# Numerical Techniques for the American Put

**Sean David Randell**

A dissertation submitted to the Faculty of Science, University of the Witwatersrand, in fulfilment of the requirements for the degree of Master of Science.

November 6, 2006

*Programme in Advanced Mathematics of Finance,*
*University of the Witwatersrand,*
*Johannesburg.*

# Declaration

I declare that this dissertation is my own, unaided work. It is being submitted for the Degree of Master of Science in the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination in any other University.

_____

November 6, 2006

# Abstract

This dissertation considers an American put option written on a single underlying which does not pay dividends, for which no closed form solution exists. As a consequence, numerical techniques have been developed to estimate the value of the American put option. These include analytical approximations, tree or lattice methods, finite difference methods, Monte Carlo simulation and integral representations. We first present the mathematical descriptions underlying these numerical techniques. We then provide an examination of a selection of algorithms from each technique, including implementation details, possible enhancements and a description of the convergence behaviour. Finally, we compare the estimates and the execution times of each of the algorithms considered.

# Acknowledgements

I would like to thank my supervisor, David Taylor, for the many hours spent editing and all the other help offered.

A big thank you to Hardy Hulley, who provided the idea for the dissertation and much needed advice, all along the way.

And of course, thank you to Thomas McWalter, for editing parts of my dissertation and providing much needed company during many late nights spent at Wits.

Grant Lotter and Graeme West also deserve a special mention for helpful insights into aspects of the dissertation.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

The American put pricing problem is one of the most important open problems in mathematical finance. The statement of the problem is deceptively simple, but it is almost certainly true that a closed form solution will remain undiscovered. Numerous articles have been written in which claims to exact solutions have been made. Almost all of these claims have merely reduced the problem to some form of quadrature, which ultimately entails one or more numerical approximations.

With rapidly increasing computational power, numerical approximations have become more convenient. This dissertation will enumerate and elucidate the various numerical techniques that have been proposed whereby this quadrature is either performed or avoided.

The valuation of American-style derivatives is difficult when the early exercise opportunity (opportunities) exists and adds value to the option premium. Simply put, an early exercise opportunity occurs when the value of holding the option is worth less than immediately exercising it, and receiving the *intrinsic value*. In practise, the problem is further complicated by the fact that even making the optimal decision is extremely difficult.

This dissertation will consider an American put option written on a single underlying stock which does not pay dividends over the life of the option. In this case, the value of an American call option is exactly equal to the corresponding European call option, because there are no optimal early exercise opportunities. However, the equivalent American put option may have early exercise opportunities. In this case, the corresponding European put option price is merely a lower bound for the value of the American put option.

The need for an analytical solution has resulted in quasi-analytical solutions or analytical approximations. (MacMillan 1986) and (Barone-Adesi and Whaley 1987) developed a quadratic approximation, which is an analytical solution formed by solving an approximate equation for the true value of the American put option. (Geske and Johnson 1984) treated the American put option as an infinite sum of a series of compound options. The consequential "analytical" solution is only exact in the infinite limit. Another drawback of the approximation is that it requires the calculation of $n$-variate standard normal cumulative distribution functions. These functions are computationally expensive and almost impossible to compute for $n > 4$. Another approximation was presented in (Bjerksund and Stensland 2002), where the American option is treated as a knock-out barrier option with a rebate equal to the intrinsic value at the barrier.

Unfortunately, in general, these analytical approximations yield poor estimates. As a consequence, more sophisticated numerical techniques have been explored. (Brennan and Schwartz 1977) introduced a finite difference method. Finite difference methods estimate the solution of a partial differential equation (PDE) by discretising the solution space into a grid and then solving the PDE by recursion. (Cox, Ross, and Rubinstein 1979) estimated the value of the American put option using a binomial tree method. This was later extended by (Hull and White 1993) into a trinomial method. Tree methods also discretise the solution space, although the discretisation is chosen to represent the distribution of the underlying stock price process, instead of the entire solution space. Finite difference methods and tree methods are attractive techniques for estimating the value of the American put option because they easily incorporate the early exercise feature. Unfortunately, if the number of sources of uncertainty increases, they become intractable. This is known as "the curse of dimensionality".

(Boyle 1977) first introduced simulation, or Monte Carlo methods, to mathematical finance. In Monte Carlo methods, sample paths for the state variables are simulated, which makes it difficult to incorporate the early exercise feature of American-style derivatives. In general, the early exercise feature is usually solved by using a backward recursion, such as dynamic programming. (Tilley 1993a) first applied a Monte Carlo method to estimate the value of the American put option. Although the method is crude, it demonstrated that Monte Carlo methods could be applied to value American-style derivatives. Later, adaptations of the Monte Carlo technique to the American put problem were introduced in (Grant, Vora, and Weeks 1996), (Longstaff and Schwartz 2001), amongst others. (Boyle, Broadie, and Glasserman 1997) and (Fu, Laprise, Madan, Su, and Wu 2001) provided a summary of the Monte Carlo methods available for pricing American-style options.

The value of the American put option can also be represented as an integral equation. (Chiarella, Kucera, and Ziogas 2004) provides a survey of the various integral representations for American options. One representation decomposes the value of the American put option into a linear combination of the equivalent European put option value and an early exercise premium (see (Carr, Jarrow, and Myneni 1992), (Jacka 1991) and (Kim 1990)). The early exercise premium appears as an integral whose value can be estimated by combining a quadrature scheme with a search method for locating optimal early exercise.

This dissertation first introduces the mathematical theory underlying these numerical techniques. Each subsequent chapter will focus on a different numerical approach used in the pricing of American options, including: analytical approximations, tree or lattice methods, Monte Carlo methods, finite difference methods, and integral approximations. Each chapter will discuss a selection of the algorithms for each approach. This will include details on implementation, possible enhancements and convergence characteristics. The concluding chapter provides a comparison of the estimates produced by the methods examined.

# Chapter 2

# The Mathematical Description of the American Put Pricing Problem

## 2.1 Preliminaries

We start by defining a filtered probability space $(\Omega, \mathscr{F}_T, \mathbf{F}, \mathbb{P})$ over a time horizon $T \in (0, \infty)$. We assume that the filtration $\mathbf{F} = \{\mathscr{F}_t \mid t \in [0,T]\}$ satisfies the usual conditions (see (Karatzas and Shreve 1988, p.10)). Furthermore, we specify that $\mathscr{F}_0$ is trivial, i.e.

$$\mathscr{F}_0 = \big\{A \in \mathscr{F}_T \;\big|\; \mathbb{P}[A] = 0 \text{ or } \mathbb{P}[A] = 1\big\}.$$

A consequence of this assumption is that any process adapted to $\mathbf{F}$ is a.s. degenerate at time zero. Unless stated otherwise, it will be assumed that all processes are adapted to this filtration and all stopping times are defined with respect to it.

Our intention is to describe a complete market (Elliott and Kopp 1999, Ch. 4) consisting of two assets and driven by one source of uncertainty, namely a Brownian motion, $W = \{W_t \mid t \in [0,T]\}$. The first asset is a riskless savings account, $B = \{B_t \mid t \in [0,T]\}$ given by the deterministic equation

$$B_t = \exp\{rt\},$$

for all $t \in [0,T]$. Here $r \in (0, \infty)$ is a constant riskfree short rate. The second asset is a stock, $S = \{S_t \mid t \in [0,T]\}$, whose price process is determined by the stochastic differential equation (SDE)

$$dS_t = rS_t\, dt + \sigma S_t\, dW_t \tag{2.1}$$

for all $t \in [0,T]$, with an initial value $S_0 \in (0, \infty)$ and volatility $\sigma \in (0, \infty)$. The SDE (2.1) admits the following solution

$$S_t = S_0 \exp\left\{\left(r - \frac{1}{2}\sigma^2\right)t + \sigma W_t\right\} \tag{2.2}$$

for all $t \in [0,T]$, as can be confirmed by a straightforward application of Itô's formula. This process is a geometric Brownian motion.

Notice that the drift rate in equation (2.1) is the riskfree short rate $r$. This means that we have implicitly assumed that the probability measure $\mathbb{P}$ is, in fact, the risk-neutral probability measure. Under the real-world measure, $S$ will have a different drift rate. However, according to the standard martingale theory (Harrison and Pliska 1981), completeness and no-arbitrage guarantee the existence of a unique risk-neutral measure[1]. As is well known, under these conditions, one can perform a change of measure by a straightforward application of Girsanov's theorem (Karatzas and Shreve 1988).

We are interested in the pricing of contingent claims on the underlying $S$. If $g$ is some function such that $g : \mathbb{R}_+ \longrightarrow \mathbb{R}$, then a European contingent claim written on $S$ with payoff $g$ and expiry $T$ can be regarded as a financial instrument which pays the holder $g(S_T)$ at $T$. It is called a *European call option*, with strike price $K \in (0, \infty)$, if the payoff is given by

$$g(S) := (S - K)^+,$$

for all $S \in \mathbb{R}_+$. The corresponding *European put option*, has the payoff

$$g(S) := (K - S)^+,$$

for all $S \in \mathbb{R}_+$. The price of a European contingent claim is equal to the discounted expected value of its payoff (under the risk-neutral measure) at expiry (Elliott and Kopp 1999, Ch. 7). As a consequence, the European call option price at any time $t \in [0, T]$ is

$$c(t, T; K) = \exp\{-r(T - t)\}\mathbb{E}\big[(S_T - K)^+ \big| \mathscr{F}_t\big], \tag{2.3}$$

while the corresponding European put option price is

$$p(t, T; K) = \exp\{-r(T - t)\}\mathbb{E}\big[(K - S_T)^+ \big| \mathscr{F}_t\big]. \tag{2.4}$$

The conditional expectations in equations (2.3) and (2.4) can be evaluated using the transition density function for geometric Brownian motion, yielding the Black-Scholes (Black and Scholes 1973) formulae for the European call and put prices

$$\begin{aligned} c(t, T; K) &= S\Phi(d_1) - K \exp\{-r(T - t)\}\, \Phi(d_2), & (2.5)\\ p(t, T; K) &= K \exp\{-r(T - t)\}\, \Phi(-d_2) - S\Phi(-d_1). & (2.6) \end{aligned}$$

Here, $\Phi(x) := \left(1/\sqrt{2\pi}\right) \int_{-\infty}^{x} \exp\{-z^2/2\}\ dz$, is the standard cumulative normal distribution function for $x \in \mathbb{R}$, and

$$d_1 := \frac{\log(S_t/K) + (r + \frac{1}{2}\sigma^2)(T - t)}{\sigma\sqrt{T - t}}\,, \qquad d_2 := d_1 - \sigma\sqrt{T - t}.$$

An American option with payoff $g : \mathbb{R}_+^2 \longrightarrow \mathbb{R}_+$ and expiry $T$, pays $g(t, S_t)$ when exercised at time $t \in [0, T]$. An *American call option*, with strike price $K \in \mathbb{R}_+$, has a payoff

$$g(t, S) := (S - K)^+,$$

[1] Recall that a risk-neutral measure is a probability measure under which the discounted stock price process is a martingale.

for all $S \in \mathbb{R}_+$ at all times $t \in [0,T]$, while the payoff of the corresponding *American put option* is

$$g(t,S) := (K-S)^+, \tag{2.7}$$

for all $S \in \mathbb{R}_+$ at all times $t \in [0,T]$. It was shown in (Merton 1973) that the price of an American call option, $C(t,T;K)$, written on a stock that pays no dividends is equivalent to the price of the corresponding European call option, $c(t,T;K)$. However, due to a positive probability of early exercise, the prices of a European put option and the corresponding American put option are not equal. The exercise time $t \in [0,T]$ can be chosen by the holder to be any time up to and including the expiry $T$. The evolution of the underlying stock determines the optimal exercise time for any particular put option. No-arbitrage considerations impose the following lower bound on the price of an American put option

$$P(t,T;K) \geq p(t,T;K), \tag{2.8}$$

where $P(t,T;K)$ denotes the price of the American put option.

*No closed form solutions for the price of the American put option exist and consequently numerical methods are needed to approximate the price.* The numerical methods examined in this dissertation are: lattice or tree methods, analytical approximations, finite difference methods, Monte Carlo methods, and integral representation approximations. The remainder of this chapter is devoted to the theory underlying these numerical methods. Initially the Snell's envelope representation is presented and then reformulated as a free boundary problem and subsequently as an integral representation.

## 2.2 The Snell's Envelope Representation

The Snell's envelope representation is used in tree and Monte Carlo methods. It examines the stochastic process which majorises the discounted expected value process.

**Definition 2.2.1.** A random variable $\tau \in [0,T]$ is a *stopping time* if, for every $t \in [0,T]$, $\{\tau \leq t\} \in \mathscr{F}_t$.

**Notation 2.2.2.** Let $\mathcal{M}_{u,t}$ denote the collection of stopping times $\tau$ such that $u \leq \tau \leq t$ a.s. for all $u \leq t \in [0,T]$.

We need to find a process, $Y_t$, called the *essential supremum*, which represents the expected maximum exercise value over all stopping times until expiry.

**Proposition 2.2.3.** *For every family $F$ of real-valued measurable functions $f : \Omega \to \mathbb{R}$ defined on $(\Omega, \mathscr{F}_T, \mathbf{F}, \mathbb{P})$, there exists, one and, up to equivalence, only one measurable function $g : \Omega \to \mathbb{R}$ such that*

1. *$g \geq f$ a.s. for all $f \in F$;*
2. *if $h$ is a measurable function such that $h \geq f$ a.s. for all $f \in F$, then $h \geq g$ a.s.*

*Proof.* See (Neveu 1975). □

The function $g$, which is the least upper bound on the family of curves $F$, is called the *essential supremum* of $F$.

**Notation 2.2.4.** The essential supremum, $\{Y_t | t \in [0,T]\}$, is written as

$$Y_t := \operatorname*{ess\,sup}_{\tau \in \mathcal{M}_{t,T}} \mathbb{E}\big[g(\tau, S_\tau)\big|\mathscr{F}_t\big], \tag{2.9}$$

for all $t \in [0,T]$.

**Definition 2.2.5.** A real-valued stochastic process $\{M_t | t \in [0,T]\}$ is called a *supermartingale* with respect to the filtration $\mathscr{F}_t$, if

1. $\mathbb{E}\big[|M_t|\big] < \infty$ for all $t \in [0,T]$; and
2. $\mathbb{E}\big[M_t\big|\mathscr{F}_u\big] \leq M_u$ for all $u \leq t \in [0,T]$.

The definitions of the essential supremum and the supermartingale lead to the following specification of a process $\{Y_t\}$ which is called *Snell's envelope*. The initial value of this process gives the price of the American put option at time $T$ (Karatzas 1988).

**Theorem 2.2.6.** *The no-arbitrage price of an American put option at time $t$ is equal to the intial value $Y_t$ of the Snell's envelope given by*

$$Y_t = \operatorname*{ess\,sup}_{\tau \in \mathcal{M}_{t,T}} \mathbb{E}\big[\exp\{-r(\tau - t)\}\,(K - S_\tau)^+\big|\mathscr{F}_t\big] \tag{2.10}$$

*Proof.* See (Musiela and Rutkowski 1998, Th. 8.1.1). □

The Snell's envelope representation (2.10) is used for tree and Monte Carlo methods, by applying Bellman's principle of dynamic programming. This is achieved in a discrete setting by using backward recursion to construct a new adapted process, $Z = \big\{Z_{t_i}\big|0 = t_0 \leq t_1 \leq \cdots \leq t_n = T\big\}$, where

$$\begin{aligned} Z_{t_n} &= g(t_n, S_{t_n}), & (2.11)\\ Z_{t_i} &= g(t_{i-1}, S_{t_{i-1}}) \vee \mathbb{E}\big[e^{-r\Delta t} g(t_i, S_{t_i})\big|\mathscr{F}_{t_{i-1}}\big], & (2.12) \end{aligned}$$

where $i = n-1, \ldots, 1$ and $\Delta t = t_i - t_{i-1}$. Consequently, the process is computed by backward recursion — each successive value of $Z_{t_i}$ is set to the maximum of the intrisic value of the option or the discounted expected value one time step ahead, $Z_{t_{i+1}}$, conditional on the information at time $t_i$.

## 2.3 The Free Boundary Problem

In this exposition, we primarily follow the account of (Elliott and Kopp 1999). The solution to the Snell's envelope representation (2.10) yields a stochastic process whose initial value, $Y_0$, is the price of the American put option. This initial value is often determined numerically using the Bellman equation.

In contrast, the solution of the free boundary problem is the initial cost of a portfolio which replicates the price of the American put option. A large variety of

finite difference methods used in mathematical finance employ the free boundary representation. The approach is to numerically approximate the solution of the corresponding partial differential equation (PDE) subject to a free boundary condition.

The price of the American put option, $P(t, S_t)$ at time $t \in [0, T]$, with initial stock price $S_0 \in \mathbb{R}_+$, is expressed as the maximum, over all stopping times, of expected discounted early exercise values,

$$P(t, S) = \sup_{\tau \in \mathcal{M}_{t,T}} \mathbb{E}_{t,S_0} \left[ \exp\{-r(\tau - t)\} (K - S_\tau)^+ \right],$$

which is equivalent to the Snell's envelope representation in equation (2.10) (proved in (Jaillet, Lamberton, and Lapeyre 1990)).

The positive probability of early exercise means that there exists a time dependent function, $b(t)$, called the *early exercise boundary*, which can be used to determine if early exercise is optimal or not. If the stock price coincides with $b(t)$ at any time $t \in [0, T]$, it is optimal to exercise the option. Then,

$$P(t, b(t)) = K - b(t). \tag{2.13}$$

The early exercise boundary is defined by a continuous set of stock prices. Then, $b(t)$ partitions the time-space domain (here, space is equivalent to stock price) $[0, T) \times \mathbb{R}_+$ into a *continuation region*, $\mathcal{C}$, where exercise is not optimal,

$$\mathcal{C} = \left\{ (t, S) \in [0, T) \times \mathbb{R}^+ \mid P(t, S) > (K - S)^+ \right\}, \tag{2.14}$$

and a *stopping region*, $\mathcal{S}$, where exercise is optimal,

$$\mathcal{S} = \left\{ (t, S) \in [0, T) \times \mathbb{R}^+ \mid P(t, S) = (K - S)^+ \right\}.$$

Hence, the value of the American put option is exactly equal to the value of exercising the option (called *the intrinsic value*) in the stopping region. We now provide a theorem which proves that the value of the American put option obeys the Black-Scholes PDE (Black and Scholes 1973) in the continuation region.

**Theorem 2.3.1.** *$P(t, S)$, the price of the American put option, satisfies the Black-Scholes PDE*

$$\frac{\partial P}{\partial t} + rS\frac{\partial P}{\partial S} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 P}{\partial S^2} - rP = 0 \tag{2.15}$$

*in the continuation region, $\mathcal{C}$.*

*Proof.* Initially, apply Itô's formula to $\{e^{-rt}P(t,S_t)\}$ to obtain the SDE

$$\begin{aligned} d\left(e^{-rt}P(t,S_t)\right) &= \frac{\partial}{\partial t}\left(e^{-rt}P(t,S_t)\right)dt + \frac{\partial}{\partial S}\left(e^{-rt}P(t,S_t)\right)dS_t \\ &\quad + \frac{1}{2}\frac{\partial^2}{\partial S^2}\left(e^{-rt}P(t,S_t)\right)d\langle S,S\rangle_t \\ &= e^{-rt}\Bigg[-rP(t,S_t)\,dt + \frac{\partial P}{\partial t}(t,S_t)\,dt + \frac{\partial P}{\partial S}(t,S_t)\left(rS_t\,dt + \sigma S_t\,dW_t\right) \\ &\quad + \frac{1}{2}\sigma^2 S_t^2\frac{\partial^2 P}{\partial S^2}(t,S_t)\,dt\Bigg] \\ &= e^{-rt}\left(\frac{\partial P}{\partial t}(t,S_t) + rS_t\frac{\partial P}{\partial S}(t,S_t) + \frac{1}{2}\sigma^2 S_t^2\frac{\partial^2 P}{\partial S^2}(t,S_t) - rP(t,S_t)\right)dt \\ &\quad + e^{-rt}\sigma S_t\frac{\partial P}{\partial S}(t,S_t)\,dW_t. \end{aligned}$$

where $S = \mathbb{E}[S_t|\,\mathscr{F}_t]$. In the continuation region, all discounted tradeable securities are martingales. Consequently the drift term vanishes. Then the following PDE for $P(t,S_t)$ is satisfied for all $(t,S_t)\in\mathcal{C}$

$$\frac{\partial P}{\partial t}(t,S_t) + rS_t\frac{\partial P}{\partial S}(t,S_t) + \frac{1}{2}\sigma^2 S_t^2\frac{\partial^2 P}{\partial S^2}(t,S_t) - rP(t,S_t) = 0.$$

□

**Notation 2.3.2.** The operator $\mathcal{L}_{BS}$ given by

$$\mathcal{L}_{BS} = \frac{\partial}{\partial t} + rS_t\frac{\partial}{\partial S} + \frac{1}{2}\sigma^2 S_t^2\frac{\partial^2}{\partial S^2} - r$$

may be used to represent the Black-Scholes PDE (2.15).

Unfortunately, the early exercise boundary is unknown *a priori* and as a consequence, the stopping region and continuous region are also unknown *a priori*. The solution $P(t,S_t)$ of the Black-Scholes PDE (2.15) and the early exercise boundary make up our free boundary problem. The following Dirichlet and Neumann conditions complete the specification of the free boundary problem[2].

No-arbitrage arguments dictate that the value of the American put option must be at least equal to its intrinsic value,

$$P(t,S) \geq (K-S)^+, \quad (t,S)\in[0,T)\times\mathbb{R}_+. \tag{2.16}$$

Also, $P(t,S)$ tends to zero as the stock price tends towards infinity,

$$\lim_{S\to\infty} P(t,S) = 0, \quad t\in[0,T). \tag{2.17}$$

The last possible time for exercise occurs at expiry, $T$, where the value of the option is exactly equal to the maximum of its intrinsic value and zero. To ensure continuity $P(t,S)$ must meet the intrinsic value smoothly,

$$\lim_{t\to T} P(t,S) = (K-S)^+, \quad S\geq 0. \tag{2.18}$$

[2] Dirichlet conditions specify the value of the solution, $P(t,S)$, at the boundaries. Neumann conditions specify the value of the first derivative, $\partial P/\partial S$, at the boundaries.

The final two boundary conditions describe the behaviour of $P(t,S)$ around the early exercise boundary, $b(t)$. At the early exercise boundary,

$$\lim_{S \downarrow b(t)} P(t,S) = K - b(t), \quad t \in [0,T). \tag{2.19}$$

The smooth pasting condition requires the derivative $\partial P/\partial S$ to be continuous across the exercise boundary,

$$\lim_{S \downarrow b} \frac{\partial P(t,S)}{\partial S} = -1. \tag{2.20}$$

The derivative $\partial P/\partial S$ below the boundary is $\partial/\partial S(K-S) = -1$. To approximate a solution, the free boundary problem requires reformulation. One common method is to rewrite it as a variational inequality, which is an approach used for solving equilibrium problems.

### 2.3.1 Variational Inequality

Formulating the free boundary problem as a *variational inequality* removes the explicit calculation of $b(t)$. Theorem 2.3.1 shows that $P(t,S)$ satisfies

$$\mathcal{L}_{BS}(P(t,S)) = 0, \tag{2.21}$$

in the continuation region, $\mathcal{C}$. However, $P(t,S) = K - S$ does not satisfy this PDE in the stopping region, $\mathcal{S}$,

$$\begin{aligned} \mathcal{L}_{BS}(K-S) &= rs(-1) - r(K-S) \\ &= -rK \\ &< 0. \end{aligned}$$

Thus, over the entire domain, $P(t,S)$ satisfies,

$$\mathcal{L}_{BS}(P(t,S)) \leq 0. \tag{2.22}$$

Also, rearranging equation (2.16) gives the following inequality for $P(t,S)$ in $\mathcal{C}$,

$$P(t,S) - (K-S)^+ \geq 0, \tag{2.23}$$

while the following equality holds in $\mathcal{S}$

$$P(t,S) - (K-S)^+ = 0. \tag{2.24}$$

Combining equations (2.21) to (2.24) gives the variational inequality representation for $P(t,S)$:

$$\begin{aligned} \mathcal{L}_{BS}(P(t,S))\left(P(t,S) - (K-S)^+\right) &= 0 && (2.25) \\ \mathcal{L}_{BS}(P(t,S)) &\leq 0 && (2.26) \\ P(t,S) - (K-S)^+ &\geq 0, && (2.27) \end{aligned}$$

for all $(t,S) \in [0,T] \times \mathbb{R}_+$.

## 2.4 The Integral Representation

The integral representation of (Kim 1990, Jacka 1991, Carr, Jarrow, and Myneni 1992) decomposed the price of the American put option into a linear combination of the corresponding European put option and an *early exercise premium*. This early exercise premium may be written as an integral involving the early exercise boundary. The solution of the integral requires backward recursion in order to arrive at a value using condition (2.19). The value of the integral may be estimated using some quadrature scheme.

**Theorem 2.4.1.** *The value of an American put option, $P(t,S)$ may be represented as*

$$P(t,S) = p(t,S) + e(t,S), \tag{2.28}$$

*where $p(t,S)$ is the Black-Scholes price for the corresponding European put option, and the early exercise premium, $e(t,S)$, is given by*

$$e(t,S) = \mathbb{E}_{t,S}\left[\int_t^T \exp\{-r(T-u)\} rK \mathbb{I}_{\{S_u < b(u)\}} du\right] \tag{2.29}$$

*Proof.* See (Elliott and Kopp 1999, Th. 8.5.3) □

# Chapter 3

# Analytical Approximations

## 3.1 Introduction

The different mathematical approaches described in Chapter 2 cannot be solved analytically. They all rely on a numerical method to approximate the solution. In fact, no closed form solution for the American put option exists, except in the case of the perpetual American put option (McKean 1965).

One of the drawbacks of numerical methods is that they often require a large number of calculations and converge slowly. With the acceleration of computing power, these drawbacks are less significant. In the early 1970s, these numerical methods were computationally infeasible, so the first approximations examined were analytical. These approximations are correspondingly quick to solve, but yield poor estimates for the price of the American put option.

There have been many different analytical approximations developed (see (Broadie and Detemple 1996) for a detailed comparison of approximations). We will discuss a selection of these methods: the regression method of (Johnson 1983), the quadratic approximations of (MacMillan 1986, Barone-Adesi and Whaley 1987), the (Geske and Johnson 1984) compound option approximation and the (Bjerksund and Stensland 1993) method.

## 3.2 Bounds on the American Put Option

The solution to the perpetual American put option problem provides two pieces of useful information. The problem was separately solved by (McKean 1965) and (Merton 1973). The option has no expiry date and, consequently, the solution, $P^{\infty}(S)$, does not depend on time. The early exercise boundary is pre-computable and is constant,

$$b^{\infty} = \frac{K\gamma}{1+\gamma}.$$

Which, in turn, yields the analytical solution,

$$P^{\infty}(S) \quad = \quad \frac{K}{\big(1+\gamma\big)\big(b^{\infty}/S\big)^{\gamma}}$$

where $\gamma = 2r/\sigma^2$. These two results provide an initial estimate for the early exercise boundary, and an upper bound for the value of the American put option with finite expiry.

Another useful bound on the value of the American put option was developed in (Margrabe 1978). An upper bound on the price, $P(t,S,K)$, is given by the corresponding European put option price, $p(t,S,Ke^{rT})$, with a strike that grows at the riskless interest rate, $Ke^{rT}$. Then,

$$P(t,S,K) \leq p\big(t,S,K\exp\{rT\}\big). \tag{3.1}$$

## 3.3 Johnson's Regression Formula

The rather crude regression formula developed in (Johnson 1983) uses an upper and lower bound for the price to obtain a quasi-analytical expression. Using the lower bound (2.8) and the upper bound (3.1) we can derive the following expression,

$$P(t,S,K) = \alpha p\big(t,S,K\exp\{rT\}\big) + (1-\alpha)p(t,S,K), \quad 0 \leq \alpha \leq 1$$

since

$$p(t,S,K) \leq P(t,S,K) \leq p\big(t,S,K\exp\{rT\}\big).$$

The value of $\alpha$ depends on the ratio $S/K$, and the products $rT$ and $\sigma^2 T$. In order to implement the approximation we assume a functional form for $\alpha$ in terms of these three factors and regress this against option prices in the market.

## 3.4 Quadratic Approximations

(MacMillan 1986) examined a quadratic approximation of the put price for non-dividend paying stocks. (Barone-Adesi and Whaley 1987) extended this work and obtained an additional approximation for commodity options. The quadratic approximation is an analytical solution, which is the result of solving an approximate problem.

In equation (2.28), the American put option price is decomposed into the sum of the corresponding European price, $p(t,S)$, and an early exercise premium, $e(t,S)$. Because both the European and the American put option satisfy the Black-Scholes PDE in the continuation region, (Barone-Adesi and Whaley 1987) concluded that the early exercise premium must also satisfy the Black-Scholes PDE in this region.

A new PDE for the early exercise premium can be derived. This PDE contains a term which tends to zero for short and long dated contracts. If this term is set to zero, the resulting ordinary differential equation (ODE) can be solved in terms of a *critical stock price*. This critical stock price can be found using an iterative search technique such as the Newton-Rhapson method (Press, Teukolsky, Vetterling, and Flannery 1999). No further enhancements of the quadratic approximation can be made, because it is an exact solution of an approximation to the Black-Scholes PDE. Although it is computationally efficient, the resulting estimate is poor.

### 3.4.1 The Barone-Adesi Whaley Approximation

The (Barone-Adesi and Whaley 1987) method involves the solution of an ODE in terms of a critical stock price, $S^*$. This critical stock price is the solution of,

$$K - S^* = p(t,S) - \Big(1 - \Phi\big(-d_1(S^*)\big)\Big)S^*/q_1,$$

where $\Phi$ is the standard cumulative normal distribution function, and

$$\begin{aligned} q_1 &= \big(1 - \sqrt{1 + 8r/(K\sigma^2)}\big)/2, \\ d_1(S) &= \frac{\ln(S/K) + (r + \frac{1}{2}\sigma^2)T}{\sigma\sqrt{T}}. \end{aligned}$$

$S^*$ may be calculated using the iterative procedure described in Section 3.4.2. Once it has been calculated, the quadratic approximation for the American put option is given by,

$$P(t,S) = \begin{cases} p(t,S) + A_1\big(\frac{S}{S^*}\big)^{q_1} & S > S^* \\ K - S^* & S \le S^* \end{cases}$$

where

$$A_1 = -\big(1 - \Phi(-d_1(S^*))\big)S^*/q_1.$$

and $p(t,S)$ is then calculated using the Black-Scholes formula (2.6).

### 3.4.2 Solving for the Critical Stock Price

$S^*$ may be found using the Newton-Rhapson method. The Newton-Rhapson method is a basic iterative method for finding the zero of a function. From an inital estimate, $S_0$, the method calculates successive values, $S_i$, using

$$S_{i+1} = S_i - f(S_i)/f'(S_i),$$

until a specific estimation tolerance, $\epsilon$, is reached. The relative error between function values of successive estimates should be less than $\epsilon$,

$$\big|f(S_i) - f(S_{i-1})\big| < \epsilon.$$

We estimate the value of $S^*$ by using the Newton-Rhapson method to find the zero of,

$$f(S) = \big(K - S\big) - \big(p(t,S) - \big(1 - \Phi(-d_1(S))\big)S/q_1\big). \tag{3.2}$$

Note that

$$f'(S) = -1 + \Phi(-d_1(S))(1 - 1/q_1) + \big(1 - \phi(-d_1(S))/(\sigma\sqrt{T})\big)/q_1.$$

where $\phi(x) := \exp\{-x^2/2\}/\sqrt{2\pi}$, for all $x \in \mathbb{R}$ is the standard normal density function.

One possible initial estimate for $S^*$ is the constant early exercise boundary, $b^\infty$, of an identical but perpetual American put option given by equation (3.1). The initial estimate used in (Barone-Adesi and Whaley 1987) is,

$$S_0^* = K + (b^\infty - K)\left(1 - \exp\left\{-\frac{(rT + 2\sigma\sqrt{T})K}{b^\infty - K}\right\}\right).$$

## 3.5 The Geske and Johnson Compound Option Approximation

In (Geske 1977), a compound option formula to price risky coupon bonds was derived. This formula was later extended to pricing the American put option in (Geske and Johnson 1984) by treating the American put option as an infinite sum of a series of compound options. As a result, they did not solve the Black-Scholes equation (2.20) subject to the free boundary condition.

The "analytical" solution that the authors presented is exact in the infinite limit. However, the approximation requires the calculation of an $n$-variate standard normal cumulative distribution function as $n$ tends to infinity. These functions are computationally expensive and almost impossible to compute when $n > 4$. Consequently, (Geske and Johnson 1984) suggested a Richardson's extrapolation technique to obtain a closed form solution. The solution extrapolates the value from three otherwise identical put options: $P_1$ which is exercisable only at $T$; $P_2$ which exercisable at $T/2$ and $T$; and $P_3$ which is exercisable at $T/3$, $2T/3$, and $T$. It is not necessary that these exercise points be equally spaced. As a extension of this result, (Bunch and Johnson 1992) derived an adaptation to this approximation by picking the exercise nodes in an optimal fashion in order to maximise the estimate.

However, (Omberg 1987) criticised the validity of their extrapolation method because an extrapolation method only improves convergence when the convergence is uniform. The set of early exercise times for $P_3$, $[T/3, 2T/3, T]$ does not include the set of early exercise times for $P_2$, $[T/2, T]$. As a consequence, the estimate for $P_3$ may be less than the estimate for $P_2$. (Omberg 1987) suggested using a geometric sequence of estimates, $P_1, P_2, P_4, \ldots$, to ensure that the early exercise set in successive approximations always includes those of the previous approximation. As a result, each successive estimate should be worth at least as much as the previous.

### 3.5.1 The Geske and Johnson Analytical Equation

The price of a contingent claim can be expressed as the discounted sum of expected future cashflows, provided a riskless hedge can be constructed (Cox, Ross, and Rubinstein 1979). In a discrete time setting, an American put option can be exercised at $n$ intervals, $t = t_0 < t_1 < \cdots < t_n = T$. However, once the option has been exercised at a particular time, it cannot be exercised at any later time. As a consequence we can treat the American put option as a series of potential cashflows, contingent on the option having not been exercised at all previous time steps. This requires the calculation of multivariate standard normal cumulative distribution functions because we need to calculate the joint probabilities of the option ending below the exercise boundary and remaining above the early exercise boundary at all prior exercise times.

Without loss of generality, we can assume that $t_0 = 0$. The solution derived in (Geske and Johnson 1984) is expressed as a series of options dependent on a discrete early exercise boundary: $b_i \equiv b(t_i) : i = 1, \ldots, n$. Each option with a expiry of $t_i$ is exercised, provided it is in-the-money ($S(t_i) < b_i$), and that it was not previously exercised. As a consequence, we need to use the joint probabilities that the stock

price is above the early exercise boundary at all prior exercise times and is below the early exercise boundary at the considered early exercise time. The correlations of overlapping Brownian motion increments, $\Delta z_i$ and $\Delta z_j$ for $i < j$ is,

$$\rho_{ij} = \frac{\mathbb{C}\text{ov}(\Delta z_i, \Delta z_j)}{\sqrt{\mathbb{V}\text{ar}(\Delta z_i)\mathbb{V}\text{ar}(\Delta z_j)}} = \sqrt{t_i/t_j}.$$

The value of an option that is exercised at $t_1$, is the discounted strike price multiplied by the probability that $S(t_1)$ is less than the early exercise boundary value, $b_1$,

$$K\exp\{-rt_1\}\Phi\big(-d_2(b_1,t_1)\big) - S\Phi\big(-d_1(b_1,t_1)\big),$$

where

$$\begin{aligned} d_1(b_1,t_1) &= \frac{\ln(b_1/S(t_1)) + (r+\sigma^2/2)t_1}{\sigma\sqrt{t_1}}, \\ d_2(b_1,t_1) &= d_1(b_1,t_1) - \sigma\sqrt{t_1}. \end{aligned}$$

The value of an option that is exercised at $t_2$, is the discounted strike price multiplied by the joint probability that $S(t_2)$ is less than the early exercise boundary value, $b_2$; and that $S(t_1)$ is greater than the early exercise boundary value, $b_1$,

$$K\exp\{-rt_2\}\Phi_2\big(-d_2(b_2,t_2), d_2(b_1,t_1), \rho\big) - S\Phi_2\big(-d_1(b_2,t_2), d_1(b_1,t_1), \rho\big),$$

where $\rho$ is the correlation between the two events. Inductively, this leads to the Geske-Johnson expression for the price of the American put option (using the notation of (Bunch and Johnson 1992)),

$$P(0,S) = K\sum_{i=1}^{\infty}\exp\{-rt_i\}\Phi_i\big(\boldsymbol{d}^*_{i2}, R^*_i\big) - S\sum_{i=1}^{\infty}\Phi_i\big(\boldsymbol{d}^*_{i1}, R^*_i\big) \tag{3.3}$$

where $\Phi_i$ is the $i$-th variate standard normal cumulative distribution function, and

$$\begin{aligned} \boldsymbol{d}^*_{i1} &= (d_{11}, d_{21}, \ldots, d_{i1})' \\ \boldsymbol{d}^*_{i2} &= \boldsymbol{d}^*_{i1} - (\sigma\sqrt{t_1}, \ldots, \sigma\sqrt{t_i})' \\ d_{j1} &= \frac{\ln(b_j/S(t_j)) + (r+\sigma^2/2)t_j}{\sigma\sqrt{t_j}} \end{aligned}$$

where $j = 1, \ldots, i$, and $R^*_i = D_i R_i D_i$, where $R_i$ is the correlation matrix and $D_i = \text{diag}(1, \ldots, 1, -1)$. Evaluation of equation (3.3) requires calculation of the $n$-variate standard cumulative normal distribution functions. The univariate and bivariate standard cumulative normal algorithms are given in Appendix B. An algorithm for the trivariate standard cumulative normal function can be found in (West 2005). However, algorithms for higher orders are unknown.

### 3.5.2 Solving for the Early Exercise Boundary

The discrete early exercise boundary is found recursively using the Newton-Rhapson method (see Section 3.4.2). Truncating equation (3.3) gives an expression for an estimate of the put price, $\widehat{P}^k(0,S)$,

$$\widehat{P}^k(0,S) = K\sum_{i=1}^{k}\exp\{-rt_i\}\Phi_i\big(\boldsymbol{d}^*_{i2},R^*_i\big) - S\sum_{i=1}^{k}\Phi_i\big(\boldsymbol{d}^*_{i1},R^*_i\big). \tag{3.4}$$

The estimate has $k$ discrete exercise times and $t_k$ coincides with $T$. In contrast to many of the methods for estimating the value of the American put option, we move forward in time to estimate the discrete early exercise boundary, $\widehat{b}_i : i = 1,\ldots,n$. On each iteration, we use the known estimates that have already been calculated. Therefore, at $t_i$ we calculate $\widehat{b}_i$ by finding the zero of the function,

$$f(S) = (K-S) - \widehat{P}^i(0,S), \tag{3.5}$$

for $i = 1,\ldots,n$, using the Newton-Rhapson method.

### 3.5.3 Geske-Johnson Richardson's Extrapolation

Richardson's extrapolation is a technique to combine different numerical approximations in such a manner that a better estimate is obtained (Johnson and Riess 1982). Geske and Johnson apply the Richardson's extrapolation to a finite sum of American put options. For each $i : 1,\ldots,n$, the corresponding option, $\widehat{P}^i(0,S)$, is exercisable at $i$ equally spaced time steps, $T/i, 2T/i, \ldots, T$. The two point-, $\widehat{P}^2_{RE}(0,S)$; three point-,$\widehat{P}^3_{RE}(0,S)$; and four point-, $\widehat{P}^4_{RE}(0,S)$ Richardson's extrapolation estimates are given by (where $\widehat{P}^i := \widehat{P}^i(0,S)$):

$$\begin{aligned}
\widehat{P}^2_{RE} &= 2\widehat{P}^2 - \widehat{P}^1,\\
\widehat{P}^3_{RE} &= \widehat{P}^3 + 7/2\big(\widehat{P}^3 - \widehat{P}^2\big) - 1/2\big(\widehat{P}^2 - \widehat{P}^1\big),\\
\widehat{P}^4_{RE} &= \widehat{P}^4 + 29/3\big(\widehat{P}^4 - \widehat{P}^3\big) - 23/6\big(\widehat{P}^3 - \widehat{P}^2\big) + 1/6\big(\widehat{P}^2 - \widehat{P}^1\big).
\end{aligned}$$

### 3.5.4 The Modified Geske-Johnson Method

The Geske-Johnson Richardson's extrapolation uses evenly spaced exercise points, even though there is nothing about the function in equation (3.3) which requires this. (Bunch and Johnson 1992) examined a technique to pick these exercise points optimally in such a way that maximises the premium.

As an example, in the two-point Geske-Johnson approximation, one of the optimal points is chosen to be the expiry of the option, $T$. The maximisation procedure is then simplified to locating the second optimal exercise time. Quite arbitrarily, (Bunch and Johnson 1992) picked seven possible early exercise times, $T/8, 2T/8, \ldots, 7T/8$, and then chose that time which maximised the value of $\widehat{P}^2$. The two point extrapolation formula, $\widehat{P}^2_{RE}$, is then employed to yield an improved approximation.

## 3.6 The Bjerksund and Stensland Method

Another interesting approach to pricing the American put option was suggested in (Bjerksund and Stensland 1993). An analytical approximation for the American call option is calculated by considering the American call option as a European up-and-out barrier option with knock-out barrier, $B$, strike price, $K$, expiry date, $T$, and a rebate, $B - K$. The parity result in Appendix A allows the value of the American put option to be estimated using this method.

### 3.6.1 Bjerksund and Stensland Approximation for the American Call Option

Using a constant early exercise boundary, $B$, the following approximation may be derived for the price of the American call option,

$$\begin{aligned} C(t,S) =& \alpha(B)S^{\beta} - \alpha(B)\varphi(S,T|\beta,B,B) + \varphi(S,T|1,B,B) - \\ & \varphi(S,T|1,K,B) - K\varphi(S,T|0,B,B) + K\varphi(S,T|0,K,B), \end{aligned}$$

where

$$\begin{aligned} \alpha(B) &\equiv (B-K)B^{-\beta}, \\ \beta &\equiv \big(1/2 - \delta/\sigma^2\big) + \sqrt{(\delta/\sigma^2 - 1/2)^2 + 2r/\sigma^2}, \end{aligned}$$

$\sigma$ is the volatility, $\delta$ is the cost of carry. The function $\varphi$ is given by

$$\begin{aligned} \varphi(S,T|\gamma,H,B) = \exp\big\{\lambda T\big\} S^{\gamma} \Bigg[ & \Phi\Big(-\big(\log(S/H) + (\delta + (\gamma - 1/2)\sigma^2)T\big)/\sigma\sqrt{T}\Big) \\ & - (B/S)^{\kappa}\Phi\Big(-\big(\log(B^2/xH) + (\delta + (\gamma - 1/2)\sigma^2)T\big)/\sigma\sqrt{T}\Big)\Bigg], \end{aligned}$$

where

$$\begin{aligned} \lambda &= -r + \gamma\delta + \gamma(\gamma-1)\sigma^2/2, \\ \kappa &\equiv 2\delta/\sigma^2 + (2\gamma - 1). \end{aligned}$$

(Bjerksund and Stensland 2002) used a combination of the strike price and the early exercise boundary for the perpetual American put option, $b^{\infty}$, to calculate the constant early exercise boundary, $B$,

$$B = b^0 + \big(b^{\infty} - b^0\big)\big(1 - \exp\big\{h(T)\big\}\big)$$

where

$$\begin{aligned} h(T) &= -\big(\delta T + 2\sigma\sqrt{T}\big)\big(K^2/(b^{\infty} - b^0)b^0\big) \\ b^0 &\equiv \max\big(K, (r/r - \delta)K\big). \end{aligned}$$

# Chapter 4

# Tree or Lattice Methods

## 4.1 Introduction

Lattice methods are flexible and robust techniques for approximating the value of many derivatives, especially when no analytical solution exists. This is achieved by discretising the true behaviour of the underlying stochastic process (2.1) (in a deterministic manner) and choosing a number of representative paths over which to perform expectations (pricing). Both lattice methods and Monte Carlo methods (to be explored in Chapter 5) employ dynamic programming techniques. In contrast, Monte Carlo methods use a finite number of randomly chosen sample paths to value the option (Hull and White 1988).

