The Direct Control and Penalty Methods for American Put Options

by

Ama Peprah Asare

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computational Mathematics

Supervisors: Prof. Peter Forsyth and Prof. George Labahn

Waterloo, Ontario, Canada, 2013

Abstract

Pricing American options gives rise to nonlinear Hamilton-Jacobi-Bellman (HJB) Partial Differential Equations (PDEs). These HJB PDEs can be solved using a penalty or a direct control approach. In this essay, we explore the characteristics of these two methods — the direct control method and penalty method, under the American put option pricing domain. We also examine a third method, a combination of the best properties of the above mentioned methods. Included are algorithms and numerical results to support our findings. In all cases, we use a positive coefficient discretization of the PDE to ensure the numerical scheme converges to the viscosity solution.

Acknowledgements

It is my pleasure to thank all the people who have made this work possible.

My heartfelt appreciation goes to my supervisors, Peter Forsyth and George Labahn for their useful comments, guidance, intellectual and financial support, through the writing of this work. Especially for their impeccable attention to detail, which has made this essay much better than it would have been otherwise, I am deeply grateful.

To Yuying Li, I say thank you for the corrections and all the Computational Finance classes.

I am also particularly appreciative of the support of Justin Wan, the current Director of the Centre of Computational Mathematics, University of Waterloo, for the helpful discussions and great advice.

In the same vein, I will like to thank Anthea Dunne, the Administrative Officer for the Computational Mathematics program, for the awesome get-togethers and her resourcefulness.

For the intellectual and emotional support from all the wonderful classmates I have had in the Computational Mathematics program, I am very thankful.

To the professors I have had the pleasure of studying under at the University of Waterloo, I want to say thanks for the educational and life lessons I have had. I will surely put them into practice.

To my family, by blood and by choice all over the world, and to all the kind people I have met whose actions and words have given me encouragement to carry this work through, I want to say thank you.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

A financial option is a contract that entitles its holder to buy or sell an asset under certain conditions at a future time. This asset, usually a stock, is referred to as an underlying asset or simply an underlying. Options may be classified based on the rights a holder has and/or when they can be exercised. For example, a put option is a contract that gives the holder the right but not the obligation to sell an underlying at a specific price, known as a strike price. The writer, the buyer of the option, is obliged to buy the underlying at the specified strike price. A call option, on the other hand, is a contract that gives a holder the right but not the obligation to buy an underlying at a specified strike price. The writer, this time the seller, is obliged to sell the underlying at the specified strike price. An American option allows its holder to exercise anytime before or up to the specified expiration date. A European option, however, allows its holder to exercise an option only on the expiration or maturity date. The profit an option holder stands to make from exercising an option is called a payoff. This essay discusses the methods used to price the fair value of an American put option at initial time.

Valuation of financial options is of both theoretical interest and practical relevance. Consequently, considerable work has been done to find new approaches to solve for the pricing of options, in addition to continuously improve upon existing ones. Existing approaches include Monte Carlo simulations [11], lattice methods [17], and partial differential equation (PDE) approaches [24].

For an underlying asset, Monte Carlo simulations generate several possible random price paths forward in time. The value of an option is calculated by computing the present day value of the average of the payoffs calculated for each simulated path. This approach is advantageous when we want to handle multiple underlying assets. However, it cannot be easily used to handle an early exercise constraint. It also has the disadvantage of slow convergence to a solution for problems it can handle.

For pricing an option using a lattice method, the duration from the initial time to the option's maturity date is split into discrete intervals (time discretization) known as time steps, and possible stock prices are calculated at each discrete time step, over a lattice. Then working backwards from the expiry date, the option prices at each point on the lattice are calculated until initial time. The option price calculated for initial time is the value of the option. It may seem that with this method, the entire lattice will need to be stored in memory. But this is not the case, as calculating the price of an underlying depends only on the price of the underlying in the previous time step, and the option values, on the underlying price and option values at the next time step. This approach is easy to implement for simple cases such as pricing a European option, but may become complex for more complicated cases. The number of computations and approximation errors grow exponentially with increasing number of time steps, and as a result, the lattice method can become computationally expensive and numerically unstable unless the number of time steps is restricted [24].

The price of an option over time may be posed as a partial differential equation (PDE). The PDE approach may be used for many different types of options.To get the price of an option given a PDE, the latter needs to be discretized, i.e. converted into a set of discrete algebraic equations. There are several techniques that may be used to discretize a PDE. These include finite difference, finite volume, and finite element methods. Finite difference methods are more popular, mainly because they are relatively easy to implement, and allow for easy calculation of hedging parameters, which are essential for risk-managing financial derivatives [5]. After discretization, the equations are solved iteratively over discrete time intervals in order to arrive at the option price. Our work in this essay focuses on solving for the price of an American option using a PDE approach with finite difference discretization. Under this approach, we will use two different means to represent the option pricing problem, called the direct control and penalty methods.

1.1 Overview: The Direct Control and Penalty Methods

There are many models that have been used to represent the behavior of stock prices in the market today. The most popular of these is the Black-Scholes model and the methods under discussion in this essay will employ the famous Black-Scholes PDE, first developed in 1973 by Black and Scholes in "The Pricing of Options and Corporate Liabilities", to assist in solving the American option pricing problem. The Black-Scholes PDE cannot be solved as it is, but can be used in its closed form to solve for the price of a European option [24]. Such a closed-form solution generally does not exist for an American option. Instead, we use numerical methods, in this case, finite difference methods, to approximate the price of an American-type option. An American option is more difficult to price than its European counterpart, mainly because of its early exercise feature.

The early exercise constraint is an important factor to be taken into account when pricing American options. For an American call paying no dividends, the option value is the same as that of a European call on the same underlying with the same maturity date [22]. Hence the optimal exercise policy here is to not exercise. However, for a put option, also paying no dividends, it may be optimal to exercise before expiry time and doing so will affect the value of the option. Therefore the optimal exercise policy must be known beforehand and taken into consideration when pricing American options.

To price an American option using the PDE approach, we pose the pricing problem as a differential linear complementarity problem[1] (LCP). This LCP is formulated as a Hamilton-Jacobi-Bellman (HJB) PDE, discretized and solved using fully implicit or Crank-Nicolson time stepping schemes, which are numerical schemes used to approximate partial derivatives. We will explore two methods for formulating the LCP, a direct control method and a penalty method.

The direct control method was originally proposed in [9] in the context of HJB PDEs and adapted to the case of an early exercise option in [15, 21]. Using a direct control method, the HJB PDE may be posed as an optimal control problem, and solved directly. This is done by discretizing the optimal control problem, and solving the resulting non-linear equations iteratively. The direct control method has the advantage of solving the exact discretized LCP, with second order convergence within a specified convergence tolerance.

The penalty method has been studied by many authors [10, 27, 5] and found to be a very efficient way to approximate the solution of an American option pricing LCP. In this approach, a penalty term is introduced in the LCP in order to enforce the early exercise constraint. The resulting problem is discretized and solved iteratively. This method can be used on any type of discretization for one or multidimensional problems, and on unstructured meshes. Quadratic convergence is observed for a reasonable penalty value as time steps and mesh size tend to zero, using a variable time step selector as proposed in [10], within a specified convergence tolerance. The addition of the penalty term, however,

[1]In general, given x, $y \in \mathbb{R}$, a linear complementarity problem is a problem of the form $x \geq 0$, $y \geq 0$; $x > 0 \Longrightarrow y = 0$, $y > 0 \Longrightarrow x = 0$ and $y = ax + b$ for some $a, b \in \mathbb{R}$.

introduces a penalty error in the solutions of the discretized PDEs and as a result, the American constraint is only satisfied approximately [10].

Earlier work done on the direct control and penalty methods have shown that the two are very competitive in terms of efficiency when solving for the price of an option. The question we will like to answer with this work is which of the two is more efficient and why, for the problems under consideration in this essay.

1.2 Objectives

In this essay, we discuss two methods used to solve the problem of the valuation of American options, namely the direct control and penalty methods, both of which fall under the PDE approach, with finite difference discretization. We present an iterative algorithm, which can be modified to work with either direct control and penalty approaches. Solving the American option pricing problem with these methods using our iterative algorithm, we observe the convergence rates associated with the methods, the maximum number of non-linear iterations required by each method to solve the problem, and the performance of the methods on a time-dependent American constraint.

The main results of this essay are:

- The direct control method can be modified to make it more efficient than it already is.
- The penalty method is superior to the direct control method in converging to a solution.

