STOCHASTIC MESH METHOD FOR HIGH DIMENSIONAL BACKWARD STOCHASTIC DIFFERENTIAL EQUATION

PENG YUAN

(B.Eng., Ecole Centrale Paris, France)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF MATHEMATICS

NATIONAL UNIVERSITY OF SINGAPORE

2016

DECLARATION

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Peng YUAN

May 6, 2016

Acknowledgements

I would like to express my sincerest thanks to my supervisor Dr. Zhou Chao who suggested me to apply the stochastic mesh method in solving BSDEs and for providing me with constant guidance and support throughout. I am also thankful to my colleague Xia Haoyang for his kind help and participation. Furthermore I would express my thanks to my wife Li Wu for her constant support and kind consideration.

Contents

Summary

This thesis focuses on solving high dimensional BSDEs with stochastic mesh method. Stochastic mesh method was first designed to price American options. Inspired by its backward induction technique, we apply it in BSDE background as a new numerical scheme. The whole thesis is organized as follows.

Chapter 1 introduces the history of numerical schemes in solving BSDEs and formulate our BSDE problems in mathematical form. Chapter 2 mainly deals with the way to discretize the time space and the details of stochastic mesh method. The way to construct the mesh and to define associated weight function is the key point in this thesis. Chapter 3 presents some empirical results first with one-dimensional cases and then with high-dimensional cases. In both cases stochastic mesh method proves to be a very promising method in solving BSDE. The last chapter makes a conclusion.

This thesis serves as an important step in applying stochastic mesh method in solving high-dimensional BSDEs. The major problem comes from the computation effort which needs a lot of improvement.

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Literature review

The Backward Stochastic Differential Equation or BSDE for short is first studied by Pardoux and Peng [7] in 1990. Since then extensive studies in both theoretical and practical areas have been rapidly developed. In mathematical finance, BSDE has become an important tool. Numerous application could be found in [6] and [5]. Among those applications we will focus on options pricing problem or option replication problem.

Generally speaking there is no explicit solution to a given BSDE problem except for certain simple cases. Thus numerical resolution is often employed and has made recent progress. However even though several numerical methods have been proposed, they often suffer various problems including high dimensionality, path dependent payoff and etc. Generally there are two kinds of numerical schemes to solve BSDE. One is a four step scheme to solve general FBSDE proposed by Ma, Protter and Yong [12]. This algorithm focus on solving the parabolic partial differential equations connected by the BSDE. The other numerical scheme consists of solving the BSDE directly in backward sense by Monte-Carlo simulation due to $\mathbb{L}_2$-regularity proven in [10]. This regularity allows us to create a sequence of conditional expectation which can be computed using Monte-Carlo simulation.

This thesis applys a new method, namely stochastic mesh method to help solve BSDE. Basically our numerical scheme belongs to the second category. Unlike the approach proposed by [9] where regression-based method is used to compute the sequence of conditional expectations in each time step, we interconnect nodes between paths and uses particular weights function to compute the conditional expectations. Stochastic mesh method was first proposed by Broadie and Glasserman[2] to price high-dimensional American option. Essentially it is an efficient way to compute conditional expectation in back-style dynamic programming technique, which is very similar to solving BSDE numerically by Monte Carlo simulation. That is where we got the inspiration to propose this new perspective. Furthermore this method allows high dimensionality and path dependent terminal condition which is often a difficult part in other numerical scheme. Later in our simulation we will cover more about high-dimensional BSDE problems.

1.2 Problem formulation

In this thesis, we focus on a numerical method for solving backward stochastic differential equations or BSDE. Let's begin with the problem formulation.

1.2.1 BSDE

Decoupled forward backward stochastic differential equation (FBSDE) is the most common BSDE problem and the solution to other BSDE problems often relies on FBSDE. In its financial application, FBSDE is most commonly used to solve European style option pricing problem. The European style option gives the holder the right but not the obligation to exercise the option at *maturity*. To define FBSDE, we use the following set up throughout this thesis. Let T be a fixed and deterministic time and $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{0\le t\le T}, \mathbb{P})$ be a probabilistic space supporting a d-dimensional

Brownian motion $(W_t)_{0\leq t\leq T}$ where $(\mathcal{F}_t)_t$ is the augmented natural filtration of W. Then FBSDE is defined by

$$X_t = X_0 + \int_0^t \mu(s, X_s)ds + \int_0^t \sigma(s, X_s)dW_s \tag{1.1}$$

$$Y_t = \Phi(\mathbf{X}) + \int_t^T f(s, X_s, Y_s, Z_s)ds - \int_t^T Z_s dW_s. \tag{1.2}$$

In this representation, $\mathbf{X} = (X_t; 0 \leq t \leq T)$ is the d-dimensional forward process and Y_t is the one-dimensional backward one. μ, σ, f, Φ are all deterministic functions. Throughout the paper we assume the following assumptions are fulfilled.

Assumption 1. *The function $(t,x) \mapsto \mu(t,x)$ and $(t,x) \mapsto \sigma(t,x)$ are uniformly Lipschitz continuous w.r.t $(t,x) \in [0,T] \times \mathbb{R}^d$.*

Assumption 2. *The driver f satisfies the following continuity estimate:*

$$|f(t_2, x_2, y_2, z_2) - f(t_1, x_1, y_1, z_1)| \leq C_f(\sqrt{|t_2 - t_1|} + |x_2 - x_1| + |y_2 - y_1| + |z_2 - y_1|)$$

Assumption 3. *The terminal condition Φ satisfies the functional Lipschitz condition, i.e for any continuous functional $\mathbf{x}^1$ and $\mathbf{x}^2$, one has*

$$|\Phi(\mathbf{x}^1) - \Phi(\mathbf{x}^2)| \leq C \sup_{t\in[0,T]} |x_t^1 - x_t^2|.$$

These assumptions are sufficient to ensure the existence and uniqueness of a triplet $(\mathbf{X}, \mathbf{Y}, \mathbf{Z})$ to solution 1.1-1.2(see [13] and the reference therein). In addition the assumption 3 allows a large class of terminal conditions. When related to option pricing problems, $\Phi(\mathbf{X})$ is the usual payoff of contingent claim, Y is the value of the replicating portfolio and Z relates to the hedging strategy. In complete market, the driver f is linear w.r.t Y and Z. Furthermore since the driver f allows Lipschitz condition, some market imperfections can also be incorporated such as higher interest rate for borrowing. Related numerical experiments are developed in Chapter 3. In incomplete markets, the Föllmer-Schweizer strategy is obtained by the solution of a BSDE. In some incomplete markets, certain trading constraints are imposed on some assets. This includes the situation when the assets are impossible to trade such as the index, temperature level, rainfall level; or when it is more preferable not to trade these assets due to transaction cost, liquidity or physical locations problems. In this situation, one can either use tradable assets to partially hedge these non-tradable assets and evaluate the price subject to risk preference or use super-replication strategy to get the price as the limit of nonlinear BSDEs. There are many other applications of BSDE such as the connection with semi-linear PDEs, stochastic optimization and control theory. We refer to [6] for more applications. Indeed BSDE has become an inevitable tool in mathematical finance.

1.2.2 RBSDE

RBSDE is short for reflected backward stochastic differential equation. RBSDE is often used to price American style option. American style option gives the holder the right but not the obligation to exercise the option at anytime *prior to maturity*. With the same payoff, American option gives the holder more rights than European option. As a consequence American option at least has the same value as European option provided with the same payoff. The difficulty of pricing American option is to find the optimal stopping policy that maximize the expected wealth. At any time prior to maturity, the value of an American option is the maximum of continuation value and exercise value at that time. When the continuation value is greater than the exercise value, the early exercise feature at that time is useless and the option follows the same dynamics as the European option. When the continuation value is smaller than exercise value, the option value is just the exercise value which means it's time to exercise the option. Therefore RBSDE has a very deep connection with the usual FBSDE. Indeed the numerical resolution of RBSDE requires only a small modification to the numerical scheme of solving FBSDE. Later, we will see the difference.

Given the same condition as BSDE in previous section, RBSDE is defined by

$$Y_t = \Phi(T, \mathbf{X_T}) + \int_t^T f(s, X_s, Y_s, Z_s)ds + K_T - K_t - \int_t^T Z_s dW_s \tag{1.3}$$

$$Y_t \geq \Phi(t, X_t) \tag{1.4}$$

$$\int_0^T (Y_t - \Phi(s, X_s))dK_t = 0, K_0 = 0 \tag{1.5}$$

where the K_t is a continuous increasing process. Φ is an obstacle function. Under the assumptions that

Assumption 4. *Driver f is Lipschitz continuous w.r.t (x, y, z) and $1/2$ Holder w.r.t to t . X_t, Y_t, Z_t, K_t are adapted to the natural filtration generated by the Brownian motion.*

Assumption 5.

$$\Phi \in \mathbb{L}_2(\mathcal{F}_T)$$

Assumption 6.

$$\mathbb{E}[\sup_{0 \leq t \leq T} \Phi(t, X_t)^2] < \infty,$$

there exist a unique triplet solution (Y, Z, K) [11].

In the above notations, $K_T - K_t$ stands for the extra value brought by the early exercise feature for the remaining life. Since K_t is a continuous increasing process, the option value becomes less when K_t increases. The equation (1.5) could be interpreted in this way. Whenever the increment of K_t is strictly positive, the option value equals the exercise value. Holding the option gives us less value than exercise it. Whenever the option value is greater than exercise value, the increment of K_t is zero. It is better to hold the option rather than to exercise it.

Chapter 2

Methodology

2.1 Discretization

2.1.1 BSDE discretization

In the following we will assume an equidistant time discretization ($t_i = iT/N$) to explain the methodology as in practice this is the usual case. However the results derived in this chapter are still valid when time discretization is chosen in other way. We apply the standard Euler scheme to forward process

$$\hat{X}_{t_{i+1}} = \hat{X}_{t_i} + \mu(t_i, \hat{X}_{t_i})\Delta t_i + \sigma(t_i, \hat{X}_{t_i})\Delta W_{t_i} \tag{2.1}$$

where $\hat{X}_{t_i}$ is the estimate of X_{t_i}, $\Delta t_i = t_{i+1} - t_i, \Delta W_i = W_{i+1} - W_i$.