The time-state space is discretised, which results in a number of nodes with associated transition probabilities[1]. The initial node is the parent of all subsequent nodes and the number of future nodes increases. The exact number of future nodes will depend on the type of tree and whether or not it is recombining. Different lattice methods produce different node values and transition probabilities. The general approach is to match the moments (usually the means and variances) of the discrete lattice with those of the lognormal distribution of the underlying stochastic process. An application of the central limit theorem provides convergence to the true process as the number of time steps tends towards infinity (Cox, Ross, and Rubinstein 1979).

The binomial option pricing method (so called because they have two transitions at each node) was introduced simultaneously by (Cox, Ross, and Rubinstein 1979) and (Rendleman and Bartter 1979). (Cox, Ross, and Rubinstein 1979) used no-arbitrage arguments and a replicating portfolio technique to calculate the node values and transition probabilities. Later models used different formulations of node values and transition probabilities in order to improve convergence rates. Trinomial trees (so called because they have three transitions at each node) were first introduced to mathematical finance by (Boyle 1986). This approach provides increased flexibility and an improvement in the convergence rate.

Other work on lattice methods explored numerical techniques to improve the convergence rate, instead of altering the parameter values of the lattice structure itself (see (Leisen and Reimer 1996) for a thorough investigation of the convergence of various binomial trees). (Hull and White 1988) applied a control variate technique

[1] These are the probabilities of moving between nodes in the lattice.

developed for Monte Carlo methods to binomial trees. The accelerated binomial option pricing model (Breen 1991) used extrapolation to improve the convergence. Lattice methods are sensitive to the non-linearity of the option payoff, and perform poorly if the strike price does not coincide with a node in the lattice. Consequently, a large non-linearity error is introduced one time step before expiry. The Black-Scholes modification (Broadie and Detemple 1996) attempted to remove this error by calculating Black-Scholes prices for all the nodes one time step before expiry. Another approach to solving this problem is the adaptive mesh model developed in (Figlewski and Gao 1999) which alters the resolution of the lattice. Initially a coarse mesh is created and then a finer mesh is inserted around the strike price one time step before expiry.

The remainder of this chapter will compare the convergence rates of some of the binomial trees (Cox, Ross, and Rubinstein 1979, Rendleman and Bartter 1979) and trinomial trees (Hull 2000, Figlewski and Gao 1999), as well as various enhancements that have been developed (Broadie and Detemple 1996, Hull and White 1988, Breen 1991, Figlewski and Gao 1999) .

## 4.2 The Generalised Lattice

In this section, the underlying state variable is chosen to be a stock price paying no dividends over the life of the option, i.e. $S_t$. A generalised lattice divides the life of the option $[0, T]$ into $n$ equal time steps of size $\Delta t = T/n$. The values for the state variable are also discretised, with the number of states or nodes, $m_i : i = 0, \ldots, n$, being dependent on time. The number and values of the $m_i$'s are dependent on the lattice method used. The time-state space is thus divided into $\sum_{i=0}^{n} m_i$ nodes,

$$S_{(i,j)},$$

where $i = 0, \ldots, n$, and $j = 1, \ldots, m_i$. The initial stock price node $S_{(0,1)}$ is set equal to the initial stock price, $S_0$. Subsequent nodes are calculated using a multiplicative factor which is determined within the choice of lattice method. The remaining required inputs are the transition probabilities,

$$q_{i,jk}$$

where $i = 0, \ldots, n$, $j = 1, \ldots, m_i$, and $k = 1, \ldots, m_{i+1}$. The transition probability $q_{i,jk}$ is the probability of moving from the parent node $S_{(i,j)}$ to node $S_{(i+1,k)}$ in the lattice and must satisfy the following properties,

$$\sum_{k=1}^{m_{i+1}} q_{i,jk} = 1, \quad \text{for all } i, j, \tag{4.1}$$

$$0 \leq q_{i,jk} \leq 1, \quad \text{for all } i, j, k. \tag{4.2}$$

This particular form of numbering for the nodes is chosen so that the lattice can be easily coded in Matlab. The lattice is stored in a sparse matrix, since there is no natural tree structure in Matlab architecture (Figure 4.1 is an example of the lattice numbering).

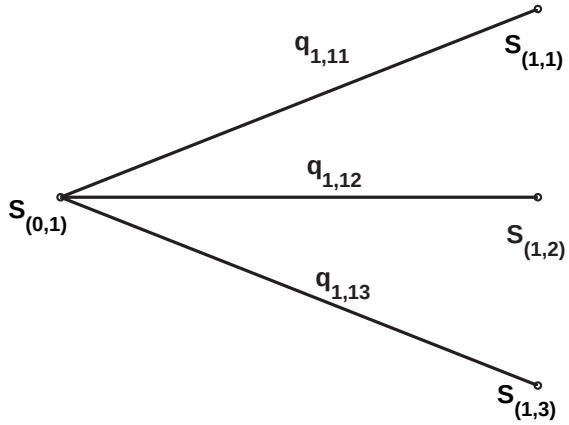


**Fig. 4.1:** A single time step trinomial tree (lattice) showing node values and transition probabilities.

### 4.2.1 Dynamic Programming for Lattice Methods

Dynamic programming is a central feature of computer science. It refers to the decisions that are made while solving a particular problem. A dynamic programming approach ensures that decisions are based on the current state rather than making predefined state independent choices (Baase and Van Gelder 2000). In the context of the pricing of the American put option, dynamic programming refers to the calculation of node values by using previously calculated node values. This is achieved by backward recursion.

The value of the American put option nodes at expiry is equal to the intrinsic value (since this is the last possible chance for exercise)

$$P_{(n,j)} = \big(K - S_{(n,j)}\big)^+, \tag{4.3}$$

where $j = 1, \ldots, m_n$. At each internal node, $(i, j)$, two values are calculated: the intrinsic value of the option, $P^{Int}_{(i,j)}$, and the continuation value, $P^{Cont}_{(i,j)}$. The intrinsic value is equal to $\big(K - S_{(i,j)}\big)$, while the continuation value is calculated using the previously calculated nodes $(l, k)$, where $l > i$ and $k = 1, \ldots, m_l$. The option node value, $P_{(i,j)}$, is set to the maximum of the two,

$$P_{(i,j)} = \max\big\{P^{Int}_{(i,j)}, P^{Cont}_{(i,j)}\big\},$$

where $i = 0, \ldots, n-1$ and $j = 1, \ldots, m_i$.

#### Calculating the Continuation Value

The chosen lattice method is parameterised such that the mean of the lattice matches the mean of the underlying stochastic process under the risk neutral measure. Consquently the value at each node in the lattice must be the discounted expected value under the risk neutral measure,

$$S_t = \exp\big\{-r\Delta t\big\}\mathbb{E}\big[S_{t+1}\big|S_t\big]$$

where $i = 0, \ldots, n-1$, $j = 1, \ldots, m_i$ and $k = 1, \ldots, m_{i+1}$. Which implies that,

$$S_{(i,j)} = \exp\{-r\Delta t\} \sum_{k=1}^{m_{i+1}} q_{i,jk} S_{(i+1,k)}$$

where $i = 0, \ldots, n-1$ and $j = 1, \ldots, m_i$.

This formulation permits the risk-neutral valuation of contingent claims on this tree. The transition probabilities that apply to the stock price lattice also apply at the same nodes for any contingent claim. Consequently, the continuation value for the American put option can be calculated at each node in the tree as,

$$P^{Cont}_{(i,j)} = \exp\{-r\Delta t\} \sum_{k=1}^{m_{i+1}} q_{i,jk} P_{(i+1,k)}, \tag{4.4}$$

where $i = 0, \ldots, n-1$ and $j = 1, \ldots, m_i$.

### 4.2.2 Implementation Issues for Lattice Methods

#### Log-price Lattice or Stock Price Lattice?

Some of the lattice methods construct a tree of the log-price process (the stochastic process of $X = \ln S$),

$$X_{(i,j)},$$

where $i = 0, \ldots, n$ and $j = 1, \ldots, m_i$. The multiplicative factor in the stock price tree formulation now translates into an additive factor in the log-price tree. This has the effect of producing a more regular tree. The stock price lattice is then created using

$$S_{(i,j)} = \exp\{X_{(i,j)}\},$$

where $i = 0, \ldots, n$ and $j = 1, \ldots, m_i$. These two trees are obviously conformally equivalent, although it is often easier to work with the log-price tree.

### 4.2.3 Convergence Behaviour of Lattice Methods

The most important criterion for deciding on a lattice method is the convergence rate. The convergence rate is the rate at which the absolute difference between the lattice price and the "actual" price decreases over each iteration. (Omberg 1987) remarked that for an acceleration technique to be successful, the convergence of the lattice must be uniform.

The approximation error caused by lattice methods can be split into two components: *distribution error* and *non-linearity error* (Figlewski and Gao 1999). The distribution error arises since we are approximating the continuous lognormal distribution of the underlying stock price by the discrete distribution of the lattice. The non-linearity error is caused by non-linear behaviour of the option value, which is exaggerated when the underlying has a value near the strike price. This non-linearity cannot be captured by a discrete lattice. The distribution and non-linearity errors occur for both European and American options.

A third error, caused by the approximation of the correct early exercise decision, is common to all numerical approximation techniques for pricing American options.

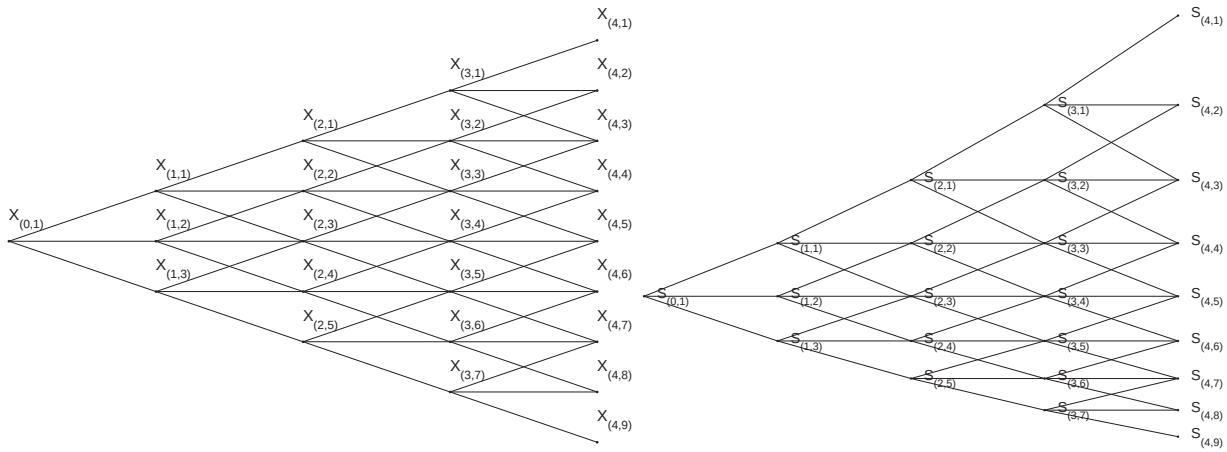


**Fig. 4.2:** A four time step trinomial lattice. The lattice on the left is the log-price lattice.

#### Non-linearity Errors for Lattice Methods

The non-linearity errors in the lattice methods cause a peculiar even-odd convergence (see Figure 4.5 and Figure 4.6). This oscillatory convergence is most noticeable in binomial trees although it also occurs in trinomial trees. The adaptive mesh model (Figlewski and Gao 1999) and the Black-Scholes modification (Broadie and Detemple 1996) are attempts at smoothing this convergence.

#### Distribution Errors for Lattice Methods

The distribution error can be reduced by matching higher moments of the underlying distribution. The (Figlewski and Gao 1999) trinomial tree (Section 4.4.2) matches the kurtosis of the trinomial tree with that of the lognormal distribution in order to decrease the distribution error.

## 4.3 Binomial Trees

In the binomial tree method, the stock price may make one of two moves (either up or down) at each time step. The magnitude of an upward additive move for the log-price process is $u_i$ while a downward additive movement has magnitude $d_i$. Consequently, successive nodes may be calculated by,

$$X_{(i,j)} = \begin{cases} X_{(i-1,j-1)} + u_{i-1} & \text{for an upward movement} \\ X_{(i-1,j)} + d_{i-1} & \text{for a downward movement.} \end{cases}$$

The magnitude of an upward (resp. downward) multiplicative movement for the stock price process is $U_i = \exp\{u_i\}$ (resp. $D_i = \exp\{d_i\}$). Then,

$$S_{(i,j)} = \begin{cases} U_{i-1} S_{(i-1,j-1)} & \text{for an upward movement} \\ D_{i-1} S_{(i-1,j)} & \text{for a downward movement.} \end{cases}$$

From the above it is seen that the binomial option pricing model has three free parameters: $U_i$, $D_i$, and the transition probability $q_i$. These parameters are chosen such that the mean and standard deviation of the tree matches that of the underlying stochastic process ($r$ and $\sigma$ respectively). The system is underdetermined since there are two constraints and three unknowns. This implies an additional degree of freedom in the limit, and more than one degree of freedom when a finite number of steps is used (Omberg 1987). Assuming that the tree is recombining reduces the number of nodes in the binomial tree and leads to the the following restriction on $U_i$ and $D_i$,

$$U_i D_{i+1} = D_i U_{i+1}. \tag{4.5}$$

Consequently, an upward movement followed by a downward movement arrives at the same node as a downward movement followed by an upward movement, throughout the tree. The number of nodes at each point in time is $m_i = i + 1 : i = 0, \ldots, n$ (see Figure 4.3 for an illustration of the binomial lattice). The transition probabilities are given by,

$$q_{i,jk} = \begin{cases} q_i & \text{if } k = j \text{ i.e. upward move} \\ 1 - q_i & \text{if } k = j + 1 \text{ i.e. downward move} \\ 0 & \text{otherwise.} \end{cases}$$

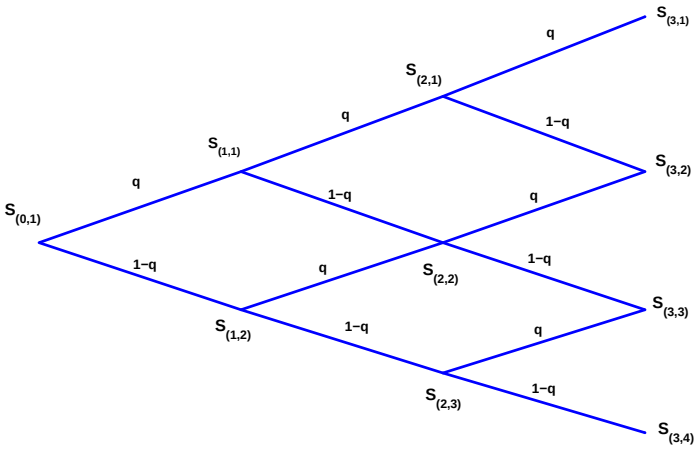


**Fig. 4.3:** A three step binomial tree.

This property of being able to arbitrarily choose one of the parameters leads to various formulations of the binomial pricing model. The rest of this section compares the different formulations that have been proposed.

### 4.3.1 The Cox, Ross and Rubinstein Binomial Tree (CRR)

The first binomial tree to appear in print applied the principle of perfect replication using a portfolio containing the underlying and a simple bank account in order to calculate the lattice parameters (Cox, Ross, and Rubinstein 1979). The CRR

model parameters match the mean of the binomial tree to that of the continuous distribution exactly for any step size. However, the variance is only matched in the limit (Omberg 1987). The movements and probabilities are identical for each time step, although they are dependent on the size of the time step,

$$\begin{aligned} U &= \exp\{\sigma\sqrt{\Delta t}\} \\ D &= \exp\{-\sigma\sqrt{\Delta t}\} \\ q &= \frac{\exp\{r\Delta t\} - D}{U - D}. \end{aligned}$$

These parameters ensure that the stock price tree is centred around the initial stock price, $S_0$ (see Figure 4.4). This greatly aids calculation, because the entire stock price tree does not need to be calculated. The upward deterministic drift is incorporated in the probabilities and consequently the stock price tree appears driftless. Only the stock price node values at expiry, and one time step before expiry, need to be stored because all other internal stock price node values correspond to these.

### 4.3.2 The Jarrow and Rudd Binomial Tree (JR)

The JR binomial tree reduces to the model of (Rendleman and Bartter 1979). Both these models differ from the CRR model in that the mean and variance of the binomial tree matches that of the underlying process over any time step. A constant risk neutral probability of 1/2 is chosen (this restricts the free degree of freedom), while the up and down factors are functions of the the mean and variance. Then,

$$\begin{aligned} U &= \exp\{(r - \sigma^2/2)\Delta t + \sigma\sqrt{\Delta t}\} \\ D &= \exp\{(r - \sigma^2/2)\Delta t - \sigma\sqrt{\Delta t}\} \\ q &= \frac{1}{2}. \end{aligned}$$

The binomial tree with symmetrical probabilities grows asymmetrically (see Figure 4.4 below).

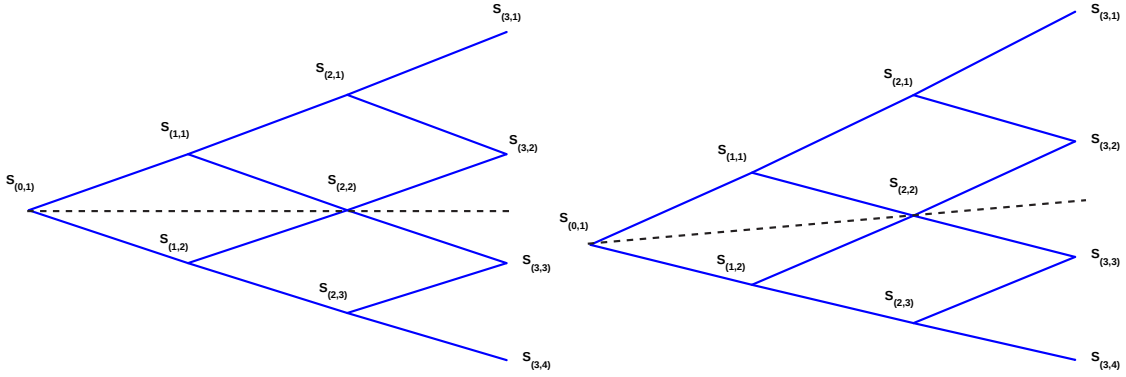


**Fig. 4.4:** The CRR binomial tree (left) is centred around $S_{(0,1)}$. The JR binomial tree (right) is centred around $S_{(0,1)} \exp\{(r - \sigma^2/2)i\Delta t\}, i = 0, \ldots, n$.

### 4.3.3 Calculating Option Values Using a Binomial Tree

The node values of the option at expiry are calculated using equation (4.3). The continuation value at each node is then calculated using equation (4.4) applied to binomial trees,

$$P_{(i,j)}^{Cont} = \exp\{-r\Delta t\}\left[q_i P_{(i+1,j)} + (1-q_i)P_{(i+1,j+1)}\right]$$

where $i = 0, \ldots, n-1$ and $j = 1, \ldots, m_i$. Note that the values used in the calculation of the continuation value may be intrinsic values.

### 4.3.4 Convergence of Binomial Trees

The convergence of binomial trees is oscillatory. However, this convergence is regular for the CRR model when the option is at-the-money (see Figure 4.5). This is because the tree is centred around the initial stock price $S_{(0,1)}$ which is equal to the strike price. Consequently, the strike price coincides with a node. In general this is not

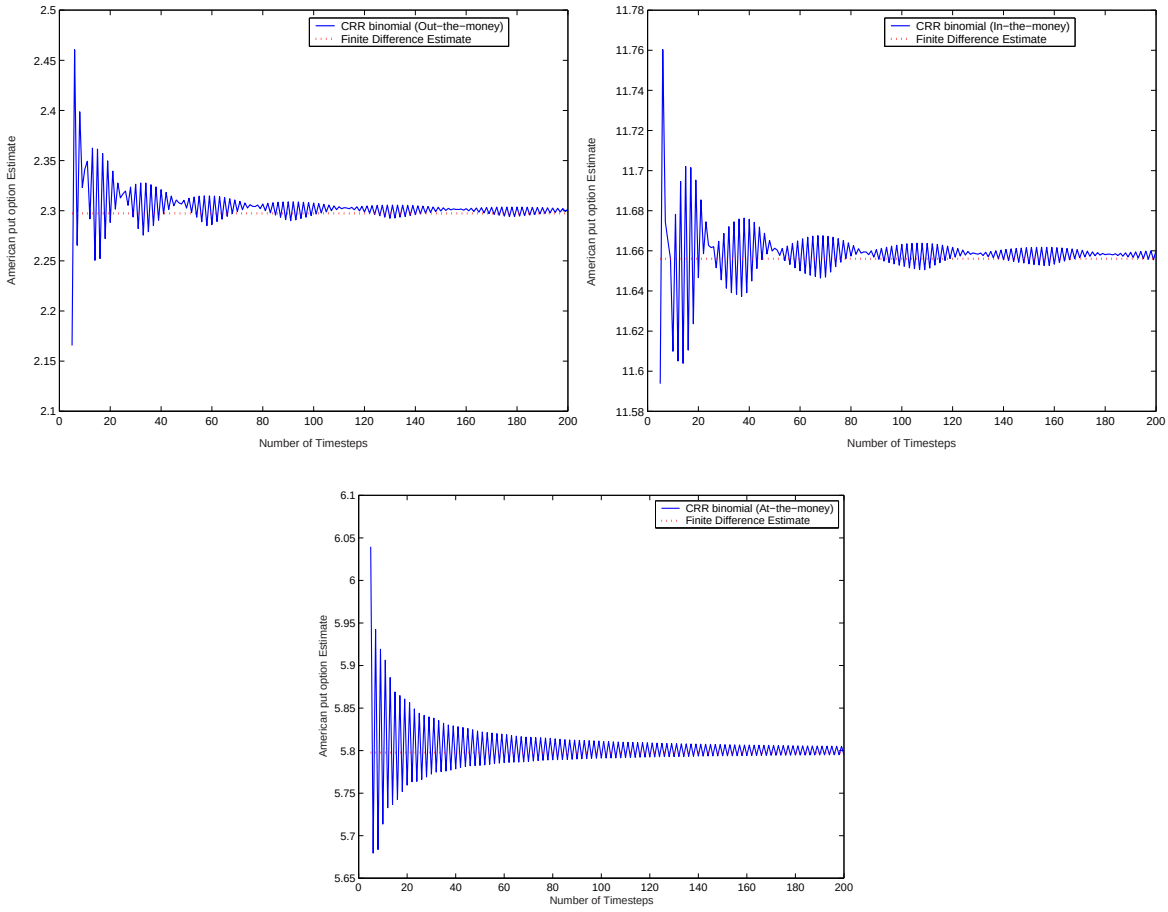


**Fig. 4.5:** The convergence of CRR binomial trees for in-the-money, out-the-money and at-the-money American put options.

the case for the JR model because the stock price tree is not centred around the

initial stock price (see Figure 4.6). Consequently, the strike price does not fall on a node, unless by coincidence.

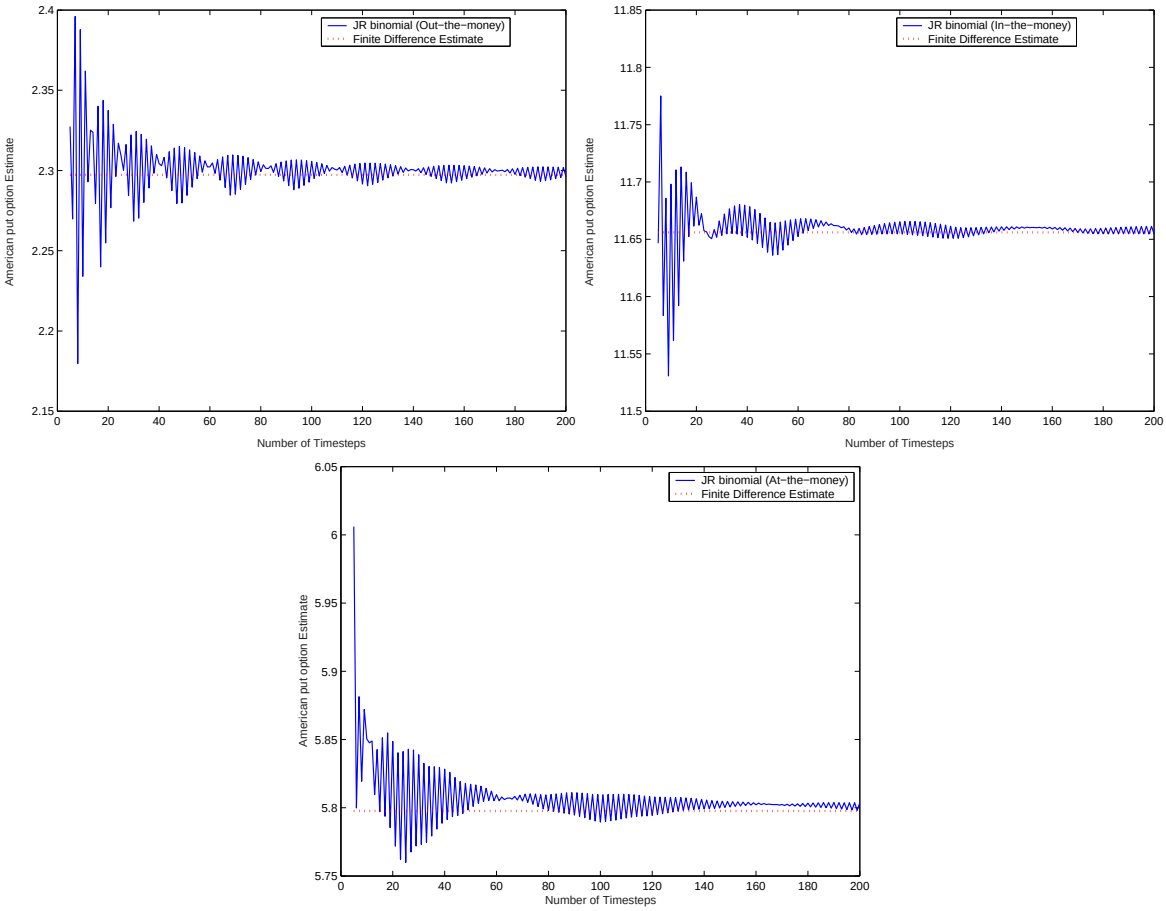


**Fig. 4.6:** The convergence of JR binomial tree for in-the-money, out-the-money and at-the-money American put options.

## 4.4 Trinomial Trees

In the trinomial tree method, the stock price may make one of three moves (up, down or remain the "same") at each time step. The magnitude of an upward additive movement for the log-price process is $u_i$ with probability $q_i^u$ and for a downward additive move is $d_i$ with probability $q_i^d$. It remains the "same" with probability $q_i^m$ (the tree may incorporate a deterministic drift). Then,

$$X_{(i,j)} = \begin{cases} X_{(i-1,j)} + u_{i-1} & \text{for an upward movement} \\ X_{(i-1,j-1)} + r & \text{for no random movement} \\ X_{(i-1,j-2)} + d_{i-1} & \text{for a downward movement,} \end{cases}$$

where $r$ is the deterministic drift of the trinomial tree. The magnitude of an upward (resp. downward) multiplicative movement for the stock price process is

$U_i = \exp\{u_i\}$ (resp. $D_i = \exp\{d_i\}$). Then,

$$S_{(i,j)} = \begin{cases} U_{i-1} S_{(i-1,j)} & \text{for an upward movement} \\ \exp\{r\Delta t\} S_{(i-1,j-1)} & \text{for no random movement} \\ D_{i-1} S_{(i-1,j-2)} & \text{for a downward movement.} \end{cases}$$

The trinomial tree must ensure conservation of probability and consequently the sum of the three probabilities is one. Therefore, the trinomial option pricing model has four free parameters: $U_i$, $D_i$, $q_i^u$ and $q_i^m$ (since $q_i^d = 1 - q_i^u - q_i^m$). As was the case with binomial trees, these parameters are chosen such that the mean and standard deviation of the tree matches that of the underlying stochastic process ($r$ and $\sigma$ respectively). This system is underdetermined because there are three constraints with four unknowns. This implies an additional degree of freedom in the limit, and more than one degree of freedom when a finite number of steps is used (Omberg 1987). One of these degrees of freedom can be restricted by assuming that the tree is recombining, and equation (4.5) still holds. The number of nodes at each time step is $m_i = 2i + 1, i = 0, \ldots, n$ (see Figure 4.7 for an example of a trinomial tree). The transition probabilities are given by,

$$q_{i,jk} = \begin{cases} q_i^u & \text{if } k = j \text{ for an upward movement} \\ q_i^m & \text{if } k = j + 1 \text{ for no random movement} \\ q_i^d & \text{if } k = j + 2 \text{ for a downward movement} \\ 0 & \text{otherwise} \end{cases}$$

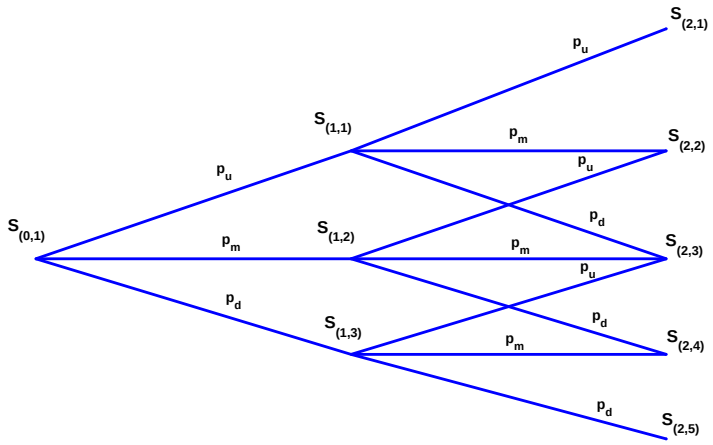


**Fig. 4.7:** A three step trinomial tree.

This freedom to choose one of the parameters leads to a variety of trinomial tree models.

### 4.4.1 Hull-White Trinomial Tree (HW)

One commonly used parametrisation is described in (Hull and White 1993). This formulation relies on embedding the drift in the risk neutral probabilities and con-

sequently is slightly different from the general form above. The parametrisation appears to be dependent on the original methodology of (Boyle 1986).

The Hull-White model is fully described by,

$$\begin{aligned} U &= \exp\{\sigma\sqrt{3\Delta t}\} \\ M &= 1 \\ D &= \exp\{-\sigma\sqrt{3\Delta t}\} \\ q_u &= -\sqrt{\frac{\Delta t}{12\sigma^2}}(r-\sigma^2/2)+1/6 \\ q_m &= 2/3 \\ q_d &= \sqrt{\frac{\Delta t}{12\sigma^2}}(r-\sigma^2/2)+1/6. \end{aligned} \tag{4.6}$$

### 4.4.2 Figlewski and Gao Trinomial Tree (FG)

The additional degree of freedom is used to match the kurtosis of the lattice with that of the lognormal distribution of the underlying continuous time process. The Figlewski and Gao model is fully described by,

$$\begin{aligned} U &= \exp\{\big(r-\sigma^2/2\big)\Delta t+\sigma\sqrt{3\Delta t}\} \\ M &= \exp\{\big(r-\sigma^2/2\big)\Delta t\} \\ D &= \exp\{\big(r-\sigma^2/2\big)\Delta t-\sigma\sqrt{3\Delta t}\} \\ q_u &= 1/6 \\ q_m &= 2/3 \\ q_d &= 1/6. \end{aligned} \tag{4.7}$$

### 4.4.3 Calculating Option Values Using a Trinomial Tree

The node values of the option at expiry are calculated using equation (4.3). The continuation value at each node is then calculated using equation (4.4) applied to trinomial trees,

$$P^{Cont}_{(i,j)} = \exp\big\{-r\Delta t\big\}\left[q_u P_{(i+1,j)} + q_m P_{(i+1,j+1)} + q_d P_{(i+1,j+2)}\right]$$

where $i = 0, \ldots, n-1$ and $j = 1, \ldots, m_i$.

### 4.4.4 Convergence of Trinomial Trees

The convergence of the trinomial trees does not display the same oscillatory convergence as the binomial trees. Although the convergence is improved, it is not uniform (see Figure 4.8).

## 4.5 The Black-Scholes Modification

The Black-Scholes modification of the binomial tree model was first suggested in (Broadie and Detemple 1996). The inspiration for the modification comes from the

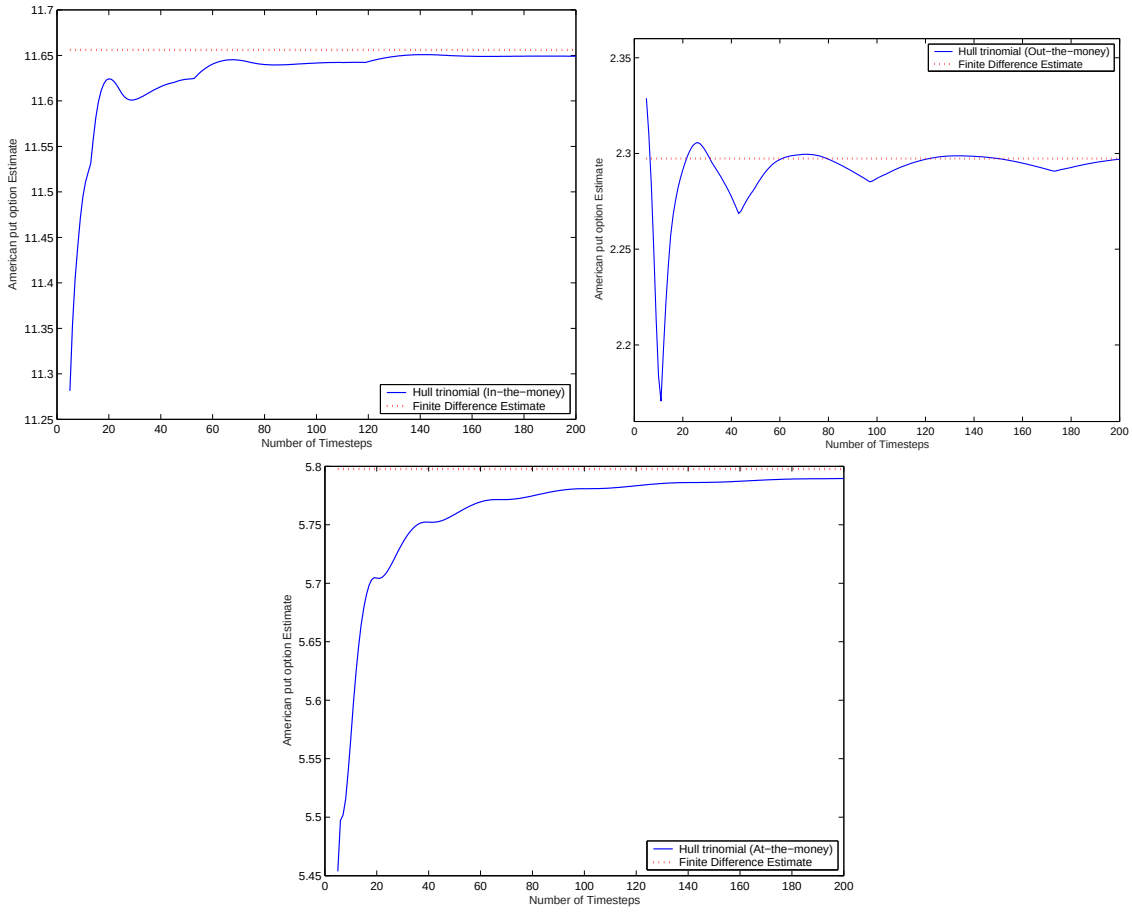


**Fig. 4.8:** The convergence of a Hull-White trinomial tree for an in-the-money, out-the-money and at-the-money American put option.

observation that the continuation value of the American put option, one time step before expiry, is equivalent to the value of an identical European put option with an expiry of $\Delta t$. (This holds because the time step before expiry is the last possible early exercise time).

This value can be calculated using the Black-Scholes option pricing formula (2.6). This simple trick leads to an improved convergence speed and, more importantly, a smoother convergence of the implemented binomial tree method. The simple reason for this is that the non-linearity of the option payoff is removed and no longer propagated through the tree. This does not result in completely uniform convergence, but the large oscillations are removed (see Figure 4.9).

The continuation value at the $n-1$ time step, $P^{Cont}_{(n-1,\,j)}$ is set equal to the Black-Scholes price,

$$P^{Cont}_{(n-1,\,j)} = K \exp\left\{-r(\Delta t)\right\} \Phi(-d_{2,j}) - S_{(n-1,j)} \Phi(-d_{1,j}),$$

where $j = 1, \ldots, m_{n-1}$, and

$$d_{1,j} := \frac{\log(S_{(n-1,j)}/K) + (r + \frac{1}{2}\sigma^2)\Delta t}{\sigma\sqrt{\Delta t}} \quad , \quad d_{2,j} := d_{1,j} - \sigma\sqrt{\Delta t}.$$

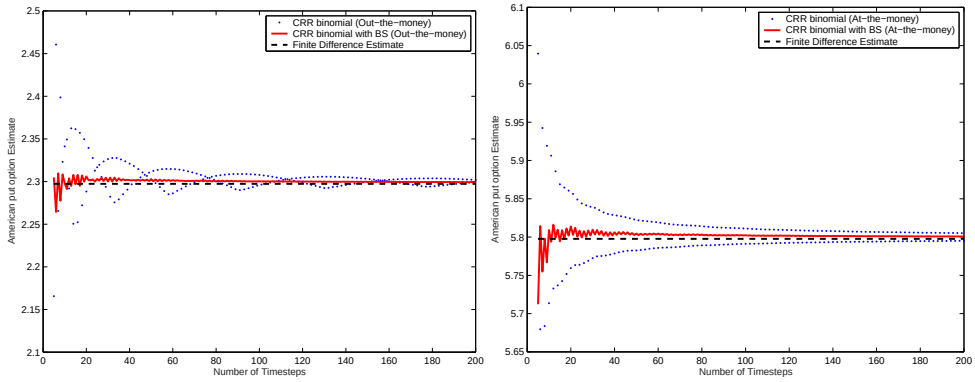


**Fig. 4.9:** The convergence of a CRR binomial tree with the Black-Scholes modification and without the modification for an at-the-money and out-the-money American put option.

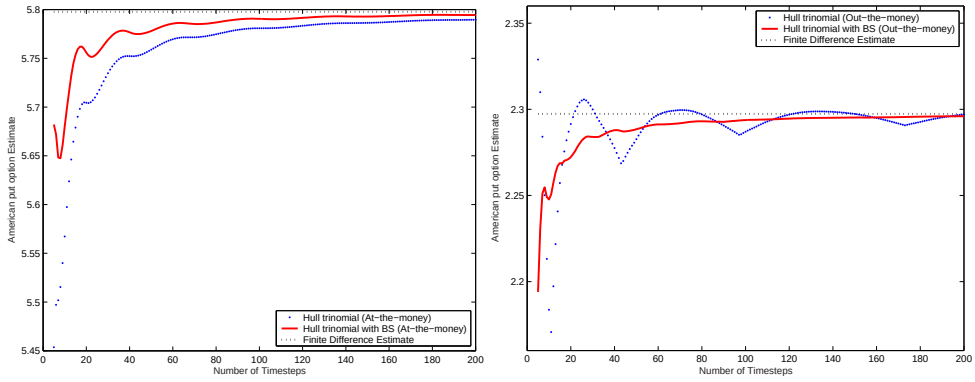


**Fig. 4.10:** The convergence of a Hull-White trinomial tree with the Black-Scholes modification and without the modification for an at-the-money and out-the-money American put option.

## 4.6 The Accelerated Binomial Option Pricing Model

The Richardson's extrapolation approach used in (Geske 1977) (see Section 3.5) was adapted for binomial trees in (Breen 1991). The accelerated binomial option pricing model uses the extrapolation equations from Section 3.5.3 on three types of options: $P_1$, $P_2$, and $P_3$. Again, $P_1$ which can be exercised only at expiry $T$ (i.e. a European option); $P_2$ which can be exercised at times $T/2$ and $T$; and $P_3$ which can

be exercised at times $T/3$, $2T/3$, and $T$. These option values are estimated using a binomial tree.

The convergence acceleration of the extrapolation method only works well when the convergence of the approximating sequence is uniform (Omberg 1987). For many binomial tree formulations, the convergence of the approximating sequence is highly oscillatory (Section 4.2.3). The Black-Scholes modification (Section 4.5) removes this oscillation and the accelerated binomial option pricing method can be applied successfully.

### 4.6.1 Calculating Option Values Using the Accelerated Binomial Option Pricing Method

The accelerated binomial option pricing method computes the American put option price by decomposing the option into a weighted sum of Bermudan options[2]. This sum is analogous to the extrapolation formulae in Section 3.5.3. Each of the Bermudan options are estimated using a standard binomial tree. The Bermudan options can only be exercised at certain times and are European in nature between possible exercise times. As a consequence, we can employ the closed form solution of the (Cox, Ross, and Rubinstein 1979) binomial tree for the European put option, $p(t,s)$,

$$p(t,s) = \exp\{-r(T-t)\} \sum_{i=0}^{n} \binom{n}{i} q^{n-i} (1-q)^{i} \left(K - sU^{n-i}D^{i}\right)^{+}. \tag{4.8}$$

between the exercise times. This reduces the number of calculations and hence a large tree can be built with only a few possible exercise times.

Consider an option that can be exercised at $m$ equally spaced times on the tree: $t_1 = T/m, t_2 = 2T/m, \ldots, t_m = T$. To ensure that the exercise times fall on specific nodes on the tree, the number of time steps must be divisible by the number of exercise times i.e. $n/m \in \mathbb{Z}$. The exercise times on the tree are given by $n/m, 2n/m, ..., n$. The only nodes at which exercise is possible are

$$S_{(i,j)} \quad i = n/m, 2n/m, \ldots, n, \;\; j = 1, \ldots, m_i.$$

Modifying equation (4.8) gives the following expression for the continuation value at each relevant node,

$$P^{Cont}_{(i,j)} = \exp\{-r(T-t)/m\} \sum_{k=0}^{n/m} \binom{n/m}{k} q^{n/m-i} (1-q)^{i} P_{((i+1)n/m, j+k)}$$

where $i = n/m, 2n/m, \ldots, (m-1)n/m, n$ and $j = 1, \ldots, m_i$. Then the value of the American put option at each relevant node is given by,

$$P_{(i,j)} = \max\left(P^{Cont}_{(i,j)}, P^{Int}_{(i,j)}\right)$$

where $i = n/m, 2n/m, \ldots, n$ and $j = 1, \ldots, m_i$.

