Numerical Comparisons of Iterative Methods for Pricing American Options Under Regime Switching

by

Josh Babbin

A research paper
presented to the University of Waterloo
in partial fulfillment of the
requirement for the degree of
Master of Mathematics
in
Computational Mathematics

Supervisors: Prof. Peter Forsyth and Prof. George Labahn

Waterloo, Ontario, Canada, 2011

I hereby declare that I am the sole author of this report. This is a true copy of the report, including any required final revisions, as accepted by my examiners.

I understand that my report may be made electronically available to the public.

Abstract

Pricing American options under a regime switching model requires the solution to a set of coupled nonlinear partial differential equation—variational inequalities. This paper explores three iterative methods used to solve these American option pricing equations. These include fixed-point policy, local policy, and global-in-time iteration methods. Numerical experiments were conducted to compare the iteration techniques using a direct control discretization for an American butterfly contract. Results demonstrate that the global-in-time method is outperformed by the two alternatives as grid and timestep refinements are made. Convergence inefficiencies are particularly prominent for longer duration contracts. Additionally, the global-in-time iteration requires a large amount of memory storage making it an unfavourable technique for pricing options under a regime switching model. The fixed-point method performs well requiring less computational work than both the local policy and global-in-time methods.

Table of Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

In their seminal 1973 paper, Fischer Black and Myron Scholes derived a partial differential equation for correctly pricing financial derivatives, now famously referred to as the Black-Scholes equation [6]. While this equation forms the foundation for much of the current financial mathematics research, the basic assumptions leading to its derivation have nonetheless been scrutinized extensively. In particular, practitioners have been critical of the following assumptions:

- Cash can be borrowed and lent at a constant risk-free interest rate.
- No transaction costs are incurred when buying or selling the underlying asset.
- The position in the underlying asset can be hedged continuously.
- There are no arbitrage opportunities.
- The underlying asset price movement is modeled by Geometric Brownian Motion (GBM) with constant drift and volatility.

Selling options at the Black-Scholes price and dynamically hedging the position can expose the writer to kurtosis, liquidity, and volatility risks, all a direct result of the assumptions above [16]. The fundamental assumption that the underlying asset price follows GBM does not accurately reflect what is observed in the real marketplace. The price jumps and fat-tails associated with historical stock returns cannot be replicated by a GBM model with constant volatility and drift [15]. As such, appropriate model selection for the underlying asset movement now plays a key role in the pricing of derivative contracts.

A popular alternative approach to Black-Scholes is to propose a more complicated model for the underlying price movement. The risk-neutral pricing equations can then be derived using an appropriate hedging portfolio. Some examples of these models include stochastic volatility, jump diffusion, and, in the case of this paper, regime switching.

A regime switching model defines a specified number of regimes/states, each with their own parameter set (interest rate, volatility, etc.) and governing stochastic process. Transitions between the regimes occur at a specified rate which corresponds to a jump in price and volatility. The model can be further complicated by allowing the parameters to vary with time and price to better fit market data. Regime switching is a practical alternative to the popular stochastic volatility, jump diffusion models because its model parameters can be easily interpreted by practitioners and it is computationally less expensive. Furthermore, regime switching models are capable of replicating the volatility smile phenomenon present in the Black-Scholes model [11].

Recently, regime switching models have found successful use in pricing assets in electricity markets, a day-ahead market characterized by seasonality effects, mean-reversion and (typically shortlived) jumps [5]. The 2010 daily electricity spot prices from the Nord Pool exchange can be seen in Figure 1. It has been shown that the seasonal spike intensity and consecutive spikes or price drops common to electricity spot prices can be effectively replicated with a Markov regime switching model [10]. A sample price path resulting from a two-state regime switching simulation showing short-lived prices spikes can be seen in Figure 2. Evidently, solving options pricing equations under a regime switching model accurately and efficiently is important in certain types of commodity markets.

Numerical Approach

Pricing an American option under a regime switching process requires the solution of a system of Hamilton-Jacobi-Bellman equations, typical of optimal control problems. Due to their nonlinear nature, discretizing and numerically solving these equations is a nontrivial matter. When implementing a numerical solver, one needs to ensure that the method converges to the financially meaningful solution, termed the viscosity solution [14]. Three discretizations of the pricing equations have been proposed in the literature including an explicit American constraint, a direct-control formulation, and a penalty method approach [9]. All three formulations have been shown to converge to the viscosity solution. Huang et al. [9] generalized this approach by constructing a framework to write the nonlinear algebraic equations resulting from an implicit-constraint discretization. This framework can be easily extended for solving any nonlinear algebraic equations that arise from

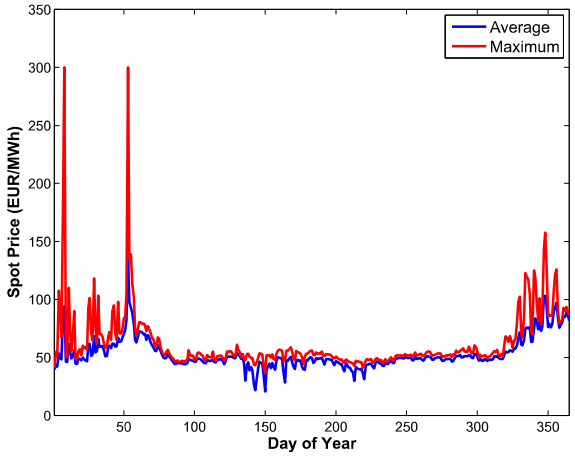

Figure 1: Average and maximum daily spot electricity prices in 2010 from the Nord Pool exchange. Large price fluctuations occur at an hourly rate as evidenced by the maximum daily values.

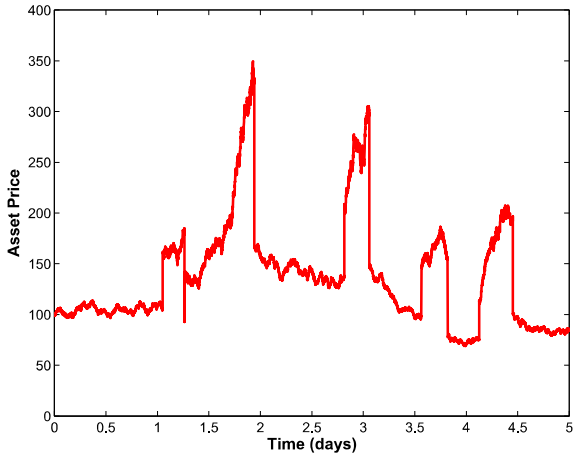

Figure 2: Two-state regime switching simulation showing price spikes.

a stochastic optimal control problem in finance. Four iterative solution methods used by various authors were consolidated and analyzed in the context of this framework. These include policy iteration, fixed-point policy iteration, local policy iteration, and global-in-time iteration methods.

The primary goal of this paper is to demonstrate that the global-in-time iteration is inefficient as compared to the fixed-point and local policy iterations. The global-in-time method, also known as iterated optimal stopping, has found use in solving impulse control problems [13]. Bayraktar [2] used the method in a proof related to the value of an American put option under a jump diffusion process. This method was then numerically implemented for pricing American options [3] and Asian options [4] under jump diffusion. The authors suggested that the iterative optimal stopping procedure is an efficient numerical scheme that converges uniformly and linearly, though they called this "exponentially fast". Le and Wang [12] later implemented this technique to numerically solve an optimal stopping problem with regime switching.

Subsequently, Huang et al. [9] demonstrated that the global-in-time solving method has a weaker convergence bound than the other policy iteration methods and also has hefty memory storage requirements, however, no numerical work has been done to verify that the global-in-time iteration performs better or worse than the alternatives. Thus, it is worthwhile to numerically investigate and compare the performance of this iteration algorithm to the others in order to conclusively establish its inferior performance.

Objectives

The main purposes of this paper are as follows:

- Outline the regime-switching model and show how it can be used to price options
- Perform numerical experiments comparing the efficiency of several iterative methods[1] consolidated in [9]
- Demonstrate through numerical experimentation that the global-in-time method is inferior to the proposed alternatives, thereby discounting its unnecessary use in future publications

[1]Numerical experiments are conducted using the fixed-point policy, local policy, and global-in-time iterations. These three methods are revisions of the full policy iteration also analyzed in [9].

