University of Oxford

Valuation of American and other path-dependent options via Monte Carlo methods

Candidate Number:
213032

Course:
MMath Mathematics

CCD: Dissertations on a Mathematical Topic

Hilary Term
2016

Contents

1 Introduction

Options are a commonly traded and sought after commodity in the financial sector and, as a consequence, their pricing is of critical importance and has come under significant mathematical attention. While under no-arbitrage assumptions European options can be modelled via Black-Scholes and an explicit solution found, no such solution exists for the American counterpart. This is due to its early exercise property which means that, while the holder of a European option can only exercise the option at its maturity, an American option can be exercised by the holder at any time prior to maturity. This path dependence leads to extreme difficulty in pricing and the approximation and bounding of the price has been vastly researched [11].

The arbitrage-free price of an American option is the discounted value achieved at the optimal exercise time, transposing the problem to an optimal stopping one. Varying techniques have been attempted to value the American option. Monte Carlo methods do not suffer the curse of dimensionality and so recently have carried more favour [11]. Two main Monte Carlo methods have been used. The first uses a backwards recursion to solve the optimal stopping problem, providing a negatively biased approximation. The calculation of nested conditional expectations becomes the main difficulty and numerous techniques have been suggested with Longstaff and Schwartz's method of using least-squares regression one of the most popular [40]. The second method considers the dual of the problem, giving a positively biased approximation, and which was developed independently by Rogers [46], and Haugh and Kogan [27]. Andersen and Broadie combined the two approaches and formulated an algorithmic hybrid approach which utilised the primal simulation values to compute the dual [2].

Coined by Yuri Kifer in his 1998 paper [33], 'Game Options' (also known as 'Israeli options') provide an extension to American options in a similar way to the extension of European options by American options. In addition to the holder of the option possessing an early exercise premium, the writer has the ability to terminate the contract at any time up to maturity at a penalty. Kifer claims that while these commodities are not commonly traded they are important in the sense that existing traded contracts such as callable and convertible bonds allow the writer a cancellation feature. Motivated by the dual formulation of the American option we show that corresponding dual results can be formulated in several ways for the game option, each suggesting a different means of computing the option price.

The computation of the game option has not received nearly as much attention as its American counterpart. This is addressed here by adapting methodologies proposed for the American option to the game option case. An extension of regression methods for American options will value the game option. In the American option this provided a negatively biased approximation however, due to the nature of its formulation, the approximation produced for the game option will be neither positively nor negatively biased. We propose a new method which extends the hybrid methods for American options. Theoretically our method produces upper and lower bounds for the value of the game option, something that has not been addressed in the literature. In the case of continuous-time, due to discretisation errors which introduce a positive bias, true lower bounds require a large number of time steps. This is not the case when exercise is only permitted at discrete time intervals, as in Bermudan options, where our method should produce true lower and upper bounds.

The structure of the dissertation is as follows. Chapter 2 introduces basic theoretical results used in mathematical finance and needed in the valuation of options, in particular American options. Chapter 3 concerns itself with the pricing of American options. We provide theoretical results and a discussion on previously proposed methods. This is extended to the case of Game Options in Chapter 4. Here, we emulate similar dual results for the value of the option and propose algorithms to compute the value of the option based on these results, giving numerical results and comparing algorithm efficiency. The dissertation is concluded with a discussion on the success of proposed algorithms and detail on how this research could be continued.

2 Background Mathematics

Consider a filtered probability space $(\Omega, \mathscr{F}, (\mathscr{F}_t)_{t\geq 0}, \mathbb{P})$ where Ω is the sample space, $\mathscr{F}$ is a σ-algebra, $(\mathscr{F}_t))_{t\geq 0}$ is a filtration on the probability space and $\mathbb{P}$ is a probability measure.

Definition 2.1. A *stopping time* τ with respect to $(\mathscr{F}_t)_{t\geq 0}$ is a random variable, $\tau(\cdot) : \Omega \to [0, T]$ such that $\{\tau \leq t\} \in \mathscr{F}_t$ for all $t \in [0, T]$.

Definition 2.2. A stochastic process $(M_t)_{t\geq 0}$ is a *martingale* with respect to the filtration $(\mathscr{F}_t)_{t\geq 0}$ if,

(i) (adapted) for each $t \geq 0$, M_t is $\mathscr{F}_t$-measurable,

(ii) (integrable) for each $t \geq 0$, $\mathbb{E}[|M_t|] < \infty$, and

(iii) (martingale property) for all $0 \leq s < t$, $\mathbb{E}[M_t \mid \mathscr{F}_s] = M_s$.

Similarly we define a super-martingle with altered final condition,

(iii) (super-martingale property) for all $0 \leq s < t$, $\mathbb{E}[M_t \mid \mathscr{F}_s] \leq M_s$,

and for a sub-martingale,

(iii) (sub-martingale property) for all $0 \leq s < t$, $\mathbb{E}[M_t \mid \mathscr{F}_s] \geq M_s$.

Definition 2.3. A stochastic process $(M_t)_{t\geq 0}$ is a *local martingale* with respect to the filtration $(\mathscr{F}_t)_{t\geq 0}$ if there exists a sequence of almost surely increasing stopping times $(\tau_k)_{k\geq 0}$ which diverge almost surely, and such that the stopped process $M_t^{\tau_k}$ is an $\mathscr{F}_t$-martingale for every k.

We can consider the natural filtration $(\mathscr{F}_t^X)_{t\geq 0}$ with respect to a stochastic process $(X_t)_{t\geq 0}$, $\mathscr{F}_t^X = \sigma(X_s \mid s \leq t)$. A martingale is *càdlàg (or RCLL)* if it is right-continuous with left limits. A filtration $(\mathscr{F}_t))_{t\geq 0}$ satisfies the usual conditions if it is right-continuous and contains all null events. We can always consider an augmentation of a filtration so that it satisfies the usual conditions and it is known that any martingale has a càdlàg modification [43].

Definition 2.4. A stochastic process $(X_t)_{t\geq 0}$ is *uniformly integrable* if

$$\sup_{t\geq 0} \mathbb{E}\left[|X_t| \mathbb{1}_{|X_t|\geq K}\right] \to 0 \quad \text{as } K \to \infty.$$

Definition 2.5. A measurable stochastic process $(X_t)_{t\geq 0}$ is of *class D* if $X_0 = 0$ and the collection $\{X_\tau \mid \tau \text{ a finite-valued stopping time}\}$ is uniformly integrable.

Definition 2.6. A stochastic process $(A_t)_{t\geq 0}$ is *predictable* if it is measurable with respect to the σ-algebra generated by all left-continuous adapted processes.

Doob-Meyer decompositions allow us to break down a super/sub-martingale into a martingale part and a predictable part. A proof of the following theorem can be found in [18].

Theorem 2.7 (Doob-Meyer decomposition theorem). *Consider a càdlàg super-martingale X of class D. Then there exists a unique, integrable, increasing, predictable process A, and unique, uniformly integrable martingale M with $M_0 = A_0 = 0$ such that*

$$X_t = X_0 + M_t - A_t.$$

A fundamental and important example of a martingale is Brownian motion.

Definition 2.8. A stochastic process $(W_t)_{t\geq 0}$ is a standard *Brownian motion* (BM) if,

(i) $W_0 = 0$ almost surely,

(ii) W_t is almost surely continuous,

(iii) (stationary and independent increments) for any $0 \leq s \leq t$, $W_t - W_s \sim \mathcal{N}(0, t-s)$ and is independent of $\mathscr{F}_t^W$.

It was originally proposed that the value of a stock could be modelled as a Brownian motion. This was consequently updated to the more appropriate geometric Brownian Motion, a stochastic process where the logarithm of the process follows a Brownian motion.

Definition 2.9. A stochastic process $(S_t)_{t\geq 0}$ follows a *geometric Brownian motion* (GBM) if

$$S_t = S_0 \exp\left[\left(\mu - \frac{1}{2}\sigma^2\right)t + \sigma W_t\right]$$

where μ and $\sigma > 0$ are constants and $(W_t)_{t\geq 0}$ is a standard BM. We call μ the drift and σ the volatility of the process.

Remark. We note that GBM satisfies the stochastic differential equation (SDE)

$$dS_t = \mu S_t \, dt + \sigma S_t \, dW_t$$

which helps explain the naming of the constants μ and σ.

We consider an asset whose price follows a stochastic process $(S_t)_{t\geq 0}$ and, in general, we will assume that this is a GBM. One should ask whether this is a reasonable assumption to make. Indeed certain properties make it an attractive model: it only takes positive values as a stock would, its returns are independent of the value of the process and it appears similar to a stock with comparable jaggedness. Combining this with the relative ease of calculations ensures that is a widely used assumption, however it should be noted that it is not a perfect model. Faults include that the volatility is assumed to be constant whereas in practice this could be dependent on the price of the asset. Additionally GBM is a continuous model and hence discounts the practice of jumps, which stock prices do exhibit. The study of stocks that follow jump processes is a prosperous area of research [11].

Consider a risk-free asset $(S_t)_{t\geq 0}$ that increases over time with interest rate process $(r_t)_{t\geq 0}$. An amount invested will grow with this interest rate, so if an amount C where invested initially then at time t the value of this will grow continuously to $Ce^{\int_0^t r_t dt}$. Equally, if we are given a final amount C at time t then at time zero the *discounted value* of this was $Ce^{-\int_0^t r_t dt}$. We call the factor $b_t = e^{-\int_0^t r_t dt}$ the *discount factor*, defined for $0 \leq t \leq T$. We also use the notation, $b_{s,t} = e^{-\int_s^t r_t dt}$. In this dissertation we focus only on a constant rate of interest, $r_t = r$, which simplifies to give $b_t = e^{-rt}$ and $b_{s,t} = e^{-r(t-s)}$.

Definition 2.10. An option is a contract which gives the holder the right, but not the obligation, to buy or sell an underlying asset at a price specified on the signing of the contract, the strike price K. An option has a payoff function $h : \mathbb{R}_+ \to \mathbb{R}_+$, which pays at time $t \in [0, T]$ the intrinsic value h_t (or $h(S_t)$) to the holder of the option, where T is the time of maturity.

There exist numerous types of options with varying payoff functions and features. The two most basic and commonly traded options are European and American. In European options the holder may only exercise the option at maturity whereas in American options they may be exercised at any time up to and including maturity. These can be traded as either *put* or *call* options. For a *put option* the holder of the option has the right to sell the asset at the strike price whereas in a *call option* this right is to buy the asset at the strike price. The payoff functions for these simple vanilla options are therefore given by

$$\textbf{Call} \qquad h_t = (S_t - K)^+ \tag{2.1}$$

$$\textbf{Put} \qquad h_t = (K - S_t)^+, \tag{2.2}$$

where $x^+ = \max(x, 0)$. Finally, we will call a stock *in the money* if its payoff function is greater than zero and *out of the money* otherwise.

3 American Options

Numerical approaches to the valuation problem of American options are a thoroughly researched area and still provide a challenge today [11]. The majority of approaches fall into the following three categories. Formulas and approximations, which comprise transform methods [28, 48] and asymptotic expansions techniques [21, 25, 29]. Lattice and finite difference methods which use discrete-state approximations to SDEs to compute the option prices, first proposed by Parkinson [41] and Cox et. al. [16]. These also include attempts to solve the problem as a partial differential equation with free boundary conditions applied [8, 9, 42].

Finally, there are Monte Carlo simulation methods. Whereas computation rates of the above approaches greaten as the dimension of the problem increases, by the Central Limit Theorem the convergence rate of Monte Carlo method is $\mathcal{O}(1/\sqrt{n})$ independent of the dimension. This property makes Monte Carlo an attractive approach when the options are over multiple assets and similarly when the pricing of the option is complex and path-dependent. Monte Carlo methods are numerous in their ways [7] and the problem can be approached from two directions. Solving the primal problem by approximating an optimal stopping time gives a lower bound to the value of the option. Alternatively, using the dual formulation, construction of a martingale provides an upper bound.

In general finding a stopping strategy seems to be a much easier task than finding a martingale. To compute an optimal exercise policy a backwards dynamic programming principle is applied. Under this principle an approximation can be computed through various means. Andersen utilises a functional optimisation approach [3] while Broadie and Glasserman propose a method based on simulated trees [12].

Broadie and Glasserman also propose a stochastic mesh method which, unlike the above methods, generates an error bound and was shown to converge to the option value [13]. A popular approach is through simple regression, which can be viewed as a special case of the stochastic mesh method [23]. This approach has been taken by Longstaff and Schwartz [40], Carriere [14], and Tsitsiklis and Van Roy [50, 51] as well as others. Tsitsiklis and Roy provide theoretical proof of convergence for their algorithm whilst Clement, Lamberton and Protter provide this for the algorithm proposed by Longstaff and Schwartz [15]. Glasserman and Yu investigate the convergence when both the number of basis functions and sample paths increase to infinity [24].

Instead of linear regression, non-parametric regression could be used as in [14]. Todorovic showed that non-parametric regression can lead to better estimates if the payoff function is not chosen as a basis function [49]. The choice of basis functions can be extremely sensitive. Recently the idea of policy iteration has been introduced by Kolodko and Schoenmakers to help remove this sensitivity [34]. The efficiency of the algorithm was improved upon by Broadie and Cao [10] with Beveridge and Joshi providing further improvements [6].

Using the dual approach we can attempt to build a martingale from scratch and therefore build a method that is, in some sense, a pure dual solution. A pure solution to the dual problem is sought by Rogers who considers a linear combination of martingales that bear some resemblance to the value function [46] . More recently he has proposed a more algorithmic approach to the pure solution based on principles similar to that of backwards dynamic programming [44, 45].

The majority of dual approaches utilise a primal approximation to construct a martingale, giving a hybrid algorithm. Haugh and Kogan propose such a hybrid method, using a primal approximation for the value of the option to generate the martingale [27]. Andersen and Broadie apply a variation of this technique producing a more computationally effective method and utilising nested simulations [2]. Belomestny, Bender and Schoenmakers propose a method avoiding the use of nested Monte Carlo simulations [4]. Recently the use of Multilevel Monte Carlo has been investigated by Belomestny, Schoenmakers and Dickmann [5] after recent proposals by Mike Giles [22]. An advantage of these hybrid approaches is that since both upper and lower bounds are produced an approximate confidence interval can be formed for the value of the option.

In the following sections we provide theoretical valuation of an American option, introducing both the primal and dual problems. We describe and review the algorithmic methods for the primal and dual problems proposed by Longstaff and Schwartz, and Andersen and Broadie, respectively. For each we discuss and utilise several time-saving features, variance reduction techniques and compare the possible different implementations. The final section provides numerical results and exhibits the success of the algorithms for an American option, which we will extend to the case of the game option in the following chapter.

3.1 Pricing

We price options through risk-neutral derivative pricing theory assuming a complete market, with all expectations taken with respect to the equivalent local martingale measure. Further details on this theory, including the Fundamental Theorems of Asset Pricing, can be found in [17].

Definition 3.1. Consider an American option with payoff function $(h_t)_{0\leq t\leq T}$. The *arbitrage-free* price for such an option is given by

$$V_t = \sup_{\tau\in\mathscr{T}(t,T)} \mathbb{E}\left[b_{t,\tau}h_\tau \mid \mathscr{F}_t\right], \quad t\in[0,T] \tag{3.1}$$

where $\mathscr{T}(t,T)$ denotes the set of stopping times taking values in $[t,T]$.

This definition is intuitive. The price of the option is the highest value that can be achieved through exercise discounted back to the present value. This is the primal problem and gives the time zero price for the American option as

Primal
$$V_0 = \sup_{\tau\in\mathscr{T}(0,T)} \mathbb{E}\left[b_\tau h_\tau\right]. \tag{3.2}$$

Using any τ generates a lower bound,

$$V_0 \geq \mathbb{E}\left[Z_\tau\right].$$

3.1.1 Dual Formulation

To derive the dual formulation we define a useful process called the Snell envelope.