[2] Bermudan options are options which have a fixed number of possible exercise times during the life of the option.

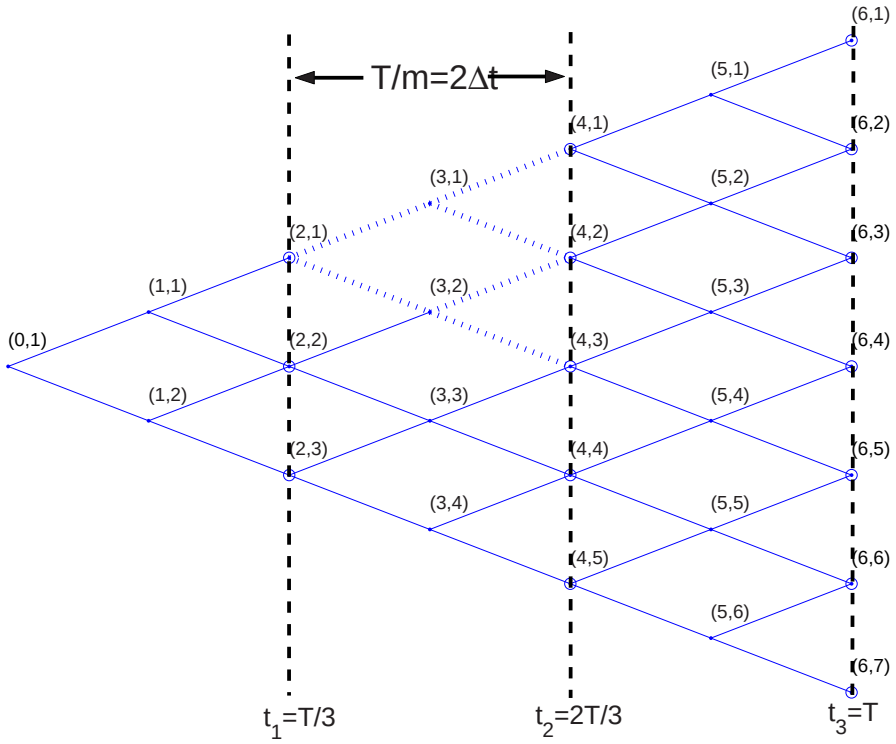


**Fig. 4.11:** A binomial tree with six time steps ($n = 6$). The Bermudan option $P_3$ is exerciseable at $m = 3$ points in time: $t_1 = T/3$, $t_2 = 2T/3$ and $t_3 = T$. The continuation value for node $(2, 1)$ is calculated from values at nodes $(4, 1)$, $(4, 2)$, and $(4, 3)$ (i.e. the continuation value for node $(n/m, 1)$ is calculated from nodes $(2n/m, 1 + k)$, where $k = 0, \ldots, n/m$).

### 4.6.2 Convergence of the Accelerated Binomial Option Pricing Model

The convergence performance of this method is shown in Figure 4.12. The graph on the left shows the individual option values for $P_1$, $P_2$ and $P_3$ as a function of the number of time steps in the binomial tree. The American put option price using the standard CRR binomial tree is then compared to the approximation of the accelerated binomial tree. As can be seen, the convergence of the method is oscillatory about the binomial price and does not appear to converge with increasing number of time steps. Following the logic of (Omberg 1987) it would appear that this is probably due to the non-coincidence of the exercise dates of $P_2$ and $P_3$. This oscillatory behaviour may be mitigated if the extrapolation is performed on $P_1$,$P_2$, $P_4$ (which can be exercised at times $T/4$, $T/2$, $3T/4$, and $T$), where the exercise dates of $P_1$ and $P_2$ are a subset of $P_4$.

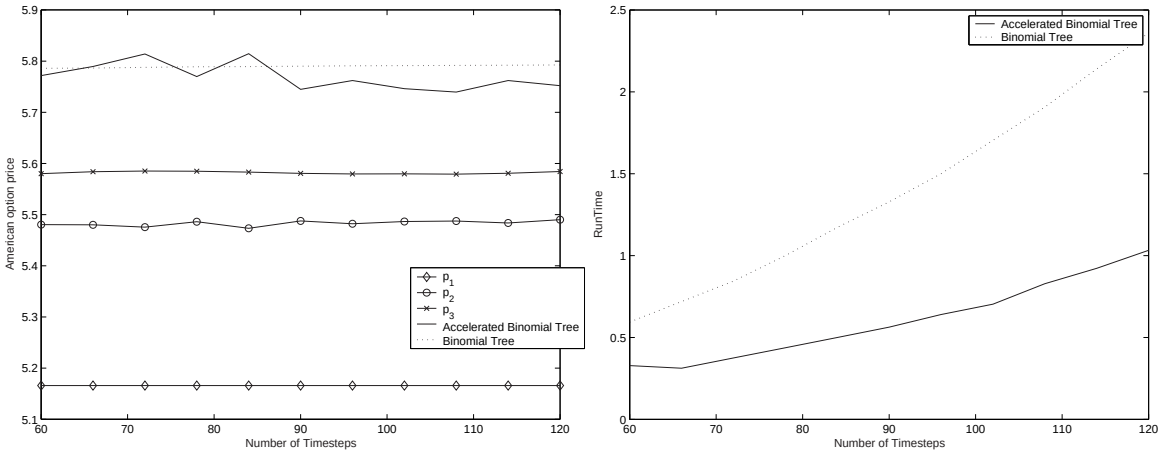


**Fig. 4.12:** The graph on the left shows the individual option values for $P_1$, $P_2$ and $P_3$ as a function of the number of time steps in the binomial tree. It also shows the standard CRR binomial put price approximation compared with the accelerated CRR binomial put price. The graph on the right shows that the runtime of the standard CRR binomial method is considerably higher for increasing time steps than that of the accelerated CRR binomial tree method.

## 4.7 Adaptive Mesh Models

The adaptive mesh model was developed in (Figlewski and Gao 1999). The mesh is designed to reduce the non-linearity errors that arise in lattice methods. (Reminder: the non-linearity error results from the propagation of the non-linear payoff function above and below the strike price, into the lattice by backward induction.) The method increases the resolution of the lattice in the critical areas where the price is most non-linear. The method can be used for a wide range of exotic options. For European options this non-linearity only occurs around the strike price at expiry. Therefore, the nodes that are most affected by the non-linearity are those nodes one time step before expiry in the region of the strike price.

This non-linearity error is amplified when the strike price does not coincide with a node. To address this issue, a finer lattice than the initial pricing tree is constructed one time step before expiry around the strike price (AMM-1). It is possible to iterate this approach a second time. A second level of refinement closer to the strike price can be constructed to improve the convergence rate further (this is called the adaptive mesh model with two levels of refinement (AMM-2)).

(Figlewski and Gao 1999) pointed out, that for American options, the non-linearity may be propagated at any time step because of the possibility of early exercise. The smooth-pasting condition may be used to reduce the non-linearity. However, the adaptive mesh model may also be used to reduce the error around the early exercise boundary. Consequently, an improvement in the approximation of the American put option price may be achieved if a fine lattice is created at each time step in the region around the early exercise boundary.

### 4.7.1 The Adaptive Mesh Model with One Level of Refinement (AMM-1)

We have applied AMM-1 to the non-linearity at expiry and not to the non-linearity near the early exercise boundary.

Initially we need to locate the closest node, $k$, to the strike price, $K$, at expiry. This node occurs within the initial lattice and not the adapted mesh. Node $k$ is chosen such that,

$$S_{(n,j)} \geq K, \quad j \geq k.$$

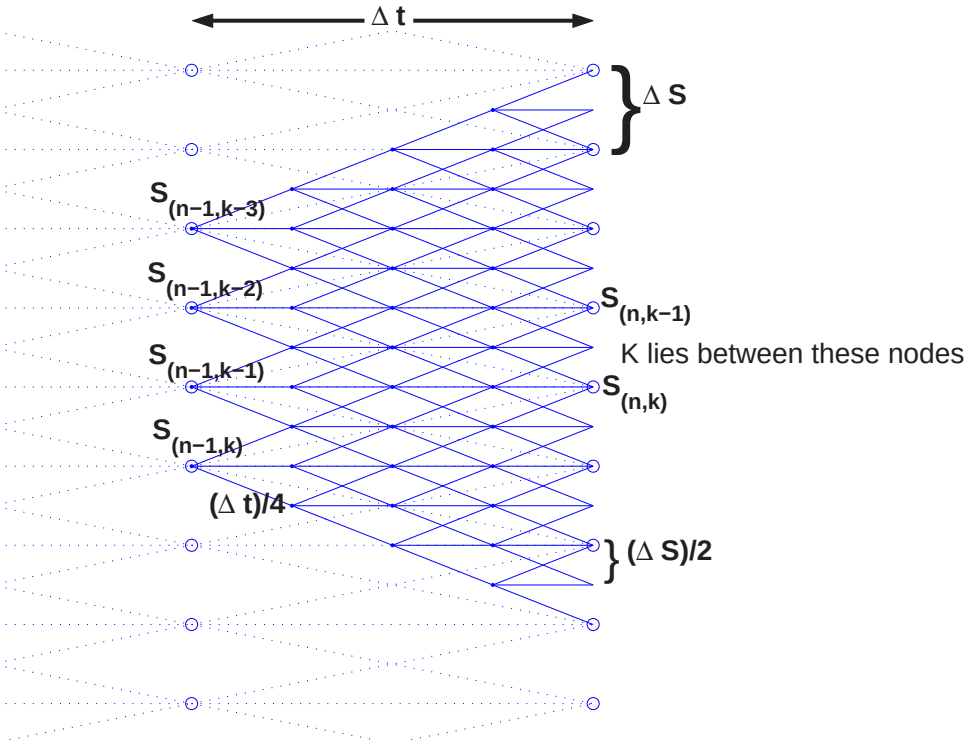


**Fig. 4.13:** A schematic of the adaptive mesh model for a standard trinomial tree. The schematic details the area around the strike price one time step before expiry. The strike price occurs between $S_{(n,k-1)}$ and $S_{(n,k)}$. The parent nodes connected to these two values are: $S_{(n-1,k-3)}$, $S_{(n-1,k-2)}$, $S_{(n-1,k-1)}$ and $S_{(n-1,k)}$.

The strike price is located between $S_{(n,k-1)}$ and $S_{(n,k)}$. By backward recursion the only nodes which are directly affected by these two are: $S_{(n-1,k-3)}$, $S_{(n-1,k-2)}$, $S_{(n-1,k-1)}$ and $S_{(n-1,k)}$. A fine mesh is constructed around these points. The size of the time step for the new mesh is $\Delta^* t = \Delta t/4$, which leads immediately to a new set of node values in the fine mesh.

Depending on the trinomial scheme used, new parameter values for the fine mesh are calculated, viz. $U^*$, $D^*$, $p_u^*$, $p_m^*$ and $p_d^*$. Using these parameter values, a new lattice $S^*_{(i,j)}$ (where $i = 0, \ldots, 4$ and $j = 1, \ldots, m_i^*$) is created. In the new lattice

$m_0^* = 7$, $m_1^* = 9$, $m_2^* = 11$, $m_3^* = 13$, and $m_4^* = 15$. The fine lattice lies on top of the coarse lattice ensuring that

$$\begin{aligned} S_{(n-1,k-3)} &= S^*_{(0,1)} \\ S_{(n-1,k-2)} &= S^*_{(0,3)} \\ S_{(n-1,k-1)} &= S^*_{(0,5)} \\ S_{(n-1,k)} &= S^*_{(0,7)}. \end{aligned}$$

Option valuation on the fine lattice is then completed to create the option values $P^*_{(0,j)}, j = 1, \ldots, 7$. The four option values on the coarse grid are set equal to the corresponding values on the fine lattice

$$\begin{aligned} P_{(n-1,k-3)} &= P^*_{(0,1)} \\ P_{(n-1,k-2)} &= P^*_{(0,3)} \\ P_{(n-1,k-1)} &= P^*_{(0,5)} \\ P_{(n-1,k)} &= P^*_{(0,7)}. \end{aligned}$$

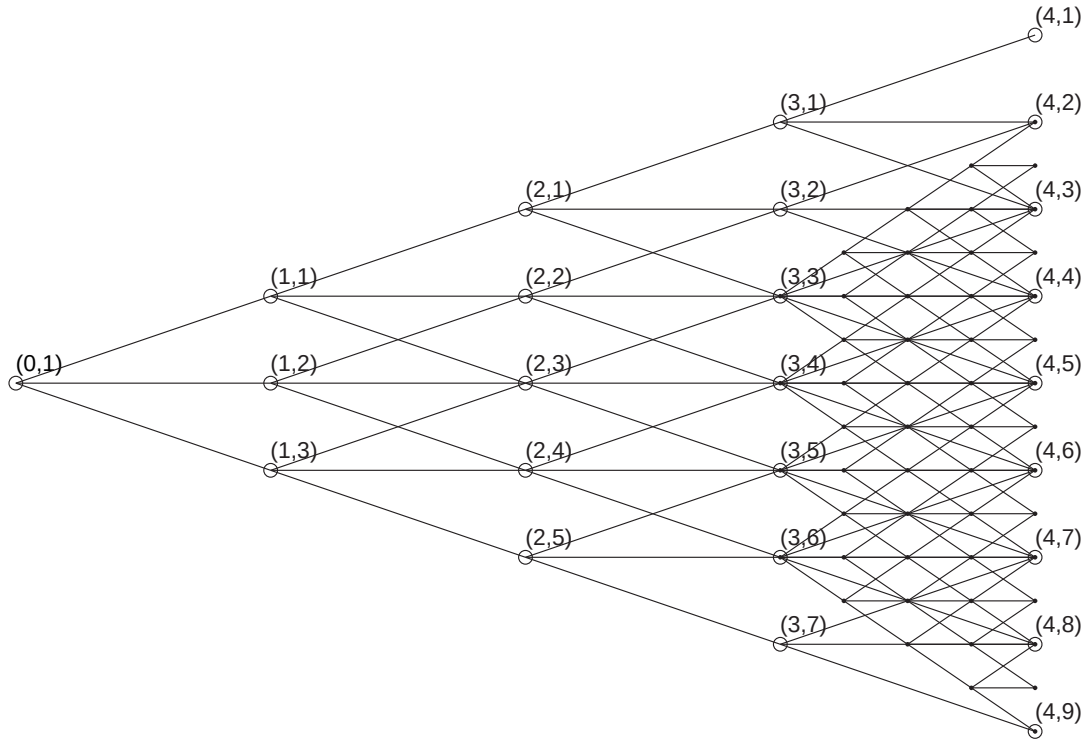


**Fig. 4.14:** The adaptive mesh model for a four time step trinomial tree. The strike price is located between nodes $(4, 5)$ and $(4, 6)$. The fine lattice is constructed emanating from nodes $(3, 3)$, $(3, 4)$, $(3, 5)$ and $(3, 6)$.

### 4.7.2 The Adaptive Mesh Model with Two Levels of Refinement (AMM-2)

A second level of refinement can be added to the last time step of the first adaptive mesh. The procedure for doing this is exactly the same as for the first level of refinement except that of course the new time step jump is now $\Delta^{**}t = \Delta^{*}t/4$, with a consequentially new set of value nodes.

### 4.7.3 Convergence of Adaptive Mesh Models

The effect that the adaptive mesh model has on the convergence rate is similar to that of the Black-Scholes modification. It is interesting to note that using the adaptive mesh on a trinomial tree only alters the rate at which the American put option price converges to a solution, and not the manner in which the American put price converges to a solution (see Figure 4.15). Thus the amplitude of the convergence of the American put option price estimate (as the number of time steps is increased) using the first level adaptive mesh model is less than the amplitude of the convergence of the American put option price estimate calculated using the standard Hull-White trinomial tree. This amplitude is reduced further when the second adaptive mesh is added.

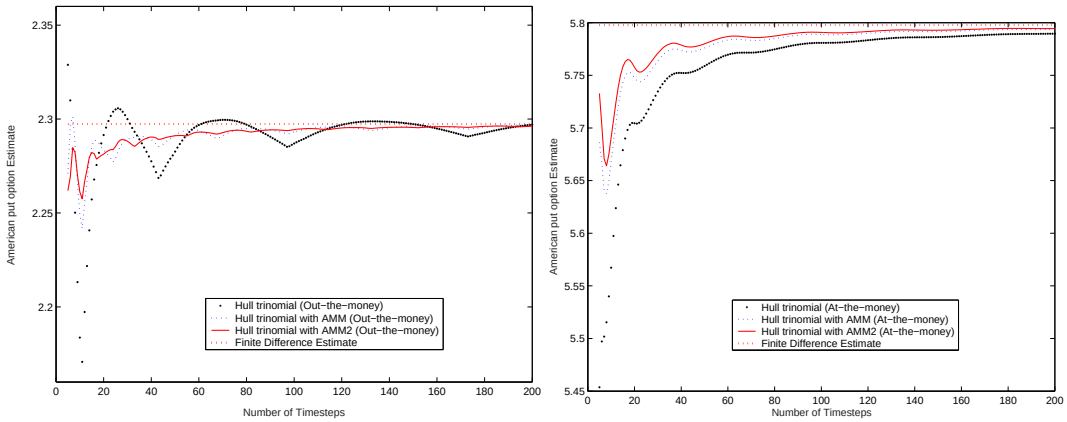


**Fig. 4.15:** The convergence of Hull-White trinomial tree with adaptive mesh model with one and two levels of refinement for an out-the-money and an at-the-money American put option.

# Chapter 5

# Monte Carlo Methods

## 5.1 Introduction

### 5.1.1 Mathematical Foundations

Numerical integration schemes typically approximate the value of an integral by partitioning the integration region into a set of discrete areas or volumes with associated weights e.g. quadrature methods (Abramowitz and Stegun 1970, Press, Teukolsky, Vetterling, and Flannery 1999). A sum of weighted function values, form an estimate of the total integral value. The specific quadrature scheme determines both the spacing of the areas (volumes) and their associated weights. Monte Carlo integration uses a similar approach to this, except that the weights are chosen to be equal and the terminal discrete points/areas/volumes are sampled randomly from the distribution associated with the integration region (see (Glasserman 2005) for a comprehensive treatment of Monte Carlo methods.) Under certain technical conditions the law of large numbers may be invoked to ensure that the integral approximation tends towards the correct value as the number of random terminal values increases to infinity.

The integral of a function $f(x)$ with an associated probability density function $h(x)$, such that $\int_A h(x)dx = 1$,

$$I_A = \int_A f(x)h(x)dx,$$

is approximated by randomly sampling numbers $X_1, \ldots, X_n$ from the density function $h(x)$ over the integration region $A$. The Monte Carlo estimate[1], $\widehat{I}_A^{(M)}$ is then the arithmetic mean of the function values,

$$\widehat{I}_A^{(M)} = \frac{1}{M}\sum_{i=1}^{M} f(X_i).$$

The convergence of the Monte Carlo method is of order $\mathcal{O}(1/M^{1/2})$ where $M$ is the number of samples used. In the one-dimensional case this convergence is slow, even when compared to the simple quadrature techniques such as the trapezoidal

[1] All Monte Carlo estimates are denoted by a hat, where $^{(M)}$ denotes the number of samples used to generate the estimate.

rule which has a convergence rate in one dimension of $\mathcal{O}(1/M^2)$. However, the convergence rates of most numerical integration schemes suffer from the "curse of dimensionality", i.e. the convergence rates decrease as the dimension of the problem increases. The advantage of Monte Carlo methods is that the convergence rate remains constant even as the dimension of the problem increases (Glasserman 2005).

### 5.1.2 Monte Carlo Methods for the European Put

(Cox, Ross, and Rubinstein 1979) showed, amongst other things, that the existence of a replicating portfolio for a European option implies that the value of the option can be expressed as the discounted expected value of its terminal payoff. They approximate the value process of the stock price by constructing a lattice with the price distribution embedded therein. The option payoff is calculated from the terminal lattice values and the expectations from the lattice construct. A discounted expectation follows trivially (see Chapter 4). The application of Monte Carlo methods to calculate this expected value was first introduced in (Boyle 1977). The value of a European put option $p(t, S)$ can be estimated by simulating the distribution of the terminal option values. Trivially, this translates into simulating the terminal stock price distribution.

The lognormal distribution of terminal stock prices can be simulated using the solution $S_t$ (equation (2.2)) of the SDE of geometric Brownian (equation (2.1)). The solution is in terms of the Brownian motion $W_{T-t}$. Under a given measure, Brownian motion increments are normally distributed with a mean of zero and a variance equal to the time increment,

$$W_t - W_s \sim \mathcal{N}(0, t-s), \quad t > s,$$

and so the Brownian motion can be simulated using $W_{T-t} = \sqrt{(T-t)}\, Z$ where $Z \sim \mathcal{N}(0,1)$. We can simulate $M$ terminal stock prices, $S_T^{(m)}, m = 1, \ldots, M$ from $M$ simulated Brownian motions, $W_{T-t}^{(m)}, m = 1, \ldots, M$. The Monte Carlo estimate, $\widehat{p}(t)$, of the European option price is given by,

$$\widehat{p}(t) = \exp\{-r(T-t)\} \frac{1}{M} \sum_{m=1}^{M} \left(K - S_T^{(m)}\right)^+$$

where $r$ is the riskless rate of return over $(T-t)$. The closed form solution (2.6) for the European put option suggests that the Monte Carlo technique should converge. When considering American put options, we are no longer concerned with terminal stock prices, because early exercise is a possibility. In order to value the American put, we need to generate a large number of discrete time stock price paths and dynamically program rules to determine whether early exercise is optimal at any stage during the path.

### 5.1.3 Generating Monte Carlo Sample Paths

We discretise the life of the option , $T-t$ into $n$ equal intervals of size $\Delta t = (T-t)/n$, with associated time steps, $t = t_0 < t_1 < \cdots < t_n = T$. Monte Carlo sample paths

are created by simulating the geometric Brownian motion process (2.1) exactly at each of these discrete times. This is achieved by using the solution (2.2) adapted to each time step. This automatically forms a Markov chain. Consequently, the percentage increments,

$$\frac{S_{t_1} - S_{t_0}}{S_{t_0}}, \frac{S_{t_2} - S_{t_1}}{S_{t_1}}, \ldots, \frac{S_{t_n} - S_{t_{n-1}}}{S_{t_n}}$$

are independent and solution (2.2) can be written in terms of Brownian motions $\{W_i\} \equiv \{W_{t_i}\}$, $i = 1, \ldots, n$ as,

$$S_i = S_{i-1} \exp\big\{\big(r - \sigma^2/2\big)(t_i - t_{i-1}) + \sigma\big(W_i - W_{i-1}\big)\big\}, \tag{5.1}$$

where $i = 1, \ldots, n$ and $S_i \equiv S_{t_i}$. $W_i - W_{i-1}$ is simulated using $\sqrt{(t_i - t_{i-1})}\, Z_i$, where $Z_i \sim \mathcal{N}(0, 1)$ for all $i = 1, \ldots, n$. Solving equation (5.1) recursively, creates a stock price path, $\{S_i\}_{i=0,\ldots,n}$ whose marginal distributions matches the lognormal distribution of the geometric Brownian process (2.1). For optimal implementation in Matlab we define the vector of values,

$$S_i = S_0 \exp\Big\{\big(r - \sigma^2/2\big)(t_i - t_0) + \sigma \sum_{j=1}^{i} \sqrt{(t_j - t_{j-1})}\, Z_j\Big\}$$

where $i = 1, \ldots, n$. This creates one sample path of the stock price. We need to simulate many stock price paths to approximate the lognormal distribution at each time step. This is achieved by sampling $n \times M$ independent standard normal random numbers $Z_i^m$, where $Z_i^m \sim \mathcal{N}(0, 1)$ for all $i = 1, \ldots, n$ and $m = 1, \ldots, M$. These numbers can then be used to generate $M$ stock price paths $\{S_i^{(m)}\}$.

In order to price an American put option we need to determine whether the sample path has crossed the early exercise boundary. Of course, the early exercise boundary is unknown and also needs to be estimated. This is difficult to do, however, because we need to implement a backward recursion method using Monte Carlo, which is forward looking.

### 5.1.4 Difficulties Encountered Using Monte Carlo Methods for Pricing American Put Options

The right to exercise the option at any time prior to expiry implies that we are no longer just estimating the discounted expected value of the terminal payoff. The Snell's envelope representation (2.9) defines the price of the American put option as the maximum value over all possible stopping times $\tau \in [t, T]$. But stopping times are $\mathscr{F}_t$-measurable, i.e. an exercise decision is made at $\tau$ using information only known at $\tau$. This results in a non-trivial maximisation problem.

A naive method (Tilley 1993a) can be implemented by replacing the optimal stopping time for each sample path, $\tau^*(m), m = 1, \ldots, M$, with a time $\tau_*(m)$ such that this time maximises the discounted exercised value for each stock price path. This is no longer a stopping time and since it uses the information inherent in the entire sample path, $\big\{S_i^{(m)} : i = 0, \ldots, n\big\}$, for all $m = 1, \ldots, M$, in order to determine

its value. We generate sample paths, $\{S_i^{(m)}\}$, each with $n$ time steps and then use equation (5.1.2) to determine $\tau_*(m)$. Then a naive estimate of the American put option price is given by,

$$\widehat{P}^M_{\text{naive}}(t) = \frac{1}{M}\sum_{m=1}^{M} e^{-r(\tau^*(m)-t)}\big(K - S^m_{\tau_*(m)}\big)^+.$$

This exercise strategy results in a systematic upward bias in the premium estimate since the exercise time is chosen to yield the maximum payoff per path and no other strategy can yield a higher premium estimate.

A common approach to evaluating stopping times is to use Bellman's principle. In this case, the price of the American put option, $P(t)$ is approximated as a discrete dynamic programming problem which is then solved using backward recursion. The difficulty with using this approach in a Monte Carlo environment is that the Monte Carlo technique is forward recursive. Using Bellman's principle, we move backwards from expiry making exercise decisions at each time step for each sample stock price path by comparing the intrinsic value with the continuation value.

An important consideration when implementing a Monte Carlo method for American option pricing is understanding the bias which is introduced by the method. Some methods have a upward bias, a downward bias, or a mix of both biases. A upward bias usually results from the use of information about the future to make early exercise decisions (an example of this is the naive method above). Backward recursion algorithms often have this type of bias. A downward bias is often introduced by making suboptimal early exercise decisions. By separating methods which introduce opposing biases we may calculate an upper and a lower estimate for the option premium.

### 5.1.5 Dynamic Programming for Monte Carlo Methods

The Snell's envelope representation is solved as the dynamic programming problem given by equations (2.11) and (2.12). The value at expiry is the intrinsic value or payoff function $g(T, S_T)^+$ where $g(\cdot)$ is the payoff function (2.7). Prior to expiry, at each time step we have two choices: exercise the option or hold onto the option. We calculate successive option value estimates by backward recursion. Unlike the European option estimation we are required to make exercise decisions at each point in time. The estimates are given by, $\widehat{P}_i(S) : i = n-1, \ldots, 0$.

The value of exercising the option at each time step is the *intrinsic value*, $g(t_i, S_i) : i = 0, \ldots, n$ where $g(\cdot)$ is the payoff function (2.7) . Monte Carlo allows calculation in more than one dimension. The individual discount factors for each time period may depend on a stochastic interest rate which could be state and time dependent. In the Black-Scholes environment the interest rate, $r$, is a constant riskless rate of return. Consequently, the standardised discount factor between $t_i$ to $t_{i+1}$ is dependent only on the time step size,

$$d_i = \exp\{-r(t_{i+1} - t_i)\}, \tag{5.2}$$

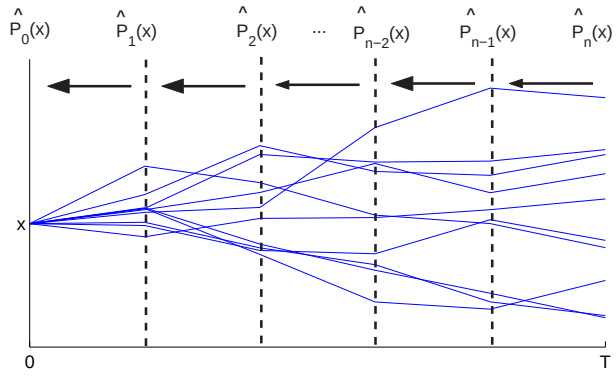


**Fig. 5.1:** Once the sample paths have been constructed, the recursive option estimates, $\widehat{P}_i(S), i = n-1, \ldots, 0$ are evaluated using both the option values one time step ahead $\widehat{P}_{i+1}(S)$ and the intrinsic value.

where $i = 0, \ldots, n-1$. The total discount factor, $D_i$, discounts any value at time $t_i$, from $t_i$ to an earlier time $t_0$ such that $D_i = \prod_{j=1}^{i} d_j$. Then,

$$D_i = \exp\{-r(t_i - t_0)\},$$

where $i = 1, \ldots, n$. The discrete dynamic programming problem (equations (2.11) and (2.12)) for the American put option, $P_i(S)$ reduces to,

$$P_n(S) = g(t, S) \tag{5.3}$$

$$P_{i-1}(S) = \max\{g(t, S), H_i(S)\}, \quad i = n, \ldots, 1. \tag{5.4}$$

where the *continuation value*, $H_i(S) \equiv d_{i-1}\mathbb{E}\big[P_i(S_i)\big|S = S_{i-1}\big] : i = 0, \ldots, n-1$, is the value of the option if it is unexercised at $t_{i-1}$. Estimating the continuation value is not straightforward. The Monte Carlo methods discussed below use different approaches to estimate this value and it can be used to either approximate the early exercise boundary (Grant, Vora, and Weeks 1996), or to backward propagate the option value by approximating the optimal exercise time (Barraquand and Martineau 1995, Carriere 1996, Longstaff and Schwartz 2001, Tilley 1993a, Tsitsiklis and Van Roy 2001).

### 5.1.6 The Early Exercise Boundary and Optimal Stopping for Monte Carlo Methods

A majority of the methods do not explicitly calculate the early exercise boundary. The option values are calculated recursively without mention of the early exercise boundary. In these methods, exercise occurs when the continuation value drops below the intrinsic value. The optimal stopping time, $t_{\tau^*(m)}$ is chosen as the earliest time step that this occurs,

$$\tau^*(m) = \min\{t_i \leq t_n \big| \widehat{H}_i\big(S_i^{(m)}\big) \leq g\big(t_i, S_i^{(m)}\big)\},$$

where $\widehat{H}_i(\cdot)$ is estimated using Monte Carlo. Once all the stopping times have been calculated, the Monte Carlo estimate, $\widehat{P}(t)$ is calculated using,

$$\widehat{P}(t) = \frac{1}{M}\sum_{i=1}^{M} D_{\tau^*(m)}\big(K - S^{(m)}_{\tau^*(m)}\big)^+.^2 \tag{5.5}$$

The method of (Grant, Vora, and Weeks 1996) used Monte Carlo simulation to explicitly approximate the early exercise boundary. The discrete approximation of the early exercise boundary, $\{\widehat{b}_i\}_{i=0,\ldots,n-1}$, is calculated by comparing intrinsic values with approximate continuation values at each time step for a variety of values of the underlying. The exercise boundary at expiry is equal to the strike price, $b_n = K$. At each preceding time step an approximation of the early exercise boundary value is found by searching for the stock price which makes the intrinsic value equal to the continuation value,

$$\big\{\widehat{b}_i = S \big| g(t,S) = \widehat{H}_i(S)\big\}. \tag{5.6}$$

Each continuation value, $\widehat{H}_j(\cdot)$, is calculated using Monte Carlo methods and the already calculated portion of the early exercise boundary, $\{\widehat{b}_i\}_{i=j+1,\ldots,n}$. Once the entire boundary has been calculated, independent stock price paths are generated using standard Monte Carlo. The optimal stopping times for these new sample paths are evaluated with respect to the early exercise boundary, $\{\widehat{b}_i\}$ (see Figure 5.2). These optimal stopping times, $\tau^*(m)$ for all $m = 1, \ldots, M$, are defined by,

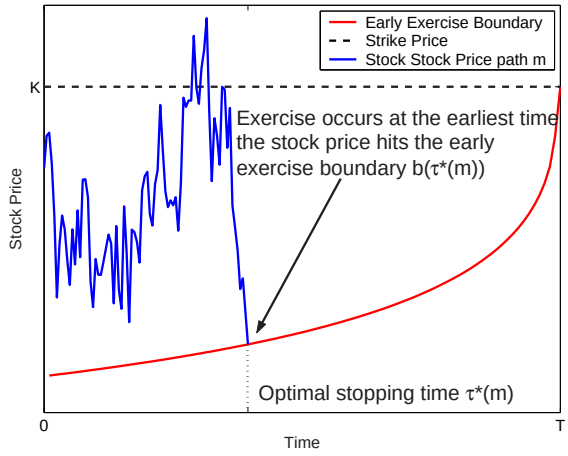


**Fig. 5.2:** The continuation region is separated from the exercise region by the early exercise boundary.

$$\tau^*(m) = \min\big\{t_i \le t_n \big| S_i^{(m)} \le \widehat{b}_i\big\}.$$

The Monte Carlo estimate, $\widehat{P}(t)$ is then calculated using equation (5.5).

[2] Here, $S^{(m)}_{\tau^*(m)} = S_i^{(m)}$ where $\tau^*(m) = t_i$.

## 5.2 Tilley's Bundling Algorithm

The first application of Monte Carlo techniques to the estimation of the American put option price was in (Tilley 1993a). The Monte Carlo estimate $\widehat{P}(t)$ uses the dynamic programming algorithm equations (5.3) and (5.4). The continuation values, $\widehat{H}_i(S) : i = 1, \ldots, n$, are estimated by applying a crude regression technique at each time step to the $M$ sample paths $\{S_i^{(m)}\}_{i=0,\ldots,n}$ where $m = 1, \ldots, M$.

The terminal values, $P(T)$ are calculated in the usual fashion using equation (5.3). At each preceding time step, $t_i : i = n-1, \ldots, 1$, we order the stock prices $S_i^{(m)} : m = 1, \ldots, M$. Once in descending order, the stock prices are divided into distinct bundles each containing $r$ stock prices. If there are $a$ such distinct bundles, $A_j : j = 1, \ldots, a$, then $a \times r = M$. An average continuation value, $\widehat{H}_i(A_j)$, is associated with each distinct bundle, at each time step $t_i$. This is calculated by taking the average of the discounted option values one time step ahead for each stock price in the bundle,

$$\widehat{H}_i\big(A_j\big) = \frac{1}{r} \sum_{m=1}^{M} d_i \widehat{P}_{i+1}\big(S_{i+1}^{(m)}\big) \mathbb{I}_{\{S_i^{(m)} \in A_j\}}. \tag{5.7}$$

These average continuation values are used to make exercise decisions for each individual sample path in each bundle. The associated average being compared with the intrinsic value at each stock price, $S_i^{(m)}$, only. Hence the option values are calculated using,

$$\widehat{P}_i\big(S_i^{(m)}\big) = \begin{cases} g\big(t_i, S_i^{(m)}\big) & \text{if } g\big(t_i, S_i^{(m)}\big) \geq \widehat{H}_i\big(A_j\big) \\ d_i \widehat{P}_{i+1}\big(S_{i+1}^{(m)}\big) & \text{if } g\big(t_i, S_i^{(m)}\big) < \widehat{H}_i\big(A_j\big) \end{cases}, \tag{5.8}$$

for each $S_i^{(m)} \in A_j$, and where $m = 1, \ldots, M$.

Using the average continuation value for each bundle to make exercise decisions introduces a *transition zone* instead of an exact early exercise boundary. Each transition zone is a subset of the ordered stock prices. There may be contradicting exercise decisions between stock paths in neighbouring bundles, e.g. consider two stock prices $S_i^{(m_1)} < S_i^{(m_2)}$, each in a separate bundle. Due to the average nature of the exercise decision, exercise may be calculated to be optimal at stock price $S_i^{(m_2)}$ but not at $S_i^{(m_1)}$, even though the intrinsic value at $S_i^{(m_1)}$ is obviously greater than the intrinsic value at $S_i^{(m_2)}$. This is because the intrinsic value is being compared with an average calculated continuation value.

The effect of the transition zone can be ameliorated by introducing a *sharp boundary* which increases the convergence rate of the method and places fewer restrictions on the choice of the bundling parameters $a$ and $r$. Methods for choosing the sharp boundary were discussed in (Tilley 1993b). The initial idea was to pick the first stock price below which decisions to exercise dominate decisions not to exercise. The sharp boundaries can then be used to make exercise decisions to calculate the option estimates, $\widehat{P}_i(S) : i = n-1, \ldots, 1$. The final approximation $\widehat{P}(t)$ is calculated by taking the discounted arithmetic mean of the approximate option values one time

step in the future,

$$\widehat{P}(t) = \frac{1}{M}\sum_{m=1}^{M} d_0 \widehat{P}_1\big(S_1^{(m)}\big). \tag{5.9}$$

Analysing the different sources of bias for this method is difficult due to the manner in which exercise decisions are made. It is unclear exactly how the calculation of average continuation values over each bundle affects the bias of the estimate. The rest of this section details the algorithm, and a comprehensive discussion of the bias.

### 5.2.1 Calculating Exercise-or-Continue Indicator Values

In (Tilley 1993a) an exercise-or-continue indicator function, $z_i(S)$, was introduced which gave the early exercise time for each stock price path. The indicator function takes the value one at only one time, and is zero everywhere else. Once the indicator function has been evaluated for each path, the option is approximated by,

$$\widehat{P}(t) = \frac{1}{M}\sum_{m=1}^{M}\sum_{i=1}^{n} D_i z_i(S_i^{(m)}) g(t_i, S_i^{(m)}).$$

This indicator function approach is, however, unnecessarily time consuming because we store the option values, $\widehat{P}_i(S)$, through the backward recursion process. Consequently, the final option value can be calculated using equation (5.9).

The option values at expiry are equal to the intrinsic value, $\widehat{P}_n(S_n^{(m)}) = g(t_n, S_n^{(m)})$ for all $m = 1, \ldots, M$. The ensuing dynamic programming problem (5.4) is evaluated using eight steps at each point in time $t_i : i = n-1, \ldots, 1$. At each point in time,

**Step 1:** Reorder the stock prices $S_i^{(m)} : m = 1, \ldots, M$, from highest to lowest.

**Step 2:** Compute the intrinsic value for each stock price, $g(t_i, S_i^{(m)}) : m = 1, \ldots, M$.

**Step 3:** Partition the $M$ ordered stock prices into $a$ distinct bundles, $A_j : j = 1, \ldots, a$ containing $r$ paths in each bundle. Define the *bundling parameter*, $\alpha$, such that,

$$a = M^{\alpha}, \quad r = M^{1-\alpha}.$$

Then $M = ar$. The value of the bundling parameter $\alpha$ influences the convergence of the algorithm and is discussed in the next section.

**Step 4:** Calculate the continuation value for each bundle, $\widehat{H}_i(A_j) : j = 1, \ldots, r$, using equation (5.7). The continuation value for each stock price path is then the continuation value of the bundle it belongs to.

**Step 5:** Calculate a temporary indicator variable, $y_i(\cdot)$, for each stock price,

$$y_i\big(S_i^{(m)}\big) = \begin{cases} 1 & g\big(t_i, S_i^{(m)}\big) \geq \widehat{H}_i\big(S_i^{(m)}\big) \quad \text{Exercise} \\ 0 & g\big(t_i, S_i^{(m)}\big) < \widehat{H}_i\big(S_i^{(m)}\big) \quad \text{Continue} \end{cases} \tag{5.10}$$

where $m = 1, \ldots, M$.

**Step 6:** Exercise decisions are given by sequences of zeros and ones generated by the temporary indicator variable. The transition zone begins with the first "Exercise" indication and ends with the last "Continue" indication, e.g.

$$\begin{array}{cc} & \downarrow \text{ Transition zone } \downarrow \\ y_i\big(S_i^{(m)}\big) & 0\ldots\|000001\|001111\|000111\|011111\|1\ldots1 \\ & A_k \quad A_{k+1} \quad A_{k+2} \quad A_{k+3} \end{array}$$

Eliminating this transition zone can improve the convergence. It therefore makes sense to replace the transition zone with a sharp boundary, $\widehat{b}_i$, which is an approximation of the early exercise boundary at each time $t_i$. Notice that the transition zone contains a number of false indicators. In the example above the first "1" which marks the beginning of the transition zone is a false exercise indicator. The final "0" is also a false continue indicator. The first string of "1"s which is longer than any subsequent string of "0"s is effectively the true early exercise boundary indicator. The method for calculating $\widehat{b}_i$ (Tilley 1993a) is to choose the unique value of $m$, say $k^*$, which corresponds to the position of the first temporary indicator value of "1" in this particular sequence, e.g.

$$\begin{array}{cc} & \textit{Sharp Boundary, } k^* \\ & \big\downarrow \\ y_i\big(S_i^{(m)}\big) & 0\,0\,0\,0\,1\,0\,0\,1\,1\,1\,1\,0\,0\,0\,1\,1\,1\,0\,1\,1\,1\,1\,. \end{array}$$

This method of determining the sharp boundary is both time consuming and a little arbitrary. The value of the option within this method is relatively insensitive to the exact position of the early exercise boundary. An alternative method suggested in (Tilley 1993b) is to move all the zeros to the left and all the ones to the right. Then $k^*$ is the number of zeros in the temporary indicator variable,

$$k^* = \sum_{m=1}^{M} \mathbb{I}_{y_i\left(S_i^{(m)}\right)=0} \ .$$

Either method yields a value for $k^*$. The stock price which corresponds to $m = k^*$, $S_i^{k^*}$, is an approximation for the early exercise boundary at any time $t_i$.

**Step 7:** Decide which sharp boundary method to use. Then, define a new exercise-or-continue indicator variable, $\bar{y}_i(\cdot)$

$$\bar{y}_i\big(S_i^{(m)}\big) = \begin{cases} 1 & \text{for } m \geq k^* \\ 0 & \text{for } m < k^*. \end{cases}$$

Using the original method of (Tilley 1993a) and the same example as above,

$$\begin{array}{cc} & \text{Sharp Boundary, } k^* \\ & \big\downarrow \\ \bar{y}_i\big(S_i^{(m)}\big) & 0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,. \end{array}$$

**Step 8:** The approximate option value at $t_i$, $\widehat{P}_i(\cdot)$, is given by

$$\widehat{P}_i\big(S_i^{(m)}\big) = \begin{cases} g\big(t_i, S_i^{(m)}\big) & \text{if } \bar{y}_i\big(S_i^{(m)}\big) = 1 \\ d_i \widehat{P}_{i+1}\big(S_{i+1}^{(m)}\big) & \text{if } \bar{y}_i\big(S_i^{(m)}\big) = 0. \end{cases}$$

The final approximate option value, $\widehat{P}(t)$, is calculated using (5.9).

### 5.2.2 Bias of Tilley's method

We introduce a hypothetical estimate, $\widehat{P}^M_{\text{Exact}}$, for the price of the American put option which uses $M$ sample paths but *makes the correct early exercise decisions*. This leads to an unbiased estimate, by definition. In Section 5.1.4 we defined a naive estimate, $\widehat{P}^M_{\text{Naive}}$, in equation (5.3). Because the naive estimate is computed with perfect hindsight it always returns the maximal estimate. The naive estimate dominates both the hypothetical estimate and Tilley's estimate

$$\widehat{P}^M_{\text{Exact}} \leq \widehat{P}^M_{\text{Naive}}, \quad \widehat{P}^M_{\text{Tilley}} \leq \widehat{P}^M_{\text{Naive}},$$

although it is unclear how the bias of $\widehat{P}^M_{\text{Tilley}}$ compares with $\widehat{P}^M_{\text{Exact}}$. Tilley's method contains two sources of bias: one upward and one downward (Tilley 1993b).