Euler scheme is an efficient scheme where the weak error as shown by Bally and Talay[1] is of order $O(1/N)$ for a bounded European style payoff. The backward process is evaluated in a backward manner.

$$\hat{Y}_{t_N} = \Phi(\hat{X}_{t_N}) \tag{2.2}$$

$$\hat{Z}_{t_i} = \mathbb{E}[\hat{Y}_{i+1}\frac{\Delta W_i}{\Delta t_i}|\mathcal{F}_{t_i}] \tag{2.3}$$

$$\hat{Y}_{t_i} = \mathbb{E}[\hat{Y}_{t_{i+1}} + f(t_i, \hat{X}_{t_i}, \hat{Y}_{t_{i+1}}, \hat{Z}_{t_i})\Delta t_i|\mathcal{F}_{t_i}] \tag{2.4}$$

This scheme is an *explicit* type.$\hat{Y}_{t_i}$ and $\hat{Z}_{t_i}$ are estimates for Y_{t_i} and Z_{t_i} respectively. Unlike the forward process X_t, we can not use a similar Euler scheme to back from $\hat{Y}_{t_{i+1}}$ to $\hat{Y}_{t_i}$ by the difference of process values at consecutive time steps. The backward process $\hat{Y}_i$ estimated in that way does not adapt to the natural filtration $\mathcal{F}_{t_i}$ since it depends on future values. In order to overcome this difficulty, we propose the above discretization scheme. The idea is to discretize the backward process in a similar fashion as the Eular Scheme and take conditional expectation on all non-measurable terms so that all of them become adapted to the filtration. Then we replace $\hat{Y}_{t_i}$ by $\hat{Y}_{t_{i+1}}$ in the driver to make it explicit. However in this scheme 2.2-2.4 we need to estimate two conditional expectations. And in our case by stochastic mesh method, new source of error from estimating Z is introduced.

There exists another similar numerical scheme which is an *implicit* type. It's the same idea without the replacing step.

$$\hat{Y}_{t_N} = \Phi(\hat{X}_{t_N}) \tag{2.5}$$

$$\hat{Z}_{t_i} = \mathbb{E}[\hat{Y}_{i+1}\frac{\Delta W_i}{\Delta t_i}|\mathcal{F}_{t_i}] \tag{2.6}$$

$$\hat{Y}_{t_i} = \mathbb{E}[\hat{Y}_{t_{i+1}}|\mathcal{F}_{t_i}] + f(t_i, \hat{X}_{t_i}, \hat{Y}_{t_i}, \hat{Z}_{t_i})\Delta t_i \tag{2.7}$$

Thanks to $\mathbb{L}_2$-regularity of Y_t, the problem 2.5-2.7 is equivalent to a least square problem over $\mathbf{L}_2(\mathcal{F}_{t_i})$ functions

$$(\hat{Y}_{t_i}, \hat{Z}_{t_i}) = arg \min_{(Y,Z)\in\mathbf{L}_2(\mathcal{F}_{t_i})} \mathbb{E}[\hat{Y}_{t_{i+1}} + f(t_i, \hat{X}_{t_i}, Y, Z)\Delta t_i - Y - Z\Delta W_{t_i}] \tag{2.8}$$

Gobet, Lemor and Warin [9] has proposed a regression-based Monte-Carlo method to solve this least square problem.

The rate of convergence of implicit and explicit coincides for Lipschitz driver. The explicit scheme is the simplest and presumably sufficient for Lipschitz driver. If we define the measure of quadratic error as

$$\epsilon(\hat{Y} - Y, \hat{Z} - Z)) = \max_{0\le i\le N} \mathbb{E}|\hat{Y}_{t_i} - Y_{t_i}|^2 + \sum_{i=0}^{N-1}\int_{t_i}^{t_{i+1}} \mathbb{E}|\hat{Z}_{t_i} - Z_t|^2 dt \tag{2.9}$$

For a Lipschitz driver f w.r.t (x, y, z) and 1/2 Holder w.r.t to t, Gobet [8] has shown for both case that

$$\epsilon(\hat{Y}-Y, \hat{Z}-Z)) \leq C(\mathbb{E}|\Phi(\hat{X}_T)-\Phi(X_T)|^2+\sup_{0\leq i\leq N}\mathbb{E}|\hat{X}_{t_i}-X_{t_i}|^2+|\pi|+\sum_{i=0}^{N-1}\int_{t_i}^{t_{i+1}}\mathbb{E}|\bar{Z}_{t_i}-Z_t|^2dt) \tag{2.10}$$

where $\bar{Z}_{t_i} = \frac{1}{\Delta t_{t_i}}\mathbb{E}[\int_{t_i}^{t_{i+1}} Z_s ds|\mathcal{F}_{t_i}]$ and $\pi = \sup_{0\leq i\leq N-1}|t_{i+1}-t_i|$.

From 2.10 we can see three different error contributions.

- Approximation of the terminal condition $\mathbb{E}|\Phi(\hat{Y_T}) - \Phi(Y_T)|^2$ which depends on the forward scheme not the backward problem
- Approximation of the forward SDE $\sup_{0\leq i\leq N}\mathbb{E}|\hat{X}_{t_i}-X_{t_i}|^2$ which again depends on the forward scheme not the backward problem
- $\mathbb{L}_2$-regularity of Z ,$\sum_{i=0}^{N-1}\int_{t_i}^{t_{i+1}}\mathbb{E}|\bar{Z}_{t_i} - Z_t|^2dt$

Gobet and Labart [8] has shown the error 2.10 is in order $O(\frac{1}{N})$ provided the conditional expectation in each time step is evaluated accurately. In general the error arisen from the approximation of the conditional expectations will propagate and accumulate to later steps. Thus a larger number of time steps will not always reduce the total error. And an optimal time discretization N should be chosen carefully. However as will show later under certain circumstance, we do not need to worry about N.

2.2 Stochastic Mesh method

In this section, we will present the details about stochastic mesh method. The stochastic mesh method was first introduced by Broadie and Glasserman [2] to price high-dimensional American option. They have defined a mesh estimator and a path estimator which are theoretically high biased and low biased estimator respectively. Both estimators converge to the true value when the number of paths b tends to infinity. Combining these 2 estimators forms an efficient confidence interval. Our method is largely inspired by the mesh estimator.

A general construction of the mesh is illustrated in Figure 2.1. In the first phase, we generate independent paths of forward process $X_{t_0}, X_{t_1}, \ldots, X_{t_N}$. In the second phase, we "forget" which node i generated which node $i+1$ and interconnect all nodes at consecutive time steps for the backward induction which is shown in Figure 2.2.

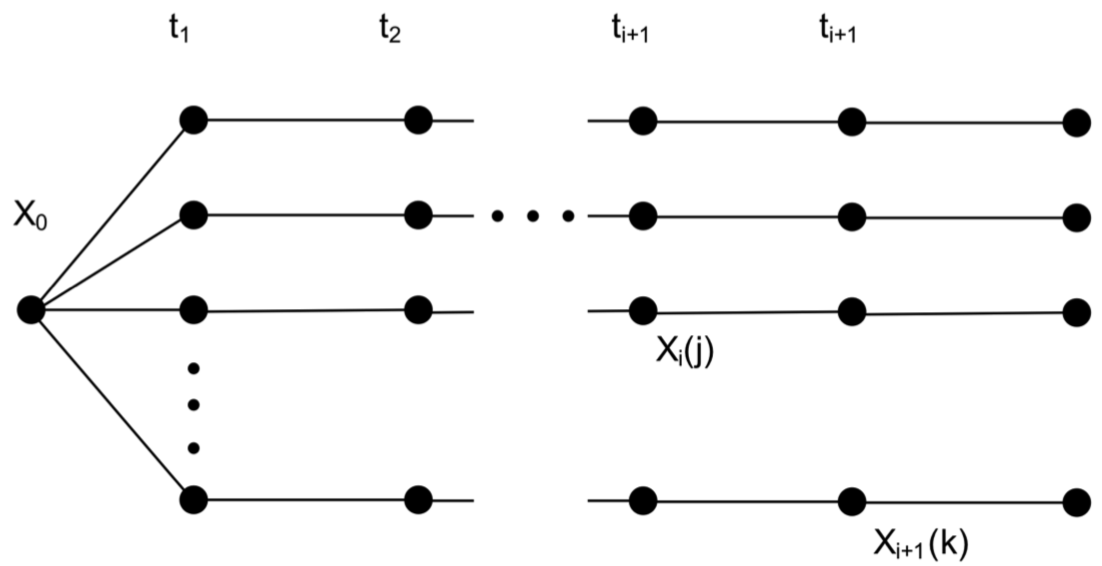

Figure 2.1: The real construction of mesh

In our BSDE problem, we use formulations 2.2-2.4 to solve Y and Z at each node. We use $X_i(j)$ to denote the j th node at t_i date for $i = 0, \ldots, N$ and $j = 1, \ldots, b$ and $W_i(j)$ to denote the associated Brownian Motion.

$$X_{i+1}(j) = X_i(j) + \mu(t_i, X_i(j))\Delta t_i + \sigma(t_i, X_i(j))\Delta W_i(j) \tag{2.11}$$

Similarly we use $\hat{Z}_i(j)$ and $\hat{Y}_i(j)$ to denote the estimated value at that node. At terminal nodes, we set $\hat{Y}_N(j) = \Phi(X_N(j))$. Then we work backward recursively by

