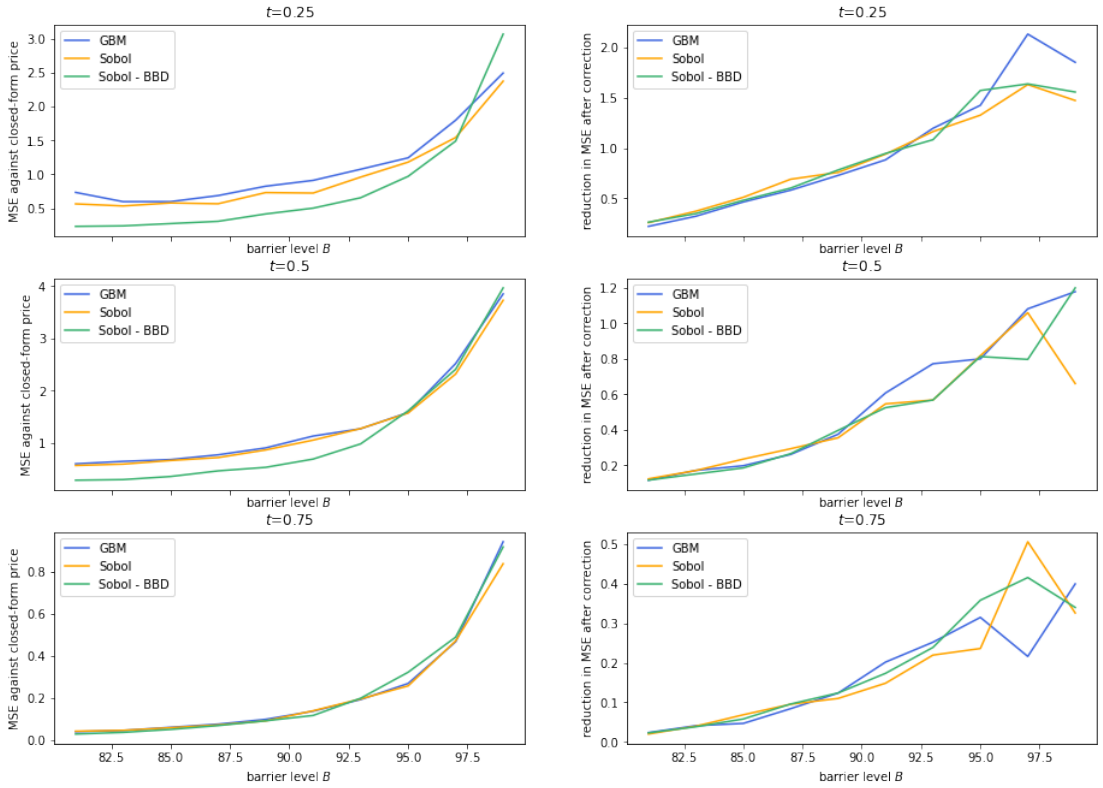

Figure 3.2: MSE comparison for GBM, Sobol, and Sobol with Brownian bridge discretisation for barrier options after continuity correction (left); reduction in MSE after continuity correction (right) (average of 100 repetitions)

3.2 Basket option

We now extend the new algorithm to higher dimensional products.

Basket options are popular multivariate derivatives in financial markets, the payoff of which depends on the weighted average of the underlying asset prices. We suppose the price of the basket option depends on the weighted arithmetic average of the prices of d assets with weights $\omega_1, \omega_2, \ldots, \omega_d$, where $\omega_u > 0$, $u = 1, \ldots, d$ and $\sum_{u=1}^{d} \omega_u = 1$. Then, the payoff of the Call option is represented by

$$V_{t_T} = \left(\sum_{u=1}^{d} \omega_u S_{ut_T} - K \right)^+, \tag{3.2.1}$$

where S_{ut_T} denotes the price of the asset u at time T. For Put options, we just interchange the two terms in the bracket.

Each entry of the d-dimensional pathwise derivative of the basket option $\phi_{t_{T-1}}$ is given by

$$\phi_{ut_{T-1}} = \omega_u \frac{S_{ut_T}}{S_{t_0}} \mathbb{1}_{\{\sum_{u=1}^{d} \omega_u S_{ut_T} > K\}}.$$

For Put options, we just replace '$>$' with '$<$'.

3.2.1 Sequences of multiple assets

We now introduce how to generate the sequences of multiple assets. The asset price sequences in Equation (3.2.1) has the expression

$$S^j_{ut_{i+1}} = S^j_{ut_i} \exp\left(\left(r - \frac{1}{2}{\sigma_u}^2 \right) (t_{i+1} - t_i) + \sigma_u \left(W_{ut_{i+1}} - W_{ut_i} \right) \right),$$

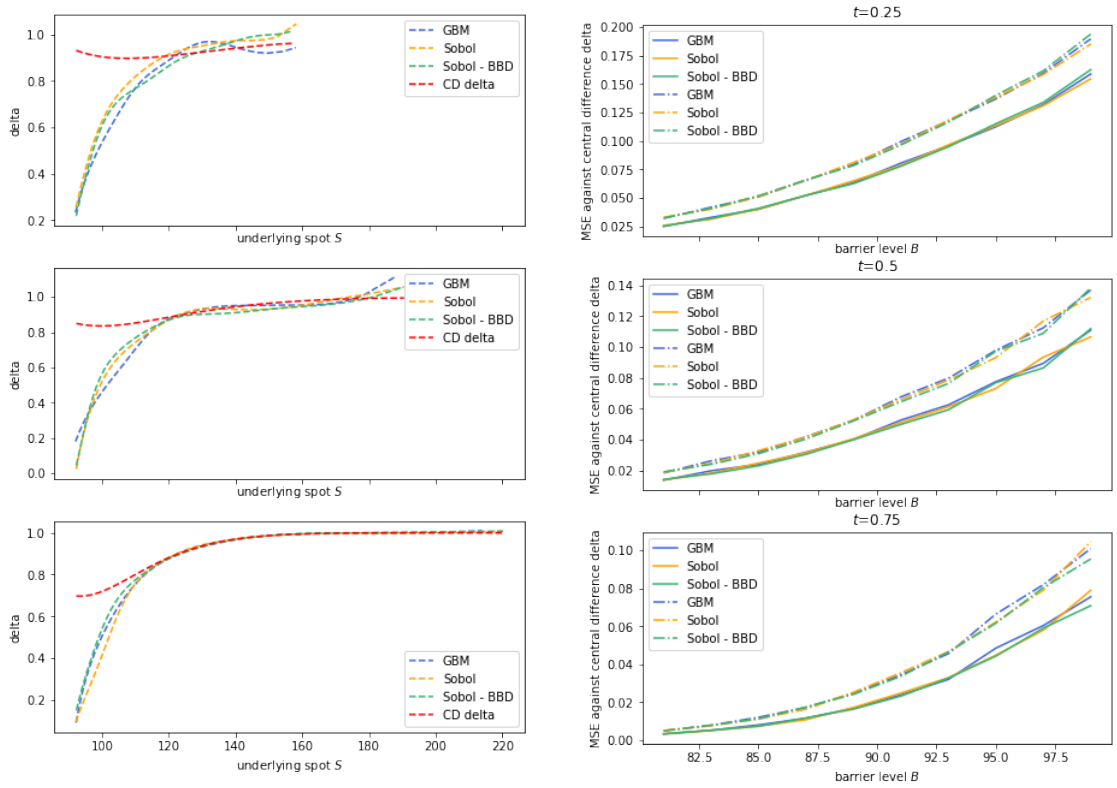

Figure 3.3: Delta estimates for barrier options with continuity correction against central difference deltas (left); MSE comparison for GBM, Sobol, and Sobol with Brownian bridge discretisation for barrier options with and without continuity correction (right): before continuity correction ('-·'); after continuity correction ('-') (average of 100 repetitions)

where $W_{ut_{i+1}} - W_{ut_i} = W_{u\Delta t} \sim N(0, \Delta t)$ are correlated standard Brownian motions with correlations ρ_{uv}, where $v = 1, \ldots, d$. We define $\boldsymbol{R}$ as the correlation matrix for the d assets with entries ρ_{uv}, and suppose $\boldsymbol{L}$ is the solution of $\boldsymbol{L}\boldsymbol{L}^T = \mathrm{R}$ obtained by Cholesky factorisation. As $(W_{1\Delta t}, \ldots, W_{d\Delta t})^T$ follows a multivariate normal distribution $\boldsymbol{N}(\boldsymbol{0}, \boldsymbol{\Sigma})$, where

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sqrt{\Delta t} & & \\ & \sqrt{\Delta t} & \\ & & \ddots \\ & & & \sqrt{\Delta t} \end{pmatrix} \begin{pmatrix} \rho_{11} & \cdots & \cdots & \rho_{1d} \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \rho_{d1} & \cdots & \cdots & \rho_{dd} \end{pmatrix} \begin{pmatrix} \sqrt{\Delta t} & & \\ & \sqrt{\Delta t} & \\ & & \ddots \\ & & & \sqrt{\Delta t} \end{pmatrix} = \Delta t \boldsymbol{R}$$

Note that if $\xi = [\xi_1, \xi_2, \ldots, \xi_d]^T \sim \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{I})$, where $\xi_1, \ldots, \xi_d$ are identically and independent standard normal distributions, then $\boldsymbol{L}\xi \sim \boldsymbol{N}\left(\boldsymbol{0}, \boldsymbol{L}\boldsymbol{L}^{\boldsymbol{T}}\right)$ and moreover,

$$\sqrt{\Delta t}\boldsymbol{L}\xi \sim N\left(\boldsymbol{0}, \Delta t \boldsymbol{L}\boldsymbol{L}^T\right) = N(\boldsymbol{0}, \Delta t \boldsymbol{R}) = N(\boldsymbol{0}, \boldsymbol{\Sigma}),$$

which indicates $\sqrt{\Delta t}\boldsymbol{L}\xi$ can be used to replace $(W_1(T), \ldots, W_d(T))$ in the simulation procedure. The uth element of the vector $\boldsymbol{L}\xi$ can be written as $\sum_{v=1}^{u} L_{uv}\xi_v$ because the correlation matrix $\boldsymbol{R}$ is symmetric and hence, $\boldsymbol{L}$ is lower triangular. Consequently, the stock price for asset u at time t_{i+1} can be written as the form

$$S^j_{ut_{i+1}} = S^j_{ut_i} \exp\left(\left(r - \frac{1}{2}\sigma_u^2\right)\Delta t + \sigma_u \sqrt{\Delta t} \sum_{v=1}^{u} L_{uv}\xi_v\right).$$

3.2.2 Algorithm - Multi-dimensional Dynamically Controlled Kernel Estimation

Algorithm 3 describes the multi-dimensional version of the DCKE algorithm in pseudo-code as an extension to Algorithm 2. Algorithm 4 shows three different approaches for the estimation of basket options and rainbow options, which will be introduced in Section 3.3, making use of Algorithm 2 and 3.

Section 3.2.3 gives further explanations about how the algorithms are applied in the practical experiment.