The essay is arranged as follows: Chapter 2 presents the American option pricing problem mathematically, and defines models and formulas, including direct control and penalty forms of the problem, that will be used in the rest of the essay. Following this, we discretize the equations from Chapter 2 in Chapter 3, then describe an algorithm which will be used to solve the discretized equations in Chapter 4. Chapter 5 presents some numerical results. We show how the direct control method can be made more efficient in Chapter 6, and summarize our work in Chapter 7.

Chapter 2

Model

In this chapter, we present the mathematical models and formulae which will lay a foundation for the rest of the work in this essay.

Let S be the price of an underlying risky asset which we assume follows a stochastic process in time t,

$$dS = \mu S dt + \sigma S dZ, \qquad S \in (0, \infty) \tag{2.1}$$

where dZ is the increment of a Weiner process, μ is the drift, and σ is the volatility.

We define $V(S,t)$ as the price of an option at t when the market price is S, for some function $V : (0,\infty) \times [0,T] \to \mathbb{R}$, where T is the expiry date of the contract. Applying Ito's Lemma [22] to equation (2.1), leads to the Black-Scholes equation [22],

$$\frac{\sigma^2}{2} S^2 V_{SS} + rSV_S + V_t = rV. \tag{2.2}$$

where r is the risk-free rate of return. Let $\tau = T - t$ be the time in the backward direction. Then V can be redefined in terms of S and τ as $V(S,\tau)$, and equation (2.2) can be rewritten as

$$\frac{\sigma^2}{2} S^2 V_{SS} + rSV_S - V_\tau = rV. \tag{2.3}$$

Equation (2.3) is the PDE which will aid us in solving for the price of an American put option. We prefer to use backward time, τ, when solving for the price of an American option, because then we can use as an initial condition to equation (2.3), the payoff,

$$V^* = \max(K - S, 0),$$

which occurs at time T or $\tau = 0$. K is the strike price.

At $\tau = 0$,

$$V(S, 0) = V^*.$$

2.1 The American Option Constraint and the LCP

To prevent arbitrage (risk-free gain) for an American option, we enforce the constraint $V \geq V^*$. If $V < V^*$, arbitrage can be easily realized by buying the option for V, exercising it by selling the underlying for K, and then repurchasing the underlying in the market for S, making a risk-free profit of $K - V - S$.

So with the constraint $V \geq V^*$ enforced, where the option value $V > V^*$, it is more profitable for the holder to sell than to exercise. There, V satisfies the Black-Scholes equation (2.3). Where $V = V^*$, the option should be exercised. At those points, V is not governed by the Black-Scholes equation (2.3). These constraints combined satisfy the inequality

$$\frac{\sigma^2}{2} S^2 V_{SS} + rSV_S - V_\tau - rV \leq 0. \tag{2.4}$$

Figure 2.1 shows the payoff, V^* for an underlying, and its option value, V, before expiry. We see in Figure 2.1 that the constraint $V \geq V^*$ is enforced. So until it is profitable to exercise, we have

$$\begin{aligned} V &\geq V^*, & (2.5)\\ \frac{\sigma^2}{2} S^2 V_{SS} + rSV_S - V_\tau - rV &= 0. & (2.6) \end{aligned}$$

When it is profitable to exercise we have,

$$\begin{aligned} V &= V^*, & (2.7)\\ \frac{\sigma^2}{2} S^2 V_{SS} + rSV_S - V_\tau - rV &\leq 0. & (2.8) \end{aligned}$$

Putting equations (2.5), (2.6), (2.7) and (2.8) together, we arrive at this partial differential linear complementarity problem (LCP):

$$\begin{aligned} V_\tau - \mathcal{L}\mathrm{V} &\geq 0, \\ (V - V^*) &\geq 0, \\ \text{and } \left[(V - V^* = 0) \right. &\text{ or } \left. (V_\tau - \mathcal{L}V = 0) \right], \end{aligned} \tag{2.9}$$

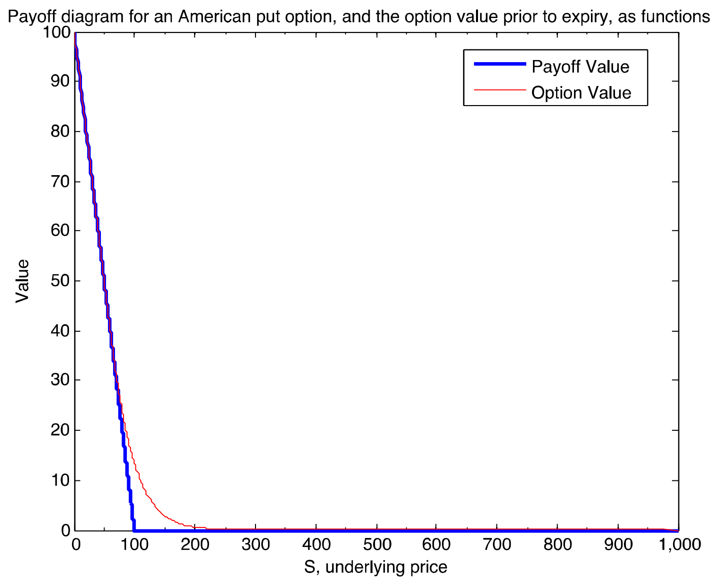

Figure 2.1: Payoff diagram for an American put option, illustrating the payoff, V^*, option value, V.

where $\mathcal{L}$ is a linear differential operator and

$$\mathcal{L}V \equiv \left(\frac{\sigma^2}{2}S^2V_{SS} + rSV_S - rV\right). \tag{2.10}$$

The LCP in (2.9) can also be formulated as the HJB equation

$$\min\left[V_\tau - \mathcal{L}V, V - V^*\right] = 0. \tag{2.11}$$

Initial and Boundary Conditions

The Black-Scholes equation has infinitely many solutions, so we impose some initial and boundary conditions to ensure we arrive at the viscosity solution.

For the American put option, our initial condition, as mentioned earlier, is

$$V(S,0) = \max\left(K - S, 0\right). \tag{2.12}$$

For the boundary conditions, as S, the price of the underlying approaches zero at any time τ, the option is likely to be exercised and so the option price, V approaches K. However, as S grows larger, the option is unlikely to be exercised and so V loses value and approaches zero. Therefore, we have the following boundary conditions:

Left-boundary condition:

$$\lim_{S\to 0} V(S,\tau) = K. \tag{2.13}$$

Right-boundary condition:

$$\lim_{S\to\infty} V(S,\tau) = 0. \tag{2.14}$$

Now, with HJB equation (2.11) defined, and initial and boundary conditions specified, we can define direct control and penalty forms of equation (2.11).

2.2 Direct Control Formulation

In control form, the HJB equation (2.11) can be written as

$$\max_{\varphi\in\{0,1\}} \left[\varphi\left(V - V^*\right) - (1-\varphi)\left(V_\tau - \mathcal{L}V\right)\right] = 0. \tag{2.15}$$

where φ is a control parameter. Equation (2.15), may be solved directly. Hence the name direct control. A value of $\varphi = 1$ implies early exercise is optimal. That of $\varphi = 0$ implies it is optimal to hold the option.

It is natural to scale a direct control formulation (2.15) of the HJB PDE (2.11). We do so to ensure that all the variables to be compared in equation (2.15) are of the same units. In addition, the use of an appropriate scaling factor allows for more accuracy in determining the optimal control, φ on a finite precision arithmetic machine [13]. Using Ω as a scaling factor, equation (2.11) becomes

$$\min\left[V_\tau - \mathcal{L}V, \Omega(V - V^*)\right] = 0,\ \Omega > 0, \tag{2.16}$$

and then is re-written in a non-linear direct control form as

$$\max_{\varphi\in\{0,1\}} \left[\Omega\varphi\left(V - V^*\right) - (1-\varphi)\left(V_\tau - \mathcal{L}V\right)\right] = 0. \tag{2.17}$$

2.3 Penalty Formulation

In penalized form, the HJB equation (2.11) may be written non-linearly as

$$\lim_{\varepsilon\to 0} \min[V_\tau - \mathcal{L}V, V - V^* + \varepsilon(V_\tau - \mathcal{L}V)] = 0. \tag{2.18}$$

The penalty form (2.18) of the LCP (2.9) is attained by replacing it with the non-linear PDE

$$V_\tau - \left(\frac{\sigma^2}{2} S^2 V_{SS} + rSV_S - rV\right) = \frac{\max\left(V^* - V, 0\right)}{\varepsilon}. \tag{2.19}$$

Adding the penalty term, $\frac{\max(V^*-V,0)}{\varepsilon}$ to equation (2.19) ensures that the solution satisfies $(V^* - V) \le \delta$ for $0 < \delta \ll 1$, as the positive penalty parameter $\varepsilon \to 0$. Essentially, where $V^* - V \le 0$, equation (2.19) resembles the Black-Scholes equation (2.3). However, where $0 < V^* - V < \delta$, the Black-Scholes inequality is satisfied, assuring the early exercise rule is not violated [5].