Definition 3.2. The *Snell envelope* of the discounted payoff $b_t h_t$ is the process

$$Z_t := b_t V_t = \operatorname*{ess\,sup}_{\tau \in \mathscr{T}(t,T)} \mathbb{E}\left[b_\tau h_\tau \mid \mathscr{F}_t\right]. \tag{3.3}$$

Definition 3.3. A family $\{X_i : i \in I\}$ is said to have the *lattice property* if for all $i, j \in I$, there exists $k \in I$ such that $X_k \geq X_i \vee X_j$ almost surely.

Lemma 3.4. *Let $\{X_i : i \in I\}$ be a family of non-negative random variables with the lattice property and $\mathscr{G}$ any sub-σ-algebra. Then*

$$\mathbb{E}[\operatorname*{ess\,sup}_{i \in I} X_i \mid \mathscr{G}] = \operatorname*{ess\,sup}_{i \in I} \mathbb{E}[X_i \mid \mathscr{G}].$$

Proof. By the lattice property we can define a sequence $(i_n)_{n \in \mathbb{N}}$ such that $(X_{i_n})_{n \in \mathbb{N}}$ is a.s non-decreasing and $\operatorname{ess\,sup}_{i \in I} X_i = \sup_{n \in \mathbb{N}} X_{i_n} = \lim_{n \to \infty} X_{i_n}$ a.s. The result then follows through monotone convergence.

□

Remark. We can replace the essential supremum with an essential infimum, which will be required later when valuing game options.

Lemma 3.5. *For any $t \geq 0$, the family $\{\mathbb{E}[b_\tau h_\tau \mid \mathscr{F}_t] : \tau \in \mathscr{T}(t,T)\}$ has the lattice property.*

Proof. Fix $t \geq 0$ and let $\tau, \sigma \in \mathscr{T}(t,T)$. Define

$$X_\tau = \mathbb{E}[b_\tau h_\tau \mid \mathscr{F}_t]$$
$$X_\sigma = \mathbb{E}[b_\sigma h_\sigma \mid \mathscr{F}_t].$$

Let

$$\rho = \tau \mathbb{1}_{X_\tau \geq X_\sigma} + \sigma \mathbb{1}_{X_\tau < X_\sigma}.$$

Then certainly $\rho \in \mathscr{T}(t,T)$ and $\mathbb{E}[b_\rho h_\rho \mid \mathscr{F}_t] \geq \mathbb{E}[b_\tau h_\tau \mid \mathscr{F}_t] \vee \mathbb{E}[b_\sigma h_\sigma \mid \mathscr{F}_t]$. □

Combining the two previous lemmas for all $t \geq 0$ and sub-σ-algebras $\mathscr{G}$ we have

$$\mathbb{E}[\operatorname*{ess\,sup}_{\tau \in \mathscr{T}(t,T)} \mathbb{E}[b_\tau h_\tau \mid \mathscr{F}_t] \mid \mathscr{G}] = \operatorname*{ess\,sup}_{\tau \in \mathscr{T}(t,T)} \mathbb{E}[\mathbb{E}[b_\tau h_\tau \mid \mathscr{F}_t] \mid \mathscr{G}]. \tag{3.4}$$

Theorem 3.6. *Assume that the discounted price process $b_t h_t$ is right-continuous and that for some $p > 1$, $\sup_{0 \leq t \leq T} |b_t h_t| \in L^p$, that is $\mathbb{E}[\sup_{0 \leq t \leq T} |b_t h_t|^p] < \infty$. Then the Snell envelope process is a càdlàg super-martingale of Class D.*

Proof. We first prove the super-martingale property. Taking expectations under the risk-neutral measure, for $0 \leq s \leq t$

$$\begin{aligned}
\mathbb{E}[Z_t \mid \mathscr{F}_s] &= \mathbb{E}[\operatorname*{ess\,sup}_{\tau \in \mathscr{T}(t,T)} \mathbb{E}[b_\tau h_\tau \mid \mathscr{F}_t] \mid \mathscr{F}_s] \\
&= \operatorname*{ess\,sup}_{\tau \in \mathscr{T}(t,T)} \mathbb{E}[\mathbb{E}[b_\tau h_\tau \mid \mathscr{F}_t] \mid \mathscr{F}_s] \\
&= \operatorname*{ess\,sup}_{\tau \in \mathscr{T}(t,T)} \mathbb{E}[b_\tau h_\tau \mid \mathscr{F}_s] \\
&\leq \operatorname*{ess\,sup}_{\tau \in \mathscr{T}(s,T)} \mathbb{E}[b_\tau h_\tau \mid \mathscr{F}_s] \\
&= Z_s,
\end{aligned}$$

using (3.4) for the second equality, and where the third equality follows from the Tower Law. The inequality follows since $\mathscr{T}(t,T) \subseteq \mathscr{T}(s,T)$. As Z_t has paths that are right-continuous with left limits, it is càdlàg. By our L^p-bounded assumption, the Snell envelope will also be L^p-bounded,

$$\begin{aligned}
\sup_{t \in [0,T]} \mathbb{E}[|Z_t|^p] &= \sup_{0 \leq t \leq T} \mathbb{E}[|\operatorname*{ess\,sup}_{\tau \in \mathscr{T}(t,T)} \mathbb{E}[b_\tau h_\tau \mid \mathscr{F}_t]|^p] \\
&\leq \sup_{0 \leq t \leq T} \mathbb{E}[|\mathbb{E}[\operatorname*{ess\,sup}_{\tau \in \mathscr{T}(t,T)} b_\tau h_\tau \mid \mathscr{F}_t]|^p] \\
&\leq \sup_{0 \leq t \leq T} \mathbb{E}[\mathbb{E}[|\sup_{s \in [t,T]} b_s h_s|^p \mid \mathscr{F}_t]] \\
&= \sup_{0 \leq t \leq T} \mathbb{E}[|\sup_{s \in [t,T]} b_s h_s|^p] \\
&= \mathbb{E}[|\sup_{s \in [0,T]} b_s h_s|^p] \\
&< \infty,
\end{aligned}$$

where the inequality follows by Jensen's inequality and the third equality by the Tower Law. Since $p > 1$ this indicates our Snell envelope will be UI and hence Z is of Class D. $\square$

The Snell envelope is the smallest super-martingale that dominates the discounted payoff $b_t h_t$. By the Doob-Meyer decomposition we know there exists unique $\widetilde{M}$ and $\widetilde{A}$ such that

$$Z_t = b_t V_t = V_0 + \widetilde{M}_t - \widetilde{A}_t \tag{3.5}$$

where $\widetilde{M}$ is a uniformly integrable martingale, $\widetilde{A}$ is an integrable, increasing, predictable process and $\widetilde{M}_0 = \widetilde{A}_0 = 0$.

Theorem 3.7.

Dual
$$V_0 = \inf_{M \in H_0^1} \left(\mathbb{E}[\sup_{t \in [0,T]} (b_t h(t) - M_t)] \right) \tag{3.6}$$

where H_0^1 is the space of L^1-bounded martingales M with $M_0 = 0$. Moreover, this infimum is attained when $M = \widetilde{M}$, where $\widetilde{M}$ is the martingale from the Doob-Meyer decomposition of the Snell envelope process Z.

Proof. Consider $M \in H_0^1$. Then by (3.2)

$$\begin{aligned} V_0 &= \sup_{\tau \in \mathscr{T}(0,T)} \mathbb{E}\left[b_\tau h_\tau + M_\tau - M_\tau\right] \\ &= M_0 + \sup_{\tau \in \mathscr{T}(0,T)} \mathbb{E}\left[b_\tau h_\tau - M_\tau\right] \\ &\leq \mathbb{E}[\sup_{t \in [0,T]} (b_t h_t - M_t)], \end{aligned}$$

where we have used the Optional Stopping Theorem for the second equality. As M was arbitrary,

$$V_0 \leq \inf_{M \in H_0^1} \left(\mathbb{E}[\sup_{t \in [0,T]} (b_t h_t - M_t)] \right).$$

To finish the proof it suffices to show that when the martingale part of the Doob-Meyer decomposition of the Snell envelope process Z_t is used, this in fact gives equality. Letting $M = \widetilde{M}$,

$$\begin{aligned} V_0 &\leq \mathbb{E}[\sup_{t \in [0,T]} \left(b_t h_t - \widetilde{M}_t\right)] \\ &= V_0 + \mathbb{E}[\sup_{t \in [0,T]} \left(b_t \left(h_t - V_t\right) - \widetilde{A}_t\right)] \\ &\leq V_0. \end{aligned}$$

The second inequality follows from $h_t \leq V_t$ and $\widetilde{A} \geq 0$, since $\widetilde{A}$ is an increasing process with $\widetilde{A}_0 = 0$. □

3.2 Algorithmic Methods

3.2.1 Dynamic Programming Principle

Whilst an American option can be exercised at any time before maturity, to computationally model we must discretise so early exercise is available only at certain times $\{0 = t_1 < \cdots < t_d = T\}$. This is known as a *Bermudan* option. As the number of time steps increases the Bermudan option better approximates the American option. To suppress notation we will write i instead of t_i, so that V_{t_i} becomes V_i and similarly for other processes and filtrations. At maturity,

$$V_d = h_d,$$

since the holder exercises at maturity only if in the money. At time $i = 1, \ldots, d-1$, the holder can either exercise or continue. Immediate exercise presents the holder with h_i. The expected value of the option conditional on not exercising at time i is

$$C_i = \mathbb{E}[b_{i,i+1} V_{i+1} \mid \mathscr{F}_i], \tag{3.7}$$

and called the *continuation value.* Intuitively, at each exercise time the holder only exercises if the value of immediate exercise is greater than or equal to the continuation value, if $h_i \geq C_i$. Therefore the value of the option at time i is

$$V_i = \max(h_i, C_i) \tag{3.8}$$

for $i = 1, \ldots, d-1$. Since exercise is not possible at time 0,

$$V_0 = C_0.$$

We obtain the recursion formula

$$\begin{aligned} C_i &= \mathbb{E}[b_{i,i+1} \max(h_{i+1}, C_{i+1}) \mid \mathscr{F}_i] \\ C_d &= 0 \end{aligned} \tag{3.9}$$

for $i = 0, \ldots, d-1$.

The construction of continuation values gives rise to an exercise policy; we exercise at the first time i such that $h_i \geq C_i$, if it occurs. We formalise this as

$$\tau = \min\{i = 1, \ldots, d : h_i \geq C_i\}. \tag{3.10}$$

The exercise value is discounted back to time 0 to give a lower bound for the true value of the option,

$$L_0 = \mathbb{E}[b_\tau h_\tau].$$

Since C is a function of the asset price at time i we can consider the set of prices $\{s \in \mathbb{R}^+ : h_i(s) \geq C_i(s)\}$ where exercise is indicated. We call this the *exercise region.* Its complement in $\mathbb{R}^+$ is called the *continuation region.* The exercise policy defined by (3.10) is the first time the asset price enters the exercise region.

3.2.2 Least-Squares Monte Carlo Method

V_0 is found by using the recursion formula (3.9) to compute the continuation values. This computation requires calculation of a conditional expectation which Longstaff and Schwartz estimate through a least-squares regression [40]. Consider basis functions $\{\phi_i\}_{i\in\mathbb{N}}$. We approximate C_i by the first M basis functions,

$$C_i(s) = \sum_{j=0}^{M-1} \beta_j^i \phi_j(s) \tag{3.11}$$

for $i = 0, \ldots, d-1$.

Working backwards, at each time step the required coefficients are computed by performing a least-squares regression on all paths in the money. It can be shown that this approximated conditional expectation converges in mean square and in probability to the true continuation value as $M \to \infty$ [53]. The value of the option is the continuation value at time zero and taking the average over all sample paths obtains the Monte Carlo estimate.

Of importance is the choice of basis polynomials. Longstaff and Schwartz make use of weighted Laguerre polynomials. Rasmussen proposes basis functions which depend on the current state of the asset, the equivalent European option price and a product of the two [47]. In particular,

$$\begin{aligned}\phi_0(x,t) &= K\\ \phi_1(x,t) &= x\\ \phi_2(x,t) &= \mathrm{BS}(x,t)\\ \phi_3(x,t) &= x\mathrm{BS}(x,t),\end{aligned} \tag{3.12}$$

where $\mathrm{BS}(x,t)$ is the Black-Scholes value of the corresponding European option started from x at time t with maturity at time T.

The initial LSM algorithm proposed by Longstaff and Schwarz is not at optimal efficiency and we make several modifications to decrease computation time. Their algorithm makes use of a continuation matrix which holds the continuation value for each sample path at each time point. This is unneeded and a large use of memory; we replace it with a vector that stores the continuation value for each path and updates backwards in time, similar to the efficiency suggestions proposed in [31].

Using the same set of sample paths to estimate the regression coefficients and estimate the value of the option introduces the risk of possible bias. Thus we modify the Longstaff-Scwartz algorithm (as in [24, 47]) by using an independent set of sample paths to approximate the value of the option once regression coefficients have been found.

To reduce the standard deviation of the Monte Carlo estimator we use two popular techniques, both investigated by Rasmussen in relation to option pricing [47]. The first is the use of antithetic variates, where path estimates are computed by averaging the discounted payoffs from a 'positive' and 'negative' path. The second method is the introduction of a control variate. Further information on variance reduction techniques can be found in [30]. We call the LSM method with variance reduction techniques applied the LSMVR method.

3.2.3 Hybrid Method

The dual approach constructs a martingale which provides a positively biased approximation through (3.6). We call this algorithm since we utilise a primal algorithm, such as the LSM method, to build the dual martingale. Recalling (3.10), we define an exercise indicator process $\{I_i\}_{i=1,\dots,d}$ at each exercise time

$$I_i = \begin{cases} 1 & \text{if } h_i \geq C_i \\ 0 & \text{otherwise.} \end{cases} \tag{3.13}$$

It is given that $I_0 = 0$. At any time $0 \leq i < d$ we consider the next time the option should be exercised

$$\tau_i = \inf\left\{i < k \leq d : I_k = 1\right\}. \tag{3.14}$$

Recall this exercise strategy defines a lower bound price process

$$b_i L_i = \mathbb{E}[b_{\tau_i} h_{\tau_i} \mid \mathscr{F}_i]. \tag{3.15}$$

We define the approximating martingale for $0 \leq i \leq d-1$,

$$\begin{aligned} M_0 &= L_0 \\ M_{i+1} &= M_i + b_{i+1}L_{i+1} - b_i L_i - I_i \mathbb{E}[b_{i+1}L_{i+1} - b_i L_i \mid \mathscr{F}_i]. \end{aligned} \tag{3.16}$$

Theorem 3.8. *The constructed process M is a martingale.*

Proof. Consider first when $I_i = 0$. Then $\tau_i = \tau_{i+1}$, since we continue at time i. Then

$$\begin{aligned} b_i L_i &= \mathbb{E}[b_{\tau_i} h_{\tau_i} \mid \mathscr{F}_i] \\ &= \mathbb{E}[\mathbb{E}[b_{\tau_i} h_{\tau_i} \mid \mathscr{F}_{i+1}] \mid \mathscr{F}_i] \\ &= \mathbb{E}[\mathbb{E}[b_{\tau_{i+1}} h_{\tau_{i+1}} \mid \mathscr{F}_{i+1}] \mid \mathscr{F}_i] \\ &= \mathbb{E}[b_{i+1}L_{i+1} \mid \mathscr{F}_i], \end{aligned}$$

where the first and fourth equalities follow by definition and the second equality follows by the Tower Law. Since the expectation part vanishes when $I_i = 0$, the martingale property follows. Now consider when $I_i = 1$. Then

$$\begin{aligned} \mathbb{E}[M_{i+1} \mid \mathscr{F}_i] &= M_i + \mathbb{E}[b_{i+1}L_{i+1} - b_i L_i \mid \mathscr{F}_i] - \mathbb{E}[b_{i+1}L_{i+1} - b_i L_i \mid \mathscr{F}_i] \\ &= M_i. \end{aligned}$$

□

We note that since $b_i C_i = b_i L_i$ when $I_i = 0$ and $b_i C_i = \mathbb{E}[b_{i+1}L_{i+1} \mid \mathscr{F}_i]$ when $I_i = 1$ then the martingale simplifies to

$$\begin{aligned} M_0 &= L_0 \\ M_{i+1} &= M_i + b_{i+1}L_{i+1} - b_i C_i. \end{aligned} \tag{3.17}$$

We should briefly note that since $M_0 \neq 0$, then $M \notin H_0^1$. However this is solved through use of the martingale $N_t = M_t - M_0$, which is in H_0^1. Using (3.6),

$$U_0 := L_0 + \mathbb{E}[\max_{1 \leq i \leq d} (b_i h(S_i) - M_i) \mid \mathscr{F}_0] \tag{3.18}$$

gives an upper bound for the value of the option. The tightness of this bound is investigated by Andersen and Broadie [2].

