THE OPTIMAL EXERCISING PROBLEM FROM AMERICAN OPTIONS: A COMPARISON OF SOLUTION METHODS

by

SARA DEHAVEN

B.S., Kansas State University, 2007

A THESIS

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Industrial and Manufacturing Systems Engineering
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2007

Approved by:

Major Professor
Dr. Chih-Hang Wu

Abstract

The fast advancement in computer technologies in the recent years has made the use of simulation to estimate stock/equity performances and pricing possible; however, determining the optimal exercise time and prices of American options using Monte-Carlo simulation is still a computationally challenging task due to the involved computer memory and computational complexity requirements. At each time step, the investor must decide whether to exercise the option to get the immediate payoff, or hold on to the option until a later time.

Traditionally, the stock options are simulated using Monte-Carlo methods and all stock prices along the path are stored, and then the optimal exercise time is determined starting at the final time period and continuing backward in time. Also, as the number of paths simulated increases, the number of simultaneous equations that need to be solved at each time step grow proportionally. Currently, two theoretical methods have emerged in determining the optimal exercise problem. The first method uses the concept of least-squares approach in linear regression to estimate the value of continuing to hold on to the option via a set of randomly generated future stock prices. Then, the value of continuing can be compared to the payoff at current time from exercising the option and a decision can be reached, which gives the investor a higher value. The second method uses the finite difference approach to establish an exercise boundary for the American option via an artificially generated mesh on both possible stock prices and decision times. Then, the stock price is simulated and the method checks to see if it is inside the exercise boundary.

In this research, these two solution approaches are evaluated and compared using discrete event simulation. This allows complex methods to be simulated with minimal coding efforts. Finally, the results from each method are compared. Although a more conservative method cannot be determined, the least-squares method is faster, more concise, easier to implement, and requires less memory than the mesh method.

The motivation for this research stems from interest in simulating and evaluating complicated solution methods to the optimal exercise problem, yet requiring little programming effort to produce accurate and efficient estimation results.

Table of Contents

List of Figures

List of Tables

CHAPTER 1 - INTRODUCTION

From the earliest conception of stocks and bonds, a person wished to invest in a company in order to return a profit. This basic concept has transformed into an entire financial industry, including trading companies, financial advisors, and corporate investment corporations, in addition to personal investors.

In addition to simply buying and selling stocks, investors can also buy an option on the futures market. An option gives the owner the right to buy or sell a set of shares of stocks at a prescribed price before or at an expiration time into the future. This prescribed price is called the strike price, and the time in the future is called the expiration date. A put option gives the owner the right to sell the shares, while a call option gives the owner the right to buy the shares. Although each of these option types gives the owner a right to exercise, the owner can choose not to exercise the option. Therefore, an owner's choices for exercising the option are as follows (Ross, 2003, McLeish, 2005, Hull, 2006, and others):

- Put Option: Max { Strike price – Market price, 0 }
- Call Option: Max { Market price – Strike price, 0 }

Each investor is seeking a larger gain with minimal risk; therefore, the areas of mathematics, statistics, finance, and even engineering have become intricately involved with attempting to predict the stock market. In particular, financial engineering centers on using mathematically models of stock prices to simulate the unknown future stock prices.

1.1 - Background on Pricing Options

Before any stock option can be examined, the underlying assumptions to the model must be explored. The most basic assumption for any stock is that it will either increase or decrease by an amount during a single time period. The current stock price can therefore be represented as a function of the previous period's price. In this section, the movement of the stocks is discussed in relation to the Brownian motion distribution. Next, the American option type is explained, along with methods to solve the optimal exercising problem.

1.1.1 - Brownian Motion

Although early researchers first thought that stock prices followed Brownian motion, it is now known that Brownian motion represents the percent change in stock price. The stock price can be modeled by a geometric Brownian motion distribution (Ross, 2003). For a given stochastic process $\{X(t), t \geq 0\}$ to be a Brownian motion process, the following three conditions must be met:

(i) $X(0) = 0$

(ii) $\{X(t), t \geq 0\}$ has stationary and independent increments

(iii) For every $t > 0$, $X(t)$ is normally distributed with mean 0 and variance $\sigma^2 t$.