The use, at each point in time, of the complete information set, up to expiry, produces an upward bias in the estimate. This bias is common to all techniques using dynamic programming because the continuation value is calculated from values one time step in the future. The downward bias in Tilley's method results from using an imprecise rule to select the sharp boundary. Suboptimal exercise decisions invariably result in an estimate which is downward biased. The exact position of the sharp boundary does not, however, have a significant impact on the option premium (Tilley 1993b). Although this bias cannot easily be measured, it has a slight cancelling effect which improves the convergence of the estimate.

### 5.2.3 Convergence of Tilley's method

Tilley's method produces an estimate for the American put option price which converges as the number of time steps, $n$, tends towards infinity. In general it is unclear whether this estimate has an upward or downward bias. We will instead investigate the factors which influence the convergence rate of this method: the number of time steps $n$, the number of stock price sample paths $M$, the bundling parameter $\alpha$ and the choice of definition of the sharp early exercise boundary.

#### Sample Size Convergence

The dominant bias in any Monte Carlo methods will determine whether it converges from above or below as the number of stock price paths increases. Tilley's method has offsetting biases, which make the nature of the convergence unclear. Figure 5.3 shows no discernable convergence bias. Hence, all we can say is that the estimate should tend toward the true value as the number of sample paths increases.

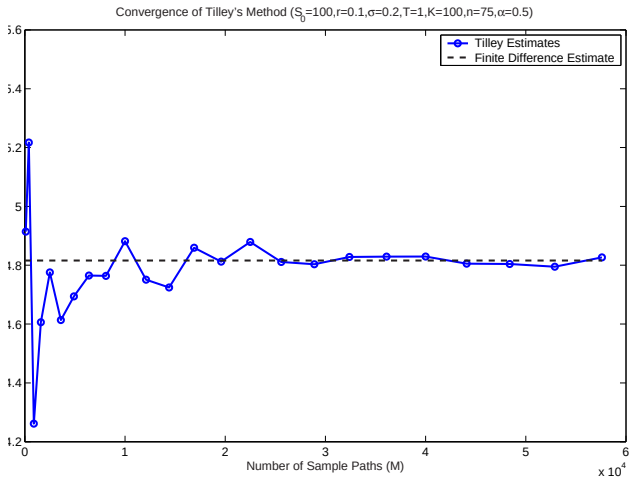


**Fig. 5.3:** Convergence behaviour of Tilley's estimates as the number of sample paths, $M$, increases. There is a fixed number of time steps ($n = 75$) for each sample path and a fixed bundling parameter ($\alpha = 0.5$). The Crandall-Douglas finite difference value is assumed to be the true value of the option.

### Time step Convergence

The majority of numerical methods used for pricing the American put option converge to the true value from below, as the number of time steps increases. The limit of any numerical method will be the continuous time process which, in the case of the American put option, triggers exercise as soon as the sample path hits the early exercise boundary. Any discrete approximation of the continuous time process has an infinite number of continuous paths between each discrete realisation point which could have hit the boundary. This results in (theoretically) an infinite number of missed early exercise opportunities (see Figure 5.4). From Figure 5.5 we can see that the convergence in Tilley's method is from below.

### Bundling Parameter $\alpha$ and the Transition Zone

The bundling parameter $\alpha$ dictates the number of bundles, where this number lies between 1 and $M$. A choice of $\alpha = 0$ coincides with one bundle containing $M$ sample stock paths, while $\alpha = 1$ produces $M$ bundles each containing one sample stock price path. A choice of $\alpha$ may introduce one of two errors:

1. A small $\alpha$ (which produces a small number of bundles) results in an over-averaging of the continuation value. There are too many sample paths in each bundle which results in an over-aggregation error.

2. A large $\alpha$ (which produces a large number of bundles) reduces the accuracy of the mathematical expectation, because too few samples are used.

A balance must be found when deciding on a value for $\alpha$. If the number of sample stock price paths is increased, the effect of the bundling parameter is mitigated. For

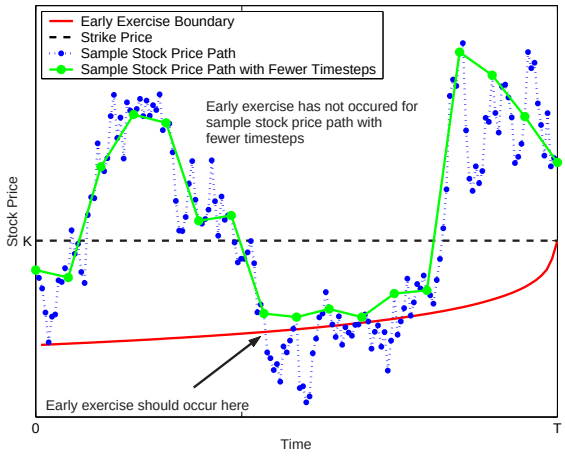


**Fig. 5.4:** The stock price path with fewer time steps has a higher probability that an early exercise decision will be missed.

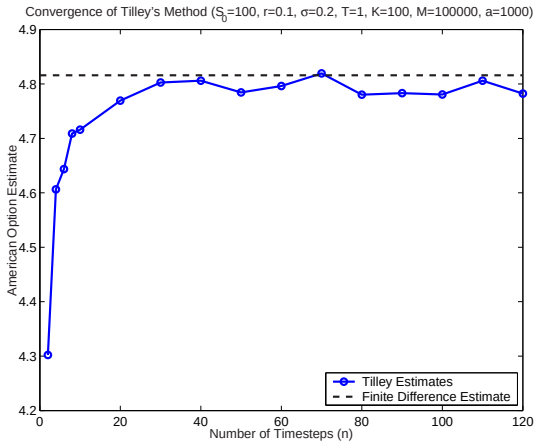


**Fig. 5.5:** The convergence behaviour for Tilley's estimates as the number of time steps, $n$, is increased. There is a fixed number of sample paths ($M = 100000$) for each estimate and a fixed bundling parameter ($\alpha = 0.5$). The Crandall-Douglas finite difference value is again assumed to be the true value of the option.

a large number of sample paths, the Tilley estimate converges to the true value of the option. Figure 5.6 shows that, empirically, an $\alpha \approx 0.5$ yields the most stable results.

Removal of the transition zone by introducing a sharp boundary improves the convergence behaviour of the method. It, simultaneously, also allows for a larger stable range of the bundling parameter. The presence of a transition zone forces $\alpha$ into a range of $[0.45, 0.55]$, while replacing this with a sharp boundary increases this range to $[0.4, 0.9]$ (see Figure 5.6). The lower the choice of $\alpha$, the faster the approximation. Consequently, a sharp boundary with $\alpha = 0.5$ is optimal.

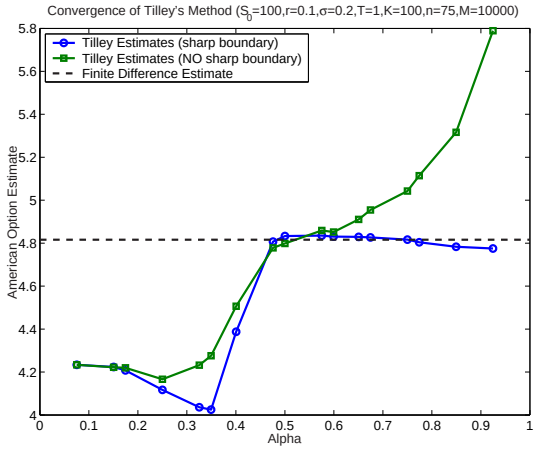


**Fig. 5.6:** The option estimates have been calculated using the same set of random numbers. When a transition zone exists, the estimate is far more sensitive to a choice of $\alpha$.

## 5.3 The Grant, Vora and Weeks Method

The algorithm suggested by Grant, Vora and Weeks in (Grant, Vora, and Weeks 1996) (GVW) uses an adaptation of equations (5.5) and (5.6). The algorithm is used to produces a discrete estimate of the early exercise boundary, $\{\widehat{b}_i\}_{i=1,\ldots,n}$, instead of calculating explicit continuation values, $\widehat{H}_i(S_i^{(m)})$, at each time step. Independent sample paths $\{S_i^{(m)}\}_{i=0,\ldots,n} : m = 1,\ldots,M$ are subsequently generated once the early exercise boundary has been estimated. A value for the put price corresponding to each sample path is calculated using the early exercise boundary and a Monte Carlo estimate, $\widehat{P}(t)$, is the average of these estimates over $M$.

One advantage GVW has over Tilley's bundling algorithm is that it can be easily applied to higher dimensions. There is no straightforward extension of Tilley to higher dimensions. The GVW method, however, reduces to a search for the specific state at each time step where early exercise is optimal. The rest of this section details the algorithm to locate the early exercise boundary, and then discusses the bias and convergence characteristics of the method.

### 5.3.1 The Early Exercise Boundary at Each Time step

The early exercise boundary $\widehat{b}_i$ at time $t_i$ is defined be equation (5.6). The GVW algorithm is backward recursive. The boundary is known at $t_n : b_n = K$ and the algorithm then calculates earlier estimates recursively by solving for the root of the function $f(b)$,

$$f(b) = g(t_i, b) - \widehat{P}_i(b).$$

$\widehat{P}_i(\cdot)$ is an independent Monte Carlo estimate, calculated at $t_i$, originating at some $S_i$, and using the existing estimate of the early exercise boundary, $\{\widehat{b}_j\}_{j=i+1,\ldots,n}$.

For each Monte Carlo estimate $\widehat{P}_i(b)$, we need to generate $M'$ stock price sample

paths $\{S_j^{(m)}\}_{j=i,\ldots,n} : m = 1,\ldots,M'$ conditional on $S_i^{(m)} = b$ for all $m$,

$$S_j^{(m)} = b\exp\Big\{\big(r-\sigma^2/2\big)(t_j-t_i) + \sigma\sum_{k=i+1}^{j}\sqrt{t_k-t_{k-1}}\,Z_k^{(m)}\Big\},$$

where $j = i+1,\ldots,n$ and $Z_k^{(m)} \sim \mathcal{N}(0,1)$. The same set of random numbers is used for each successive value of $S_j^{(m)}$.
The payoffs are decided by calculating an optimal exercise index $\tau(m)$ where,

$$\tau(m) = \min\big\{t_{i+1} \le t_j \le t_n \big| S_j^{(m)} \le \widehat{b}_j\big\},$$

where $\{\widehat{b}_j\}_{j=i+1,\ldots,n}$ are the early exercise boundary estimates calculated using Monte Carlo in previous steps. Then,

$$f(b) = g(t_i,b) - \frac{1}{M'}\sum_{m=1}^{M'}\exp\big\{-r(\tau(m)-t_i)\big\}g\big(\tau(m), S_{\tau(m)}^{(m)}\big). \tag{5.11}$$

Equation (5.11) does not really simplify the process. The estimation of the expectation using Monte Carlo creates a numerically induced instability. Any bracketing method requires smoothness in the estimate. The Monte Carlo estimate affects the monotonicity of the early exercise boundary. This critically excludes using the calculated value one time step ahead as an upper bound. A crude, but in this case effective, root finding technique is suggested in (Grant, Vora, and Weeks 1996).

The globally admissible early exercise domain is $[0,K]$. Divide this domain into $Q$ possible values, $\beta_q : q = 1,\ldots,Q$. Evaluate $f(\beta_q)$ using equation (5.11) for each $\beta_q$. At some value, $q^*$, this value will change sign. This gives an interval $(\beta_{q^*-1},\beta_{q^*}]$ in which the root of the equation must lie. A final approximation for the early exercise boundary value $\widehat{b}_i$ is linearly interpolated from $(\beta_{q^*-1},\beta_{q^*}]$. This procedure is completed at each $t_i$ to estimate the entire discrete early exercise boundary $\{\widehat{b}_i\}_{i=1,\ldots,n}$ (see Figure 5.7).

(Fu, Laprise, Madan, Su, and Wu 2001) suggest using a bisection or a secant method to calculate the zero of equation (5.11). Any bisection method in this context requires monotonicity in the function that is being estimated. The instability caused by the Monte Carlo estimation of the expectation often means that bisection methods do not converge. Consequently, the suggested technique does not improve on the previous crude root finding technique.

Once the early exercise boundary, $\{\widehat{b}_i\}_{i=1,\ldots,n}$, has been approximated, $M$ independent stock price paths are generated. Early exercise times are evaluated and the final approximate option value, $\widehat{P}(t)$, is calculated using equation (5.5).

### 5.3.2 Bias of the GVW Algorithm

The upward bias of Tilley's method (and most dynamic programming methods) is not present in the GVW algorithm because future information is not used to make early exercise decisions. The approximate early exercise boundary is calculated using samples which are deliberately independent of the samples used for the approximation of the premium.

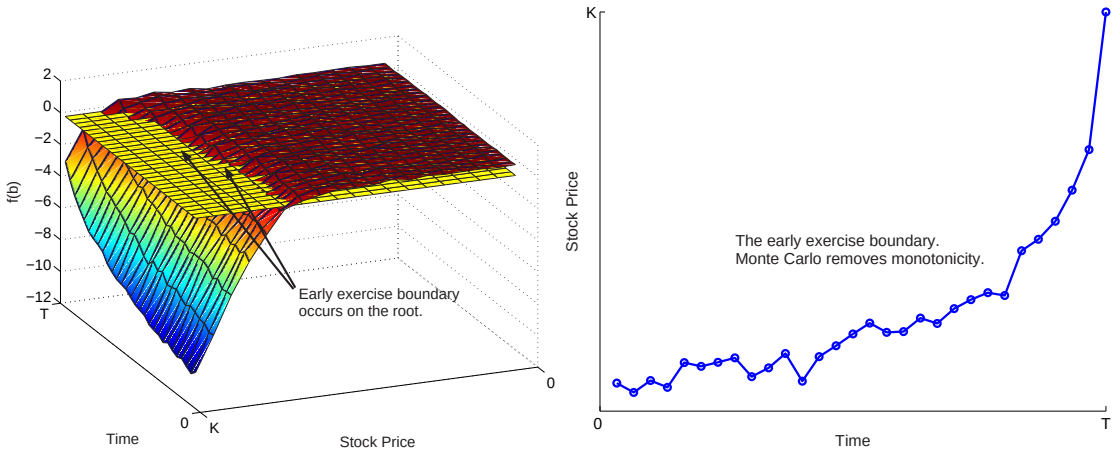


**Fig. 5.7:** The surface, $f(b)$ for stock prices in $[0, K]$, and time in $[0, T]$. The early exercise boundary occurs at the root which corresponds to equality between the exercise value and the continuation value. The figure on the right is the estimated early exercise boundary.

However a downward bias is introduced because of the approximate nature of the early exercise boundary. Almost any method which makes suboptimal exercise decisions reduces the value of the option estimate. Increasing the accuracy of the approximate boundary is difficult because of the instability of equation (5.11). It is not possible to specify the precision of the approximation. Small instabilities in the early exercise boundary value, however, do not have a large impact on the value of the option. Consequently, even though the boundary is just a rough approximation, a good estimate for the premium is obtained.

### 5.3.3 Convergence of the GVW Algorithm

The GVW algorithm produces an estimate for the American put option price which converges as the number of time steps $n$ tends towards infinity. Again, there are various factors which influence the convergence rate: the number of time steps $n$, the number of sample stock price paths $M$ and the number of stock prices $M'$ used to calculate the early exercise boundary.

#### Sample Size for Boundary Estimation $M'$

Similarly to Tilley's method, the value of the option is relatively insensitive to the early exercise boundary. Therefore, increasing the number of samples used in estimating the boundary, $\widehat{b}(t)$, does not necessarily have a large effect. Figure 5.8 shows the convergence behaviour of the GVW estimate as we increase $M'$. For each estimate of the boundary, the same set of sample stock paths is then used to estimate the price of the American put option, $\widehat{P}(t)$. These estimates seem to converge from below, although it is not a strictly increasing sequence because the early exercise boundary is unstable.

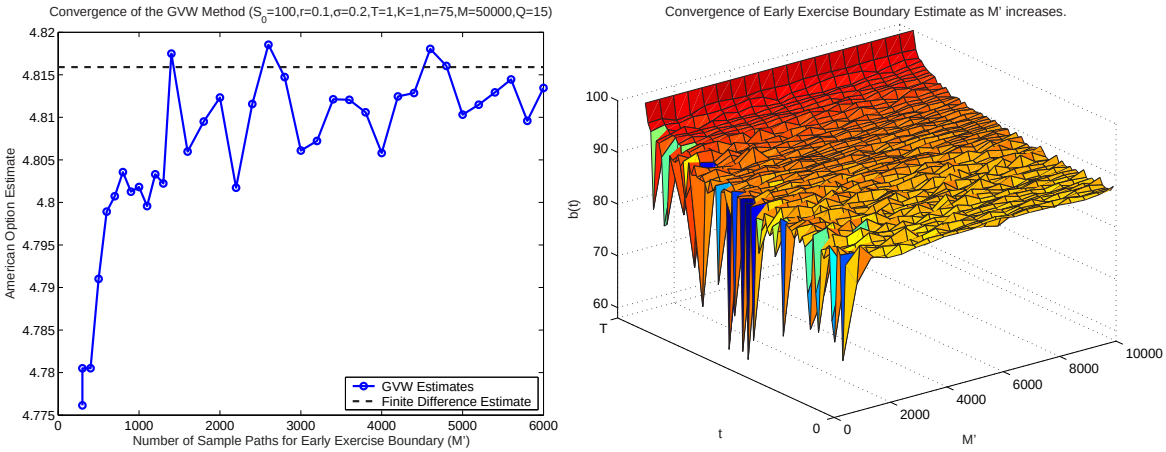


**Fig. 5.8:** The figure on the left shows the convergence behaviour of GVW estimates as the number of sample paths used to estimate the early exercise boundary, $M'$, increases. There is a fixed number of time steps ($n = 75$) for each sample stock path and a fixed number of samples, $M = 50000$. For each boundary estimate, the same sample paths are used to estimate $\widehat{P}(t)$. In all cases $Q = 15$. The Crandall-Douglas finite difference value is assumed to be the true value of the option.
The figure on the right shows the convergence behaviour of the early exercise boundary estimates as $M'$ increases.

### Sample Size Convergence

The early exercise boundary, $\widehat{b}(t)$ is estimated independently of the sample stock prices that are used to estimate the price of the American put option, $\widehat{P}(t)$. A convergence from below is noticed when the same early exercise boundary is used for estimates calculated using an increasing number of sample stock price paths for the option estimate. The downward bias implies that the GVW estimate tends towards the correct price from below as the number of sample paths increases (see Figure 5.9).

### Time step Convergence

The GVW method has the same time step convergence behaviour from below as Tilley's method (see Figure 5.10), i.e. the estimate converges to the true value from below.

## 5.4 State Space Partitioning

The state space partitioning algorithm (SSAP) for the approximation of the American put option price (Barraquand and Martineau 1995) uses a similar dynamic programming technique to Tilley's method (Tilley 1993a). Instead of organising the stock prices into bundles at each time step, however, we partition the stock

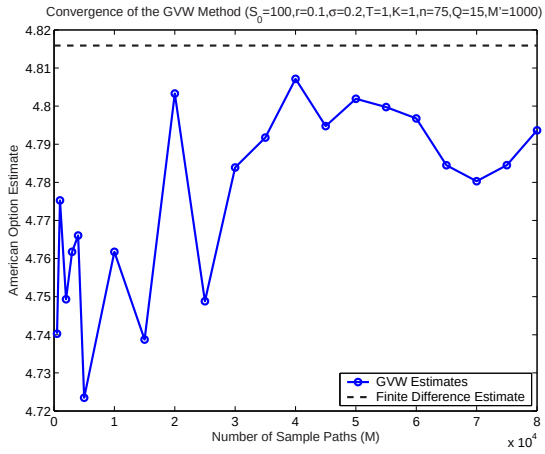


**Fig. 5.9:** Convergence behaviour of GVW estimates as the number of sample paths, $M$, increases. There is a fixed number of time steps ($n = 75$) for each sample path and a fixed number of sample paths ($M' = 1000$) for the early exercise boundary. The same estimated early exercise boundary, $\widehat{b}(t)$ is used for all estimates. The Crandall-Douglas finite difference value is assumed to be the true value of the option.

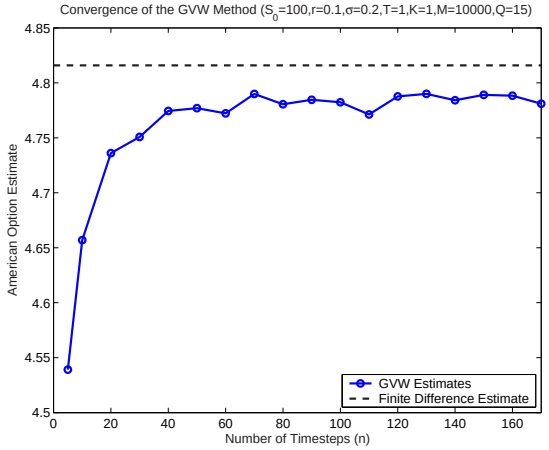


**Fig. 5.10:** The convergence behaviour of GVW estimates as the number of time steps, $n$, is increased. There is a fixed number of sample paths for each estimate with $M = 100000$ and $Q = 15$. The Crandall-Douglas finite difference value is again assumed to be the true value of the option.

price domain (state space) into a tractable number of states. These states are time dependent and are chosen from a subset of the state space. We use Monte Carlo simulation to estimate the continuation value associated with each state by calculating the conditional probabilities of moving between states. Dynamic programming techniques are then used to estimate the price of the American put option, $\widehat{P}(t)$.

In Tilley's method we use average continuation values for each stock price path.

In contrast, in the state space partitioning method we estimate continuation values for each state and not for each stock price path.

We first need to decide a partitioning of the state space. The average intrinsic value of each state must then be calculated. We then calculate the transition probabilities, and conclude by applying dynamic programming techniques to estimate $\widehat{P}(t)$.

### 5.4.1 Partitioning the State Space

Given $n$ equal time steps such that $0 = t_0 < t_1 < \cdots < t_n = T$, at each $t_i$, the state space $[0, \infty)$ is partitioned into $a_i$ disjoint states, $A_i^j : j = 1, \ldots, a_i$. The number of partitions varies with $i$. These partitions are chosen *a priori* in order to encompass the majority of the simulated stock price paths. In other words, the partitioning is over the expected densest region of paths and consigns the outliers to one of two marginal regions. The runtime of the algorithm is highly dependent on the number of partitions. Consequently, we should partition the sample space carefully to ensure that an accurate estimate is obtained from the smallest number of partitions.

(Barraquand and Martineau 1995) use a real valued function to partition a region of interest bounded by an upper and lower value, $S_{\min}$ and $S_{\max}$. At $t_0$, this region of interest is the initial stock price, $S_0$. There is only one state $a_0 = 1$, so $A_0^1 = \{S_0\}$. At all future times, $t_i$, the partitioning is chosen using,

$$A_i^j = \Big(\alpha_i \exp\{\beta_i(j-2)\}, \alpha_i \exp\{\beta_i(j-1)\}\Big], \tag{5.12}$$

where $j = 2, \ldots, a_i - 1$ and $\alpha_i$, $\beta_i$ are determined by restricting the region of interest to the set of values where simulated stock price paths are most likely to occur. This region is $[S_{\min}, S_{\max}]$ with $S_{\min}$ and $S_{\max}$ chosen such that

$$\mathbb{P}(S_t < S_{\min}) \approx \mathbb{P}(S_t > S_{\max}) \approx 0.01. \tag{5.13}$$

The probability that a sample stock price path lies outside the region of interest is small, but not impossible. Consequently, we define the first and last states to make allowance for these events: $A_i^1 = [0, S_{\min}]$ and $A_i^{a_i} = [S_{\max}, \infty)$. Using the partitioning (5.12) gives

$$\begin{aligned} \alpha_i &= S_{\min} \\ S_{\min} \exp\{\beta_i(a_i - 2)\} &= S_{\max}, \end{aligned}$$

at each time, $t_i$.

Values for $S_{\min}$ and $S_{\max}$ are chosen using the inverse cumulative normal function[3], $\Phi^{-1}(x)$ on the log price process of $S_t$ which is distributed normally with a mean, $\mu = \log(S_0) + (r - \sigma^2/2)t$, and a standard deviation, $\sigma\sqrt{t}$. This gives the following expressions for $\alpha_i$ and $\beta_i$,

$$\begin{aligned} \alpha_i &= \exp\big\{\bar{\mu}_i - \Phi^{-1}(0.01)\bar{\sigma}_i\big\} \\ \beta_i &= \big(\bar{\mu}_i - \Phi^{-1}(0.99)\bar{\sigma}_i\big)/\big(\log(\alpha_i)/(a_i - 2)\big) \end{aligned}$$

[3] $y = \Phi^{-1}(x)$ solves $\mathbb{P}(X \leq y) = x$ for a standard normal random variable $X \sim \mathcal{N}(0, 1)$.

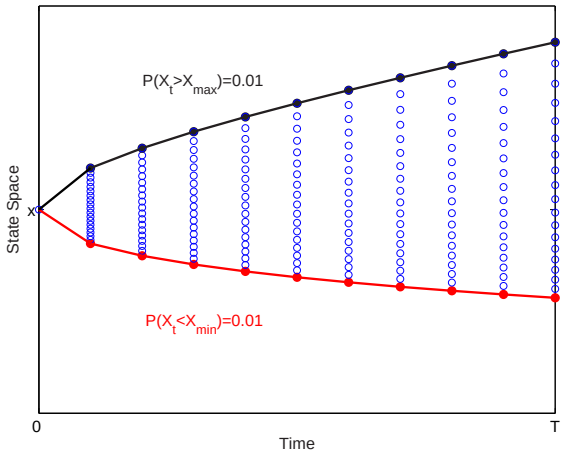


**Fig. 5.11:** An example of a partitioning of the state space with $n = 10$ time intervals and $a_i = 20$ for all $i > 0$.

where $\bar{\mu}_i = \log(S_0) + (r - \sigma^2/2)i\Delta t$ and $\bar{\sigma}_i = \sigma\sqrt{i\Delta t}$. The remaining partitions are then calculated using equation (5.12) (see Figure 5.11). At each time step, a fixed number ($a_i = a$, for all $i > 0$) of partitions is decided upon *a priori*. This number can be varied to optimise the accuracy and convergence rate of the resulting estimate.

### 5.4.2 Calculating the Average Intrinsic Value for Each State

For each state we define an average intrinsic value, $\widehat{g}(A_i^j)$, where $i = 0, \ldots, n$ and $j = 1, \ldots, a_i$. These intrinsic values are calculated from a set of Monte Carlo sample stock price paths. We first generate $M$ sample stock price paths, $\{S_0^{(m)}\}_{i=0,\ldots,n} : m = 1 \ldots, M$. We use these paths to calculate the average intrinsic value for each state, at each point in time. We do this by averaging the intrinsic value over the number of stock prices in a particular state. There is no specific number of stock prices which will automatically fall in any one state (in fact, there may be states which contain no stock prices). The average intrinsic values are calculated as the arithmetic mean of the intrinsic values of sample stock price paths which lie in the particular state,

$$\widehat{g}(A_i^j) = \frac{1}{N_i^j} \sum_{m=1}^{M} \left(K - S_i^{(m)}\right)^+ \mathbb{I}_{S_i^{(m)} \in A_i^j},$$

where $i = 1, \ldots, n$, $j = 1, \ldots, a_i$ and $N_i^j = \sum_{m=1}^{M} \mathbb{I}_{S_i^{(m)} \in A_i^j}$ is the number of sample stock price paths that belong to each state $A_i^j$ (see Figure 5.12).

### 5.4.3 Calculating Transition Probabilities

The SSAP algorithm uses the dynamic programming algorithm equations (5.3) and (5.4) applied to each of the states $A_i^j$. The continuation value associated with each state is calculated as the discounted expected value of the option one time step in

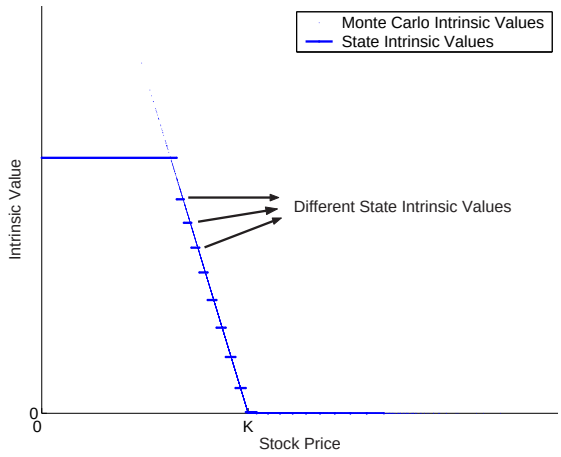


**Fig. 5.12:** The state intrinsic values are equal to the mean of the Monte Carlo stock price path intrinsic values.

the future, conditional on the originating stock price being in this state. In order to calculate this, we require the transition probabilities, $q_i^{j,k}$. These represent the probability of moving from state $A_i^j$, at $t_i$, to state $A_{i+1}^k$, at $t_{i+1}$. To calculate these probabilities we require the solution of

$$q_i^{j,k} = \mathbb{P}\big(S_{i+1} \in A_{i+1}^k \big| S_i \in A_i^j\big),$$

where $i = 1, \ldots, n-1$, $j = 1, \ldots, a_i$ and $k = 1, \ldots, a_{i+1}$. We can estimate these probabilities by using the same sample stock price paths used to estimate the intrinsic values of each state. Let this estimate be $\widehat{q}_i^{j,k}$, where,

$$\widehat{q}_i^{j,k} = N_i^{j,k}/N_i^j$$

where $N_i^{j,k}$ is the number of stock price paths that originate in state $A_i^j$ and subsequently pass through state $A_{i+1}^k$, and $N_i^j$ is as previously defined.

Once the transition probabilities have been estimated, the final task is simply to calculate the continuation values for each state, and compare them with their intrinsic values.

### 5.4.4 Estimating the Put Option Price

The sample stock price paths $\{S_0^{(m)}\}$ are only used to calculate the average intrinsic values $\widehat{g}(A_i^j)$ and to estimate the transition probabilities $\widehat{q}_i^{j,k}$. Thereafter, only state option values are calculated at each time step.

The continuation values, $\widehat{H}_i(A_i^j)$, are calculated for each state at each time step,

$$\widehat{H}_i(A_i^j) = \exp\{-r\Delta t\} \sum_{k=1}^{a_{i+1}} q_i^{j,k} \widehat{P}_{i+1}(A_{i+1}^k),$$

where $i = 1, \ldots, n-1$ and $j = 1, \ldots, a_i$. The dynamic programming problem then becomes,

$$\widehat{P}_n(A_n^j) = \widehat{g}_n(A_n^j), \quad j = 1, \ldots, a_n$$
$$\widehat{P}_{i-1}(A_{i-1}^j) = \max\{\widehat{g}_{i-1}(A_{i-1}^j), \widehat{H}_{i-1}(A_{i-1}^j)\},$$

where $i = n, \ldots, 1$ and $j = 1, \ldots, a_i$. The final estimated option value, $\widehat{P}(t)$, is set equal to the initial state value $\widehat{P}(A_0^1)$.

### 5.4.5 Bias of the SSAP Algorithm

In a similar fashion to the convergence behaviour of Tilley's method, the regression-based algorithms display a combination of an upward and a downward bias. The upward bias results from the use, at each point in time, of the complete future information set. Early exercise decisions based on an estimated early exercise boundary cause the downward bias. Although neither bias can easily be measured, the fact that they counteract each other ensures a more accurate estimate. Part of the upward bias can be eliminated by using one set of sample stock price paths to estimate the functional form of the continuation values, and another independent set of sample stock price paths to estimate the premium (see Figure 5.17).

### 5.4.6 Convergence of the SSAP Algorithm

The SSAP algorithm produce an estimate for the American put option price which converges as the number of time steps $n$ tends towards infinity. Again, there are various factors which influence the convergence rate: the number of time steps, $n$, the number of sample stock price paths, $M$ and the number of states, $a$, used to calculate the estimate for the American put option.

#### Sample Size Convergence

Figure 5.13 shows that the SSAP estimates seem to converge to the correct solution from above. Although this convergence is not uniform, the estimate should tend to the true value as the number of sample stock price paths increases.

#### Time step Convergence

The time step convergence is difficult to examine. There is a relationship between the number of states and the number of time steps. As a consequence, keeping all the other parameters fixed results in a convergence which is difficult to interpret. Figure 5.14 shows that with few time steps the estimate is undervalued.

#### Number of States $a$

The number of states, $a$, has a significant impact on both the quality of the estimated option value and the computational time of the algorithm. As we increase the number of states, the option estimate converges from above the true value (see Figure 5.15). Our results suggest that the number of states needed is proportional

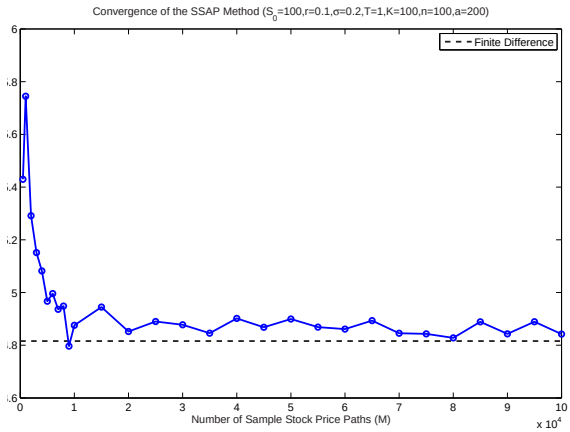


**Fig. 5.13:** Convergence behaviour of SSAP estimates as the number of sample paths, $M$, increases. There is a fixed number of time steps ($n = 100$) for each sample path and a fixed number of states ($a = 200$). The Crandall-Douglas finite difference value is assumed to be the true value of the option.

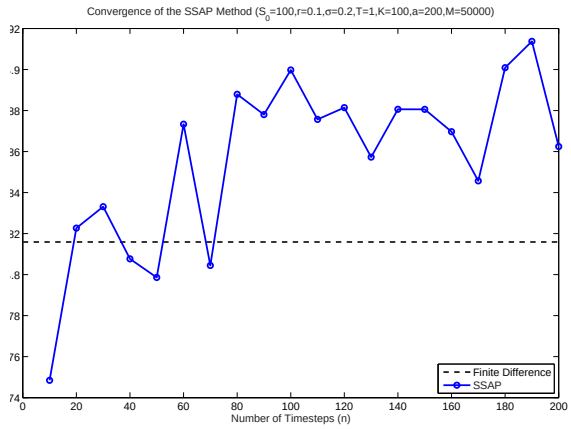


**Fig. 5.14:** Convergence behaviour of SSAP estimates as the number of time steps, $n$, is increased. There is a fixed number of sample paths ($M = 50000$) for each estimate and a fixed number of states ($a = 200$). The Crandall-Douglas finite difference value is again assumed to be the true value of the option.

to the number of time steps, $n$, used for the approximation. Experiments show that a value of $a = 2n$ is an optimal balance between precision and computational time.

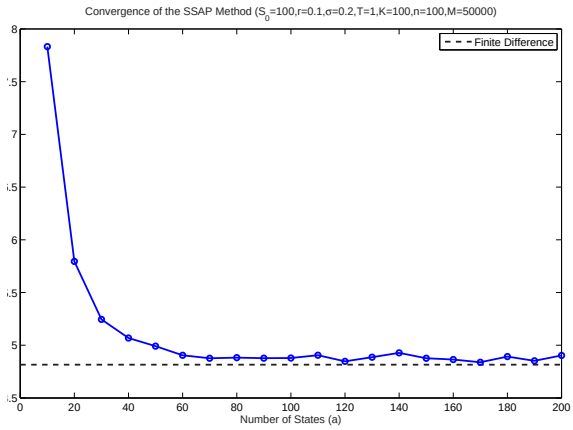


**Fig. 5.15:** The convergence behaviour of SSAP estimates as the number of states, $a$, increases. There is a fixed number of time steps ($n = 100$) for each sample stock path and a fixed number of samples ($M = 50000$). The Crandall-Douglas finite difference value is assumed to be the true value of the option.

## 5.5 Regression-Based Methods

It was first noted in (Carriere 1996) that the bundling algorithm of (Tilley 1993a) is simply a crude regression technique to estimate the continuation values, $\widehat{H}_i(S) : i = 1, \ldots, n-1$ at each time step, $t_i$. Consequently, (Carriere 1996) introduced the first Monte Carlo method to estimate the price of the American put option using nonparametric regression. One drawback of the nonparametric regression method is that it is extremely time consuming. Subsequently, (Longstaff and Schwartz 2001, Tsitsiklis and Van Roy 2001) introduced a parametric regression method which improves the speed of the algorithm. The increase in speed is achieved by fitting a predetermined functional form to the continuation value at each time step. This is done by regressing the function against the discounted option values from one time step ahead. This is in contrast to smoothing the data by local regression.

The Monte Carlo estimate $\widehat{P}(t)$ uses the dynamic programming algorithm equations (5.3) and (5.4). The continuation values, $\widehat{H}_i(S) : i = 1, \ldots, n$, are estimated by applying some regression technique at each time step to the $M$ sample paths $\left\{S_i^{(m)}\right\}_{i=0,\ldots,n} : m = 1, \ldots, M$. The terminal values, $P(S_n^{(m)})$, are set equal to the payoff function in the usual fashion using equation (5.3). At each preceding time step we estimate the continuation values, $\widehat{H}_i(S_i^{(m)}) : m = 1, \ldots, M$, by performing a regression of discounted option values against the stock prices. Thus at time $t_i$, the independent (explanatory) variables are the stock prices $S_i^{(m)} : m = 1, \ldots, M$, and the dependent (explained) variables are the discounted $t_{i+1}$ option values, $d_i \widehat{P}_{i+1}(S_{i+1}^{(m)}) : m = 1, \ldots, M$.

The continuation value for each sample stock price path is then compared to the intrinsic value in order to determine an option estimate, $\widehat{P}_i(S_i^{(m)}) : m = 1, \ldots, M$.

In (Longstaff and Schwartz 2001, Carriere 1996) the continuation values are used *to make exercise decisions only.* As a consequence, the dynamic programming equation (5.6) becomes,

$$\widehat{P}_{i-1}(S) = \begin{cases} g(t,S) & \text{if } \widehat{H}_{i-1}(S) \leq g(t,S) \\ d_{i-1}\widehat{P}_i(S) & \text{if } \widehat{H}_{i-1}(S) > g(t,S) \end{cases},$$

where $i = n, \ldots, 2$. Alternatively, (Tsitsiklis and Van Roy 2001) use the estimated continuation value as an approximation for the option price if early exercise is not optimal. As a consequence, the dynamic programming equation (5.6) becomes,

$$\widehat{P}_{i-1}(S) = \max\{g(t,S), \widehat{H}_{i-1}(S)\},$$

where $i = n, \ldots, 2$. The final approximate option value, $\widehat{P}(t)$, is then calculated using (5.9).

### 5.5.1 Nonparametric Regression

The continuation values are estimated in (Carriere 1996) using nonparametric local regression techniques. Nonparametric regression allows a data set to be smoothed without specifying any functional form for the smoothed relationship.

Local regression techniques are applied to data sets to smooth the set, one data point at a time. The locally weighted regression (*loess*) technique (Cleveland and Devlin 1988) smoothes the dependent data using the local data around each independent point. The procedure is similar in principle to calculating a moving average for a time series[4]. Figure 5.16 was created using the *loess* function.

In general, the benefit of local fitting is that a large class of data can be smoothed. Parametric regression is constrained to a certain class of data. The drawback of nonparametric regression is that the technique is computationally intensive.

### 5.5.2 Parametric Regression

The kind of data that results from option pricing lends itself to parametric regression. We decide *a priori* on a functional form and perform a least-squares regression on the data to calculate the coefficients of the function. The benefit of parametric regression is that it improves the computational time. Least squares regression only requires a small number of matrix calculations at each time step. This compares favourably to the large number of calculations required for the nonparametric regression to smooth each data point.

The function can be a linear combination of more complex functions (*linear-in-the-parameters regression*) which can improve the fit if these functions are chosen appropriately. We can approximate the continuation values, $\widehat{H}_{i-1}(S) : i-1 = n, \ldots, 1$, by fitting a function consisting of linear combination of $\mathscr{F}_{t_i}$-measurable functions (i.e. functions depending on $S_i$) to the set of discounted option values, $d_{i-1}\widehat{P}_i(S_i^m) : m = 1, \ldots, M$.

[4] The *loess* method is implemented in Matlab using the `smooth` function with the method variable set to '`loess`'.

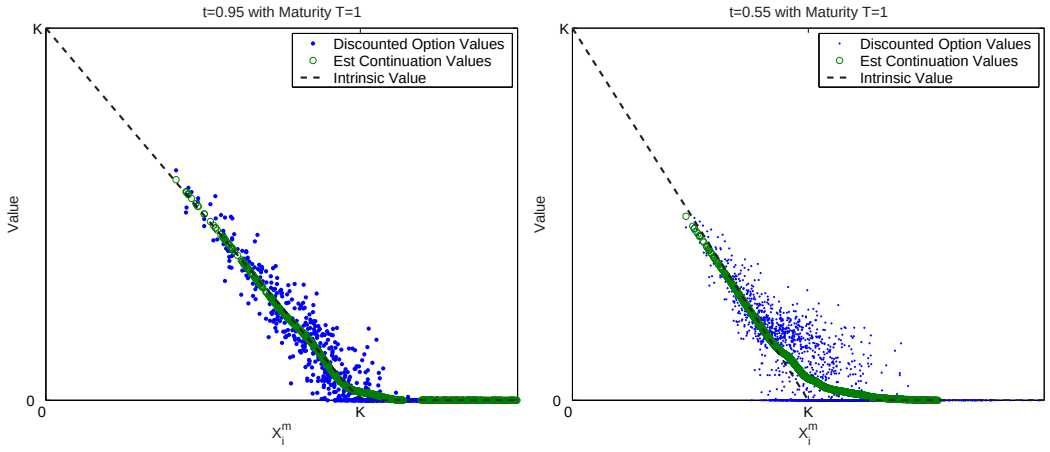


**Fig. 5.16:** At $t_i$, the continuation values are estimated by regressing the stock prices, $\boldsymbol{S_i} = \left(S_i^1, \ldots, S_i^M\right)^T$, against the discounted option values, $d_i \widehat{\boldsymbol{P}}_{\boldsymbol{i+1}} = d_i\left(\widehat{P}_{i+1}^1, \ldots, \widehat{P}_{i+1}^M\right)^T$.

A functional form for the continuation values is created by choosing a set of basis functions, $F^j(S)$, with associated coefficients, $\beta^j$,

$$H_i(S) = \sum_{j=0}^{\infty} \beta^j F^j(S). \tag{5.14}$$

An approximation for the continuation value is obtained by truncating this series. The coefficients $\beta^j : j = 0, \ldots, n'$ are unknown and must be estimated using least squares regression.

A technique to increase the rate of convergence of this method is suggested in (Longstaff and Schwartz 2001). Here they consider only those paths which are in-the-money. The estimated continuation values are used to make exercise decisions at each time step. Hence, if the option is out-of-the-money, there is no possibility of exercise and consequently the estimated continuation value is redundant. Considering only in-the-money paths reduces the number of basis functions needed to estimate the conditional expectation.