To summarise, the LSMVR algorithm is run to produce a lower bound and regression coefficients for the approximated continuation values, which we denote $\widetilde{C}_i$. These are then used to construct the martingale for each sample path and then compute the Monte Carlo estimate to the expectation in (3.18).

The computation of M entails calculation of further conditional expectations. Andersen and Broadie propose that at each time step sub-simulations are run to compute $b_i C_i$. These sub-simulations are costly and greatly increase the computation time of the algorithm. We apply two techniques to reduce this computational cost.

Sub-Optimality Checking. We note that for paths in the continuation region the upper bound increment is zero since $b_i C_i = b_i L_i$. When certain that the path is in the continuation region ($h_i < C_i$), we can therefore skip these sub-simulations. Using the LSMVR approximation $\widetilde{C}_i$ for this check introduces the risk of incorrect continuation when

$$C_i \leq h_i < \widetilde{C}_i.$$

To combat this a lower bound of the continuation value $\underline{C}_i$ can be used, removing the risk of incorrect continuation. A sensible and easily attainable lower bound is the corresponding European option value. Avoiding these sub-simulations via this technique, which Broadie and Cao dub 'sub-optimality checking', will provide considerable efficiency improvements, especially in the case of deep out of the money options.

The lower bound provided by the European value allows many sub-simulations to occur as incorrect exercises. If the regression coefficients provided by the LSMVR algorithm are reasonably accurate then use of $\widetilde{C}$ would allow avoidance of a larger number of sub-simulations. In the following section we provide results from simulations and compare the usage of $\underline{C}_i$ and $\widetilde{C}_i$ in both computational time saving and accuracy.

One-Step Computation Andersen and Broadie allow the sub-simulations to run until stopped by the exercise policy. This is incredibly costly computationally. We instead perform a one-step computation, whereby we simulate only one-step into the future of the path and approximate the continuation value using the regression coefficients.

3.3 Numerical Results

We price a simple American put option on a single underlying asset following a GBM. The following parameters are used:

$$\begin{aligned} K &= 100, \\ r &= 6\%, \\ \sigma &= 40\%, \\ T &= 0.5, \end{aligned}$$

with S_0 varying. The simple American put on a single asset can be evaluated easily though a binomial scheme with a high number of time steps allowing comparison on the accuracy of the numerical results produced.

3.3.1 Least Squares Monte Carlo Method Results

We compare the use of basis functions. Through testing we observe that four basis functions suffices to provide accuracy with any of the polynomials chosen. For the first set of results we employ as basis functions the first four Laguerre polynomials. For the LSM method we use

Method	S_0	European Value	True Value	Simulated Value	Standard Error	Relative Error	Time (s)
	80	20.6893	21.6059	21.5985	0.0387	0.0344	5.8374
	90	14.4085	14.9187	14.9038	0.0382	0.1001	5.0766
LSM	100	9.6642	9.9458	9.8832	0.0354	0.6294	3.8809
	110	6.2797	6.4352	6.3783	0.0306	0.8843	3.0973
	120	3.9759	4.0611	4.0208	0.0243	0.9935	2.4243
	80	20.6893	21.6059	21.5889	0.0008676	0.0786	6.0103
	90	14.4085	14.9187	14.9013	0.0007440	0.1169	4.6148
LSMVR	100	9.6642	9.9458	9.9306	0.0005453	0.1531	3.8209
	110	6.2797	6.4352	6.4243	0.0004293	0.1698	3.0200
	120	3.9759	4.0611	4.0531	0.0003015	0.1981	2.3938

Table 3.1: Comparison between the LSM and LSMVR algorithms on a simple American put with parameters $K = 100$, $r = 6\%$, $\sigma = 40\%$ and $T = 0.5$. The algorithms are run with $100,000$ sample paths to estimate the regression coefficients and $100,000$ sample paths to then price the option. In the LSMVR algorithm this is split into $50,000$ antithetic pairs. There are 50 time steps used in the simulation. For these simulations Laguerre polynomials are used as the basis functions for the regression. The European option value is produced via the Black-Scholes formula and the true American option values are quoted from [1].

$100,000$ sample paths to estimate the regression coefficients and a further $100,000$ to then estimate the option price. For the LSMVR we use $50,000$ antithetic sample paths for both parts and utilise the discounted European option as a control variate. For either algorithm 50 time steps are used. Table 3.1 displays the results.

Instantly it is seen that the variance reductions techniques have succeeded, giving much decreased standard errors in roughly equivalent times. In addition for the LSVMR method, paths started out of the money ($S_0 \geq K$) have given superior simulation values and only slightly worse simulation values for paths started in the money ($S_0 < K$).

All the LSMVR simulation values are within 0.2% of the true value, providing excellent approximations to the value of our option. The relative error increases with the initial price value whilst the computation times decrease. This is because these options are out of the money more frequently and so require their continuation values to be re-evaluated less often.

We now utilise the polynomials defined in (3.12) as the basis functions, which we call from here on the choice functions. The results can be seen in Table 3.2. Using these polynomials produces the same patterns as the Laguerre polynomials, namely the increasing relative error and decreasing computation times. Variance reduction has again worked well.

The choice functions have produced better estimations with lower relative errors and similar standard errors. Despite their larger computation times we prefer the use of choice functions

Method	S_0	European Value	True Value	Simulated Value	Standard Error	Relative Error	Time (s)
LSM	80	20.6893	21.6059	21.6007	0.0377	0.0240	7.7036
	90	14.4085	14.9187	14.8979	0.0387	0.1392	6.3043
	100	9.6642	9.9458	9.9292	0.0355	0.1669	5.1246
	110	6.2797	6.4352	6.4171	0.0302	0.2810	4.3003
	120	3.9759	4.0611	4.0506	0.0244	0.2579	3.4814
LSMVR	80	20.6893	21.6059	21.5926	0.0008619	0.0617	7.8681
	90	14.4085	14.9187	14.9062	0.0007354	0.0840	6.3428
	100	9.6642	9.9458	9.9319	0.0005496	0.1400	5.1857
	110	6.2797	6.4352	6.4251	0.0003901	0.1577	4.3354
	120	3.9759	4.0611	4.0542	0.0003098	0.1694	3.5256

Table 3.2: Comparison between the LSM and LSMVR algorithms on a simple American put with parameters $K = 100$, $r = 6\%$, $\sigma = 40\%$ and $T = 0.5$. The algorithms are run with $100,000$ sample paths to estimate the regression coefficients and $100,000$ sample paths to then price the option. In the LSMVR algorithm this is split into $50,000$ antithetic pairs. There are 50 time steps used in the simulation. For these simulations the polynomials defined in (3.12) are used as the basis functions for the regression. The European option value is produced via the Black-Scholes formula and the true American option values are quoted from [1].

due to their superior accuracy and will utilise the LSMVR method with these as the basis functions in the hybrid algorithm.

3.3.2 Hybrid Method Results

We compare the implementation of sub-optimality checking with both $\underline{C}$ and $\widetilde{C}$. We let $\underline{C}$ be the corresponding European value of the option and recall $\widetilde{C}$ is the LSMVR approximation to the continuation value.

For both simulations we run $100,000$ pairs of antithetic sample paths to estimate the coefficients. To compute the approximations in reasonable time we use $5,000$ pairs of antithetic paths to compute the option price, with $5,000$ pairs of sub-paths launched when required. The corresponding European option is utilised as a control variant as before. Due to the algorithm's large computation times and since paths out of the money require much less time to run we alter the number of time steps used for each initial price so that each approximation is computed in roughly equivalent time, attempting to even the amount of resources used for each case. Each simulation takes roughly 750 seconds. The results are displayed in Table 3.3.

All the confidence intervals contain the true value for the option, indicating the success of the hybrid algorithm. The upper bounds computed are not as tight as their lower bound

C	S_0	True Value	LSMVR Value	Hybrid Value	Relative Error	Standard Error	Confidence Interval	d
	80	21.6059	21.5898	21.6307	0.1146	0.0013	[21.5881, 21.6338]	50
	90	14.9187	14.9097	14.9587	0.2680	0.0014	[14.9083, 14.9617]	100
$\underline{C}$	100	9.9458	9.9318	10.0140	0.6862	0.0013	[9.9308, 10.0168]	170
	110	6.4352	6.4269	6.4930	0.8985	0.0011	[6.4261, 6.4953]	300
	120	4.0611	4.0556	4.1010	0.9824	0.00081	[4.0550, 4.1027]	550
	80	21.6059	21.5923	21.6682	0.2883	0.0013	[21.5907, 21.6713]	80
	90	14.9187	14.9123	15.0352	0.7807	0.0015	[14.9110, 15.0383]	150
$\widetilde{C}$	100	9.9458	9.9382	10.0528	1.10755	0.0015	[9.9372, 10.0560]	290
	110	6.4352	6.4273	6.5474	1.7434	0.0023	[6.4265, 6.5520]	550
	120	4.0611	4.0543	4.0701	0.2223	0.0049	[4.0533, 4.0797]	950

Table 3.3: Simulation values from the dual method algorithm with sub-optimality checking implemented through comparison with either $\underline{C}$ or $\widetilde{C}$. Run on a simple American put with parameters $K = 100$, $r = 6\%$, $\sigma = 40\%$ and $T = 0.5$. The algorithms are run with $50,000$ pairs of antithetic sample paths to estimate the regression coefficients. $5,000$ pairs are then used to price the option with $5,000$ sub-simulations launched at exercise points. The number of time steps used is chosen for each initial price to ensure each initial price approximation uses a roughly equivalent amount of resources, each computation taking approximately 750 seconds. The relative and standard errors relate to the hybrid value approximation. The true American option values are quoted from [1].

counterparts, and we note that the tighter the lower bound is then the tighter its upper bound equivalent is.

By using $\widetilde{C}$ more sub-simulations have been rejected in sub-optimality checking, allowing for quicker computation times and therefore the use of a larger number of time steps. However, as expected, the utilisation of $\underline{C}$ in sub-optimality checking has achieved tighter upper bounds, since more sub-simulations are launched and the chance of incorrect continuation is nullified. The confidence intervals are tighter when $\underline{C}$ has been used and they form a symmetrical pattern, being tightest for $S_0 = 80$ and $S_0 = 120$ and loosest for $S_0 = 100$. This is expected since the $S_0 = 100$ case has the largest variance. In near all cases use of $\widetilde{C}$ has achieved tighter lower bounds but looser upper, indicating that allowing the incorrect continuations introduces a positive bias.

The confidence intervals produced are extremely tight. Comparing to the existing literature, the computation times are reasonably fast with the utilisation of sub-optimality checking. From the two methods tried the use of the European option has led to superior approximations, performing better even though its computational cost limits the number of time steps that can be used. In the following chapter we extend the hybrid method to price a game option and in particular combine this with the LSMVR method, aiming to form similar confidence intervals for the game option price.

4 Game Options

A Game Option extends an American option by allowing the writer of the option to prematurely cancel the option at a penalty. Their set-up is that of a Dynkin game, introduced by the namesake in his 1969 paper [19]; a Dynkin game is a class of zero-sum games that are based on optimal stopping. Kyprianou provides explicit solutions to the perpetual version of the option [39] whilst Kunita and Seko show for a call game option with no dividends the value function can be represented as a European option with the writer's premium subtracted [38]. Kühn and Kyprianou show that the case of a game option with a put payoff (what they call a 'callable put') is the composition of exotic options [36].

Not much research has looked into the computational methods for pricing such an option. Vaczlavik discusses finite element methods [52] whilst solving the stochastic differential game problem through backward stochastic differential equations is treated by Hamadene [26]. Eliasson approaches the problem through Monte Carlo, utilising a regression based method and proving convergence to the true value [20]. Kühn et al. proposes a similar dual set-up in parallel to Rogers work for American options [37]. However their computation is limited, as in Rogers, by being problem specific and not algorithmic.

Due to their nature, we can obtain several dual formulations to the problem, with each providing a basis for algorithmically computing the option price. In the American option case with relative ease we produce positively and negatively biased approximations. This is something that has not been algorithmically addressed, however, in the case of a game option. We remedy this by proposing an algorithm, based on the hybrid methods of the American option, which gives positively and negatively biased approximations. Theoretically this will lead to the production of upper and lower bounds for the price of a game option, from which we can construct confidence intervals for the price.

We begin with a subsection formulating the problem formally and deriving dual results. We extend the LSM method and hybrid algorithm to price game options. We then propose an algorithm combining these two methods and producing positively and negatively biased approximations for the game option price. We finish with numerical results and a comparison of the methods.

4.1 Pricing

We work in a complete market as before, the treatment of game options in incomplete markets is seen in Kallsen and Kühn [32], and Kühn [35]. Consider two càdlàg processes $(h_t)_{0\leq t\leq T}$ and $(g_t)_{0\leq t\leq T}$ such that $g_t = h_t + \delta_t$, where $\delta_t \geq 0$ is a penalty process. A game option is a contract between the writer and the holder such that either party may exercise the option at any time, up to the maturity time T. If the holder exercises first they receive the payoff h_t. If the writer cancels first then they must pay the sum of $g_t = h_t + \delta_t$ to the holder. If they exercise at the same time the payoff is h_t. The pricing of the option translates to an optimal stopping problem as before, with two stopping times to now consider. Let the holder of the option stop according to a stopping strategy τ and the writer of the option stops according to σ. The payoff of the option becomes

$$R(\tau, \sigma) = h_\tau \mathbb{1}_{\tau \le \sigma} + g_\sigma \mathbb{1}_{\sigma < \tau}. \tag{4.1}$$

We let $\delta_T = 0$. Intuitively, the writer of the option will pick σ to try and minimise the expected value $R(\tau, \sigma)$ whereas the holder chooses τ to maximise it. The theory of zero-sum Dynkin games gives the unique value of the option.

Definition 4.1. The value of a game option with payoff function (4.1) is given by the process $(V_t)_{0 \le t \le T}$, where

$$\begin{aligned} V_t &= \inf_{\sigma \in \mathscr{T}(t,T)} \sup_{\tau \in \mathscr{T}(t,T)} \mathbb{E}[b_{\tau \wedge \sigma} R(\tau, \sigma) \mid \mathscr{F}_t] \\ &= \sup_{\tau \in \mathscr{T}(t,T)} \inf_{\sigma \in \mathscr{T}(t,T)} \mathbb{E}[b_{\tau \wedge \sigma} R(\tau, \sigma) \mid \mathscr{F}_t]. \end{aligned} \tag{4.2}$$

Kifer showed the following, which we state without proof.

Theorem 4.2. *Consider a game option as above, with the fair price given by* (4.2) *such that h_t has no negative jumps and g_t has no positive jumps. Then at time t, the optimal stopping strategies for the holder and the writer are given by*

$$\tau_t^* = \inf \{ s \ge t : h_s \ge V_s \} \tag{4.3a}$$
$$\sigma_t^* = \inf \{ s \ge t : g_s \le V_s \} \wedge T. \tag{4.3b}$$

The holder's stopping strategy is as in the American option. The writer's strategy is again intuitive, stopping at the first time that continuing gives a higher expected payoff than exercising immediately.