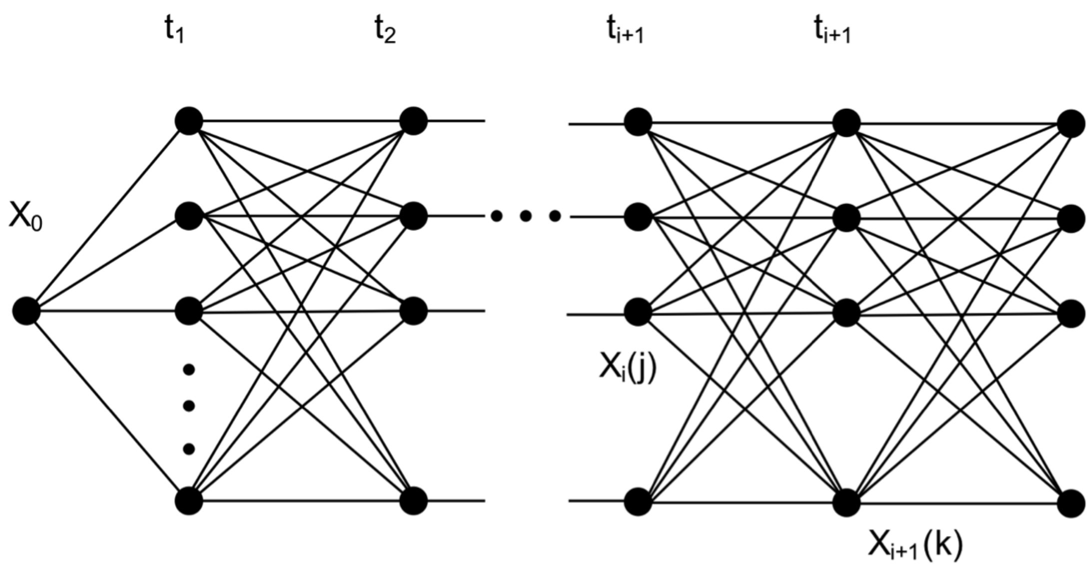

Figure 2.2: The interconnected mesh

defining

$$\hat{Z}_i(j) = \frac{1}{b}\sum_{k=1}^{b} \hat{Y}_{i+1}(k)\frac{W_{i+1}(k) - W_i(j)}{t_{i+1} - t_i} w(i, j, k) \tag{2.12}$$

$$\hat{Y}_i(j) = \frac{1}{b}\sum_{k=1}^{b} [\hat{Y}_{i+1}(k) + f(t_i, X_i(j), \hat{Y}_{i+1}(k), \hat{Z}_i(j)] w(i, j, k) \tag{2.13}$$

where $w(i, j, k)$ is a weight function that connects the node $X_i(j)$ to node $X_{i+1}(k)$. $\hat{Z}_i(j)$ and $\hat{Y}_i(j)$ are estimates that approximate $Z(t_i, X_i(j))$ and $Y(t_i, X_i(j))$ respectively. At time $t = 0$ only $i = 1$ is applicable in equation 2.12 and 2.13 and $\hat{Y}_0 \equiv \hat{Y}_0(1)$ is our final estimator of Y_0. In the application of option pricing problem , it means the price estimator of an option.

In our experiments we use the average density function which is defined in [2] as our weight function. We rewrites the definition here

$$w(i, j, k) = \frac{f(t_i, X_i(j), X_{i+1}(k))}{\frac{1}{b}\sum_{j'=1}^{b} f(t_i, X_i(j'), X_{i+1}(k))} \tag{2.14}$$

where $f(t_i, x, y)$ is the transition density of X_{i+1} conditioning on $X_i = x$. Although other choice of weight function is possible (see [3]) when transition density is not available, they require much more computation effort and remain another topic. This can be investigated in further research. In this experience, we focus on average density weight function and will test against cases when such function is available.

The justification of using (2.14), its intrinsic connection with the mesh construction and the "forget" mechanism has been explained by Mark and Paul in [2]. With this choice of weights function, they have shown the convergence of the mesh estimator. In addition they have shown in the framework of pricing American option that the potential exponential variance reduces to zero using average density function. Similar conclusion could not be drawn in the framework of BSDE since Y and Z are twisted in each time step. Here we give a sketch of proof to show the convergence and the reason why we choose average density function.

Suppose that we want to evaluate the following conditional expectation

$$C(t_{i+1}, x) = \mathbb{E}[Q(t_{i+1}, X_{i+1})|X_i = x] \tag{2.15}$$

where $Q(t, x)$ is smooth enough to ensure the existence of this conditional expectation. In our application, Q is the function either used to compute Z or Y and C can either be Z or Y respectively. Observe that

$$\begin{aligned}\mathbb{E}[Q(t_{i+1}, X_{i+1})|X_i = x] &= \int Q(t_{i+1}, u) f(t_i, x, u) du \\ &= \int Q(t_{i+1}, u) \frac{f(t_i, x, u)}{g(t_{i+1}, u)} g(t_{i+1}, u) du \\ &= \mathbb{E}[Q(t_{i+1}, X_{i+1}) \frac{f(t_i, x, X_{i+1})}{g(t_{i+1}, X_{i+1})}]\end{aligned} \tag{2.16}$$

where $g(t_{i+1}, .)$ is the density function of X_{i+1}.

This expression allows us to approximate the conditional expectation $\mathbb{E}[Q(t_{i+1}, X_{i+1})|X_i = X_i(j)]$ for $j = 1, \ldots, b$ by an expectation which can be computed via Monte Carlo simulations. Define

$$\hat{C}(t_i, X_i(j)) = \frac{1}{b} \sum_{k=1}^{b} \hat{Q}(t_{i+1}, X_{i+1}(k)) w(i, j, k) \tag{2.17}$$

where $w(i, j, k) = f(t_i, X_i(j), X_{i+1}(k))/g(t_{i+1}, X_{i+1}(j))$. Therefore $\hat{C}$ is an unbiased estimator of C as b tends to infinity with rate of convergence $O(1/\sqrt{b})$.* Compared to weights function (2.14), average density function actually means

$$g(t_{t+1}, X_{i+1}(k)) = \frac{1}{b}\sum_{j=1}^{b} f(t_i, X_i(j), X_{i+1}(k)) \tag{2.18}$$

$g(t_{i+1}, \cdot)$ by definition is the density function of X_{i+1} which relies heavily on how we construct the mesh, precisely on how we generate the node $X_{i+1}(j)$ for $j = 1, \ldots, b$. Here comes the reason why we need to "forget" which node generates which node. The use of average density function has an intrinsic connection to this reason because it can be interpreted in the following way. Suppose that from each of the mesh nodes $X_i(j)$,$j = 1, \ldots, b$, we generate exactly one successor X_{i+1},$j = 1, \ldots, b$ by using the transition density $f(t_i, X_i(j), \cdot)$. If then we draw a value randomly and uniformly from $\{X_{i+1}(1), X_{i+1}(2), \ldots, X_{i+1}(b)\}$, the value drawn is distributed according to the average density (2.18) conditioning on $\{X_i(1), X_i(2), \ldots, X_i(b)\}$. As a result, using the average density is equivalent to generating b independent paths of the forward process and then "forgetting" which nodes were from which nodes.

2.3 Algorithm enhancements

Following [2], we use the idea of control variate to improve the efficiency of our basic mesh algorithm. We define two application of control variate for improving the mesh estimator. Let me make a detailed description of these applications.

The first application of control variate is used to improve the estimates $Y_i(j)$ and $Z_i(j)$ at each mesh node. These control variates are called *inner controls*, because they are applied within the mesh. The second application of control variate is used to improve the mesh estimates of Y_0. They are called *outer controls* because they are applied when M individual mesh estimates at time $t = 0$ are computed.

*Think about a normal Monte Carlo scheme to estimate the expectation $\mathbb{E}[f(X)]$ and X is a random variable, then $1/b\sum_{i=1}^{b} f(\hat{X}(i))$ is a unbiased estimator of $\mathbb{E}[f(X)]$ with order $O(1/\sqrt{b})$

We begin by describing the inner controls. From previous section, we aim to approximate one condition expectation for either Y or Z. In general form,

$$\mathbb{E}[Q(t_{i+1}, X_{i+1})|X_i = x]$$

is approximated by

$$\frac{1}{b}\sum_{k=1}^{b} \hat{Q}(t_{i+1}, X_{i+1}(k))w(i,j,k)$$

Suppose that we have a known formula for $\nu = \mathbb{E}[\nu(t_{i+1}, X_{i+1})|X_i = X_i(j)]$ or ν can be evaluated quickly and accurately. For example, ν could be the expected value of first underlying (first component in X) $\nu = \mathbb{E}[X^1_{i+1}|X_i = X_i(j)]$. Or ν could be the related European option $\nu = \mathbb{E}[h(t_{i+1}, X_{i+1})|X_i = X_i(j)]$ which has a Black-Scholes formula to evaluate. Then we construct the inner control estimate.

$$\hat{\nu} = \frac{1}{b}\sum_{k=1}^{b} \nu(t_{i+1}, X_{i+1}(k))w(i,j,k)$$

Following the same argument leading to equation (2.16), we can see that $\mathbb{E}[\hat{\nu}] = \nu$. Information about the known error between $\hat{\nu}$ and ν can be used to reduce the unknown error between the $\hat{C}$ and C. However, the presence of the weights complicates the procedure. With the introduction of inner control variate ν, we can define a new estimator to C as

$$\hat{C}_\nu(t_{i+1}, X_i(j)) = \frac{\frac{1}{b}\sum_{k=1}^{b} w(i,j,k)[\hat{Q}(t_{i+1}, X_{i+1}(k)) - \beta(\nu(t_{i+1}, X_{i+1}(k)) - \nu)]}{\frac{1}{b}\sum_{k=1}^{b} w(i,j,k)} \tag{2.19}$$

Note that the expectation of this formula is the same as (2.17) because $\mathbb{E}[1/b\sum_{k=1}^{b}(\nu(t_{i+1}, X_{i+1}(k)) - \nu)w(i,j,k)] = 0$ and the expectation of denominator is one. The variance of (2.19) could be reduced as long as Q and ν are correlated. The optimal β is chosen to solve the weighted least-square problems:

$$\min_{\alpha,\beta} \frac{1}{b}\sum_{k=1}^{b} w(i,j,k)[\hat{Q}(t_{i+1}, X_{i+1}(k) - (\alpha + \beta(\nu(t_{i+1}, X_{i+1}(k))]$$

We call the β that solves this least-square problem β_{opt}. Then the controlled estimator is

$$\alpha + \beta_{opt}\nu$$

After further simplification, one can show that

$$\beta_{opt} = \frac{CE - AB}{DE - B^2} \tag{2.20}$$

$$\alpha = \frac{AD - BC}{DE - B^2} \tag{2.21}$$

where

$$A = \sum_{k=1}^{b} \hat{Q}(t_{i+1}, X_{i+1}(k))w(i,j,k),$$
$$B = \sum_{k=1}^{b} \nu(t_{i+1}, X_{i+1}(k))w(i,j,k),$$
$$C = \sum_{k=1}^{b} \hat{Q}(t_{i+1}, X_{i+1}(k))\nu(t_{i+1}, X_{i+1}(k))w(i,j,k),$$
$$D = \sum_{k=1}^{b} \nu^2(t_{i+1}, X_{i+1}(k))w(i,j,k),$$
$$E = \sum_{k=1}^{b} w(i,j,k)$$

The outer control variates are fairly standard. We use M independent meshes to generate the estimate $\hat{Y}^{(i)}$, $i = 1, \dots, M$ of the backward process Y_0 at time $t = 0$. Suppose we have a known formula for $u = \mathbb{E}[h(T, X_T)]$ or u could be evaluated quickly and accurately. For example, in option pricing problem, h could be Black-Schole European option. We use each mesh to generate one estimate $\hat{u}^{(i)}$,$i = 1, \dots, M$ of u. Then we can form our outer control estimate of Y_0.

$$\hat{Y}_0^u = \frac{1}{M}\sum_{i=1}^{M} \hat{Y}^{(i)} - \beta(\frac{1}{M}\sum_{i=1}^{M} \hat{u}^{(i)} - u) \tag{2.22}$$

Unlike the inner control variate, we do not define an outer control variate for process Z simply because we are only interested in the current price of the option. We can define a similar outer control variate for Z if we want to know the precise hedging strategy at initial time.