Chapter 2

Regime Switching

This chapter outlines the development of the regime switching model leading to the equations used to price American options. A brief description of some financial terms and contracts referred to in this paper is provided (§2.1) followed by a description of the classic Black-Scholes model (§2.2). The regime switching model is covered in Section 2.3.

2.1 Basic Background

Arbitrage Opportunity

An arbitrage opportunity refers to an investment with a guaranteed profit in excess of the risk-free rate. That is, an investment portfolio with zero value today, zero probability of having a negative value in the future and a positive probability of having a positive value in the future. In classical finance theory, it is assumed that no arbitrage opportunities exist in the markets [16]. The principle of no-arbitrage is best summarized by the colloquial phrase "There's no such thing as a free lunch."

European Options

The simplest example of a financial derivative is a *European call option*, a contract that gives the holder the right, but not the obligation, to buy an asset at a specific *strike* price, K, at a **specified** *expiry time* in the future, T. A European *put option* gives the

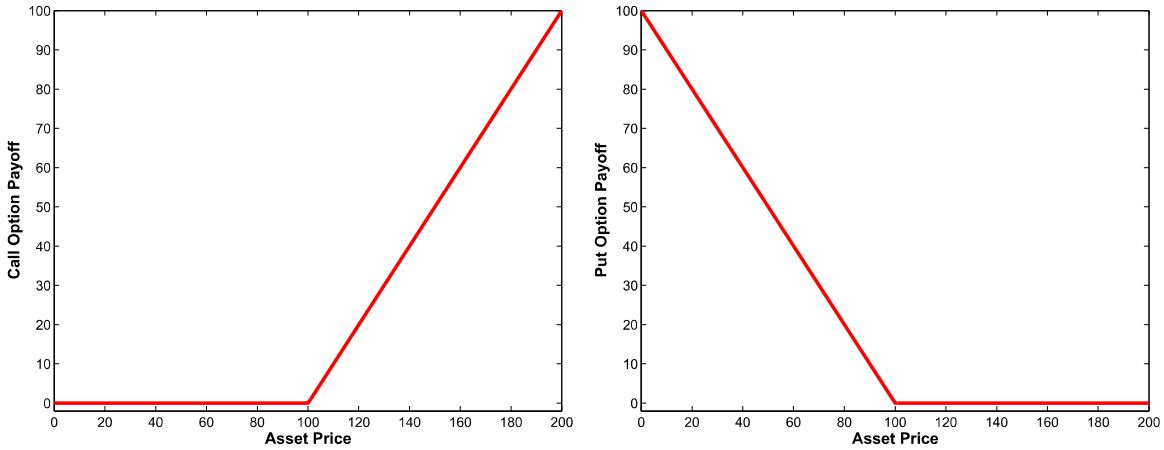

Figure 3: Payoff of a European call option (left) and put option (right), both with strike price $K = 100$.

holder the right, but not the obligation, to sell the asset at the specified strike and expiry. Mathematically, the payoff of a European option is given by

$$\text{Option Payoff} = \begin{cases} \max(S - K, 0) & \text{for call options} \\ \max(K - S, 0) & \text{for put options} \end{cases} \tag{2.1}$$

where S is the price of the underlying asset. The payoff is plotted in Figure 3.

American Options

An American option may be exercised at **any** time before the specified expiry date. The payoff received upon exercise is identical to the European payoff given by equation (2.1). An American option is never valued less than a European option with the same strike and expiry, otherwise one can make a riskless profit by purchasing the American version and writing a European contract, immediately pocketing the difference in price. Upon expiry, any money owed to the holder of the European contract can be paid by exercising the American option. Hence, in order to prevent an arbitrage, American options are always priced greater than or equal to their European counterpart.

2.2 Black-Scholes Model

2.2.1 European Options

Fischer Black and Myron Scholes derived a partial differential equation for pricing European options by assuming the underlying asset follows the lognormal stochastic process

$$dS = \mu\, S\, dt + \sigma\, S\, dZ \tag{2.2}$$

where S is the price of the underlying asset, μ is the drift, σ is the volatility, dt is the infinitesimal increment of time, and dZ is the increment of a Wiener process. The Wiener process is defined by

$$dZ = \phi\sqrt{dt} \tag{2.3}$$

where $\phi \sim \mathcal{N}(0, 1)$ is a random draw from a standard normal distribution.

A hedging portfolio, P, is constructed

$$P = V - \alpha S \tag{2.4}$$

consisting of a long position in one option of value V and a short position in α shares of the underlying asset. By applying Ito's Lemma to (2.2) and choosing $\alpha = \partial V/\partial S$, all risk associated with the randomness of the underlying can be eliminated from the value of the portfolio. That is, the change in portfolio value over an infinitesimal increment of time is deterministic. Therefore, by the no arbitrage principle, the portfolio should receive the risk free rate of return, r. This leads directly to the famous Black-Scholes equation

$$V_t + \frac{\sigma^2 S^2}{2} V_{SS} + rSV_S - rV = 0 \quad . \tag{2.5}$$

By defining the Black-Scholes operator, $\mathcal{L}$, as follows

$$\mathcal{L}[V] = \frac{\sigma^2 S^2}{2} V_{SS} + rSV_S - rV \tag{2.6}$$

and applying a transformation of variable $t \to \tau : \tau = T - t$, equation (2.5) can be rewritten

$$V_\tau = \mathcal{L}V \tag{2.7}$$

where τ represents time running backwards.

Together with the initial condition

$$V(S, \tau = 0) = \text{Option Payoff} \tag{2.8}$$

equation (2.7) can be solved to find the value of a European option. Note that an analytical solution to the Black-Scholes equation exists for a European option payoff.

2.2.2 American Options

If an American option is valued less than its payoff, an arbitrage opportunity will exist. In this case, one could purchase the option and immediately exercise, thereby making a riskless profit. To preclude this possibility, the option value must satisfy the additional constraint

$$V(S,\tau) \geq \text{Option Payoff} \tag{2.9}$$

at all times and for all asset prices.

Because the American option holder controls the early-exercise feature, the no-arbitrage portfolio in equation (2.4) may earn less than the risk-free rate of return if not exercised optimally. This means that the equality in equation (2.7) must be relaxed as follows

$$V_\tau \geq \mathcal{L}V \tag{2.10}$$

recalling that τ is time running backwards. Equations (2.9) & (2.10) can be combined to form a Hamilton-Jacobi-Bellman equation for pricing American options

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

where $V^* = V(S, \tau = 0)$ is the payoff at expiry. Equation (2.11) has no analytical solution so numerical techniques are required for pricing American options in the Black-Scholes construction.

2.3 Regime Switching Model

As discussed in Chapter 1, the GBM stochastic process does not capture the price dynamics observed in the financial marketplace. The GBM model can be extended by allowing discrete price jumps and volatility shifts to occur. In constructing a regime switching model consisting of K states, the following set of parameters are associated with each regime

- Discrete volatilities, $\sigma_j \quad j = 1, \ldots, K$
- Drift rates, $\mu_j \quad j = 1, \ldots, K$
- Transition probabilities, $\lambda_{jk} \quad j, k = 1, \ldots, K$
  - The probability of transitioning from regime $j \to k$

- Jump amplitudes, $\xi_{jk} \quad j,k = 1,\ldots,K$
  - Asset price jumps from $S \to \xi_{jk} S$ when transitioning from regime $j \to k$

A continuous Markov chain process is used to transition between any two states. The system of stochastic equations governing the regime switching process is a simple extension of equation (2.2) given by

$$dS = \mu_j\, S\, dt + \sigma_j\, S\, dZ + \sum_{k=1}^{K} (\xi_{jk} - 1)\, S\, dX_{jk} \qquad j = 1,\ldots,K \tag{2.12}$$

where dZ is the increment of a Wiener process as in equation (2.3) and

$$dX_{jk} = \begin{cases} 1 & \text{with probability} \quad \lambda_{jk}\, dt + \delta_{jk} \\ 0 & \text{with probability} \quad 1 - \lambda_{jk}\, dt - \delta_{jk} \end{cases} . \tag{2.13}$$

The transition probabilities are defined such that

$$\lambda_{jk} \geq 0 \qquad \text{if} \quad j \neq k$$

$$\lambda_{jj} = -\sum_{\substack{k=1 \\ k \neq j}}^{K} \lambda_{jk}$$

To prevent price jumps from occuring without a regime transition, we set $\xi_{jj} = 1$.