4.1.1 Dual Formulations

To obtain the dual formulations, we first produce analogue results to that of Theorems 3.6 and 3.7 in the case where only the writer may cancel.

Definition 4.3. Consider a European option with payoff $(h_t)_{0 \le t \le T}$ in which the writer of the option may cancel the option at any time up to and including the maturity and pay the holder of the option a sum $g_t = h_t + \delta_t$. We call such an option a *cancellable European option* and define the arbitrage-free price W for such an option by

$$W_t = \inf_{\sigma \in \mathscr{T}(t,T)} \mathbb{E}[b_{t,\sigma} g_\sigma \mid \mathscr{F}_t]. \tag{4.4}$$

This option obeys similar decompositions to the American option. The proofs of these results follow through identical arguments to that of the American option. Indeed that the

Snell envelope of the cancellable European option is a lattice process is shown as before and the two following Lemmas follow as Theorems 3.6 and 3.7, respectively, with inequality signs reversed. As such, we omit their proofs.

Lemma 4.4. *Assume that the discounted cancellation process $b_t g_t$ is right-continuous and that for some $p > 1$, $\sup_{0 \le t \le T} |b_t g_t| \in L^p$. Then the lower Snell envelope of the cancellable European option,*

$$Y_t := b_t W_t = \operatorname*{ess\,inf}_{\sigma \in \mathscr{T}(t,T)} \mathbb{E}[b_\sigma g_\sigma \mid \mathscr{F}_t],$$

is a càdlàg sub-martingale of Class D.

Thus Y has a Doob-Meyer decomposition

$$Y_t = b_t W_t = W_0 + \widetilde{N}_t + \widetilde{B}_t, \tag{4.5}$$

where $\widetilde{N}$ is a unique, uniformly integrable martingale, $\widetilde{B}$ is a unique, integrable, increasing, predictable process and $\widetilde{N}_0 = \widetilde{B}_0 = 0$.

Lemma 4.5.

$$W_0 = \sup_{N \in H_0^1} \mathbb{E}[\inf_{s \in [0,T]} (b_s g_s - N_s)], \tag{4.6}$$

and this supremum is attained when $N = \widetilde{N}$, where $\widetilde{N}$ is the martingale from the Doob-Meyer decomposition in (4.5)*.*

Let Z denote the Snell envelope of the game option price, V. Under suitable conditions we can form a decomposition of this process up to the maximum of the two stopping times τ and σ.

Theorem 4.6. *Assume that the discounted payoff process $b_t h_t$ and penalty process δ_t are right-continuous and for some $p > 1$, $\sup_{0 \le t \le T} |b_t h_t| \in L^p$ and $\sup_{0 \le t \le T} |\delta_t| \in L^p$. Then the Snell envelope process up to time $\tau \vee \sigma$ can be decomposed into*

$$Z_t^{\tau \vee \sigma} = b_t V_t^{\tau \vee \sigma} = V_0 + M_t^* - A_t + B_t, \tag{4.7}$$

where $M^* \in H_0^1$, A is an integrable, non-decreasing, predictable process such that $A^\tau = 0$ and B is an integrable, non-decreasing, predictable process such that $B^\sigma = 0$.

Proof. Let τ^* denote the optimal stopping time for the holder of the option and σ^* denote the optimal stopping time for the writer of the option, which we can both stop at time T to guarantee existence. Consider the game option given prior knowledge of σ^*. This has payoff $R(t, \sigma^*)$ and value

$$V_t = \sup_{\tau \in \mathscr{T}} \mathbb{E}[b_{\tau \wedge \sigma^*} R(\tau, \sigma^*) \mid \mathscr{F}_t].$$

This can be viewed as an American option with maturity σ^* and payoff $R(t, \sigma^*) = h_t$ for $t \le \sigma^*$. By Theorem 3.6, since the regularity conditions are satisfied, up to time σ^* the discounted price process decomposes as a super-martingale

$$b_t V_t = V_0 + \widetilde{M}_t - \widetilde{A}_t, \tag{4.8}$$

where $\widetilde{M} \in \mathrm{H}_0^1$ is unique and $\widetilde{A}$ is a unique, integrable, increasing, predictable process with $\widetilde{M}_0 = \widetilde{A}_0 = 0$. Conversely, by considering the game option with prior knowledge of τ^* we reduce the option to a cancellable European option with maturity τ^* and payoff $R(\tau^*, s) = g_s$. From Lemma 4.4, since regularity conditions are again satisfied, the discounted price process up to time τ^* is a sub-martingale and decomposes as

$$b_t V_t = V_0 + \widetilde{N}_t + \widetilde{B}_t, \tag{4.9}$$

where $\widetilde{N} \in \mathrm{H}_0^1$ is unique and $\widetilde{B}$ is a unique, integrable, increasing, predictable process with $\widetilde{N}_0 = \widetilde{B}_0 = 0$. From (4.8) and (4.9) then $A^{\tau^*} = 0$ and $B^{\sigma^*} = 0$, and $\widetilde{M}^{\tau^* \wedge \sigma^*} = \widetilde{N}^{\tau^* \wedge \sigma^*}$. Therefore the discounted price process has decomposition

$$b_t V_t^{\tau^* \vee \sigma^*} = V_0 + M_t^* - A_t + B_t, \tag{4.10}$$

where $M^* \in \mathrm{H}_0^1$, with A and B both predictable increasing processes such that $A^{\tau^*} = 0$ and $B^{\sigma^*} = 0$. Further, $M^* = \widetilde{M}$ before time σ^* and $M^* = \widetilde{N}$ before time τ^*. □

Similarly to the American option, the martingale part of the above decomposition, M^*, provides the means of computing the value of the option path-wise.

Theorem 4.7. *Consider the game option as described above. Then*

$$\begin{aligned} V_0 &= \mathbb{E}\left[\sup_{t\in[0,T]} \inf_{s\in[0,T]} \left(b_{t\wedge s} R(t,s) - M^*_{t\wedge s}\right)\right] \\ &= \mathbb{E}\left[\inf_{s\in[0,T]} \sup_{t\in[0,T]} \left(b_{t\wedge s} R(t,s) - M^*_{t\wedge s}\right)\right]. \end{aligned} \tag{4.11}$$

Proof. Let τ^* and σ^* be as above, and write $\mathscr{T}$ for $\mathscr{T}(0,T)$. Then,

$$\begin{aligned} V_0 &= \sup_{\tau\in\mathscr{T}} \mathbb{E}[b_{\tau\wedge\sigma^*} R(\tau, \sigma^*)] \\ &= \mathbb{E}[\sup_{t\in[0,T]} \left(b_{t\wedge\sigma^*} R(t,\sigma^*) - \widetilde{M}_{t\wedge\sigma^*}\right)] \\ &= \mathbb{E}[\sup_{t\in[0,T]} (b_{t\wedge\sigma^*} R(t,\sigma^*) - M^*_{t\wedge\sigma^*})] \\ &\geq \mathbb{E}[\inf_{s\in[0,T]} \sup_{t\in[0,T]} (b_{t\wedge s} R(t,s) - M^*_{t\wedge s})], \end{aligned}$$

since $\widetilde{M} = M^*$ before the optimal stopping time, and where the second equality follows by Theorem 3.7. Conversely, we consider knowledge of τ^*,

$$\begin{aligned} V_0 &= \inf_{\sigma\in\mathscr{T}} \mathbb{E}[b_{\tau^*\wedge\sigma} R(\tau^*, \sigma)] \\ &= \mathbb{E}[\inf_{s\in[0,T]} \left(b_{\tau^*\wedge s} R(\tau^*, s) - \widetilde{N}_{\tau^*\wedge s}\right)] \\ &= \mathbb{E}[\inf_{s\in[0,T]} (b_{\tau^*\wedge s} R(\tau^*, s) - M^*_{\tau^*\wedge s})] \\ &\leq \mathbb{E}[\sup_{t\in[0,T]} \inf_{s\in[0,T]} (b_{t\wedge s} R(t,s) - M^*_{t\wedge s})], \end{aligned}$$

since $\widetilde{N} = M^*$ before the optimal stopping time, and where the second equality follows by Lemma 4.5. Together these two inequalities, along with the max-min inequality, give the required equality. □

Further, we construct a formulation of the option price by combining the choice of martingales with a stopping strategy.

Theorem 4.8. *Consider the game option as described above. Then*

$$V_0 = \inf_{M\in H_0^1} \inf_{\sigma\in\mathscr{T}} \mathbb{E}[\sup_{t\in[0,T]} (b_{t\wedge\sigma}R(t,\sigma) - M_{t\wedge\sigma})] \tag{4.12a}$$

$$= \sup_{M\in H_0^1} \sup_{\tau\in\mathscr{T}} \mathbb{E}[\inf_{s\in[0,T]} (b_{\tau\wedge s}R(\tau,s) - M_{\tau\wedge s})], \tag{4.12b}$$

where the infimum and supremum is achieved by M^*, the martingale part from the decomposition (4.10)*.*

Proof. This is shown through use of Theorem 3.7 and Lemma 4.5. Assuming prior knowledge of σ^* and using Theorem 3.7 on the induced American option with maturity σ^*,

$$\begin{aligned} V_0 &= \sup_{\tau\in\mathscr{T}} \mathbb{E}[b_{\tau\wedge\sigma^*}R(\tau,\sigma^*)] \\ &= \inf_{M\in \mathrm{H}_0^1} \mathbb{E}[\sup_{t\in[0,T]} (b_{t\wedge\sigma^*}R(t,\sigma^*) - M_{t\wedge\sigma^*})] \\ &= \inf_{M\in \mathrm{H}_0^1} \inf_{\sigma\in\mathscr{T}} \mathbb{E}[\sup_{t\in[0,T]} (b_{t\wedge\sigma}R(t,\sigma) - M_{t\wedge\sigma})], \end{aligned}$$

where the infimum is attained through $\widetilde{M}$ from the decomposition of the American option, which is equal to M^* before time σ^*. Similarly using prior knowledge of τ^* and by Lemma 4.5 on the induced cancellable European option,

$$\begin{aligned} V_0 &= \inf_{\sigma\in\mathscr{T}} \mathbb{E}[b_{\tau^*\wedge\sigma}R(\tau^*,\sigma)] \\ &= \sup_{N\in \mathrm{H}_0^1} \mathbb{E}[\inf_{s\in[0,T]} (b_{\tau^*\wedge s}R(\tau^*,s) - M_{\tau^*\wedge s})] \\ &= \sup_{N\in \mathrm{H}_0^1} \sup_{\tau\in\mathscr{T}} \mathbb{E}[\inf_{s\in[0,T]} (b_{\tau\wedge s}R(\tau,s) - M_{\tau\wedge s})], \end{aligned}$$

where the supremum is attained through $\widetilde{N}$ from the decomposition of the cancellable European option, which is equal to M^* before time τ^*. □

One final dual formulation is possible. From (4.12b), for any suitable martingale M we can can consider the embedded optimal stopping problem

$$V_0^M = \sup_{\tau\in\mathscr{T}} \mathbb{E}[\inf_{s\in[0,T]} (b_{\tau\wedge s}R(\tau,s) - M_{\tau\wedge s})].$$

If M is suitably bounded, this is seen as an American option and using Thoerem 3.7 we can consider the dual. This gives the price of the game option as

$$V_0 = \sup_{M\in \mathrm{H}_0^1} \inf_{N\in \mathrm{H}_0^1} \mathbb{E}\left[\sup_{t\in[0,T]} \left\{\inf_{s\in[0,T]} (b_{t\wedge s}R(t,s) - M_{t\wedge s}) - N_t\right\}\right],$$

with equality when M is the martingale part of the decomposition of the game option and N is the martingale part from the decomposition of the Snell envelope of the embedded optimal stopping problem V^M. This formulation is not as intuitive as the above and will be hard to use computationally, since the construction of a good approximating martingale to N is costly. Additionally, the approximation produced will be neither positively nor negatively biased. We will therefore make no use of this result in algorithmically pricing the option.

4.2 Algorithmic Methods

Whilst the LSM method has previously been utilised to produce approximations for the value of the game option, no paper has dealt with the production of positively and negatively biased approximations. Here, we present a new algorithm to do such, extending the hybrid approach of the American option to the game option case. Through this, we theoretically produce approximations of a positive and negative bias which in the continuous time case would give upper and lower bounds to the value of the game option.

We build to the presentation of this algorithm by outlining the dynamic programming principle for the game option, the LSMVR approach that accompanies it, and discussing how a constructed martingale can be used to approximate the price. We then present the proposed algorithm giving a discussion of techniques and possible improvements. We conclude the chapter with numerical results and a comparison of algorithms.

4.2.1 Least Squares Monte Carlo Method

Let $\{0 = t_0 < \cdots < t_d = T\}$ and suppress notation as before, letting i denote t_i. The continuation value of the game option at time i is

$$C_i = \mathbb{E}[b_{i,i+1} V_{i+1} \mid \mathscr{F}_i], \tag{4.13}$$

where V_i is the value of the game option at time i and $C_d = 0$. At each time step the holder exercises if their immediate exercise value h_i is greater than or equal to the continuation value C_i, and the writer cancels if the immediate cancellation value g_i is less than or equal to the continuation value C_i. This defines a backwards programming principle,

$$\begin{aligned} V_d &= g_d = h_d \\ V_i &= \min(g_i, \max(h_i, C_i)), \end{aligned} \tag{4.14}$$

for $i = 1, \dots, d-1$. Then $V_0 = C_0$, and the C-value recursion is given by

$$\begin{aligned} C_i &= \mathbb{E}[b_{i,i+1} \min\left(g_{i+1}, \max\left(h_{i+1}, C_{i+1}\right)\right) \mid \mathscr{F}_i] \\ C_d &= 0. \end{aligned} \tag{4.15}$$

Construction of the continuation values provides exercise strategies for both the holder and the writer,

$$\begin{aligned} \tau &= \min\{i = 1, \dots, d : h_i \geq C_i\} \\ \sigma &= \min\{i = 1, \dots, d : g_i \leq C_i\} \wedge T, \end{aligned} \tag{4.16}$$

discretised versions of (4.3a) and (4.3b). The option is valued via

$$V_0 = C_0 = \mathbb{E}[b_{\tau\wedge\sigma} R(\tau, \sigma)]. \tag{4.17}$$

The LSMVR method altered to reflect the new programming principle computes regression coefficients for the continuation value functions. The option is then priced by (4.17) using the the stopping strategies defined by (4.16). In the American option only paths in the money were used for the regression. Since now the writer can cancel early, paths out of the money still contain valuable information and must be included in the least-squares regression.

There is an important distinction from the American option. The value of the option in (4.2) is formulated through both a supremum and an infimum. Therefore we cannot expect the approximation produced by the LSMVR method to form an upper or lower bound for the option price.

4.2.2 Martingale Method

By Theorem 4.7, knowledge of M^* values the game option through (4.11). As with the American option we construct an approximation to M^*. We define an indicator process I_t, which at each point in time indicates whether the holder should exercise or the writer should cancel, and is zero otherwise. Note that that this indicator process takes into account both the holder and writer's decision process. We construct the martingale as before,

$$\begin{aligned} M_0 &= \widetilde{V}_0 \\ M_{i+1} &= M_i + b_{i+1}\widetilde{V}_{i+1} - b_i C_i. \end{aligned} \tag{4.18}$$

for $i = 0, \ldots, d-1$, where $\widetilde{V}$ is an approximate value process for the game option.

The game option variant of the LSMVR algorithm is used to find coefficients for the continuation value functions, which we use to build the above martingale. An approximation is then computed via (4.11). As with the LSMVR method we cannot expect the approximation to form either an upper or lower bound for the option price.