With a control φ, equation (2.18) can be written as

$$\lim_{\varepsilon \to 0} \left[V_\tau - \mathcal{L}V - \max_{\varphi \in \{0,1\}} \varphi \left(\frac{V^* - V}{\varepsilon} \right) \right] = 0. \tag{2.20}$$

In (2.20), a value of $\varphi = 1$, implies early exercise is optimal, and $\varphi = 0$ implies holding the option is optimal.

Typically we discretize equation (2.18). If we denote the discretization parameter by h, then it is common to choose $\varepsilon = O\left(h\right)$, which means that the penalty method is formally only first order convergent.

Chapter 3

Discretization

In this chapter, we transform equations (2.17) and (2.18) into a set of discrete non-linear algebraic equations, using finite difference approximations to the partial derivatives.

Let

$$S = \left\{S_1^n, S_2^n, \ldots, S_{i_{max}}^n\right\}, \qquad 0 \leq S_i \leq S_{i_{max}}, \tag{3.1}$$

be a finite grid of points to approximate the underlying,

$$\tau^n = n\Delta\tau,\ 0 \leq \tau^n \leq N\Delta\tau = T \tag{3.2}$$

be a set of discrete times, and

$$V^n = \left\{V_1^n, V_2^n, \ldots, V_{i_{max}}^n\right\} \tag{3.3}$$

be the option price, where

$$V_i^n = V\left(S_i, \tau^n\right). \tag{3.4}$$

Then we can approximate the partial derivatives in equation (2.3) by

$$\begin{aligned}
(V_\tau)_i^n &\simeq \frac{V_i^{n+1} - V_i^n}{\Delta\tau} &(3.5)\\
\left(\frac{\sigma^2 S^2 V_{SS}}{2}\right)_i^n &\simeq \left(\frac{\sigma^2 S^2}{2}\right)\left(\frac{\left(\frac{V_{i+1}^n - V_i^n}{S_{i+1}-S_i}\right) - \left(\frac{V_i^n - V_{i-1}^n}{S_i - S_{i-1}}\right)}{\frac{S_{i+1}-S_{i-1}}{2}}\right) &(3.6)\\
(rV)_i^n &\simeq rV_i^n &(3.7)
\end{aligned}$$

For the case of rSV_s, there are three ways to discretize. We can use either forward, backward or central differencing to yield

$$(rSV_s)_i^n \simeq rS_i\left(\frac{V_{i+1}^n - V_i^n}{S_{i+1} - S_i}\right), \text{ forward difference} \tag{3.8}$$

$$\text{or } (rSV_s)_i^n \simeq rS_i\left(\frac{V_i^n - V_{i-1}^n}{S_i - S_{i-1}}\right), \text{ backward difference} \tag{3.9}$$

$$\text{or } (rSV_s)_i^n \simeq rS_i\left(\frac{V_{i+1}^n - V_{i-1}^n}{S_{i+1} - S_{i-1}}\right), \text{ central difference.} \tag{3.10}$$

Central differencing gives the highest accuracy among the three. But it may not always be appropriate for discretizing equation (2.3), for reasons which we discuss in the next paragraph. Substituting equations (3.5), (3.6), (3.7) and one of (3.8), (3.9), (3.10), depending on the choice of differencing into equation (2.3) and rearranging, we arrive at

$$V_i^{n+1} = V_i^n\left(1 - (\alpha_i + \beta_i + r)\,\Delta\tau\right) + V_{i-1}^n \Delta\tau\alpha_i + V_{i+1}^n \Delta\tau\beta_i, \tag{3.11}$$

with

$$\alpha_i = \alpha_i^{central} = \frac{\sigma^2 S_i^2}{(S_i - S_{i-1})(S_{i+1} - S_{i-1})} - \frac{rS_i}{S_{i+1} - S_{i-1}} \tag{3.12}$$

$$\beta_i = \beta_i^{central} = \frac{\sigma^2 S_i^2}{(S_{i+1} - S_i)(S_{i+1} - S_{i-1})} - \frac{rS_i}{S_{i+1} - S_{i-1}} \tag{3.13}$$

or

$$\alpha_i = \alpha_i^{forward} = \frac{\sigma^2 S_i^2}{(S_i - S_{i-1})(S_{i+1} - S_{i-1})} \tag{3.14}$$

$$\beta_i = \beta_i^{forward} = \frac{\sigma^2 S_i^2}{(S_{i+1} - S_i)(S_{i+1} - S_{i-1})} + \frac{rS_i}{S_{i+1} - S_{i-1}} \tag{3.15}$$

or

$$\alpha_i = \alpha_i^{backward} = \frac{\sigma^2 S_i^2}{(S_i - S_{i-1})(S_{i+1} - S_{i-1})} - \frac{rS_i}{S_i - S_{i-1}} \tag{3.16}$$

$$\beta_i = \beta_i^{backward} = \frac{\sigma^2 S_i^2}{(S_{i+1} - S_i)(S_{i+1} - S_{i-1})}. \tag{3.17}$$

We select α and β such that the coefficients of V_i^{n+1}, V_i^n, V_{i-1}^n are always non-negative (positive coefficient discretization). We satisfy this constraint by choosing central differencing whenever possible, provided α and β are positive. Naturally, it would be advantageous to use central differencing always, since it gives second order accuracy, while forward or backward differencing gives only first order accuracy. However we cannot choose central differencing always, because α and β may not always be positive when central differencing is used. So at each node, we select a differencing method, using Algorithm 3.0.1, considering at each point, if we will obtain a positive coefficient or not. One of the central, forward or backward differences will always have α and β positive. The PDE can converge to many different solutions. Using a positive coefficient discretization ensures that the solution converges to the viscosity solution [10].

Algorithm 3.0.1 Positive Coefficient Selection Algorithm

```
if $\alpha_i^{central} \geq 0$ and $\beta_i^{central} \geq 0$ then
    $\alpha_i = \alpha_i^{central}$
    $\beta_i = \beta_i^{central}$
else if $\alpha_i^{forward} \geq 0$ and $\beta_i^{forward} \geq 0$ then
    $\alpha_i = \alpha_i^{forward}$
    $\beta_i = \beta_i^{forward}$
else
    $\alpha_i = \alpha_i^{backward}$
    $\beta_i = \beta_i^{backward}$
end if
```

With the terms we have just discretized, we proceed to discretize the direct control (2.17) and penalty (2.18) forms of equation (2.11).

3.0.1 Direct Control

Let $\mathcal{L}^h$ be the discrete form of the $\mathcal{L}$ operator and θ indicate the time stepping scheme to be used. Then the discretization of equation (2.17) gives

$$\begin{aligned}(1-\varphi_i^{n+1})\left(\frac{V_i^{n+1}}{\Delta\tau} - \theta\mathcal{L}_i^h V_i^{n+1}\right) &+ \Omega\varphi_i^{n+1}V_i^{n+1} \\ &= (1-\varphi_i^{n+1})\frac{V_i^n}{\Delta\tau} + \Omega\varphi_i^{n+1}V_i^* \\ &\quad +(1-\varphi_i^{n+1})(1-\theta)(\mathcal{L}_i^h V_i^n); \qquad i < i_{max} \quad (3.18)\\ \frac{V_i^{n+1}}{\Delta\tau} &= \frac{V_i^*}{\Delta\tau}; \qquad\qquad i = i_{max}\end{aligned}$$

with

$$\{\varphi_i^{n+1}\} \in \underset{\varphi\in\{0,1\}}{\arg\max} \left\{ \Omega\varphi(V_i^* - V_i^{n+1}) - (1-\varphi)\left(\frac{V_i^{n+1}-V_i^n}{\Delta\tau} - \theta(\mathcal{L}_i^h V_i^{n+1}) - (1-\theta)(\mathcal{L}_i^h V_i^n)\right)\right\}. \tag{3.19}$$

We use

$$\theta = \begin{cases} 1/2, & \text{for a Crank-Nicolson time stepping scheme} \\ 1, & \text{for a fully implicit time stepping scheme} \end{cases}.$$

Fully implicit schemes arise from the use of backward differences, and Crank-Nicolson schemes from central differences [24]. With second order accuracy, the Crank-Nicolson scheme is the most logical choice for discretization. However, previous work has shown that solutions of these discretized forms using a Crank-Nicolson scheme show spurious oscillations. However, by using a Rannacher modification [20], we are able to observe smoothness in the solution. The Rannacher modification consists of using, instead of a Crank-Nicolson scheme throughout the duration of the problem, a fully implicit scheme in the first two time steps. The variable θ enables us to easily switch between schemes.