Analogous to the Black-Scholes development, a hedging portfolio is constructed as follows

$$P = -V_j + e\, S + \sum_{k=1}^{K-1} w_k\, F_k \tag{2.14}$$

where e is the amount held of the underlying asset with price S, w_k is the amount held of the additional hedging instruments with price F_k, and V_j is the no-arbitrage value of the option in regime j. Provided that the underlying asset and the additional hedging instruments form a non-redundant set, all risk can be eliminated through a dynamic hedging strategy [11].

Using Ito's Lemma and applying a no-arbitrage argument to the value of the hedging portfolio, it is possible to derive the pricing equations for a European option shown below

$$\frac{\partial V_j}{\partial \tau} = \mathcal{L}_j V_j + \lambda_j \mathcal{J}_j V_j \qquad j = 1,\ldots,K \tag{2.15}$$

where $\mathcal{L}_j$ is the analogous Black-Scholes operator in regime j, and $\mathcal{J}_j$ captures the impact of regime transitions on the option value. These operators are defined as

$$\mathcal{L}_j V_j = \frac{\sigma^2 S^2}{2}\frac{\partial^2 V_j}{\partial S^2} + (r - \rho_j)S\frac{\partial V_j}{\partial S} - (r + \lambda_j)V_j \tag{2.16a}$$

$$\mathcal{J}_j V = \sum_{\substack{k=1 \\ k\neq j}}^{K} \frac{\lambda_{jk}}{\lambda_j} V_k(\xi_{jk} S, \tau) \tag{2.16b}$$

where r is the risk-free rate and ρ_j, λ_j are defined by

$$\rho_j = \sum_{\substack{k=1 \\ k\neq j}}^{K} \lambda_{jk}(\xi_{jk} - 1)$$

$$\lambda_j = \sum_{\substack{k=1 \\ k\neq j}}^{K} \lambda_{jk} = -\lambda_{jj} \quad .$$

As described in Section 2.2.2, American options must satisfy the additional constraint in equation (2.9), as well as a modified version of equation (2.10) shown below

$$\frac{\partial V_j}{\partial \tau} \geq \mathcal{L}_j V_j + \lambda_j \mathcal{J}_j V \quad . \tag{2.17}$$

Imposing these constraints, the American option value is the solution to the set of Hamilton-Jacobi-Bellman type equations

$$\min\left[V_j - V^*, \quad \frac{\partial V_j}{\partial \tau} - \mathcal{L}_j V_j - \lambda_j \mathcal{J}_j V\right] = 0 \qquad j = 1, \ldots, K \tag{2.18}$$

where $V^* = V(S, \tau = 0)$ is the initial condition.

Remark. The pricing of a European option under a regime switching model requires the solution to a set of coupled partial differential equations, while an American option requires solving a nonlinear free-boundary optimization problem.

Chapter 3

Discretized Equations

In this chapter, the American option pricing equations (2.18) will be discretized for the purposes of implementing a numerical solving method on a computer.

3.1 Computational Domain

The partial differential equations are solved on a pricing grid consisting of a set of M nodes

$$\{S_1, S_2, \ldots, S_M\}$$

following a sequence of J discrete timesteps

$$\{\tau^0, \tau^1, \ldots, \tau^J\}$$

where $\tau^{n+1} = \tau^n + \Delta\tau^n$ such that $\Delta\tau^n$ is the time step size in going from $\tau^n \to \tau^{n+1}$. By convention, we set $\tau^0 = 0$ and $\tau^J = T$, recalling that τ^0 corresponds to the expiry time $t = T$, and τ^J corresponds to the start time $t = 0$. In general, the n^{th} time step is given by

$$\tau^n = \sum_{i=0}^{n-1} \Delta\tau^i \tag{3.1}$$

If equally spaced time steps of size $\Delta\tau$ are taken, equation (3.1) simplifies to

$$\tau^n = n\Delta\tau \quad . \tag{3.2}$$

The approximate solution at (S_i, τ^n) in regime j is denoted by $V_{i,j}^n$. A vector of solution values at all nodes in every regime at time level n is constructed as follows

$$V^n = [V_{1,1}^n, ..., V_{M,1}^n, \ldots, V_{1,K}^n, ..., V_{M,K}^n]' \tag{3.3}$$

where K is the number of regimes. The solution vector V^n has length $N = K \cdot M$.

For succinctness, a single row index will often be used to refer to entries in V^n as illustrated below

$$\begin{aligned} V_\ell^n &= V_{i,j}^n \\ \ell &= (j-1)M + i \quad . \end{aligned}$$

Hence V_{ij}^n may be interchanged with V_ℓ^n from here on in.

3.2 Discretized Operators

3.2.1 $\mathcal{L}$ Operator Discretization

The Black-Scholes operator in equation (2.16a) is discretized using central, forward or backwards differencing for the $\partial V_j/\partial S$ term to yield a positive coefficient discretization. Specifically, forward or backward differencing is used only at nodes for which central differencing does not yield positive coefficients. This ensures that the second-order central differencing approximation is used as much as possible. Denote $\mathcal{L}_j^h$ as the discrete form of the Black-Scholes operator $\mathcal{L}_j$. The general form of the discrete Black-Scholes operator is then given by

$$(\mathcal{L}_j^h V^n)_{ij} = \alpha_{i,j} V_{i-1,j}^n + \beta_{i,j} V_{i+1,j}^n - (\alpha_{i,j} + \beta_{i,j} + r + \lambda_j) V_{i,j}^n \tag{3.4}$$

where $\alpha_{i,j}$ and $\beta_{i,j}$ are selected using Algorithm 3.1 to ensure a positive coefficient discretization. The expressions for α and β using central, forward, and backward differencing are included in Appendix A.

3.2.2 $\mathcal{J}$ Operator Discretization

The regime-switching operator in equation (2.16b) is discretized through a linear interpolation approximation

$$(\mathcal{J}_j^h V^n)_{ij} = \sum_{\substack{k=1 \\ k \neq j}}^{K} \frac{\lambda_{jk}}{\lambda_j} \mathcal{I}_{i,j,k}^h V^n \tag{3.5}$$

Algorithm 3.1 Positive Coefficient Algorithm for $\mathcal{L}_j^h$ operator. The α and β terms are defined in Appendix A.

if $\alpha_{i,j}^{\text{central}} \geq 0$ **and** $\beta_{i,j}^{\text{central}} \geq 0$ **then**
    $\alpha_{i,j} = \alpha_{i,j}^{\text{central}}$
    $\beta_{i,j} = \beta_{i,j}^{\text{central}}$
else if $\alpha_{i,j}^{\text{forward}} \geq 0$ **and** $\beta_{i,j}^{\text{forward}} \geq 0$ **then**
    $\alpha_{i,j} = \alpha_{i,j}^{\text{forward}}$
    $\beta_{i,j} = \beta_{i,j}^{\text{forward}}$
else if $\alpha_{i,j}^{\text{backward}} \geq 0$ **and** $\beta_{i,j}^{\text{backward}} \geq 0$ **then**
    $\alpha_{i,j} = \alpha_{i,j}^{\text{backward}}$
    $\beta_{i,j} = \beta_{i,j}^{\text{backward}}$
end if

where

$$\mathcal{I}_{i,j,k}^h V^n = \begin{cases} V_{M,k}^n & \text{if } \xi_{jk} S_i > S_M \\ \omega V_{p,k}^n + (1-\omega) V_{p+1,k}^n & \text{otherwise} \end{cases} \tag{3.6}$$

where $\omega \in [0,1]$ is the linear interpolation parameter and indices p and $p+1$ correspond to the grid nodes adjacent to the jump price $\xi_{jk} S_i$. Ie. $S_p \leq \xi_{jk} S_i \leq S_{p+1}$. Note that in the above discretization, jumps which extend beyond the largest grid price S_M are approximated with that value. If S_M is chosen to be sufficiently large, the error introduced by this truncation approximation will be negligible near grid points of interest.