4.2.3 Biased Approximation Method

We develop a new approach, extending the hybrid method and combining it with the LSMVR method to produce theoretically positively and negatively biased approximations. Theorem 4.8 provides the basis for this. By selection of a martingale M and stopping strategy σ for the writer we theoretically attain an upper bound via (4.12a). Similarly, through a martingale M and stopping strategy τ for the holder we theoretically achieve a lower bound via (4.12b).

The optimal martingale can be approximated by (4.18) and the optimal stopping strategies for either party are approximated by (4.16). We note that when approximated there is no guarantee that the values produced will give true upper and lower bounds due to any simulation errors. A pseudo-algorithm for the computation of the positively biased approximation is seen in Algorithm 4.1.

Algorithm 4.1 Game Option Upper Bound algorithm

```
 1: Launch $N_1$ paths and use LSMVR method to compute regression coefficients.
 2: Launch $N_2$ independent paths.
 3: Use LSMVR method to compute approximation $\widetilde{V}_0$.
 4: for $n = 1 : N_2$ do
 5:     Set $\widetilde{M}_0^n = \widetilde{V}_0$.
 6:     for $i = 1 : d$ do
 7:         Compute $C_i^n$, $h_i^n$, $g_i^n$ and $\widetilde{V}_i^n$
 8:         if $C_i^n \geq g_i^n$ or $i = d$ then
 9:             Set $\widetilde{\sigma} = i$.
10:             Set $R(j, \widetilde{\sigma}) = R(i, \widetilde{\sigma})$ and $M_j^n = M_i^n$ for $j = i + 1, \ldots, d$.
11:             break
12:         else
13:             Set $R(i, \widetilde{\sigma}) = h_i^n$.
14:             if $C_i \leq h_i$ then
15:                 Launch $N_3$ subpaths from $S_i^n$.
16:                 Compute $b_i C_i^n = \mathbb{E}[b_{i+1} \widetilde{V}_{i+1}^n \mid \mathscr{F}_i]$.
17:             else
18:                 Set $b_i C_i^n = b_i \widetilde{V}_i^n$.
19:             end if
20:             Set $M_{i+1}^n = M_i^n + b_{i+1} \widetilde{V}_{i+1}^n - b_i C_i^n$.
21:         end if
22:     end for
23:     Compute $\bar{V}_0^n = \max_{i=1,\ldots,d} (b_{i \wedge \widetilde{\sigma}} R(i, \widetilde{\sigma}) - M_{i \wedge \widetilde{\sigma}}^n)$.
24: end for
25: Compute $\bar{V}_0 = \frac{1}{N_2} \sum_{n=1}^{N_2} \bar{V}_0^n$.
```

For clarity we briefly run through the steps. The first steps run the game option alteration of the LSMVR method from subsection 4.2.1 to compute the regression coefficients for the continuation value functions at each time point. To compute the value of the option N_2 independent paths are generated and we compute an approximation to the value of the option via the LSMVR method. Recalling (4.12a) to compute the positively biased value we approximate the optimal stopping strategy for the writer, $\widetilde{\sigma}$, and the martingale part from the decomposition of the Snell envelope. To do this, for each of the N_2 subpaths we build a martingale M^n. This martingale is set as $\widetilde{V}_0$ initially and built through (4.18). At each time step we also need know the immediate option value $R^n(t, \widetilde{\sigma})$.

To find both M^n and $R^n(t, \widetilde{\sigma})$ at each time step we compute the continuation value C_i^n using the regression coefficients and the LSMVR approximation for the value of the option at that time. The immediate exercise and cancellation prices are also found. We then check whether the writer should cancel using (4.16). If cancellation is indicated, the approximate stopping time $\widetilde{\sigma}$ is set as the current time step and M^n and $R^n(j, \widetilde{\sigma})$ are stopped, exiting the time step loop.

If cancellation is not indicated then we need compute new values for M^n and $R^n(j, \widetilde{\sigma})$. $R^n(j, \widetilde{\sigma})$ is set as the exercise value. To compute M^n we need find $b_i C_i^n$. To do so, we first check whether the holder should exercise. If exercise is indicated launch N_3 subpaths from the current state and compute $b_i C_i^n$ through a one-step computation. If exercise is not indicated set $b_i C_i^n$ to the current option value.

Now $R^n(i, \widetilde{\sigma})$ and M^n have been computed for all time steps we find the largest difference between the discounted immediate option value and the approximating martingale. This gives a positively biased approximation for the value of the game option for this single path. Finally we compute the Monte Carlo estimate.

An algorithm is similarly composed to produce a negatively biased value, by stopping the processes as soon as the holder exercises.

4.3 Numerical Results

We price a game option on a single underlying asset following a GBM, with put payoff $h_t = (K - S_t)^+$ and constant penalty process $\delta_t = \delta$. The following parameters are used:

$$\begin{aligned} K &= 100, \\ r &= 6\%, \\ \sigma &= 40\%, \\ T &= 0.5, \\ \delta &= 5, \end{aligned}$$

with S_0 varying. The true values quoted are from Kühn et al. [37].

4.3.1 Least Squares Monte Carlo Method Results

We utilise the choice functions defined by (3.12) as our basis functions for the regression. We tested the use of Laguerre polynomials and, whilst these were found to be quicker, they provided significantly less accurate results. We use $100,000$ pairs of antithetic pairs to estimate the regression coefficients and a further $100,000$ pairs to value the option. The game option is much more sensitive to the number of time steps used and we therefore utilise at least 500 time steps. We alter the number of time steps used for each initial price so that each approximation is computed in roughly equivalent time, with each simulation taking about 143 seconds. The results can be seen in Table 4.1.

The algorithm has produced reasonably accurate approximations to the true value of the option in sensible time. Noticeably the approximations worsen when the asset is started out of the money. The writer looks to cancel when the continuation value of the option is higher than the current exercise value plus the penalty. This region of space is located close to the

S_0	European Value	True Value	LSMVR Value	Standard Error	Relative Error	Relative SE	d
80	20.6893	20.6	20.5998	0.0196	0.000971	0.0949	500
90	14.4085	12.4	12.4183	0.0228	0.1475	0.1837	520
100	9.6642	5.00	5.4248	0.0061	8.4967	0.1119	570
110	6.2797	3.64	3.7836	0.0056	3.9457	0.1468	600
120	3.9759	2.54	2.6228	0.0061	3.2312	0.2325	642

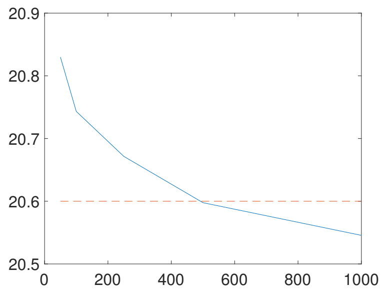
Table 4.1: Using the LSMVR method to approximate the value of a put game option with parameters $K = 100$, $r = 6\%$, $\sigma = 40\%$, $T = 0.5$ and $\delta = 5$. The algorithms are run with $100,000$ sample path pairs to estimate the regression coefficients and $100,000$ sample path pairs to then price the option. The number of time steps used is chosen for each initial price to ensure each initial price approximation uses a roughly equivalent amount of resources, each computation taking approximately 143 seconds. The European option value is produced via the Black-Scholes formula and the true game option values are quoted from [37].

	European	True			**d**		
S_0	Value	Value	50	100	250	500	1000
80	20.6893	20.6	20.8299	20.7433	20.6716	20.5977	20.5456
90	14.4085	12.4	12.8824	12.6971	12.4805	12.4250	12.3393
100	9.6642	5.00	6.3442	5.9596	5.6340	5.4480	5.3222
110	6.2797	3.64	4.0715	3.9513	3.8397	3.7937	3.7501
120	3.9759	2.54	2.8279	2.7487	2.6692	2.6354	2.6080

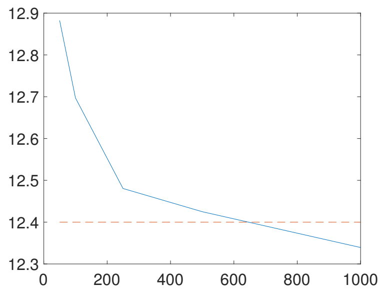
Table 4.2: Investigating changing number of time steps in the LSMVR algorithm for simulating a put game option with parameters $K = 100$, $r = 6\%$, $\sigma = 40\%$, $T = 0.5$ and $\delta = 5$. The algorithms are run with $100,000$ sample path pairs to estimate the regression coefficients and $100,000$ sample path pairs to then price the option. The European option value is produced via the Black-Scholes formula and the true game option values are quoted from [37].

strike price of the option, where the current exercise value is low but the continuation value is still relatively high. These cancellations occur more frequently in the algorithm for paths in the money more often, driving down the value of the option near its true value. As with American options, the computation time decreases for options more frequently out of the money since they require less frequent re-evaluation of their continuation values, allowing use of a greater number of time steps.

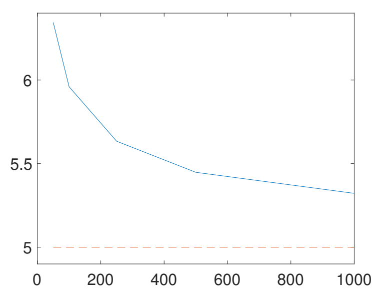
The use of at least 500 time steps to attain a good approximation is much greater than the 50 needed for an American option. We investigate how changing the number of time steps affects the approximation. We run the algorithm with 50, 100, 250, 500 and $1,000$ time steps for each initial price. The results can be seen in Table 4.2 and graphically in Figure 4.1.



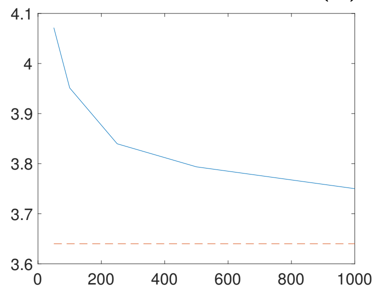
(a) $S_0 = 80$



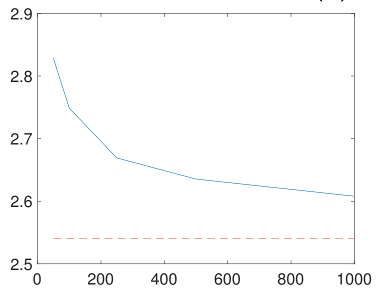
(b) $S_0 = 90$



(c) $S_0 = 100$



(d) $S_0 = 110$



(e) $S_0 = 120$

Figure 4.1: Convergence of the LSMVR simulation value of a put game option for increasing number of time steps. Parameters used are $K = 100$, $r = 6\%$, $\sigma = 40\%$, $T = 0.5$ and $\delta = 5$. The algorithms are run with $100,000$ sample path pairs to estimate the regression coefficients and $100,000$ sample path pairs to then price the option.

In each of the cases we see convergence towards the true value with overshooting occurring in the first two cases. As expected, increasing the number of time steps has improved the approximations. We note that this convergence always occurs from above. This is expected since it is known that the value of the game option is lower than the value of the corresponding European option, thus providing an early exercise feature is more beneficial to the writer of the option than the holder. Therefore increasing the number of time steps benefits the writer over the holder, decreasing the approximated value.

4.3.2 Martingale Method Results

For comparison, it is of interest to see how well any martingale will price the game option through (4.11). We compare the use of the discounted European option martingale and the constructed martingale (4.18), built through LSMVR approximations.

Since the construction of the European martingale is simple via Black-Scholes, we can implement high numbers of sample paths with a high number of time steps. This is not possible for the constructed martingale where building the martingale is costly on time resources. This brings forth a comparison between a good approximating martingale built on few sample paths and time steps, and a mediocre approximating martingale built on a larger number of sample paths. For the European option we use $50,000$ pairs of sample paths over 500 time steps, with each computation taking roughly 900 seconds. For the approximating mar-

Martingale	S_0	True Value	Martingale Value	Standard Error	Relative Error	Relative SE	d
European	80	20.6	21.6616	0.0028	5.1533	0.0131	500
	90	12.4	12.4501	0.0050	0.4042	0.0406	500
	100	5.00	5.2035	0.00066	4.0710	0.0127	500
	110	3.64	3.0587	0.0034	15.9698	0.1105	500
	120	2.54	2.4381	0.0043	4.0130	0.1748	500
Constructed	80	20.6	20.7765	0.0016	0.8568	0.00770	100
	90	12.4	12.5684	0.0020	1.3581	0.0159	175
	100	5.00	5.6406	0.00050	12.8113	0.0089	250
	110	3.64	3.7817	0.00038	3.8918	0.0100	400
	120	2.54	2.5934	0.00037	2.1034	0.0141	600

Table 4.3: Using (4.11) with two different martingales to approximate the value of a put game option with parameters $K = 100$, $r = 6\%$, $\sigma = 40\%$, $T = 0.5$ and $\delta = 5$. For the European martingale $50,000$ sample path pairs are used. For the constructed martingale $100,000$ pairs of antithetic paths are used to compute the regression coefficients but then only $5,000$ pairs of antithetic paths to compute the value of the option and $5,000$ pairs of sub-paths. The number of time steps used is chosen for each initial price to ensure each approximation uses a roughly equivalent amount of resources, each computation taking approximately 900 seconds. The true game option values are quoted from [37].

tingale we use $100,000$ pairs of antithetic paths to compute the regression coefficients but then only $5,000$ pairs of antithetic paths to compute the value of the option and $5,000$ pairs of sub-paths. We alter the number of time steps used to ensure that each initial price uses an equivalent amount of resources in computing the value, with each computation taking roughly 900 seconds. Since the corresponding European option no longer provides a lower bound on the value option we cannot implement it in sub-optimality checking. Instead we rely on $\widetilde{C}$ to be a good approximation to the continuation values. A control variant was excluded as when tested it did not significantly improve results and, with such large numbers of time steps, added to the computational time. The results can be seen in Table 4.3.

The European option gives a wide array of results with a good approximation for $S_0 = 90$ but poor approximations for the other cases, especially in the disastrous case of $S_0 = 110$. By using the European option we are in some sense attempting to solve the problem blind, ignoring any knowledge we may have of what form the optimal martingale takes. It is thus not surprising that for most cases the use of the European martingale is not brilliant.

The constructed martingale produces similar results to the LSMVR algorithm. Approximations are reasonable for each case except when the initial price is the same as the strike price. With starting prices in or at the money the LSMVR algorithm has outperformed this martingale approach, especially considering the LSMVR algorithms were run in roughly a sixth of the time. Interestingly in the cases initially out of the money the martingale

S_0	True Value	Negative Value	Positive Value	Lower SE	Upper SE	Confidence Interval	d
80	20.6	20.4704	20.8596	0.0281	0.0013	[20.4153, 20.8621]	100
90	12.4	12.5237	12.6132	0.00034	0.0021	[12.5230, 12.6173]	180
100	5.00	5.6403	5.6643	0.00062	0.00068	[5.6396, 5.6657]	260
110	3.64	3.7762	3.7914	0.00074	0.00042	[3.7747, 3.7923]	500
120	2.54	2.5821	2.5971	0.00031	0.00038	[2.5815, 2.5978]	900

Table 4.4: Results from proposed Algorithm 4.1 in approximating the value of a put game option with parameters $K = 100$, $r = 6\%$, $\sigma = 40\%$, $T = 0.5$ and $\delta = 5$. The algorithms are run with $100,000$ sample path pairs to estimate the regression coefficients but then only $5,000$ sample path pairs to then price the option, with $5,000$ sub-paths launched at sub-simulations. The number of time steps used is chosen for each initial price to ensure each initial price approximation uses a roughly equivalent amount of resources, each computation taking approximately 900 seconds. The true game option values are quoted from [37].

approximation outperforms the LSMVR approach, especially when $S_0 = 120$. However, it was found that when the LSMVR algorithm was allowed to run on an equal amount of time resources it subsequently performed better than the martingale approach. Therefore, if an approximation to the price of a game option is required it is preferable to utilise the LSMVR algorithm, unless the computation of a good approximating martingale can be achieved in much faster time.