Chapter 3

Application

3.1 Pricing of European style option

In this section we aim to employ previous method to the application of option pricing problem. The structure of European style contingent claims are very similar to that of BSDE. We know the terminal structure of an option in future and want to find the initial value or its price today. Later we will show some simulation results. Before modeling the pricing problem,we need some assumptions for the financial market.

1. D risky assets, $dS_t^d = S_t^d(\mu_t^d dt + \sum_{q=1}^{Q} \sigma_t^{d,q} dW_t^q), d = 1, \ldots, D$. The drift μ^d and volatility $\sigma^{d,q}$ are bounded and predictable.

2. 1 risk-free asset (money market). $dS_t^0 = S_t^0 r_t dt$, where r_t is the short rate which is bounded and predictable.

3. Existence of θ_t which is bounded and predictable such that $\theta_t \sigma_t = \mu_t - r_t \mathbf{1}$ ($\mathbf{1}$ is the vector with all its component equals 1).

4. Complete market which means σ_t is invertible. A necessary and sufficient condition for σ_t to be invertible is $Q = D$ and the d risky assets are not redundant. That means we can write previous assumption in this way $\theta_t = \sigma_t^{-1}(\mu_t - r_t \mathbf{1})$.

5. ϕ_t is the row vector of money amount invested in the risky assets assuming no investment constraint (no transaction cost, short-selling is allowed and etc).

First we assume no constraint on portfolio selection. With self-financing strategy, the wealth process Y_t can be written as

$$\begin{aligned} dY_t =& \sum_{d=1}^{D} \phi_t^d \frac{dS_t^d}{S_t^d} + (Y_t - \sum_{d=1}^{D} \phi_t^d) r_t dt \\ =& \phi_t(\mu_t dt + \sigma_t dW_t) + (Y_t - \phi_t \mathbf{1}) r_t dt \\ =& [Y_t r_t + \phi_t(\mu_t - r_t \mathbf{1})]dt + \phi_t \sigma_t dW_t \\ =& [Y_t r_t + Z_t \theta_t]dt + Z_t dW_t \end{aligned} \tag{3.1}$$

where $Z_t = \phi_t \sigma_t$. With a specification for terminal value $Y_T = \Phi(S_t; 0 \leq t \leq T)$, (Y, Z) solves a linear BSDE problem with the driver defined by

$$f(t, x, y, z) = -r_t y - z\theta_t \tag{3.2}$$

One thing to note is that Φ allows a path-dependent payoff at maturity T. Then Y_0 is the option price given the payoff Φ. The driver defined in (3.2) is linear in (y, z) thus obvious satisfies the Lipschitz assumption in chapter 1. This linear driver actually means the famous Black-Scholes model. To see this, we need some standard results for linear BSDE [6].

Proposition 3.1. *Let (β, μ) be a bounded $(\mathbb{R}, \mathbb{R}^d)$-valued progressively measurable process, ϕ is a progressively measurable process, s.t $\mathbb{E}[\int_0^T |\phi|^2 ds] < \infty$, ξ is a progressively measurable process, s.t $\mathbb{E}[|\xi|^2 ds] < \infty$. We consider the following linear BSDE:*

$$-dY_t = (\phi_t + \beta_t Y_t + \mu_t Z_t)dt - Z_t dW_t; \quad Y_T = \xi_T \tag{3.3}$$

Then the above equation has a unique solution (Y, Z), and Y is given explicitly by

$$Y_t = \mathbb{E}[\xi_T \Gamma_{t,T} + \int_t^T \Gamma_{t,s} \phi_s ds | \mathcal{F}_t] \tag{3.4}$$

where $(\Gamma_{t,s})_{t \leq s}$ is the adjoint process defined by the forward linear SDE

$$d\Gamma_{t,s} = \Gamma_{t,s}(\beta_s ds + \mu_s dW_s); \quad \Gamma_{t,t} = 1 \tag{3.5}$$

or

$$\Gamma_{t,T} = \exp[\int_t^T (\beta_s - \frac{1}{2}\mu_s^2)ds + \int_t^T \mu_s dW_s] \tag{3.6}$$

With the identification of $\phi_t = 0, \beta_t = -r, \mu_t = -\theta_t$, we can conclude that the solution to (3.1) is

$$Y_t = \mathbb{E}[\exp(-r(T-t) - \frac{1}{2}\int_t^T \theta_s^2 ds - \int_t^T \theta_s dW_s)\xi_T|\mathcal{F}_t] \tag{3.7}$$

which is exactly the risk-neutral pricing formula for European-style option with Radon-Nikodym derivative defined as

$$\frac{d\mathbb{Q}}{d\mathbb{P}} = \exp(-\frac{1}{2}\int_0^T \theta_s^2 ds - \int_0^T \theta_s dW_s) \tag{3.8}$$

Second we impose a bid-ask spread constraint for interest rates. The investor borrows money at higher rate R_t and lends money to money market at rate r_t. A slight modification of self-financing strategy leads to the following wealth process

$$\begin{aligned} dY_t =& \sum_{i=1}^d \phi_t^i \frac{dS_t^i}{S_t^i} + (Y_t - \sum_{i=1}^d \phi_t^i)_+ r_t dt - (Y_t - \sum_{i=1}^d \phi_t^i)_- R_t dt \\ =& \phi_t(\mu_t dt + \sigma_t dW_t) + [(Y_t - \phi_t \mathbf{1}) + (Y_t - \phi_t \mathbf{1})_-] r_t dt - (Y_t - \phi_t \mathbf{1})_- R_t dt \\ =& \phi_t(\mu_t dt + \sigma_t dW_t) + (Y_t - \phi_t \mathbf{1}) r_t dt - (R_t - r_t)(Y_t - \phi_t \mathbf{1})_- dt \\ =& [Y_t r_t + \phi_t(\mu_t - r_t \mathbf{1}) - (R_t - r_t)(Y_t - \phi_t \mathbf{1})_-]dt + \phi_t \sigma_t dW_t \\ =& [Y_t r_t + Z_t \theta_t - (R_t - r_t)(Y_t - \phi_t \mathbf{1})_-]dt + Z_t dW_t \\ =& [Y_t r_t + Z_t \theta_t - (R_t - r_t)(Y_t - Z_t \sigma_t^{-1} \mathbf{1})_-]dt + Z_t dW_t \end{aligned} \tag{3.9}$$

Under bid-ask spread constraint for interest rate, the new driver is obtained as

$$f(t,x,y,z) = -r_t y - z\theta_t + (R_t - r_t)(y - z\sigma_t^{-1}\mathbf{1})_- \tag{3.10}$$

The additional term in the first equality of (3.9) compared to (3.1) represents the additional cost when borrowing from the money market. The driver (3.10) is still Lipschitz continuous w.r.t (y, z). As a result the existence and uniqueness of the solution is guaranteed. This non-linear constraint shows one advantage of BSDE over Black-Scholes model. The latter cannot deal with case of bid-ask spread for interest rate.

Throughout our simulation, we run the whole algorithm 10 times to get the variance. Inspired by [2], we use the following inner control variate and outer control variate to reduce the variance.

The first inner control variate is

$$\nu^{(1)} = \mathbb{E}[S_{i+1}^{d^*} | S_i = X_i(j)] = S_i^{d^*} \exp(\mu_{t_i} \Delta t_i) \tag{3.11}$$

where S_i is short for S_{t_i}, $d^* = \mathrm{argmax}\{\mathrm{S_i^d}, \mathrm{d} = 1, \ldots, \mathrm{D}\}$ and S_i^d is the dth underlying value at time $t = t_i$. This is the forward value of the largest underlying at mesh node $X_i(j)$.
A second inner variate control defined as

$$\nu^{(2)} = \mathbb{E}[\exp(-r_{t_i} \Delta t_i)(S_{i+1}^{d^*} - K)_+ | S_i = X_i(j)] \tag{3.12}$$

is also applicable. The notation remains unchanged as the first inner control variate. This variate is indeed the European call value on the largest underlying at mesh node $X_i(j)$ which has a well-known Black-Scholes formula to evaluate. However in practice the empirical estimation of $\nu^{(2)}$ when b is small is often zero thus leading to an infinite value for Z and Y. We prefer not to use it.
The out control variate is defined as

$$u = \mathbb{E}[\exp(-rT)(\max(S_T^1, \ldots, S_T^D) - K)_+] \tag{3.13}$$

This is an European max call option. Though it does not have a closed formula, it can be evaluated by standard Monte-Carlo method quickly and accurately. In particular, we use the uniform random number generator called by Park and Miller [16] and then generate the Gaussian distribution numbers via an approximation to the inverse of normal cumulative distribution function [15].

3.1.1 1-dimensional BSDE

We begin our experience by one-dimensional case and use the same problem as [9] which is a good benchmark for our experiment. Considering the bid-ask spread interest rate as a constraint, we use the driver defined in (3.10). To reduce the variance, we use the first inner control variate and the outer control variate at the same time. Parameter setting can be found in following table.

μ	σ	r	R	T	S_0	K_1	K_2
0.05	0.2	0.01	0.06	0.25	100	95	105

Table 3.1: Parameter settings for $\Phi(\mathbf{S}) = (S_T - K_1)_+ - 2(S_T - K_2)_+$

The terminal condition is a call combination :$\Phi(\mathbf{S}) = (S_T - K_1)_+ - 2(S_T - K_2)_+$. The empirical result shown by [9] is 2.95. In our stochastic mesh method ,we have three parameters concerning the variance, the number of time steps N, the number of paths b for each mesh and the number of simulations M. From the error analysis (2.10) in Chapter 2, we can see the error from simulating forward process can be totally eliminated because under Black-Scholes model the dynamic of forward process have a closed formula. That is to say increasing N should not have any substantial improvement to our test result. The main error comes from the approximation of two conditional expectations(2.3)-(2.4). In this experiment, we test with $N = 5, 10, 20$ and increase b and M independently to see the accuracy of this method.