Another benefit of using parametric regression is that two sets of sample stock price paths can be used. One sample is used to estimate the functional form for the continuation value at each time step, while another set of independent sample paths is used to estimate the price of the American put option. This removes some of the upward bias inherent in dynamic programming methods. It also allows us to test the quality of the estimator. A good estimator will give accurate values when we use an independent set of sample stock price paths.

Various functional forms which are linear-in-the-parameters are suggested in (Longstaff and Schwartz 2001), but are not investigated in (Tsitsiklis and Van Roy 2001). Here, the functional form will be investigated simultaneously for both approaches. The rest of the section details the linear-in-the-parameters regression and then examines the convergence and bias of the methods.

### The Functional Form of the Continuation Value

The definition of the continuation value in terms of a linear combination of basis functions is given by equation (5.14). Clearly, we need to truncate this definition at some value, $n'$. Consequently, we need to decide on the type of basis functions to be used, and the truncation value, $n'$. In (Longstaff and Schwartz 2001), the continuation value is expressed as,

$$\widehat{H}_i(S) = \sum_{j=0}^{n'} \widehat{\beta}_i^j F^j(S), \tag{5.15}$$

where $i = n-1, \ldots, 1$ and $F^j(S) : j = 0, \ldots, n'$ is a set of orthogonal polynomials. (Longstaff and Schwartz 2001) suggested the use of Laguerre, Hermite, Legendre, Chebyshev, Gegenbauer, or Jacobi polynomials. It is possible to calculate numerical approximations of these polynomials for relatively high values of $n'$. (For a full definition of the numerical approximations for these polynomials, see (Abramowitz and Stegun 1970).) Using a small number of basis functions improves computational speed.

(Longstaff and Schwartz 2001) used weighted Laguerre polynomials with a dampening factor $\exp\{-x/2\}$. They give no justification or explanation for this additional factor. Empirically this dampening factor has a negative effect on the functional fit. In all future calculations, we omit this factor altogether.

### Linear-in-the-Parameters Regression

Consider a single sample stock price path, $S_i^{(m)} : m \in [1, \ldots, M]$. The function (5.15) relates the stock price at time $t_i$ with the corresponding option value at $t_{i+1}$, for each sample stock price path. Algorithmically, at time $t_i$, we regress stock prices, $\boldsymbol{S_i} = \left(S_i^1, \ldots, S_i^M\right)^T$, against their corresponding discounted option values, $d_i\widehat{\boldsymbol{P}}_{\boldsymbol{i+1}} = d_i\left(\widehat{P}_{i+1}^1, \ldots, \widehat{P}_{i+1}^M\right)^T$. This regression is performed at each time step $t_i : i = n-1, \ldots, 1$ to obtain the coefficients, $\widehat{\beta}_i^j : j = 0, \ldots, n'$, of the continuation value in equation (5.15).

As a modification, the regression should be performed only for in-the-money values, $\boldsymbol{S_i'} = \left(S_i^1, \ldots, S_i^{M'}\right)^T$, where $S_i^m \in \boldsymbol{S_i'}$ if $S_i^m < K$. $\boldsymbol{S_i'}$ is an $(M' \times 1)$-column vector containing only those stock prices at time $t_i$ which correspond to the option being in-the-money at $t_i$. The least squares regression used, returns the set $\{\widehat{\beta}_i^j\} : j = 1, \ldots, n'$ which minimises

$$\min_{\{\widehat{\beta}_i^j\}} \sum_{m=1}^{M} \left(d_i\widehat{P}_{i+1}(S_{i+1}^m) - \sum_{j=0}^{n'} \widehat{\beta}_i^j F^j(S_i^m)\right)^2. \tag{5.16}$$

It should be clear that the regression procedure is identical (merely replacing $M$ with $M'$ in equation (5.16)) if only the in-the-money paths are used. The continuation values, $\widehat{H}_i(S)$, are then calculated using the estimated coefficients, $\{\widehat{\beta}_i^j\}$, in equation (5.15).

(Longstaff and Schwartz 2001) (LSM) use the estimated continuation values for early exercise decisions, but do not use the estimated values from the regression for the option value at $t_i$. The discounted $t_{i+1}$ value is used instead. (Tsitsiklis and Van Roy 2001) (TVR) do not make this distinction and use the regression estimates for exercise decisions and as the option value at $t_i$. However, this results in a large upward bias (see Figure 5.18 and Figure 5.19). Consequently, we only examine the convergence and bias results of the LSM algorithm.

### 5.5.3 Bias of the Regression-Based Algorithms

In a similar fashion to the convergence behaviour of Tilley's method, the regression-based algorithms display a combination of upward and downward bias and as before they counteract each other. The upward bias is produced by the use at each point in time of the complete future information set. The downward bias is produced by making early exercise decisions based on an estimated early exercise boundary. We can reduce the upward bias by using one set of sample stock price paths to estimate the functional form of the continuation values, and an independent set of sample stock price paths to estimate the premium (see Figure 5.17).

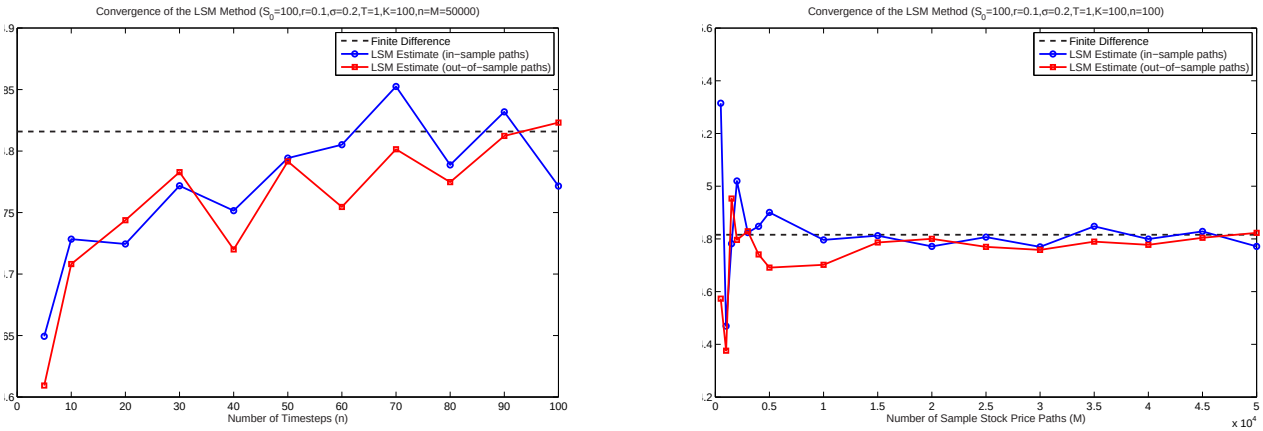


**Fig. 5.17:** The convergence of the LSM algorithm comparing estimates calculated using in and out-of-sample stock price paths. The figure on the left shows the convergence as the number of time steps, $n$, is increased. The figure on the right shows the convergence as the number of sample stock price paths, $M$, is increased.

### 5.5.4 Convergence of the Regression-Based Algorithms

The regression-based algorithms produce an estimate for the American put option price which converges as the number of time steps $n$ tends towards infinity. Again, there are various factors which influence the convergence rate: the number of time steps, $n$, the number of sample stock price paths, $M$ and the choice of basis functions.

### Sample Size Convergence

The regression-based algorithms have a similar convergence behaviour with increasing sample size to that of Tilley's method. Figure 5.18 shows no discernable convergence bias, which is a result of the offsetting bias mentioned earlier. The estimate does not converge from above or below the true value. The estimate should tend to the true value, however, as the number of sample stock price paths increases.

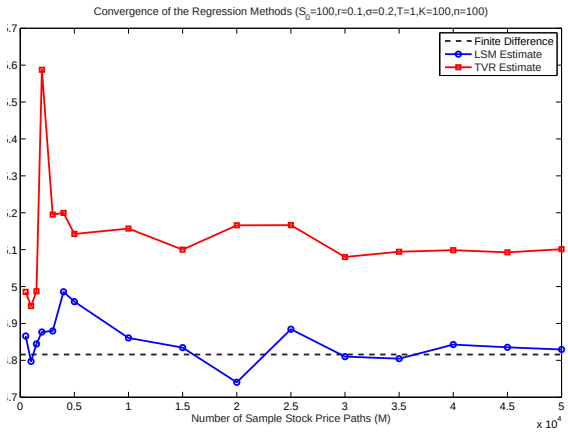


**Fig. 5.18:** The convergence behaviour for LSM and TVR estimates as the number of sample stock price paths, $M$, is increased. There is a fixed number of time steps ($n = 100$) for each estimate. Only in-the-money sample stock price paths are used in the regression for the LSM estimates (which uses the first six Hermite polynomials). The Crandall-Douglas finite difference value is assumed to be the true value of the option.

### Time step Convergence

The time step convergence is also similar to that of Tilley's method. It initially converges from below as the number of time steps increases. This downward bias is gradually counteracted, however, by the upward bias resulting from the use of the complete information set. This, coupled with the random behaviour of Monte Carlo estimation, results in a convergence behaviour which is not uniform (see Figure 5.19).

### Choice of Basis Function

When considering the convergence of the parametric regression methods, we examine the type of orthogonal polynomial used for the basis functions, as well as the truncation value, $n'$. The choice of polynomial does not seem to have a significant impact on the estimate (see Figure 5.20). The truncation value does, however, have a significant impact on the quality of the estimated option value. Using a small number of basis functions does not allow for an accurate approximation of the continuation value. An improvement occurs by increasing $n'$ up to a certain threshold.

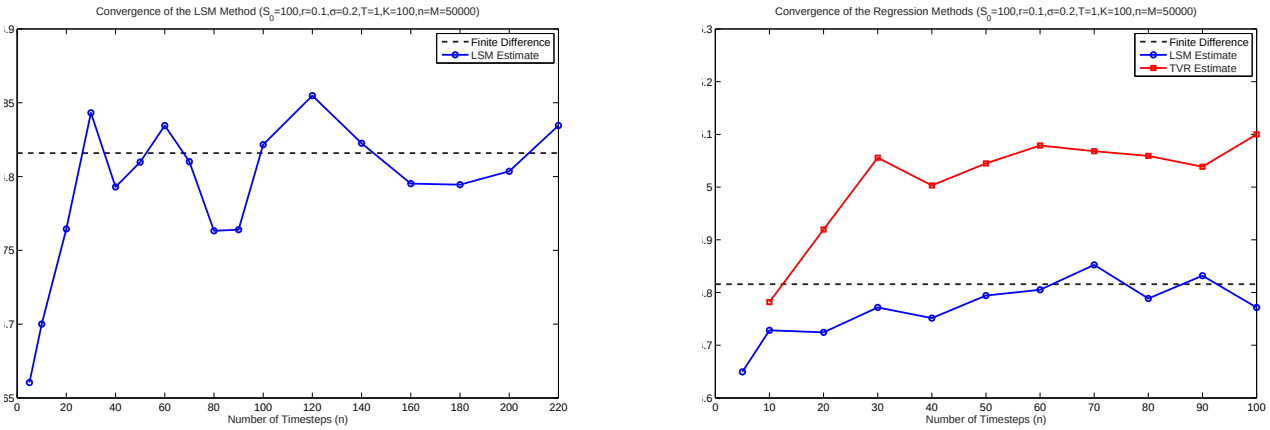


**Fig. 5.19:** The convergence behaviour for LSM and TVR estimates as the number of time steps, $n$, is increased. There is a fixed number of sample paths ($M = 50000$) for each estimate. Only in-the-money sample stock price paths are used in the regression for the LSM estimates (which uses the first six Hermite polynomials). The Crandall-Douglas finite difference value is assumed to be the true value of the option.

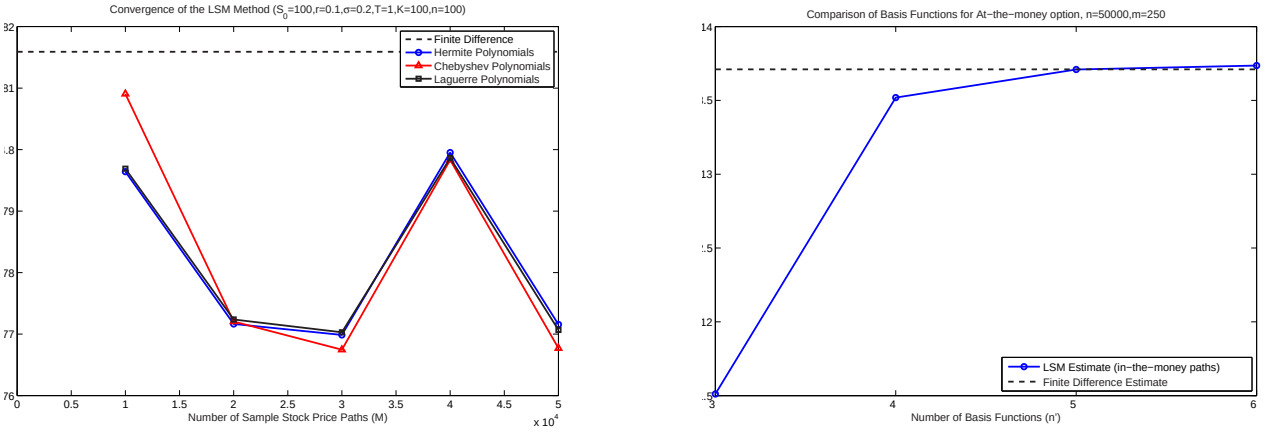


**Fig. 5.20:** The figure on the left shows the convergence behaviour as the number of time steps is increased for different basis functions. The figure on the right shows the effect of increasing the number of basis functions.

Thereafter, increasing the number of basis functions increases the computation time, but with a diminishing improvement in the option estimate. Hence, we should consider an optimal balance of precision and speed when choosing the number of basis functions. Our results suggest that using five basis functions is usually sufficient (see Figure 5.20).

### Sample Paths Used for the Regression

Using the entire set of sample stock price paths at each $t_i$ to perform the regression has an adverse effect on the estimated option value. This results because the functional form of only in-the-money samples is easier to fit than the functional form of the entire set of sample stock price paths.

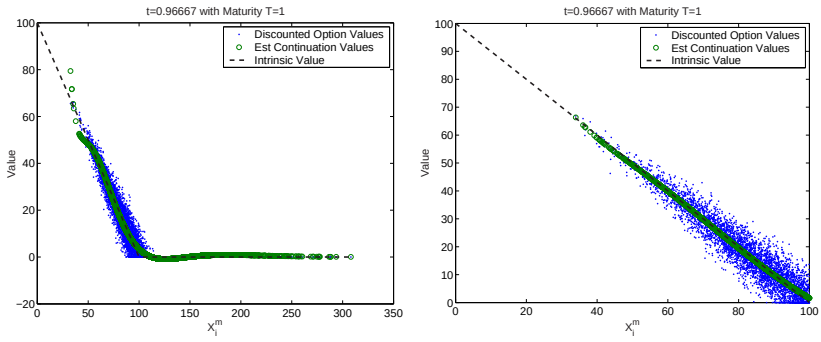


**Fig. 5.21:** The estimated continuation values one time step from expiry. The figure on the left (right) uses all the stock price paths (only in-the-money paths). Regression is done using the first six Laguerre polynomials.

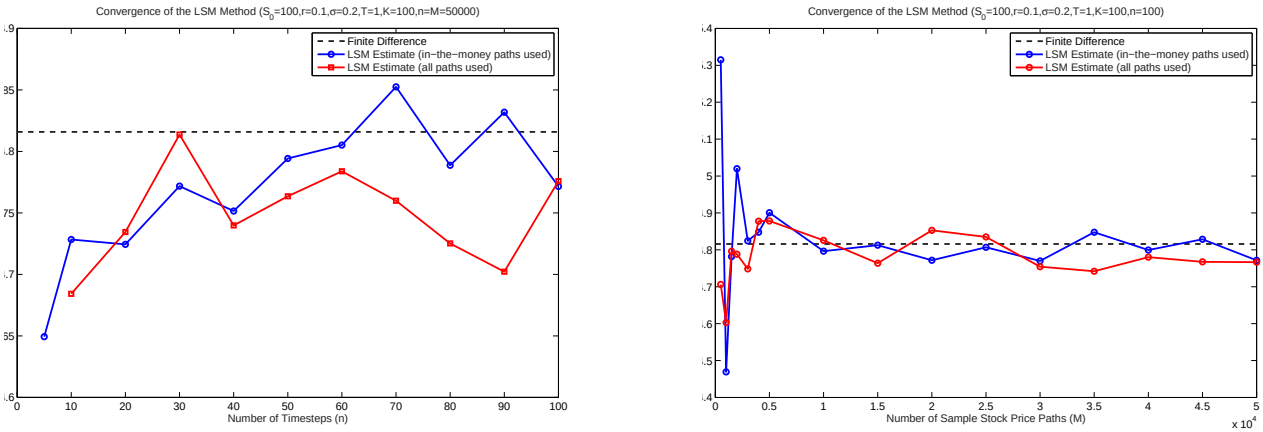


**Fig. 5.22:** The convergence of the LSM algorithm comparing the estimates calculated using regression on all the sample stock price paths and on only in-the-money paths. The figure on the left shows the convergence as the number of time steps, $n$ is increased. The figure on the right shows the convergence as the number of sample stock price paths, $M$, is increased.

# Chapter 6

# Finite Difference Methods

## 6.1 Introduction

Finite difference methods are a set of numerical techniques which may be used to approximate the solution of a partial differential equation (PDE). In applications of the technique, the PDE is transformed into a series of algebraic finite difference equations which are then solved iteratively. The American put option price, $P(t, S)$ satisfies the Black-Scholes PDE,

$$\frac{\partial P}{\partial t} + rS\frac{\partial P}{\partial S} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 P}{\partial S^2} - rP = 0 \tag{6.1}$$

over the region $t \in [0, T]$ and $S \in [0, \infty)$. This continuous region, $[0, \infty) \times [0, T]$, is approximated within the scheme by a two dimensional discrete grid. The boundary condition (2.17) for the American put is an asymptotic condition. The grid must be truncated at some finite value, $S_{\max}$, where $S_{\max}$ is sufficiently large for the value of the option to be near zero. Consequently, we seek a solution over the region $t \in [0, T]$ and $S \in [0, S_{\max}]$ :

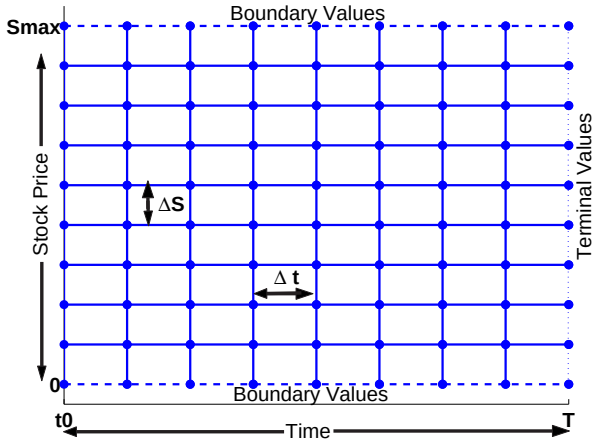


**Fig. 6.1:** A finite difference grid.

Once the grid has been defined, the PDE is approximated as a set of finite difference equations taking values at each node on the grid. Initially, the only values

on the grid that are known are the boundary values,

$$P(t, S_{\max}) \text{ and } P(t, 0) \text{ for all } t \in [0, T]$$
$$\text{and } P(T, S) \text{ for all } S \in [0, S_{\max}].$$

The finite difference equations propagate the terminal boundary condition through the grid by approximating the dynamic effect of the PDE. This is achieved in a backwardly recursive fashion. At each time step prior to $T$, an option value is calculated at each grid point by using the grid points previously calculated.

Finite difference techniques are extremely straightforward for calculating an approximation of a European put option price. However, an application of these techniques to American option pricing is less straightforward (see (Wilmott 2000) for an explanation of these difficulties). The major problem is caused by the free boundary (early exercise boundary) which divides the solution space into a region where the price obeys the Black-Scholes PDE (the continuation region, $\mathcal{C}$) and the stopping region, $\mathcal{S}$. In keeping with other numerical techniques the primary problem becomes one of estimating the early exercise boundary in order to determine where the continuation region is.

## 6.2 Finite Difference Methods for the American Put

Discretise $[0, T]$ into $n$ equal time steps, $0 = t_0 < t_1 < \cdots < t_n = T$, where $t_i = i\Delta t$ for all $i = 0, \ldots, n$ and $\Delta t = T/n$. The truncated domain $[0, S_{\max})$ is discretised into $m$ equal stock price steps, $0 = S_0 < \cdots < S_m = S_{\max}$, where $S_j = j\Delta S$ for all $j = 0, \ldots, m$ and $\Delta S = S_{\max}/m$.

Next, choose a finite difference scheme to approximate the Black Scholes PDE (6.1). It can be shown that this PDE may be reduced, with a suitable change of variables, to the heat equation,

$$\frac{\partial u}{\partial \tau} = \frac{\partial^2 u}{\partial \xi^2}.$$

The heat equation contains one first order temporal derivative and one second order spatial derivative. Solving the heat equation using finite difference schemes is simpler than solving an untransformed Black-Scholes PDE. The remainder of this section presents the transformation to the heat equation; the explicit, implicit, Crank-Nicolson and Crandall-Douglas finite difference schemes are proved; followed by an examination of the technique.

### 6.2.1 Transforming the Black-Scholes PDE into the Heat equation

Transforming the Black-Scholes PDE into the heat equation reduces the number of finite difference approximations needed to calculate the option value. The heat equation only has two derivative terms, while the Black-Scholes PDE has four derivative terms which need to be approximated. To transform the Black-Scholes PDE into the heat equation requires a new spatial variable, $\xi$, a new temporal variable, $\tau$ and a scaled option price, $\widetilde{P}(\tau, \xi)$.

Let, $\xi = \ln(S/K)$ and $\tau = \frac{1}{2}\sigma^2(T-t)$, with corresponding transformed limits, $\xi \in (-\infty, \ln(S_{\max}/K))$ and $\tau \in [0, T\sigma^2/2]$. Note that the new temporal variable has reversed the direction of time ($\tau = 0$ corresponds to $t = T$). The following scaled transformation of the solution,

$$P(t,S) = K\exp\big\{-\xi(\kappa-1)/2 - \tau(\kappa+1)^2/4\big\}\widetilde{P}(\tau,\xi), \tag{6.2}$$

where $\kappa = 2r/\sigma^2$, reduces the Black-Scholes PDE to the heat equation

$$\frac{\partial \widetilde{P}}{\partial \tau} = \frac{\partial^2 \widetilde{P}}{\partial \xi^2}. \tag{6.3}$$

What remains is to transform both the intrinsic values and the boundary conditions. The transformed intrinsic value for the American put option, $\widetilde{g}(\tau,\xi)$, is,

$$\widetilde{g}(\tau,\xi) = \Big(\exp\big\{\xi\big(\kappa+1\big)/2\big\} - \exp\big\{\xi\big(\kappa-1\big)/2\big\}\Big)^+. \tag{6.4}$$

The Dirichlet conditions become:

$$\lim_{\xi\to\infty} \widetilde{P}(\tau,\xi) = 0 \tag{6.5}$$

$$\lim_{\xi\to-\infty} \widetilde{P}(\tau,\xi) = \widetilde{g}(\tau,\xi) \tag{6.6}$$

$$\widetilde{P}(0,\xi) = \widetilde{g}(0,\xi). \tag{6.7}$$

From equation (6.2), the final option price $P(t,S)$ is,

$$P(t,S) = K^{(\kappa+1)/2}S^{(1-\kappa)/2}\exp\big\{-\sigma^2\left(\kappa+1\right)^2\left(T-t\right)/8\big\}\widetilde{P}(\tau,\xi). \tag{6.8}$$

### 6.2.2 Building a Mesh for the Heat Equation

Discretise $[0,T]$ into $n$ equal temporal steps, $0 = \tau_0 < \tau_1 < \cdots < \tau_n = T$, where $\tau_i = i\Delta\tau$ for all $i = 0,\ldots,n$ and $\Delta\tau = \sigma^2 T/2n$. The truncated spatial domain $[\xi_{min}, \xi_{\max}]$, where $\xi_{\max} = \ln(S_{\max}/K)$, is discretised into $m$ equal spatial steps, $\xi_{\min} = \xi_0 < \cdots < \xi_m = \xi_{\max}$, where $\xi_j = \xi_{\min} + j\Delta\xi$ for all $j = 0,\ldots,m$ and $\Delta\xi = (\xi_{\max} - \xi_{\min})/m$. This discretisation results in grid values, $(\tau_i, \xi_j) : i = 0,\ldots,n, j = 0,\ldots,m$ (see Figure 6.2).

The grid is constructed to ensure that the initial transformed stock price, $\ln(S_0/K)$, falls on a grid point, say $(\tau_n, \xi_k)$. Once we have calculated the option values at all the grid points, we can read off the value at $(\tau_n, \xi_k)$ and calculate the approximate American put option price using equation (6.8). If it is inconvenient to ensure that the initial transformed stock price falls on a grid point, we interpolate between the transformed option values at neighbouring grid points and then calculate the approximate American put option price using equation (6.8).

### 6.2.3 Estimating Partial Derivatives from the Grid

Estimations of the partial derivatives are calculated using finite difference approximations. We estimate first derivatives using either backward or forward differences,

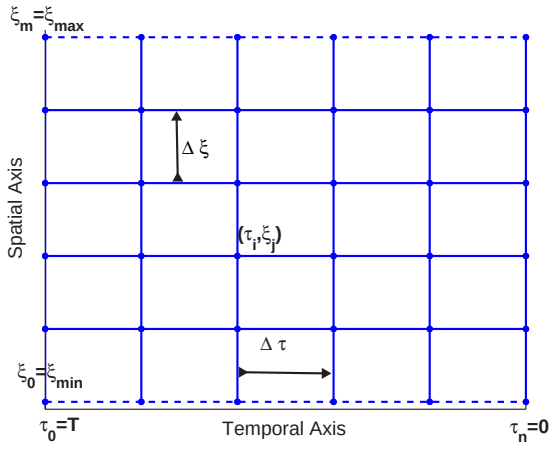


**Fig. 6.2:** The finite difference grid for the transformed heat equation with spatial variable $\xi$ and temporal variable $\tau$.

and second derivatives using central differences (see Figure 6.3),

$$\text{Forward differences} \quad \frac{\partial \widetilde{P}(\tau_i, \xi_j)}{\partial \tau} \approx \frac{\widetilde{P}(\tau_{i+1}, \xi_j) - \widetilde{P}(\tau_i, \xi_j)}{\Delta \tau}$$

$$\text{Backward differences} \quad \frac{\partial \widetilde{P}(\tau_i, \xi_j)}{\partial \tau} \approx \frac{\widetilde{P}(\tau_i, \xi_j) - \widetilde{P}(\tau_{i-1}, \xi_j)}{\Delta \tau}$$

$$\text{Central differences} \quad \frac{\partial^2 \widetilde{P}(\tau_i, \xi_j)}{\partial \xi^2} \approx \frac{\widetilde{P}(\tau_i, \xi_{j+1}) - 2\widetilde{P}(\tau_i, \xi_j) + \widetilde{P}(\tau_i, \xi_{j-1})}{(\Delta \xi)^2}$$

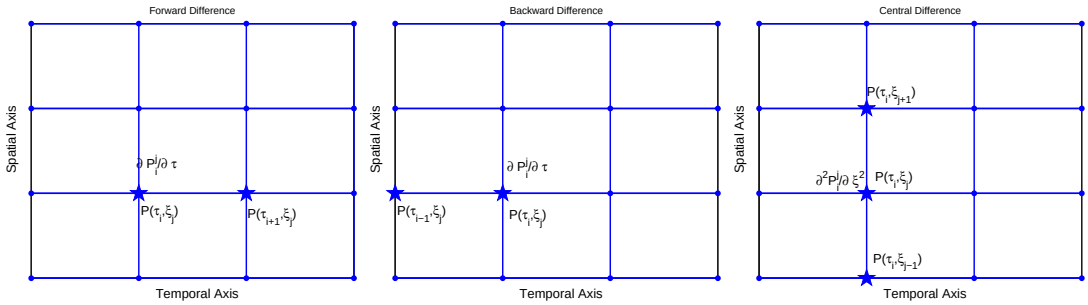


**Fig. 6.3:** The forward difference (left) and backward difference (middle) approximation for $\partial \widetilde{P}/\partial \tau$, and the central difference (right) approximation $\partial^2 \widetilde{P}/\partial \xi^2$.

### 6.2.4 The Explicit Method

The explicit method is implemented by replacing $\partial \widetilde{P}/\partial \tau$ with the forward difference approximation, and $\partial^2 \widetilde{P}/\partial \xi^2$ with the central difference approximation. Define $\widetilde{P}_i^j \equiv$

$\widetilde{P}(\tau_i, \xi_j)$. Then,

$$\frac{\widetilde{P}_{i+1}^{j} - \widetilde{P}_{i}^{j}}{\Delta\tau} + O(\Delta\tau) = \frac{\widetilde{P}_{i}^{j+1} - 2\widetilde{P}_{i}^{j} + \widetilde{P}_{i}^{j-1}}{(\Delta\xi)^2} + O\big((\Delta\xi)^2\big). \tag{6.9}$$

Rearranging equation (6.9) gives the explicit finite difference scheme at each time step, $\tau_i : i = 1, \ldots, n$,

$$\widetilde{P}_{i}^{j} = \alpha\widetilde{P}_{i-1}^{j+1} + (1-2\alpha)\widetilde{P}_{i-1}^{j} + \alpha\widetilde{P}_{i-1}^{j-1}, \tag{6.10}$$

where $j = 1, \ldots, m-1$ and

$$\alpha = \Delta\tau/(\Delta\xi)^2. \tag{6.11}$$

The recurrence scheme equation (6.10) may be represented as a matrix calculation

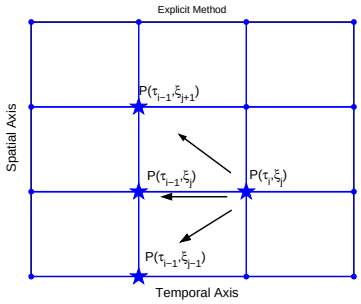


**Fig. 6.4:** The explicit method.

in order to speed up computation. At each time step the matrix system for the explicit method is given by,

$$\begin{pmatrix} \widetilde{P}_i^1 \\ \widetilde{P}_i^2 \\ \vdots \\ \widetilde{P}_i^{m-2} \\ \widetilde{P}_i^{m-1} \end{pmatrix} = \begin{pmatrix} 1-2\alpha & \alpha & 0 & \ldots & 0 \\ \alpha & \ddots & \alpha & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \alpha & 1-2\alpha & \alpha \\ 0 & \ldots & 0 & \alpha & 1-2\alpha \end{pmatrix} \begin{pmatrix} \widetilde{P}_{i-1}^1 \\ \widetilde{P}_{i-1}^2 \\ \vdots \\ \widetilde{P}_{i-1}^{m-2} \\ \widetilde{P}_{i-1}^{m-1} \end{pmatrix} + \alpha \begin{pmatrix} \widetilde{P}_{i-1}^0 \\ 0 \\ \vdots \\ 0 \\ \widetilde{P}_{i-1}^m \end{pmatrix}$$

which can be written as,

$$\widetilde{\boldsymbol{P}}_i = \mathbf{N}_{\mathrm{E}}\widetilde{\boldsymbol{P}}_{i-1} + \alpha\boldsymbol{\beta}_{i-1}, \tag{6.12}$$

where $i = 1, \ldots, n$ and $\widetilde{\boldsymbol{P}}_i = (\widetilde{P}_i^1, \ldots, \widetilde{P}_i^{m-1})$ is the vector of option prices, $\boldsymbol{\beta}_i = (\widetilde{P}_i^0, 0, \ldots, 0, \widetilde{P}_i^m)$ is the vector of boundary values and $\mathbf{N}_{\mathrm{E}}$ is an $(m-1) \times (m-1)$ tridiagonal matrix.

The advantage of this method is that the system (6.12) can be solved explicitly at each time step. As a consequence, it is computationally the fastest method, because the solution is found using simple matrix multiplication. The drawback to this method, however, is that the choice of step sizes $\Delta\tau$ and $\Delta\xi$ has a large impact on the convergence of the scheme. In fact, the method is only stable when $\alpha < 1/2$ (Morton and Mayers 1994).

### 6.2.5 The Implicit Method

The implicit method is implemented by replacing $\partial\widetilde{P}/\partial\tau$ with the backward difference approximation, and $\partial^2\widetilde{P}/\partial\xi^2$ with the central difference approximation. Again, define $\widetilde{P}_i^j \equiv \widetilde{P}(\tau_i, \xi_j))$. Then,

$$\frac{\widetilde{P}_i^j - \widetilde{P}_{i-1}^j}{\Delta\tau} + O(\Delta\tau) = \frac{\widetilde{P}_i^{j+1} - 2\widetilde{P}_i^j + \widetilde{P}_i^{j-1}}{(\Delta\xi)^2} + O\big((\Delta\xi)^2\big). \tag{6.13}$$

Rearranging equation (6.13) gives the implicit finite difference scheme at each time step, $\tau_i : i = 1, \ldots, n$,

$$\alpha\widetilde{P}_i^{j+1} + (1-2\alpha)\widetilde{P}_i^j + \alpha\widetilde{P}_i^{j-1} = \widetilde{P}_{i-1}^j, \tag{6.14}$$

where $j = 1, \ldots, m-1$ and $\alpha$ is defined in equation (6.11). The recurrence scheme

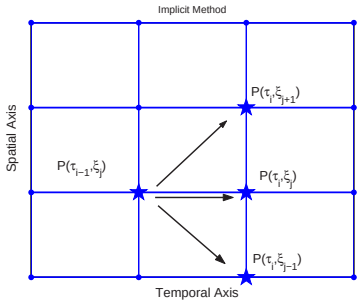


**Fig. 6.5:** The implicit method.

equation (6.14) can also be represented as a matrix calculation in order to facilitate fast computation. At each time step, the matrix system for the implicit method is given by,

$$\begin{pmatrix} 1-2\alpha & \alpha & 0 & \ldots & 0 \\ \alpha & \ddots & \alpha & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \alpha & 1-2\alpha & \alpha \\ 0 & \ldots & 0 & \alpha & 1-2\alpha \end{pmatrix} \begin{pmatrix} \widetilde{P}_i^1 \\ \widetilde{P}_i^2 \\ \vdots \\ \widetilde{P}_i^{m-2} \\ \widetilde{P}_i^{m-1} \end{pmatrix} + \alpha \begin{pmatrix} \widetilde{P}_i^0 \\ 0 \\ \vdots \\ 0 \\ \widetilde{P}_i^m \end{pmatrix} = \begin{pmatrix} \widetilde{P}_{i-1}^1 \\ \widetilde{P}_{i-1}^2 \\ \vdots \\ \widetilde{P}_{i-1}^{m-2} \\ \widetilde{P}_{i-1}^{m-1} \end{pmatrix}$$

which can be written as,

$$\mathbf{M}_{\mathrm{I}}\widetilde{\boldsymbol{P}}_i + \alpha\boldsymbol{\beta}_i = \widetilde{\boldsymbol{P}}_{i-1}, \tag{6.15}$$

where $i = 1, \ldots, n$, $\widetilde{\boldsymbol{P}}_i = (\widetilde{P}_i^1, \ldots, \widetilde{P}_i^{m-1})$ is the vector of option prices, $\boldsymbol{\beta}_i = (\widetilde{P}_i^0, 0, \ldots, 0, \widetilde{P}_i^m)$ is the vector of boundary values and $\mathbf{M}_{\mathrm{I}}$ is an $(m-1)\times(m-1)$ tridiagonal matrix.

The solution of equation (6.15) can be found either by using an LU-decomposition or an iterative scheme known as the successive over relaxation (SOR) technique. The implicit method is computationally slower than the explicit method but it is more stable. In fact, the method is stable for all positive values of $\alpha$ (Wilmott, Howison, and Dewynne 1995). As a consequence there is no restriction on step size or time step size.

### 6.2.6 The Crank-Nicolson Method

The explicit finite difference method uses three grid points at $\tau_{i-1}$ to generate one point at $\tau_i$. The implicit finite difference method inverts this and uses one grid point at $\tau_{i-1}$ to generate three points at $\tau_i$. A generalisation of these two methods would be to employ all six grid points together (Morton and Mayers 1994). The Crank-Nicolson method averages the explicit and implicit finite difference schemes (equations (6.10) and (6.14)) to produce,

$$\frac{\widetilde{P}_i^j - \widetilde{P}_{i-1}^j}{\Delta\tau} + O((\Delta\tau)^2) = \frac{1}{2}\left(\frac{\widetilde{P}_i^{j+1} - 2\widetilde{P}_i^j + \widetilde{P}_i^{j-1}}{(\Delta\xi)^2} + \frac{\widetilde{P}_{i-1}^{j+1} - 2\widetilde{P}_{i-1}^j + \widetilde{P}_{i-1}^{j-1}}{(\Delta\xi)^2}\right) + O\big((\Delta\xi)^2\big). \quad (6.16)$$

The truncation error for this scheme is $O((\Delta\tau)^2)$, which is an improvement on the first order error found in both the implicit and explicit schemes. This improvement is a result of a cancelation of errors in the averaging process.

Rearranging equation (6.16) gives the Crank-Nicolson finite difference scheme at each time step, $\tau_i : i = 1, \ldots, n$,

$$-(\alpha/2)\widetilde{P}_i^{j+1} + (1+\alpha)\widetilde{P}_i^j - (\alpha/2)\widetilde{P}_i^{j-1} = (\alpha/2)\widetilde{P}_{i-1}^{j+1} + (1-\alpha)\widetilde{P}_{i-1}^j + (\alpha/2)\widetilde{P}_{i-1}^{j-1}, \quad (6.17)$$

where $j = 1, \ldots, m-1$ and $\alpha$ is defined in equation (6.11). The recurrence scheme

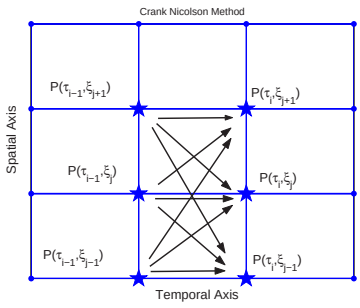


**Fig. 6.6:** The Crank-Nicolson method and the Crandall-Douglas method.

equation (6.17) may be represented as a matrix calculation in order to increase computational efficiency. At each time step, the matrix system for the Crank-

Nicolson method is given by,

$$\begin{pmatrix} 1+\alpha & -\alpha/2 & 0 & \dots & 0 \\ -\alpha/2 & \ddots & -\alpha/2 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & -\alpha/2 & 1+\alpha & -\alpha/2 \\ 0 & \dots & 0 & -\alpha/2 & 1+\alpha \end{pmatrix} \begin{pmatrix} \widetilde{P}_i^1 \\ \widetilde{P}_i^2 \\ \vdots \\ \widetilde{P}_i^{m-2} \\ \widetilde{P}_i^{m-1} \end{pmatrix} - \alpha/2 \begin{pmatrix} \widetilde{P}_i^0 \\ 0 \\ \vdots \\ 0 \\ \widetilde{P}_i^m \end{pmatrix} =$$
$$\begin{pmatrix} 1-\alpha & \alpha/2 & 0 & \dots & 0 \\ \alpha/2 & \ddots & \alpha/2 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \alpha/2 & 1-\alpha & \alpha/2 \\ 0 & \dots & 0 & \alpha/2 & 1-\alpha \end{pmatrix} \begin{pmatrix} \widetilde{P}_{i-1}^1 \\ \widetilde{P}_{i-1}^2 \\ \vdots \\ \widetilde{P}_{i-1}^{m-2} \\ \widetilde{P}_{i-1}^{m-1} \end{pmatrix} + \alpha/2 \begin{pmatrix} \widetilde{P}_{i-1}^0 \\ 0 \\ \vdots \\ 0 \\ \widetilde{P}_{i-1}^m \end{pmatrix} \quad (6.18)$$

which can be written as,

$$\mathbf{M}_{\mathrm{CN}} \widetilde{\boldsymbol{P}}_i - (\alpha/2)\boldsymbol{\beta}_i = \mathbf{N}_{\mathrm{CN}} \widetilde{\boldsymbol{P}}_{i-1} + (\alpha/2)\boldsymbol{\beta}_{i-1},$$

where $i = 1, \dots, n$, $\widetilde{\boldsymbol{P}}_i = (\widetilde{P}_i^1, \dots, \widetilde{P}_i^{m-1})$ is the vector of option prices, $\boldsymbol{\beta}_i = (\widetilde{P}_i^0, 0, \dots, 0, \widetilde{P}_i^m)$ is the vector of boundary values and $\mathbf{M}_{\mathrm{CN}}, \mathbf{N}_{\mathrm{CN}}$ are both $(m-1) \times (m-1)$ tridiagonal matrices.

The Crank-Nicolson method is more accurate than both the implicit and the explicit schemes because the truncation error is smaller. It also inherits the stability of the implicit scheme for positive values of $\alpha$ (Wilmott, Howison, and Dewynne 1995).

### 6.2.7 The Crandall-Douglas Method

The Crandall-Douglas finite difference method is an error reducing optimisation of the Crank-Nicolson method. The scheme further reduces the local error by using six grid points in a recurrence relation (see (Ames 1969) for a derivation). The Crandall-Douglas finite difference scheme at each time step, $\tau_i : i = 1, \dots, n$, is given by,

$$(1-6\alpha)\widetilde{P}_i^{j+1} + (10+12\alpha)\widetilde{P}_i^j + (1-6\alpha)\widetilde{P}_i^{j-1} = (1+6\alpha)\widetilde{P}_{i-1}^{j+1} + (10-12\alpha)\widetilde{P}_{i-1}^j + (1+6\alpha)\widetilde{P}_{i-1}^{j-1}, \quad (6.19)$$

where $j = 1, \dots, m-1$ and $\alpha$ is defined in equation (6.11). The recurrence equation (6.19) may also be represented as a matrix calculation to increase computational efficiency. At each time step, the matrix system for the Crandall-Douglas method

is given by,

$$\begin{pmatrix} 10+12\alpha & 1-6\alpha & 0 & \dots & 0 \\ 1-6\alpha & \ddots & 1-6\alpha & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & 1-6\alpha & 10+12\alpha & 1-6\alpha \\ 0 & \dots & 0 & 1-6\alpha & 10+12\alpha \end{pmatrix} \begin{pmatrix} \widetilde{P}_i^1 \\ \widetilde{P}_i^2 \\ \vdots \\ \widetilde{P}_i^{m-2} \\ \widetilde{P}_i^{m-1} \end{pmatrix} (1-6\alpha) \begin{pmatrix} \widetilde{P}_i^0 \\ 0 \\ \vdots \\ 0 \\ \widetilde{P}_i^m \end{pmatrix} =$$
$$\begin{pmatrix} 10-12\alpha & 1+6\alpha & 0 & \dots & 0 \\ 1+6\alpha & \ddots & 1+6\alpha & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & 1+6\alpha & 10-12\alpha & 1+6\alpha \\ 0 & \dots & 0 & 1+6\alpha & 10-12\alpha \end{pmatrix} \begin{pmatrix} \widetilde{P}_{i-1}^1 \\ \widetilde{P}_{i-1}^2 \\ \vdots \\ \widetilde{P}_{i-1}^{m-2} \\ \widetilde{P}_{i-1}^{m-1} \end{pmatrix} + (1+6\alpha) \begin{pmatrix} \widetilde{P}_{i-1}^0 \\ 0 \\ \vdots \\ 0 \\ \widetilde{P}_{i-1}^m \end{pmatrix} \tag{6.20}$$

which can be written as,

$$\mathbf{M}_{\mathrm{CD}} \widetilde{\boldsymbol{P}}_i + (1-6\alpha)\boldsymbol{\beta}_i = \mathbf{N}_{\mathrm{CD}} \widetilde{\boldsymbol{P}}_{i-1} + (1-6\alpha)\boldsymbol{\beta}_{i-1}, \tag{6.21}$$

where $i = 1, \dots, n$, $\widetilde{\boldsymbol{P}}_i = (\widetilde{P}_i^1, \dots, \widetilde{P}_i^{m-1})$ is the vector of option prices, $\boldsymbol{\beta}_i = (\widetilde{P}_i^0, 0, \dots, 0, \widetilde{P}_i^m)$ is the vector of boundary values and $\mathbf{M}_{\mathrm{CD}}, \mathbf{N}_{\mathrm{CD}}$ are both $(m-1) \times (m-1)$ tridiagonal matrices.