4.3.3 Biased Approximation Method Results

We now implement proposed Algorithm 4.1 and a parallel one to compute a negatively biased approximation. We run both sides of the algorithm along the same computation to save time. The ultimate aim is that these positively and negatively biased results will form upper and lower bounds to the value of the option, respectively, and which can be used to produce a confidence interval for the game option price.

The computation time for this algorithm is similar to that of the martingale approach. Therefore, as before, we use $100,000$ pairs of antithetic paths to compute the regression coefficients but then only $5,000$ pairs of antithetic paths to compute the value of the option and $5,000$ pairs of sub-paths at sub-simulations. Again, we alter the number of time steps used to ensure that each initial price uses an equivalent amount of resources in computing the value, with each computation taking roughly 900 seconds. Results are seen in Table 4.4.

Disappointingly, in all cases except the first the negatively biased value produced is higher than the true value of the option. Indeed only for $S_0 = 80$ has a confidence interval been produced which contains the true value of the option. Discretising has introduced a positive bias into the simulations which has dominated the negative bias, preventing lower bounds to the value of the option being produced. If true values to the corresponding Bermudan game option were known then the algorithm certainly produces upper and lower bounds to these. However here, as we approximate continuous time, simulation errors have not made this possible.

All the confidence intervals produced do contain the simulated values from the martingale approach. This bounding of the martingale approach is expected from the formulation. Should the martingale method achieve its supremum over t at a time similar to that of the optimal stopping strategy of the holder then the path-wise approximation will be tighter to the upper bound. Conversely, if the infimum over s is achieved at a time close to the optimal stopping strategy of the writer then the lower bound produced will be much tighter to the martingale method value. If the martingale method produces simulation values close to the true value our proposed algorithm will give good confidence intervals for the game option price. For this to be the case we must reduce the discretisation error by increasing the number of time steps used in the simulation.

We therefore compute simulation values for increasing numbers of time steps. We run Algorithm 4.1, with the lower bound computation included, for 50, 100, 250, 500 and $1,000$ time steps. In the same simulation we compute the martingale method approximation to observe how tight the bounds are to this value. The results are seen in Table 4.5 and graphically in Figure 4.2, where the path-wise approximations are plotted as points.

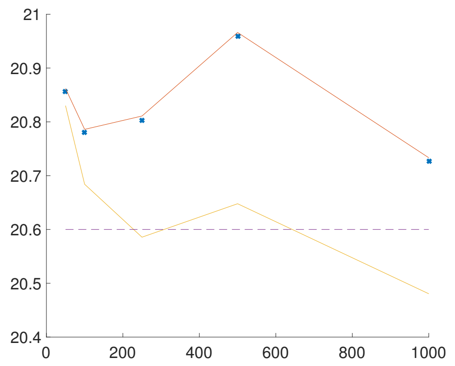
As above, we observe that for lower numbers of time steps the positive bias introduced through the discretisation error has prevented the production of lower bounds. However, in all cases convergence towards a confidence interval containing the true value occurs with both $S_0 = 80$ and $S_0 = 90$ giving true upper and lower bounds when $1,000$ time steps have been used. The values form tight bounds around the martingale approximation in all cases except for the unusually wide interval produced in the first case. This case was ran multiple times and the same pattern occurred each time. The positive approximation error and the convergence from above strengthen the hypothesis that increasing the number of time steps benefits the writer more than the holder.

Some interesting trends have emerged. In paths more frequently in the money the martingale method approximation is much tighter to the upper bound. From the above comments, we deduce that for paths more frequently in the money taking a supremum over all the time steps is roughly equivalent to the usage of the discretised optimal stopping strategy of the holder in (4.16). Conversely, when paths are more frequently out of the money tight lower bounds to the martingale method approximation are achieved and looser upper bounds. Similarly, we infer that for paths often out of the money taking an infimum over all time steps is roughly equivalent to the discretised optimal cancellation strategy for the writer in (4.16). The most impressive results have been produced for $S_0 = 100$. Here the martingale method approximation lies roughly at the midpoint of the biased values, which are both tight around the martingale method approximation. We attribute this to the fact that paths are out and in the money an even amount of time and so the taking of supremums and infimums is roughly equivalent to both the optimal stopping strategy for the holder and the writer, respectively.

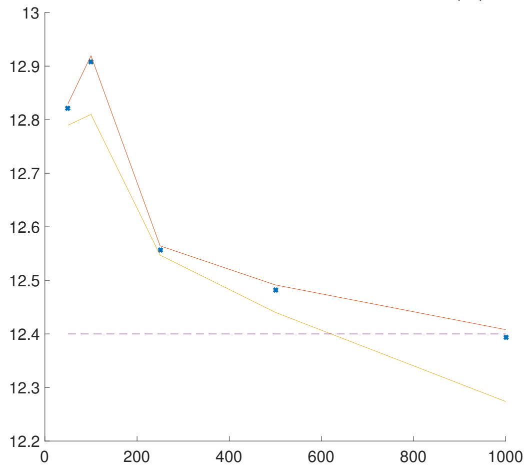
If a large enough number of time steps were used then true upper and lower bounds would

S_0	True Value	Method	**d** 50	100	250	500	1000
80	20.6	**Negative**	20.8302	20.6840	20.5857	20.6478	20.4805
		Martingale	20.8566	20.7802	20.8031	20.9584	20.7262
		Positive	20.8620	20.7860	20.8109	20.9662	20.7333
90	12.4	**Negative**	12.7896	12.8099	12.5471	12.4403	12.2738
		Martingale	12.8217	12.9080	12.5561	12.4820	12.3934
		Positive	12.8297	12.9194	12.5644	12.4915	12.4080
100	5.00	**Negative**	6.3024	6.0131	5.6655	5.4122	5.2807
		Martingale	6.3073	6.0230	5.6703	5.4153	5.2837
		Positive	6.3113	6.0302	5.6785	5.4233	5.2919
110	3.64	**Negative**	4.0819	3.9212	3.8242	3.8010	3.7545
		Martingale	4.0849	3.9252	3.8267	3.8046	3.7591
		Positive	4.0921	3.9345	3.8367	3.8161	3.7696
120	2.54	**Negative**	2.8334	2.6968	2.6256	2.6168	2.5889
		Martingale	2.8385	2.6993	2.6262	2.6214	2.5954
		Positive	2.8453	2.7081	2.6364	2.6338	2.6079

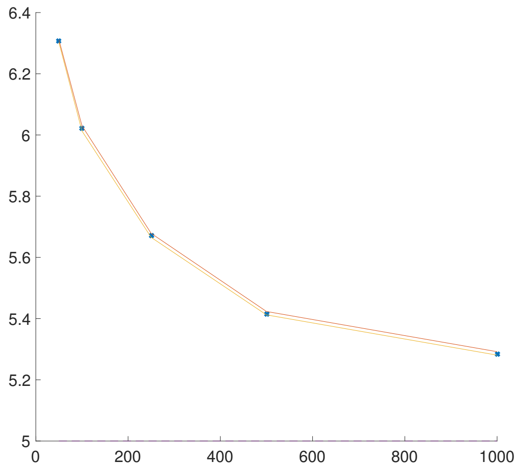
Table 4.5: Investigating changing number of time steps in Algorithm 4.1 for simulating a put game option with parameters $K = 100$, $r = 6\%$, $\sigma = 40\%$, $T = 0.5$ and $\delta = 5$. The algorithms are run with $100,000$ sample path pairs to estimate the regression coefficients and $5,000$ sample path pairs to then price the option, with $5,000$ sub-paths launched at sub-simulations. The true game option values are quoted from [37].



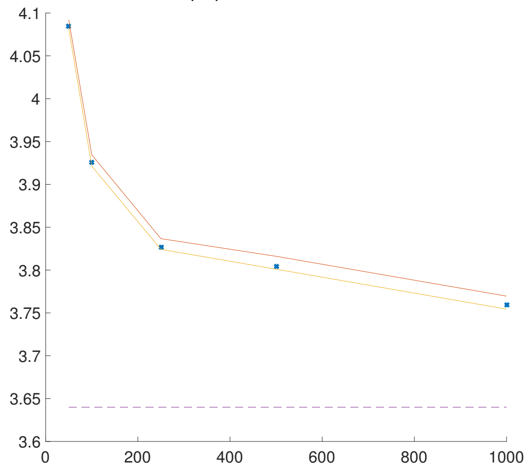
(a) $S_0 = 80$



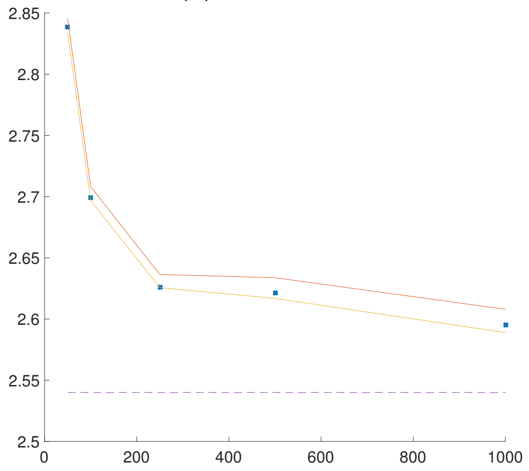
(b) $S_0 = 90$



(c) $S_0 = 100$



(d) $S_0 = 110$



(e) $S_0 = 120$

Figure 4.2: Convergence of the Algorithm 4.1 simulation values for a put game option with an increasing number of time steps. Martingale method approximations are plotted as crosses. Parameters used are $K = 100$, $r = 6\%$, $\sigma = 40\%$, $T = 0.5$ and $\delta = 5$. The algorithms are run with $100,000$ sample path pairs to estimate the regression coefficients and $5,000$ sample path pairs to then price the option, with $5,000$ sub-paths launched at sub-simulations.

be produced for the game option price. We run the three cases $S_0 = 100$, $S_0 = 110$ and $S_0 = 120$ with larger number of time steps to test this. However, even with $4,000$ time steps the negatively biased approximation lies above the true value. In each case the stated convergence is still present but computational limitations have prevented us to achieve the desired bounds, with convergence slowest for the $S_0 = 100$ case.

5 Conclusions

Due to their importance in the financial sector, the pricing of American options is still a vibrant area of research. Monte Carlo methods are a popular method for pricing these options, especially in cases where the option depends on multiple assets or has complex payoffs. In this dissertation we have reviewed two of the main algorithms for pricing an American option, namely the least-squares regression method (LSM) proposed by Longstaff and Schawrtz [40] and the hybrid algorithm proposed by Andersen and Broadie [2]. We have then extended these methodologies to the game option case and proposed an algorithm to produce positively and negatively biased values for the game option price.

Applying variance reduction techniques and several time-saving features the LSMVR method was shown to produce extremely good approximations to the value of the American option, as has been previously seen in the literature. Due to the negative bias of this method, this has led to the production of tight lower bounds to the American option value. Using the dual formulation of the problem, by use of a martingale, we achieved a positively biased approximation. Using the LSMVR method, we constructed such a martingale and with this attained upper bounds for the American option price. This hybrid approach produced confidence intervals for the value of the option. These confidence intervals were shown to be very tight around the true value of the option and indicate the success of the hybrid algorithm.

Using the American option methods as inspiration, we hoped to emulate similar results for the game option. In this direction, we began by formulating several different dual results for the value of the game option, with each indicating a varying method for computation of the option price. The LSMVR method was shown to be adaptable to the game option case and produce reasonable approximations with good convergence when increasing the number of time steps used. Equally via the construction of a martingale a reasonable approximation to the option price is found.

No attempt at constructing confidence intervals for the price of a game option had been seen in the literature. Attempting to remedy this, and using Theorem 4.8 as the theoretical basis, we proposed an algorithm that, through the selection of a martingale and a stopping strategy for either the writer or the holder, produced positively or negatively biased approximations to the game option price, respectively.

Time discretisation introduces a positive bias into the simulation which counteracts the negatively biased result and when few numbers of time steps were used a true lower bound was not produced. This positive bias is due to increasing the number of exercise opportunities benefiting the writer of the option more than the holder, as can be seen by the fact that a game option is priced lower than the corresponding European option. If true values were known for the Bermudan variant of the game option then these values would provide bounds for them. Instead, we must increase the number of time steps used to reduce the effect of the discretisation error. When a high enough number of time steps are used then the proposed algorithm should produce a true confidence interval for the value of the option. We were able to achieve this point for both of the cases started in the money. However, even when an extremely high number of time steps were used for cases started out of the money the convergence was shown to not be fast enough to achieve true confidence intervals in sensible time.

The high number of time steps required is extremely costly computationally taking well over an hour to simulate. Therefore the optimisation of the above algorithm is a sensible next step. Constructing the martingale is the time limiting step and its optmisation should come under first examination. Alternatively, one could look at constructing a better approximating martingale. Letting sub-paths run until they are stopped instead of the one-step computation used here may provide more accurate results, however this comes at a computational cost unless better optimised. The boundedness conditions that need to be satisfied by the payoff and cancellation processes h_t and g_t are relatively general and so a more complex payoff function that depends on the path of the asset could be implemented, as in Asian options. It would be of interest to see whether the proposed algorithm works as successfully in these cases, or when the option depends on a higher number of underlying assets.

References

[1] F. AitSahlia and P. Carr. American options: A comparison of numerical methods. *Numerical methods in finance*, pages 67–87, 1997.

[2] L. Andersen and M. Broadie. Primal-dual simulation algorithm for pricing multidimensional american options. *Management Science*, 50(9):1222–1234, 2004.

[3] L. B. Andersen. A simple approach to the pricing of bermudan swaptions in the multifactor libor market model. *Available at SSRN 155208*, 1999.

[4] D. Belomestny, C. Bender, and J. Schoenmakers. True upper bounds for bermudan products via non-nested monte carlo. *Mathematical Finance*, 19(1):53–71, 2009.

[5] D. Belomestny, J. Schoenmakers, and F. Dickmann. Multilevel dual approach for pricing american style derivatives. *Finance and Stochastics*, 17(4):717–742, 2013.

[6] C. Beveridge, M. S. Joshi, and R. Tang. Practical policy iteration: generic methods for obtaining rapid and tight bounds for bermudan exotic derivatives using monte carlo simulation. *Available at SSRN 1331904*, 2009.

[7] P. Boyle, M. Broadie, and P. Glasserman. Monte carlo methods for security pricing. *Journal of economic dynamics and control*, 21(8):1267–1321, 1997.

[8] M. J. Brennan and E. S. Schwartz. The valuation of american put options. *Journal of Finance*, pages 449–462, 1977.

[9] M. J. Brennan and E. S. Schwartz. Finite difference methods and jump processes arising in the pricing of contingent claims: A synthesis. *Journal of Financial and Quantitative Analysis*, 13(03):461–474, 1978.

[10] M. Broadie and M. Cao. Improved lower and upper bound algorithms for pricing american options by simulation. *Quantitative Finance*, 8(8):845–861, 2008.

[11] M. Broadie and J. B. Detemple. Anniversary article: Option pricing: Valuation models and applications. *Management science*, 50(9):1145–1177, 2004.

[12] M. Broadie and P. Glasserman. Pricing american-style securities using simulation. *Journal of Economic Dynamics and Control*, 21(8):1323–1352, 1997.

[13] M. Broadie and P. Glasserman. A stochastic mesh method for pricing high-dimensional american options. *Journal of Computational Finance*, 7:35–72, 2004.

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

[15] E. Clément, D. Lamberton, and P. Protter. An analysis of a least squares regression method for american option pricing. *Finance and Stochastics*, 6(4):449–471, 2002.

[16] J. C. Cox, S. A. Ross, and M. Rubinstein. Option pricing: A simplified approach. *Journal of financial Economics*, 7(3):229–263, 1979.