Algorithm 3 Multi-dimensional Dynamically Controlled Kernel Estimation

Input: d: dimension of assets; r: risk-free rate; Δt: lag between current time step and maturity; n: number of simulated paths; u: number of meshed samples; S: d-dimensional asset price sequences S_{t_i} with each length n for $i = 0, \ldots, 2$; S^*: validation asset price sequences $S^*_{t_i}$; h_p: hyperparameter input for bandwidth adjustment of price; h_d: hyperparameter input for bandwidth adjustment of delta; C: cap of variable kernel ratio (usually 1-30)

Output: X: mesh points of spot price sequence X_{t_i}; $\hat{V}^*_{t_1}$: option prices sequence with length n

Set arrays: $V_{t_2} = P(\cdot)$: payoff function; ϕ_{t_1}: payoff derivative (from Equation (3.3.1)); $\hat{V}^*_{t_1}$
$X_{t_1} =$ samples between $1\% - 99\%$ percentiles of S_{t_1} via *SobolEngine*
$X^*_{t_1} =$ samples between $1\% - 99\%$ percentiles of $S^*_{t_1}$ via *SobolEngine*
$\Sigma = cov(S_{t_2} - S_{t_1})$
$L^\circ = Ln^{-\frac{1}{d+4}}$ s.t. $LL^T = \Sigma^{-1}$ (according to Section 2.2.1.3)
$L^\circ_p = L^\circ * h_p$
$L^\circ_d = L^\circ * h_d$
for each sample point $k = 1$ up to $k = u$ **do**
$K^k_{pt_1} = \sum_{j=1}^{n} e^{-\sum_{v=1}^{d}(S^j_{vt_2} - tile(X^k_{vt_2}))L^{\circ 2}_p}$ (*tile*: convert $X^k_{vt_2}$ into the same dimension as $S^j_{vt_2}$ by repeating each entry for $\frac{n}{u}$ times)
end for
$K^* = max(K^k_{pt_1})$ for $k = 1, 2, \ldots, u$
for each mesh point $k = 1$ up to $k = u$ **do**
$K^k_{ratio} = min(\frac{K^*}{K^k_{t_1}}, C)$
$K^k_{pt_1} = e^{-\sum_{v=1}^{d}(S^j_{vt_2} - tile(X^k_{vt_2}))L^{\circ 2}_p} * K^k_{ratio}$
$K^k_{dt_1} = e^{-\sum_{v=1}^{d}(S^j_{vt_2} - tile(X^k_{vt_2}))L^{\circ 2}_d} * K^k_{ratio}$
$\hat{V}^k_{t_2} = \frac{\sum_{j=1}^{n} K^k_{pt_1} V_{t_2}}{\sum_{j=1}^{n} K^k_{pt_1}}$
$\hat{S}^k_{t_2} = \frac{\sum_{j=1}^{n} tile(K^k_{pt_1}) S_{t_2}}{\sum_{j=1}^{n} K^k_{pt_1}}$
$\hat{\phi}^k_{t_1} = \frac{\sum_{j=1}^{n} tile(K^k_{dt_1}) \phi_{t_1}}{\sum_{j=1}^{n} K^k_{dt_1}}$
end for
$\hat{V}^*_{t_1} = e^{-r\Delta t}$GPR(RBF, X_{t_1}, $\hat{V}_{t_2} - \hat{\phi}_{t_1}(\hat{S}_{t_2} - e^{r\Delta t} X_{t_1}); X^*_{t_1}$)
return $\hat{V}^*_{t_1}$

3.2.3 Practical experiment

We use high-dimensional Sobol low discrepancy sequences in our simulation due to its higher efficiency compared to geometric Brownian motion as shown in Section 2.3.2.

Since there is no closed form solution for the price of basket options, we use nested Monte Carlo as the evaluation benchmark. We start with the simplest case when there are two assets in total. In order to make a more robust bandwidth choice for the kernel estimation, we create a validation set and employ the dedicated hyperparameter optimization Python library *Hyperopt* to obtain an optimal bandwidth among uniformly distributed samples on a constrained two-sided interval for the price and delta estimates, respectively by minimising the validation error. We

General features	
Initial stock price S_{t_0}	100
Strike price K	100
Number of assets d	2
Asset weight ω_u	0.5
Risk-free rate r	0% p.a.
Volatility σ_u	20% p.a.
Correlation ρ_{uv}	0.2
Maturity time t_T	1 year
Terminal value P	$max\left(\sum_{u=1}^{d} \omega_u S_{ut_T} - K, 0\right)$

Table 3.4: Example basket Call option

keep 20,000 simulation paths, use 2,000 paths for nested MC and 2,000 validation observations to test the model performance. Similar to before, we extract the 1% to 99% quantiles of each underlying asset for both the training and validation portion, and obtain 100 sample points in each dimension by drawing random quantiles from *SobolEngine*, since the dimensionality of the assets is too high to apply the simple meshed method as before.

Figure 3.4 displays the difference between option prices obtained from nested MC and the results of the raw kernel estimates with and without the inclusion of control variates when we apply multi-dimensional kernel regression. Firstly, it can be observed that we get more robust estimates for the price when the option is closer to maturity no matter if control variates are switched on. Also, it is obvious that by adding control variates, the estimation errors have been significantly reduced compared with nested MC results. It is worth noted that when two assets prices differ a lot and when the option is around-the-money, consistent deviations exist especially for earlier time steps. Similar results are obtained when we fit the price surface using multi-dimensional GPR, applying onto the raw kernel estimates, as shown in the left plot in Figure 3.5. However, such problem is partially solved when we reduce the strike price to 70 according to the right plot, but in this case, the 'tails' are not well-estimated.

So far, we applied the multi-dimensional DCKE algorithm, which consists of the application of high-dimensional kernels, control variates and GPR. Another way to deal with the pricing problem of basket options could be concatenating multiple assets into a single dimension by computing the weighted average, and repeat the basic DCKE approach. Figure 3.6 shows that when the strike is 100, the predicted price has even greater errors when there is a huge gap between two asset prices and when the option is around the money. However, when the option is more in-the-money ($K = 70$), this approach gives perfect fits except the extreme 'tails'.

We finally use multi-dimensional GPR directly onto nested MC results. Not surprisingly, we get tiny difference compared to nested MC benchmarks, as plotted in Figure 3.7. However, it is worth investigating how will this method perform for higher-dimensional options.

We create Figure 3.8 and 3.9 to observe how each model performs with the change of number of assets. First of all, the multi-dimensional raw kernel fit gives similar MSE to the model in which we further include GPR. With the increase of the number of assets, the inclusion of GPR makes the model more robust, especially around the 'tails'. When time approaches maturity, the compressed 1D model outperforms the high-dimensional models, even the pure GPR model on nested MC, in terms of MSE when the number of asset is greater than 10, and in terms of smaller absolute errors around both two 'tails' with almost the whole set of asset numbers. Nevertheless, such remarkable performance is not consistent across time steps. As time comes to $t = 0.3$, i.e., far from maturity, the 1D model has largest overall prediction errors and 'tails' deviations against nested MC among the four models under comparison, whereas the multi-dimensional GPR applied to nested MC gives most precise estimates.

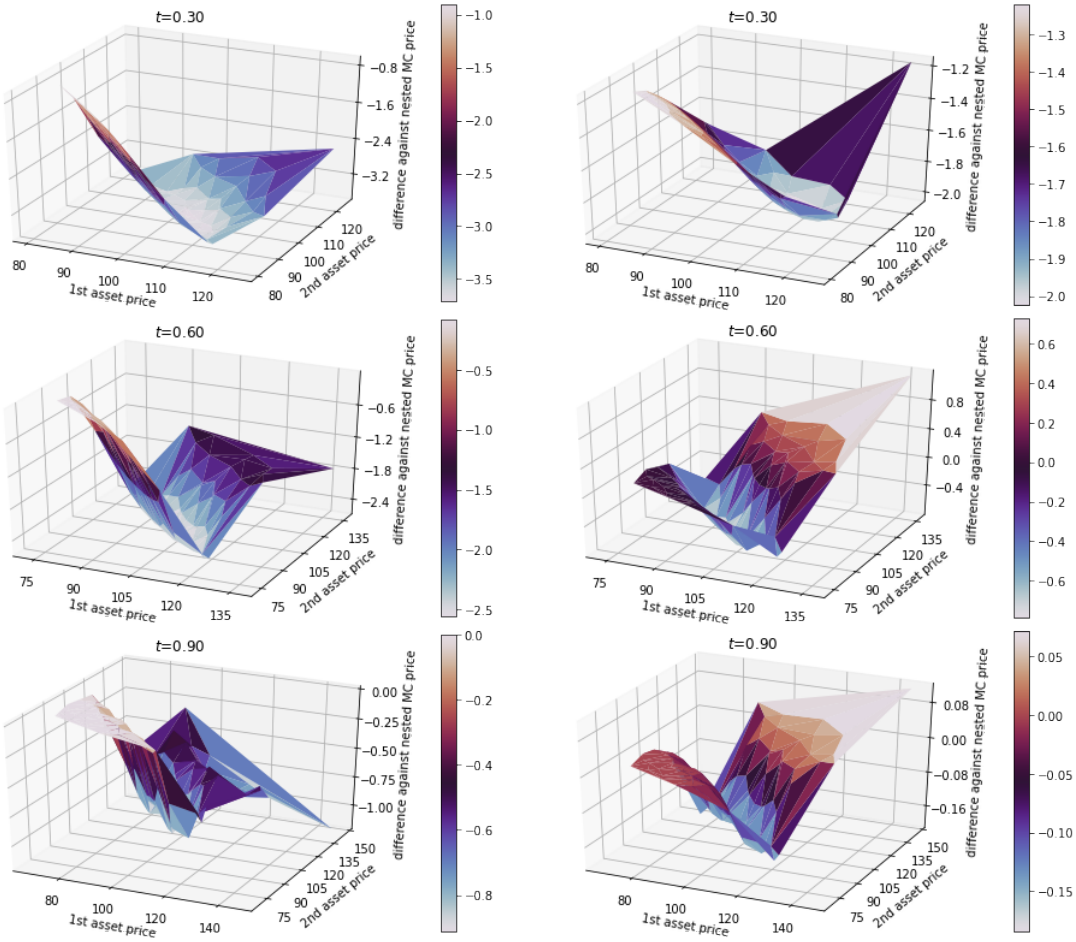

Figure 3.4: Raw kernel estimates of prices for 2D basket option without control variates (left); with control variates (right) minus nested MC results

3.3 Rainbow option

3.3.1 Generalisation

Like the basket option we introduce above, which is written on a group of assets and pays out on a weighted-average gain on the basket as a whole, a rainbow option also considers a group of assets, but usually pays out on the level of one of them depending on their performance at maturity. Rainbow options are usually calls or puts on the best or worst of d underlying assets of the basket. When the rainbow only pays the best (worst) performing asset, it is also called best-of (worst-of). Other popular options that can be reformulated as a rainbow option are spread and exchange options.

Type	**Best-of**	**Worst-of**
Payoff (%)	$\left(max(\frac{S_{1t_T}}{S_{1t_0}}, \ldots, \frac{S_{dt_T}}{S_{dt_0}}) - K\right)^+$	$\left(min(\frac{S_{1t_T}}{S_{1t_0}}, \ldots, \frac{S_{dt_T}}{S_{dt_0}}) - K\right)^+$

Table 3.5: Payoffs (%) of different types of rainbow Call options

Table 3.5 shows that a worst-of call option is exercised only if all underlying instruments have performed better than the strike price, whereas a best-of call option is exercised if at least one underlying instruments have performed better than the strike. For Put options, we just exchange the order of the two terms in the bracket. Note that the strike here is defined as a percentage. It must be a positive number and can be over 100%. The strike is compared with the best (worst) performed asset's yield (percentage ITM), and the payout is then the notional multiplied by the

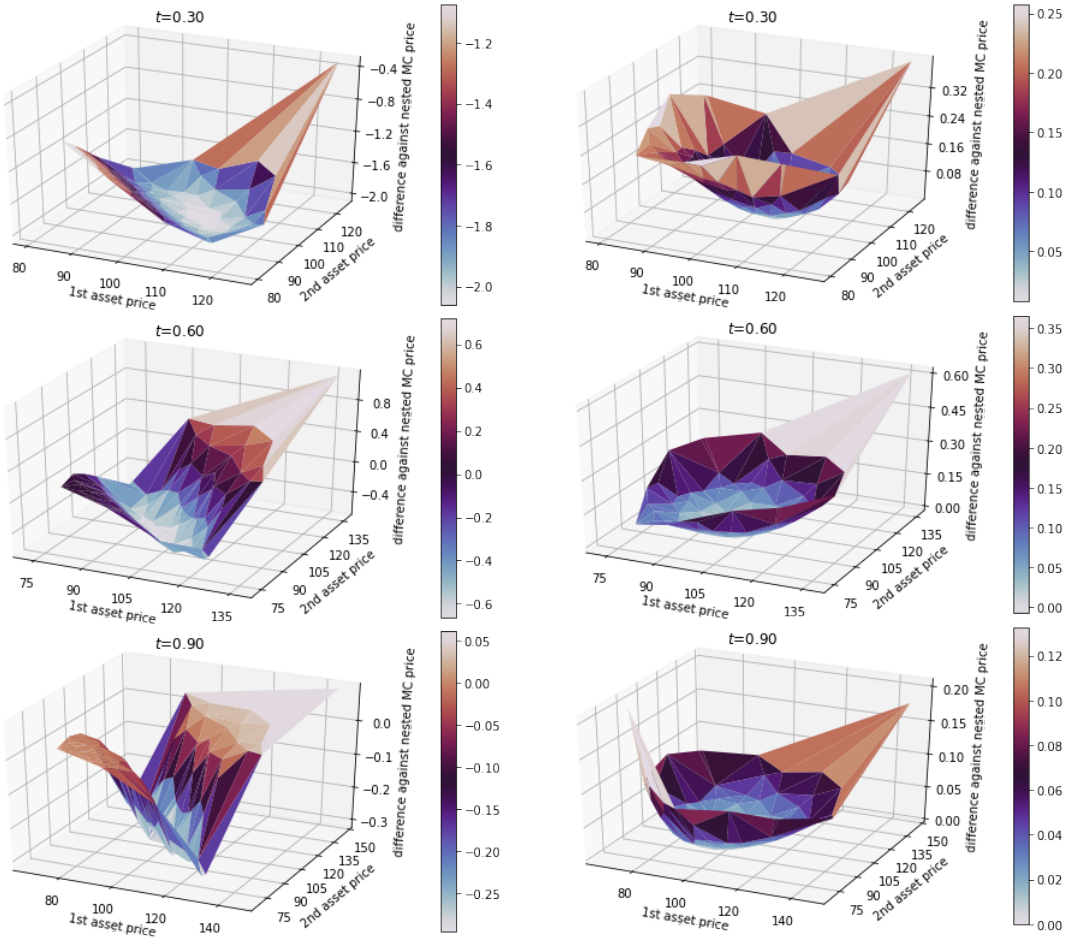

Figure 3.5: Price estimates with 2D kernel and GPR for 2D basket option with control variates: $K = 100$ (left); $K = 70$ (right)

payoff percentage.