With the variance dependent on the volatility of the stock price σ and the time t, Brownian motion is a modified normal distribution. If the stock volatility σ is equal to 1, then the process is said to behave as a standard Brownian motion with mean 0 and variance t.

For every Δt the stock price will change by a step size Δx. This step size can also be thought of as the percent change in price, either up or down. An assumption of Brownian motion is that $\Delta x = \sigma\sqrt{\Delta t}$, which prevents the stock price $X(t) = 0$ with probability 1 (Ross, 2003). As a result, it can be shown that as $\Delta t \to 0$, the expected value of the stock price $E[X(t)] = 0$ and the variance of the stock price $\text{Var}(X(t)) \to \sigma^2 t$ (Ross, 2003). Therefore, the longer the time period the stock prices are studied, the larger the variance in the stock price.

Black and Scholes (1973) and Merton (1973) were the first to present mathematical models to represent the stock market. From their work, a stochastic differential equation was used to give the price of an underlying stock, given below:

$$\frac{dX(t)}{X(t)} = rdt + \sigma W(t).$$

The stock price is a function of the interest free rate r, the time t, the stock volatility σ, and W following the standard Brownian motion. The solution to this stochastic differential equation is:

$$X(T) = X_0 \exp([r - \tfrac{1}{2}\sigma^2]T + \sigma W(T)).$$

Therefore, the stochastic differential equation to describe the price of a stock and its solution both use the Brownian motion distribution parameter.

1.1.2 - American Options

As previously stated, options give the owner a right to buy or sell shares of stock at a future date. The difference between the option types is how they calculate the strike prices and when the owner is allowed to exercise the option. For example, the simplest option type is the European options since the strike price is equal to the prescribed price per share at the expiration date and the option can only be exercised on the date specified, not before and not after. Another option type that also allows a single expiration time is the Asian options, but for this option type, the strike price is equal to the average market price over the period ending at the expiration date.

There are several other types of options, but this research focuses on the American options. For the American options, the strike price is equal to the prescribed price, but the owner has the right to exercise the option at any point up until the expiration date. This continuous exercise period makes it difficult to compute the optimal exercise time for American options. The owner has to choose between taking the gain at the current time period and continuing to hold the option for a future gain.

To solve the optimal exercising problem, some algorithms determine the exercise time by 'seeing' into the future, while others hit a predetermined termination level and then exercise. Dynamic programming [Broadie, et al 2000, Tsitsiklis and Van Roy 2001, Haugh 2004] and least-squares [Longstaff and Schwartz 2001, Glasserman and Yu 2004, Chan, et al 2004] methods look into the future to solve the optimal exercising problem. The stock price paths are first stimulated for the entire exercise life of the stock option. Starting at the final time period, both methods work backwards to determine when to exercise the option. Dynamic programming uses a recursive equation to solve for the maximum value that can be obtained at each time step. The value at each time step is a function of exercising at the current time period and holding the option until a future date. Since the method starts at the last period and works backwards, the optimal value of holding the option until a future date is known from the analysis of the previous periods.

Similarly, the least-squares method starts the analysis at the final time period. In this method, the value of continuing to hold the option past the current time period is determined by the least-squares technique from linear regression. All paths that would return a profit in the current time period are regressed to form a single function that gives the value of continuing to the next period. Then, each path that is in the money is evaluated using the function to determine

if the option should be exercised at the current time period or held to a future date. Finally, the algorithm moves to the next previous time period and continues the analysis.

Contrasting dynamic programming and least-squares methods, the methods of stochastic meshes including the finite difference, and finite element algorithms solve the optimal exercising problem by hitting a predetermined termination level and then exercising. First, the available region is divided into a grid, in which each point corresponds to a time period and a stock price. After the grid is completed, the optimal time to exercise at each time period can be computed. The method of computing this exercise boundary differs between each algorithm. Finally, the stock prices are simulated and once the exercise region is hit, the option is exercised.