[17] F. Delbaen and W. Schachermayer. A general version of the fundamental theorem of asset pricing. *Mathematische annalen*, 300(1):463–520, 1994.

[18] C. Dellacherie and P.-A. Meyer. *Probabilities and Potential, C: Potential Theory for Discrete and Continuous Semigroups.* Elsevier, 2011.

[19] E. Dynkin. Game variant of a problem on optimal stopping. In *Soviet Math. Dokl.*, volume 10, pages 270–274, 1969.

[20] D. Eliasson. Game contingent claims. 2012.

[21] J.-P. Fouque, G. Papanicolaou, and K. R. Sircar. *Derivatives in financial markets with stochastic volatility.* Cambridge University Press, 2000.

[22] M. B. Giles. Multilevel monte carlo path simulation. *Operations Research*, 56(3):607–617, 2008.

[23] P. Glasserman. *Monte Carlo methods in financial engineering*, volume 53. Springer Science & Business Media, 2003.

[24] P. Glasserman, B. Yu, et al. Number of paths versus number of basis functions in american option pricing. *The Annals of Applied Probability*, 14(4):2090–2119, 2004.

[25] P. S. Hagan and D. E. Woodward. Equivalent black volatilities. *Applied Mathematical Finance*, 6(3):147–157, 1999.

[26] S. Hamadène. Mixed zero-sum stochastic differential game and american game options. *SIAM Journal on Control and Optimization*, 45(2):496–518, 2006.

[27] M. B. Haugh and L. Kogan. Pricing american options: a duality approach. *Operations Research*, 52(2):258–270, 2004.

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

[29] J. Hull and A. White. The pricing of options on assets with stochastic volatilities. *The journal of finance*, 42(2):281–300, 1987.

[30] P. Jackel. Monte carlo methods in finance. *Stochastic Dynamics*, 3:3–2, 2001.

[31] C. Jonen. An efficient implementation of a least squares monte carlo method for valuing american-style options. *International Journal of Computer Mathematics*, 86(6):1024–1039, 2009.

[32] J. Kallsen and C. Kühn. Pricing derivatives of american and game type in incomplete markets. *Finance and Stochastics*, 8(2):261–284, 2004.

[33] Y. Kifer. Game options. *Finance and Stochastics*, 4(4):443–463, 2000.

[34] A. Kolodko and J. Schoenmakers. Iterative construction of the optimal bermudan stopping time. *Finance and Stochastics*, 10(1):27–49, 2006.

[35] C. Kühn. Game contingent claims in complete and incomplete markets. *Journal of Mathematical Economics*, 40(8):889–902, 2004.

[36] C. Kühn and A. E. Kyprianou. Callable puts as composite exotic options. *Mathematical Finance*, 17(4):487–502, 2007.

[37] C. Kühn, A. E. Kyprianou, and K. Van Schaik. Pricing israeli options: a pathwise approach. *Stochastics An International Journal of Probability and Stochastic Processes*, 79(1-2):117–137, 2007.

[38] H. Kunita and S. Seko. Game call options and their exercise regions. *Preprint*, 2004.

[39] A. E. Kyprianou. Some calculations for israeli options. *Finance and Stochastics*, 8(1):73–86, 2004.

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

[41] M. Parkinson. Option pricing: the american put. *Journal of Business*, pages 21–36, 1977.

[42] A. Pelsser. *Efficient methods for valuing interest rate derivatives.* Springer Science & Business Media, 2000.

[43] D. Revuz and M. Yor. *Continuous martingales and Brownian motion*, volume 293. Springer Science & Business Media, 1999.

[44] L. Rogers. Dual valuation and hedging of bermudan options. *SIAM Journal on Financial Mathematics*, 1(1):604–608, 2010.

[45] L. Rogers. Bermudan options by simulation. *arXiv preprint arXiv:1508.06117*, 2015.

[46] L. C. Rogers. Monte carlo valuation of american options. *Mathematical Finance*, 12(3):271–286, 2002.

[47] N. Søndergaard Rasmussen. Efficient control variates for monte-carlo valuation of american options. *Available at SSRN 325260*, 2002.

[48] E. M. Stein and J. C. Stein. Stock price distributions with stochastic volatility: an analytic approach. *Review of financial Studies*, 4(4):727–752, 1991.

[49] N. Todorovic. Bewertung amerikanischer optionen mit hilfe von regressionsbasierten monte-carlo-verfahren. *Shaker, Aachen*, 2007.

[50] J. N. Tsitsiklis and B. Van Roy. Optimal stopping of markov processes: Hilbert space theory, approximation algorithms, and an application to pricing high-dimensional financial derivatives. *Automatic Control, IEEE Transactions on*, 44(10):1840–1851, 1999.

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

[52] L. Vaczlavik. Computational methods for game options. 2011.

[53] H. White. *Asymptotic theory for econometricians.* Academic press, 2014.

A MATLAB Code

Included here is MATLAB code written by the author in the valuation of American and game options. The first section includes various small functions written and used in the computations. The second section contains two of the main scripts used. One is the valuation of an American option with sub-optimality checking through comparison with the European option. The other is the proposed Algorithm 4.1 to compute positively biased approximations to the game option price, with the negatively biased approximation computation also included. These codes are chosen as they utilise several of the techniques used in the dissertation including: computation of regression coefficients, valuation of both options through the LSM method, variance reduction techniques, sub-optimality checking and other time-saving features.

A.1 Sub-functions

A.1.1 Black-Scholes European Option Function

```
function [BSprice] = BSput(K,T,r,s,S0);
% Function that takes in parameters for a European put option and outputs
% the Black-Scholes solution.
% K = strike price
% T = maturity
% r = interest rate
% s = volatility
% S0 = initial price

% Now calculate the BS explicit value
d1 = (log(S0/K) + (r + s^2/2)*T)/(s*sqrt(T));
d2 = (log(S0/K) + (r - s^2/2)*T)/(s*sqrt(T));
d2 = normcdf(-d2)*K*exp(-r*T);
d1 = normcdf(-d1).*S0;

BSprice = d2 - d1;
```

A.1.2 Laguerre Polynomial Generator

```
function Y = Laguerre(k,S);
% Function that takes in a data vector S and calculates the kth Laguerre
% polynomial.

if (k==0)
    Y = 1;
elseif (k==1)
    Y = 1-S;
```

```
else
    Y = (1/k)*((2*k-1-S).*Laguerre(k-1,S)-(k-1)*Laguerre(k-2,S));
end

end
```

A.1.3 Generate Basis Functions with Laguerre Polynomials

```
function basisFunctions = generateBasisFunctions(S,M);
% Function that takes in a data vector S and generates a matrix containing
% the basis values for each path, using the Laguerre polynomials as basis
% functions.
% M = number of basis functions
% K = strike price
% T = maturity
% r = interest rate
% s = volatility

% Initialise
basisFunctions = zeros(length(S),M);

% %% Laguerre
for i = 1:M
    % Find the Laguerre polynomial value for the numer of basis functions
    % required.
    basisFunctions(:,i) = Laguerre(i-1,S);
end

end
```

A.1.4 Choice Polynomial Generator

```
function Y = choiceFunctions(k,S,K,T,r,s);
% Function that takes in a data vector S and calculates the kth choice
% polynomial, as defined in the text.
% K = strike price
% T = maturity
% r = interest rate
% s = volatility

if (k==0)
    % Strike price.
    Y = K;
elseif (k==1)
    % Current state.
    Y = S;
elseif (k==2)
```

```
    % Equivalent European put option price.
   Y = BSput(K,T,r,s,S);
else
    % Current state multiplied by the equivalent European put option price.
   Y = S.*BSput(K,T,r,s,S);
end

end
```

A.1.5 Generate Basis Functions With Choice Functions

```
function basisFunctions = generateChoiceFunctions(S,M,K,T,r,s);
% Function that takes in a data vector S and generates a matrix containing
% the basis values for each path, using the choice functions as basis
% functions.
% M = number of basis functions
% K = strike price
% T = maturity
% r = interest rate
% s = volatility

% Initialise
basisFunctions = zeros(length(S),M);

for i = 1:M
    % Find the choice function value for the numer of basis functions
    % required.
    basisFunctions(:,i) = choiceFunctions(i-1,S,K,T,r,s);
end


end
```

A.1.6 LSMVR American Option Coefficients Function

```
function [beta] = LSMregressioncoefficientsAntithetic(K,T,r,s,S0,N,d,M)
% Function that calculates the coefficients for the continuation value
% approximation function by regression using the LSM method. Variance
% reduction techniques (antithetic variates) are
% applied.
% K = strike price
% T = maturity
% r = interest rate
% s = volatility
% S0 = initial price
% N = number of sample paths
% d = number of time steps
% M = number of basis functions

dt = T/d; % Calculate time step size.

%% Generate sample paths
% First generate the sample paths in a matrix. Each column corresponds to a
% different path.

% Initialise
S = zeros(d+1,2*N);

S(1,:) = S0; % first entry is initial price

for i = 2:d+1;
    Z = randn(1,N);
    Z = [Z,-Z]; % create antithetic pairs
    S(i,:) = S(i-1,:).*exp((r - s^2/2)*dt + s*Z*sqrt(dt));
end

%% Calculate the payoff matrix
% h is the payoff matrix. Each column corresponds to the immediate payoffs
% along a path at each time. Note that time 0 is not included.
h = max(K-S(2:d+1,:),0);

%% Calculate regression coefficients
C = zeros(1,2*N); % continuation values for each path.
beta = zeros(d,M);  % matrix to store the regression coefficients includes
                    % time 0 but not time d.

% Work backwards from maturity. First at time d.
C(1,:) = exp(-r*dt).*h(d,:); % set continuation value as exercise value

for i = d-1:-1:1
    % At time i find where asset is in the money.
    I = S(i+1,:) < K;
    Iindex = find(S(i+1,:)<K);

    X = S(i+1,I)'; % only paths in the money.

    % We retrieve the continuation values for paths in the money.
    Csub = C(1,I);

    Y = Csub'; % transpose.

    % CHOOSE either basis polynomials or choice functions.
    %D = generateBasisFunctions(X,M); % design matrix
    D = generateChoiceFunctions(X,M,K,(d-i)*dt,r,s);

    betaSub = D\Y; % coefficients using least squares regression
    beta(i+1,:) = betaSub; % add to the coefficient matrix


    contValue = D*betaSub; % continuation value
    exerciseValue = max(K - X,0); % immediate exercise value

    % Check when to update the continuation values for paths in the money.
    Icont = find(contValue <=  exerciseValue  & exerciseValue > 0)';

    C(1,Iindex(Icont)) = exerciseValue(Icont); % update these paths

    % Continuation values are discounted one more time step.
    C(1,:) = exp(-r*dt).*C(1,:);
end
```

A.1.7 LSMVR Game Option Coefficients Function

```
function [beta] = gameOptionCoefficients(K,T,r,s,S0,N,d,M,delta)
% Function to calculate the regression coefficients using the LSM method.
% These regression coefficients can then be used to provide an exercise
% strategy to be used in Anderon/Broadie simulation. Function that
% calculates the coefficients for the continuation value approximation
% function for a game option by regression using the LSM method. Variance
% reduction techniques (antithetic variates and control variates) are
% applied.
% K = strike price
% T = maturity
% r = interest rate
% s = volatility
% S0 = initial price
% N = number of sample paths
% d = number of time steps
% M = number of basis functions
% delta = fixed penalty

dt = T/d; % calculate the time step size

%% Generate sample paths
% First generate the sample paths in a matrix. Each column corresponds to a
% different path.

% Initialise
S = zeros(d+1,2*N);

S(1,:) = S0; % first entry is initial price

for i = 2:d+1;
    Z = randn(1,N);
    Z = [Z,-Z]; % create antithetic pairs
    S(i,:) = S(i-1,:).*exp((r - s^2/2)*dt + s*Z*sqrt(dt));
end

%% Calculate the payoff matrix
% h is the payoff matrix. Each column corresponds to the immediate payoffs
% along a path at each time. Note that time 0 is not included.
```

```
h = max(K-S(2:d+1,:),0);

%% Calculate regression coefficients
C = zeros(1,2*N); % continuation values for each path.
beta = zeros(d,M);  % matrix to store the regression coefficients includes
                    % time 0 but not time d.

% Work backwards from maturity. First at time d.
C(1,:) = exp(-r*dt).*h(d,:); % set continuation value as exercise value

for i = d-1:-1:1

    X = S(i+1,:)'; % paths at current time step
    Y = C(1,:)'; % tranpose of continuation values

    % CHOOSE either basis polynomials or choice functions.
    %D = generateBasisFunctions(X,M); % design matrix
    D = generateChoiceFunctions(X,M,K,(d-i)*dt,r,s);

    betaSub = D\Y; % coefficients using least squares regression
    beta(i+1,:) = betaSub; % add to the coefficient matrix

    contValue = D*betaSub; % continuation value
    exerciseValue = max(K-X,0); % immediate exercise value
    penaltyValue = exerciseValue + delta; % immediate cancellation value

    % Update the continuation values and discount back one time step.
    C(1,:) = exp(-r*dt).*min(penaltyValue,max(exerciseValue,contValue));


end

end
```