Let M be a tridiagonal matrix with entries

$$[MV^n]_i = \alpha_i V_{i-1}^n - (\alpha_i + \beta_i + r)\, V_i^n + \beta_i V_{i+1}^n. \tag{3.20}$$

Then $\mathcal{L}V$ from equation (2.10) can be represented in discretized form as equation (3.20). Using this substitution, the matrix form of equation (3.18) can be written as

$$\left[(1-\varphi^{n+1})\left(\frac{I}{\Delta\tau} - \theta M\right) + \Omega\varphi^{n+1} I\right] V^{n+1} = (1-\varphi^{n+1})\left[\frac{I}{\Delta\tau} + (1-\theta)\, M\right] V^n + \Omega\varphi^{n+1} V^* \tag{3.21}$$

and equation (3.19) rewritten as

$$\varphi_i^{n+1} = \begin{cases} 1, & \text{if } (V^* - V^{n+1})_i\,\Omega > \left[MV^n - \left(\frac{I}{\Delta\tau} - \theta M\right)(V^{n+1} - V^n)\right]_i \\ 0, & \text{otherwise} \end{cases}. \tag{3.22}$$

I is an $i_{max} \times i_{max}$ identity matrix and V is a vector of size i_{max}.

3.0.2 Penalty Method

Let $\mathcal{L}^h$ be the discrete form of the $\mathcal{L}$ operator and θ indicate the time stepping scheme to be used. Then the discretization of equation (2.18) gives

$$\begin{aligned} \frac{V_i^{n+1}}{\Delta\tau} - \theta\mathcal{L}_i^h V_i^{n+1} + \frac{\varphi_i^{n+1}}{\varepsilon} V_i^{n+1} &= \frac{V_i^n}{\Delta\tau} + \frac{\varphi_i^{n+1}}{\varepsilon} V_i^* + (1-\theta)(\mathcal{L}_i^h V_i^n); && i < i_{max} \\ \frac{V_i^{n+1}}{\Delta\tau} &= \frac{V_i^*}{\Delta\tau}; && i = i_{max}. \end{aligned} \tag{3.23}$$

with

$$\{\varphi_i^{n+1}\} \in \underset{\varphi\in\{0,1\}}{\arg\max} \left\{ \frac{\varphi}{\varepsilon} \left(V_i^* - V_i^{n+1} \right) \right\}, \tag{3.24}$$

where ε represents the penalty parameter.

Again

$$\theta = \begin{cases} 1/2, & \text{Crank-Nicolson} \\ 1, & \text{fully implicit} \end{cases}.$$

Using M from equation (3.20), equation (3.23) can be simplified and written in matrix form as

$$\left[\frac{I}{\Delta\tau} - \theta M + \frac{\varphi^{n+1}}{\varepsilon} I \right] V^{n+1} = \left[\frac{I}{\Delta\tau} + (1-\theta)M \right] V^n + \frac{\varphi^{n+1}}{\varepsilon} V^* \tag{3.25}$$

and equation (3.24) as

$$\varphi_i^{n+1} = \begin{cases} 1, & \text{if } (V^* - V^{n+1})_i > 0 \\ 0, & \text{otherwise} \end{cases}. \tag{3.26}$$

I is an $i_{max} \times i_{max}$ identity matrix and V is a vector of size i_{max}.

Chapter 4

Solutions to the Discrete Equations

There is no analytical formula for the exact solution to the LCP in (2.9). We use the penalty and direct control formulations of the HJB equation (2.11) to numerically approximate the solution of the LCP in (2.9) using a policy iteration algorithm. The policy iteration algorithm used to solve discretized (2.11) starts with the payoff, V^*, as an initial state, and at each time step, repeatedly generates a guess of the optimal control policy and uses that guess in a linear system to solve for the option price until convergence. With the exception of the first time step which uses V^* as its initial state, all other policy iterations used within the lifetime of the option use as an initial state, the state from the previous iteration.

Formally put, policy iteration solves non-linear algebraic equations of the form

$$A\,(P)\,U = C(P),$$

with

$$P_i = \underset{P\in\{0,1\}}{\arg\max}\{-A(P)U + C(P)\}_i.$$

A is a matrix of size $i_{max} \times i_{max}$, and U, C are vectors of size i_{max}. P is a set of controls.

In the next sections, we define A, C, and P precisely for the direct control and penalty methods, and illustrate the policy iteration algorithm.

4.1 Direct Control method

Assuming we are at the kth iterative step of the policy iteration algorithm, during the nth discrete time step, we have for $i < i_{max}$,

$$\left[A\left(P^k\right)V\right]_i = (1-P_i^{k+1})\left[\left(\frac{I}{\Delta\tau}-\theta M\right)V\right]_i + \Omega P_i^{k+1}V_i. \tag{4.1}$$

$$C(P^k)_i = \left((1-P_i^{k+1})\left[\frac{I}{\Delta\tau}+(1-\theta)M\right]V^n\right)_i + \Omega P_i^{k+1}V_i^*. \tag{4.2}$$

$$P_i^{k+1} = \begin{cases} 1, & \text{if} \left(V_i^* - V_i^{k+1}\right)\Omega > \left(\left[MV^k\right]_i - \left(\left(\frac{I}{\Delta\tau}-\theta M\right)\left(V^{k+1}-V^k\right)\right)_i\right) \\ 0, & \text{otherwise} \end{cases}. \tag{4.3}$$

At $i = i_{max}$,

$$\left[A^kV\right]_i = \frac{V_{i_{max}}}{\Delta\tau}; \qquad C_i^k = \frac{V_{i_{max}}^*}{\Delta\tau}. \tag{4.4}$$

I is an $i_{max} \times i_{max}$ identity matrix, and M is as defined in (3.20). Ω is the scaling factor.

4.2 Penalty Method

Assuming we are at the kth iterative step of the policy iteration algorithm, during the nth discrete time step, we have for $i < i_{max}$,

$$\left[A\left(P^k\right)V\right]_i = \left[\left(\frac{I}{\Delta\tau}-\theta M\right)V\right]_i + \frac{P_i^{k+1}}{\varepsilon}V_i. \tag{4.5}$$

$$C(P^k) = \left[\left(\frac{I}{\Delta\tau}+(1-\theta)M\right)V^n\right]_i + \frac{P_i^{k+1}}{\varepsilon}V_i^*. \tag{4.6}$$

$$P_i^{k+1} = \begin{cases} 1, & \text{if } \left(V_i^* - V_i^{k+1}\right) > 0 \\ 0, & \text{otherwise} \end{cases}. \tag{4.7}$$

At $i = i_{max}$,

$$\left[A^kV\right]_i = \frac{V_{i_{max}}}{\Delta\tau}; \qquad C_i^k = \frac{V_{i_{max}}^*}{\Delta\tau}. \tag{4.8}$$

I is an $i_{max} \times i_{max}$ identity matrix, and M is as defined in (3.20). ε is the penalty parameter.

4.3 The Policy Iteration Algorithm

The following policy iteration algorithm, Algorithm 4.3.1 is used for both direct control and penalty method implementations, substituting the values for $A(P^k)$ and $C(P^k)$ from equations (4.1) and (4.2) for a direct control method, or (4.5) and (4.6) for a penalty method. The policy iteration algorithm is used within each discrete time step, until the LCP is solved.

Algorithm 4.3.1 Policy iteration

```
$k = 0$
$V^k = V$. $V$ is the solution vector from the previous time step, or the payoff for the first time step. $V$ is of size $i_{max}$.
while ($k \geq 0$) do
    $P_i^k \in \underset{P \in \{0,1\}}{\arg\max} \{-A(P)V^k + C(P)\}_i$
    Solve $A(P^k)V^{k+1} = C(P^k)$
    if $k > 0$ and $\max_i \frac{|V_i^{k+1} - V_i^k|}{\max\left[scale, |V_i^{k+1}|\right]} < tol$ then
        break
    end if
    $k = k + 1$
end while
```

The term *scale* used in Algorithm 4.3.1 is to ensure that unrealistic levels of accuracy are not enforced. Typically, for options valued in dollars, we use $scale = 1$.

If the difference between time steps is constant throughout the solution of the problem, we say we have used a policy iteration algorithm with *constant time steps*. Otherwise, we have used *variable time steps*, where the difference between time steps changes. In this essay, whenever we use variable time steps, we use the variable time step selector presented in [10]. Using this selector, we take small time steps where the solution, V, changes rapidly over the last two time steps, and larger time steps where the change between solutions is not great.