The testing result can be found in Table 3.2, 3.3 and 3.4. The value in parentheses is the estimated standard deviation. By comparing the estimated standard deviation across b and M, we found following relationship

$$\textit{standard deviation} \sim \frac{1}{\sqrt{bM}}$$

This proportional relation could be explained by Central Limit Theorem in the following way. First, the order $O(1/\sqrt{M})$ is merely the standard result of the standard Monte-Carlo simulation because each mesh is independent of each other and the arithmetic average across M meshes is used to estimate the price. Then since in Chapter 2 where we have used again the standard Monte-Carlo simulation (2.17) to approximate the conditional expectations (2.16) inside the emesh, thus the order $O(1/\sqrt{b})$ at each node could be obtained by similar argument. Note that the final price is the first node of our mesh and thus the error is in the same order as other nodes $O(1/\sqrt{b})$. Though b and M have the same influence over the rate of convergence, the computation cost for increasing b and M is highly different. For example when $N = 10$, it takes 12 seconds to run the whole algorithm once with $b = 32, M = 512$ however it takes 169 seconds with $b = 512, M = 32$ although these 2 settings give approximately the same standard deviation (0.032 and 0.035 respectively). Empirical results show that the computation effort is approximately:

$$Computation\ time \sim b^2 M$$

By comparing the estimated price within one table, we see that increasing b gives a slightly less biased estimate than increasing M. By comparing the estimated prices across these tables, we don't see any substantial improvements regarding estimated price or standard deviation by increasing the number of time steps N. This fact confirms our prior conclusion which suggests that it is not necessary to keep N large to get a good result. This could be very useful to the following tests, especially high-dimensional cases where large computation effort is required.

The Table 3.5 shows the computation cost expressed in seconds for the case $N = 10$ (on a 1.86GHz Intel Core2 Duo processor). The equivalent log-log plot is shown in Figure 3.1. Both the table and the plot show that the cost grows linearly in M and quadratically in b which serves as a general rule for other all the rest tests as well.

In conclusion, the number of simulation M and the number of path b have the same influence over standard deviation while the computation cost for b is twice

M	b=32	b=64	b=128	b=256	b=512
32	2.931(0.113)	2.938(0.12)	2.863(0.063)	2.899(0.043)	2.938(0.026)
64	3.028(0.101)	2.922(0.065)	2.91(0.046)	2.928(0.033)	2.96(0.025)
128	2.915(0.064)	2.879(0.044)	2.916(0.033)	2.930(0.024)	2.928(0.016)
256	2.879(0.049)	2.939(0.033)	2.927(0.023)	2.902(0.016)	2.919(0.012)
512	2.984(0.036)	2.950(0.022)	2.939(0.016)	2.924(0.012)	2.936(0.008)

Table 3.2: 1D call combinations with $N = 5$

M	b=32	b=64	b=128	b=256	b=512
32	2.946(0.127)	3.044(0.098)	3.093(0.074)	2.949(0.043)	2.997(0.035)
64	3.034(0.092)	3.010(0.062)	2.996(0.049)	2.970(0.032)	2.937(0.021)
128	3.007(0.063)	2.916(0.041)	2.986(0.032)	2.950(0.022)	2.942(0.016)
256	3.016(0.047)	3.001(0.035)	2.946(0.024)	2.942(0.015)	2.969(0.011)
512	2.997(0.032)	2.953(0.023)	2.984(0.016)	2.957(0.011)	2.959(0.008)

Table 3.3: 1D call combinations with $N = 10$

heavier with only slight bias improvement. Under Black-Scholes Model the time step N is really not important. However in general, if the forward process does not follow such a simple dynamic as Black-Schole world the right choice of N is crucial. Because we would expect to increase N to reduce the global error from estimating forward process and terminal value while the error from estimating the conditional expectations will propagate through time steps. In general case the time step N should have some dependencies on mesh size b. In our very special case where the forward process could be solved explicitly and the payoff is European style, the choice of $N = 1$ is not only a sufficient condition but also a necessary condition in order to minimize the error. For the rest of test, we will use $N = 1$ as long as the payoff is European style and the dynamic has an explicit solution.

M	b=32	b=64	b=128	b=256	b=512
32	3.118(0.155)	3.004(0.077)	2.930(0.064)	3.022(0.048)	2.960(0.026)
64	2.885(0.096)	3.000(0.055)	2.930(0.040)	2.948(0.032)	2.958(0.022)
128	2.989(0.064)	3.015(0.043)	2.990(0.030)	2.953(0.023)	2.938(0.017)
256	3.095(0.045)	3.007(0.033)	2.965(0.024)	2.998(0.016)	2.954(0.012)
512	3.044(0.036)	3.025(0.022)	2.965(0.016)	2.966(0.011)	2.960(0.008)

Table 3.4: 1D call combinations with $N = 20$

M	b=32	b=64	b=128	b=256	b=512
32	0.85	2.87	10.83	42.45	169.22
64	1.62	5.63	21.66	84.77	338.36
128	3.17	11.21	43.12	169.64	676.66
256	6.18	22.32	86.27	339.24	1,354.25
512	11.95	44.56	172.43	678.19	2,707.34

Table 3.5: 1D computation time for $N = 10$ expressed in seconds

3.1.2 High-dimensional BSDE

Multi-dimensional BSDE is notorious for its dimensionality curse. In numerical methods of solving BSDE, high-dimensional BSDE problem is often known as high computational cost, low rate of convergence or sometimes biased. In this experiment various examples will be tested to show the efficiency of our algorithm in high-dimensional cases. The weight function is slightly modified to accommodate high-dimensionality.

$$w(i,j,k) = \frac{\prod_{d=1}^{D} f^d(t_i, X_i^d(j), X_{i+1}^d(k))}{\frac{1}{b}\sum_{j'=1}^{b}\prod_{d=1}^{D} f^d(t_i, X_i^d(j'), X_{i+1}^d(k))} \tag{3.14}$$

where $f^d(t_i, X_i^d(j'), X_{i+1}^d(k))$ is the transition density function of dth asset.

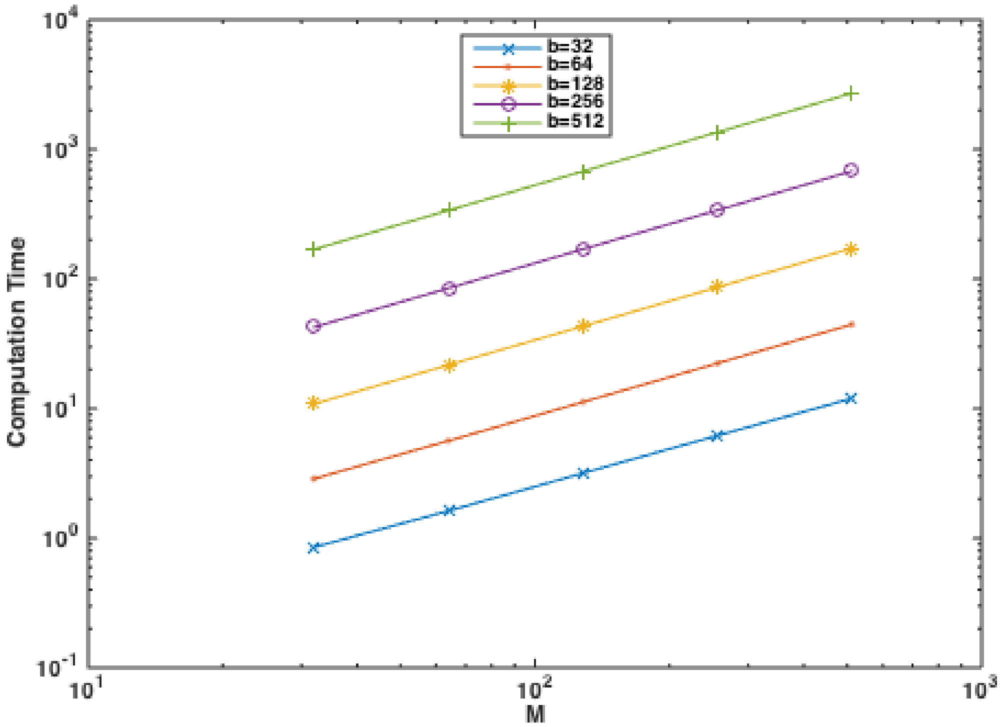

Figure 3.1: Computation time cost

High-dimensional basket call option

We begin our test with a basket call option. There are 20 assets with each asset independently modeled as geometric Brownian Motion. The dynamics are unchanged and different interest rates are not considered. That is to say we are setting $r = R$ which reduces the non-linear driver of (3.10) to linear driver (3.2). Parameter setting can be found in Table 3.6.

μ	σ	r	R	T	S_0	K
0.05	0.2 Id	0.06	0.06	0.25	100	95

Table 3.6: Parameter settings for $\Phi(\mathbf{S}) = (\frac{1}{20}\sum_{d=1}^{20} S_T^d - K)_+$

The payoff is a basket call option : $\Phi(\mathbf{S}) = (\bar{S}_T - K)_+$ where $\bar{S}_T = 1/20 \sum_{d=1}^{d=20} S_T^d$. Via the normal Monte Carlo simulation using the Park and Miller [16] generator

M	b=32	b=64	b=128	b=256	b=512
32	6.296(0.091)	6.429(0.057)	6.459(0.044)	6.412(0.037)	6.436(0.024)
64	6.346(0.066)	6.398(0.042)	6.452(0.037)	6.445(0.021)	6.432(0.017)
128	6.353(0.042)	6.446(0.032)	6.425(0.022)	6.399(0.016)	6.428(0.009)
256	6.442(0.035)	6.434(0.022)	6.426(0.015)	6.404(0.011)	6.401(0.007)
512	6.440(0.023)	6.446(0.016)	6.418(0.011)	6.413(0.008)	6.409(0.005)

Table 3.7: 20D basket call combination with N=1 without control variates

M	b=32	b=64	b=128	b=256	b=512
32	6.331(0.064)	6.393(0.044)	6.456(0.035)	6.401(0.027)	6.443(0.018)
64	6.375(0.049)	6.436(0.031)	6.431(0.028)	6.435(0.015)	6.411(0.013)
128	6.397(0.032)	6.426(0.025)	6.426(0.017)	6.404(0.011)	6.416(0.007)
256	6.410(0.026)	6.421(0.016)	6.420(0.011)	6.408(0.008)	6.408(0.005)
512	6.422(0.018)	6.433(0.012)	6.415(0.008)	6.411(0.006)	6.412(0.004)

Table 3.8: 20D basket call combination with N=1

as random number engine, we can get a very accurate price 6.414(0.002) as our reference. We will first run our algorithm without the control variates and then the same algorithm with control variates to see the variance reducing efficiency.