Type	**Best-of**	**Worst-of**
Pathwise derivative	$max(\frac{S_{1t_T}}{S_{1t_0}}, \ldots, \frac{S_{dt_T}}{S_{dt_0}})$	$min(\frac{S_{1t_T}}{S_{1t_0}}, \ldots, \frac{S_{dt_T}}{S_{dt_0}})$

Table 3.6: Pathwise derivative of different types of 1D rainbow options

We define both d-dimensional and 1-dimensional pathwise derivatives, which is helpful for the later experiment, in which we compare models of different dimensions. Each entry of the d-dimensional pathwise derivative of the rainbow option $\phi_{t_{T-1}}$ is given by

$$\phi_{ut_{T-1}} = \begin{cases} \frac{1}{K_v}(\max(\frac{S_{1t_T}}{S_{1t_0}}, \ldots, \frac{S_{dt_T}}{S_{dt_0}}) - K)\mathbf{1}_{\{\max(\frac{S_{1t_T}}{S_{1t_0}}, \ldots, \frac{S_{dt_T}}{S_{dt_0}}) > K\}} & \text{if } v = \underset{u \in \{1,\ldots,d\}}{\operatorname{argmax}}(\frac{S_{ut_T}}{S_{ut_0}}) \\ 0 & \text{otherwise} \end{cases} \tag{3.3.1}$$

For worst-of options, we replace 'max' with 'min', and for Put options, replace '$>$' with '$<$'.

The 1-dimensional pathwise derivative of the rainbow option is listed in Table 3.6.

3.3.2 Practical experiment

Following the general features in Table 3.4, we specify some additional settings in Table 3.7.

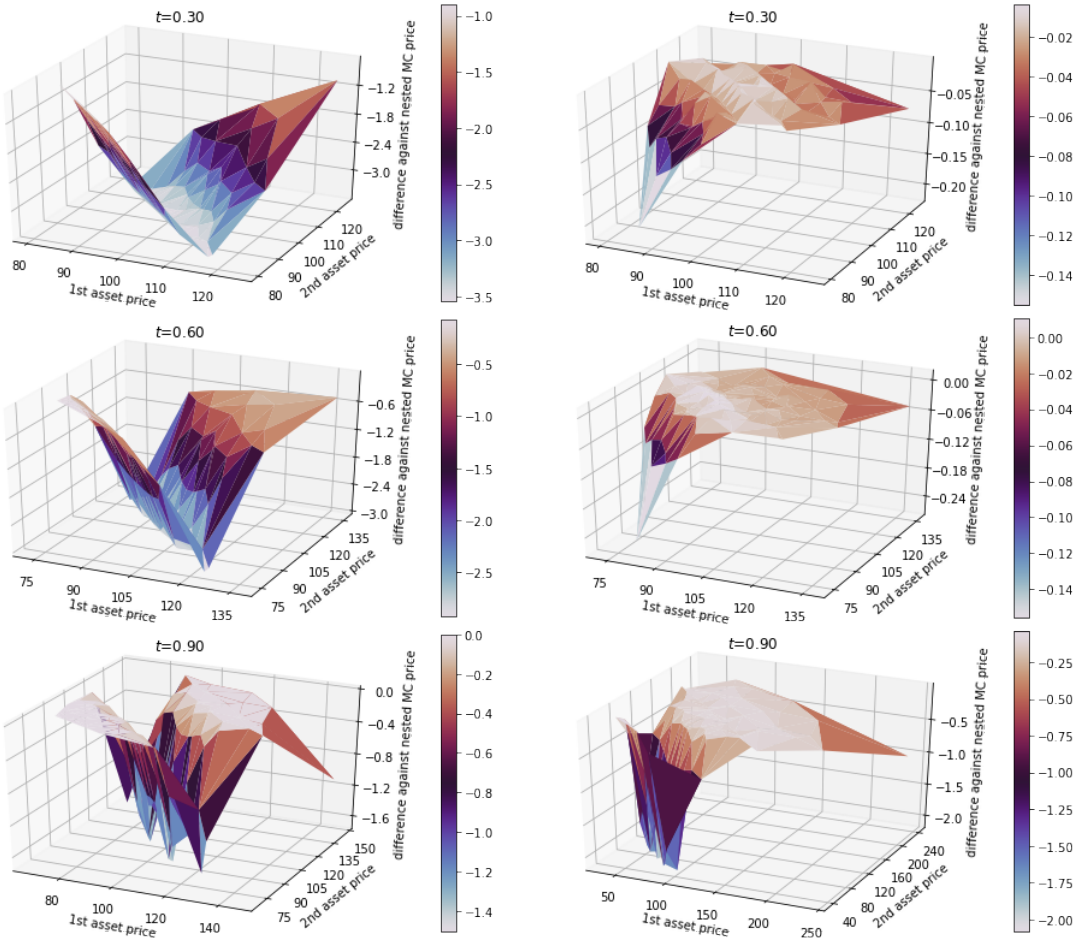

Figure 3.6: Price estimates with 1D kernel and GPR for weighted 1D basket option with control variates: $K = 100$ (left); $K = 70$ (right)

General features	
Initial stock price S_{ut_0}	100
Strike price K_u	100
Basket strike price K	90% p.a.
Terminal value P (%)	$max\left(max(\frac{S_{1t_T}}{S_{1t_0}}, \ldots, \frac{S_{dt_T}}{S_{dt_0}}) - K, 0\right)$

Table 3.7: Example best-of Call option

Same as when we investigate basket options, we try d-dimensional DCKE approach, 1-dimensional DCKE algorithm that only includes the best (worst) performed asset as input, and direct d-dimensional GPR to fit the nested MC results. Figure 3.10 shows the predicted payout for best-of-two and worst-of-two options against the ratio of each asset price to its initial price when we use the outperformed model with time close to maturity in the estimation of the basket option above. The best-performed asset of two can achieve effectively 30% in the money, whilst the worst-performed asset only has a maximum payoff proportion of around 10%.

From now on, we only focus on best-of option by convention. We plot the difference between the predicted payout against nested MC results for the 1-dimensional model and the direct GPR model, respectively in Figure 3.11. We can find when $M = 2$, both models give promising estimates, and by applying GPR directly onto nested MC prices, smaller errors are associated and hence becomes more robust.

To get a better understanding of the performance of each model with different numbers of assets in a basket, we refer to Figure 3.12 where the predictions are plotted against nested MC. There might be some less-than-zero predictions, and will be floored to zeros in the real application. We

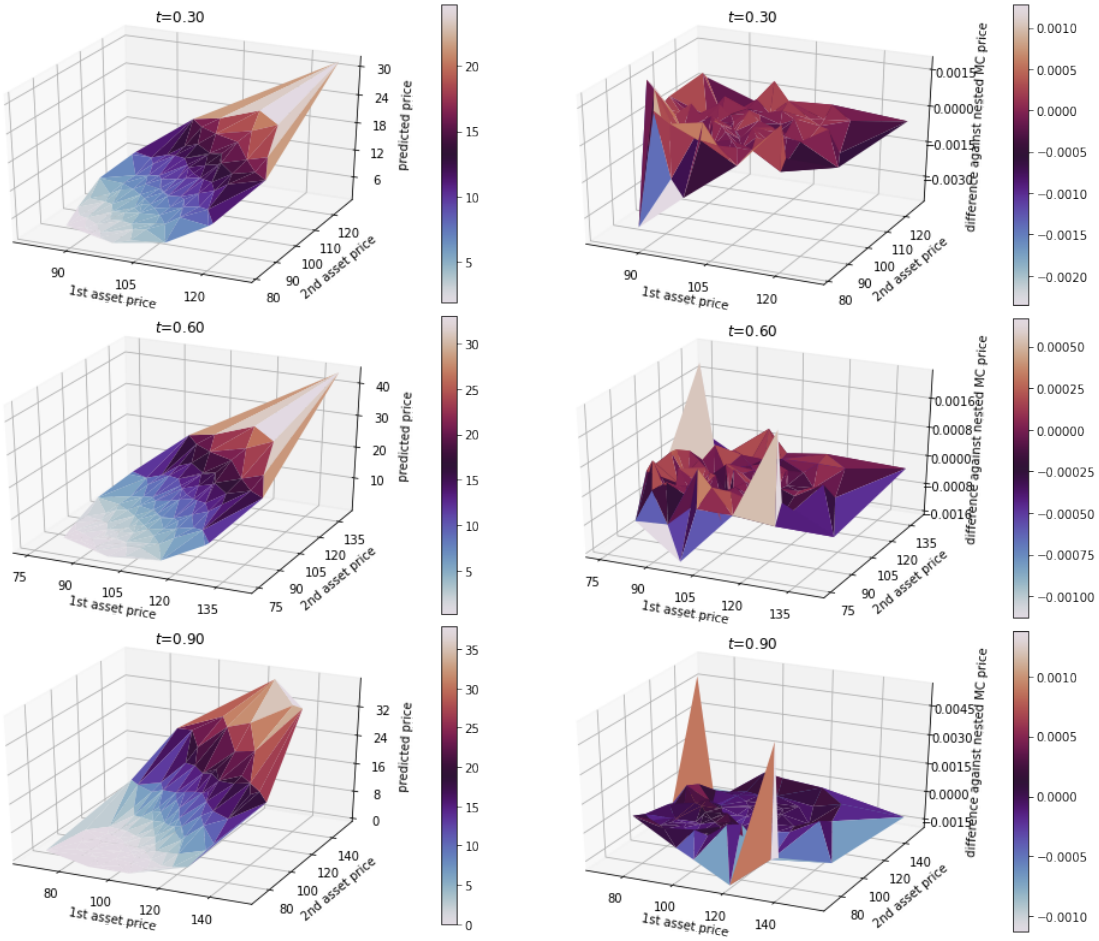

Figure 3.7: Price estimates with direct GPR on nested MC (left); predicted results minus nested MC results (right)

can obtain the following key conclusions:

Remark 3.3.1. When the number of asset is small, 5 in our case, three models show indistinguishable performance.

Remark 3.3.2. Unlike the nearly perfect performance of the direct GPR approach according to Figure 3.11 when $d = 2$, the predictions starts to get far away from the benchmark when 5 assets are considered, and becomes even worse when the number of assets further increases.

Remark 3.3.3. High-dimensional models, even the one applying GPR directly on nested MC results gives terrible price estimates for the high-dimensional options, while the predictions of the 1-dimensional DCKE keep staying around the line that represents '$prediction = nestedMC$'. It follows that the high-dimensional kernel and GPR approach performs even worse due to the multi-dimensional bandwidth selection problems. This fully reveals the curse of dimensionality problems.

Therefore, it is evident that the 1-dimensional model works effectively when time is close to maturity for rainbow options. It is actually not too surprising since the payoff depends on the best-performed asset at the maturity, and by applying 1-dimensional DCKE model, we substitute in the best-performed asset for the prediction whereas the high-dimensional models contain less informative inputs.