3.3 Direct Control Discretization

One can introduce a scaling factor, Ω, to the variational-inequality form of equation (2.18) as follows

$$\begin{aligned} \frac{\partial V_j}{\partial \tau} - \mathcal{L}_j V_j - \lambda_j \mathcal{J}_j V &\geq 0 \\ \Omega \left(V_j - V^* \right) &\geq 0 \\ \left(\frac{\partial V_j}{\partial \tau} - \mathcal{L}_j V_j - \lambda_j \mathcal{J}_j V \right) \Omega \left(V_j - V^* \right) &= 0 \quad . \end{aligned} \tag{3.7}$$

Equations (3.7) can then be combined to give the direct control formulation

$$\max_{\phi\in\{0,1\}}\left[\Omega\,\phi\,(V^*-V_j)-(1-\phi)\left(\frac{\partial V_j}{\partial\tau}-\mathcal{L}_jV_j-\lambda_j\mathcal{J}_jV\right)\right]=0\quad . \tag{3.8}$$

Observe that the equation above is simply a reformulation of the optimization problem in equation (2.18); However, the introduction of the scaling factor allows one to more accurately determine the optimal control on a finite precision machine. Comparison of the two terms in equation (3.8) in a floating point number system is aided by selection of an appropriate Ω value [9]. More intuitive reasoning for introducing Ω follows from the fact that a conversion factor is required to compare the two terms with differing units.

In discretizing the direct control form equations, a parameter θ is introduced such that

$$\theta=\begin{cases}1 & \text{for fully implicit discretization}\\ 0.5 & \text{for Crank-Nicholson discretization}\end{cases}\quad .$$

The discrete form of equation (3.8) is then written

$$\begin{aligned}(1-\phi_{i,j}^{n+1})\left(\frac{V_{i,j}^{n+1}}{\Delta\tau}-\theta\mathcal{L}_j^hV_{i,j}^{n+1}\right)+\Omega\,\phi_{i,j}^{n+1}V_{i,j}^{n+1}&\\ =(1-\phi_{i,j}^{n+1})\frac{V_{i,j}^{n}}{\Delta\tau}+\Omega\,\phi_{i,j}^{n+1}V_i^*+(1-\phi_{i,j}^{n+1})\lambda_j\theta[\mathcal{J}_j^hV^{n+1}]_{i,j}&\\ +(1-\phi_{i,j}^{n+1})(1-\theta)\big[\mathcal{L}_j^hV_{i,j}^n+\lambda_j[\mathcal{J}_j^hV^n]_{i,j}\big]\qquad & i<M\\ V_{i,j}^{n+1}=V_i^*\qquad & i=M\end{aligned} \tag{3.9}$$

where

$$\begin{aligned}\{\phi_{i,j}^{n+1}\}\in\arg\max_{\phi\in\{0,1\}}\Big\{\Omega\,\phi(V_i^*-V_{i,j}^{n+1})-(1-\phi)\Big(\frac{V_{i,j}^{n+1}-V_{i,j}^n}{\Delta\tau}&\\ -\theta\big(\mathcal{L}_j^hV_{i,j}^{n+1}+\lambda_j[\mathcal{J}_j^hV^{n+1}]_{i,j}\big)-(1-\theta)\big(\mathcal{L}_j^hV_{i,j}^n+\lambda_j[\mathcal{J}_j^hV^n]_{i,j}\big)\Big)\Big\}.&\end{aligned} \tag{3.10}$$

The set $\{\phi_{i,j}^n : i=1,\ldots,M; j=1,\ldots,K\}$ are the control parameters at every node in each regime at time level n. A value of 0 indicates exercise is *not* optimal while a value of 1 indicates early exercise is optimal. The equation above constitutes a set of coupled nonlinear algebraic equations.

Remark. Huang et al. [9] analyze two additional discretizations: an explicitly imposed American constraint, and a penalty method discretization. The explicit constraint approach results in a delta ($\partial V/\partial S$) that is not continuous across the early exercise boundary and will only achieve a first order convergence rate. Together, these issues make the explicit method undesirable. The penalty method, while unconditionally stable, has been criticized for introducing an additional source of error into the numerical solution [3]. The numerical work that follows in this paper will thus focus on implementations of the direct control method.

3.4 General Form of Equations

It will be advantageous to write equations (3.9) and (3.10) in a generalized matrix form to be solved computationally using one of the iterative algorithms of Chapter 4. A vector of control parameters for all nodes in each regime is defined by

$$Q = [\phi_{1,1}, ..., \phi_{M,1}, \ldots, \phi_{1,K}, ..., \phi_{M,K}]' . \tag{3.11}$$

The matrix form is structured as follows

$$\begin{aligned} A^*(Q)\, V^{n+1} &= C(Q) \\ \text{with } Q_\ell &= \arg\max_{Q \in Z} \Big[-A^*(Q) V^{n+1} + C(Q) \Big]_\ell \end{aligned} \tag{3.12}$$

where $Z = \{\phi \mid \phi \in \{0, 1\}\}$. The A^* matrix has dimensions $N \times N$ and V^{n+1} and C are vectors of length N. These are nonlinear discretized algebraic equations since both A^* and C are a function of the solution V^{n+1}.

The coefficients from the Black-Scholes operator contribute tridiagonal elements in A^* while the terms coupling different regimes contribute to off-tridiagonal elements. These off-tridiagonal elements add irregularities that break the nice tridiagonal sparsity pattern that would otherwise be present. Therefore, it is useful to separate the regime-switching coefficients by splitting the A^* matrix as follows

$$A^*(Q) = A(Q) - B(Q) \tag{3.13}$$

with $A(Q)$ containing the coefficients of nodes coupled within the same regime, while $B(Q)$ contains the coefficients that couple nodes in different regimes. This splitting will be useful when implementing numerical algorithms to solve the nonlinear equations.

The matrix coefficients for nodes $i < M$ can be easily deduced from equation (3.9) as shown below

$$\begin{aligned}
[A(Q)\,V^{n+1}]_\ell &= (1-\phi_\ell)\left(\frac{V_\ell^{n+1}}{\Delta\tau} - \theta\mathcal{L}_j^h V_\ell^{n+1}\right) + \phi_\ell\,\Omega\,V_\ell^{n+1} \\
[B(Q)\,V^{n+1}]_\ell &= (1-\phi_\ell)\lambda_j\theta[\mathcal{J}_j^h V^{n+1}]_\ell \\
[C(Q)]_\ell &= (1-\phi_\ell)\frac{V_\ell^n}{\Delta\tau} + \phi_\ell\,\Omega\,V_i^* \\
&\quad + (1-\phi_\ell)(1-\theta)\left[\mathcal{L}_j^h V_\ell^n + \lambda_j[\mathcal{J}_j^h V^n]_\ell\right]
\end{aligned} \tag{3.14}$$

The boundary condition at node $i = M$ requires that

$$\begin{aligned}
[A(Q)\,V^{n+1}]_\ell &= V_{M,j}^{n+1} \\
[B(Q)\,V^{n+1}]_\ell &= 0 \qquad\qquad \text{when } \ell = (j-1)M + M \\
[C(Q)]_\ell &= V_M^* \quad .
\end{aligned} \tag{3.15}$$

Chapter 4

Iterative Solving Methods

This chapter outlines the iterative algorithms that will be implemented to solve the non-linear algebraic equations (3.12). Three numerical algorithms will be discussed, all variants of Howard's policy iteration method [8] for dynamic programming problems. In the following sections, the notation $(V^n)^k$ is used to refer to the k^{th} iterate of the solution at timestep n.

4.1 Fixed Point Policy Iteration

The fixed point policy iteration takes advantage of the splitting in equation (3.13) to reduce the amount of computational work required to find a solution. The iteration method is outlined in Algorithm 4.1[1]. Observe that the splitting of the regime-coupling terms in line 4 results in a tridiagonal coefficient matrix. Thus, a sparse matrix algorithm can be used to efficiently solve the linear equations.

In [9], the authors demonstrate that an upper bound on convergence of the fixed point iteration is

$$\| A^{-1}(Q^k)B(Q^k) \|_\infty \leq \max \left[\max_j \frac{\theta\hat{\lambda}\Delta\tau}{1+\theta(r+\hat{\lambda})\Delta\tau}, \quad \max_j \frac{\lambda_j\theta}{\Omega} \right] \tag{4.1}$$

where $\hat{\lambda} = \max_j \lambda_j$.