The results are shown in Table 3.7 and the Table 3.8. To make a comparison we have plotted the standard error against b for $M = 32$ for both case in Figure 3.2. This plot shows that the control variate technique is very efficient in our algorithm. Thus we will run all the rest tests with control variates. In both tables the price converges to 6.41 which is in accordance with previous Monte-Carlo simulations.

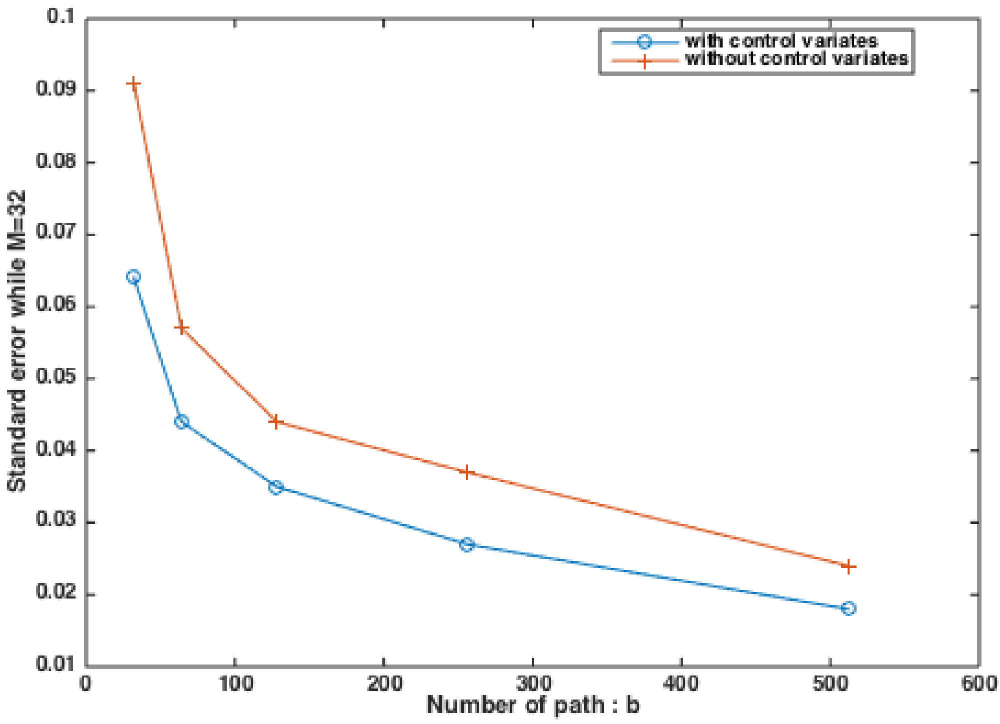

Figure 3.2: Standard error comparison with and without control variate

high-dimensional noised call combination option

In the second test, only the first asset is used to evaluate terminal payoff. All assets follow the same dynamics(μ, σ, S_0 are equal) and are independent (with zero correlations). Different interest rates are considered. Then the driver is only Lipschitz continuous. Using the same parameter setting as in Table 3.1 and a call combination option payoff:$\Phi(\mathbf{S}) = (S_T^1 - K_1)_+ - 2(S_T^1 - K_2)_+$, the problem is equivalent one-dimensional case where the price equals 2.95. Other 19 assets serves merely as noise. The purpose of this test is to see how accurate and robust our algorithm is if further noise is introduced.

The result could be found in Table 3.10. Unlike the one-dimensional case where the price converges quickly to the reference value 2.95, the noised price is low biased to 2.83 from 2.95. This suggests that the stochastic mesh method using average density function as weight function may not capture the dominant component of assets.

$\boldsymbol{\mu}$	$\boldsymbol{\sigma}$	$\boldsymbol{r}$	$\boldsymbol{R}$	$\boldsymbol{T}$	$\boldsymbol{S_0}$	$\boldsymbol{K_1}$	$\boldsymbol{K_2}$
0.05	0.2 Id	0.01	0.06	0.25	100	95	105

Table 3.9: Parameter settings for $\Phi(\mathbf{S}) = (S_T^1 - K_1)_+ - 2(S_T^1 - K_2)_+$

M	b=32	b=64	b=128	b=256	b=512
32	2.978(0.147)	2.885(0.117)	2.944(0.07)	2.801(0.047)	2.766(0.034)
64	2.952(0.09)	2.91(0.062)	2.938(0.051)	2.839(0.033)	2.839(0.025)
128	3.093(0.068)	2.949(0.048)	2.886(0.037)	2.841(0.024)	2.855(0.017)
256	3.098(0.048)	2.974(0.031)	2.854(0.024)	2.875(0.016)	2.833(0.012)
512	3.127(0.033)	2.979(0.024)	2.87(0.017)	2.85(0.011)	2.83(0.008)

Table 3.10: 20D noised call combination with N=1

Average density function defined in (3.14) assumes all assets have equal influence over the weight function. However only the transition density function of first asset is used in the payoff. Other transition density function "averaged" the dominant one and introduced the bias. Therefore we can expect the bias to disappear when the number of assets shrinks to one. Thankfully when pricing options, irrelevant processes are rarely introduced. This problem in robustness is not taken as a serious one. We expect other choice of weight function to solve this problem. In [3], Paul introduced the optimized weight function whose idea is that if the weight function can correctly price some simple instruments if can correctly price other complex instruments. That is a promising candidate especially when transition density is not available however requires more computation effort. We leave it as further research. Here we propose another remedy for this special noised problem. We can further define a second weight function for the transition density used in original weight function. The second weight function is proportional to the weight of assets in final payoff. In mathematical form, we replace the weight function of (3.14) by

$$w(i,j,k) = \frac{f^*(t_i, X_i(j), X_{i+1}(k))}{\frac{1}{b}\sum_{j'=1}^{b} f^*(t_i, X_i(j'), X_{i+1}(k))} \tag{3.15}$$

where

$$\log f^*(t_i, X_i(j), X_{i+1}(k)) = \sum_{d=1}^{D} u^d \log f^d(t_i, X_i^d(j), X_{i+1}^d(k)) \tag{3.16}$$

and

$$\begin{aligned} u^d &\sim \left|\frac{\partial \Phi((S)}{\partial S^d}\right| \\ \sum_{d=1}^{D} u^d &= 1 \end{aligned} \tag{3.17}$$

Using this revised weight function, the noised problem no longer exists since it is actually reduced to one-dimensional case.

high-dimensional call combination option without noise

Our third multi-dimensional BSDE experiment evaluates the terminal payoff using all assets while keeping different interest rates. Therefore the driver is Lipschitz continuous again. Parameter setting is exactly the same as Table 3.9. Only payoff function is slightly modified to incorporate all underlying changes : $\Phi(\mathbf{S}) = (\bar{S}_T - K_1)_+ - 2(\bar{S}_T - K_2)_+$ where $\bar{S}_T = 1/20\sum_{d=1}^{d=20} S_T^d$. All underlings are theoretically equally important. No more redundant underlying basically means no more noise. From previous simulation results, without noise our stochastic mesh method using average density weight function normally plays a very good job in pricing problem. Since we don't have a reference price to compare with, we present our result in Table 3.12 as a first attempt for further research comparison. However we can compare with the same experiment except for keeping single interest rate. The result for single interest rate is presented in Table 3.13. The corresponding parameter setting is in Table 3.11 Comparing these two tables, we find an interesting yet reasonable result. The two tables are identical in all aspects which means the lending interest rate r is never used. Standard Monte-Carlo simulation shows the price of call combination option with single interest rate is 6.283(0.002). Both of our experiments are very accurate and have proved the conclusion that the lending interest rate is never used in hedging process.

μ	σ	r	R	T	S_0	K_1	K_2
0.05	0.2 Id	0.06	0.06	0.25	100	95	105

Table 3.11: Parameter settings for $\Phi(\mathbf{S}) = (\bar{S}_T - K_1)_+ - 2(\bar{S}_T - K_2)_+$

M	b=32	b=64	b=128	b=256	b=512
32	6.235(0.068)	6.281(0.04)	6.331(0.031)	6.268(0.024)	6.309(0.016)
64	6.236(0.045)	6.328(0.027)	6.3(0.023)	6.3(0.014)	6.28(0.011)
128	6.273(0.029)	6.29(0.023)	6.303(0.015)	6.276(0.01)	6.286(0.006)
256	6.288(0.023)	6.3(0.014)	6.292(0.01)	6.283(0.007)	6.286(0.005)
512	6.294(0.016)	6.305(0.011)	6.288(0.007)	6.283(0.005)	6.283(0.003)

Table 3.12: 20D averaged call combination with different interest rates

M	b=32	b=64	b=128	b=256	b=512
32	6.235(0.068)	6.281(0.04)	6.331(0.031)	6.268(0.024)	6.309(0.016)
64	6.236(0.045)	6.328(0.027)	6.3(0.023)	6.3(0.014)	6.28(0.011)
128	6.273(0.029)	6.29(0.023)	6.303(0.015)	6.276(0.01)	6.286(0.006)
256	6.288(0.023)	6.3(0.014)	6.292(0.01)	6.283(0.007)	6.286(0.005)
512	6.294(0.016)	6.305(0.011)	6.288(0.007)	6.283(0.005)	6.283(0.003)

Table 3.13: 20D averaged call combination with single interest rate

To be further convincing, we have run another three similar experiments. First one in Table 3.14 starts with a lower initial level $\mathbf{S_0} = 90$ and uses single interest rate whose value as shown by standard Monte-Carlo simulation is 0.0339(0.0004). The second one in Table 3.15 starts with a higher initial level $\mathbf{S_0} = 115$ and again uses single interest rate whose value is -1.711(0.002). The purpose of these two experiments is to show the accuracy and robustness of our stochastic mesh method. Note that for $\mathbf{S_0} = 115$ the price becomes negative. We are buying risk instead of selling risk. The option seller needs to pay the option buyer in order to sell this product. The reason comes as followed. This call combination option consists of a long position in a call struck at 95 and two short position in a call struck at 105. The hedging cost is positive for the first option while negative for the second option. Negative hedging cost actually means hedging benefit. When we start at 115, both options is very likely to end in the money. The initial value is too large that the hedging cost of the second option outperforms that of first option. So the total hedging cost is negative. Negative hedging cost implies negative price. The money lent is more than the money borrowed in total. Therefore lending rate should become important. Next test shows how important it is.