In the stochastic mesh methods [Broadie, et al. 2000, Broadie and Glasserman 2004, Achdou and Pironneau 2005], a transition density function or constrained optimization problem is used to determine the exercise boundary. For finite difference methods [Brennan and Schwartz 1977, Shu et al. 2005, Achdou and Pironneau 2005], a grid is also used, but the algorithm discretizes the stochastic differential equation. Then the exercise boundary can be computed by solving the resulting simultaneous equations. Conversely, the finite element method [Courant 1943, Allegretto, et al. 2003, Achdou and Pironneau 2005, Zhang 2005] approximates the solution to the differential equation, instead of the actual differential equation like finite difference method. Therefore, the partial differential equation is either eliminated completely or modified into an equivalent ordinary differential equation. Again, the exercise boundary is then computed and lastly the stock prices can be simulated.

1.2 - Background on Variance Reduction Techniques

As with most simulations, the accuracy of Monte Carlo simulations is dependent on the number of trials or replications that are run. When reporting the outputs of the simulations, the sample mean and the standard deviation for the discounted payoffs are general of interest. The simulation's estimate of the value of the option is given by the mean, while the standard error of the estimate is as follows:

$$\text{Sample Standard Deviation} / (\text{number of paths})^{1/2} .$$

Therefore, the uncertainty regarding the value of an option is inversely proportional to the square root of the number of replications. In order to double the accuracy of the simulation, the number of paths must be quadrupled. As an alternative to simply increasing the number of paths to

increase the accuracy, variance reduction techniques can be employed to help increase the accuracy using the same number of replications. Therefore, variance reductions techniques also help reduce computational time, compared to simply running more replications. In this section two variance reduction techniques are discussed. Antithetic variate approach artificially induces a correlation between the replications, while the control variate method uses an indirect measurement to help eliminate any variability due to the simulation.

1.2.1 - Antithetic Variate

The antithetic variate technique should reduces the variance by artificially inducing a correlation between replication of the model. It makes *n* independent pairs of correlated replications of the same system. The first replication has a smaller than expected observations, while the second replication offsets the first by having a larger than expected observation, or vice versa. Let X_1 and X_2 be the observations from two consecutive replications of the same simulation, and Y be defined as $(X_1+X_2)/2$. The expected values for both X_1 and X_2 are the same as the expected values of the base system, since X_1 and X_2 are observations from the base system. From the definition of the antithetic variate, the expected value and variance of Y are as follows:

$$E(Y) = \frac{E(X_1) + E(X_2)}{2} = E(X_1) = E(X_2) \tag{1.1}$$

$$Var(Y) = \frac{Var(X_1) + Var(X_2) + 2Cov(X_1, X_2)}{4}. \tag{1.2}$$

When the two replications are averaged together they should produce a lower variance, but same expected value. The expected value of the antithetic variate is the same as the two original observations, while the variance is different. If the two original observations have equal variances, $Var(X)$, then Eq. (1.2) can be rewritten as:

$$\begin{aligned} Var(Y) &= \frac{Var(X_1) + Var(X_2) + 2Cov(X_1, X_2)}{4} \\ &= \frac{2Var(X) + 2Cov(X_1, X_2)}{4} \\ 2Var(Y) &= Var(X) + Cov(X_1, X_2). \end{aligned}$$

Therefore, to reduce the variance in the system the following must hold:

$$2Var(Y) < Var(X) + Cov(X_1, X_2)$$

The above relationship demonstrates that the covariance between the two original observations can be positive or negative; however, a negative covariance will always reduce the variance in the system. Hence, if the simulation can guarantee a negative covariance between the two original observations, the antithetic variate technique will reduce the variance in the system.