[1]The *Scale* term in line 5 of the algorithm is used to prevent division by very small numbers. A value of Scale = 1 is typically used for dollar currencies.

Algorithm 4.1 Fixed Point Policy Iteration

1: $(V^{n+1})^0 = V^n \quad \rightarrow$ the solution vector at the previous time level

2: **for** $k = 0, 1, 2, \ldots$ until converged **do**

3: $\quad Q_\ell^k = \arg\max_{Q_\ell \in Z} \Big\{ -\big[A(Q) - B(Q)\big](V^{n+1})^k + C(Q) \Big\}_\ell$

4: $\quad$ Solve: $\quad A(Q^k)(V^{n+1})^{k+1} = B(Q^k)(V^{n+1})^k + C(Q^k)$

5: $\quad$ **if** $k > 0$ and $\max_\ell \dfrac{|(V_\ell^{n+1})^{k+1} - (V_\ell^{n+1})^k|}{\max\big[\text{Scale}, |(V_\ell^{n+1})^{k+1}|\big]} <$ Tolerance **then**

6: $\quad\quad$ Break from the iteration

7: $\quad$ **end if**

8: **end for**

Theorem 4.1. *If the direct control parameter Ω satisfies*

$$\Omega > \theta \cdot \hat{\lambda} \qquad \text{where } \hat{\lambda} = \max_j \lambda_j$$

then the fixed point policy iteration in Algorithm 4.1 converges.

Proof. Follows from the requirement that the convergence bound in equation (4.1) be less than one. □

4.2 Local Policy Iteration

In the local policy iteration the American control problem is solved with the regime coupling terms lagged behind one iteration. The method is outlined in Algorithm 4.2. Note that the local policy iteration can be thought of as a generalization of the fixed point iteration discussed in the previous section. Line 3 in Algorithm 4.2 requires solving a nonlinear equation through an iteration of $(V^{n+1})^{k+1}$. If this "inner" solve is terminated after one iteration, then the local policy method is equivalent to the fixed point method.

Theorem 4.2. *Define $E^k \equiv (V^{n+1})^k - \tilde{V}$ where $\tilde{V}$ is the solution to equation* (3.12)*. If the matrices A and B are given by equation* (3.14) *then the local policy iteration in Algorithm 4.2 converges at the rate*

$$\frac{\| E^{k+1} \|_\infty}{\| E^k \|_\infty} \leq \frac{\theta \hat{\lambda} \Delta\tau}{1 + \theta(r + \hat{\lambda})\Delta\tau} \quad . \tag{4.2}$$

Algorithm 4.2 Local Policy Iteration

1: $(V^{n+1})^0 = V^n \quad \rightarrow$ the solution vector at the previous time level

2: **for** $k = 0, 1, 2, \ldots$ until converged **do**

3: Solve: $\max_{Q_\ell \in Z} \left\{ -A(Q)(V^{n+1})^{k+1} + B(Q)(V^{n+1})^k + C(Q) \right\} = 0$

4: **if** $k > 0$ and $\max_\ell \dfrac{|(V_\ell^{n+1})^{k+1} - (V_\ell^{n+1})^k|}{\max\left[\text{Scale}, |(V_\ell^{n+1})^{k+1}|\right]} <$ Tolerance **then**

5: Break from the iteration

6: **end if**

7: **end for**

Proof. See [9]. □

4.3 Global-in-Time Iteration

A global-in-time method attempts to generate a solution at all timesteps during each iteration. Essentially, the loop structure is reordered such that the timestep loops take place within each iterative loop as seen in Algorithm 4.3. The obvious downside is that the solution at every timestep must be stored in memory.

Theorem 4.3. *Define* $(E^n)^k \equiv (V^n)^k - \tilde{V}^n$ *where* $\tilde{V}^n$ *is the solution to equation* (3.12) *at timestep* n*. If the matrices* A *and* B *are given by equation* (3.14) *and fully implicit time stepping* ($\theta = 1$) *is used, then the global in time iteration converges at the rate*

$$\frac{\max_i \| (E^i)^{k+1} \|_\infty}{\max_j \| (E^j)^k \|_\infty} \leq \left(1 - \frac{1}{\left[1 + \Delta\tau(\hat{\lambda} + r)\right]^J} \right) \left(\frac{\hat{\lambda}}{\hat{\lambda} + r} \right) \tag{4.3}$$

Proof. See [9]. □

Remark. Equation (4.3) places a bound on the number of correct decimal places in the numerical solution from one iteration to the next. This bound differs from the convergence criteria in line 6 of Algorithm 4.3 which is based on the maximum relative change in the

Algorithm 4.3 Global in Time Iteration

1: $(V^n)^0 = \text{payoff} \qquad n = 0, \ldots, J \qquad \Delta\tau = T/J$

2: **for** $k = 0, 1, 2, \ldots$ until converged **do**

3: **for** $n = 0, 1, \ldots, J-1$ **do**

4: Solve: $\max_{Q_\ell \in Z} \left\{ -A(Q)(V^{n+1})^{k+1} + B(Q)(V^{n+1})^k + C(Q) \right\} = 0$

5: **end for**

6: **if** $\max_\ell \dfrac{|(V_\ell^{n+1})^{k+1} - (V_\ell^{n+1})^k|}{\max\left[\text{Scale}, |(V_\ell^{n+1})^{k+1}|\right]} <$ Tolerance **then**

7: Break from the iteration

8: **end if**

9: **end for**

solution between iterations. As such, using equation (4.3) to predict the approximate number of iterations required to meet a particular convergence tolerance may not correspond to the number of iterations observed in practice.

As the timestep size decreases the global convergence bound becomes

$$\lim_{\substack{\Delta\tau \to 0 \\ J \to \infty}} \left(1 - \frac{1}{\left[1+\Delta\tau(\hat{\lambda}+r)\right]^J}\right)\left(\frac{\hat{\lambda}}{\hat{\lambda}+r}\right) = \left(1 - e^{-T(\hat{\lambda}+r)}\right)\left(\frac{\hat{\lambda}}{\hat{\lambda}+r}\right) \quad . \tag{4.4}$$

Equation (4.4) illustrates that refining the timestep size has a limited impact on the convergence of the global in time method. Additionally, contracts spanning longer periods of time have a theoretically worse convergence bound.

Furthermore, some straightforward analysis can be used to demonstrate that

$$\frac{\hat{\lambda}\Delta\tau}{1 + (r+\hat{\lambda})\Delta\tau} \leq \left(1 - \frac{1}{\left[1 + \Delta\tau(\hat{\lambda}+r)\right]^J}\right)\left(\frac{\hat{\lambda}}{\hat{\lambda}+r}\right) \tag{4.5}$$

indicating that the convergence bound on the local policy iteration in equation (4.2) is less than that on the global in time iteration.

Chapter 5

Numerical Work

This chapter outlines the numerical results obtained for pricing American options under a regime switching model. The results presented will focus on performance comparisons between the different iterative algorithms outlined in Chapter 4. The option values, cpu runtime, and the number of iterations required to satisfy the specified convergence criteria will be tabulated for the fixed-point policy, local policy, and global-in-time schemes. All algorithms were implemented in Matlab R2010b on an Intel ULV SU7300 machine overclocked at 1.7GHz with 4GB of RAM.

5.1 Numerical Procedure and Data

Numerical experiments were performed using a three state regime switching model. The transition probability matrix $\boldsymbol{\lambda}$, jump amplitude matrix $\boldsymbol{\xi}$, and discrete volatilities $\boldsymbol{\sigma}$ are given in equation (5.1). Additional data can be found in Table 5.1.

$$\boldsymbol{\lambda} = \begin{bmatrix} -3.2 & 0.2 & 3.0 \\ 1.0 & -1.08 & .08 \\ 3.0 & 0.2 & -3.2 \end{bmatrix} \quad \boldsymbol{\xi} = \begin{bmatrix} 1.0 & 0.90 & 1.1 \\ 1.2 & 1.0 & 1.3 \\ 0.95 & 0.8 & 1.0 \end{bmatrix} \quad \boldsymbol{\sigma} = \begin{bmatrix} 0.2 \\ 0.15 \\ 0.30 \end{bmatrix} \tag{5.1}$$

An American butterfly contract is used in all numerical runs with a payoff given by

$$V^* = \max(S - K_1, 0) - 2\max(S - (K_1 + K_2)/2, 0) + \max(S - K_2, 0) \quad . \tag{5.2}$$

Exercise	American
Butterfly Parameters: K_1, K_2	90, 110
Risk free rate r	0.02
Scale factor, Ω	$10^6/\Delta\tau$
Maximum Grid Price, $S_{\max}$	5000
Convergence Tolerance	10^{-8}

Table 5.1: Data for the regime switching implementation.