4.4 Effects of the Direct Control Scaling Factor and Penalty Parameter on the Solution to the Discrete Equations

4.4.1 Scaling Factor

Different choices of the scaling factor, Ω, will lead to different direct control forms, all converging to the same solution [14]. The size of Ω affects how fast the policy iteration algorithm, Algorithm 4.3.1 converges to a solution in finite precision arithmetic. The speed referred to here is measured in terms of the iterations per step, which is the number of iterative steps used by the policy iteration algorithm within each discrete time step.

Although Ω should only affect the speed of the algorithm for convergence to a solution, for sufficiently small Ω, the solution does not converge, as a result of using floating point arithmetic [15]. Ideally,

$$\Omega = \frac{1}{C\Delta\tau}, \tag{4.9}$$

with

$$C > \frac{2\delta}{tolerance}, \tag{4.10}$$

and

$$C < \frac{1}{\Delta\tau}\left(\frac{tolerance}{\delta}\right)\min_i\left(\frac{(\Delta S)_i^2}{2\theta S_i^2\sigma^2}\right). \tag{4.11}$$

$\delta \simeq 10^{-6}$ (double precision). *tolerance* is the desired level of accuracy. θ refers to either a fully implicit or Crank-Nicolson time stepping scheme [15].

The scaling factor, Ω, should have inverse units of time, so that the same units are being compared in equation (2.17).

4.4.2 Penalty Parameter

The number iterations per step used by Algorithm 4.3.1 when solving (2.18) is insensitive to the size of the penalty parameter, ε, unless the algorithm does not converge due to floating point error. However, a poor choice in penalty parameter, i.e. too large a value, will result in poor convergence as the number of cells used to approximate the grid S

approaches infinity and time steps are refined. This is because the introduction of the penalty parameter, $\varepsilon(V_\tau - \mathcal{L}V)$, in equation (2.18) also introduces a penalization error, which will be present at any finite grid size. Choosing too small a value can also introduce round-off error, which causes the penalty method to not converge.

Let $\varepsilon = C\Delta\tau$, where C is a dimensionless constant. Then using the bounds for C as presented in equations (4.10) and (4.11), the consistency error due to the penalty parameter, $\varepsilon(V_\tau - \mathcal{L}V)$, is small compared to the discretization error at reasonable grid sizes and time steps [15]. Using these bounds, convergence can be expected despite inexact arithmetic effects on the solution of equation (2.18).

Chapter 5

Numerical Results

This chapter outlines the numerical results obtained for pricing American put options using both direct control and penalty methods using the policy iteration algorithm, Algorithm 4.3.1. The convergence properties of both methods are investigated, and the effects of the scaling factor and penalty parameter as discussed in the previous chapter are demonstrated numerically. The results also include a comparison of the total iterations, maximum number of iterations out of all the iterations used within each time step, and the time used in solving for the price of standard and time-dependent American put options, using the direct control and penalty methods. This is to ascertain which of the methods is more efficient. The tests are carried out on MATLAB R2012b on an Intel Core i7 machine with processor speed 3.1 GHz and 16GB of RAM.

5.1 Standard American put option

A standard American put option is an American put option with payoff $\max(K - S, 0)$, where K is the strike price and S is the price of the underlying.

5.1.1 Convergence properties

Using the data in Table 5.1 and a non-uniformly spaced grid with properties in Table 5.2, we run Algorithm 4.3.1 for both methods, observing the differences in the option price attained for the two methods at $S = 100$, as the non-uniform grid S is refined, and their convergence ratios. Refinement of the grid is achieved by inserting new fine grids between each two coarse grid nodes. At each level of refinement, the time step control parameter is halved. We also observe the average number of iterations each method uses per time step, for each level of refinement.

The term convergence ratio or simply put, ratio, given a specific level of refinement, indicates the ratio between (a) the change in solution between the previous level of refinement and the one before it, and (b) the change in solution at the current level of refinement and the one before it.

Value refers to the option price realized.

Itn/Step refers to the average number of iterations used per time step.

Risk free rate, r	Volatility, σ	Maturity time, T	Strike, K	tol	$Smax$
0.02	0.2	0.25	100	10^{-6}	1000

Table 5.1: Data for American put.

Level of Refinement	S Nodes	Time steps
1	129	38
2	257	72
3	513	140
4	1025	276
5	2049	546
6	4097	1087
7	8193	2166

Table 5.2: Grid/time step data for convergence study, American put. On each grid refinement, new fine grids are inserted between each two coarse grid nodes, and the time step control parameter is halved.

We can see from Tables 5.3, 5.4, and 5.5 that the convergence ratios for the penalty method decrease as the penalty parameter, ε is increased. The decreasing convergence ratio can be explained by the penalization error, caused by the term $\varepsilon(V_T - LV)$ in the penalty formulation of the LCP in equation (2.18). As $\varepsilon \to 0$, the penalty form of the LCP is consistent with the LCP itself. But as ε increases, the penalization error increases, thus affecting the solution and the order of convergence.

Mathematically, the scaling factor does not affect the solution to the option pricing problem. As discussed in the previous chapter, we see from Tables 5.3, 5.4, and 5.5 that the

Level	Penalty Method			Direct Control Method		
	Value	Ratio	Itn/Step	Value	Ratio	Itn/Step
1	3.7665475159	-	2.66	3.7665475191	-	2.42
2	3.7678662573	-	2.71	3.767866259	-	2.50
3	3.7681993041	3.96	2.74	3.768199305	3.96	2.48
4	3.7682839352	3.94	2.71	3.7682839356	3.94	2.41
5	3.7683053232	3.96	2.53	3.7683053234	3.96	2.22
6	3.7683107358	3.95	2.16	3.768310736	3.95	2.06
7	3.7683120902	4.00	2.03	3.7683120911	3.99	2.01

Table 5.3: Convergence study, option value at S=100. $tol = 10^{-6}, \varepsilon = 1/\Omega = 10^{-6} \times \triangle\tau$.

Level	Penalty Method			Direct Control Method		
	Value	Ratio	Itn/Step	Value	Ratio	Itn/Step
1	3.7665149948	-	2.66	3.7665475191	-	2.42
2	3.7678484971	-	2.71	3.767866259	-	2.50
3	3.7681900567	3.90	2.71	3.768199305	3.96	2.48
4	3.7682793905	3.82	2.70	3.7682839356	3.94	2.41
5	3.7683031144	3.77	2.57	3.7683053234	3.96	2.22
6	3.7683096559	3.63	2.14	3.768310736	3.95	2.06
7	3.7683115727	3.41	2.03	3.7683120859	4.01	2.01

Table 5.4: Convergence study, option value at S=100. $tol = 10^{-6}, \varepsilon = 1/\Omega = 10^{-2} \times \triangle\tau$.

Level	Penalty Method			Direct Control Method		
	Value	Ratio	Itn/Step	Value	Ratio	Itn/Step
1	3.7636890939	-	2.63	3.7665475191	-	2.42
2	3.7663588379	-	2.69	3.767866259	-	2.58
3	3.7674436984	2.46	2.69	3.768199305	3.96	2.82
4	3.7679111774	2.32	2.61	3.7682839356	3.94	3.35
5	3.7681223967	2.21	2.21	3.7683053234	3.96	3.89
6	3.7682208685	2.14	2.06	3.7683106216	4.04	4.55
7	3.7682678192	2.10	2.00	3.7683114187	6.65	5.92

Table 5.5: Convergence study, option value at S=100. $tol = 10^{-6}, \varepsilon = 1/\Omega = 1 \times \triangle\tau$.

size of the scaling factor affects how fast the policy iteration algorithm, Algorithm 4.3.1 converges to a solution, in terms of the number of iterations used per time step.

We also see from Tables 5.3, 5.4, and 5.5 that the iterations per time step are generally lower for the direct control method, for a good choice of the scaling parameter. This indicates that the direct control method may be more efficient. We study the performance in detail in the next subsection.

5.1.2 Performance

Here, we observe the iterations per step used for both direct control and penalty methods in detail. We study the maximum and total number of iterations used by these methods to solve the American option pricing problem at different levels of refinement. Both constant and variable time steps are used on a non-uniform grid, densely populated in the neighborhood of K, for 7 levels of refinement. Data used to carry out these tests is given in Table 5.6.

M here refers to the total number of time steps, and N, the number of spatial nodes.

Max Itns refers to the maximum number of iterations used within a time step out of all the iterations per time step.

Total Itns refers to the total number of iterations used.

Elapsed Time is measured in seconds and is the total time used to solve for the price of an option at a given level of refinement.