With the implementation of antithetic variate technique given above, the first replication has a smaller than expected observations while the second has a larger observation, or vice versa. Thus a negative covariance will always exist between the two observations. Since the covariance is forced to be negative, the average has a smaller variance than either of the original observation's variance.

The most common way to create a negative covariance is using a uniform distribution. If the j^{th} random number of the first replication in the pair is denoted by r_j, then is uniformly distributed between 0 and 1. Then, the second replication's random number r_j' is the complement of the first or r_j'$=1-r_j$. If a uniform distribution was not desired, the uniform distribution parameters can then be transformed into the desired distribution.

1.2.2 - Control Variate

Another variance reduction technique frequently used to increase the accuracy of the price of a more complex option is the control variate technique. It assumes there is a relationship between the control variate C and the observation X, but an exact mathematical relationship is necessary. The relationship is estimated using the simulation and then the observed values of X are adjusted by this estimated relationship. Although the exact relationship is unknown, the relationship is generally considered linear. The adjusted variate (X_C) using m control variates C_l for $l = 1, \ldots, m$ is defined as the following:

$$X_C = X - \sum_{l=1}^{m} a_l (C_l - E(C)) \qquad (1.3)$$

where

$$a_l = \frac{Cov(X, C)}{Var(C)}.$$

X_C is the adjusted observation, and a_l is the weighting factor for each of the control variates. The general form allows for multiple control variates, precisely m of them. With a single control variate, the corrected observation and reduced variance is shown in the following equations:

$$E(X_C) = E(X) - aE(C - E(C)) = E(X) \tag{1.4}$$

$$Var(X_C) = Var(X) + a^2 Var(C) - 2aCov(X, C). \tag{1.5}$$

To find the minimum variance $Var(X_C)$, the first derivate of Eq. (1.5) is set to zero,

$$\frac{dVar(X_C)}{da} = 2aVar(C) - 2Cov(X, C) = 0.$$

Hence, the value of a that minimizes $Var(X_C)$ can be obtained by,

$$a = \frac{Cov(X, C)}{Var(C)}.$$

As stated, the expected values are the same, but the variance can be reduced because of the selection of a. The formula for a given in Eq. (1.3) can be calculated in the similar fashion that minimizes the variance for multiple control variates. Therefore, by selecting a from (1.3) the variance of X_C is automatically reduced and the user does not even need to predetermine if the covariance between the control variate and the original observation is positive or negative.

To implement control variate technique, Hull (2006) suggests the value of the complex option – such as Asian or American options – and the European option are simulated, and the Black-Scholes price of the European option is calculated. The errors resulting from simulating more complicated options and the European option are assumed to be similar and positively correlated. This gives an estimate of the prices (f) of the American option as the following expression:

$$f_C = f_A - a(f_E - f_{BS}). \tag{1.6}$$

Therefore, the estimate of the price of the complicated option as the simulated price plus the Black-Scholes European price minus the simulated European price. The simulated European value is supposed to be the same as the calculated Black-Scholes European option; therefore, subtracting one while adding the other should result no change of the American option value. Also, the error from both simulated values is assumed to be the same, so subtracting the European option removes this unnecessary error from the simulated American option.

1.3 - Contributions and Motivations

There are several techniques used to price American options, including stochastic meshes [Broadie, et al. 2000, Broadie and Glasserman 2004, Achdou and Pironneau 2005], finite difference [Brennan and Schwartz 1977, Shu et al. 2005, Achdou and Pironneau 2005], finite

element [Courant 1943, Allegretto, et al. 2003, Achdou and Pironneau 2005, Zhang 2005], dynamic programming [Broadie, et al 2000, Tsitsiklis and Van Roy 2001, Haugh 2004], and least-squares [Longstaff and Schwartz 2001, Glasserman and Yu 2004, Chan, et al 2004] approaches. From these different methods, finite difference and least-squares approaches have become the two most popular ways to estimate the price of American options. The motivation for this research stems from an interest in simulating and evaluating complicated solution methods to the optimal exercise problem for the American options, with modest programming effort to produce accurate and efficient estimation results.