Refinement Level	Nodes	Timesteps	Unknowns
0	76	50	228
1	151	100	453
2	301	200	903
3	601	400	1803
4	1201	800	3603
5	2401	1600	7203

Table 5.2: Grid refinement levels and timestep data used during numerical tests. On each refinement, a new grid point was placed halfway between all old grid points and the number of timesteps was doubled. A constant timestep size is used.

It is assumed that the American contracts in equation (5.2) must be exercised all together at one time. This contract has been used by several authors as a test case for their numerical work [1, 7].

A non-uniformly spaced grid is used for all trials. This consists of a fine mesh nearby the butterfly strikes and an increasingly coarser grid further away. Both fully implicit and Crank-Nicolson timesteps are implemented for contracts lasting 0.5, 0.7, 5, and 10 years.

Numerical tests explore the convergence of each iteration method as ΔS and $\Delta\tau$ tend to zero. This is achieved by simultaneously refining the grid spacing and timestep size; Upon each successive refinement, a new grid point is placed halfway between all old grid points and the number of timesteps is doubled. The succession of grid refinement levels can be seen in Table 5.2.

Remark. Huang et al. [9] found that the direct control scale factor must increase as grid and timestep refinements are made in order to efficiently determine the location of the exercise boundary. Experiments showed that $\Omega = 10^6/\Delta\tau$ is a suitable value.

5.2 Fixed Point Policy Iteration Results

The fixed-point policy iteration was implemented using both fully implicit and Crank-Nicolson (with Rannacher smoothing) timestepping. The average number of iterations at each timestep is tabulated for various grid refinements and contract lengths. These results are shown in Table 5.3.

Observe that the convergence bound in equation (4.1) simplifies to

$$\| A^{-1}(Q^k)B(Q^k) \|_\infty \leq \frac{\theta\hat{\lambda}_j\Delta\tau}{1+\theta(r+\hat{\lambda}_j)\Delta\tau}$$

when using the Ω and $\Delta\tau$ values from these trials. This indicates that the number of iterations required to reach convergence decreases as the timesteps are refined. This is verified in Table 5.3 for a variety of contract lengths.

5.3 Local Policy Iteration Results

The local policy iterations results for fully implicit and Crank-Nicolson timestepping can be seen in Table 5.4 and Table 5.5, respectively. These include the number of outer iterations and average number of inner iterations required for convergence. Note that "inner iterations" refers to those used to solve line 3 in Algorithm 4.2.

The results agree with the convergence bound in equation (4.2); As the refinements increase, the number of outer and inner iterations both decrease. This is true for all contract lengths and for both timestepping methods. Notice that the outer iteration numbers are nearly the same as those for the fixed-point method seen in Table (5.3). Solving the local nonlinear equations until convergence appears to have little impact on convergence of the outer iteration. Thus, the fixed-point iteration requires less computation and may be preferred to the local policy iteration.

5.4 Global-In-Time Iteration Results

The global-in-time iteration outlined in Section 4.3 was implemented using fully implicit and Crank-Nicolson timesteps. The refinement results for the fully implicit method are shown in Table 5.6. We observe that refining the grid has little to no impact on the

		Fully Implicit			Crank-Nicolson		
	Refinement	Value	Iterations per Timestep	CPU Time (seconds)	Value	Iterations per Timestep	CPU Time (seconds)
$T = 0.5$	0	6.4191813304	6.1	0.4	6.438496257	5.2	0.3
	1	6.4232059378	5.1	0.9	6.433060318	4.9	0.9
	2	6.4270814983	4.5	2.8	6.432079690	4.1	2.6
	3	6.4294392692	4.1	9.1	6.431962775	4.1	9.5
	4	6.4306491996	3.9	33	6.431919116	3.3	31
	5	6.4312818970	3.1	113	6.431919625	3.1	119
$T = 0.7$	0	6.9211848385	6.4	0.4	6.940245784	5.8	0.3
	1	6.9217067638	5.4	0.9	6.931349509	5.1	0.9
	2	6.9246710048	5.1	2.9	6.929546911	4.2	2.7
	3	6.9267636123	4.1	9.1	6.929219411	4.1	9.6
	4	6.9278901495	4.1	35	6.929124507	3.5	33
	5	6.9284917981	3.3	117	6.929111223	3.1	120
$T = 5$	0	8.6377593132	12.3	0.6	8.645142805	9.5	0.5
	1	8.6265534087	9.3	1.4	8.630349017	7.4	1.3
	2	8.6246975799	7.3	3.9	8.626633685	6.2	3.7
	3	8.6249530256	6.0	12	8.625926753	5.2	12
	4	8.6252546274	5.2	42	8.625743441	4.4	39
	5	8.6254577917	4.3	142	8.625702939	4.2	149
$T = 10$	0	8.9274131596	17	0.9	8.932920175	13	0.7
	1	8.9035484051	12	1.8	8.906182627	9.4	1.5
	2	8.8944722582	8.8	4.6	8.895774529	7.3	4.2
	3	8.8936349381	7.0	14	8.894290245	6.0	13
	4	8.8936219219	5.6	44	8.893950546	5.2	45
	5	8.8937090116	5.0	161	8.893873781	4.3	152

Table 5.3: Numerical results for the fixed-point policy iteration. Option values are at $S = 93$ in Regime 1.

		Fully Implicit			
	Refinement	Value	Outer Iterations per Timestep	Inner Iterations per Outer Iteration	CPU Time (seconds)
$T = 0.5$	0	6.419181330	6.0	1.92	0.6
	1	6.423205938	5.0	1.90	1.5
	2	6.427081498	4.4	1.89	4.4
	3	6.429439269	4.0	1.87	14
	4	6.430649200	3.8	1.85	52
	5	6.431281897	3.1	1.81	170
$T = 0.7$	0	6.921184838	6.4	1.92	0.6
	1	6.921706764	5.3	1.91	1.6
	2	6.924671005	5.0	1.90	4.9
	3	6.926763612	4.0	1.88	15
	4	6.927890150	4.0	1.87	56
	5	6.928491798	3.2	1.82	176
$T = 5$	0	8.637759313	12	1.97	1.2
	1	8.626553409	9.1	1.96	2.7
	2	8.624697580	7.1	1.93	6.9
	3	8.624953026	5.8	1.91	20
	4	8.625254627	5.0	1.90	68
	5	8.625457792	4.1	1.88	221
$T = 10$	0	8.927413160	17	1.98	1.6
	1	8.903548405	12	1.97	3.4
	2	8.894472258	8.6	1.95	8.3
	3	8.893634938	6.8	1.93	24
	4	8.893621922	5.4	1.91	73
	5	8.893709012	4.8	1.89	258

Table 5.4: Refinement results for the local policy iteration using fully implicit timesteps. Option values are at $S = 93$ in Regime 1. "Inner iterations" refers to the iterations used to solve line 3 in Algorithm 4.2.