A note of caution to the reader, concerning the elapsed time, is to not rely too heavily on the time used as a means to judge the performance of both methods, since the time used is dependent on the algorithm used, as well as the actual computer program used to run the algorithm, assuming both programs are run on the same machine. Concerning the algorithms, it is more expensive to compute the control for the direct control method (equation (4.3)), than it is for the penalty method (equation (4.7)). However, it is less expensive to solve the linear system within the policy iteration algorithm for the direct control method, since the matrix in (4.2) can be created so that it is more sparse than the matrix in (4.6). The elapsed time therefore can be used as a good measure of comparison between tests for the same method, but not so reliable a measure for tests between different methods, as the methods are implemented with different computer programs and with different controls and linear systems which require different computational times to solve. The value of the total iterations used is a better measure to be used for comparisons of the performances of both methods.

We observe from Tables 5.7 and 5.8 that the number of maximum iterations, total iterations, and time taken for the direct control and penalty methods increase with increasing

r	σ	T	$Smax$	tol	ε or $1/\Omega$	K
0.05	0.4	1	1000	10^{-8}	$10^{-6}\Delta\tau$	100

Table 5.6: Parameters for direct control and penalty methods, iterations study.

Direct Control Method	Max Itns	Total Itns	Elapsed Time
$M, N = 201$	35	455	0.03
$M, N = 401$	53	911	0.08
$M, N = 801$	80	1826	0.22
$M, N = 1601$	121	3655	0.67
$M, N = 3201$	183	7288	2.59
$M, N = 6401$	275	14549	9.36
$M, N = 12801$	411	28953	35.81
Penalty Method	Max Itns	Total Itns	Elapsed Time
$M, N = 201$	8	442	0.03
$M, N = 401$	13	894	0.07
$M, N = 801$	19	1805	0.21
$M, N = 1601$	28	3638	0.66
$M, N = 3201$	40	7319	2.57
$M, N = 6401$	59	14695	9.31
$M, N = 12801$	85	29303	34.12

Table 5.7: Maximum iterations study with constant time steps.

N. This suggests that the speed of convergence to a solution depends on N, the number of spatial nodes. The dependence of the speed of convergence on N can be seen in the total number of iterations, as they roughly increase by a factor of two, as the grid is refined and N is increased approximately by a factor of two. The time used for both methods does not roughly double, like N and the total number of iterations do, for the following reason: For a particular method, a different sparse linear system is solved for each iteration, depending on the control used (equation (4.3) or (4.7)). The time required to solve each sparse linear system depends on the number of nonzero elements, and so may vastly differ across iterations and time steps. As a result, the total time used may not be proportional to the total number of iterations.

It can also be seen that when both constant and variable time steps are used, the penalty method starts off with a lower number of total iterations, but as grid and time step size are

Direct Control Method	Max Itns	Total Itns	Elapsed Time
$M = 44,\ N = 201$	19	141	0.02
$M = 86, N = 401$	22	273	0.05
$M = 170, N = 801$	24	529	0.13
$M = 337, N = 1601$	27	1065	0.43
$M = 671, N = 3201$	29	2087	1.77
$M = 1337, N = 6401$	32	4118	6.30
$M = 2669, N = 12801$	34	8111	25.18
Penalty Method	Max Itns	Total Itns	Elapsed Time
$M = 44,\ N = 201$	5	137	0.02
$M = 86, N = 401$	6	283	0.05
$M = 170, N = 801$	7	577	0.13
$M = 337, N = 1601$	7	1165	0.44
$M = 671, N = 3201$	8	2342	1.81
$M = 1337, N = 6401$	8	4677	6.52
$M = 2669, N = 12801$	8	9227	26.00

Table 5.8: Iterations study. Variable time steps. On grid refinement, time step control parameter is halved, and new fine grids are inserted between each two coarse grid nodes. Initially, $dnorm = 0.2$, $\triangle\tau = 0.001$.

refined, the number of iterations used grow at a faster rate than that of the direct control method, resulting in a higher number of total iterations than the direct control method. Even though the penalty method uses more iterations than the direct control method for constant time steps, it uses less time. This behavior does not occur when variable time steps are used.

The maximum iterations used is a key element in finding out what happens to the total number of iterations in Tables 5.7 and 5.8 as the grid size increases. Using the direct control method, changing the data in Table 5.6, and the grid, by modifying its size and inter-node distances, numerical experiments reveal that the direct control method uses its highest number of iterations in its first time step. The penalty method does this at latter time steps, close to the end of the lifetime of the option.

The free boundary is the dividing point between the region to exercise and the region to hold the option. Both methods move one node per iteration [21] until the free boundary is reached. However, the direct control method always starts to move from the exercise boundary, the point on the grid where the strike price is, while the penalty method starts a little closer to the optimal solution for that time step. This difference in starting points is due to the set of controls used (equations (4.3) and (4.7)). As a result, the direct control method always uses more iterations in the first time step to arrive at the same intermediate solution as the penalty method. After that time step, both methods behave in a similar manner, arriving at the same intermediate solution at the end of each time step, with the direct control method sometimes using one less iteration than the penalty method. This explains why the penalty method uses a lesser number of maximum iterations than the direct control method, but may end up with a higher number of total iterations. Table 5.9 compares the number of iterations per time step, using data from Table 5.6, on the same grid used to carry out the tests in this subsection.

Time step #	1	2	3	4	5	6	7	8	9	10	11	12	13	...	168	169	170
Direct control	24	4	3	4	5	6	6	6	5	6	6	5	6	⋯	2	3	2
Penalty	5	4	3	4	6	6	6	6	6	7	6	6	7	⋯	2	3	2

Table 5.9: Iterations per step with strike, $K = 100$, $N = 801$. Variable time steps used.

Clearly, from Tables 5.7 and 5.8, the use of variable time steps is more advantageous in comparison to that of constant time steps. It requires a smaller number of iterations and uses less time. For this problem, using variable time steps, we can say that the direct control method is more efficient than the penalty method.

5.2 Time-dependent American constraint

The numerical examples so far have used a fixed value for the strike price K, for the American constraint. We refer to this as a standard American constraint. A time-dependent American constraint is one in which the strike price, K, is a function of time. For this constraint, we change Algorithm 4.3.1 slightly by using within our while loop, values of $K(\tau)$ between $K(0) = 100$ and $K(T) = 200$, depending on the time left until the end of the algorithm. Given τ and T, we approximate $K(\tau)$ using linear interpolation.

The next set of tests, using the above constraint, simulate the practical problem of pricing a convertible bond with accrued interest effects [1, 12]. We test the performance of the direct control and penalty methods for the simulations.

5.2.1 Performance

The non-uniform grid from the previous section is modified for the tests in this section, by making it densely populated in the neighborhood of the possible strike price values, $100 \leq K(\tau) \leq 200$. Data used to carry out these tests is given in Table 5.10.

M refers to the total number of time steps and N, the number of spatial nodes.

Max Itns refers to the maximum number of iterations used within a time step out of all the iterations per time step.

Total Itns refers to the total number of iterations used.

Elapsed time is measured in seconds and is the total time used to solve for the price of an option at a specified level of refinement.

r	σ	T	$Smax$	tol	$\varepsilon=1/\Omega$	$K(\tau)$
0.05	0.4	1	2000	10^{-8}	$10^{-6}\Delta\tau$	$K(0) + \frac{\tau}{T} \times (K(T) - K(0))$

Table 5.10: Parameters: direct control and penalty methods, time-dependent American constraint study.

We see from Tables 5.11 and 5.12 that using constant time steps, the penalty method uses a lower number of total iterations and time than the direct control method.

For this constraint the direct control and penalty methods do not always move one node per iteration until convergence. There is a point after which they both move one node per iteration until the free boundary is reached. We call the nodes from that point the path to the free boundary. Before that point, the direct control mostly moves one node per iteration. The penalty method may move one node per iteration, or even more, especially for large grid sizes. As a result, the penalty method is more likely to use a lesser number of total iterations than the direct control method does.

We realize through the numerical experiments conducted on this problem that the penalty method is able to converge to a solution, much faster than the direct control method. It uses a lesser amount of total iterations and time.

5.3 Conclusion

For a standard American put option, the direct control method solves the discretized HJB equation using less iterations and time than the direct control method, especially when variable time steps are used.

Direct Control Method	Max Itns	Total Itns	Elapsed Time
$M, N = 201$	9	1205	0.11
$M, N = 401$	14	3003	0.34
$M, N = 801$	21	7676	1.27
$M, N = 1601$	31	20039	5.35
$M, N = 3201$	47	53376	28.69
$M, N = 6401$	69	144408	143.21
$M, N = 12801$	104	395182	731.11
Penalty Method	Max Itns	Total Itns	Elapsed Time
$M, N = 201$	7	1199	0.10
$M, N = 401$	9	2993	0.33
$M, N = 801$	12	7660	1.22
$M, N = 1601$	16	20014	5.18
$M, N = 3201$	21	53337	27.68
$M, N = 6401$	29	144344	138.00
$M, N = 12801$	39	395065	696.66

Table 5.11: Time-dependent American constraint study with constant time steps. Other parameters in Table 5.10.