The method is stable when $\alpha > 1/6$ (McCartin and Labadie 2003).

### 6.2.8 Difficulties Encountered Using Finite Difference Methods for Pricing American Put Options

The European put option price satisfies the Black-Scholes PDE over the entire domain. Hence it is straightforward to calculate option values at each grid point in the truncated domain. In contrast to this, the American put option value satisfies the Black-Scholes PDE only in the continuation region ($\mathcal{C}$), and must be set equal to intrinsic value in the stopping region ($\mathcal{S}$). In order to implement this condition, we need to estimate the location of the early exercise boundary which separates these two regions.

In (Brennan and Schwartz 1977) a simple technique estimates the early exercise boundary at each time step. Unfortunately, this technique can only be used in conjunction with the explicit finite difference scheme, which has the stability condition $\alpha < 1/2$. This places a severe restriction on the choices of $\Delta\tau$ and $\Delta\xi$. In order to define the early exercise boundary accurately, a small value of $\Delta\xi$ is required. This, in turn, requires a small value of $\Delta\tau$, which results in a large number of computations.

Consequently, it would be beneficial to implement a similar technique for the implicit finite difference scheme, which does not have any restriction on the choice of step size. The implicit finite difference method calculates three option values at the same time step. As a consequence, a system of equations must be solved at

each time step, which complicates the estimation of the early exercise boundary. The projected successive over-relaxation (PSOR) method developed in (Wilmott, Howison, and Dewynne 1995) iteratively calculates the solution of this system of equations. The possibility of early exercise is incorporated as part of the solution and hence the value of the early exercise boundary is estimated implicitly.

In (McCartin and Labadie 2003) the solution domain is extended to the entire region. The Black-Scholes PDE with an early exercise boundary is transformed into the variational inequality system, equations (2.25) to (2.27). This system can be solved over the entire domain using the Elliot-Ockendon algorithm. This solution also contains an implicit approximation of the early exercise boundary.

## 6.3 Brennan and Schwartz Method

(Brennan and Schwartz 1977) were the first to use finite difference methods to approximate the price of the American put option, $P(t, S)$. They implemented an explicit finite difference scheme (Section 6.2.5). The solution of the option pricing problem, in general, requires backward recursion. Consequently, the use of the explicit finite difference method is preferable in that it is complementary to this process.

At each time step, the Brennan and Schwartz method calculates explicit prices as though it were pricing a European put option. However, these values need to be compared to the intrinsic value. At each node, we set the option value to the maximum of the calculated explicit value and the intrinsic value. Note that the calculated value is not trivially the Black-Scholes value because, near the early exercise boundary, the calculation will include a value (or values) from beyond the boundary.

The compatibility of the explicit scheme with the early exercise decision is revealed in the implementation of the algorithm. At each time step, $\tau_i$, a single option value is calculated from three earlier option values. It is then trivial to compare this value with the intrinsic value without requiring any recalculation. At each $\tau_i$, there exists a value $k$ such that

$$\begin{aligned} \widetilde{P}_i^j &\leq g_i^j \quad , \quad j \leq k \\ \widetilde{P}_i^j &> g_i^j \quad , \quad j > k. \end{aligned}$$

This implies that the early exercise boundary, $\widetilde{b}(\tau_i) \in [\xi^k, \xi^{k+1}]$. The exact value of $\widetilde{b}(\tau_i)$ is irrelevant. Once $k$ has been found, the option values for $\widetilde{P}_i^j, j = 1, \ldots, k$ are set equal to intrinsic value.

### 6.3.1 The Brennan and Schwartz Algorithm

The elements of the initial transformed value vector, $\widetilde{\boldsymbol{P}}_0$, are set equal to the transformed initial value (equation (6.4)),

$$\widetilde{P}_0^j = \widetilde{g}(0, \xi_j),$$

where $j = 1, \ldots, m-1$. Note that the definition of the intrinsic value in equation (6.4) explicitly defines a positive payoff. The following recursion is implemented for each subsequent time step, $\tau_i : i = 1, \ldots, n$:

1. Equations (6.5) and (6.6) give the boundary values $P_i^m$ and $P_i^0$ which are the non-zero elements of the boundary value vector, $\boldsymbol{\beta}_i$.
2. The remaining values of the vector $\widetilde{\boldsymbol{P}}_{\boldsymbol{i}}$ for $j = 1, \ldots, m-1$ are calculated using the explicit finite difference matrix scheme (6.12).
3. Each element in $\widetilde{\boldsymbol{P}}_{\boldsymbol{i}}$ is compared with the transformed intrinsic value $\widetilde{g}(\tau_i, \xi_j)$ and a new vector $\widetilde{\boldsymbol{P}}_{\boldsymbol{i}}$ is calculated by,

$$\widetilde{P}_i^j = \max\{\widetilde{P}_i^j, \widetilde{g}(\tau_i, \xi_j)\},$$

where $j = 1, \ldots, m-1$.

The final vector of transformed values, $\widetilde{\boldsymbol{P}}_{\boldsymbol{n}}$, can be used to produce a vector of initial values for the American put using equation (6.8). If the transformed initial stock price does not lie on a node, interpolation can be used to obtain an estimate at that point (see Section 6.2.2).

### 6.3.2 Convergence of Brennan and Schwartz Algorithm

The Brennan-Schwartz algorithm produces an estimate, $P_{\mathrm{BS}}(t, S)$, for the American put option price which converges as the number of time steps, $n$, and the number of spatial steps, $m$, tends towards infinity. Thus, the estimate converges to the true solution as the grid is made finer. However, the restriction, $\alpha < 1/2$, limits the choice of grid parameters.

#### Spatial Step Convergence

The Brennan-Schwartz estimates converge uniformly from below to the American option price as the number of spatial steps, $m$, is increased. See Figure 6.7.

#### Time step Convergence

The Brennan-Schwartz estimates converge uniformly from below to the American option price as the number of time steps, $n$, is increased. See Figure 6.8.

## 6.4 Projected Successive Over-Relaxation

The algorithm of Brennan and Schwartz (Brennan and Schwartz 1977) is simple and fast but unfortunately works only for the explicit scheme. This is problematic because of the attendant restriction on $\alpha$. On the other hand, the implicit nature of the other schemes (implicit, Crank-Nicolson and Crandall-Douglas scheme) do not allow the option values to be changed once the matrix system has been solved. As a consequence, comparing the calculated value with the intrinsic value is tedious and may affect all the option values at each time step.

For European options, implicit schemes are usually solved using a direct method such as LU decomposition or matrix inversion. However, they can also be solved using an iterative scheme called the successive over-relaxation (SOR) method, which

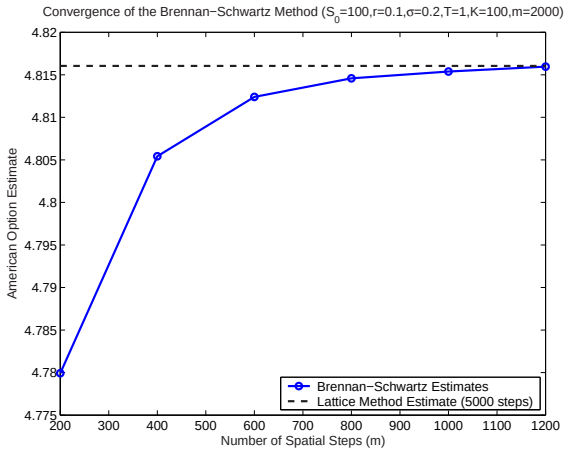


**Fig. 6.7:** The convergence behaviour for Brennan-Schwartz estimates as the number of spatial steps, $m$, is increased. There is a fixed number of time steps ($n = 100$). The trinomial tree value using 5000 time steps is assumed to be the true value of the option.

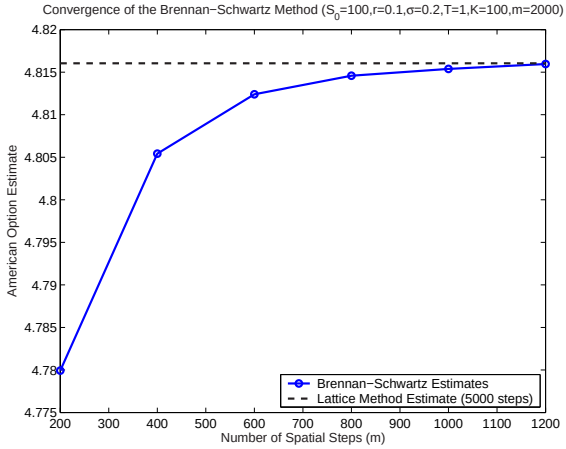


**Fig. 6.8:** The convergence behaviour for Brennan-Schwartz estimates as the number of time steps, $n$, is increased. There is a fixed number of spatial steps ($M = 50000$). The trinomial tree value using 5000 time steps is assumed to be the true value of the option.

is derived from the Jacobi numerical method (Wilmott, Howison, and Dewynne 1995, Wilmott 2000). These iterative schemes are used to solve the general matrix equation,

$$\mathcal{A}\boldsymbol{S} = \boldsymbol{y}. \tag{6.22}$$

The SOR method begins with an initial estimate for the solution, $\boldsymbol{S_0}$, and follows some iteration scheme until a convergence criteria is met, e.g. $|\boldsymbol{S_{k+1}} - \boldsymbol{S_k}| < \epsilon$ where $\epsilon$ is some predefined tolerance. This iterative scheme can be adapted to implicit methods for the American put option, where it is called the projected successive

over-relaxation (PSOR) method.

### 6.4.1 Jacobi Method

The Jacobi method is a simple iterative scheme to solve matrix equations of the form (6.22). The Jacobi method works for all invertible matrices. We will restrict attention here to $(m \times m)$ tridiagonal matrices, since all the implicit schemes mentioned above result in tridiagonal matrices. The iterative scheme recursively calculates estimates $\boldsymbol{x_k} = (x_k^1, \ldots, x_k^m)$, beginning with an initial vector $\boldsymbol{x_0}$. It is possible to rewrite equation (6.22) componentwise as,

$$\begin{aligned} a_{11}x^1 + a_{12}x^2 &= y^1 \\ a_{21}x^1 + a_{22}x^2 + a_{23}x^3 &= y^1 \\ &\cdots \\ a_{m-1,m}x^{m-1} + a_{m,m}x^m &= y^m. \end{aligned}$$

Which, can then be rewritten as,

$$\begin{aligned} a_{11}x^1 &= y^1 - \big(a_{12}x^2\big) \\ a_{22}x^2 &= y^1 - \big(a_{21}x^1 + a_{23}x^3\big) \\ &\cdots \\ a_{m,m}x^m &= y^m - \big(a_{m-1,m}x^{m-1}\big). \end{aligned}$$

The iteration scheme is obvious. We use previous estimates, $x_k^1, \ldots, x_k^m$, on the right of the system of equations to calculate new estimates, $x_{k+1}^1, \ldots, x_{k+1}^m$, on the left. The Jacobi method for this system is,

$$\begin{aligned} x_{k+1}^1 &= \big(y^1 - \big(a_{12}x_k^2\big)\big)/a_{11} \\ x_{k+1}^2 &= \big(y^1 - \big(a_{21}x_k^1 + a_{23}x_k^3\big)\big)/a_{22} \\ &\cdots \\ x_{k+1}^m &= \big(y^m - \big(a_{m-1,m}x_k^{m-1}\big)\big)/a_{m,m}. \end{aligned}$$

We will iterate on $k$ until we converge to some desired tolerance, $|\boldsymbol{x}_{k+1} - \boldsymbol{x}_k| < \epsilon$. Some of the values on the right may have been calculated, i.e. $x_{k+1}^{j-1}$ is already known when we calculate $x_{k+1}^j$. Using this updated value leads to the Gauss-Seidel technique with an attendant improved convergence.

### 6.4.2 SOR Method

The successive over-relaxation method is a further adaptation on the original Jacobi method which improves the speed of convergence (Wilmott 2000). An improved convergence is achieved by using a weight factor, $\omega$, called the relaxation parameter. The system for this method is given by,

$$\begin{aligned} x_{k+1}^1 &= x_k^1 + \omega\big(y^1 - \big(a_{12}x_k^2\big)\big)/a_{11} \\ x_{k+1}^2 &= x_k^2 + \omega\big(y^1 - \big(a_{21}x_{k+1}^1 + a_{23}x_k^3\big)\big)/a_{22} \\ &\cdots \\ x_{k+1}^m &= x_k^m + \omega\big(y^m - \big(a_{m-1,m}x_{k+1}^{m-1}\big)\big)/a_{m,m}. \end{aligned}$$

The value of $\omega$ depends on the convergence behaviour. If convergence is monotonic then $1 < \omega < 2$, otherwise $\omega < 1$. Note that the Gauss-Seidel technique has also been implemented. All of the implicit methods can be solved using the SOR method by choosing the relevant form for $\mathcal{A}$, $\boldsymbol{x}$, and $\boldsymbol{y}$.

**Implicit Scheme**

$$\begin{aligned} \boldsymbol{x} &= \widetilde{\boldsymbol{P}}_{\boldsymbol{i}} \\ \mathcal{A} &= \mathbf{M}_{\mathrm{I}} \\ \boldsymbol{y} &= \widetilde{\boldsymbol{P}}_{\boldsymbol{i-1}} - \alpha\boldsymbol{\beta}_{\boldsymbol{i}} \end{aligned} \tag{6.23}$$

**Crank-Nicolson Scheme**

$$\begin{aligned} \boldsymbol{x} &= \widetilde{\boldsymbol{P}}_{\boldsymbol{i}} \\ \mathcal{A} &= \mathbf{M}_{\mathrm{CN}} \\ \boldsymbol{y} &= \mathbf{N}_{\mathrm{CN}}\widetilde{\boldsymbol{P}}_{\boldsymbol{i-1}} + (\alpha/\mathbf{2})(\boldsymbol{\beta}_{\boldsymbol{i-1}} + \boldsymbol{\beta}_{\boldsymbol{i}}) \end{aligned} \tag{6.24}$$

**Crandall-Douglas**

$$\begin{aligned} \boldsymbol{x} &= \widetilde{\boldsymbol{P}}_{\boldsymbol{i}} \\ \mathcal{A} &= \mathbf{M}_{\mathrm{CD}} \\ \boldsymbol{y} &= \mathbf{N}_{\mathrm{CD}}\widetilde{\boldsymbol{P}}_{\boldsymbol{i-1}} + (\mathbf{1} - \mathbf{6}\alpha)(\boldsymbol{\beta}_{\boldsymbol{i-1}} - \boldsymbol{\beta}_{\boldsymbol{i}}) \end{aligned} \tag{6.25}$$

### 6.4.3 The PSOR Method for the American Put Option

The SOR method is a popular method for solving the matrix systems that are formed in finite difference schemes. It is relatively simple to implement and, in this application, can also be applied to American options. As previously noted, implicit schemes have the following drawback: it is not possible to replace a calculated grid value which is less than intrinsic value at that point, with the intrinsic value as an integral part of the calculation. The iterative nature of the SOR method sidesteps this issue. On each iteration we are free to assign the value at each node to be the maximum of the intrinsic value and the calculated value using one of the systems: (6.23), (6.24) or (6.25). This method is called the projected successive over-relaxation method and an application of it to the American put option problem can be found in (Wilmott, Howison, and Dewynne 1995, Wilmott 2000). In applying PSOR, we modify our system of equations to,

$$\begin{aligned} \widetilde{P}^1_{i,k+1} &= \max\{\widetilde{g}(\tau_i,\xi_1), \widetilde{P}^1_{i,k} + \omega\big(y^1 - \big(a_{12}\widetilde{P}^2_{i,k}\big)\big)/a_{11}\} \\ \widetilde{P}^2_{i,k+1} &= \max\{\widetilde{g}(\tau_i,\xi_2), \widetilde{P}^2_{i,k} + \omega\big(y^1 - \big(a_{21}\widetilde{P}^1_{i,k+1} + a_{23}\widetilde{P}^3_{i,k}\big)\big)/a_{22}\} \\ &\cdots \\ \widetilde{P}^m_{i,k+1} &= \max\{\widetilde{g}(\tau_i,\xi_m), \widetilde{P}^m_{i,k} + \omega\big(y^m - \big(a_{m-1,m}\widetilde{P}^{m-1}_{i,k+1}\big)\big)/a_{m,m}\}, \end{aligned}$$

where $\boldsymbol{x} = \widetilde{\boldsymbol{P}}_{\boldsymbol{i}}$, and $\boldsymbol{y}$, $\boldsymbol{\mathcal{A}}$ are chosen from (6.23),(6.24) and (6.25).

The initial vector, $\widetilde{\boldsymbol{P}}_0$, is set equal to the transformed intrinsic value,

$$\widetilde{P}_0^j = \widetilde{g}(0, \xi_j),$$

where $j = 1, \ldots, m-1$. The following recursion is followed for each subsequent time step, $\tau_i : i = 1, \ldots, n$:

1. Calculate the boundary vector $\boldsymbol{\beta}_i$ from equations (6.5) and (6.6).

2. Implement the PSOR method to recursively calculate estimates, $\widetilde{\boldsymbol{P}}_i^k$, until the convergence criterion, $|\widetilde{\boldsymbol{P}}_i^k - \widetilde{\boldsymbol{P}}_i^{k-1}| \leq \epsilon$, is met. Use the final estimate from the preceding time step as the initial guess, i.e. $\widetilde{\boldsymbol{P}}_i^0 = \widetilde{\boldsymbol{P}}_{i-1}$. The transformed option premium vector is set equal to the final iteration, $\widetilde{\boldsymbol{P}}_i = \widetilde{\boldsymbol{P}}_i^k$.

The final vector of transformed values, $\widetilde{\boldsymbol{P}}_{\boldsymbol{n}}$, can be used to produce a vector of initial values for the American put using equation (6.8). If the transformed initial stock price does not lie on a node interpolation can be used to obtain an estimate at that point (see Section 6.2.2).

### 6.4.4 Convergence of the PSOR Method

The PSOR algorithm produces an estimate, $P_{\text{PSOR}}(t, S)$, for the American put option price which converges as the number of time steps, $n$, and the number of spatial steps, $m$, tends towards infinity. Thus, the estimate converges to the true solution as the grid is made finer.

#### Spatial Step Convergence

The PSOR estimates converge uniformly from below to the American option price as the number of spatial steps, $m$, is increased. See Figure 6.9.

#### Time step Convergence

The PSOR estimates converge uniformly from below to the American option price as the number of time steps, $n$, is increased. See Figure 6.10.

## 6.5 Variational Inequality

The finite difference methods discussed in this chapter estimate the value of the American put option by explicitly calculating where exercise is optimal. In essence, the position of the early exercise boundary value can be found at each time step. This free boundary is unknown *a priori* and it may be useful to reformulate the problem in order to eliminate the explicit dependence on it (Wilmott, Howison, and Dewynne 1995). Independence can be achieved by rewriting the free boundary problem as a variational inequality (equations (2.25) to (2.27)).

The variational inequality formulation is a framework for solving a class of equilibrium problems in mathematics. Solving the variational inequality makes no explicit reference to the free boundary. Instead, the embedded free boundary is implicit

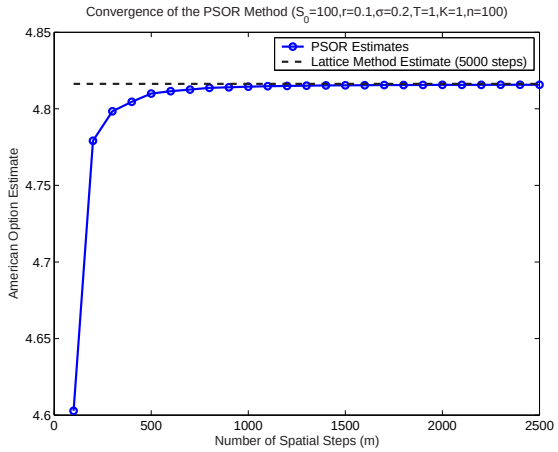


**Fig. 6.9:** The convergence behaviour for PSOR estimates as the number of spatial steps, $m$, is increased. There is a fixed number of time steps ($n = 100$). The trinomial tree value using 5000 time steps is assumed to be the true value of the option.

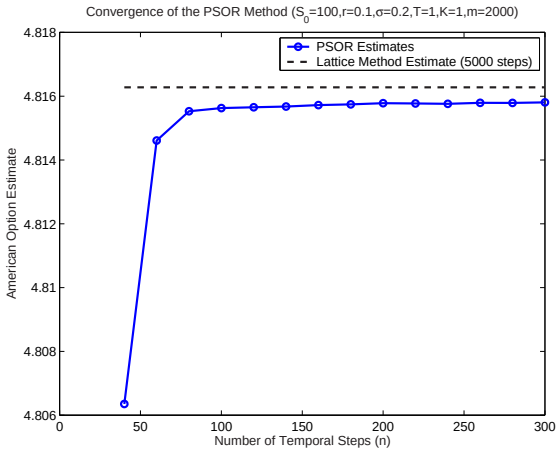


**Fig. 6.10:** The convergence behaviour for PSOR estimates as the number of time steps, $n$, is increased. There is a fixed number of spatial steps ($m = 2000$). The trinomial tree value using 5000 time steps is assumed to be the true value of the option.

in the conditions that are simultaneously satisfied in the continuation region and the stopping region. In Chapter 2 we showed that,

$$\frac{\partial P}{\partial t} + rS\frac{\partial P}{\partial S} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 P}{\partial S^2} - rP \leq 0, \tag{6.26}$$

over the entire domain. The value of the American put option also semi-dominates the intrinsic value, giving a second inequality,

$$P(t, S) - g(t, S) \geq 0.$$

The final equation (2.25) combines the behaviour of the option in the continuation region and the stopping region:

1. In the continuation region the Black-Scholes equation is identically equal to zero.

2. In the stopping region the value of the American put option is identically equal to the intrinsic value, $P(t,S) - g(t,S) = 0$.

Thus,

$$\Big(\frac{\partial P}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 P}{\partial S^2} + rS\frac{\partial P}{\partial S} - rP\Big)\Big(P(t,S) - g(t,S)\Big) = 0,$$

over the entire domain. We can solve the variational inequality by discretising the PDE and converting the resulting system into a set of *linear complementarity problems*.

### 6.5.1 The American Put as a Linear Complementarity Problem

The linear complementarity problem (LCP) is a specific type of optimisation problem in which a system with a set of constraints but no objective function is considered. In this instance, we seek two vectors, $\boldsymbol{w}$ and $\boldsymbol{z}$, which satisfy,

$$\begin{aligned} \boldsymbol{q} &= \boldsymbol{w} - \mathcal{A}\boldsymbol{z} && (6.27)\\ \boldsymbol{w} &\geq 0 && (6.28)\\ \boldsymbol{z} &\geq 0 && (6.29)\\ \boldsymbol{w}^{\mathrm{T}}\boldsymbol{z} &= 0, && (6.30) \end{aligned}$$

where $\boldsymbol{q}$ and $\mathcal{A}$ are known. The American put option pricing problem can be represented as an LCP. After reducing the Black-Scholes equation to the heat equation, we first discretise it and then choose one of the finite difference schemes from this chapter. Once we have chosen a scheme, an LCP must be constructed at each time step,

$$\begin{aligned} \boldsymbol{z} &:= \widetilde{\boldsymbol{P}}_i - \widetilde{\boldsymbol{g}}_i \\ \boldsymbol{q} &:= \mathbf{M}\widetilde{\boldsymbol{g}}_{\mathbf{i}} - \mathbf{N}\widetilde{\boldsymbol{P}}_{\mathbf{i-1}} \\ \boldsymbol{w} &:= \mathbf{M}\boldsymbol{z} + \boldsymbol{q} \\ \mathcal{A} &:= \mathbf{M}, \end{aligned} \qquad (6.31)$$

where $\mathbf{M}$, $\mathbf{N}$, are chosen based on the finite difference method used:

$$\begin{aligned} \text{Implicit} \quad & \mathbf{M} = \mathbf{M}_{\mathrm{I}}, \quad \mathbf{N} = \mathbf{I} \\ \text{Crank-Nicolson} \quad & \mathbf{M} = \mathbf{M}_{\mathrm{CN}}, \quad \mathbf{N} = \mathbf{N}_{\mathrm{CN}} \\ \text{Crandall-Douglas} \quad & \mathbf{M} = \mathbf{M}_{\mathrm{CD}}, \quad \mathbf{N} = \mathbf{N}_{\mathrm{CD}}. \end{aligned}$$

When estimating the value of the American put option, we rewrite the finite difference scheme as an LCP at each time step. As a result, we will estimate the price of an American put option by solving $n$ LCPs. We must therefore find the fastest

method available to solve the LCPs. Many are available to solve the LCP (Cottle and Pang 1992), although most of these are time consuming. Cryer's algorithm (Cryer 1983) is an efficient method that has been developed to solve the LCP for tridiagonal Minkowski matrices. A tridiagonal matrix, $\mathbf{M}$, is a *Minkowski matrix* if it satisfies,

- $m_{ij} = 0, \quad \text{if } |i-j| > 1$
- $m_{ij} \leq 0, \quad \text{if } i \neq j$
- $\mathbf{M}$ has positive principal minors (Strictly diagonally dominant with positive diagonals).

This method has a convergence of $\mathcal{O}(n^2)$, which is faster than most algorithms. However, there is a special property of the American put option which allows the LCP to be solved with linear convergence $\mathcal{O}(n)$ (McCartin and Labadie 2003) using the Elliot-Ockendon algorithm (Elliot and Ockendon 1982).

### 6.5.2 Elliot-Ockendon Algorithm

The (Elliot and Ockendon 1982) algorithm is an efficient method which was developed to solve a specific type of LCP. (McCartin and Labadie 2003) showed that the LCP resulting from the finite difference scheme for an American put option obeys the conditions required for application of the Elliot-Ockendon algorithm. The solution algorithm follows from the proof of the properties of the problem,

**Proposition 6.5.1.** *Consider the linear complementarity problem,*

$$\boldsymbol{w} = \boldsymbol{q} + \mathcal{A}\boldsymbol{z} \geq 0, \quad \boldsymbol{z} \geq 0, \quad \boldsymbol{w}^T\boldsymbol{z} = 0, \tag{6.32}$$

*Let $\boldsymbol{q} \in \mathbb{R}^m$, and $\mathcal{A} \in \mathbb{R}^{m\times m}$ be a tridiagonal matrix with diagonal elements $a_i > 0 : i = 1, \ldots, m$, super-diagonal elements $c_i < 0 : i = 1, \ldots, m-1$, and sub-diagonal elements $b_i < 0 : i = 2, \ldots, m$ which satisfy,*

$$a_1 > -c_1, \quad a_m > -b_m, \quad a_j \geq -(b_j + c_j),$$

*where $j = 2, \ldots, m-1$. Then*

1. There exists a unique solution to problem.
2. If $q_i < 0 : i = 1, \ldots, k_0$, and $q_i \geq 0 : i = k_0 + 1, \ldots, n$, then $\exists k > k_0$ such that $z_i = 0 : i > k$ and $z_i > 0 : \ i \leq k$.

*Proof.* See (Elliot and Ockendon 1982, p. 115). □

If these conditions are satisfied, we can simplify the form of the LCP. Then there exists $k$ such that $z_i > 0 : \ i \leq k$ which implies that $w_i = 0 : \ i \leq k$. We can rewrite (6.32) as,

$$\begin{aligned} a_1 z_1 + c_1 z_2 + q_1 &= 0 \\ b_i z_{i-1} + a_i z_i + c_i z_{i+1} + q_i &= 0, \quad i = 2, \ldots, k-1 \\ b_k z_{k-1} + a_k z_k + q_k &= 0. \end{aligned}$$

This value of $k$ relates to the position of the early exercise boundary, $b(t)$.

The resulting set of equations can be solved easily using standard Gaussian elimination to find the $z_i : i = 1, \ldots, k$. The remaining $z_i = 0 : i > k$ which implies that $w_i \geq 0 : \quad i > k$. This results in a set of equations for the the remaining inequalities (6.32),

$$\begin{aligned} b_{k+1} z_k + q_{k+1} &= w_{k+1} \geq 0 \\ q_i &= w_i, \end{aligned}$$

where $i > k + 1$. Thus the LCP is reduced to a set of simple equations. The only difficulty is locating $k$, which is found when $bz_k + q_{k+1} \geq 0$.

**The Algorithm**

**Decomposition:**

$$\begin{aligned} i = 1, \quad \alpha_1 = a_1, \quad f_1 = -q_1/a_1 \\ \text{Step:} \quad i = i + 1 \\ \gamma_{i-1} = c_{i-1}/\alpha_{i-1} \\ \alpha_i = a_i - b_i \gamma_{i-1} \\ f_i = -(q_i + b_i f_{i-1})/\alpha_i \end{aligned}$$

If $f_i > -q_{i+1}/b_{i+1}$ then go to Step.

**Backsolving:**

$$\begin{aligned} k &= i \\ z_k &= f_k \\ \text{Step:} \quad i &= i - 1 \\ z_i &= f_i - \gamma_i z_{i+1} \end{aligned}$$

If $i > 1$ then go to Step.

### 6.5.3 Convergence of the Elliot-Ockendon Method

The Elliot-Ockendon algorithm produces an estimate, $P_{\text{EO}}(t, S)$, for the American put option price, which converges as the number of time steps, $n$, and the number of spatial steps, $m$, tends towards infinity. Thus, the estimate converges to the true solution as the grid is made finer.

**Spatial Step Convergence**

The Elliot-Ockendon estimates converge uniformly from below to the American option price as the number of spatial steps, $m$, is increased. See Figure 6.11.

**Time step Convergence**

The Elliot-Ockendon estimates converge uniformly from below to the American option price as the number of time steps, $n$, is increased. See Figure 6.12.

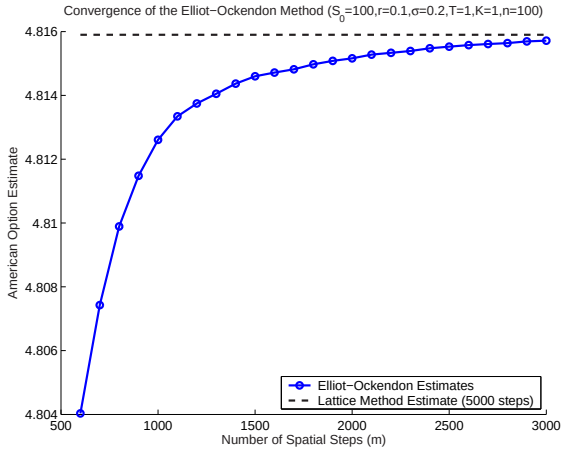


**Fig. 6.11:** The convergence behaviour for Elliot-Ockendon estimates as the number of spatial steps, $m$, is increased. There is a fixed number of time steps ($n = 100$). The trinomial tree value using 5000 time steps is assumed to be the true value of the option.

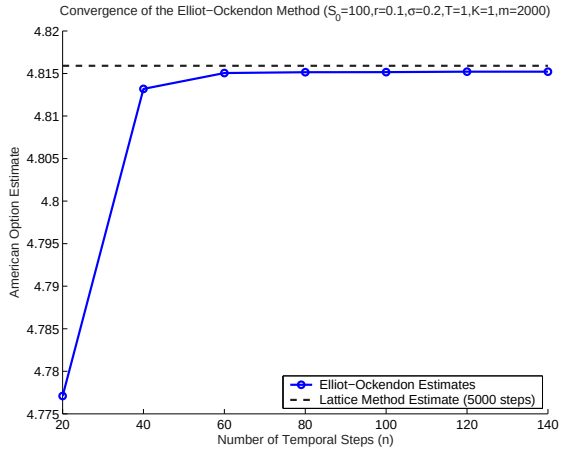


**Fig. 6.12:** The convergence behaviour for Elliot-Ockendon estimates as the number of time steps, $n$, is increased. There is a fixed number of spatial steps ($m = 2000$). The trinomial tree value using 5000 time steps is assumed to be the true value of the option.

# Chapter 7

# The Early Exercise Premium

## 7.1 Introduction

Semi-analytical solutions to the American put pricing problem invariably contain an integral representation of all or part of the value. Examples of this can be found in (Carr, Jarrow, and Myneni 1992), (Jacka 1991) and (Kim 1990), amongst others. In all three approaches, the American put option value, $P(t,S)$, is decomposed into a linear combination of the equivalent European put option value, $p(t,S)$, and an early exercise premium, $e(t,S)$, as in equation (2.28). The discounted expected interest earned while the stock price is below the early exercise boundary, $b(t)$, is equivalent in value to the early exercise premium. (Huang, Subrahmanyam, and Yu 1996) detailed a method to approximate the early exercise premium using numerical integration.

The early exercise premium can be expressed as an integral containing the cumulative standard normal distribution function,

$$e(t,S) = rK\int_t^T \exp\{-r(T-\tau)\}N\left(\frac{\ln(b(\tau)/S) - (r-\sigma^2/2)(T-\tau)}{\sigma\sqrt{T-\tau}}\right)d\tau. \quad (7.1)$$

Clearly, estimation of the integral requires the functional form of the early exercise boundary over the life of the option. Once the early exercise boundary is calculated, numerical integration can be used to calculate an estimate of the early exercise premium, $\widetilde{e}(t,S)$[1]. An estimate for the American put option value follows from equation (2.28).

We may calculate an estimation of the early exercise boundary by employing the dynamic programming techniques of Chapter 4 and Chapter 5. However, we use numerical integration estimates, $\widetilde{P}(t)$, rather than Monte Carlo estimates, $\widehat{P}(t)$. Consequently, we discretise the life of the option into $n$ intervals, $t = t_0 < t_1 < \cdots < t_n = T$, and use backward recursion to calculate $\widetilde{b}(t_i)$. The algorithm employed is similar to that of the GVW method (Chapter 5.3), which solves the smooth pasting condition (2.19) for the early exercise boundary value.

The algorithm to estimate the early exercise premium can be divided into two related procedures,

1. Estimation of the early exercise boundary, $\{\widetilde{b}(t_i)\}_{i=0,\ldots,n}$, using numerical integration.

[1] All integral approximation estimates are denoted by a tilde.

2. Estimation of the early exercise premium, $\widetilde{e}(t,S)$, using numerical integration and the previously estimated early exercise boundary, $\left\{\widetilde{b}(t_i)\right\}_{i=0,\ldots,n}$.

When estimating the early exercise boundary we must locate the largest stock price where the value of the American put option is equal to its intrinsic value. As a consequence, we need a method to estimate the value of the American put option for a variety of stock prices. This is achieved using numerical integration of equation (7.1).

## 7.2 Estimation of the Early Exercise Boundary

We use backward recursion, in a similar manner to the GVW algorithm, to calculate a discrete estimation, $\widetilde{b}_i \equiv \widetilde{b}(t_i)$, of the early exercise boundary at each time step $t_i : i = 0,\ldots,n$. This estimate is calculated recursively using the the smooth pasting condition (2.19). As usual, we start by using the boundary value at expiry, $t_n : b_n = K$, which is known. We then recursively calculate earlier estimates, $\widetilde{b}_i : i = n-1,\ldots,0$. The recursion process examines the function, $f(b)$, where

$$f(b) = g(b) - \widetilde{P}_i(b). \tag{7.2}$$

The function, $g(b)$, is the payoff of the option given by equation (2.7) and $\widetilde{P}_i(\cdot)$ is the numerical estimate for the American put option value. The put option value is calculated using numerical integration at time $t_i$, and the already calculated portion of the early exercise boundary, $\left\{\widehat{b}_j\right\}_{j=i+1,\ldots,n}$. In estimating the early exercise boundary, we solve for the largest value, $b^* : f(b^*) = 0$.

### 7.2.1 Difficulties Encountered in Locating the Early Exercise Boundary

The initial difficulty encountered is that the first derivative of $f(b)$ cannot be calculated analytically for all values of $b$. In particular,

$$\frac{\partial f}{\partial S} \to -\infty \quad \text{for all} \quad S > b^* \text{ as } t \to T \tag{7.3}$$

$$\frac{\partial f}{\partial S} = 0 \quad \text{for all} \quad S < b^* \text{ for all } t \in [0,T]. \tag{7.4}$$

In addition, the first derivative of the early exercise boundary itself tends to infinity as $T - t$ tends to zero. This excludes many sophisticated and commonly used root finding techniques such as Newton-Rhapson for estimating $f(b^*) = 0$. We are compelled to use techniques that do not employ the first derivative.

Some of the root finding techniques employ an interpolation method to fit a curve between successive guesses. This usually identifies the root more quickly than the bisection method. These techniques only work well on functions that are well behaved. Unfortunately, equation (7.2) is not well behaved.

The difficulty arises as the accuracy of our approximation increases (see Fig 7.2). For all values below the early exercise boundary the function $f$ is close to zero. As the approximation improves, this function rapidly tends towards zero,

$$\lim_{\widetilde{P}(S,t)\to P(S,t)} f(b) = 0,\ 0 \le b \le b^*,$$

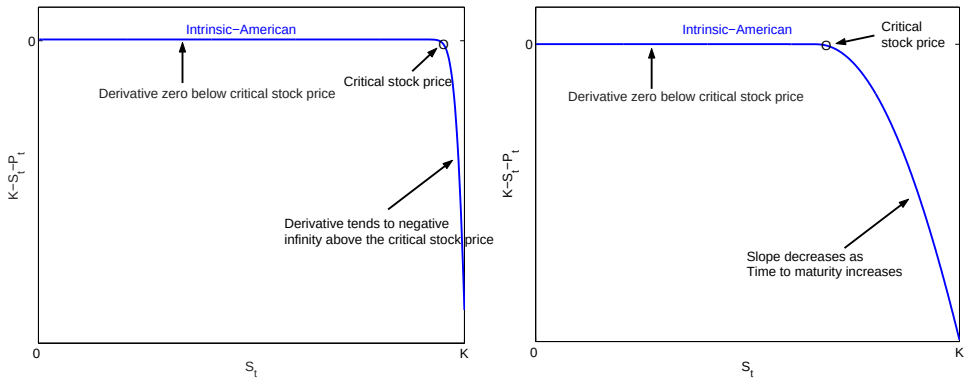


**Fig. 7.1:** The function $f(b)$ at one time step before expiry. The slope increases as time to expiry decreases.

where $f(b)$ is defined by equation (7.2). When searching for the root of equation (7.2), we are in effect looking for the largest value, $b^* : f(b^*) = 0$. The function decreases rapidly after this value and its slope tends to negative infinity with alarming rapidity when close to expiry. This is the primary source of difficulty when trying to solve for the root of equation (7.2).

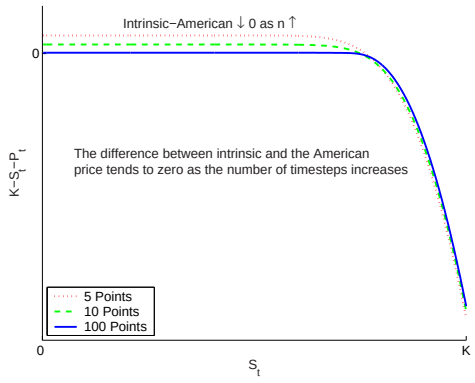


**Fig. 7.2:** The function $f(b)$ with an increasing number of time steps $n$.

In many of the methods which we will discuss, the current best approximation to the root is used as a decision-making tool. Below the early exercise boundary, the function $f(b)$ is close to zero. This complicates the decision making, and often results in increased iterations to find the root. The decision making process and the poor characteristics of $f(b)$ lead to slow performance for interpolation techniques.

### 7.2.2 Root Finding Techniques for Estimating the Early Exercise Boundary

When searching for the early exercise boundary, the convergence speed can be improved by reducing the search domain, $[0, K]$. (Merton 1973) showed that the early

exercise boundary of a perpetual American put option is,

$$B_\infty = K\frac{2\frac{r}{\sigma^2}}{2\frac{r}{\sigma^2}+1}.$$

The "perpetual early exercise boundary" is a lower bound for the corresponding American put option with finite expiry $T$, and, as a consequence, we can apply a root finding technique on the interval $[B_\infty, K]$ to find the zero of equation (7.2). We must then decide which root finding technique to use on the interval. Bracketing methods produce a sequence of intervals $\{[\alpha_n, \beta_n]\}$ such that,

$$[\alpha_{n+1}, \beta_{n+1}] \subseteq [\alpha_n, \beta_n], \quad f(\alpha_n)f(\beta_n) \leq 0,\ n = 1, \ldots.$$

Other commonly used methods use the derivative of the function to enhance the performance of the root finding technique, although we will not employ these types of methods because of the difficulties mentioned above. As a consequence, we will first consider the following standard bracketing methods: bisection method, secant method, and the false position method (Press, Teukolsky, Vetterling, and Flannery 1999). We then examine the bracketing methods of (Bus and Dekker 1975) and (Brent 1973). (The details of these algorithms can be found in Appendix D.)

### Bisection Method

The bisection method is the simplest technique for finding the root of a function (Press, Teukolsky, Vetterling, and Flannery 1999). It always locates the zero provided the initial interval contains the zero. Thus, to find the zero of $f(b)$ in equation (7.2) we require an interval, $[\alpha, \beta]$, such that

$$f(\alpha)f(\beta) \leq 0 \tag{7.5}$$

At each step the function of the midpoint is calculated and a new interval is chosen so that equation (7.5) holds. The successive halving of intervals leads to convergence in a fixed number of steps, $\log_2\left(\frac{\alpha-\beta}{\epsilon}\right)$, where $\epsilon$ is the desired tolerance.

### Secant Method and False Position Method

The secant method also requires an interval $[\alpha, \beta]$, where $\alpha$ is less than $\beta$ and the initial condition (7.5) holds. The tacit assumption is that the function is approximately linear between the two points (Press, Teukolsky, Vetterling, and Flannery 1999). We proceed by fitting a straight line between the two points and then choose the next lower bound of the interval to be the intercept of this straight line. This is in contrast with choosing the midpoint of the interval, as in the bisection method. This results in a new value $\gamma$ between $\alpha$ and $\beta$,

$$\gamma = \beta - \frac{\beta-\alpha}{f(\beta)-f(\alpha)}f(\beta).$$

We then evaluate $f(\gamma)$ and terminate the procedure when it is sufficiently close to zero.

The secant method does not preserve the bracketing property, although the false position method is a modification which ensures that the root is bracketed. However, the difficulty with both methods is that convergence can take very long if the function is not well behaved. In our case, the function in equation (7.2) is just such an example (see Figure 7.3). As a consequence we should not use either of these root finding methods.