		Crank-Nicolson			
	Refinement	Value	Outer Iterations per Timestep	Inner Iterations per Outer Iteration	CPU Time (seconds)
$T = 0.5$	0	6.438496257	5.1	1.89	0.5
	1	6.433060318	4.8	1.87	1.5
	2	6.432079690	4.0	1.85	4.3
	3	6.431962775	4.0	1.84	15
	4	6.431919116	3.2	1.79	47
	5	6.431919625	3.0	1.77	176
$T = 0.7$	0	6.940245784	5.8	1.91	0.6
	1	6.931349509	5.0	1.89	1.6
	2	6.929546911	4.1	1.84	4.3
	3	6.929219411	4.0	1.85	15
	4	6.929124507	3.4	1.81	50
	5	6.929111223	3.0	1.77	177
$T = 5$	0	8.645142805	9.4	1.96	0.9
	1	8.630349017	7.3	1.93	2.2
	2	8.626633685	6.1	1.91	6.3
	3	8.625926753	5.1	1.89	19
	4	8.625743441	4.2	1.86	62
	5	8.625702939	4.0	1.85	231
$T = 10$	0	8.932920175	13	1.97	1.3
	1	8.906182627	9.2	1.96	2.8
	2	8.895774529	7.1	1.93	7.3
	3	8.894290245	5.8	1.91	22
	4	8.893950546	5.0	1.89	73
	5	8.893873781	4.1	1.86	237

Table 5.5: Refinement results for the local policy iteration using Crank-Nicolson timesteps. Option values are at $S = 93$ in Regime 1. "Inner iterations" refers to the iterations used to solve line 3 in Algorithm 4.2.

number of outer iterations as is suggested by the bound in equation (4.4). Furthermore, the average number of inner iterations increases upon each refinement, making the global-in-time method highly inefficient. As the contract length, T, increases, the number of outer iterations required to converge also increases as expected. The refinement results for the Crank-Nicolson method are shown in Table 5.7. While the Crank-Nicolson method requires fewer iterations and less CPU runtime than the fully implicit case, it still suffers from the same refinement issues and problems with longer contract lengths. Note that the option values obtained using the global-in-time iteration are nearly identical to the values from the local policy iteration in Tables 5.4 and 5.5, however, the global-in-time results required significantly longer runtimes to achieve convergence.

Remark. Assuming the bound in equation (4.3) holds with equality, a contract length of $T = 0.7$ would require approximately twice as many iterations as a $T = 0.5$ contract for convergence to 8 decimal points. See Appendix B.

Clearly, the number of iterations does not double for the $T = 0.7$ contract in Tables 5.6 and 5.7. There are two possible explanations:

1. The relative change convergence criteria in line 6 of Algorithm 4.3 terminates the iteration before an accuracy of 8 decimal points is achieved (see Remark in §4.3).

2. The global bound does not hold with equality.

Even if the global bound does not hold with equality, the ratio of successive errors may still be significantly worse than the fixed-point or local policy iterations. This is verified by approximating the exact option values with a numerical solution generated using a convergence tolerance of 10^{-12}. The approximate error terms $\| (E^n)^k \|_\infty$ can then be calculated at each iteration and timestep. The average and maximum ratio of successive error estimates can be seen in Table 5.8. These are computed using fully implicit timesteps for direct comparison with the theoretical bound. Although the error ratios never achieve the global-in-time bound, the ratios are significantly worse than the local policy bound in all runs, making it a comparably inefficient method. Additionally, the error ratios increase as the contract duration increases, growing closer to the convergence bound. Although not tabulated, the approximate error ratios for a $T = 20$ contract was also computed which was consistent with this pattern. At the 3^{rd} refinement level, the average and maximum ratios are 0.83 and 0.96, respectively, with a global-in-time convergence bound of 0.994.

In summary, the global-in-time iteration suffers from the following deficiencies:

- Burdensome memory allocation. Storage of the solution vectors at all timesteps is required.
- Refinement inefficiencies. As grid and timesteps are refined, the number of outer iterations remains approximately the same while more inner iterations are required to reach convergence.
- Poor convergence bounds. The fixed-point and local policy iterations outperform the global-in-time method, particularly for longer duration contracts.

		Fully Implicit			
	Refinement	Value	Outer Iterations	Inner Iterations per Timestep per Outer Iteration	CPU Time (seconds)
$T = 0.5$	0	6.419181331	14	1.92	2.1
	1	6.423205939	14	2.04	6.5
	2	6.427081500	14	2.21	22
	3	6.429439269	14	2.49	85
	4	6.430649201	14	2.91	363
	5	6.431281898	14	3.58	1644
$T = 0.7$	0	6.921184841	17	1.88	2.5
	1	6.921706766	16	2.05	7.4
	2	6.924671005	16	2.24	25
	3	6.926763613	16	2.54	98
	4	6.927890150	16	3.00	423
	5	6.928491800	16	3.74	1924
$T = 5$	0	8.637759320	47	1.81	6.8
	1	8.626553413	44	1.97	20
	2	8.624697583	43	2.18	67
	3	8.624953031	42	2.58	260
	4	8.625254628	42	3.13	1144
	5	8.625457795	42	3.99	5325
$T = 10$	0	8.927413169	75	1.76	11
	1	8.903548417	70	1.87	30
	2	8.894472269	67	2.03	101
	3	8.893634945	66	2.36	390
	4	8.893621933	65	2.86	1670
	5	8.893709017	65	3.63	7707

Table 5.6: Refinement results for the global-in-time iteration using fully implicit timesteps. Option values are at $S = 93$ in Regime 1. "Inner iterations" refers to the iterations used to solve line 4 in Algorithm 4.3.

		Crank-Nicolson			
	Refinement	Value	Outer Iterations	Inner Iterations per Timestep per Outer Iteration	CPU Time (seconds)
$T = 0.5$	0	6.438496257	11	1.92	1.7
	1	6.433060318	11	1.96	5.2
	2	6.432079691	11	2.12	18
	3	6.431962775	11	2.32	68
	4	6.431919117	11	2.64	285
	5	6.431919625	11	3.12	1247
$T = 0.7$	0	6.940245785	13	1.87	2.0
	1	6.931349509	13	1.94	6.0
	2	6.929546912	12	2.16	20
	3	6.929219411	12	2.40	76
	4	6.929124508	12	2.76	319
	5	6.929111223	12	3.30	1416
$T = 5$	0	8.645142808	31	1.91	4.8
	1	8.630349020	29	1.99	14
	2	8.626633686	29	2.18	48
	3	8.625926754	28	2.47	181
	4	8.625743444	28	2.91	775
	5	8.625702939	28	3.53	3474
$T = 10$	0	8.932920179	47	1.86	7.2
	1	8.906182630	44	1.91	21
	2	8.895774531	42	2.11	68
	3	8.894290248	42	2.34	261
	4	8.893950547	41	2.77	1096
	5	8.893873783	41	3.32	4866

Table 5.7: Refinement results for the global-in-time iteration using Crank-Nicolson timesteps. Option values are at $S = 93$ in Regime 1. "Inner iterations" refers to the iterations used to solve line 4 in Algorithm 4.3.

	Refinement	Convergence Bounds		Approximate Error Ratio	
		Local Policy	Global-in-Time	Average Value	Maximum Value
$T = 0.5$	0	0.031	0.790	0.22	0.37
	1	0.016	0.793	0.21	0.36
	2	0.008	0.794	0.21	0.36
	3	0.004	0.794	0.20	0.36
	4	0.002	0.795	0.20	0.36
$T = 0.7$	0	0.043	0.887	0.26	0.44
	1	0.022	0.889	0.26	0.45
	2	0.011	0.890	0.26	0.44
	3	0.006	0.891	0.25	0.44
	4	0.003	0.891	0.25	0.44
$T = 5$	0	0.242	0.994	0.65	0.85
	1	0.138	0.994	0.63	0.85
	2	0.074	0.994	0.62	0.85
	3	0.038	0.994	0.62	0.85
	4	0.020	0.994	0.62	0.85
$T = 10$	0	0.389	0.994	0.77	0.93
	1	0.242	0.994	0.75	0.92
	2	0.138	0.994	0.75	0.92
	3	0.074	0.994	0.74	0.92
	4	0.038	0.994	0.74	0.92

Table 5.8: Approximate error ratios, $\max_i \| (E^i)^{k+1} \|_\infty / \max_j \| (E^j)^k \|_\infty$, for the global-in-time-iteration. The approximate error ratio values are computed using fully implicit timesteps. The convergence bounds also correspond to fully implicit timesteps. The exact solution was approximated by generating a solution with a convergence tolerance of 10^{-12}.

Chapter 6

Conclusions

In this paper, the efficiency of several iterative methods used to price American options under a regime switching process were numerically compared. Regime switching models attempt to capture phenomenological market effects such as jumps and mean-reversion by allowing asset prices to be controlled by different stochastic processes at different times. Pricing American options under regime switching required the solution of a set of coupled partial differential equation variational inequalities. The American constraint was implicitly handled through a direct control formulation. The direct control form was then discretized and written as a system of nonlinear algebraic equations. These nonlinear equations were solved using three iterative methods including fixed-point policy, local policy, and global-in-time iterations.