The penalty method outperforms the direct control method, generally using lesser iterations and time when both constant and variable time steps are used for solving for the option price of a simulated convertible bond.

In [16], a semi-smooth Newton method for the underlying continuous variational inequality in a suitable function space is analyzed and superlinear convergence shown. Reisinger and Witte in [21] propose that direct control method in some sense is inherently discrete, and that the penalty method can be seen as a discretization of the iteration in [16] and so has a well-defined limit for vanishing grid size. This sheds some light on why the penalty method seems to converge faster than the direct control method, especially noticeable in the case of the time-dependent constraint, as the grid is refined.

Overall, the numerical performances of the methods on both problems are very similar for small grid sizes, but the distinction between their performances grows clearer as grid data and time step are refined. The direct control method has the advantage of solving the exact discretized LCP in equation 2.9, while the penalty method solves an approximation of the LCP, due to the addition of a penalty parameter. While this can be a disadvantage for the penalty method, the choice of a *good* penalty parameter can make the error introduced

Direct Control Method	Max Itns	Total Itns	Elapsed Time
$M = 31,\ N = 201$	14	247	0.03
$M = 68, N = 401$	19	695	0.09
$M = 141, N = 801$	26	1874	0.34
$M = 286,\ N = 1601$	37	5135	1.44
$M = 575, N = 3201$	51	14097	7.88
$M = 1154, N = 6401$	72	39084	39.58
$M = 2310, N = 12801$	101	108845	219.68
Penalty Method	Max Itns	Total Itns	Elapsed Time
$M = 31,\ N = 201$	14	242	0.02
$M = 68, N = 401$	19	689	0.09
$M = 141, N = 801$	26	1866	0.33
$M = 286,\ N = 1601$	37	5126	1.43
$M = 575, N = 3201$	50	14038	7.42
$M = 1154, N = 6401$	68	38432	37.59
$M = 2310, N = 12801$	86	103555	202.65

Table 5.12: Time-dependent American constraint study with variable time steps. On grid refinement, time step control parameter is halved, and new fine grids are inserted between each two coarse grid nodes. Initially, $dnorm = 0.2$, $\triangle\tau = 0.001$. Other parameters in Table 5.10.

by the addition of the penalty term smaller than the error introduced by discretization for typical grid sizes [15], so that the penalization error may have less impact on the solution. A good penalty parameter also causes the error due to round-off to be negligible. Using a good penalty parameter, the penalty method has the advantage of being able to locate the free boundary faster than the direct control method does.

With that being said, the author believes the use of the penalty method with a good penalty parameter, should be the preferred method.

Chapter 6

Proposal

In this chapter, we propose an alternative way of using the direct control method to solve for the price of an American option, which makes it slightly more efficient than it already is. This approach is a combination of the direct control and penalty methods and so we will call it a *hybrid* approach. The hybrid approach is also proposed in [21], where the similarities of the two methods are explored.

6.1 Approach

It has been mentioned earlier in this essay that when solving for the option price of a standard American put option, the penalty method, for the first iteration within the first time step, arrives at a solution closer to the optimal solution for that time step, than the direct control method, due to the different controls used (equations (4.3) and (4.7)). This gives it a good start, causing it to use less iterations in the first time step than the direct control method. After the first time step, both methods behave in almost the same manner. For the hybrid method, we propose that in that first iterative step of the first time step, we use the control for the penalty method and in the following time steps until the free boundary is found, we use the control for the direct control method. In doing so, we take advantage of the penalty method's good starting point.

The hybrid method should show an improvement over the direct control method, but the improvement is not expected to be very great, since the only modification is only made in the first time step of the direct control method.

6.2 Numerical Results

We solve for the price of an American option using the hybrid approach in this section. We observe the convergence rations, iterations and time used during the computation of the option prices, to see how much of an improvement the hybrid method is over the direct control method.

6.2.1 Convergence

Using data from Table 6.1, and a non-uniform grid with properties in Table 6.2 we study convergence behavior when the hybrid method is used.

The term ratio or convergence ratio, given a specific level of refinement indicates the ratio between (a) the change in solution between the previous level of refinement and the one before it, and (b) the change in solution at the current level of refinement and the one before it.

Value refers to the option price realized.

Itn/Step refers to the average number of iterations used per time step.

Risk free rate, r	Volatility, σ	Maturity time, T	Strike, K	tol	$Smax$
0.02	0.2	0.25	100	10^{-6}	1000

Table 6.1: Data for American put.

Level of Refinement	S Nodes	Time steps
1	129	38
2	257	72
3	513	140
4	1025	276
5	2049	546
6	4097	1087
7	8193	2166

Table 6.2: Grid/time step data for convergence study of the hybrid method, American put. On each grid refinement, new fine grids are inserted between each two coarse grid nodes, and the time step control parameter is halved.

Level	Value	Ratio	Itn/Step
1	3.7665475191	-	2.39
2	3.767866259	-	2.49
3	3.768199305	3.96	2.46
4	3.7682839356	3.94	2.40
5	3.7683053234	3.96	2.22
6	3.768310736	3.95	2.06
7	3.7683120911	3.99	2.01

Table 6.3: Hybrid Method. Convergence study, option value at S=100. $\varepsilon = 1/\Omega = 10^{-6} \times \triangle\tau$.

Level	Value	Ratio	Itn/Step
1	3.7665475191	-	2.39
2	3.767866259	-	2.49
3	3.768199305	3.96	2.46
4	3.7682839356	3.94	2.40
5	3.7683053234	3.96	2.22
6	3.768310736	3.95	2.06
7	3.7683120859	4.01	2.01

Table 6.4: Hybrid Method. Convergence study, option value at S=100. $\varepsilon = 1/\Omega = 10^{-2} \times \triangle\tau$.

Level	Value	Ratio	Itn/Step
1	3.7665475191	-	2.39
2	3.767866259	-	2.57
3	3.768199305	3.96	2.81
4	3.7682839356	3.94	3.34
5	3.7683053234	3.96	3.89
6	3.7683106216	4.04	4.54
7	3.7683114187	6.65	5.92

Table 6.5: Hybrid Method. Convergence study, option value at S=100. $\varepsilon = 1/\Omega = 1 \times \triangle\tau$.

We see from the results in Tables 6.3, 6.4, and 6.5 that modifying the direct control method results in a slightly lower average number iterations per step, thus improving upon the direct control method's performance in Tables 5.3, 5.4, and 5.5. The hybrid method, just like the direct control method still requires an appropriate scaling factor for faster convergence.

6.2.2 Performance

We study the performance of the hybrid method in greater detail here, for both standard and time-dependent American constraints using the grid with properties in Table 6.7 and data in Table 6.6.

Max Itns refers to the maximum number of iterations used within a time step out of all the iterations per time step.

Total Itns refers to the total number of iterations used.

Elapsed time is measured in seconds and is the total time used to solve for the price of an option at a specified level of refinement.

r	σ	T	$Smax$	tol	ε or $1/\Omega$	K
0.05	0.4	1	1000	10^{-8}	$10^{-6}\Delta\tau$	100

Table 6.6: Parameters for direct control and penalty methods, iterations study.

Level	S Nodes	Time steps	
		Standard Constraint	Time-dependent Constraint
1	201	44	31
2	401	86	68
3	801	170	141
4	1601	337	286
5	3201	671	575
6	6401	1337	1154
7	12801	2669	2310

Table 6.7: Grid/time step data for hybrid method, American put option. On each grid refinement, new fine grids are inserted between each two coarse grid nodes, and the time step control parameter is halved.

Table 6.8 shows a comparison of the maximum iterations, time and total iterations used for both standard and time-dependent constraints using a hybrid method. We see that this approach causes a reduction in the maximum number of iterations used in the case of a standard American constraint, and consequently, the total number of iterations, as expected. However, its impact on the time is barely seen, except on the finest of the grids. For the time-dependent constraint, the hybrid method barely shows an improvement, in both total iterations and time values. This is because unlike the standard constraint, the direct control method on the time-dependent problem does not use its maximum iterations in the first time step, neither is its number of iterations bounded above by that of the penalty method. So modifying its first time step does not address the issue of why the penalty method performs more efficiently for this problem.