To be more precise, we repeat 100 times and plot the MSE and PFE for the 1-dimensional model and direct GPR approach with the change of number of assets in Figure 3.13 and 3.14. The 1-dimensional DCKE estimate gives smaller MSE and 1% quantile absolute errors than the direct GPR approach when the asset number is large and when time is close enough to maturity. However, for earlier time steps, the direct sampling model is more reliable. A possible improvement

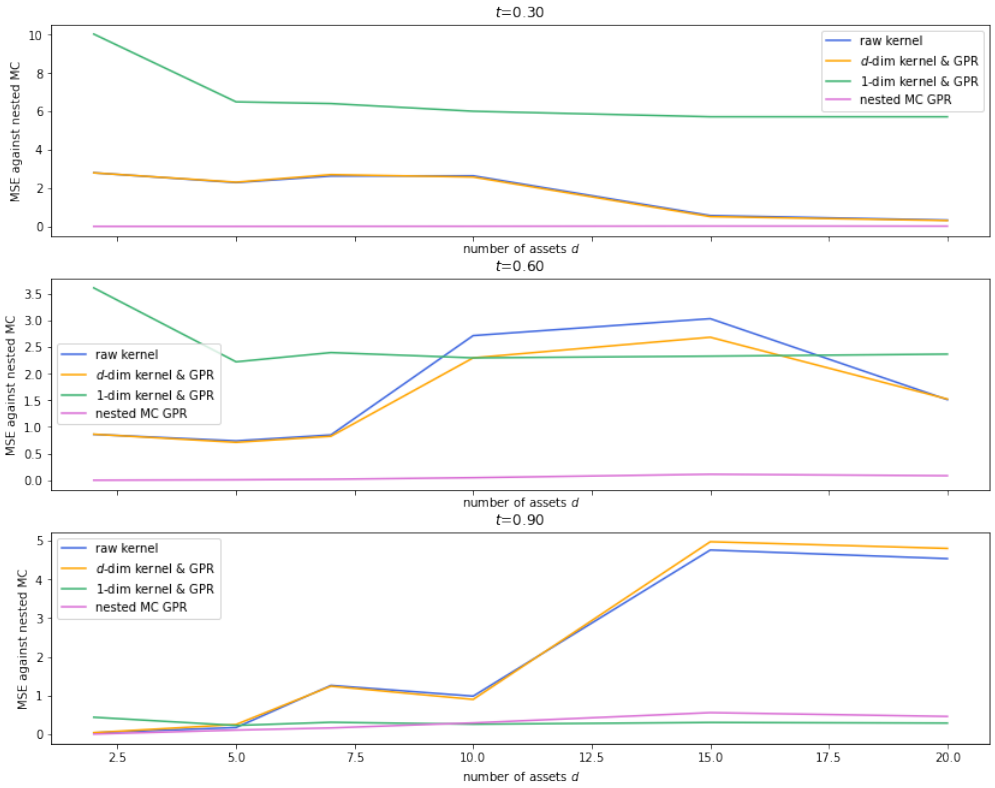

Figure 3.8: Mean squared errors of estimated basket option prices against nested MC under different models with 2, 5, 7, 10, 15, 20 assets (average of 100 repetitions)

could be made for the 1-dimensional DCKE model might be applying backward recursion on the predicted price and keep the time step size to be small enough.

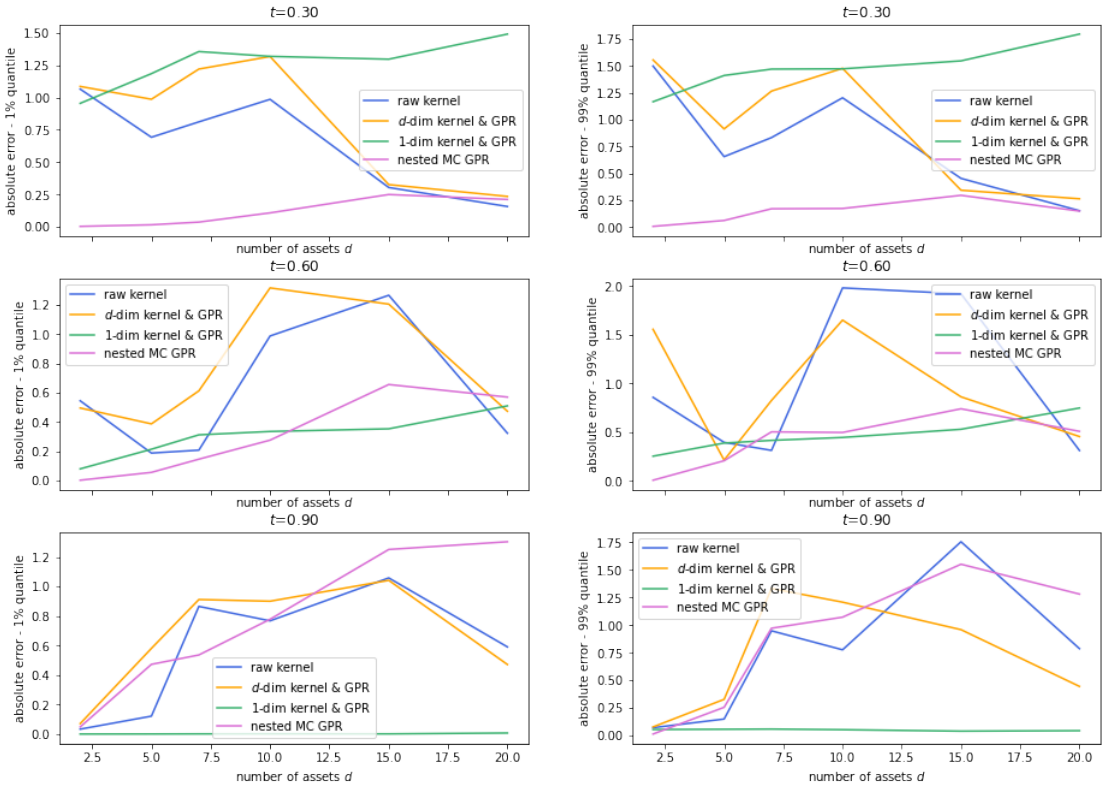

Figure 3.9: Absolute errors in 1% (left) and 99% (right) quantiles of estimated basket option prices against nested MC under different models with 2, 5, 7, 10, 15, 20 assets (average of 100 repetitions)

Algorithm 4 Dynamically Controlled Kernel Estimation on Multi-dimensional Assets

Input: d: dimension of assets; W: sequence of basket option weights of d assets; K: sequence of strikes of d assets; K_r: rainbow option strike (%); r: risk-free rate; Δt: lag between current time step and maturity; n: number of simulated paths; u: number of meshed samples; S: d-dimensional asset price sequences S_{t_i} with each length n for $i = 0, \ldots, 2$; S^*: validation asset price sequences $S^*_{t_i}$; C: cap of variable kernel ratio (usually 1-30)

Output: X: mesh points of spot price sequence X_{t_i}; $\hat{V}^*_{t_1}$

Set arrays: $V^*_{t_2}$: validation payoff; ϕ_{t_1}: payoff derivative (from Equation (3.3.1)); $\hat{V}^*_{t_1}$
$X_{t_1} =$ samples between $1\% - 99\%$ percentiles of S_{t_1} via *SobolEngine*
$X^*_{t_1} =$ samples between $1\% - 99\%$ percentiles of $S^*_{t_1}$ via *SobolEngine*

⚠ **1-dim kernel & GPR approach**

if basket option **then**
$S^W_{t_1} = [\Sigma^d_{v=1} S^1_{vt_1} W_v, \ldots, \Sigma^d_{v=1} S^n_{vt_1} W_v]$
$S^W_{t_2} = [\Sigma^d_{v=1} S^1_{vt_2} W_v, \ldots, \Sigma^d_{v=1} S^n_{vt_2} W_v]$
$\phi^W_{t_1} = [\Sigma^d_{v=1} \phi^1_{vt_1} W_v, \ldots, \Sigma^d_{v=1} \phi^n_{vt_1} W_v]$
end if
if rainbow (best-of) option **then**
$S^W_{t_1} = [max(\frac{S^1_{1t_1}}{K_1}, \ldots, \frac{S^1_{dt_1}}{K_1}), \ldots, max(\frac{S^n_{dt_1}}{K_d}, \ldots, \frac{S^n_{dt_1}}{K_d})]$
$S^W_{t_2} = [max(\frac{S^1_{1t_2}}{K_1}, \ldots, \frac{S^1_{dt_2}}{K_1}), \ldots, max(\frac{S^n_{dt_2}}{K_d}, \ldots, \frac{S^n_{dt_2}}{K_d})]$
$\phi^W_{t_1}$ from Tabel 3.6
end if
do Algorithm 2 with $S_{t_T} = S^W_{t_2}$; $S_{t_{T-1}} = S^W_{t_1}$; $\hat{\phi}_{t_1} = \phi^W_{t_1}$; $K = K_r$ (rainbow option)
$h^*_p, h^*_d = \underset{h_p, h_d \sim U(0.01,10)}{\text{argmin}} \|\hat{V}'^*_{t_1} - e^{-r\Delta t} V^*_{t_2}\|$ via *hyperopt* (minimise validation error)
do Algorithm 2 with $h_p = \frac{h_p}{\sqrt{h^*_p}}$ (bandwidth for price), $h_d = \frac{h_d}{\sqrt{h^*_d}}$ (bandwidth for delta)
if rainbow option **then**
for each mesh point $k = 1$ up to $k = u$ **do**
if $\hat{V}'^{*k}_{t_1} < 0$ **then**
$\hat{V}'^{*k}_{t_1} = 0$
end if
end for
end if
return $\hat{V}'^*_{t_1}$

⚠ **d-dim kernel & GPR approach**

do Algorithm 3
$h^*_p, h^*_d = \underset{h_p, h_d \sim U(0.01,10)}{\text{argmin}} \|\hat{V}^*_{t_1} - e^{-r\Delta t} V^*_{t_2}\|$ via *hyperopt* (minimise validation error)
do Algorithm 3 with $h_p, h_d = h^*_p, h^*_d$
if rainbow option **then**
for each mesh point $k = 1$ up to $k = u$ **do**
if $\hat{V}^{*k}_{t_1} < 0$ **then**
$\hat{V}^{*k}_{t_1} = 0$
end if
end for
end if
return $\hat{V}^*_{t_1}$

⚠ **direct GPR approach**

$\hat{V}^*_{t_1} = e^{-r\Delta t}$GPR(RBF, X_{t_1}, nestedMCprice(X_{t_1}); $X^*_{t_1}$)
return $\hat{V}^*_{t_1}$

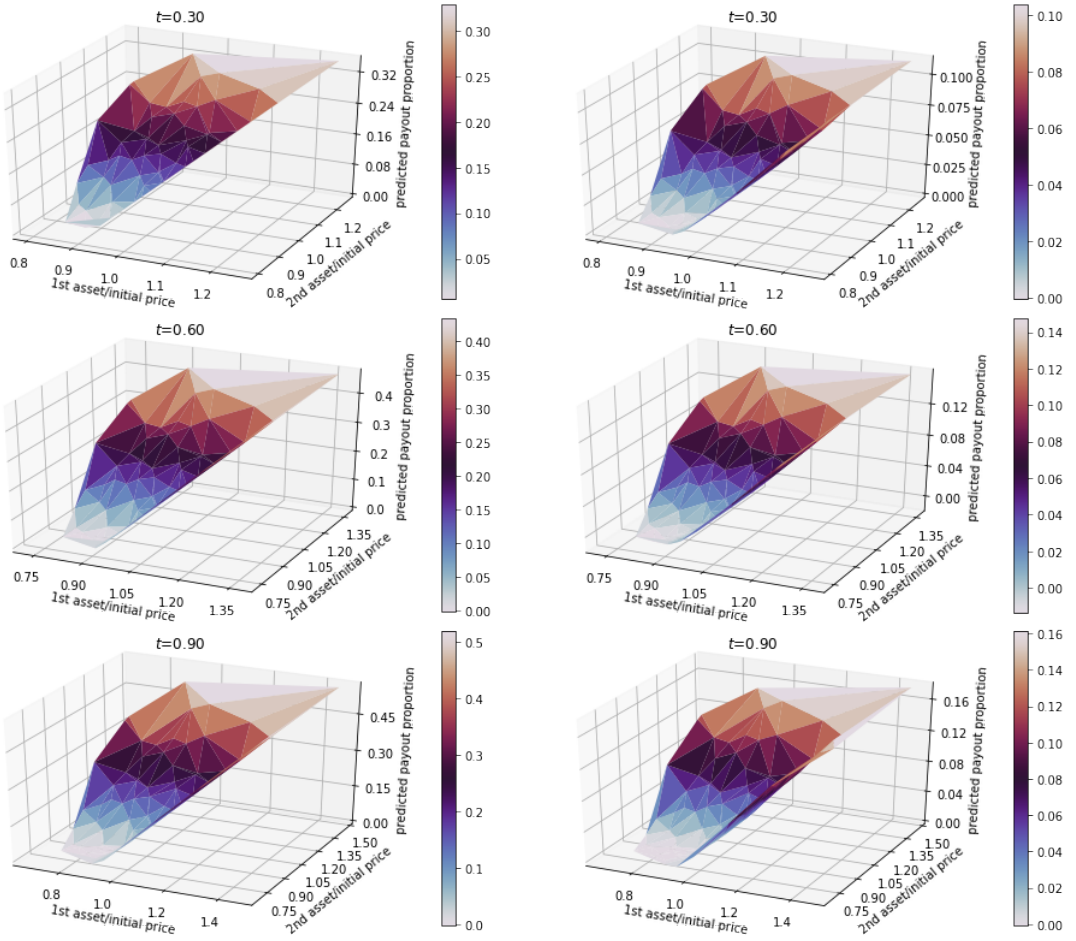

Figure 3.10: Price estimates with 1D kernel and GPR for best-of-two option (left); worst-of-two option (right)