A three-state regime switching model was used to compare the iterative algorithms by tabulating the number of iterations required for each method to converge. This was done for various grid and timestep refinements in order to observe the efficiency of each algorithm as the discretized equations converge to the continuous time equations. The results demonstrated that the fixed-point and local policy algorithms are efficient methods requiring fewer iterations to converge as the grid spacing and timestep size was decreased. By comparison, the global-in-time method required approximately the same number of outer iterations as refinements were made, but more inner iterations were required to solve the local American problem, thus making the method less efficient than the two others. Furthermore, longer contracts required more iterations for the global-in-time method which did not see much improvement as the grid spacing was refined. In all numerical trials, longer CPU runtimes were required for the global-in-time method as compared to both fixed-point and local policy methods. These runtimes were significantly greater for highly refined grids and for longer duration contracts. Additionally, the global-in-time method

requires storage of the solution vectors at all time levels whereas the fixed-point and local policy methods only require a single solution vector to be stored at any given time.

In conclusion, the numerical trials conclusively demonstrate that the global-in-time method is significantly outperformed by the alternatives on the basis of a comparably poor convergence rate and severe memory allocation requirements. These results should hopefully preclude the use of a global-in-time iteration in future numerical work, in contrast to [2, 3, 4, 12]. It is suggested that a fixed-point policy iteration be used instead as it was shown to be a much better alternative requiring the least amount of computation.

APPENDICES

Appendix A

Expressions for α and β in Discretization of $\mathcal{L}_j$ operator

In Section 3.2, the discretization of the Black-Scholes operator was given by

$$(\mathcal{L}_j^h V^n)_{ij} = \alpha_{i,j} V_{i-1,j}^n + \beta_{i,j} V_{i+1,j}^n - (\alpha_{i,j} + \beta_{i,j} + r + \lambda_j) V_{i,j}^n$$

where $\alpha_{i,j}$ and $\beta_{i,j}$ are selected using Algorithm 3.1 to yield a positive coefficient discretization.

The central difference expressions are

$$\alpha_{i,j}^{\text{central}} = \left[\frac{\sigma_j^2 S_i^2}{(S_i - S_{i-1})(S_{i+1} - S_{i-1})} - \frac{(r - \rho_j) S_i}{S_{i+1} - S_{i-1}} \right] \tag{A.1a}$$

$$\beta_{i,j}^{\text{central}} = \left[\frac{\sigma_j^2 S_i^2}{(S_{i+1} - S_i)(S_{i+1} - S_{i-1})} + \frac{(r - \rho_j) S_i}{S_{i+1} - S_{i-1}} \right] \tag{A.1b}$$

The forward difference expressions are

$$\alpha_{i,j}^{\text{forward}} = \left[\frac{\sigma_j^2 S_i^2}{(S_i - S_{i-1})(S_{i+1} - S_{i-1})} \right] \tag{A.2a}$$

$$\beta_{i,j}^{\text{forward}} = \left[\frac{\sigma_j^2 S_i^2}{(S_{i+1} - S_i)(S_{i+1} - S_{i-1})} + \frac{(r - \rho_j) S_i}{S_{i+1} - S_i} \right] \tag{A.2b}$$

The backward difference expressions are

$$\alpha_{i,j}^{\text{backward}} = \left[\frac{\sigma_j^2 S_i^2}{(S_i - S_{i-1})(S_{i+1} - S_{i-1})} - \frac{(r - \rho_j) S_i}{S_i - S_{i-1}} \right] \tag{A.3a}$$

$$\beta_{i,j}^{\text{backward}} = \left[\frac{\sigma_j^2 S_i^2}{(S_{i+1} - S_i)(S_{i+1} - S_{i-1})} \right] \tag{A.3b}$$

Appendix B

Number of Iterations of Global-in-Time Method

In Section 5.4, it was stated that if the convergence bound in equation (4.3) holds with equality, then approximately twice as many iterations are required for a $T = 0.7$ contract to converge as compared to a $T = 0.5$ contract. The rationale for this statement is sketched below.

Define $\delta(T)$ as

$$\delta(T) = \left(1 - e^{-T(\hat{\lambda}+r)}\right)\left(\frac{\hat{\lambda}}{\hat{\lambda}+r}\right)$$

which is the right-hand side of equation (4.4). Writing equation (4.4) as an equality gives

$$\| E^{k+1} \|_\infty = \delta \| E^k \|_\infty$$

By recursive substitution, the error after n iterations is

$$\| E^n \|_\infty = \delta^n \| E^0 \|_\infty$$

Taking logarithms on both sides and solving for n gives

$$n = \frac{\log(\| E^n \|_\infty) - \log(\| E^0 \|_\infty)}{\log \delta} \tag{B.1}$$

Assume the initial error is of order unity $\| E^0 \|_\infty = 1$, the solution converges to 8 decimal points $\| E^n \|_\infty = 10^{-8}$, and the model parameters are $\lambda_j = 3.2$ and $r = 0.02$

(see §5.1). The convergence bounds are then

$$\delta(T = 0.5) = 0.795$$
$$\delta(T = 0.7) = 0.892$$

which can be substituted into equation (B.1). This gives a ratio of iteration numbers of

$$\frac{n_{T=0.7}}{n_{T=0.5}} \approx 2.0$$

indicating twice as many iterations are required for the longer contract.

References

[1] A. Almendral and C. W. Oosterlee. Accurate evaluation of European and American options under the CGMY process. *SIAM Journal on Scientific Computing*, 29(1): 93–117, 2007.

[2] E. Bayraktar. A proof of the smoothness of the finite time horizon American put option for jump diffusions. *SIAM Journal on Control and Optimization*, 48(2):551–572, 2009.

[3] E. Bayraktar and H. Xing. Pricing American options for jump diffusions by iterating optimal stopping problems for diffusions. *Mathematical Methods of Operations Research*, 70:505–525, 2009.

[4] E. Bayraktar and H. Xing. Pricing Asian options for jump diffusions. *Mathematical Finance*, 21(1):117–143, January 2011.

[5] M. Bierbrauer, S. Truck, and R. Weron. Modeling electricity prices with regime switching models. In *Computational Science - ICCS 2004*, volume 3039 of *Lecture Notes in Computer Science*, pages 859–867. Springer Berlin / Heidelberg, 2004.

[6] F. Black and M. Scholes. The pricing of options and corporate liabilities. *The Journal of Political Economy*, 81(3):637–654, May-June 1973.

[7] Y. d'Halluin, P.A. Forsyth, and G. Labahn. A penalty method for American options with jump diffusion processes. *Numerische Mathematik*, 97(2):321–352, April 2004.

[8] R.A. Howard. *Dynamic Programming and Markov Processes*. The MIT Press, Cambridge, MA, 1960.

[9] Y. Huang, P.A. Forsyth, and G. Labahn. Methods for pricing American options under regime switching. Working paper, University of Waterloo, submitted to SIAM Journal on Scientific Computing, 2011.

[10] J. Janczura and R. Weron. An empirical comparison of alternate regime-switching models for electricity spot prices. *Energy Economics*, 32(5):1059–1073, 2010.

[11] J.S. Kennedy. *Hedging contingent claims in markets with jumps*. PhD thesis, School of Computer Science, University of Waterloo, 2007.

[12] H. Le and C. Wang. A finite horizon optimal stopping problem with regime switching. *SIAM Journal on Control and Optimization*, 48(8):5193–5213, 2010.

[13] B. Øksendal and A. Sulem. *Applied Stochastic Control of Jump Diffusions*. Springer Berlin Heidelberg, 2nd edition, 2007.

[14] D.M. Pooley and P.A. Forsyth. Numerical convergence properties of option pricing pdes with uncertain volatility. *IMA Journal of Numerical Analysis*, 23:241–267, 2003.

[15] P. Wilmott. *Paul Wilmott on Quantitative Finance*. John Wiley & Sons Ltd., West Sussex, U.K., 2000.

[16] P. Wilmott. *Frequently Asked Question in Quantitative Finance*. John Wiley & Sons Ltd., West Sussex, U.K., second edition, 2009.