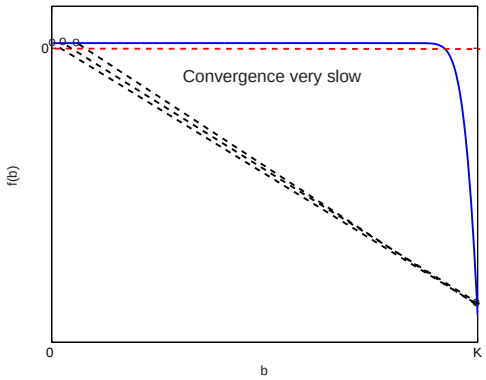


**Fig. 7.3:** Convergence of the Secant method for finding the early exercise boundary. The function $f(b)$ is not well behaved which drastically reduces the convergence speed.

### Dekker's Algorithm

Dekker's algorithm (Bus and Dekker 1975) combines the "speed" of the secant method with the guaranteed convergence of the bisection method (the algorithm is detailed in Appendix D.1). The method performs secant steps (linear interpolation) unless a bisection step (halving the interval) leads to enhanced convergence.

### Bus-Dekker's Algorithm M

Algorithm M is very similar to Dekker's method although it uses rational interpolation instead of linear interpolation (the algorithm is detailed in Appendix D.2). In the worst case, the rate of convergence is four times as slow as the bisection method. This is a result of the algorithm reverting to a bisection method if the interval has not been halved within four steps.

### Brent's Method

Brent's method (Brent 1973) extends Dekker's algorithm by using inverse quadratic interpolation at the interpolation step (the algorithm is detailed in Appendix D.3). It also includes a few minor alterations to improve the efficiency of the method.

## 7.2.3 Choosing a Root Finding Technique

When examining the hybrid bisection-interpolation methods, we need to decide whether the calculation overhead using the interpolation is offset by an improved

rate of convergence in comparison to the bisection method. The Bus-Dekker algorithm M guarantees a bisection step if four interpolation steps has not halved the search interval. There are, however, far more calculations involved with the algorithm M than with the bisection method. Figure 7.4 shows that although there are, on average, fewer iterations for the algorithm M in comparison to the bisection method, the method has a longer runtime. As a consequence, we would use the bisection method in preference to any of the hybrid algorithms.

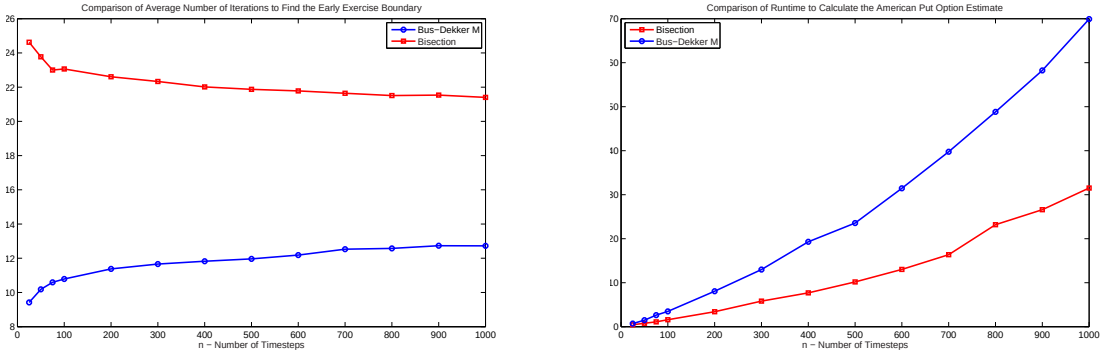


**Fig. 7.4:** Comparison of the bisection method and Bus-Dekker's algorithm M. The figure on the left compares the average number of iterations to locate the early exercise boundary. The figure on the right compares the actual method runtimes when calculating the estimate for the American put option.

## 7.3 Integral Approximation Techniques

In (Carr, Jarrow, and Myneni 1992, Jacka 1991, Kim 1990), the early exercise premium (2.29) was expressed as an integral containing the cumulative standard normal distribution function,

$$e(t,S) \quad = \quad \int_t^T f(\tau,S,b)\,d\tau \tag{7.6}$$

where

$$f(\tau,S,b) = rK\exp\{-r(T-\tau)\}N\left(\frac{\ln(b(\tau)/S)-(r-\sigma^2/2)(T-\tau)}{\sigma\sqrt{T-\tau}}\right). \tag{7.7}$$

The integral (7.6) cannot be solved analytically, but we can calculate a numerical estimate, $\widetilde{e}(t,S)$, using quadrature techniques (see (Davis and Rabinowitz 1984) for a comprehensive study of numerical integration). In numerical quadrature an integral approximation may be calculated using a linear combination of the integrand,

$$\int_a^b f(t)dt \approx w_0 f(t_0) + w_1 f(t_1) + \cdots + w_n f(t_n) = \sum_{i=0}^{n} w_i f(t_i),$$

where $t_0, \ldots, t_n$ are points in the interval $[a, b]$, and $w_0, \ldots, w_n$ are some associated weights. We discretise the remaining life of the option, $T - t$, into $n$ intervals: $t = t_0 < t_1 < \cdots < t_n = T$. The weights and, in some instances, the location of the integration points are specific to the quadrature method.

When calculating $\widetilde{e}(t, S)$, at an arbitrary time $t_i$, we will require the value of the integrand (7.7) at all times $t_j : i \leq j \leq n$. Consequently, we need the estimated values of the early exercise boundary, $\widehat{b}(t_j) : j = i, \ldots, n$ (Section 7.2 details this procedure). Hence, the partitioning of the integration region is determined by the partitioning used in the estimation of the early exercise region, unless interpolation is performed.

When performing numerical quadrature, the main factors influencing the choice of method are: computational speed and accuracy. Thus, in general, we would select the method with the fastest execution, but whose estimate lies within our desired tolerance.

The simplest quadrature methods are the closed *Newton-Cotes* quadrature formulae such as the Trapezoidal rule and the Simpson's rule. They are called closed methods because the endpoints of the interval are included in the selection of the integration points. One drawback of the Newton-Cotes formulae is that only a few integration points are used in the calculation of the estimate. *Composite (extended) rules* can, however, be formed from the Newton-Cotes formulae by partitioning the integration region and combining the approximations for each partition piecewise.

Another drawback of Newton-Cotes formulae is that the integration points must be equally spaced. In particular, it may be advantageous to place more integration points over regions with a large functional variation, and fewer integration points over regions with smaller functional variation. *Adaptive methods* use information about the function to decide on the density of integration points in a particular region. Thus, over a particular interval, we increase the number of steps until a desired convergence tolerance is reached. In *Gaussian quadrature*, both the position of the integration points, as well as the weights, are determined by the method. The estimate for the integral is calculated by choosing optimal integration points for the function. For all quadrature coefficients see (Abramowitz and Stegun 1970).

The following analysis does not consider the use of interpolation. Consequently, we are excluding adaptive methods and Gaussian quadrature. The reason for this is that the error that is introduced into the early exercise premium is dominated by the European approximation over each time step for the American put option, rather than the quadrature method itself. This error is a consequence of estimating the early exercise boundary, $\widehat{b}(t_i) : i = 0, \ldots, n$. As a consequence, once the early exercise boundary has been estimated, there is little benefit in using interpolation to obtain a greater number of time steps in the integral. For completeness we have included a discussion of closed formulae in Appendix E.

### 7.3.1 Choosing a Quadrature Scheme

As shown in Appendix E, using equal spacing for the extended formulae results in better estimates in comparison to the non-equal spacing. Consequently we only need to decide between the extended Trapezoidal rule and the extended Simpson's rule.

Figure 7.5 shows that the estimate of the early exercise boundary calculated using Simpson's rule is highly oscillatory. As mentioned in Appendix E, these oscillations are regular and can be removed by considering their midpoints. The convergence results in the next section show that there is a marginal benefit in convergence speed when using Simpson's rule. Hence, there is little to choose between the two methods. Of course, the Trapezoidal rule is simpler to implement and this could be the deciding factor.

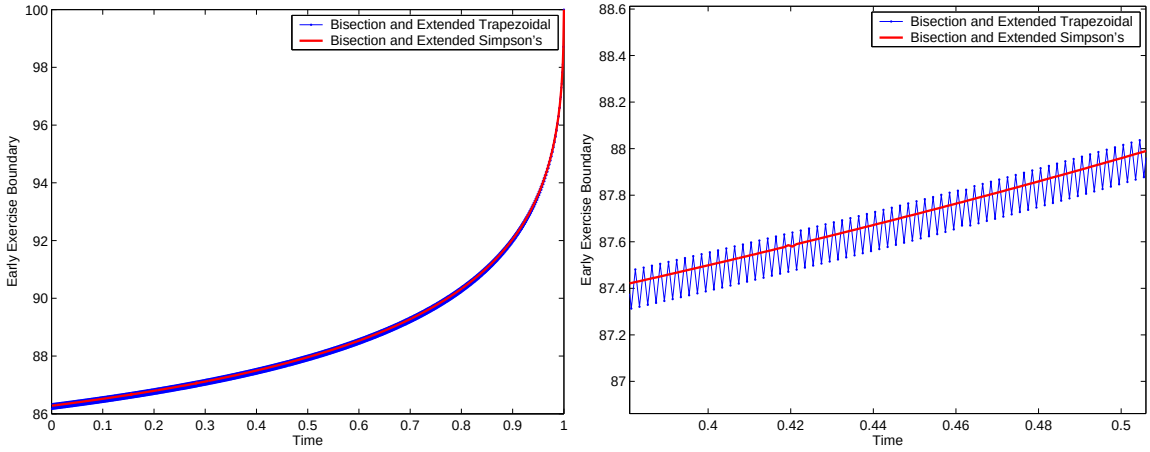


**Fig. 7.5:** The early exercise boundary estimate, $\widehat{b}$, calculated using the bisection method along with the extended Trapezoidal rule and the extended Simpson's rule.

## 7.4 Convergence Results Using the Bisection Method

We have only examined the convergence results of the bisection method because it is the preferred root finding method. The choice of quadrature method does not have a significant impact on the convergence results and here we focus on the extended Trapezoidal and extended Simpson's quadrature techniques with equal spacing. The convergence of the early exercise boundary estimates is only examined for the Trapezoidal rule. This is because of the oscillatory nature of the early exercise boundary estimate calculated using the Simpson's extended rule. We then compare the convergence results of the American put option estimate for the Trapezoidal rule and the Simpson's rule.

The first step of the algorithm is to estimate the early exercise boundary, $\widehat{b}(t_i) : i = 0, \ldots, n$. As we increase the number of time steps the early exercise boundary converges to the "true" early exercise boundary from above. We then use this estimated early exercise boundary to estimate the early exercise premium, $\widehat{e}(t, S)$. The sum of this and the equivalent corresponding European option, $p(t, S)$, results in an estimate for the American put option, $\widehat{P}(t, S)$. This estimate also converges to the true solution from above, which is a direct result of the early exercise boundary $\widehat{b}(t_i)$ converging from above. The early exercise premium is the interest earned while

the stock price is below the early exercise boundary. As a consequence, since the early exercise boundary is too high, the early exercise premium is an overestimate.

### 7.4.1 Convergence of the Early Exercise Boundary

The early exercise boundary estimate, $\widehat{b}(t_i) : i = 0, \ldots, n$, converges to the true boundary, $b(t)$, from above as the number of time steps tends towards infinity (see Figure 7.6).

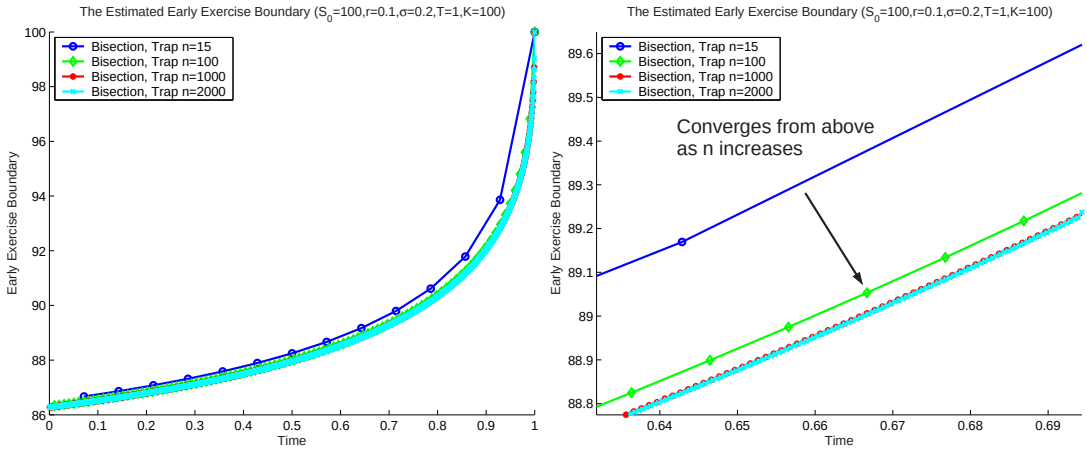


**Fig. 7.6:** The convergence behaviour of the early exercise boundary, as the number of time steps is increased. The bisection method has been used in conjunction with the extended Trapezoidal rule.

### 7.4.2 Convergence of the American Put Option Estimate

The American put option estimate, $P_{\text{EEP}}(t, S)$, calculated using the early exercise premium, converges from above as the number of time steps tends towards infinity (see Figure 7.7).

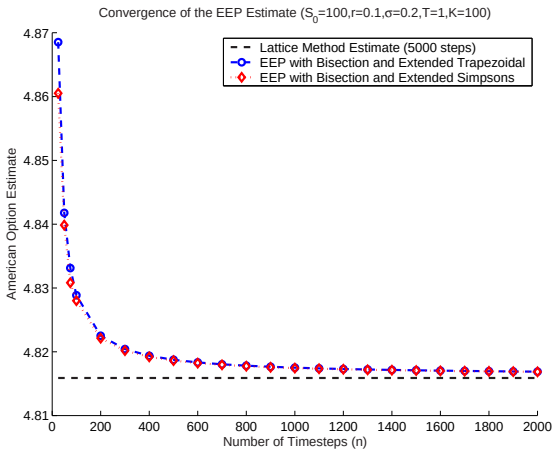


**Fig. 7.7:** The convergence behaviour of the American put option estimate using the early exercise premium, as the number of time steps is increased. The bisection method has been used in conjunction with the extended Trapezoidal rule and the extended Simpson's rule.

# Chapter 8

# Comparison of Results and Conclusions

We used the following initial stock price, $S_0 = 100$ and a constant riskless continuously compounded rate of return, $r = 10\%$. Thereafter we implemented each type of method with two volatilities, $\sigma = 20\%$ and $\sigma = 40\%$. For each table of comparisons, we varied the strike price across at-the-money by 10% and 20%, i.e. $K \in \{80, 90, 100, 110, 120\}$. Finally, we did all the calculations for a variety of maturities, $T \in \{0.25, 0.5, 1, 2\}$ years. The estimates were calculated using Matlab 6.5 on Windows XP on a Pentium 4 PC with a 2.60 GHz processor and 1 GB of memory.

## 8.1 Analytical Approximations

In Chapter 3, we examined a selection of analytical approximations that have been developed to estimate the value of the American put option. When comparing the results, we focused on the Barone-Adesi and Whaley quadratic approximation method (Section 3.4.1), the Geske-Johnson compound option formula with Richardson's extrapolation on $P_1$ and $P_2$ (Section 3.5), and the Bjerksund and Stensland approximation (Section 3.6). In each table, the values are compared to an estimate of the price calculated using a binomial tree with 5000 time steps (assuming, obviously, that this value is relatively accurate).

Tables 8.1 and 8.2 show that, under certain circumstances, the analytical approximations are relatively good. However, there is a lack of consistency in terms of which of the methods is best and under which circumstances they are best. This makes it difficult, if not impossible, to recommend any of the methods for general use. The ease and speed of use does not provide any benefit over accuracy. Consequently, using more developed numerical methods would be sensible.

## 8.2 Lattice or Tree Methods

In Chapter 4, we examined lattice or tree methods that have been developed to estimate the price of the American put option. These include modifications to the general lattice methods, which are designed to improve the convergence rate. When comparing results, we confine our attention to binomial and trinomial lattices with

5000 time steps, using the Black-Scholes modification (Section 4.5), and the adaptive mesh model (Section 4.7.1) for the trinomial tree.

Tables 8.3 and 8.4 show that, with 5000 time steps, the binomial and trinomial trees produce very similar results. However, the trinomial tree takes longer to execute. This may be as a result of using a matrix structure in Matlab, rather than a tree-type structure. Consequently, the binomial tree provides the quickest solution and should be regarded as preferable.

## 8.3 Monte Carlo Methods

In Chapter 5, we examined a selection of Monte Carlo methods that have been developed to estimate the price of the American put option. When comparing results, we used the following parameters for each method:

- Tilley's bundling algorithm (Section 5.2): number of sample paths $M = 50000$, number of time steps $n = 100$, and the number of bundles $a = 250$ ($\alpha \approx 0.5$).
- Grant, Vora and Weeks algorithm (Section 5.3): number of sample paths $M = 50000$, number of time steps $n = 100$, number of sample paths to calculate the early exercise boundary $M' = 10000$.
- Stratified Space Partitioning (Section 5.4): number of sample paths $M = 50000$, number of time steps $n = 100$, and the number of partitions $a = 200$.
- Longstaff and Schwartz regression method (Section 5.5.2): number of sample paths $M = 50000$, number of time steps $n = 100$, number of basis functions $n' = 6$. Hermite polynomials were used to construct the functional form of the continuation values. Only in-the-money sample paths were used and continuation values were used to make exercise decisions only.

Tables 8.5 and 8.6 show quite a large variation in the values that the Monte Carlo methods produce. It can be shown that increasing the number of sample stock price paths will produce less variation. A significantly larger number of sample paths will increase the runtime of the algorithms and may incur memory difficulties. Although there is no consistent criticism that can be applied across the methods, it is clear that the SSAP method tends to overestimate the value of the option. In some instances, quite dramatically. Since this is a significantly slower method, it is not the preferred solution. The Longstaff and Schwartz method is the fastest and, in contrast with Tilley's method, it can be extended to include multiple sources of uncertainty.

## 8.4 Finite Difference Methods

In Chapter 6, we examined a selection of the finite difference methods that have been developed to estimate the price of the American put option. We have used the Crandall-Douglas finite difference scheme for all methods. When comparing results, we used the following parameters for each method:

- Brennan and Schwartz method (Section 6.3): The number of spatial and time steps varied depending on the volatility. This is because of the restriction, $\alpha < 0.5$, which ensures convergence. When the volatility $\sigma = 20\%$: the number of spatial steps $m = 2000$, and the number of time steps $n = 1400$. When the volatility $\sigma = 40\%$: the number of spatial steps $m = 2000$, and the number of time steps $n = 7000$.

- The Projected Successive Over-Relaxation (PSOR) algorithm (Section 6.4): the number of spatial steps $m = 2000$, and the number of time steps $n = 250$.

- Elliot and Ockendon method (Section 6.5.2): the number of spatial steps $m = 2000$, and the number of time steps $n = 250$.

Tables 8.7 and 8.8 show that the finite difference methods give relatively similar estimates for the price of the American put option. Because of the restriction on $\alpha$, the Brennan and Schwartz method is not preferable. The PSOR and Elliot and Ockendon methods give similar results. The Elliot and Ockendon algorithm, however, is significantly faster, which would make it the preferred finite difference method.

## 8.5 The Early Exercise Premium

In Chapter 7, we decomposed the value of the American put option into a linear combination of the equivalent European put option value and an early exercise premium. We compared the two suggested quadrature schemes: the extended Trapezoidal rule and the extended Simpson's rule. The bisection method is employed, for both quadratures schemes, to locate the early exercise boundary.

Tables 8.9 and 8.10 show that estimates calculated using the early exercise premium are similar in both value and runtime. As a consequence, either quadrature scheme can be used.

## 8.6 Conclusions

Tables 8.11 and 8.12 provide a summary of all of the results discussed above. They compare the preferred method from each numerical technique, excluding the analytical approximations[1]:

- The binomial tree with 5000 timesteps and the Black-Scholes enhancement.

- The Longstaff and Schwartz regression Monte Carlo method with $M = 50000$ sample stock price paths, $n = 100$ time steps, and the first 6 Hermite polynomials. Only in-the-money sample paths were used and continuation values were used to make exercise decisions only.

- The Elliot and Ockendon algorithm using the Crandall-Douglas finite difference scheme with $m = 2000$ spatial steps and $n = 250$ time steps.

[1] Because they are so inaccurate.

- Estimation of the early exercise premium using the extended Simpson's rule as the quadrature method, and the bisection method to locate the early exercise boundary.

When deciding on which numerical technique to implement for a specific problem there are two overriding factors which should be considered: speed of convergence and flexibility. The speed of the solution is often specific to the problem considered. Some techniques are more applicable to certain problems. Techniques which are broad in application tend to be time consuming. The comparison of methods in this dissertation bear testament to this.

The (quasi-)analytical solutions give a quick but crude estimate for the price of the American put option. This price is too approximate to be useful and the methodology is unsuitable for hedging.

The finite difference methods provide the fastest estimates within a desired accuracy. One drawback of these methods is that they are not easily adapted to incorporate idiosyncratic features. They do not perform well in higher dimensions either.

The tree or lattice methods also provide a quick and accurate estimate. Although they are slower than the finite difference methods, they do have the ability to model idiosyncratic features, e.g. vesting periods. They are, however, also not easily extended to higher dimensions.

The integral representation of the early exercise premium is an interesting estimation technique. However, this particular approach is specific to the American put problem and has no obvious extension to any other derivatives. The calculation technique is extremely slow.

Not surprisingly, the slowest of the methods examined were the Monte Carlo methods. These methods are attractive because they are, in general, relatively intuitive. Monte Carlo methods have the added attraction that they are invariant in their convergence rate with respect to the number of dimensions of the problem. This means that increasing the number of stochastic factors driving the price of the derivative does not slow down the calculations. This makes them attractive for complex derivatives. However, even though they have the ability to model the early exercise feature of American-style derivatives, they are not well suited to the American put option problem, in comparison to the other methods.

In final conclusion, this dissertation suggests that the Elliot-Ockendon algorithm should be the preferred numerical method for solving the standard American put option problem.

| $S_0 = 100$<br>$r = 10\%$<br>$\sigma = 20\%$ | Method | Ave<br>CPU (s) | $T = 0.25$<br>Price | $T = 0.5$<br>Price | $T = 1$<br>Price | $T = 2$<br>Price |
|---|---|---|---|---|---|---|
| $K = 80$ | | | | | | |
| | Bar-Ade & Wha | 0.03 | 0.02373 | 0.15214 | 0.48268 | 0.97288 |
| | Geske-Johnson | 0.02 | 0.01875 | 0.12705 | 0.40691 | 0.85072 |
| | Bje & Sten | 0.01 | 0.01883 | 0.12929 | 0.42231 | 0.88202 |
| | Binomial | 30 | 0.01932 | 0.13405 | 0.43647 | 0.89791 |
| $K = 90$ | | | | | | |
| | Bar-Ade & Wha | 0.03 | 0.46161 | 1.02742 | 1.77960 | 2.55333 |
| | Geske-Johnson | 0.02 | 0.43059 | 0.95823 | 1.67904 | 2.45540 |
| | Bje & Sten | 0.01 | 0.43193 | 0.96026 | 1.67818 | 2.45289 |
| | Binomial | 30 | 0.44470 | 0.99054 | 1.71688 | 2.47936 |
| $K = 100$ | | | | | | |
| | Bar-Ade & Wha | 0.03 | 3.06755 | 3.92344 | 4.83079 | 5.65141 |
| | Geske-Johnson | 0.02 | 3.07293 | 3.94670 | 4.87336 | 5.63708 |
| | Bje & Sten | 0.01 | 3.00549 | 3.84595 | 4.75903 | 5.61275 |
| | Binomial | 30 | 3.07018 | 3.91857 | 4.81628 | 5.64344 |
| $K = 110$ | | | | | | |
| | Bar-Ade & Wha | 0.03 | 10.01811 | 10.23896 | 10.65142 | 11.11797 |
| | Geske-Johnson | 0.02 | 10.08393 | 10.35888 | 10.69454 | 10.76979 |
| | Bje & Sten | 0.01 | 10.01652 | 10.25300 | 10.68406 | 11.16913 |
| | Binomial | 30 | 10.04575 | 10.29659 | 10.71909 | 11.19463 |
| $K = 120$ | | | | | | |
| | Bar-Ade & Wha | 0.03 | 23.02968 | 21.49557 | 20.62214 | 20.20574 |
| | Geske-Johnson | 0.02 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |
| | Bje & Sten | 0.01 | 21.70227 | 20.97639 | 20.39836 | 20.08360 |
| | Binomial | 30 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |

**Tab. 8.1:** Comparison of analytical approximations with $\sigma = 20\%$. The table compares the prices for varying strike prices and maturities.

| $S_0 = 100$<br>$r = 10\%$<br>$\sigma = 40\%$ | Method | Ave<br>CPU (s) | $T = 0.25$<br>Price | $T = 0.5$<br>Price | $T = 1$<br>Price | $T = 2$<br>Price |
|---|---|---|---|---|---|---|
| $K = 80$ | | | | | | |
| | Bar-Ade & Wha | 0.03 | 0.94310 | 2.31151 | 4.41045 | 7.07488 |
| | Geske-Johnson | 0.02 | 0.91160 | 2.21860 | 4.21436 | 6.77293 |
| | Bje & Sten | 0.01 | 0.91335 | 2.22390 | 4.21688 | 6.73277 |
| | Binomial | 30 | 0.92650 | 2.26626 | 4.29963 | 6.83239 |
| $K = 90$ | | | | | | |
| | Bar-Ade & Wha | 0.03 | 2.98718 | 5.03762 | 7.66111 | 10.68234 |
| | Geske-Johnson | 0.02 | 2.93339 | 4.93494 | 7.50570 | 10.49699 |
| | Bje & Sten | 0.01 | 2.92942 | 4.91354 | 7.43733 | 10.32522 |
| | Binomial | 30 | 2.97136 | 4.99677 | 7.55743 | 10.44591 |
| $K = 100$ | | | | | | |
| | Bar-Ade & Wha | 0.03 | 6.91438 | 9.22754 | 12.02439 | 15.14502 |
| | Geske-Johnson | 0.02 | 6.89467 | 9.20675 | 12.01736 | 15.16176 |
| | Bje & Sten | 0.01 | 6.83954 | 9.09515 | 11.81074 | 14.81371 |
| | Binomial | 30 | 6.92317 | 9.21910 | 11.95858 | 14.94706 |
| $K = 110$ | | | | | | |
| | Bar-Ade & Wha | 0.03 | 12.88435 | 14.91645 | 17.50423 | 20.46673 |
| | Geske-Johnson | 0.02 | 13.01184 | 15.08729 | 17.73270 | 20.72388 |
| | Bje & Sten | 0.01 | 12.83697 | 14.82428 | 17.34654 | 20.19855 |
| | Binomial | 30 | 12.94427 | 14.96740 | 17.50368 | 20.33536 |
| $K = 120$ | | | | | | |
| | Bar-Ade & Wha | 0.03 | 20.68289 | 22.01910 | 24.07271 | 26.64601 |
| | Geske-Johnson | 0.02 | 20.96501 | 22.40189 | 24.54209 | 27.10585 |
| | Bje & Sten | 0.01 | 20.70245 | 22.00704 | 24.00384 | 26.46558 |
| | Binomial | 30 | 20.78280 | 22.13261 | 24.14868 | 26.59815 |

**Tab. 8.2:** Comparison of analytical approximations with $\sigma = 40\%$. The table compares the prices for varying strike prices and maturities.

| $S_0 = 100$<br>$r = 10\%$<br>$\sigma = 20\%$ | Method | Ave<br>CPU (s) | $T = 0.25$<br>Price | $T = 0.5$<br>Price | $T = 1$<br>Price | $T = 2$<br>Price |
|---|---|---|---|---|---|---|
| $K = 80$ | | | | | | |
| | Binomial | 30 | 0.01932 | 0.13405 | 0.43647 | 0.89791 |
| | Trinomial | 70 | 0.01933 | 0.13405 | 0.43645 | 0.89786 |
| $K = 90$ | | | | | | |
| | Binomial | 30 | 0.44470 | 0.99054 | 1.71688 | 2.47936 |
| | Trinomial | 70 | 0.44467 | 0.99047 | 1.71676 | 2.47922 |
| $K = 100$ | | | | | | |
| | Binomial | 30 | 3.07018 | 3.91857 | 4.81628 | 5.64344 |
| | Trinomial | 70 | 3.07004 | 3.91839 | 4.81604 | 5.64317 |
| $K = 110$ | | | | | | |
| | Binomial | 30 | 10.04575 | 10.29659 | 10.71909 | 11.19463 |
| | Trinomial | 70 | 10.04569 | 10.29637 | 10.71860 | 11.19404 |
| $K = 120$ | | | | | | |
| | Binomial | 30 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |
| | Trinomial | 70 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |

**Tab. 8.3:** Comparison of tree methods with $\sigma = 20\%$. The table compares the prices for varying strike prices and maturities.

| $S_0 = 100$<br>$r = 10\%$<br>$\sigma = 40\%$ | Method | Ave<br>CPU (s) | $T = 0.25$<br>Price | $T = 0.5$<br>Price | $T = 1$<br>Price | $T = 2$<br>Price |
|---|---|---|---|---|---|---|
| $K = 80$ | | | | | | |
| | Binomial | 30 | 0.92650 | 2.26626 | 4.29963 | 6.83239 |
| | Trinomial | 70 | 0.92648 | 2.26613 | 4.29937 | 6.83199 |
| $K = 90$ | | | | | | |
| | Binomial | 30 | 2.97136 | 4.99677 | 7.55743 | 10.44591 |
| | Trinomial | 70 | 2.97121 | 4.99650 | 7.55703 | 10.44538 |
| $K = 100$ | | | | | | |
| | Binomial | 30 | 6.92317 | 9.21910 | 11.95858 | 14.94706 |
| | Trinomial | 70 | 6.92291 | 9.21874 | 11.95808 | 14.94642 |
| $K = 110$ | | | | | | |
| | Binomial | 30 | 12.94427 | 14.96740 | 17.50368 | 20.33536 |
| | Trinomial | 70 | 12.94401 | 14.96701 | 17.50313 | 20.33462 |
| $K = 120$ | | | | | | |
| | Binomial | 30 | 20.78280 | 22.13261 | 24.14868 | 26.59815 |
| | Trinomial | 70 | 20.78262 | 22.13225 | 24.14810 | 26.59729 |

**Tab. 8.4:** Comparison of tree methods with $\sigma = 40\%$. The table compares the prices for varying strike prices and maturities.

| $S_0 = 100$<br>$r = 10\%$<br>$\sigma = 20\%$ | Method | Ave<br>CPU (s) | $T = 0.25$<br>Price | $T = 0.5$<br>Price | $T = 1$<br>Price | $T = 2$<br>Price |
|---|---|---|---|---|---|---|
| $K = 80$ | | | | | | |
| | Tilley's Algorithm | 20 | 0.01816 | 0.12737 | 0.40249 | 0.82381 |
| | Grant, Vora, and Weeks | 24 | 0.02001 | 0.13535 | 0.42984 | 0.90006 |
| | SSAP | 96 | 0.02297 | 0.15961 | 0.49369 | 0.96820 |
| | Longstaff-Schwartz | 17 | 0.02020 | 0.12892 | 0.44016 | 0.89342 |
| $K = 90$ | | | | | | |
| | Tilley's Algorithm | 20 | 0.43561 | 0.97127 | 1.67237 | 2.47037 |
| | Grant, Vora, and Weeks | 24 | 0.45230 | 0.97071 | 1.68724 | 2.46762 |
| | SSAP | 96 | 0.48568 | 1.02437 | 1.77044 | 2.56818 |
| | Longstaff-Schwartz | 17 | 0.43009 | 0.98318 | 1.72707 | 2.50399 |
| $K = 100$ | | | | | | |
| | Tilley's Algorithm | 20 | 3.06599 | 3.92728 | 4.81745 | 5.65319 |
| | Grant, Vora, and Weeks | 24 | 3.05122 | 3.90456 | 4.78941 | 5.58849 |
| | SSAP | 96 | 3.14663 | 3.95839 | 4.86520 | 5.64970 |
| | Longstaff-Schwartz | 17 | 3.06975 | 3.91593 | 4.80755 | 5.61290 |
| $K = 110$ | | | | | | |
| | Tilley's Algorithm | 20 | 10.08125 | 10.37458 | 10.77623 | 11.26335 |
| | Grant, Vora, and Weeks | 24 | 10.00278 | 10.26154 | 10.66579 | 11.12460 |
| | SSAP | 96 | 10.06145 | 10.32546 | 10.73167 | 11.21029 |
| | Longstaff-Schwartz | 17 | 10.03977 | 10.29696 | 10.69444 | 11.19186 |
| $K = 120$ | | | | | | |
| | Tilley's Algorithm | 20 | 20.02559 | 20.02870 | 20.02716 | 20.02664 |
| | Grant, Vora, and Weeks | 24 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |
| | SSAP | 96 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |
| | Longstaff-Schwartz | 17 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |

**Tab. 8.5:** Comparison of Monte Carlo methods with $\sigma = 20\%$. The table compares the prices for varying strike prices and maturities.

| $S_0 = 100$<br>$r = 10\%$<br>$\sigma = 40\%$ | Method | Ave CPU (s) | $T = 0.25$ Price | $T = 0.5$ Price | $T = 1$ Price | $T = 2$ Price |
|---|---|---|---|---|---|---|
| $K = 80$ | | | | | | |
| | Tilley's Algorithm | 20 | 0.92760 | 2.27764 | 4.29814 | 6.87401 |
| | Grant, Vora, and Weeks | 24 | 0.92579 | 2.25171 | 4.29104 | 6.79899 |
| | SSAP | 96 | 1.00861 | 2.34028 | 4.35260 | 6.88136 |
| | Longstaff-Schwartz | 17 | 0.90096 | 2.25177 | 4.24447 | 6.84385 |
| $K = 90$ | | | | | | |
| | Tilley's Algorithm | 20 | 2.99079 | 5.00938 | 7.58482 | 10.53429 |
| | Grant, Vora, and Weeks | 24 | 2.95383 | 4.99253 | 7.53679 | 10.40500 |
| | SSAP | 96 | 3.06677 | 5.08616 | 7.65285 | 10.60052 |
| | Longstaff-Schwartz | 17 | 2.93842 | 4.98815 | 7.57601 | 10.47224 |
| $K = 100$ | | | | | | |
| | Tilley's Algorithm | 20 | 6.95376 | 9.26785 | 12.03877 | 15.02084 |
| | Grant, Vora, and Weeks | 24 | 6.90205 | 9.18810 | 11.91771 | 14.91187 |
| | SSAP | 96 | 7.05748 | 9.33075 | 12.08429 | 15.15727 |
| | Longstaff-Schwartz | 17 | 6.94872 | 9.15635 | 11.98370 | 14.94516 |
| $K = 110$ | | | | | | |
| | Tilley's Algorithm | 20 | 12.99613 | 15.02708 | 17.55129 | 20.42396 |
| | Grant, Vora, and Weeks | 24 | 12.92895 | 14.92863 | 17.44465 | 20.24687 |
| | SSAP | 96 | 13.04329 | 15.18300 | 17.64147 | 20.31177 |
| | Longstaff-Schwartz | 17 | 13.02635 | 14.91235 | 17.28420 | 20.36189 |
| $K = 120$ | | | | | | |
| | Tilley's Algorithm | 20 | 20.83941 | 22.26088 | 24.27206 | 26.72368 |
| | Grant, Vora, and Weeks | 24 | 20.75720 | 22.10186 | 24.09454 | 26.49393 |
| | SSAP | 96 | 20.87945 | 22.23077 | 24.21324 | 26.62986 |
| | Longstaff-Schwartz | 17 | 20.79422 | 22.04040 | 24.09341 | 26.50654 |

**Tab. 8.6:** Comparison of Monte Carlo methods with $\sigma = 40\%$. The table compares the prices for varying strike prices and maturities.

| $S_0 = 100$<br>$r = 10\%$<br>$\sigma = 20\%$ | Method | Ave<br>CPU (s) | $T = 0.25$<br>Price | $T = 0.5$<br>Price | $T = 1$<br>Price | $T = 2$<br>Price |
|---|---|---|---|---|---|---|
| $K = 80$ | | | | | | |
| | Brennan-Schwartz | 27 | 0.01931 | 0.13400 | 0.43652 | 0.89799 |
| | PSOR | 38 | 0.01934 | 0.13401 | 0.43638 | 0.89782 |
| | Elliot-Ockendon | 20 | 0.01936 | 0.13408 | 0.43641 | 0.89788 |
| $K = 90$ | | | | | | |
| | Brennan-Schwartz | 27 | 0.44466 | 0.99048 | 1.71696 | 2.47940 |
| | PSOR | 38 | 0.44443 | 0.99023 | 1.71659 | 2.47920 |
| | Elliot-Ockendon | 20 | 0.44457 | 0.99042 | 1.71676 | 2.47932 |
| $K = 100$ | | | | | | |
| | Brennan-Schwartz | 27 | 3.06839 | 3.91766 | 4.81597 | 5.64330 |
| | PSOR | 38 | 3.06904 | 3.91788 | 4.81578 | 5.64325 |
| | Elliot-Ockendon | 20 | 3.06921 | 3.91795 | 4.81598 | 5.64340 |
| $K = 110$ | | | | | | |
| | Brennan-Schwartz | 27 | 10.04513 | 10.29603 | 10.71880 | 11.19440 |
| | PSOR | 38 | 10.04533 | 10.29612 | 10.71870 | 11.19430 |
| | Elliot-Ockendon | 20 | 10.04548 | 10.29618 | 10.71887 | 11.19450 |
| $K = 120$ | | | | | | |
| | Brennan-Schwartz | 27 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |
| | PSOR | 38 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |
| | Elliot-Ockendon | 20 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |

**Tab. 8.7:** Comparison of finite difference approximations with $\sigma = 20\%$. The table compares the prices for varying strike prices and maturities.

| $S_0 = 100$<br>$r = 10\%$<br>$\sigma = 40\%$ | Method | Ave<br>CPU (s) | $T = 0.25$<br>Price | $T = 0.5$<br>Price | $T = 1$<br>Price | $T = 2$<br>Price |
|---|---|---|---|---|---|---|
| $K = 80$ | | | | | | |
| | Brennan-Schwartz | 120 | 0.92656 | 2.26624 | 4.29966 | 6.83240 |
| | PSOR | 38 | 0.92635 | 2.26590 | 4.29907 | 6.83176 |
| | Elliot-Ockendon | 20 | 0.92641 | 2.26582 | 4.29913 | 6.83191 |
| $K = 90$ | | | | | | |
| | Brennan-Schwartz | 120 | 2.97070 | 4.99626 | 7.55710 | 10.44584 |
| | PSOR | 38 | 2.97087 | 4.99617 | 7.55669 | 10.44520 |
| | Elliot-Ockendon | 20 | 2.97119 | 4.99649 | 7.55700 | 10.44549 |
| $K = 100$ | | | | | | |
| | Brennan-Schwartz | 120 | 6.92159 | 9.21807 | 11.95800 | 14.94690 |
| | PSOR | 38 | 6.92228 | 9.21828 | 11.95769 | 14.94628 |
| | Elliot-Ockendon | 20 | 6.92265 | 9.21807 | 11.95772 | 14.94643 |
| $K = 110$ | | | | | | |
| | Brennan-Schwartz | 120 | 12.94372 | 14.96698 | 17.50345 | 20.33540 |
| | PSOR | 38 | 12.94363 | 14.96672 | 17.50290 | 20.33469 |
| | Elliot-Ockendon | 20 | 12.94400 | 14.96699 | 17.50317 | 20.33495 |
| $K = 120$ | | | | | | |
| | Brennan-Schwartz | 120 | 20.78186 | 22.13178 | 24.14816 | 26.59803 |
| | PSOR | 38 | 20.78244 | 22.13212 | 24.14804 | 26.59760 |
| | Elliot-Ockendon | 20 | 20.78264 | 22.13220 | 24.14819 | 26.59779 |

**Tab. 8.8:** Comparison of finite difference approximations with $\sigma = 40\%$. The table compares the prices for varying strike prices and maturities.

| $S_0 = 100$<br>$r = 10\%$<br>$\sigma = 20\%$ | Method | Ave<br>CPU (s) | $T = 0.25$<br>Price | $T = 0.5$<br>Price | $T = 1$<br>Price | $T = 2$<br>Price |
|---|---|---|---|---|---|---|
| $K = 80$ | | | | | | |
| | Ext. Trapezoidal | 79 | 0.01936 | 0.13414 | 0.43666 | 0.89838 |
| | Ext. Simpson's | 80 | 0.01936 | 0.13413 | 0.43666 | 0.89837 |
| $K = 90$ | | | | | | |
| | Ext. Trapezoidal | 79 | 0.44475 | 0.99066 | 1.71719 | 2.48014 |
| | Ext. Simpson's | 80 | 0.44474 | 0.99065 | 1.71718 | 2.48011 |
| $K = 100$ | | | | | | |
| | Ext. Trapezoidal | 79 | 3.07027 | 3.91883 | 4.81690 | 5.64482 |
| | Ext. Simpson's | 80 | 3.07026 | 3.91882 | 4.81687 | 5.64477 |
| $K = 110$ | | | | | | |
| | Ext. Trapezoidal | 79 | 10.04610 | 10.29721 | 10.72027 | 11.19687 |
| | Ext. Simpson's | 80 | 10.04608 | 10.29718 | 10.72022 | 11.19678 |
| $K = 120$ | | | | | | |
| | Ext. Trapezoidal | 79 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |
| | Ext. Simpson's | 80 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |

**Tab. 8.9:** Comparison of approximations calculated using the early exercise premium with $\sigma = 20\%$. The table compares the prices for varying strike prices and maturities.

| $S_0 = 100$<br>$r = 10\%$<br>$\sigma = 40\%$ | Method | Ave<br>CPU (s) | $T = 0.25$<br>Price | $T = 0.5$<br>Price | $T = 1$<br>Price | $T = 2$<br>Price |
|---|---|---|---|---|---|---|
| $K = 80$ | | | | | | |
| | Ext. Trapezoidal | 79 | 0.92655 | 2.26633 | 4.29986 | 6.83309 |
| | Ext. Simpson's | 80 | 0.92655 | 2.26632 | 4.29984 | 6.83305 |
| $K = 90$ | | | | | | |
| | Ext. Trapezoidal | 79 | 2.97135 | 4.99683 | 7.55774 | 10.44686 |
| | Ext. Simpson's | 80 | 2.97134 | 4.99682 | 7.55771 | 10.44680 |
| $K = 100$ | | | | | | |
| | Ext. Trapezoidal | 79 | 6.92316 | 9.21923 | 11.95905 | 14.94834 |
| | Ext. Simpson's | 80 | 6.92315 | 9.21921 | 11.95901 | 14.94826 |
| $K = 110$ | | | | | | |
| | Ext. Trapezoidal | 79 | 12.94438 | 14.96768 | 17.50439 | 20.33704 |
| | Ext. Simpson's | 80 | 12.94436 | 14.96765 | 17.50434 | 20.33695 |
| $K = 120$ | | | | | | |
| | Ext. Trapezoidal | 79 | 20.78309 | 22.13311 | 24.14969 | 26.60030 |
| | Ext. Simpson's | 80 | 20.78307 | 22.13308 | 24.14963 | 26.60019 |

**Tab. 8.10:** Comparison of approximations calculated using the early exercise premium with $\sigma = 40\%$. The table compares the prices for varying strike prices and maturities.

| $S_0 = 100$<br>$r = 10\%$<br>$\sigma = 20\%$ | Method | Ave CPU (s) | $T = 0.25$ Price | $T = 0.5$ Price | $T = 1$ Price | $T = 2$ Price |
|---|---|---|---|---|---|---|
| $K = 80$ | | | | | | |
| | Binomial | 30 | 0.01932 | 0.13405 | 0.43647 | 0.89791 |
| | Elliot-Ockendon | 20 | 0.01936 | 0.13408 | 0.43641 | 0.89788 |
| | Longstaff-Schwartz | 30 | 0.02034 | 0.13751 | 0.44198 | 0.89933 |
| | Ext. Simpson's | 80 | 0.01936 | 0.13413 | 0.43666 | 0.89837 |
| $K = 90$ | | | | | | |
| | Binomial | 30 | 0.44470 | 0.99054 | 1.71688 | 2.47936 |
| | Elliot-Ockendon | 20 | 0.44457 | 0.99042 | 1.71676 | 2.47932 |
| | Longstaff-Schwartz | 30 | 0.41551 | 1.00227 | 1.71041 | 2.46960 |
| | Ext. Simpson's | 80 | 0.44474 | 0.99065 | 1.71718 | 2.48011 |
| $K = 100$ | | | | | | |
| | Binomial | 30 | 3.07018 | 3.91857 | 4.81628 | 5.64344 |
| | Elliot-Ockendon | 20 | 3.06921 | 3.91795 | 4.81598 | 5.64340 |
| | Longstaff-Schwartz | 30 | 3.05110 | 3.75582 | 4.82682 | 5.70316 |
| | Ext. Simpson's | 80 | 3.07026 | 3.91882 | 4.81687 | 5.64477 |
| $K = 110$ | | | | | | |
| | Binomial | 30 | 10.04575 | 10.29659 | 10.71909 | 11.19463 |
| | Elliot-Ockendon | 20 | 10.04548 | 10.29618 | 10.71887 | 11.19450 |
| | Longstaff-Schwartz | 30 | 9.99125 | 10.29446 | 10.70681 | 10.74765 |
| | Ext. Simpson's | 80 | 10.04608 | 10.29718 | 10.72022 | 11.19678 |
| $K = 120$ | | | | | | |
| | Binomial | 30 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |
| | Elliot-Ockendon | 20 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |
| | Longstaff-Schwartz | 30 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |
| | Ext. Simpson's | 80 | 20.00000 | 20.00000 | 20.00000 | 20.00000 |

**Tab. 8.11:** A summary of numerical methods with $\sigma = 20\%$. The table compares the prices for varying strike prices and maturities.

| $S_0 = 100$<br>$r = 10\%$<br>$\sigma = 40\%$ | Method | Ave CPU (s) | $T = 0.25$ Price | $T = 0.5$ Price | $T = 1$ Price | $T = 2$ Price |
|---|---|---|---|---|---|---|
| $K = 80$ | | | | | | |
| | Binomial | 30 | 0.92650 | 2.26626 | 4.29963 | 6.83239 |
| | Elliot-Ockendon | 20 | 0.92641 | 2.26582 | 4.29913 | 6.83191 |
| | Longstaff-Schwartz | 30 | 0.93535 | 2.27757 | 4.27521 | 6.81552 |
| | Ext. Simpson's | 80 | 0.92655 | 2.26632 | 4.29984 | 6.83305 |
| $K = 90$ | | | | | | |
| | Binomial | 30 | 2.97136 | 4.99677 | 7.55743 | 10.44591 |
| | Elliot-Ockendon | 20 | 2.97119 | 4.99649 | 7.55700 | 10.44549 |
| | Longstaff-Schwartz | 30 | 2.98213 | 4.99134 | 7.56784 | 10.51660 |
| | Ext. Simpson's | 80 | 2.97134 | 4.99682 | 7.55771 | 10.44680 |
| $K = 100$ | | | | | | |
| | Binomial | 30 | 6.92317 | 9.21910 | 11.95858 | 14.94706 |
| | Elliot-Ockendon | 20 | 6.92265 | 9.21807 | 11.95772 | 14.94643 |
| | Longstaff-Schwartz | 30 | 6.91119 | 9.23024 | 11.96895 | 14.81716 |
| | Ext. Simpson's | 80 | 6.92315 | 9.21921 | 11.95901 | 14.94826 |
| $K = 110$ | | | | | | |
| | Binomial | 30 | 12.94427 | 14.96740 | 17.50368 | 20.33536 |
| | Elliot-Ockendon | 20 | 12.94400 | 14.96699 | 17.50317 | 20.33495 |
| | Longstaff-Schwartz | 30 | 12.93204 | 14.92766 | 17.62815 | 20.35684 |
| | Ext. Simpson's | 80 | 12.94436 | 14.96765 | 17.50434 | 20.33695 |
| $K = 120$ | | | | | | |
| | Binomial | 30 | 20.78280 | 22.13261 | 24.14868 | 26.59815 |
| | Elliot-Ockendon | 20 | 20.78264 | 22.13220 | 24.14819 | 26.59779 |
| | Longstaff-Schwartz | 30 | 20.80440 | 22.07077 | 24.14623 | 26.51720 |
| | Ext. Simpson's | 80 | 20.78307 | 22.13308 | 24.14963 | 26.60019 |

**Tab. 8.12:** A summary of numerical methods with $\sigma = 40\%$. The table compares the prices for varying strike prices and maturities.