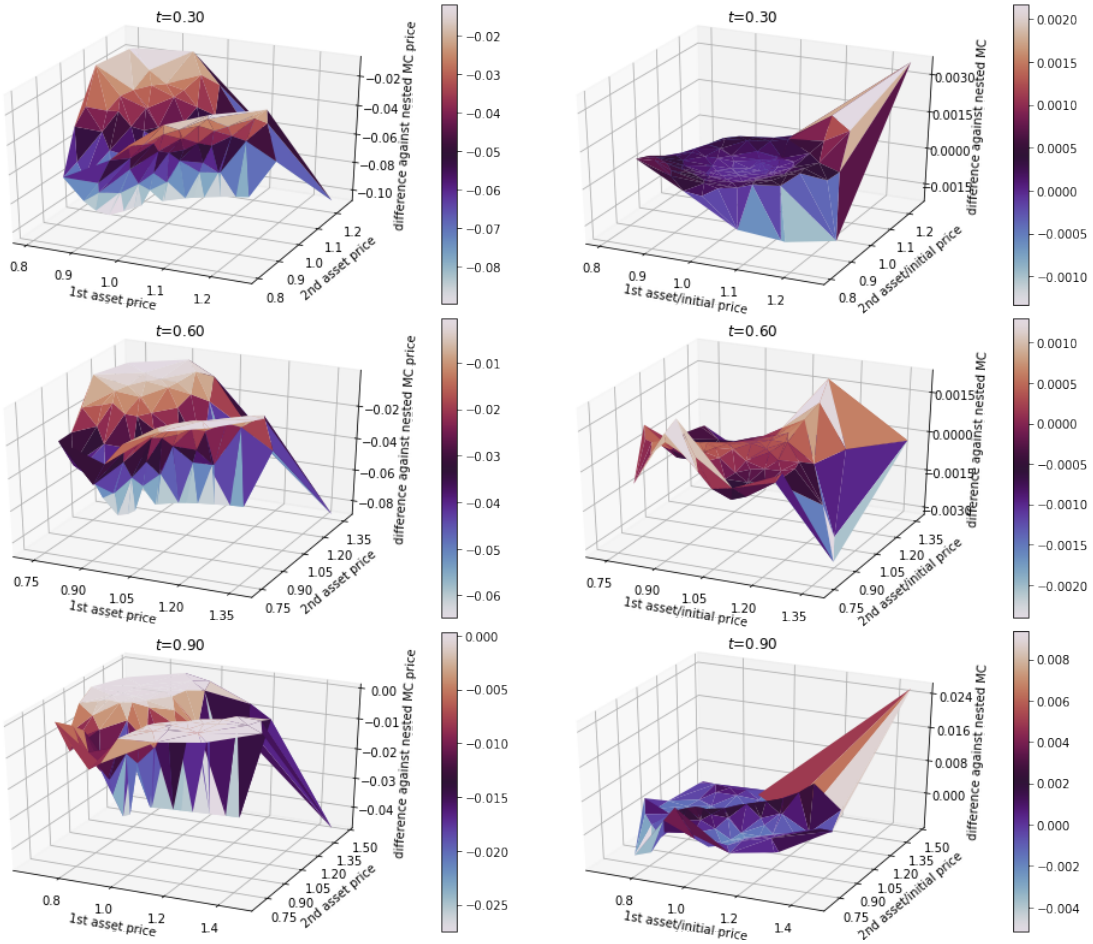

Figure 3.11: Price estimates for best-of-two option minus nested MC results: with 1D kernel and GPR (left); with direct GPR on nested MC (right)

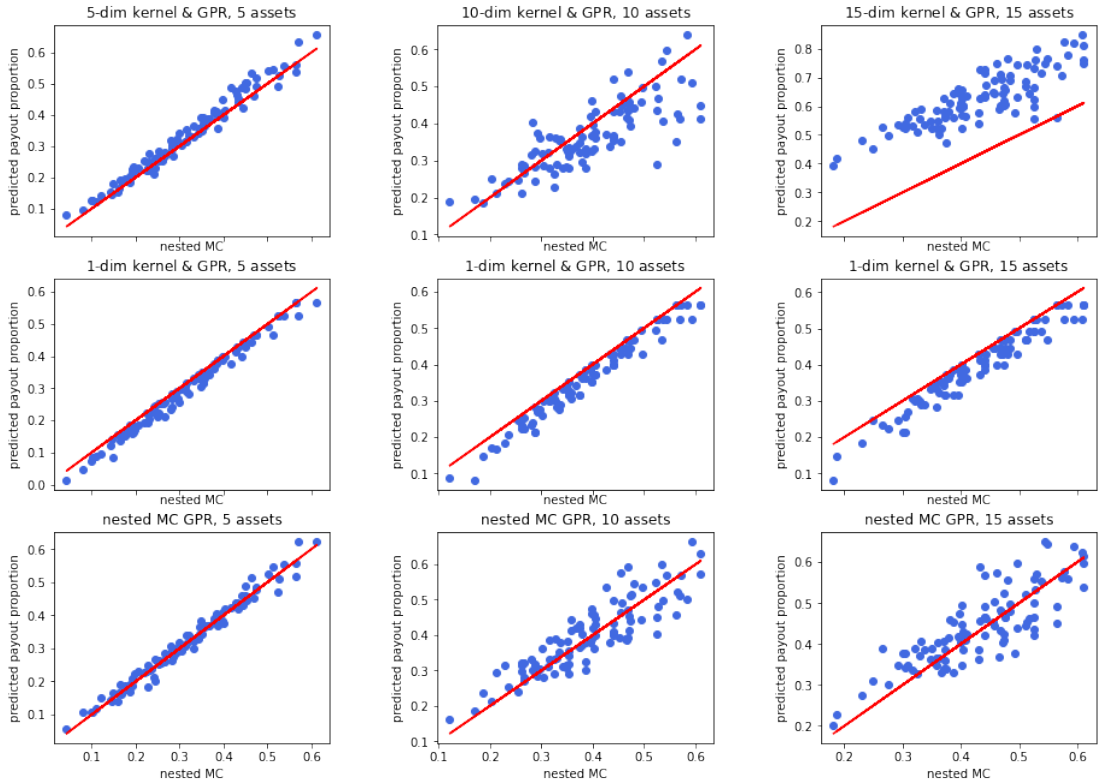

Figure 3.12: Predicted payout proportion against nested MC results for best-of option under different models at $t = 0.9$ with 5 assets (left); 10 assets (middle); 15 assets (right)

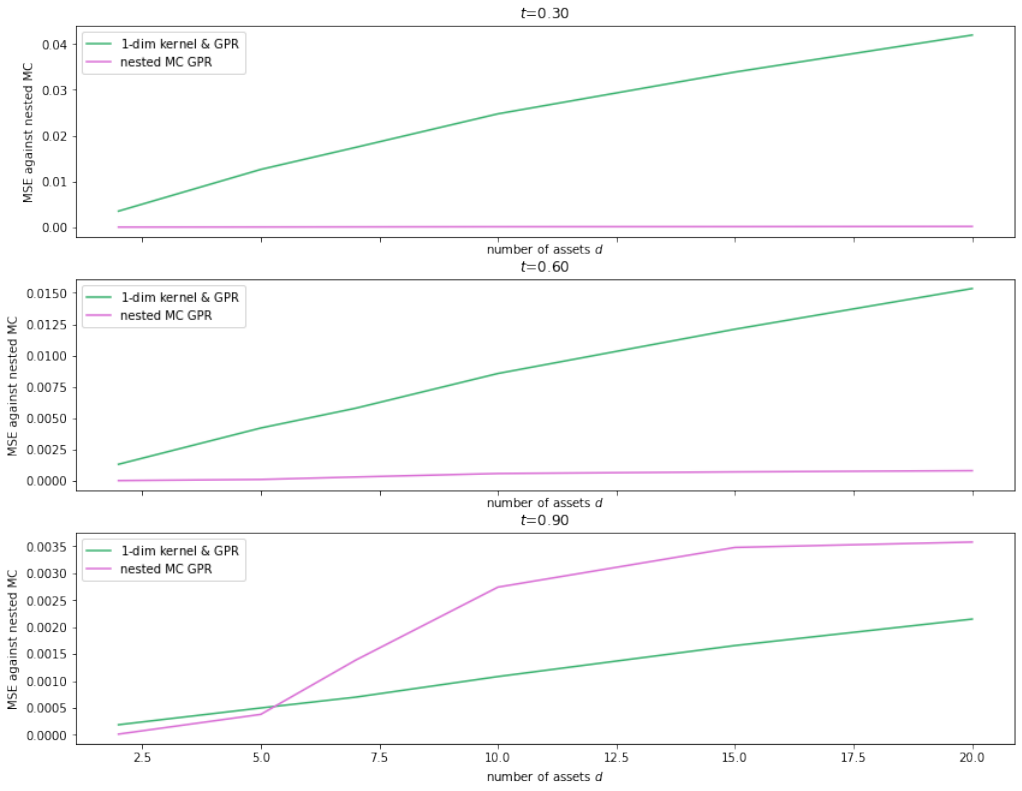

Figure 3.13: Mean squared errors of estimated best-of option prices against nested MC under different models with 2, 5, 7, 10, 15, 20 assets (average of 100 repetitions)

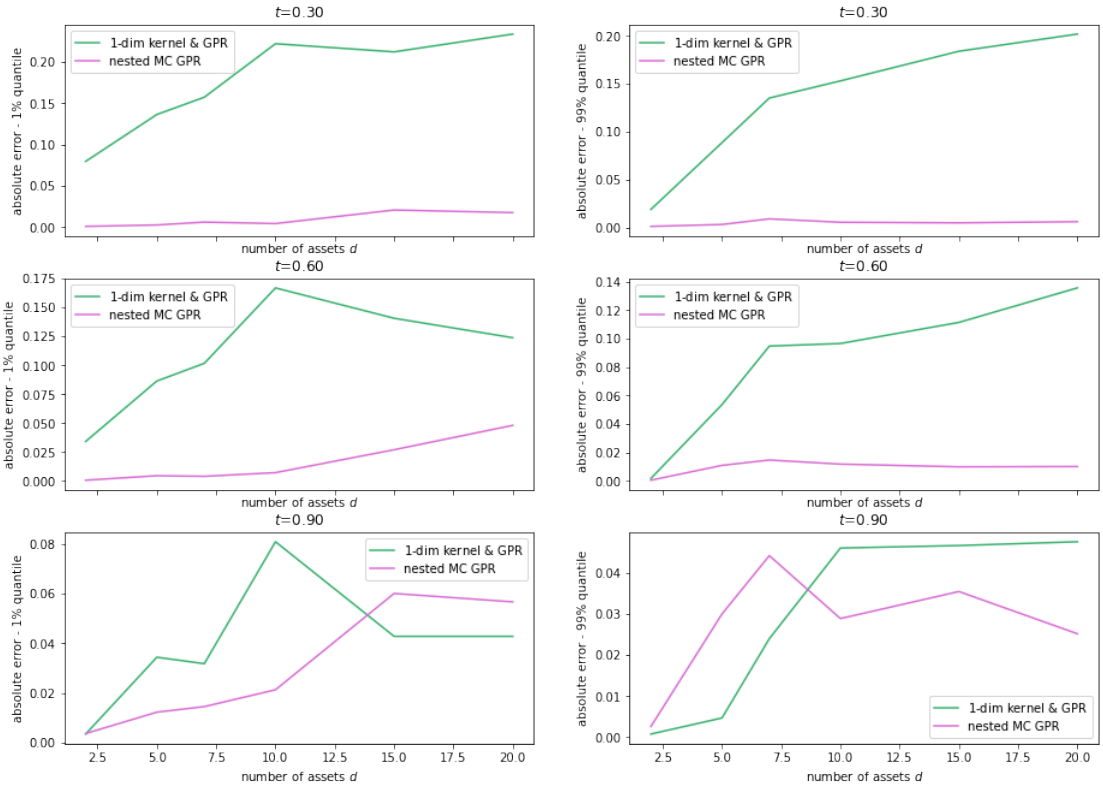

Figure 3.14: Absolute errors in 1% (left) and 99% (right) quantiles of estimated best-of option prices against nested MC under different models with 2, 5, 7, 10, 15, 20 assets (average of 100 repetitions)

Conclusion & Further Work

In this paper, we propose a new algorithm, Dynamically Controlled Kernel Estimation by drawing inspiration from Black-Scholes discrete-time hedging, which computes the conditional expectation of the option values in the future given the information at the current time step more precisely and efficiently, which directly improves the current state of the art for XVA computation.