The penalty method uses a lesser number of policy iterations for this problem because it finds the path to the free boundary faster than the direct control method, by sometimes moving more nodes per iteration than the direct control method for each time step. The hybrid method will only work if applied to the first k iterations in each time step, where k is the point after which both methods move one node at a time. For the problem of the standard put, we realized that $k = 1$. However for the standard constraint, k is unknown and varies from one time step to the next. For this reason, applying the hybrid method is does little for this problem.

The hybrid method for practical purposes may be used to yield higher efficiency, better than that of the direct control and penalty methods for a standard American put option, but for a time-dependent American put option, is of little use, as it still performs worse than the penalty method. However, where the direct control method must be used, we suggest the use of the hybrid method instead, as it gives some improvement over the direct control method, and is easy to implement.

Level	Standard constraint					
	Direct Control			Hybrid		
	Max Itns	Total Itns	Elapsed Time	Max Itns	Total Itns	Elapsed Time
1	19	141	0.02	5	127	0.02
2	22	273	0.05	6	256	0.05
3	24	529	0.13	6	510	0.13
4	27	1065	0.43	7	1043	0.43
5	29	2087	1.77	7	2063	1.75
6	32	4118	6.30	8	4091	6.25
7	34	8111	25.18	8	8081	24.72

Level	Time-dependent constraint					
	Direct Control			Hybrid		
	Max Itns	Total Itns	Elapsed Time	Max Itns	Total Itns	Elapsed Time
1	14	247	0.03	14	242	0.03
2	19	695	0.09	19	689	0.09
3	26	1874	0.34	26	1866	0.34
4	37	5135	1.44	37	5126	1.44
5	51	14097	7.88	51	14087	7.78
6	72	39084	39.58	72	39072	38.54
7	101	108845	219.68	101	108832	219.11

Table 6.8: Comparison of iterations and elapsed time (seconds) for the direct control and hybrid methods for a standard and time-dependent American constraint. Variable time steps used.

Chapter 7

Summary

Many methods have been developed to price American options today. The direct control and penalty methods are examples of these methods which have been developed. They use a partial differential equation approach to price American options, with a finite difference discretization. In this paper, the efficiency of the direct control and penalty methods used to price American put options were numerically compared. We used a Black-Scholes model to represent the behavior of stock prices in the market, and a linear complementarity problem (LCP) to represent the option pricing problem, using the Black-Scholes PDE. This LCP was written in Hamilton-Jacobi-Bellman direct control and penalty forms, discretized and solved using a policy iteration algorithm.

To compare the efficiency of the methods, we studied the number of iterations and total time required for convergence. The convergence of the methods were studied to compare the solutions the methods arrived at for the option pricing problem and how fast the methods converged to a solution. This was done for various grid and time step refinements. The performance of the direct control and penalty methods were observed, implementing them for two different problems: a standard American put option and a simulated convertible bond.

Numerical tests revealed that using variable time steps with the policy iteration algorithm, the direct control method is more efficient than the penalty method for a standard American put option, taking into consideration the number of iterations and time it takes to solve the discretized HJB equation iteratively. For a simulated convertible bond (time-dependent constraint), the penalty method performs more efficiently.

We also exploited the similarities of the direct control and penalty methods, to improve on the direct control method, calling this improved method a hybrid method. The hybrid method was found to be slightly more efficient than the direct control method, using numerical tests but not good enough to outperform the penalty method on the time-dependent constraint.

The work done in this essay has shown that penalty method's mechanism for determining the free boundary works faster than that of the direct control's. Although the direct control method works more efficiently for the standard American put option, with a good penalty parameter, the penalty method performs almost as well, and with its good mechanism for finding the free boundary, as demonstrated for the time-dependent option, the author recommends the penalty method as the method of choice. Where the direct control method must be used, the author recommends the use of the hybrid method instead.

References

[1] E. Ayache, P.A. Forsyth, and K.R. Vetzal. Valuation of convertible bonds with credit risk. *The Journal of Derivatives*, 11(1):9–29, 2003.

[2] G. Barles. *Convergence of Numerical Schemes for Degenerate Parabolic Equations Arising in Finance Theory.* Numerical Methods in Finance (LCG Rogers and D. Talay, eds.), Publ. Newton Inst., Cambridge University Press, Cambridge, 1997.

[3] G. Barles and E. Jakobsen. Error bounds for monotone approximation schemes for parabolic Hamilton–Jacobi–Bellman equations. *Mathematics of Computation*, 76(260):1861–1893, 2007.

[4] G. Barles and E.R. Jakobsen. Error bounds for monotone approximation schemes for Hamilton–Jacobi–Bellman equations. *SIAM Journal on Numerical Analysis*, 43(2):540–558, 2005.

[5] C.C. Christara and D.M. Dang. Adaptive and high-order methods for valuing American options. *Journal of Computational Finance*, 14(4):73–113, 2011.

[6] C. Cryer. The solution of a quadratic programming problem using systematic overrelaxation. *SIAM Journal on Control*, 9(3):385–392, 1971.

[7] B.C. Dieffenbach. Black-Scholes option pricing model. `http://www.albany.edu/~bd445/Eco_466Y/Slides/Black-Scholes_Option_Pricing_Model.pdf`, 2013. Online, accessed on July 4th 2013.

[8] W.H. Fleming and H.M. Soner. *Controlled Markov Processes and Viscosity Solutions*, volume 25. Springer New York, 2006.

[9] P.A. Forsyth and G. Labahn. Numerical methods for controlled Hamilton-Jacobi-Bellman PDEs in finance. *Journal of Computational Finance*, 11(2):1–43, 2007/2008.

[10] P.A. Forsyth and K. Vetzal. Quadratic convergence for valuing American options using a penalty method. *SIAM Journal on Scientific Computing*, 23(6):2095–2122, 2002.

[11] M.C. Fu, S.B. Laprise, D.B. Madan, Y. Su, and R. Wu. Pricing American options: A comparison of Monte Carlo simulation approaches. *Journal of Computational Finance*, 4(3):39–88, 2001.

[12] A.J. Grau, P.A. Forsyth, and K.R. Vetzal. Convertible bonds with call notice periods. *Proceedings of Financial Engineering and Applications, Banff Canada*, 2003.

[13] Y. Huang, P.A. Forsyth, and G. Labahn. Methods for pricing American options under regime switching. *SIAM Journal on Scientific Computing*, 33(5):2144–2168, 2011.

[14] Y. Huang, P.A. Forsyth, and G Labahn. Combined fixed point and policy iteration for Hamilton–Jacobi–Bellman equations in finance. *SIAM Journal on Numerical Analysis*, 50(4):1861–1882, 2012.

[15] Y. Huang, P.A. Forsyth, and G. Labahn. Inexact arithmetic considerations for direct control and penalty methods: American options under jump diffusion. *Working Paper*, 2012.

[16] K. Ito and K. Kunisch. Semismooth newton methods for variational inequalities of the first kind. *ESAIM: Mathematical Modelling and Numerical Analysis*, 37(1):41–62, 3 2003.

[17] L. Jiang and M. Dai. Convergence of binomial tree methods for European/American path-dependent options. *SIAM Journal on Numerical Analysis*, 42(3):1094–1109, 2004.

[18] R.C. Klemkosky and B.G. Resnick. Put-call parity and market efficiency. *The Journal of Finance*, 34(5):1141–1155, 1979.

[19] H.J. Kushner and P.G. Dupuis. *Numerical Methods for Stochastic Control Problems in Continuous Time*, volume 24. Springer, 2000.

[20] R. Rannacher. Finite element solution of diffusion problems with irregular data. *Numerische Mathematik*, 43(2):309–327, 1984.

[21] C. Reisinger and J. Witte. On the use of policy iteration as an easy way of pricing American options. *SIAM Journal on Financial Mathematics*, 3(1):459–478, 2012.

[22] S.E. Shreve, P. Chalasani, and S. Jha. *Stochastic Calculus for Finance*, volume 1. Springer New York, 2004.

[23] E. Todorov. *Bayesian brain: Probabilistic Approaches to Neural Coding*, chapter 12, pages 269–298. MIT Press, 2007.

[24] P. Wilmott, J. Dewynne, and S. Howison. *Option pricing: Mathematical Models and Computation*, volume 935. Oxford financial press Oxford, 1993.

[25] J. Witte. *Numerical Solution of Discretised HJB Equations with Applications in Finance*. PhD thesis, Balliol College, University of Oxford, 2011.

[26] J. Witte and C. Reisinger. A penalty method for the numerical solution of hamilton-jacobi-bellman (HJB) equations in finance. *ArXiv e-prints*, August 2010.

[27] J. Witte and C. Reisinger. Penalty methods for the solution of discrete HJB equationscontinuous control and obstacle problems. *SIAM Journal on Numerical Analysis*, 50(2):595–625, 2012.