The third one in Table 3.16 starts with the same initial level as the second one $\mathbf{S_0} = 115$ while using different interest rates. The different simulation result from Table 3.15 shows that the lending rate r really plays an important role in the hedging process when $\mathbf{S_0}$ is significantly higher than the strikes. In this test a lower lending interest rate $r = 0.01$ implies less hedging benefit from the second option resulting in a cheaper price (in absolute value). As shown in Table 3.15 and Table 3.16, 0.273 is cheaper than 1.708 when both b and M equal 512 which is taken as the most accurate estimations.

M	b=32	b=64	b=128	b=256	b=512
32	0.022(0.007)	0.026(0.004)	0.032(0.003)	0.033(0.002)	0.034(0.002)
64	0.036(0.005)	0.026(0.003)	0.035(0.003)	0.034(0.002)	0.034(0.001)
128	0.031(0.003)	0.036(0.003)	0.03(0.001)	0.033(0.001)	0.033(<0.001)
256	0.031(0.002)	0.030(0.001)	0.033(0.001)	0.031(<0.001)	0.031(<0.001)
512	0.032(0.001)	0.032(0.001)	0.032(0.001)	0.032(<0.001)	0.033(<0.001)

Table 3.14: 20D averaged call combination with single interest rate, $\mathbf{S_0} = 90$

M	b=32	b=64	b=128	b=256	b=512
32	-1.610(0.074)	-1.682(0.051)	-1.757(0.040)	-1.695(0.032)	-1.744(0.021)
64	-1.661(0.057)	-1.733(0.035)	-1.728(0.033)	-1.734(0.018)	-1.707(0.015)
128	-1.683(0.037)	-1.721(0.029)	-1.723(0.019)	-1.699(0.013)	-1.712(0.008)
256	-1.697(0.029)	-1.714(0.019)	-1.716(0.013)	-1.703(0.009)	-1.704(0.006)
512	-1.711(0.020)	-1.728(0.014)	-1.711(0.009)	-1.706(0.007)	-1.708(0.005)

Table 3.15: 20D averaged call combination with single interest rate,$\mathbf{S_0} = 115$

M	b=32	b=64	b=128	b=256	b=512
32	-0.313(0.114)	-0.231(0.062)	-0.338(0.044)	-0.235(0.038)	-0.316(0.019)
64	-0.24(0.069)	-0.363(0.037)	-0.319(0.033)	-0.301(0.023)	-0.266(0.018)
128	-0.325(0.043)	-0.329(0.034)	-0.303(0.024)	-0.282(0.015)	-0.287(0.01)
256	-0.354(0.035)	-0.317(0.021)	-0.291(0.016)	-0.281(0.011)	-0.274(0.008)
512	-0.329(0.024)	-0.334(0.017)	-0.289(0.011)	-0.277(0.008)	-0.273(0.005)

Table 3.16: 20D averaged call combination with different interest rate,$\mathbf{S_0} = 115$

high-dimensional geometric average call combination option

After test for arithmetic average call combination in above simulations, the natural next step would be what if we are using geometric average instead. In this test, the payoff function becomes $\Phi(\mathbf{S}) = (\bar{S}_T - K_1)_+ - 2(\bar{S}_T - K_2)_+$ where $\bar{S}_T = (\prod_{d=1}^{d=20} S_T^d)^{\frac{1}{20}}$. Parameter setting is the same as Table 3.11. Simulation result is in Table 3.17. If geometric Brownian Motion is used for underlying asset, then the geometric average is again a geometric Brownian Motion. Explicit solution could be found for related call prices. In this test, the true value is 5.864. Again the stochastic mesh method did a very good job.

We would like to run the same test again but for different interest rates. In the above arithmetic average experiment, the two tests give the same simulation result implying that the lending interest rate is never used. In this geometric average experiment, we expect to see the same effect since geometric average is always smaller than arithmetic average. The first option struck at 95 is more likely to end in the money than the second option struck at 115. Our intuition is proved in Table 3.18.

M	b=32	b=64	b=128	b=256	b=512
32	5.799(0.068)	5.861(0.04)	5.908(0.032)	5.849(0.024)	5.892(0.017)
64	5.81(0.046)	5.901(0.029)	5.881(0.023)	5.882(0.014)	5.859(0.012)
128	5.853(0.03)	5.871(0.024)	5.881(0.015)	5.853(0.01)	5.866(0.006)
256	5.865(0.024)	5.877(0.015)	5.872(0.011)	5.861(0.008)	5.864(0.005)
512	5.875(0.016)	5.885(0.011)	5.867(0.007)	5.863(0.005)	5.863(0.004)

Table 3.17: 20D geometric averaged call combination with single interest rate

M	b=32	b=64	b=128	b=256	b=512
32	5.799(0.068)	5.861(0.04)	5.908(0.032)	5.849(0.024)	5.892(0.017)
64	5.81(0.046)	5.901(0.029)	5.881(0.023)	5.882(0.014)	5.859(0.012)
128	5.853(0.03)	5.871(0.024)	5.881(0.015)	5.853(0.01)	5.866(0.006)
256	5.865(0.024)	5.877(0.015)	5.872(0.011)	5.861(0.008)	5.864(0.005)
512	5.875(0.016)	5.885(0.011)	5.867(0.007)	5.863(0.005)	5.863(0.004)

Table 3.18: 20D geometric averaged call combination with different interest rates

high-dimensional max call combination option

We have seen the lending interest rate is never used either in arithmetic average or in geometric average as long as initial level start at $\mathbf{S_0} = 100$. We would like to find an example where the lending interest rate indeed makes a difference. Following this idea, next experiment comes. In this test max function is used to evaluate the call combination option. Therefore the payoff function becomes $\Phi(\mathbf{S}) = (\hat{S}_T - K_1)_+ -$

$2(\hat{S}_T - K_2)_+$ where $\hat{S}_T = \max_{1 \le d \le 20} S_T^d$. The maximum asset out of 20 makes the second option highly probable to end deep in the money. The money borrowed must largely exceed the money lent during the hedging process. We expect to see two effects. First effect is the negative price which is proved in Table 3.19 where single interest rate is used. Second effect is that the lending interest rate significantly impacts the price which is proved in Table 3.19 where different interest rate is used. For the second test $r = 0.01$ is used again and similar argument again leads to a cheaper price (in absolute value). The standard Monte-Carlo simulation for single rate test ($R = r = 0.06$) suggests the "true" value should be -6.810(0.007). As a by-product, standard Monte-Carlo simulation for $R = r = 0.01$ gives a prices of -5.384(0.007). Our best estimate for different rates test ($R = 0.06, r = 0.01$) is -5.370(0.014) which suggests the borrowing rate is never used. This is reasonable due to the number of assets and that they are all uncorrelated. 20 uncorrelated assets ends below 115 (critical point where the payoff becomes negative) is really a rare event.

M	b=32	b=64	b=128	b=256	b=512
32	-6.728(0.208)	-6.743(0.153)	-6.773(0.092)	-6.793(0.081)	-6.795(0.041)
64	-6.720(0.126)	-6.804(0.094)	-6.789(0.077)	-6.795(0.044)	-6.791(0.038)
128	-6.716(0.098)	-6.766(0.063)	-6.775(0.051)	-6.794(0.036)	-6.79(0.023)
256	-6.720(0.069)	-6.758(0.048)	-6.799(0.036)	-6.789(0.025)	-6.802(0.017)
512	-6.750(0.050)	-6.760(0.037)	-6.782(0.024)	-6.792(0.016)	-6.799(0.012)

Table 3.19: 20D max call combination with single interest rate

2-dimensional exchange option

In this experiment we try to see an interesting example where only 2 assets are used and the payoff is $\Phi(\mathbf{S}) = (S_T^2 - S_T^1)_+$. This is called the exchange option. Analytical solution [14] exists to compare. The interesting part is the hedging process. Along

M	b=32	b=64	b=128	b=256	b=512
32	-5.374(0.269)	-5.312(0.158)	-5.479(0.122)	-5.317(0.106)	-5.416(0.044)
64	-5.433(0.162)	-5.522(0.107)	-5.371(0.091)	-5.393(0.050)	-5.378(0.046)
128	-5.446(0.111)	-5.415(0.075)	-5.398(0.06)	-5.386(0.046)	-5.372(0.027)
256	-5.493(0.082)	-5.406(0.056)	-5.383(0.041)	-5.399(0.028)	-5.372(0.021)
512	-5.422(0.057)	-5.443(0.042)	-5.399(0.028)	-5.375(0.019)	-5.370(0.014)

Table 3.20: 20D max call combination with different interest rates

the hedging process the option seller needs to borrow money and buy the first underlying asset while lending money and short selling the second underlying asset. Theoretically we expect both borrowing and lending rate are equally important. Parameter setting is the same as Table 3.11. Similar procedure is followed. We first run the test with single interest rate in Table 3.21 and then with different interest rates in Table 3.22.

Analytical solution for exchange option is

$$V_t = S_t^2 N(d_1) - S_t^1 N(d_2) \tag{3.18}$$

where

$$\begin{aligned} d_1 =& \frac{\ln \frac{S_t^2}{S_t^1} + \frac{1}{2}\sigma^2(T-t)}{\sigma\sqrt{T-t}} \\ d_2 =& d_1 - \sigma\sqrt{T-t} \\ \sigma^2 =& (\sigma_t^1)^2 - 2\rho\sigma_t^1\sigma_t^2 + (\sigma_t^1)^2 \end{aligned} \tag{3.19}$$

Thus the true price is 5.630. Our best estimate in Table 3.21 corresponding to single rate is 5.634(0.013). These two prices match well. Our best estimate in Table 3.22 corresponding to different rates is 5.684(0.013) which is slightly higher than 5.630. Even though the difference is small, it can not be ignored nor be treated as instability problem. The difference comes from the lower lending rate. Using similar arguments as before, option seller lends money to short sell the second underlying asset. The interest earned from lending rate is regarded as a hedging benefit or negative hedging cost. A lower hedging benefit requires the seller to charge more from the buyer resulting in a higher price.