The key of the algorithm is that we compute the conditional steps in the optimal solution of the variance-minimisation hedging strategy by kernel regression, which gives more precise predictions than Grau's and Potters' regression approach that heavily relies on the choice of basis functions, and also outperforms the current industry standard, Least Square Monte Carlo. Similarly, the idea of kernel regression can be applied to pathwise derivative estimation and anywhere else in our algorithm to work out the conditional expectations. Although the kernelised variance-minimisation solution is shown to be less robust than the kernelised pathwise derivative in all time steps, it provides an alternative to certain products, such as digital options and barrier options, in which the Dominated Convergence Theorem does not apply because of their payoff structures, and we can no longer interchange the order of differentiation and expectation while taking the derivative of such option prices. As the variance-minimisation solution has exactly the same form as the optimal control variate parameter, we include the prediction of deltas as control variate parameters. Through comparisons, the balloon estimator takes advantage of the varying bandwidth across the domain of the PDF based on the evaluation point and associates with smaller errors than the kernel with fixed bandwidth, especially around the edges. To obtain equally reliable estimates, far less work is needed for control variates under the Locally Linear kernel estimator than the Nadaraya-Watson kernel estimator. Also, the Locally Linear estimator outperforms Nadaraya-Watson due to fewer prediction biases at the 'tails', and is even more forgiving in bandwidth choice. With the further inclusion of Gaussian Process Regression, which helps capture the model uncertainty and makes more stable predictions, we finalise the DCKE algorithm. The discounting payoff approach is preferred to the backward recursion approach, since the latter suffers from severe error accumulation problems around the 'tails' as time goes backward step by step.

Under the Black-Scholes model for European options, the DCKE algorithm successfully reduces up to 84% MSE in predicting prices and up to 92% MSE in the estimation of deltas compared to the traditional Least Squares Monte Carlo approach. DCKE also improves the 'tails' estimates remarkably, especially for deltas, with up to hundreds of times smaller absolute errors in the 1% quantile. Evident improvements are also observed under the Heston model, which proves the algorithm is independent of the choice of model for underlying spots.

Another success of the paper is that we significantly reduce the number of simulation paths from 10,000 to less than 2,000 after employing quasi-Monte Carlo (QMC) method by replacing geometric Brownian motion with low-discrepancy Sobol sequence, which further saves the total computational time by over 50% under DCKE.

For more complex products, including path-dependent options and options with multiple underlying assets, the validity of DCKE is also proved. Making continuity adjustment in barrier options enables us to decrease the total estimation MSE by 66%. Also, we verify the faster rate of convergence to exact or almost exact result under the Brownian bridge discretisation scheme rather than the standard discretisation scheme for quasi-Monte Carlo sampling, by reducing up to 69% of the total MSE. Moreover, we extend DCKE to multiple dimensional to price basket and rainbow options. Under comparison, the one-dimensional DCKE approach with the compressed multiple asset sequence as input outperforms the high-dimensional DCKE when the time is close enough to maturity. An important finding is that for rainbow options, even if we apply direct GPR to nested MC results when the asset dimension is over 5, we obtain terrible predictions due

to the curse of dimensionality, and it is therefore not surprising to see the bad-performed multi-dimensional DCKE with a large number of assets. However, when the time is far from maturity, the one-dimensional DCKE no longer works well possibly due to the 'gamma effect', and hence, efforts are still needed to be put in finding out better solutions.

Following these observations, the high-dimensional models in our experiment suffering the curse of dimensionality. However, this is not just a challenge for GPR, but a problem that occurs in multiple domains including sampling, optimisation and machine learning, and still requires active research. Being inspired by our 1-dimensional model, which solves high-dimensional problems, attempts could still be made to reduce the dimensionality of the multiple asset sequence. For best-of options as an example, a potential future work could be selecting the first two best-performed assets to apply a 2-dimensional DCKE rather than just the single best-performed one as we do, and work out the gamma between them to get rid of the 'gamma effect', which will be helpful for the predictions in earlier times steps. Additionally, the backward recursion approach could also be applied to the current 1-dimensional model for multiple assets, given its outstanding performance when time is close to maturity. It is worth investigating how would it solve our problem when we take a small time step back each time.

Also, the validity of the DCKE algorithm under the Heston model inspires us to move towards pricing and replicating in the non-risk-neutral world, improving upon Grau's and Halperin's ideas. By combining our algorithm with a non-risk neutral market data generator, a promisingly powerful framework is expected to be obtained.

Appendix A

Technical Proofs

A.1 Proof of Proposition 2.4

Proof. Denote the estimation error for quantity x as ε_x, we have

$$\bar{Y} = \mathrm{E}[Y] + \varepsilon_Y \tag{A.1.1}$$
$$\bar{X} = \mathrm{E}[X] + \varepsilon_X \tag{A.1.2}$$
$$\widehat{\sigma_X^2} = \overline{(X - \bar{X})^2} = \sigma_X^2 + \varepsilon_{XX} \tag{A.1.3}$$
$$\widehat{\sigma_{Y,X}} = \overline{(Y - \bar{Y})(X - \bar{X})} = \sigma_{Y,X} + \varepsilon_{YX} \tag{A.1.4}$$
$$\hat{\beta}^* = \frac{\widehat{\sigma_{Y,X}}}{\widehat{\sigma_X^2}} = \frac{\sigma_{Y,X} + \varepsilon_{YX}}{\sigma_X^2 + \varepsilon_{XX}} = \beta^* + \varepsilon_{\beta^*} \tag{A.1.5}$$

With $\hat{\beta}^*$ in Equation (2.2.18), the overall estimation error is found by

$$\begin{aligned}\bar{Y} - \hat{\beta}^*\bar{X} + \hat{\beta}^*\mathrm{E}[X] &= \mathrm{E}[Y] + \varepsilon_Y - \left(\beta^* + \varepsilon_{\beta^*}\right)\left(\mathrm{E}[X] + \varepsilon_X\right) + \left(\beta^* + \varepsilon_{\beta^*}\right)\mathrm{E}[X] \\ &= \mathrm{E}[Y] + \varepsilon_Y - \beta^*\varepsilon_X - \varepsilon_{\beta^*}\varepsilon_X,\end{aligned}$$

while for the exact β^*,

$$\begin{aligned}\bar{Y} - \beta^*\bar{X} + \beta^*\mathrm{E}[X] &= \mathrm{E}[Y] + \varepsilon_Y - \beta^*\left(\mathrm{E}[X] + \varepsilon_X\right) + \beta^*\mathrm{E}[X] \\ &= \mathrm{E}[Y] + \varepsilon_Y - \beta^*\varepsilon_X,\end{aligned}$$

and therefore, the statistical error differ by $\varepsilon_{\beta^*}\varepsilon_X$.

Follow from Equation (A.1.2), we have that

$$\begin{aligned}\overline{(X - \bar{X})^2} &= \overline{X^2} - (\bar{X})^2 \\ &= \mathrm{E}\left[\varepsilon_X^2\right] + \varepsilon_{X^2} - \left(\mathrm{E}[X] + \varepsilon_X\right)^2 \\ &= \mathrm{E}\left[\varepsilon_X^2\right] - \mathrm{E}[X]^2 + \varepsilon_{X^2} - 2\mathrm{E}[X]\varepsilon_X + (\varepsilon_X)^2 \\ &= \sigma_X^2 + \varepsilon_{X^2} - 2\mathrm{E}[X]\varepsilon_X + (\varepsilon_X)^2,\end{aligned}$$

and by comparing with Equation (A.1.3), $\varepsilon_{XX} = \varepsilon_{X^2} - 2\mathrm{E}[X]\varepsilon_X + (\varepsilon_X)^2$.

From Central Limit Theorem, the first two terms on the right are $\mathcal{O}(1/\sqrt{n})$ and the last term is $\mathcal{O}(1/n)$. This is how we put statistical approximations into nonlinear formulas, and here the non-linearity is just a square.

And for β^*, from Equation (A.1.2), assuming $\varepsilon \ll \sigma_{XX}$ and using Taylor series,

$$\begin{aligned}\hat{\beta}^* &= (\sigma_{YX} + \varepsilon_{YX}) \left(\frac{1}{\sigma_X^2} - \frac{\varepsilon_{XX}}{\sigma_X^4} \right) + \mathcal{O}\left(\varepsilon_{XX}^2\right) \\ &= \frac{\sigma_{YW}}{\sigma_X^2} + \frac{1}{\sigma_X^2}\varepsilon_{YX} - \frac{\sigma_{YX}}{\sigma_X^4}\varepsilon_{XX} + \mathcal{O}\left(\varepsilon_{XX}^2\right) \\ &= \beta^* + \frac{1}{\sigma_X^2}\varepsilon_{YX} - \frac{\sigma_{YX}}{\sigma_X^4}\varepsilon_{XX} + \mathcal{O}\left(\varepsilon_{XX}^2\right).\end{aligned}$$

Therefore, ε_{β^*} is only $\mathcal{O}(1/\sqrt{n})$. □

A.2 Derivation of Equation (2.2.3)

Derivation 1.

$$\mathrm{E}[\boldsymbol{Y} \mid \boldsymbol{X} = \boldsymbol{x}] = \int \boldsymbol{y} f(\boldsymbol{y} \mid \boldsymbol{x}) d\boldsymbol{y} = \int \boldsymbol{y} \frac{f(\boldsymbol{x}, \boldsymbol{y})}{f(\boldsymbol{x})} d\boldsymbol{y}$$

Using kernel density estimation, which is introduced as Equation (2.2.12)*, for the joint distribution* $f(\boldsymbol{x}, \boldsymbol{y})$ *and* $f(\boldsymbol{x})$ *with a kernel* K*, we have*

$$\widehat{f}(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{N} \sum_{i=1}^{N} \left\{ K_h\left(\boldsymbol{x} - \boldsymbol{x}_i\right) K_h\left(\boldsymbol{y} - \boldsymbol{y}_i\right) \right\}$$

$$\widehat{f}(\boldsymbol{x}) = \frac{1}{N} \sum_{i=1}^{N} \left\{ K_h\left(\boldsymbol{x} - \boldsymbol{x}_i\right) \right\}.$$

Then, we get

$$\begin{aligned}\hat{\mathrm{E}}[\boldsymbol{Y} \mid \boldsymbol{X} = \boldsymbol{x}] &= \int \frac{\boldsymbol{y} \sum_{i=1}^{N} K_h\left(\boldsymbol{x} - \boldsymbol{x}_i\right) K_h\left(\boldsymbol{y} - \boldsymbol{y}_i\right)}{\sum_{j=1}^{N} K_h\left(\boldsymbol{x} - \boldsymbol{x}_j\right)} d\boldsymbol{y} \\ &= \frac{\sum_{i=1}^{N} K_h\left(\boldsymbol{x} - \boldsymbol{x}_i\right) \int \boldsymbol{y} K_h\left(\boldsymbol{y} - \boldsymbol{y}_i\right) d\boldsymbol{y}}{\sum_{j=1}^{N} K_h\left(\boldsymbol{x} - \boldsymbol{x}_j\right)} \\ &= \frac{\sum_{i=1}^{N} \left\{ K_h\left(\boldsymbol{x} - \boldsymbol{x}_i\right) \right\} \boldsymbol{y}_i}{\sum_{j=1}^{N} \left\{ K_h\left(\boldsymbol{x} - \boldsymbol{x}_i\right) \right\}}.\end{aligned}$$

Appendix B

Detailed descriptions

B.1 Lewis' Approximation Formula (Section 1.2.2)

According to Lewis (2009), the price of the European call can be computed as

$$V(S,v,\tau) = S - Ke^{-r\tau}\frac{1}{\pi}\int_{\frac{i}{2}}^{\infty+\frac{i}{2}} e^{-iKX}\frac{\hat{H}(\kappa,v,\tau)}{\kappa - i\kappa}d\kappa,$$

where i is the imaginary part, and

$$\tau = T - t$$

$$X = \ln\left(\frac{S}{K}\right) + r\tau$$

$$\hat{H}(\kappa,v,\tau) = \exp\left(\frac{2\kappa\theta}{\sigma^2}\left[qg - \ln\left(\frac{1-he^{-\xi q}}{1-h}\right) + \nu g\left(\frac{1-e^{-\xi q}}{1-he^{-\xi q}}\right)\right]\right)$$

$$g = \frac{b-\xi}{2}$$

$$h = \frac{b-\xi}{b+\xi}$$

$$q = \frac{\sigma^2\tau}{2}$$

$$\xi = \sqrt{b^2 + \frac{4\left(\kappa^2 - i\kappa\right)}{\sigma^2}}$$

$$b = \frac{2}{\sigma^2}(i\kappa\rho\sigma + \kappa).$$

Mrázek and Pospíšil (2017) point out the great advantage of Lewis' formula is that it is well behaved and only one numeric integration is required.