A.2 Main Functions

A.2.1 Dual Positively Biased Simulation Value Function with European Option Sub-Optimality Checking

```
function [lowerBound,europeanValue,upperBound,upperStdError,totaltime,CI]...
                                = americanDualEuropeanSubOptimality(S0,d);
%% American Option Dual Valuation with European Sub-Optimality Checking
% Function that calculates a high-biased approximation to the value of an
% American put option on single asset following a geometric Brownian
% motion. Variance reduction techniques (antithetic variates and control
% variates) are applied. Sub-optimality checking is implemented by
% comparison with the corresponding European option. A low-biased
% approximation is also produced using the LSMVR method.
% S0 = initial price
% d = number of time steps

tic; % start timer

%% Set up variables
K = 100; % strike price
T = 0.5; % maturity
r = 0.06; % interest
s = 0.4; % volatility (sigma)
N = 1*10^5; % number of sample paths for coefficients
N2 = 5*10^3; % number of sample paths for upper bound
N3 = 5*10^3; % number of sub-path loops
M = 4; % number of basis functions
dt = T/d; % size of each timestep

%% First find European value
europeanValue = BSput(K,T,r,s,S0)

%% Calculate Regression Coefficients
% Utilise the LSMVR method to compute regression coefficients which define
% the exercise policy.
[beta] = LSMregressioncoefficientsAntithetic(K,T,r,s,S0,N,d,M);

%% Generate new sample paths for lower bound
% Generate new independent sample paths. Each column corresponds to a
% different path.

% Initialise
S = zeros(d+1,2*N2);

S(1,:) = S0; % first entry is initial price

for i = 2:d+1;
    Z = randn(1,N2);
    Z = [Z,-Z]; % create antithetic pairs
    S(i,:) = S(i-1,:).*exp((r - s^2/2)*dt + s*Z*sqrt(dt));
end

%% Calculate the payoff matrix
% h is the payoff matrix. Each column corresponds to the immediate payoffs
% along a path at each time. Note that time 0 is not included.
h = max(K-S(2:d+1,:),0);

%% Compute the continuation matrix
% Continuation matrix holds the continuation value at each time point for
% each sample path.
C = zeros(d,2*N2); % no time 0, but time d

for i = 1:d-1
    % CHOOSE either basis polynomials or choice functions.
    %D = generateBasisFunctions(S(i+1,:),M);
    D = generateChoiceFunctions(S(i+1,:),M,K,(d-i)*dt,r,s);

    % Using the regression coefficients we estimate the continuation
    % values.
    C(i,:) = D*beta(i+1,:)';
end

%% Compute low-biased simulation value
% Making use of a control variate to reduce variance.

controlVariate = zeros(1,2*N2); % stores the control variate values
Y = zeros(1,2*N2); % stores the simulation values

for n = 1:2*N2
    % Find the first exercise time.
    indx = find(h(:,n) >= C(:,n) & h(:,n)>0,1);

    % If we exercise then set the simulation value and control variate
    % value as that time point discounted back to 0.
    if indx
        Y(n) = exp(-r*dt*indx)*h(indx,n);
        controlVariate(n) = ...
                        exp(-r*dt*indx)*BSput(K,(d-indx)*dt,r,s,S(indx+1,n));
    end
end

% Use the control variate to compute a variance reduced simulation value.
meanControl = mean(controlVariate);
meanNormal = mean(Y);
covarianceEstimate = mean(Y.*controlVariate);
varianceEstimate = mean(controlVariate.^2);
controlVariateConstant = -(covarianceEstimate - meanNormal*meanControl)/...
                                        (varianceEstimate - meanControl^2);
Z = Y + controlVariateConstant*(controlVariate - europeanValue);

% Z contains the estimate for each sample. Taking the mean we obtain the
% Monte Carlo estimate.
lowerBound = mean(Z)
lowerStdError = std(Z)/sqrt(2*N2)

%% Calculate the European option values for each sample path
% Time 0 not included.
europeanValues = zeros(d,2*N2);

for i = 1:d
    europeanValues(i,:) = BSput(K,(d-i)*dt,r,s,S(i+1,:));
end


%% Build the exercise indicator matrix
I = (h > europeanValues) & (h>0);
I(d,:) = h(d,:) > 0;

V = max(h,C); % option values at each time point
clear D Z;

%% Construct approximating martingale
```

```
mart = zeros(d,2*N2); % no time 0
mart(1,:) = exp(-r*dt).*V(1,:); % set time 1 value

for i=1:d-1
    % At each time step check which paths we need run sub simulations
    timePaths = S(i+1,:); % the values at the current time

    % Extract the paths we need to run sub-simulations for.
    relaventTimePaths = timePaths(I(i,:));
    subloopsNeeded = length(relaventTimePaths);

    % For paths that don't require sub-simulations we set these as the
    % discounted low-biased approximation values.
    diff = zeros(1,2*N2);
    tempV1 = V(i+1,:); % one time step ahead
    tempV2 = V(i,:);
    % Set the martingale difference as b(t+1)L(t+1) - btLt for paths not
    % requiring sub-simulations.
    diff(~I(i,:)) = ...
        exp(-r*dt*(i+1)).*tempV1(~I(i,:)) - exp(-r*dt*i).*tempV2(~I(i,:));

    % For paths requiring sub-simulations we run these and compute their
    % Monte Carlo estimates for the continuation values.
    means = zeros(1,subloopsNeeded);

    for n=1:subloopsNeeded
        % For each path that requires sub-simulations we launch N3
        % sub-paths and compute the continuation values through a one-step
        % expectation computation.
        % Generate sub-paths one step ahead.
        Z = randn(1,N3);
        Z = [Z,-Z]; % create antithetic pairs
        subS = relaventTimePaths(1,n).*exp((r-s^2/2)*dt + s*Z*sqrt(dt));

        % Exercise values.
        subH = max(K-subS,0);

        if i==d-1
            % If at the penultimate time step then one step ahead is
            % maturity where there is no continatuation so set the
            % expectation as the exercise value.
            means(1,n) = mean(subH);
        else
            % If not at the penultimate time step then find the
            % continuation values.

            % CHOOSE either basis polynomials or choice functions.
            %subD = generateBasisFunctions(subS(n,:),M);
            subD = generateChoiceFunctions(subS,M,K,(d-i-1)*dt,r,s);

            subC = (subD*beta(i+2,:)')'; % approximated continuation values
            % Discounted option value.
            subV = exp(-r*dt*(i+1)).*max(subH,subC);
            means(1,n) = mean(subV); % monte carlo estimate
        end
    end

    % Calculate martingale difference for paths that required
    % sub-simulations.
    diff(I(i,:)) = exp(-r*dt*(i+1)).*tempV1(I(i,:))- means;

    % Update martingale.
    mart(i+1,:) = mart(i,:) + diff;
end
clear tempV1 tempV2 diff subD subC subH subV relaventTimePaths timePaths;

% Modify all the exercise values so they are discounted.
for i = 1:d
    h(i,:) = exp(-r*dt*i).*h(i,:);
end

% Find the largest difference between the discounted payoffs and
% martingale.
diff = h - mart;
maximums = max(diff);

%% Compute high-biased simulation value
% Use control variate to reduce variance.
controlVariate = zeros(1,2*N2); % stores control variate values
Y = max(diff); % stores simulation values

for i = 1:2*N2
    % Find first exercise time.
    indx = find(h(:,i) >= C(:,i) & h(:,i)>0,1);
    if indx
        controlVariate(i) = exp(-r*dt*indx)*...
                                    BSput(K,(d-indx)*dt,r,s,S(indx+1,i));
    end
end

% Use the control variate to compute a variance reduced simulation value.
meanControl = mean(controlVariate);
meanNormal = mean(Y);
covarianceEstimate = mean(Y.*controlVariate);
varianceEstimate = mean(controlVariate.^2);
controlVariateConstant = -(covarianceEstimate - meanNormal*meanControl)/...
                                    (varianceEstimate - meanControl^2);
Z = Y + controlVariateConstant*(controlVariate - europeanValue);

% Z contains the estimate for each sample. Taking the mean we obtain the
% Monte Carlo estimate.
upperStdError = std(Z)/sqrt(2*N2)
upperBound = mean(Z) + lowerBound

% Construct confidence interval.
alpha = 0.05;
z = norminv(1-alpha/2);
CIlower = lowerBound - z*lowerStdError;
```

```
CIupper = upperBound + z*sqrt(lowerStdError^2 + upperStdError^2);
CI = [CIlower,CIupper]

totaltime = toc

end
```

A.2.2 Proposed Algorithm 4.1 to Compute Positively and Negatively Biased Results

```
function [europeanValue,lowerBound,lowerStdError,upperBound,...
            upperStdError,martApproximation,totaltime] ...
                                                = gameOptionBoundsPath(S0,d);
%% Game Option Bounds Valuation
% Function that calculates a high and low-biased approximation to the value
% of a game put option on single asset following a geometric Brownian
% motion. Variance reduction techniques (antithetic variates and control
% variates) are applied. A path-wise approximation is also produced using
% the approximating martingale.
% S0 = initial price
% d = number of time steps

tic; % start timer

%% Set up variables
K = 100; % strike price
T = 0.5; % maturity
r = 0.06; % interest
s = 0.4; % volatility (sigma)
N = 1*10^4; % number of sample paths for coefficients
N2 = 5*10^3; % number of sample paths for bound valuation
N3 = 5*10^3; % number of sub-path loops
M = 4; % number of basis functions
dt = T/d; % size of each timestep
delta = 5; % penalty payoff

%% First find European value
europeanValue = BSput(K,T,r,s,S0)

%% Calculate regression coefficients
% Utilise the LSMVR method to compute regression coefficients which define
% the exercise policy.
[beta] = gameOptionCoefficients(K,T,r,s,S0,N,d,M,delta);

%% Generate new sample paths for lower bound
% Generate new independent sample paths. Each column corresponds to a
% different path.

% Initialise
S = zeros(d+1,2*N2);
```

```

S(1,:) = S0; % first entry is initial price

for i = 2:d+1;
    Z = randn(1,N2);
    Z = [Z,-Z]; % create antithetic pairs
    S(i,:) = S(i-1,:).*exp((r - s^2/2)*dt + s*Z*sqrt(dt));
end

%% Calculate the payoff matrix
% h is the payoff matrix. Each column corresponds to the immediate payoffs
% along a path at each time. Note that time 0 is not included.
h = max(K-S(2:d+1,:),0);
g = h + delta; % penalty matrix
g(d,:) = h(d,:); % take off penalty at maturity

%% Compute the continuation matrix
% Continuation matrix holds the continuation value at each time point for
% each sample path.
C = zeros(d,2*N2); % no time 0, but time d

for i = 1:d-1
    % CHOOSE either basis polynomials or choice functions.
    %D = generateBasisFunctions(S(i+1,:),M);
    D = generateChoiceFunctions(S(i+1,:),M,K,(d-i)*dt,r,s);

    % Using the regression coefficients we estimate the continuation
    % values.
    C(i,:) = D*beta(i+1,:)';
end

%% Find the LSMVR value
Y = zeros(1,2*N2);
for i = 1:2*N2
    indx1 = find(h(:,i) >= C(:,i) & h(:,i)>0,1); % holder stopping time
    indx2 = find(g(:,i) <= C(:,i),1); % writer stopping time

    if isempty(indx1) == 1
        % If holder doesn't stop then set to avoid crashing.
        indx1 = d+1;
    end

    if isempty(indx2) == 1
        % Writer always exercises at maturity.
        indx2 = d;
    end

    % Now find the discounted stopped value of the option.
    if indx1 <= indx2
        Y(1,i) = exp(-r*dt*indx1)*h(indx1,i);
    else
        Y(1,i) = exp(-r*dt*indx2)*g(indx2,i);
    end
end
```

```
% Compute the Monte Carlo approximate to the LSMVR value.
lsmvrApprox = mean(Y);

%% Find the approximate sigma
sigma = zeros(1,2*N2); % stores estimated sigma for each path
for i = 1:2*N2
    indx = find(g(:,i) <= C(:,i),1);
    if indx
        sigma(1,i) = indx; % stores the cancellation time
    else
        sigma(1,i) = d; % always exercise at maturity
    end
end

%% Find the approximate tau
tau = zeros(1,2*N2); % stores estimated tau for each path
for i = 1:2*N2
    indx = find(h(:,i) >= C(:,i) & h(:,i) > 0,1);
    %indx = find(h(:,i) >= C(:,i),1);
    if indx
        tau(1,i) = indx; % stores the exercise time
    else
        tau(1,i) = d+1; % to avoid crashing
    end
end

%% Build the exercise/cancellation indicator matrix
I = ((h >= C) & (h>0)) | (g <= C);
I(d,:) = 1; % writer always cancels at maturity
V = min(g,max(h,C)); % option values at each time point
clear D Z;

%% Construct approximating martingale
mart = zeros(d,2*N2); % no time 0
mart(1,:) = exp(-r*dt).*V(1,:); % set time 1 value

for i=1:d-1

    % At each time step check which paths we need run sub simulations
    timePaths = S(i+1,:); % the values at the current time

    % Extract the paths we need to run sub-simulations for.
    relaventTimePaths = timePaths(I(i,:));
    subloopsNeeded = length(relaventTimePaths);

    % For paths that don't require sub-simulations we set these as the
    % discounted low-biased approximation values.
    diff = zeros(1,2*N2);
    tempV1 = V(i+1,:); % one time step ahead
    tempV2 = V(i,:);
    % Set the martingale difference as b(t+1)L(t+1) - btLt for paths not
    % requiring sub-simulations.
    diff(~I(i,:)) =...
        exp(-r*dt*(i+1)).*tempV1(~I(i,:)) - exp(-r*dt*i).*tempV2(~I(i,:));

    % For paths requiring sub-simulations we run these and compute their
    % Monte Carlo estimates for the continuation values.
    means = zeros(1,subloopsNeeded);

    for n=1:subloopsNeeded
        % For each path that requires sub-simulations we launch N3
        % sub-paths and compute the continuation values through a one-step
        % expectation computation.
        % Generate sub-paths one step ahead.
        Z = randn(1,N3);
        Z = [Z,-Z]; % create antithetic pairs
        subS = relaventTimePaths(1,n).*exp((r-s^2/2)*dt + s*Z*sqrt(dt));

        % Exercise values.
        subH = max(K-subS,0);
        % Cancellation values.
        subG = subH + delta;

        if i==d-1
            % If at the penultimate time step then one step ahead is
            % maturity where there is no continatuation so set the
            % expectation as the exercise value.
            means(1,n) = mean(subH);
        else
            % If not at the penultimate time step then find the
            % continuation values.

            % CHOOSE either basis polynomials or choice functions.
            %subD = generateBasisFunctions(subS,M);
            subD = generateChoiceFunctions(subS,M,K,(d-i-1)*dt,r,s);

            subC = (subD*beta(i+2,:)')'; % approximated continuation values
            % Discounted option value.
            subV = exp(-r*dt*(i+1)).*min(subG,max(subH,subC));
            means(1,n) = mean(subV); % monte carlo estimate
        end
    end

    % Calculate martingale difference for paths that required
    % sub-simulations.
    diff(I(i,:)) = exp(-r*dt*(i+1)).*tempV1(I(i,:))- means;

    % Update martingale.
    mart(i+1,:) = mart(i,:) + diff;
end
clear tempV1 tempV2 diff subD subC subH subV relaventTimePaths timePaths;

% Modify all the exercise/cancellation values so they are discounted.
for i = 1:d
    h(i,:) = exp(-r*dt*i).*h(i,:);
    g(i,:) = exp(-r*dt*i).*g(i,:);
end

```

```
%% Calculate Path-Wise Approximation
% Calculate R(s,t) and M(s,t) for each time point t = 1,...,d
% and s = 1,...,d
Rt = zeros(d,2*N2);
Rs = zeros(d,2*N2);
for s = 1:d
    % At each time step we set the immediate option value and martingale
    % value.
    martTemp = mart;
    Rt(1:s,:) = h(1:s,:); % equals t times as t<= s

    if s~= d
        % If not at maturity then let the future times as the stopped
        % cancellation value and stop the martingale.
        for j = s+1:d
            Rt(j,:) = g(s,:);  % now t > s, so stopped at s
            martTemp(j,:) = martTemp(s,:); % stop martingale
        end
    end
    tempDiff = Rt - martTemp; % find differences
    tempTMax = max(tempDiff); % take the maximum difference for each path
    Rs(s,:) = tempTMax;
end

% Now take minimums.
pathMinimums = min(Rs);

% Compute Monte Carlo estimate.
martApproximation = mean(pathMinimums) + lsmvrApprox
martStdError = std(pathMinimums)/sqrt(2*N2)

%% Calculate Bounds
% Calculate R(s,t) and M(s,t) for each time point t = 1,...,d but this time
% stopping at the approximate exercise times.
R = zeros(d,2*N2); % payoff matrix for upper bound, no time 0
R2 = zeros(d,2*N2); % payoff matrix for lower bound, no time 0
mart2 = mart; % create a copy to use for lower bound

for n = 1:2*N2
    % For each sample path stop the processes at the approximate sigma.
    sindx = sigma(n); % approximate sigma
    if sindx < d
        % Stop martingale at sigma.
        mart(sindx+1:d,n) = mart(sindx,n);
        % Stop payoff process at sigma.
        R(sindx+1:d,n) = g(sindx,n);
    end
    % Set payoff process as exercise price before sigma.
    R(1:sindx,n) = h(1:sindx,n);

    % For each sample path stop the processes at the approximate tau.
    tindx = tau(n); % approximate tau
    if tindx < d+1
        % Stop martingale at tau.
        mart2(tindx:d,n) = mart2(tindx,n);
        % Stop payoff process at tau.
        R2(tindx:d,n) = h(tindx,n);
    end
    % Set payoff process as cancellation price before tau.
    if tindx > 1
        R2(1:tindx-1,n) = g(1:tindx-1,n);
    end
end

% Calculate upper bound.
diff = R - mart;
maximums = max(diff); % maximum difference for each path
upperBound = mean(maximums) + lsmvrApprox % monte carlo estimate
upperStdError = std(maximums)/sqrt(2*N2)

% Calculate lower bound.
diff2 = R2 - mart2;
minimums = min(diff2); % minimum difference for each path
lowerBound = mean(minimums) + lsmvrApprox % monte carlo estimate
lowerStdError = std(minimums)/sqrt(2*N2)

difference = upperBound - lowerBound

% Construct confidence interval.
alpha = 0.05;
z = norminv(1-alpha/2);
CIlower = lowerBound - z*lowerStdError;
CIupper = upperBound + z*upperStdError;
CI = [CIlower,CIupper]

totaltime = toc

end
```