## Appendix A

# A Parity Result for American Put and American Call options

A parity relationship for both American and European options was derived in (McDonald and Schroder 1998). As a result, the price of an American call option with strike price $K$, maturity $T$, initial stock price $S$, riskfree interest rate $r$, volatility $\sigma$, and cost of carry[1] $\delta$ is equal to the price of an American put option with strike price $S$, maturity $T$, initial stock price $K$, riskfree interest rate $r - \delta$, volatility $\sigma$, and cost of carry $-\delta$.

If the underlying asset is driven by geometric Brownian motion, the following parity result holds,

$$C(t, S, K, r, \delta, T, \sigma) = P(t, K, S, \delta, r, T, \sigma).$$

[1] The cost of carry can be obtained from the forward price relationship, $F(t, T, S) = S \exp\ \delta T\ $. For our purposes $\delta = r$.

# Appendix B

# Standard Normal Cumulative Distribution Functions

## B.1 The Univariate Standard Normal Cumulative Normal Function

The univariate standard normal cumulative normal function, $\Phi(S)$ can easily be calculated. In Matlab the following command will give $\Phi(S)$

```
0.5+erf(S/sqrt(2))/2
```

## B.2 The Bivariate Standard Normal Cumulative Normal Function

An algorithm to calculate the bivariate standard normal cumulative normal function, $\Phi_2(a, b, \rho)$, is presented in (Hull 2000). If $a \leq 0$, $b \leq 0$, $\rho \leq 0$, then

$$\Phi_2(a, b, \rho) = \sqrt{1-\rho^2}/\pi \sum_{i,j=1}^{4} A_i A_j f(B_i, B_j),$$

where

$$f(S, y) = \exp\{a'(2S - a') + b'(2y - b') + 2\rho(S - a')(y - b')\},$$
$$a' = a/\sqrt{2(1-\rho^2)}, \quad b/\sqrt{2(1-\rho^2)},$$

$$A_1 = 0.3253030, \quad A_2 = 0.4211071, \quad A_3 = 0.1334425, \quad A_4 = 0.006374323$$
$$B_1 = 0.1337764, \quad B_2 = 0.6243247, \quad B_3 = 1.3425378, \quad B_4 = 2.2626645$$

If the conditions on $a$, $b$, and $\rho$ are not satisfied the following identity can be used,

$$\Phi(a, b, \rho) = \Phi(a, 0, \rho_1) + \Phi(b, 0, \rho_2) - \delta$$

where

$$\begin{aligned}
\rho_1 &= \text{sign}(a)(\rho a - b)/\sqrt{a^2 - 2\rho ab + b^2}, \\
\rho_2 &= \text{sign}(b)(\rho b - a)/\sqrt{a^2 - 2\rho ab + b^2}, \\
\delta &= (1 - \text{sign}(a)\text{sign}(b))/4, \\
\text{sign}(S) &= \begin{cases} +1 & \text{if } S \geq 0 \\ -1 & \text{if } S < 0 \end{cases}
\end{aligned}$$

## Appendix C

# Basis Functions

### Laguerre Polynomials

$$F_0 = \mathbf{1} \tag{C.1}$$
$$F_1 = 1 - \boldsymbol{X} \tag{C.2}$$
$$F_2 = 1 - 2\boldsymbol{X} + 1/2\boldsymbol{X}^2 \tag{C.3}$$
$$F_3 = 1 - 3\boldsymbol{X} + 3/2\boldsymbol{X}^2 - \boldsymbol{X}^3/6 \tag{C.4}$$
$$F_4 = 1 - 4\boldsymbol{X} + 3\boldsymbol{X}^2 - 2/3\boldsymbol{X}^3 + \boldsymbol{X}^4/24 \tag{C.5}$$
$$F_5 = 1 - 5\boldsymbol{X} + 5\boldsymbol{X}^2 - 4/3\boldsymbol{X}^3 + 5/24\boldsymbol{X}^4 - 1/120\boldsymbol{X}^5 \tag{C.6}$$
$$F_6 = 1 - 6\boldsymbol{X} + 15/2\boldsymbol{X}^2 - 10/3\boldsymbol{X}^3 + 5/8\boldsymbol{X}^4 - 1/20\boldsymbol{X}^5 + 1/720\boldsymbol{X}^6 \tag{C.7}$$

### Hermite Polynomials

$$F_0 = \mathbf{1} \tag{C.8}$$
$$F_1 = \boldsymbol{X} \tag{C.9}$$
$$F_2 = \boldsymbol{X}^2 - 1 \tag{C.10}$$
$$F_3 = \boldsymbol{X}^3 - 3\boldsymbol{X} \tag{C.11}$$
$$F_4 = \boldsymbol{X}^4 - 6\boldsymbol{X}^2 + 3 \tag{C.12}$$
$$F_5 = \boldsymbol{X}^5 - 10\boldsymbol{X}^3 + 15\boldsymbol{X} \tag{C.13}$$
$$F_6 = \boldsymbol{X}^6 - 15\boldsymbol{X}^4 + 45\boldsymbol{X}^2 - 15 \tag{C.14}$$

### Chebyshev of First Kind Polynomials

$$F_0 = \mathbf{1} \tag{C.15}$$
$$F_1 = \boldsymbol{X} \tag{C.16}$$
$$F_2 = 2\boldsymbol{X}^2 - 1 \tag{C.17}$$
$$F_3 = 4\boldsymbol{X}^3 - 3\boldsymbol{X} \tag{C.18}$$
$$F_4 = 8\boldsymbol{X}^4 - 8\boldsymbol{X}^2 + 1 \tag{C.19}$$
$$F_5 = 16\boldsymbol{X}^5 - 20\boldsymbol{X}^3 + 5\boldsymbol{X} \tag{C.20}$$
$$F_6 = 32\boldsymbol{X}^6 - 48\boldsymbol{X}^4 + 18\boldsymbol{X}^2 - 1 \tag{C.21}$$
$$F_7 = 64\boldsymbol{X}^7 - 112\boldsymbol{X}^5 + 56\boldsymbol{X}^3 - 7\boldsymbol{X} \tag{C.22}$$
$$F_8 = 128\boldsymbol{X}^8 - 256\boldsymbol{X}^6 + 160\boldsymbol{X}^4 - 32\boldsymbol{X}^2 + 1 \tag{C.23}$$
$$F_9 = 256\boldsymbol{X}^9 - 576\boldsymbol{X}^7 + 432\boldsymbol{X}^5 - 120\boldsymbol{X}^3 + 9\boldsymbol{X} \tag{C.24}$$

### Chebyshev of Second Kind Polynomials

$$F_0 = \mathbf{1} \tag{C.25}$$
$$F_1 = 2\boldsymbol{X} \tag{C.26}$$
$$F_2 = 4\boldsymbol{X}^2 - 1 \tag{C.27}$$
$$F_3 = 8\boldsymbol{X}^3 - 4\boldsymbol{X} \tag{C.28}$$
$$F_4 = 16\boldsymbol{X}^4 - 12\boldsymbol{X}^2 + 1 \tag{C.29}$$
$$F_5 = 32\boldsymbol{X}^5 - 32\boldsymbol{X}^3 + 6\boldsymbol{X} \tag{C.30}$$
$$F_6 = 64\boldsymbol{X}^6 - 80\boldsymbol{X}^4 + 24\boldsymbol{X}^2 - 1 \tag{C.31}$$

# Appendix D

# Root Finding Algorithms

## D.1 Dekker's Algorithm

Dekker's algorithm (Bus and Dekker 1975) combines the "speed" of the secant method with the guaranteed convergence of the bisection method. The method performs secant steps (linear interpolation) unless a bisection step (halving the interval) leads to better convergence.

$\beta$ represents the best guess for the root

$\gamma$ represents the latest guess such that $f(\beta)f(\gamma) \leq 0$

$\alpha$ represents represents the previous value of $\beta$

Given points $x_0$ and $x_1$, such that

$$\begin{aligned} f(x_0)f(x_1) &\leq 0 \\ |f(x_0)| &\leq |f(x_1)|. \end{aligned} \tag{D.1}$$

Then set

$$\beta = x_0, \alpha = x_1, \gamma = \alpha. \tag{D.2}$$

At each step three values are calculated: an interpolation value, a bisection or midpoint value, and a minimal move value. The linear interpolation value $l$, is

$$l(\beta,\gamma) = \begin{cases} \beta - f(\beta)(\beta-\gamma)/(f(\beta)-f(\gamma)) & \text{if} f(\beta) \neq f(\gamma) \\ \infty & \text{if} f(\beta) = f(\gamma) \neq 0 \\ \beta & \text{if} f(\beta) = f(\gamma) = 0. \end{cases}$$

The bisection value, $m$, is

$$m(\beta,\gamma) = \frac{1}{2}\left[\beta+\gamma\right].$$

A third value, $h$, is

$$h(\beta,\gamma) = \beta + \text{sign}(\gamma-\beta) \times \delta(\beta),$$

which represents the methods smallest possible move. Here, $\delta(\beta)$ is a relative tolerance,

$$\delta(\beta) = \alpha|\beta| + \epsilon,$$

where $\epsilon$ is an absolute tolerance and $\alpha$ is a number like the floating point precision of the computer. The new iterate variable, $x_i$ is calculated using,

$$x_i = \begin{cases} l(\beta_{i-1}, \gamma_{i-1}) & \text{if } l \text{ lies between } h(\beta_{i-1}, \gamma_{i-1}) \text{ and } m(\beta_{i-1}, \gamma_{i-1}) \\ h(\beta_{i-1}, \gamma_{i-1}) & \text{if } |l - \beta_{i-1}| \leq \delta(\beta) \\ m(\beta_{i-1}, \gamma_{i-1}) & \text{otherwise.} \end{cases}$$

To keep the bracketing property, the latest values $x_j^+$ such that $f(x_j^+) > 0$ and $x_j^-$

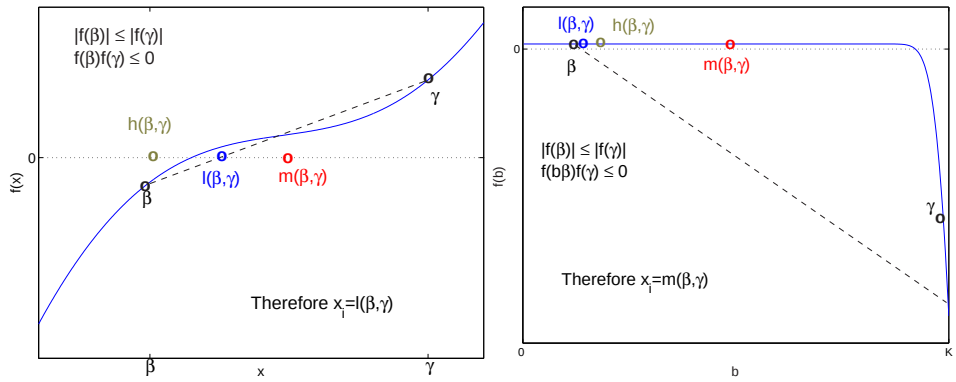


**Fig. D.1:** Dekker's method. A couple of situations showing how the new iterate point, $x_i$ is chosen.

such that $f(x_j^-) < 0$ need to be stored. Let $k$ be the largest integer values such that

$$f(x_i)f(x_k) \leq 0.$$

The values $\beta_i$, $\gamma_i$ and $\alpha_i$ are then found using,

$$\begin{aligned} \beta_i = x_i, \gamma_i = x_k, \alpha_i = \beta_{i-1} \quad & \text{if } |f(x_i)| \leq |f(x_k)| \\ \beta_i = x_k, \gamma_i = x_i, \alpha_i = x_i \quad & \text{otherwise.} \end{aligned}$$

The algorithm terminates at step $n$ when

$$|\beta_n - \gamma_n| \leq 2\delta(\beta_n).$$

The proceeding algorithms are slight modifications of Dekker's method. They employ different types of interpolation to get "improved" guesses.

## D.2 Bus-Dekker's Algorithm M

Algorithm M is very similar to Dekker's method although it uses rational interpolation instead of linear interpolation. In the worst case the rate of convergence is four times as slow as the bisection method. This is achieved by reverting to a bisection step if the interval has not been halved in four steps.

$\beta$ represents the best guess for the root

$\gamma$ represents the latest guess such that $f(\beta)f(\gamma) \leq 0$

$\alpha$ represents the previous value of $\beta$

The algorithm is setup as in step (D.1) and step (D.2). At each step a value $j = j_i$ is updated. It represents the last time that the root bracketing interval was halved,

$$|\beta_j - \gamma_j| \leq \frac{1}{2}|\beta_{j-1} - \gamma_{j-1}|.$$

The algorithm then performs normal interpolation until the interval has not been halved in three steps. It then performs a rational interpolation step. If that does not reduce the interval size sufficiently, a bisection step is performed. Each interpolation step is checked to be within the bounds as in Dekker's algorithm (here $\lambda$ represents the interpolated value),

$$w = w(\lambda, \beta, \gamma) = \begin{cases} \lambda(\beta_{i-1}, \gamma_{i-1}) & \text{if } \lambda \text{ lies between } h(\beta_{i-1}, \gamma_{i-1}) \text{ and } m(\beta_{i-1}, \gamma_{i-1}) \\ h(\beta_{i-1}, \gamma_{i-1}) & \text{if } |\lambda - \beta_{i-1}| \leq \delta(\beta) \\ m(\beta_{i-1}, \gamma_{i-1}) & \text{otherwise.} \end{cases}$$

The new iterate, $x_i$ is calculated using

$$x_i = \begin{cases} w(l(\beta_{i-1}, \gamma_{i-1}), \beta_{i-1}, \gamma_{i-1}) & \text{if } j_i \geq i - 2 \\ w(\rho(\beta_{i-1}, \gamma_{i-1}), \beta_{i-1}, \gamma_{i-1}) & \text{if } j_i = i - 3 \\ m(\beta_{i-1}, \gamma_{i-1}) & \text{otherwise,} \end{cases}$$

where $l$ is calculated as in (D.1) and $\rho$ is the rational interpolation step,

$$\rho_i = \rho(\beta_{i-1}, \alpha_{i-1}, d_{i-1}) = \begin{cases} \beta - beta(\beta - \alpha)/(\beta - \alpha) & \text{if} \beta \neq \alpha \\ \infty & \text{if} \beta = \alpha \neq 0 \\ 0 & \text{if} \beta = \alpha = 0, \end{cases}$$

where,

$$\alpha = f(\alpha)\frac{f(\beta) - f(d)}{\beta - d} \quad \beta = f(\beta)\frac{f(\alpha) - f(d)}{\alpha - d}.$$

Let $k$ be the largest integer values such that

$$f(x_i)f(x_k) \leq 0.$$

The new values for $\alpha_i$, $\beta_i$, $\gamma_i$ and $d_i$ are calculated

$$\begin{aligned} \beta_i = x_i, \gamma_i = x_k, \alpha_i = \beta_{i-1} & \quad \text{if } |f(x_i)| \leq |f(x_k)| \\ \beta_i = x_k, \gamma_i = x_i, \alpha_i = x_i & \quad \text{otherwise} \\ d_i = \alpha_{i-1} & \quad \text{if } \beta_i = x_i \text{ or } \beta_i = \beta_{i-1} \\ d_i = \beta_{i-1} & \quad \text{otherwise.} \end{aligned}$$

## D.3 Brent's Method

Brent's algorithm (Brent 1973) extends Dekker's algorithm by using inverse quadratic interpolation for the interpolation step. It also includes a few minor alterations to

improve the efficiency of the method.

| | |
|---|---|
| $\beta$ | represents the best guess for the root |
| $\gamma$ | represents the latest guess such that $f(\beta)f(\gamma) \leq 0$ |
| $\alpha$ | represents the previous value of $\beta$ |
| $d$ | represents the latest interval size |
| $e$ | represents the previous interval size |

The algorithm is again setup as in (D.1-D.2). The bisection increment, $m$

$$m(\beta, \gamma) = \frac{1}{2}(\gamma - \beta),$$

is a slight modification of Dekker's algorithm. It will return $\beta$ if $|m| \leq \delta(\beta)$ since $\beta$ is probably a better approximation to the root than $\beta + m \equiv \frac{1}{2}(\beta + \gamma)$.

The inverse quadratic interpolation is chosen over normal quadratic interpolation because the normal quadratic interpolation would yield two roots. Fitting $x$ as a quadratic in $y$ yields a distinct root. The interpolation formula (Press, Teukolsky, Vetterling, and Flannery 1999) for points $[\alpha, f(\alpha)]$,$[\beta, f(\beta)]$,$[\gamma, f(\gamma)]$ is

$$x = \frac{[y - f(\alpha)][y - f(\beta)]\gamma}{[f(\gamma) - f(\alpha)][f(\gamma) - f(\beta)]} + \frac{[y - f(\beta)][y - f(\gamma)]\alpha}{[f(\alpha) - f(\beta)][f(\alpha) - f(\gamma)]} + \frac{[y - f(\gamma)][y - f(\alpha)]\beta}{[f(\beta) - f(\gamma)][f(\beta) - f(\alpha)]}.$$

The quadratic interpolation value, $\phi(\alpha, \beta, \gamma)$ is calculated using,

$$\begin{aligned}
r_1 &= \frac{f(\alpha)}{f(\gamma)} \\
r_2 &= \frac{f(\beta)}{f(\gamma)} \\
r_3 &= \frac{f(\beta)}{f(\alpha)} \\
p(\alpha, \beta, \gamma) &= \pm r_3 \left[(\gamma - \beta)\, r_1 \,(r_1 - r_2) - (\beta - \alpha)\,(r_2 - 1)\right] \\
q(\alpha, \beta, \gamma) &= \mp (r_1 - 1)(r_2 - 1)(r_3 - 1) \\
\phi(\alpha, \beta, \gamma) &= \beta + \frac{p(\alpha, \beta, \gamma)}{q(\alpha, \beta, \gamma)}.
\end{aligned}$$

Here, $\delta(\beta)$ is a slightly different relative tolerance,

$$\delta(\beta) = 2\alpha|\beta| + \epsilon$$

where $\epsilon$ is an absolute tolerance and $\alpha$ is a number like the floating point precision of the computer.

Brent's method ensures that a bisection step will be performed every two steps, if the size of the interval has not been halved. The variable, $e$ stores the size of the interval at the previous step. The interpolation can only be a good approximation if it is single valued between $(\beta, f(\beta))$ and $(\gamma, f(\gamma))$ (Brent 1973, p. 51). Therefore an

interpolation step can only be chosen if it lies between $\beta$ and $\gamma$, up to three quarters from $\beta$ to $\gamma$. Consequently, a bisection step is chosen if,

$$\left|\frac{p}{q}\right| \geq \frac{3}{2}|m| \equiv \frac{3}{4}|\gamma - \beta|.$$

These conditions are imposed to find the new iterate $x_i$. An intermediate iterate $x_i''$ is found,

$$x_i'' = \begin{cases} x_i' & \text{if } |\beta_{i-1} - x_i'| > \delta(\beta_i) \\ \beta_{i-1} + \delta(\beta_{i-1})\text{sign}(m(\beta_{i-1}, \gamma_{i-1})) & \text{otherwise (step of } \delta). \end{cases}$$

where

$$x_i' = \begin{cases} \phi(\alpha_{i-1}, \beta_{i-1}, \gamma_{i-1}) & \text{if } \left|\frac{p}{q}\right| < \frac{3}{2}|m| \text{ (interpolation)} \\ \beta_{i-1} + m(\beta_{i-1}, \gamma_{i-1}) & \text{otherwise (bisection).} \end{cases}$$

The new iterate $x_i$ is then calculated,

$$x_i = \begin{cases} \beta_{i-1} + m(\beta_{i-1}, \gamma_{i-1}) & \text{if } \frac{p}{q} > \frac{e}{2} \\ x_i'' & \text{otherwise.} \end{cases}$$

The new values for $\alpha_i$, $\beta_i$, $\gamma_i$, $d_i$ and $e_i$ can then be found,

$$\begin{aligned} \beta_i = x_i, \gamma_i = x_k, \alpha_i = \beta_{i-1} \quad & \text{if } |f(x_i)| \leq |f(x_k)| \\ \beta_i = x_k, \gamma_i = x_i, \alpha_i = x_i \quad & \text{otherwise} \\ d_i = \beta_i - \gamma_i & \\ e_i = d_{i-1}. & \end{aligned}$$

The algorithm terminates when the size of the interval is smaller than the tolerance, or when $f(\beta_i) = 0$.

# Appendix E

# Closed Newton-Cotes Formulae

The closed Newton-Cotes formulae are derived using the Lagrange interpolating polynomials, $P_n$

$$P_n(S) = \sum_{i=0}^{n} f(S_i) \prod_{\substack{k=0 \\ k \neq i}}^{n} \frac{S - S_k}{S_i - S_k}$$

The formulae use a natural, uniform discretisation of the closed interval $[a, b] : S_i = a + ih, i = 0, \ldots, n$ where $h := \frac{b-a}{n}$. The methods are called closed because they include the interval boundary function values in the approximation. This is useful for combining estimates piecewise over an interval to form composite rules.

| **No. Points** | **Name** | **Formula** |
|---|---|---|
| 2 Points | Trapezoidal Rule | $\int_{t_0}^{t_1} f(t)dt \approx \frac{h}{2}\left[f_0 + f_1\right]$ |
| 3 Points | Simpson's 1/3 Rule | $\int_{t_0}^{t_2} f(t)dt \approx \frac{h}{3}\left[f_0 + 4f_1 + f_2\right]$ |
| 4 Points | Simpson's 3/8 Rule | $\int_{t_0}^{t_3} f(t)dt \approx \frac{3h}{8}\left[f_0 + 3f_1 + 3f_2 + f_3\right]$ |
| 5 Points | Bode's Rule | $\int_{t_0}^{t_4} f(t)dt \approx \frac{2h}{45}\left[7f_0 + 32f_1 + 12f_2 + 32f_3 + 7f_4)\right]$ |

**Tab. E.1:** Newton-Cotes Closed Formulae. Here $f_i \equiv f(t_i)$.

## E.1 Composite (Extended) Newton-Cotes Formulae

We obtain composite Newton-Cotes formulae by partitioning the integration region and applying one of the Newton-Cotes formulae from Table E.1 to each partition. The time intervals are determined *a priori*. Thus, we begin by choosing $n+1$ times, $t = t_0 < t_1 < \cdots < t_n = T$. These time steps do not have to be equally spaced, as long as we incorporate size of the time step into the composite formula. We can also combine different quadrature methods to form mixed composite formulae. However, the order of these estimates will always be equal to the worst order of the included quadrature method.

### E.1.1 Composite Formulae with Equal Spacing

We consider the estimate for the integral over the integration region, $[t, T]$. We discretise the integration region into $n$ equally spaced intervals of size $\Delta t = (T - t)/n$ with associated time steps, $t = t_0 < t_1 < \cdots < t_n = T$. The extended trapezoidal rule for $n + 1$ time steps (where there is no restriction on the size of $n + 1$) is obtained by applying the trapezoidal rule from Table E.1 to intervals $(t_0, t_1), (t_1, t_2), \ldots, (t_{n-1}, t_n)$. The extended Simpson's rule for $n + 1$ time steps is obtained in a similar fashion on intervals $(t_0, t_1, t_2), (t_2, t_3, t_4), \ldots, (t_{n-2}, t_{n-1}, t_n)$. However, the number of time steps must be strictly odd.

| No Points | Name | Formula |
|---|---|---|
| Any | Comp. Trapezoidal | $\int_{t_0}^{t_n} f(t)dt \approx \frac{\Delta t}{2}\Big[\frac{f_0}{2} + \sum_{i=1}^{n} f_i + \frac{f_N}{2}\Big]$ |
| Odd | Comp. Simpson's | $\int_{t_0}^{t_n} f(t)dt \approx \frac{\Delta t}{3}\Big[f_0 + 4\sum_{\substack{i=1 \\ i \text{ odd}}}^{N-1} f_i + 2\sum_{\substack{i=2 \\ i \text{ even}}}^{N-2} f_i + f_N\Big]$ |

**Tab. E.2:** Composite Newton-Cotes Formulae with equal spacing. Here $f_i \equiv f(t_i)$.

### E.1.2 Composite Formulae with Unequal Spacing

Equal spacing is not a necessary restriction for a composite formulae. Incorporating a time dependent step size into the formulae allows for composite formulae to be formed that use unequal spacing (See Table E.3). However, we must be careful when using an $n$-point rule when $n > 2$, because every $n$ points need to be equally spaced. However, empirically there is no benefit to using unequal spacing when estimating

| No Points | Name | Formula |
|---|---|---|
| Any | Comp. Trapezoidal | $\int_{t_0}^{t_N} f(t)dt \approx \frac{\Delta_1}{2} f_0 + \sum_{i=1}^{N-1} \frac{\Delta_i + \Delta_{i+1}}{2} f_i + \frac{\Delta_N}{2} f_N$ |
| Odd | Comp. Simpson's | $\int_{t_0}^{t_n} f(t)dt \approx \frac{1}{3}\left[\Delta_2 f_0 + \Delta_N f_N\right] + \frac{4}{3}\sum_{\substack{i=1 \\ i \text{ odd}}}^{n-1} \Delta_i f_i + \frac{1}{3}\sum_{\substack{i=2 \\ i \text{ even}}}^{n-2} \left[\Delta_i + \Delta_{i+2}\right] f_i$ |

**Tab. E.3:** Composite Newton-Cotes Formulae with unequal spacing. Here $f_i \equiv f(t_i)$ and $\Delta_i = t_i - t_{i-1}$.

the value of the early exercise premium. The error that is introduced is due to the European approximation over each time step for the American put option. As a consequence, the best approximation is calculated when the intervals are equally spaced (See Figure E.1).

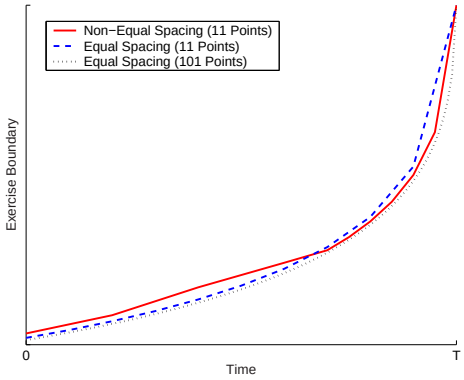


**Fig. E.1:** Non-equal spacing exercise boundary compared with an equal spacing exercise boundary using the same number of points

### E.1.3 Applying Extended formulae

When estimating the value of the early exercise boundary at the first time step we only have two points to perform the quadrature. As a consequence, any early exercise boundary estimate that we calculate must use the Trapezoidal rule at the first time step. As a result, the quadrature will have the same order of error as the Trapezoidal rule. Consequently, all the extended formulae will never have a better order of convergence than that of the Trapezoidal rule.

Another problem with using mixed formulae is that the estimated early exercise boundary oscillates (see Figure E.2). These oscillations are a result of combining odd

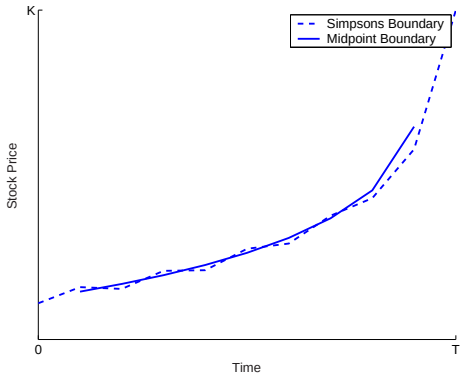


**Fig. E.2:** Exercise boundary of the extended Trapezoidal rule and the extended Simpson's rule for odd and even points showing the oscillatory behaviour.

and even quadrature rules. However, the oscillations are regular and the boundary can be "smoothed" by using the geometric midpoint of each successive midpoint (see Figure E.3).

Even though the oscillations can be smoothed once the early exercise boundary has been estimated, the major drawback is that the monotonicity is affected during the calculation. As a consequence, when calculating a boundary value, $\widehat{b}_i$, we cannot use the boundary value, $\widehat{b}_{i+1}$, as the upper bound of the bracketed root

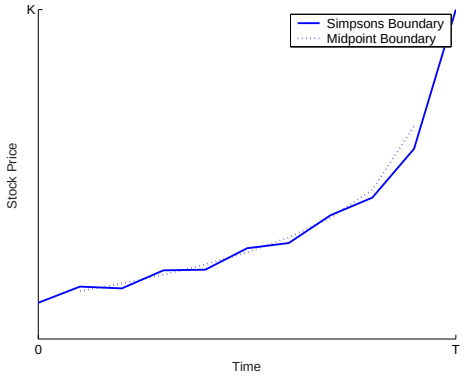


**Fig. E.3:** Using midpoints to smooth the exercise boundary generated by the extended Simpson's rule for odd and even points.

finding method. Hence, the initial bracket for the extended Simpson's rule should remain $[B_\infty, K]$ to ensure convergence. For the extended Trapezoidal rule we can use $[B_\infty, \widehat{b}_{i+1}]$, which increases the convergence speed.

# Bibliography


Abramowitz, M., and I. A. Stegun (eds.) (1970): *Handbook of Mathematical Functions*, Applied Mathematics. National Bureau of Standards, 10th edn.

Ames, W. F. (1969): *Numerical Methods for Partial Differential Equations*. Thomas Nelson and Sons.

Baase, S., and A. Van Gelder (eds.) (2000): *Computer Algorithms: Introduction to Design and Analysis*. Addison-Wesley, 3rd edn.

Barone-Adesi, G., and R. E. Whaley (1987): "Efficient Analytic Approximation of American Option Values," *The Journal of Finance*, 42(2), 301–320.

Barraquand, J., and D. Martineau (1995): "Numerical Valuation of High Dimensional Multivariate American Securities," *The Journal of Financial and Quantitative Analysis*, 30(3), 383–405.

Bjerksund, P., and G. Stensland (1993): "Closed Form Approximation of American Options," *Scandinavian Journal of Management*, 9, 87–99.

——— (2002): "Closed form valuation of American options," **http://www.nhh.no/for/seminars/previous/2002-fall/251002.pdf*.

Black, F., and M. Scholes (1973): "The Pricing of Options and Corporate Liabilities," *Journal of Political Economy*, 81(3), 637–654.

Boyle, P. (1977): "Options: a Monte Carlo Approach," *The Journal of Financial Engineering*, 4, 323–338.

——— (1986): "Option Valuation using a Three-Jump Process," *International Options Journal*, 3, 7–12.

Boyle, P., M. Broadie, and P. Glasserman (1997): "Monte Carlo methods for security pricing," *Journal of Economics and Dynamic Control*, 21, 1267–1321.

Breen, R. (1991): "The Accelerated Binomial Option Pricing Model," *The Journal of Financial and Quantitative Analysis*, 26(2), 153–164.

Brennan, M. J., and E. S. Schwartz (1977): "The Valuation of American Put Options," *The Journal of Finance*, 32(2), 449–462.

Brent, R. P. (1973): *Algorithms for Minimization without Derivatives*. Prentice-Hall, 1st edn.

Broadie, M., and J. Detemple (1996): "American option valuation: New bounds, approximations, and a comparison of existing methods," *The Review of Financial Studies*, 9(4), 1211–1250.

Bunch, D. S., and H. Johnson (1992): "A Simple and Numerically Efficient Valuation Method for American Puts Using a Modified Geske-Johnson Approach," *The Journal of Finance*, 47(2), 809–816.

Bus, J., and T. Dekker (1975): "Two Efficient Algorithms with Guarenteed Convergence for Finding a Zero of a Function," *ACM Transactions on Mathematical Software*, 1(4), 330–345.

Carr, P., R. Jarrow, and R. Myneni (1992): "Alternative characterizations of American put options," *Mathematical Finance*, 2(2), 87–106.

Carriere, J. (1996): "Valuation of the early-exercise price for otpions using simulations and nonparametric regressions," *Insurance: Mathematics and Economics*, 19, 19–30.

Chiarella, C., A. Kucera, and A. Ziogas (2004): "A Survey of the Integral Representation of American Option Prices," Research Paper Series 118, Quantitative Finance Research Centre, University of Technology, Sydney, available at http://ideas.repec.org/p/uts/rpaper/118.html.

Cleveland, W., and S. Devlin (1988): "Locally weighted regression: An approach to regression analysis by local fitting," *Journal of the American Statistical Association*, 83(403), 596–610.

Cottle, R. W., and J. Pang (1992): *The linear complementarity problem*. Academic Press, Boston.

Cox, J. C., S. A. Ross, and M. Rubinstein (1979): "Option pricing: A simplified approach," *The Journal of Financial Economics*, 7, 63–81.

Cryer, C. (1983): "The Efficient solution of linear complementarity problems for Tridiagonal Minkowski Matrices," *ACM Transactions on Mathematical Software*, 9(2), 199–214.

Davis, P. J., and P. Rabinowitz (1984): *Methods of Numerical Integration*. Academic Press, 2nd edn.

Elliot, C., and J. Ockendon (1982): *Weak and variational methods for moving boundary problems*, no. 59 in Research Notes in Mathematics. Pitman.

Elliott, R. J., and P. E. Kopp (1999): *Mathematics of Financial Markets*. Springer.

Figlewski, S., and B. Gao (1999): "The adaptive mesh model: a new approach to efficient option pricing," *The Journal of Financial Economics*, 53, 313–351.

Fu, M. C., S. B. Laprise, D. B. Madan, Y. Su, and R. Wu (2001): "Pricing American Options: A Comparison of Monte Carlo Simulation Approaches," *Journal of Computational Finance*, 4(3), 39–88.

Geske, R. (1977): "The Valuation of Corporate Liabilities as Compound Options," *The Journal of Financial and Quantitative Analysis*, 12(4).

Geske, R., and H. E. Johnson (1984): "The American Put Option Valued Analytically," *The Journal of Finance*, 39(5), 1511–1524.

Glasserman, P. (2005): *Monte Carlo Methods in Financial Engineering*. Springer.

Grant, D., G. Vora, and D. Weeks (1996): "Simulation and the Early-Exercise Option Problem," *The Journal of Financial Engineering*, 5(3), 211–227.

Harrison, J., and S. Pliska (1981): "Martingales and stochastic integrals in the theory of continuous trading," *Stochastic Process. Appl.*, 11, 215–260.

Huang, J., M. Subrahmanyam, and G. Yu (1996): "Pricing and Hedging American options: A Recursive Integration Method," *The Review of Financial Studies*, 9(1), 277–300.

Hull, J., and A. White (1988): "The use of the Control Variate technique in option pricing," *The Journal of Financial and Quantitative Analysis*, 23(3), 237–251.

——— (1993): "Efficient procedures for valuing European and American path-dependent options," *The Journal of Derivatives*, 1, 21–31.

Hull, J. C. (2000): *Options, Futures, & Other Derivatives*. Prentice-Hall International, Inc., 4th edn.

Jacka, S. (1991): "Optimal Stopping and the American Put," *Mathematical Finance*, 1(2), 1–14.

Jaillet, P., D. Lamberton, and B. Lapeyre (1990): "Variational inequalities and the pricing of American options," *Acta Applicandae Mathematicae*, 21, 263–289.

Johnson, H. (1983): "An Analytic Approximation for the American Put Price," *The Journal of Financial and Quantitative Analysis*, 18(1), 141–148.

Johnson, L. W., and R. D. Riess (1982): *Numerical Analysis*. Addison-Wesley Publishing Company, 2nd edn.

Karatzas, I. (1988): "On the Pricing of American Options," *Applied Mathematics and Optimization*, 17, 37–60.

Karatzas, I., and S. Shreve (1988): *Brownian Motion and Stochastic Calculus*. Springer-Verlag.

Kim, I. J. (1990): "The analytic valuation of American options," *The Review of Financial Studies*, 3(4), 547–572.

Leisen, D., and M. Reimer (1996): "Binomial Models for Option Pricing - Examining and Improving Convergence," *Applied Mathematical Finance (CHECK)*, 3(3), 319–346.

Longstaff, F. A., and E. S. Schwartz (2001): "Valuing American Options by Simulation: A Simple Least-Squares Approach," *The Review of Financial Studies*, 14(1), 113–147.

MacMillan, L. (1986): "Analytic Approximation for the American put option," *Advances in Futures and Options Research*, 1, 119–139.

Margrabe, W. (1978): "The Value of an option to exchange one asset for another," *The Journal of Finance*, 33(1), 177–186.

McCartin, B. J., and S. M. Labadie (2003): "Accurate and efficient pricing of vanilla stock options via the Crandall-Douglas scheme," *Applied Mathematics and Computation*.

McDonald, R. L., and M. D. Schroder (1998): "A parity result for American options," *Journal of Computational Finance*, 1(3), 5–13.

McKean, H. (1965): "Appendix A: A free boundary problem for the heat equation arising from a problem in mathematical economics," *Industrial Management Review*, 6, 32–39.

Merton, R. C. (1973): "Theory of Rational Option Pricing," *Bell Journal of Economics and Mangement Science*, 4, 141–183.

Morton, K. W., and D. Mayers (1994): *Numerical Solution of Partial Differential Equations*. Cambridge University Press, 2nd edn.

Musiela, M., and M. Rutkowski (1998): *Martingale Methods in Financial Modelling*. Springer, 2 edn.

Neveu, J. (1975): *Discrete-Parameter Martingales*. North Holland Publishing Company.

Omberg, E. (1987): "A Note on the Convergence of Binomial-Pricing and Compound-Option Models," *The Journal of Finance*, 42(2), 463–469.

Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (1999): *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 2nd edn.

Rendleman, R., and B. Bartter (1979): "Two-State Option Pricing," *The Journal of Finance*, 34(5), 1093–1110.

Tilley, J. A. (1993a): "Valuing American options in a path simulation model," *Transactions of the Society of Actuaries*, 45, 93–104.

——— (1993b): "Valuing American options in a path simulation model and discussion," *Transactions of the Society of Actuaries*, 45, 499–549.

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

West, G. (2005): "Better Approximations to Cumulative Normal Functions," *Wilmott Magazine*, pp. 70–76.

Wilmott, P. (2000): *Paul Wilmott on Quantitative Finance*, vol. 2. John Wiley and Sons, Ltd.

Wilmott, P., S. Howison, and J. Dewynne (1995): *The Mathematics of Financial Derivatives*. Cambridge University Press.