B.2 Gaussian Process Regression (Section 2.2.4)

- Prior distribution

  Gaussian processes (GPs) extend multivariate Gaussian distributions to infinite dimensionality. Formally, a Gaussian process is a random process where any point $\boldsymbol{x} \in \mathbb{R}^d$ is assigned a random variable $f(\boldsymbol{x})$ and where the joint distribution of a finite number of these variables $p(f(\boldsymbol{x}_1),\ldots,f(\boldsymbol{x}_N))$ is itself Gaussian:

  $$p(\boldsymbol{f} \mid \boldsymbol{X}) = \mathcal{N}(\boldsymbol{f} \mid \boldsymbol{\mu}, \boldsymbol{K}),$$

  where $\boldsymbol{f} = (f(\boldsymbol{x}_1),\ldots,f(\boldsymbol{x}_N))$, $\boldsymbol{\mu} = (m(\boldsymbol{x}_1),\ldots,m(\boldsymbol{x}_N))$ and the kernel matrix $K_{ij} = k(\boldsymbol{x}_i,\boldsymbol{x}_j)$. The mean function $m(\mathbf{x})$ is assumed to take zero everywhere as GPs are flexible enough to model the mean arbitrarily well.

- Kernel function

  What relates an observation $\boldsymbol{x}_i$ to another $\boldsymbol{x}_j$ is just the positive definite *kernel function* or *covariance function* $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$. A popular choice is the 'squared exponential' kernel, which is stationary and also known as RBF (Radial-basis function) kernel:

$$K\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) = \sigma_f^2 \exp\left(-\frac{1}{2l^2}\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)^T\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)\right), \tag{B.2.1}$$

  where σ_f^2 represents the vertical variation of the function, and l is the length scale of the kernel, which controls the shape (smoothness). For simplicity, we keep l the same for all input dimensions (isotropic kernel). When $\boldsymbol{x}_j$ is close to $\boldsymbol{x}_i$, $K\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right)$ approaches its maximum, which means $f(\boldsymbol{x}_j)$ is nearly perfectly correlated with $f(\boldsymbol{x}_i)$, and vice versa.

- Posterior distribution

  A GP prior $p(\boldsymbol{f} \mid \boldsymbol{X})$ can be converted into a GP posterior $p(\boldsymbol{f} \mid \boldsymbol{X}, \boldsymbol{y})$ after having observed some data $\boldsymbol{y}$. The posterior can then be used to make predictions f_* given new input $\boldsymbol{X}_*$:

$$\begin{aligned} p\left(\boldsymbol{f}_* \mid \boldsymbol{X}_*, \boldsymbol{X}, \boldsymbol{y}\right) &= \int p\left(\boldsymbol{f}_* \mid \boldsymbol{X}_*, \boldsymbol{f}\right) p(\boldsymbol{f} \mid \boldsymbol{X}, \boldsymbol{y}) d\boldsymbol{f} \\ &= \mathcal{N}\left(\boldsymbol{f}_* \mid \boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*\right), \end{aligned}$$

  which is the posterior predictive distribution which is also a Gaussian with mean $\boldsymbol{\mu}_*$ and covariance matrix $\boldsymbol{\Sigma}_*$. Since the key assumption in GP modelling is that our data can be represented as a sample from a multivariate Gaussian distribution, the joint distribution of observed data $\boldsymbol{y}$ and predictions $\boldsymbol{f}_*$ is

$$\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{pmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{pmatrix} \boldsymbol{K}_y & \boldsymbol{K}_* \\ \boldsymbol{K}_*^T & \boldsymbol{K}_{**} \end{pmatrix}\right),$$

  where $\boldsymbol{K}_y = \kappa(\boldsymbol{X}, \boldsymbol{X}) + \sigma_y^2 \boldsymbol{I} = \boldsymbol{K} + \sigma_y^2 \boldsymbol{I}$, $\boldsymbol{K}_* = \kappa\left(\boldsymbol{X}, \boldsymbol{X}_*\right)$ and $\boldsymbol{K}_{**} = \kappa\left(\boldsymbol{X}_*, \boldsymbol{X}_*\right)$, and σ_y^2 is the noise term in the diagonal of $\boldsymbol{K}_y$ to take into account the noise in the observations. The mean is set to zero for notational simplicity, and the sufficient statistics of the posterior predictive distribution $\boldsymbol{\mu}_*$ and $\boldsymbol{\Sigma}_*$ can be computed with

$$\boldsymbol{\mu}_* = \boldsymbol{K}_*^T \boldsymbol{K}_y^{-1} \boldsymbol{y},$$

$$\boldsymbol{\Sigma}_* = \boldsymbol{K}_{**} - \boldsymbol{K}_*^T \boldsymbol{K}_y^{-1} \boldsymbol{K}_*.$$

Appendix C

Figures and tables

C.1 Chapter 1 Figures

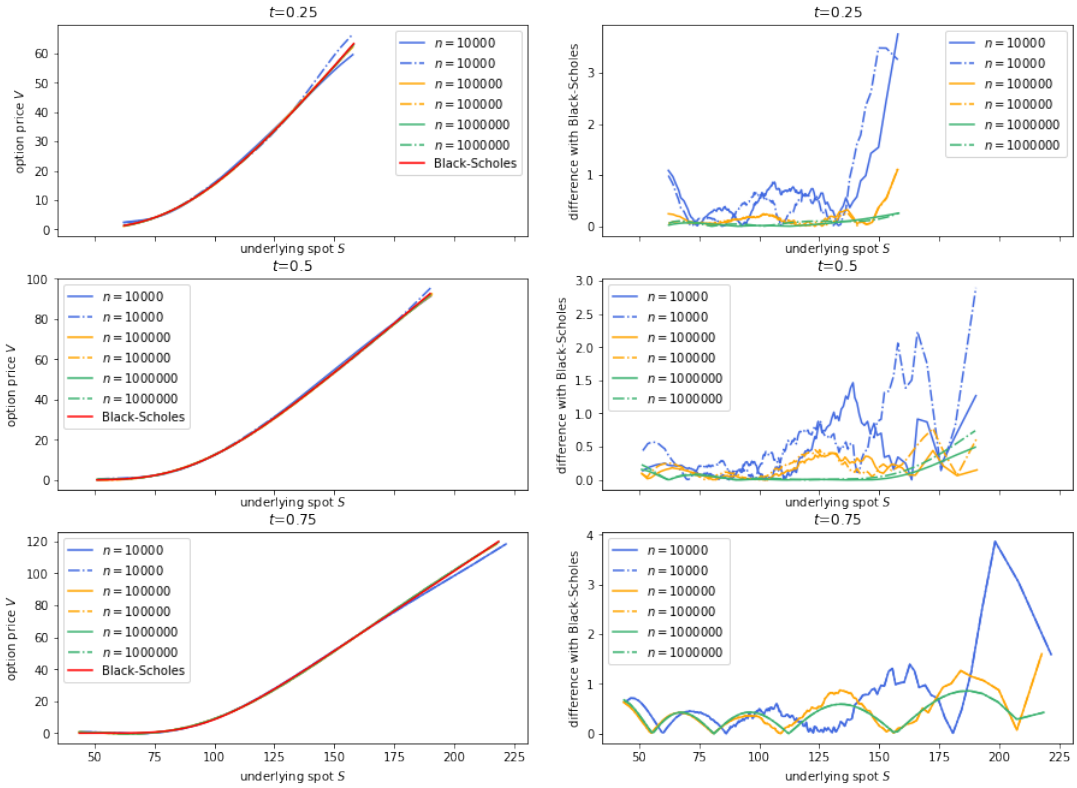

Figure C.1: LSM for option prices under different number of paths versus Black-scholes prices (left); difference with Black-Scholes (right): discounting payoff approach ('-'); backward recursion approach ('-·-')

C.2 Chapter 2 Figures

C.3 Discounting payoff approach versus backward recursion approach error statistics comparison (Section 2.2.5)

C.4 GBM versus Sobol error statistics comparison (Section 2.3.2)

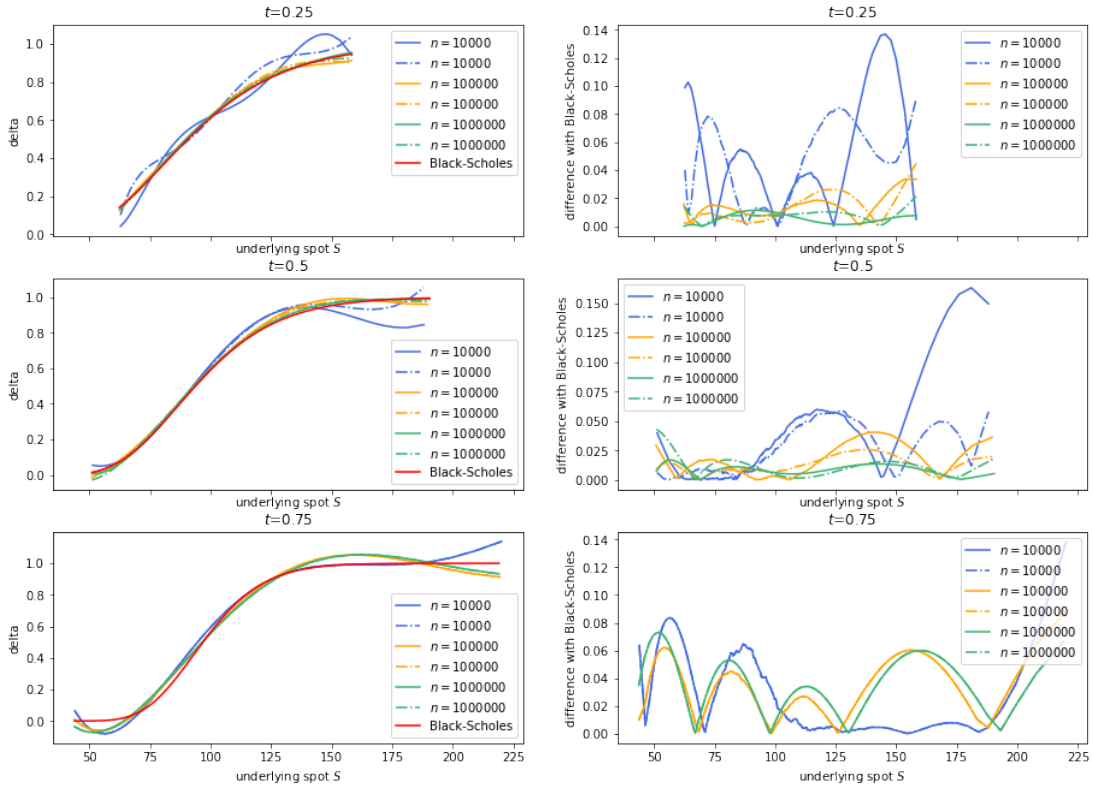

Figure C.2: LSM for deltas under different number of paths versus Black-scholes prices (left); difference with Black-Scholes (right): discounting payoff approach ('-'); backward recursion approach ('-··')