M	b=32	b=64	b=128	b=256	b=512
32	5.231(0.193)	5.529(0.144)	5.536(0.109)	5.675(0.092)	5.638(0.059)
64	5.596(0.162)	5.631(0.116)	5.578(0.089)	5.588(0.061)	5.659(0.034)
128	5.433(0.103)	5.599(0.073)	5.617(0.054)	5.634(0.039)	5.648(0.025)
256	5.51(0.076)	5.599(0.05)	5.661(0.037)	5.626(0.025)	5.627(0.017)
512	5.544(0.053)	5.577(0.037)	5.644(0.026)	5.641(0.018)	5.634(0.013)

Table 3.21: 2D exchange option with single interest rate

M	b=32	b=64	b=128	b=256	b=512
32	5.382(0.194)	5.626(0.148)	5.623(0.118)	5.73(0.092)	5.682(0.061)
64	5.721(0.169)	5.712(0.118)	5.66(0.09)	5.646(0.063)	5.708(0.035)
128	5.545(0.107)	5.677(0.075)	5.688(0.056)	5.692(0.041)	5.694(0.026)
256	5.632(0.079)	5.69(0.052)	5.734(0.038)	5.686(0.026)	5.679(0.018)
512	5.668(0.055)	5.66(0.038)	5.719(0.027)	5.697(0.019)	5.684(0.013)

Table 3.22: 2D exchange option with different interest rates

3.2 Pricing of American style option

In this section we presents one simulation results on American style option using RBSDE. In terms of numerical scheme, the only difference we need to make is to replace the estimated option value at each time step $\hat{Y}_{t_i}$ by

$$\hat{Y}_{t_i} = \max\{\mathbb{E}[\hat{Y}_{t_{i+1}} + f(t_i, \hat{X}_{t_i}, \hat{Y}_{t_{i+1}}, \hat{Z}_{t_i})\Delta t_i|\mathcal{F}_{t_i}], \Phi(\mathbf{S_{t_i}}\} \tag{3.20}$$

Unlike the European style option where the time step $N = 1$ is required to minimize the error, more exercise opportunities should be allowed. In this test we fix $N = 5$ and the payoff $\Phi(\mathbf{S}) = (\bar{S}_T - K)_+$ where $\bar{S}_T = (\prod_{d=1}^{d=5} S_T^d)^{\frac{1}{5}}$. This is known as Bermudan option with geometric average call where the five underlying assets are independently modeled as geometric Brownian Motion. Although American option requires the possibility to exercise at any time prior to maturity, in reality Bermudan option is often applied to approximate American option. Parameter setting is in Table 3.23 where d stands for continuous dividend rate. As shown by [2], this pricing problem can be reduced to a single-asset American option which can be solved accurately using a one-dimensional binomial tree. Therefore the reference price is 10.211 using binomial tree. Our simulation result could be found in Table 3.24. To see the convergence, we refer to Figure 3.3 where the estimated value is plotted against the number of Path b fixing $M = 256$.

Since the max function is involved at each time step, by Jensen's Inequality the convexity of max function will make the price high biased. This bias is reduced when b is increased which is reflected in Table 3.24. Increasing the number of simulations M does not help in reducing the bias. We have got some instability problems when $b = 64$ and $M = 512$. However this is not related to the algorithm itself. It is due to the outer control variate we have used. Recall from (3.13) that we have used max call option as outer control variate in order to reduce variance. The estimates for this option may be zero. And this event is not a rare event provided that only 5 assets are used and dividend d is higher than drift μ.

$\boldsymbol{\mu}$	$\boldsymbol{\sigma}$	$\boldsymbol{r}$	$\boldsymbol{R}$	$\boldsymbol{T}$	$\boldsymbol{S_0}$	$\boldsymbol{K}$	$\boldsymbol{d}$
0.03	0.4 Id	0.03	0.03	1	110	100	0.05

Table 3.23: Parameter settings for American averaged call option with $\Phi(\mathbf{S}) = (\bar{S}_T - K)_+$

M	b=32	b=64	b=128	b=256	b=512
32	13.189(0.308)	12.51(0.208)	11.314(0.125)	10.77(0.074)	10.401(0.041)
64	13.486(0.271)	12.064(0.142)	11.315(0.067)	10.654(0.05)	10.398(0.036)
128	13.347(0.178)	12.321(0.102)	11.306(0.055)	10.701(0.034)	10.387(0.023)
256	13.393(0.13)	12.182(0.072)	11.276(0.042)	10.739(0.025)	10.39(0.016)
512	13.49(0.091)	N/A	11.315(0.03)	10.74(0.017)	10.395(0.011)

Table 3.24: American averaged call option

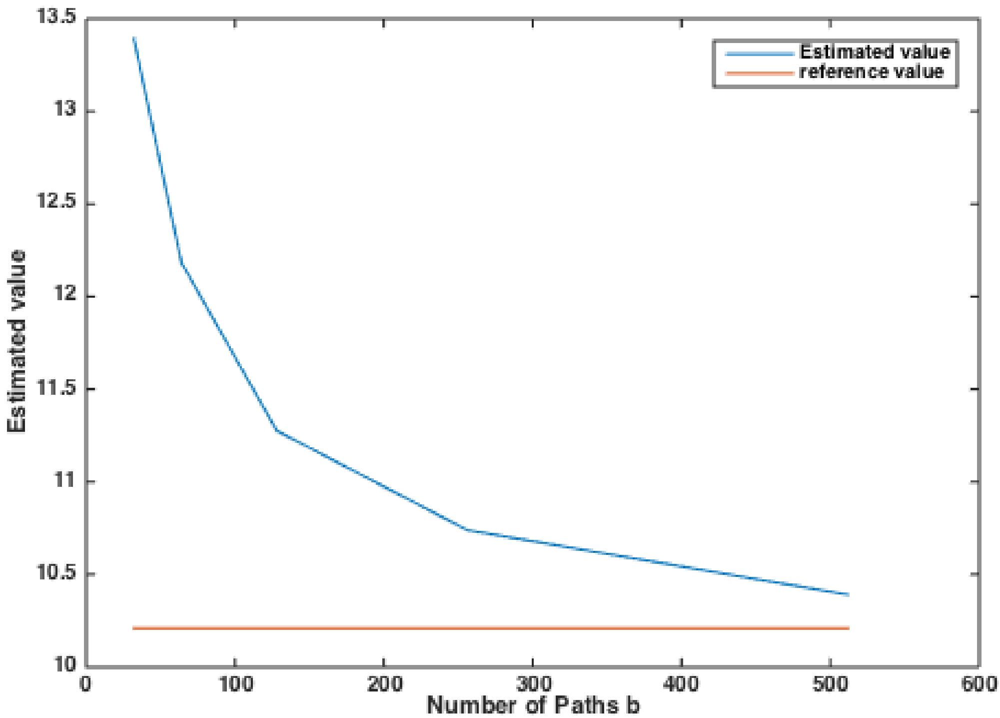

Figure 3.3: American averaged call option, M=256

Chapter 4

Conclusion

In this thesis, we review the numerical method for solving a decoupled BSDE problem. Inspired by [2] we employ a new technique called stochastic mesh method in BSDE and focus mainly on high-dimensional problems. A mesh is constructed by forward process and all nodes at consecutive time steps are interconnected. Each arc is assigned a weight function. Based on these weight function and backward induction, we compute Z and Y at each node. The weight function plays a fundamental role in this numerical scheme. In this thesis we use average density function as weight function and test against both one-dimensional and high-dimensional BSDE problems. Empirical results show that increasing b and M both help reduce standard deviation in the same order. Increase N does not have any obvious advantage under Black-Schole model provided that European option is evaluated since there exists a closed formula to compute forward process. Overall higher b usually results in a more accurate estimation.

Throughout our experiments, this numerical scheme works very well even in high-dimensional case. The only robustness problem is found when noise process is involved. With dimensionality higher than 20 we are confident this scheme still works well.

Choosing average density function as weight function has a disadvantage. That is the existence of transition density. In the absence of transition density such as Asia option or look back option where another state variable should be included in each node one should refer to other weight function such as those in [3]. We leave this as future research.

Even though we have tested only one example of RBSDE, it is enough to show the accuracy of this method. The only limitation would be the computation effort which is far from industry application. One thing to remind is that stochastic mesh method is originally designed to solve American option pricing problem. So one advantage of reusing this technique in RBSDE background is the variation of drivers such as the driver for bid-ask spread in interest rate.

The main contribution of this thesis is the attempt to apply stochastic mesh method in solving various high-dimensional BSDEs. Though further work needs to be done the perspective is very promising.

Bibliography

[1] V. Bally and D. Talay, *The law of the euler scheme for stochastic differential equations*, Probability Theory and Related Fields, 104 (1996), pp. 43–60.

[2] M. Boradie and P. Glasserman, *A stochastic mesh method for pricing high-dimensional american options*, The Journal of Computational Finance, 7 (2004).

[3] M. Boradie, P. Glasserman, and Z. Ha, *Pricing american options by simulation using a stochastic mesh with optimized weights*, Probabilistic Constrained Optimization: Methodology and Applications, (2000), pp. 32–50.

[4] B. Bouchard and N. Touzi, *Discrete time approximation and monte carlo simulation of backward stochastic differential equations*, Stochastic Processes and their Applications, 111 (2004), pp. 176–206.

[5] J. Cvitanic and J. Ma, *Hedging options for a large investor and forward-backward sde's*, Annals of Applied Probability, 6 (1996), pp. 370–398.

[6] N. El Karoui, S. Peng, and M. C. Quenez, *Backward stochastic differential equations in finance*, Mathematical Finance, 7 (1997), pp. 1–71.

[7] E.Pardoux and S. Peng, *Adapted solution of a backward stochastic differential equations*, Systems and Control Letters, 14 (1990), pp. 55–61.

[8] E. Gobet and C. Labart, *Error expansion for the discretization of backward stochastic differential equations*, Stochastic Processes and their Applications, 117 (2007), pp. 803–829.

[9] E. Gobet, J.-P. Lemor, and X. Warin, *A regression-based monte carlo method to solve backward stochastic differential equations*, The Annals of Applied Probability, 15 (2005), pp. 2172–2202.

[10] J.Zhang, *A numerical scheme for backward stochastic differential equations*, Annals of Applied Probability, 14 (2004), pp. 459–488.

[11] N. E. Karoui, C. Kapoudjian, E. Pardoux, S. Peng, and M. C. Quenez, *Reflected solutions of backward sde'e, and related obstacle problems for pde's*, The Annals of Probability, 25 (1997), pp. 702–737.

[12] J. Ma, P. Protter, and J. Yong, *Solving forward-backward stochastic differential equations explicitely - a four-step scheme*, Probability Theory and Related Fields, 98 (1994), pp. 339–359.

[13] J. Ma and J. Zhang, *Path regularity for solutions of backward stochastic differential equations*, Probability Theory and Related Fields, 122 (2002), pp. 163–190.

[14] W. Margrabe, *The value of an option to exchange one asset for another*, Journal of Finance, 33 (1978), pp. 177–186.

[15] B. Moro, *The full monte*, Risk, 8 (1995), pp. 53–57.

[16] S. K. Park and K. W. Miller, *Random number generators:good ones are hard to find*, Communication of the ACM, 31 (1988), pp. 1192–1201.