Price	Metric	$t = 0.25$	$t = 0.50$	$t = 0.75$
Discounting payoff	MSE	0.3588	0.3078	0.0541
	mean	0.1194	0.1779	0.1808
	1%	0.4271	0.2275	0.0606
	50%	0.1458	0.2321	0.3184
	99%	0.5716	0.2053	0.0562
Backward recursion	MSE	0.3811	0.2054	0.0540
	mean	0.3564	0.3215	0.1815
	1%	0.5597	0.3098	0.0616
	50%	0.5501	0.4555	0.3283
	99%	1.5281	0.6300	0.0546

Table C.1: Mean squared errors and absolute errors in EE and PFE of the estimated prices with discounting payoff and backward recursion approach compared to Black-Scholes prices (average of 100 repetitions)

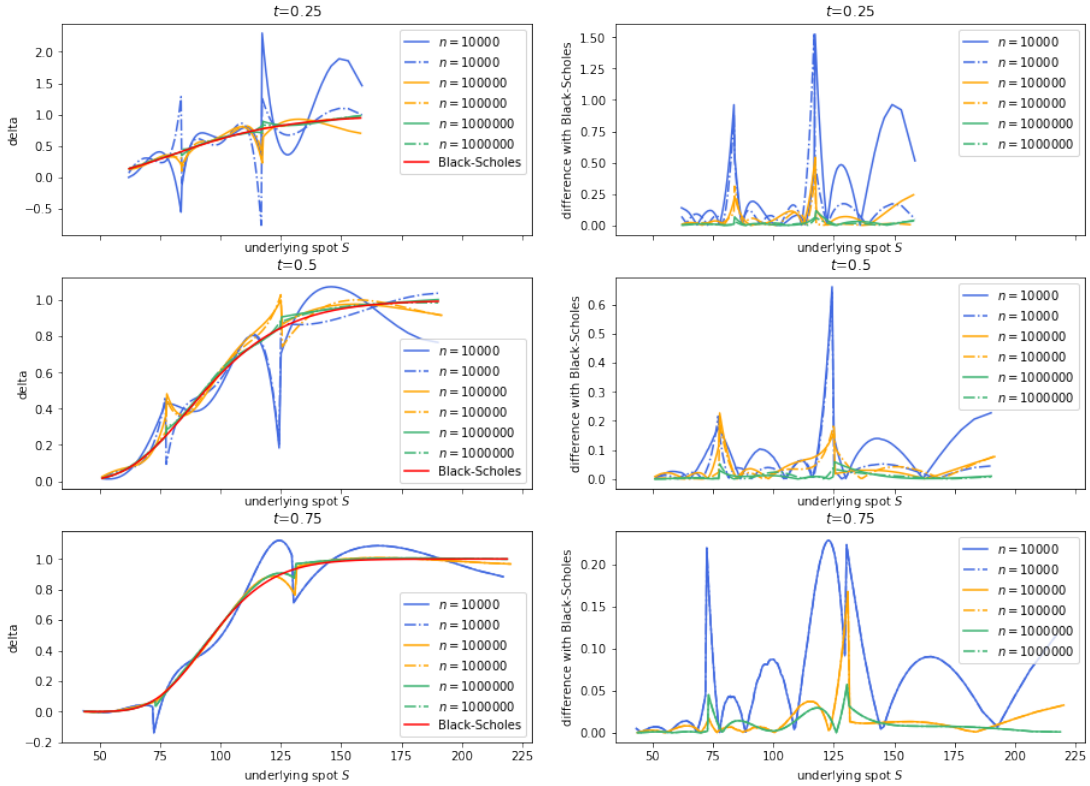

Figure C.3: LSM for deltas under different number of paths with piecewise polynomial regressions versus Black-Scholes deltas (left); difference with Black-Scholes (right): discounting payoff approach ('-'); backward recursion approach ('-·')

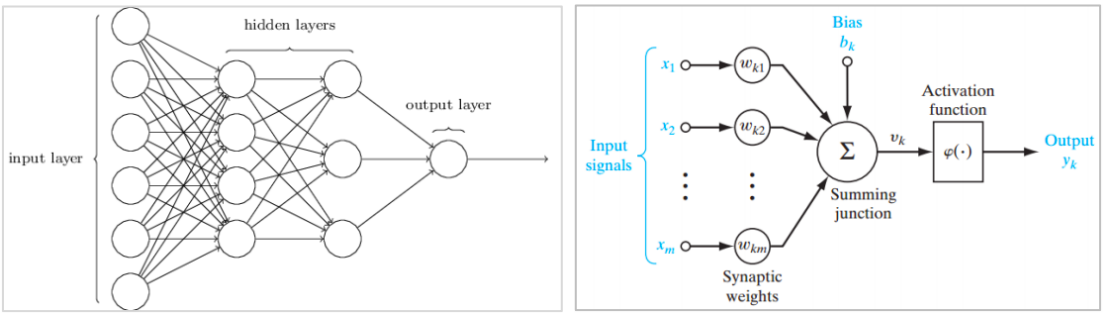

Figure C.4: Demonstration of basic concepts of a feedforward neural network (Sodhi (2018))

			Option price			Delta		
Spots	**Metric**	**Paths**	$t = 0.25$	$t = 0.50$	$t = 0.75$	$t = 0.25$	$t = 0.50$	$t = 0.75$
GBM	MSE	10,000	0.3658	0.3057	0.0799	0.0003	0.0003	0.0002
	mean		0.1550	0.1998	0.2191	0.0073	0.0069	0.0070
	1%		0.2338	0.3347	0.1059	0.0060	0.0020	0.0002
	50%		0.2185	0.2377	0.4063	0.0064	0.0114	0.0153
	99%		0.2905	0.2508	0.0921	0.0032	0.0015	0.0061
Sobol	MSE	10,000	0.0396	0.0715	0.0514	0.0000	0.0000	0.0002
	mean		0.0161	0.0120	0.0030	0.0075	0.0073	0.0069
	1%		0.0142	0.1057	0.0053	0.0020	0.0013	0.0002
	50%		0.0209	0.0601	0.0849	0.0088	0.0112	0.0132
	99%		0.0807	0.0176	0.0571	0.0033	0.0034	0.0040
	MSE	2,000	0.1510	0.3119	0.1670	0.0002	0.0003	0.0004
	mean		0.2089	0.2871	0.3204	0.0068	0.0064	0.0059
	1%		0.7241	0.6302	0.1701	0.0005	0.0002	0.0014
	50%		0.2890	0.3887	0.5869	0.0082	0.0115	0.0117
	99%		0.3037	0.0568	0.0593	0.0018	0.0022	0.0128
	MSE	1,000	0.2689	0.8007	0.2841	0.0004	0.0007	0.0009
	mean		0.2479	0.3516	0.3992	0.0090	0.0087	0.0077
	1%		0.9362	0.6999	0.3190	0.0017	0.0007	0.0027
	50%		0.3602	0.4646	0.7278	0.0114	0.0172	0.0148
	99%		0.0311	0.0291	0.1795	0.0023	0.0009	0.0190

Table C.2: Mean squared errors and absolute errors in mean and 1%, 50% and 99% quantile of option price and delta compared to Black-Scholes results for GBM and Sobol under DCKE model (average of 100 repetitions)

Bibliography

Bianchetti, M., Kucherenko, S. and Scoleri, S. (2015). Pricing and risk management with high-dimensional quasi-monte carlo and global sensitivity analysis, *Wilmott* **2015**(78): 46–70.

Boyle, P. P. (1977). Options: A monte carlo approach, *Journal of financial economics* **4**(3): 323–338.

Broadie, M., Glasserman, P. and Kou, S. (1997, pg 327). A continuity correction for discrete barrier options, *Mathematical Finance* **7**(4): 325–349.

Cacoullos, T. (1966). Estimation of a multivariate density, *Annals of the Institute of Statistical Mathematics* **18**(1): 179–189.

Carriere, J. F. et al. (1996). Valuation of the early-exercise price for options using simulations and nonparametric regression, *Insurance: mathematics and Economics* **19**(1): 19–30.

Cesari, G., Aquilina, J., Charpillon, N., Filipovic, Z., Lee, G. and Manda, I. (2009). *Modelling, pricing, and hedging counterparty credit exposure: A technical guide*, Springer Science & Business Media.

Chebyshev, P. L. (1853). *Théorie des mécanismes connus sous le nom de parallélogrammes*, Imprimerie de l'Académie impériale des sciences.

Epanechnikov, V. A. (1969). Non-parametric estimation of a multivariate probability density, *Theory of Probability & Its Applications* **14**(1): 153–158.

Fan, J. (1992). Design-adaptive nonparametric regression, *Journal of the American statistical Association* **87**(420): 998–1004.

Fei, Q.-M. (n.d.). An introduction to barrier options—closed form solution and a monte carlo approach.

Gal, Y. and Ghahramani, Z. (2016). Dropout as a bayesian approximation: Representing model uncertainty in deep learning, *international conference on machine learning*, pp. 1050–1059.

Grau, A. J. (2008). *Applications of least-squares regressions to pricing and hedging of financial derivatives*, PhD thesis, Technische Universität München.

Grau, A. J. (2008, pg 126). *Applications of least-squares regressions to pricing and hedging of financial derivatives*, PhD thesis, Technische Universität München.

Halperin, I. (2020). Qlbs: Q-learner in the black-scholes (-merton) worlds, *The Journal of Derivatives* .

Härdle, W. (1990, pg 29). *Applied nonparametric regression*, number 19, Cambridge university press.

Heston, S. L. (1993). A closed-form solution for options with stochastic volatility with applications to bond and currency options, *The review of financial studies* **6**(2): 327–343.

Judd, K. L. (1998). *Numerical methods in economics*, MIT press.

Lee, A. (2015). The triple convergence of credit valuation adjustment (cva), *Global Trading Podcast* .

Lewis, A. L. (2009). Option valuation under stochastic volatility ii, *Finance Press* .

Longstaff, F. A. and Schwartz, E. S. (2001). Valuing american options by simulation: a simple least-squares approach, *The review of financial studies* **14**(1): 113–147.

Mrázek, M. and Pospíšil, J. (2017). Calibration and simulation of heston model, *Open Mathematics* **15**(1): 679–704.

Nadaraya, E. A. (1964). On estimating regression, *Theory of Probability & Its Applications* **9**(1): 141–142.

Piterbarg, V. (2003). A stochastic volatility forward libor model with a term structure of volatility smiles, *Available at SSRN 472061* .

Pochart, B. and Bouchaud, J.-P. (2004). Option pricing and hedging with minimum local expected shortfall, *Quantitative Finance* **4**(5): 607–618.

Potters, M., Bouchaud, J.-P. and Sestovic, D. (2001). Hedged monte-carlo: low variance derivative pricing with objective probabilities, *Physica A: Statistical Mechanics and its Applications* **289**(3-4): 517–525.

Powell, W. B. (2011, pg 307). *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, Vol. 842, John Wiley & Sons.

Ruppert, D. and Wand, M. P. (1994). Multivariate locally weighted least squares regression, *The annals of statistics* pp. 1346–1370.

Schweizer, M. (1995). variance-optimal hedging in discrete time, *Mathematics of Operations Research* **20**(1): 1–32.

Silverman, B. W. (1986). *Density estimation for statistics and data analysis*, Vol. 26, CRC press.

Sobol', I. M. (1967). On the distribution of points in a cube and the approximate evaluation of integrals, *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki* **7**(4): 784–802.

Sodhi, A. (2018). American put option pricing using least squares monte carlo method under bakshi, cao and chen model framework (1997) and comparison to alternative regression techniques in monte carlo, *arXiv preprint arXiv:1808.02791* .

Terrell, G. R. and Scott, D. W. (1992). Variable kernel density estimation, *The Annals of Statistics* pp. 1236–1265.

Tsitsiklis, J. N. and Van Roy, B. (2001). Regression methods for pricing complex american-style options, *IEEE Transactions on Neural Networks* **12**(4): 694–703.

Watson, G. S. (1964). Smooth regression analysis, *Sankhyā: The Indian Journal of Statistics, Series A* pp. 359–372.