One of the very first methods to price American options comes from Brennan and Schwartz (1977), who present an algorithm that discretizes the Black-Scholes model using finite differences. Since the 1970s, there have been several modifications and adaptations to the original method of finite differences. Adapting this approach by using more efficient methods to solve the simultaneous difference equations is presented in Achdou and Pironneau (2005) and the proposed algorithm is called the Front-Tracking Algorithm.

In contrast, Longstaff and Schwartz (2001) proposed a method of estimating the value of continuing using least-squares from linear regression. Since this method involves recursively evaluating each time step starting with the final, each path has to be generated forwards and stored in the computer memory for the algorithm to work. A variation of the least-squares method was proposed by Chan, et al. (2004) in which the paths do not have to be stored, but can be regenerated backwards as the algorithm executes to greatly reduce the memory requirements for the least-squares approaches. The proposed algorithm for this research contains a slight modification to Chan, et al. (2004). The adaptation involves using the next time period t+1 cash flow values discounted back one period for the least-squares regression computations, instead of the maximum cash flow from all future time periods as presented in Chan, et al. (2004).

Although the two methods mentioned above have been presented with numerical computations, a comparison between the methods has not been preformed. The first step in comparing of these two models is to implement the two algorithms into computer programs. Traditionally, the methods to simulate stock prices involved general purpose programming languages, such as Fortran or C++, or mathematical software, such as MATLAB. In this research, a discrete-event simulation program, Arena, is used to create the two programs. The advantage of using a simulation software package, such as Arena, is the simple programming

effort is needed to create complex programs. Therefore, American options and the solutions to the optimal exercising problem can be programmed more easily and more efficiently with high level programming languages.

In addition to creating the complex programs, the primary objective of this research entails a formal comparison of the least-squares and finite difference method programs. Once both programs are simulated, the computational results from the two approaches are compared in terms of ease of implementation, speed with which the results can be obtained, present value for the option returns, and the efficiency of memory storage requirements.

1.4 - Outline

The rest of this thesis progresses as follows. Chapter 2 is a literature review of some current and past research labors in financial engineering, American options, and solution method to the optimal exercise problem. Next, Chapter 3 presents the least-squares algorithm is presented, along with the simulation model to implement the method. The basic algorithm as presented by Longstaff and Schwartz (2001) is discussed, and the backward path generation adaptation as introduced by Chan, et al. (2004) is also presetned. Finally, the section includes computational results from different input sets.

The finite difference method is presented in Chapter 4. After presenting the formal algorithm, the method of simulating the stochastic stock prices is explained. The Front-Tracking Algorithm from Achdou and Pironneau (2005) is used to calculate the exercise boundaries. The different stochastic stock prices are generated using the simulation model, in which their exercise times were determined based on the pre-calculated boundaries. The average returns from the finite difference approach are then presented. Finally, a comparison of average returns for the least-squares model and the finite difference method is presented in Chapter 5. The methods are compared on computation results, observations during implementation, and speed and efficiency of memory storage.

CHAPTER 2 - LITERATURE REVIEW

This chapter highlights some of the past and current research efforts for simulations and financial engineering in stock options. Section 2.1 is an overview of pricing options and how simulations can contribute. Section 2.2 examines the American call option, while Section 2.3 describes the American put option. Section 2.4 analyzes the literature related to stochastic meshes, finite difference methods, and finite element analysis. Finally, Section 2.5 presents the least-squares method.

2.1 - Simulation for Pricing Options

This section discusses the pricing of options, when simulation techniques are appropriate and when they are unnecessary. In the early 1970s, Black and Scholes (1973) conducted fundamental work regarding the mathematics behind pricing options. Merton (1973) expanded their work and coined the phrase 'Black-Scholes' to represent the model of the market for an equity in which the equity's price is a stochastic process. According to this model, the price of the underlying stock is given by the following stochastic differential equation:

$$\frac{dS(t)}{S(t)} = rdt + \sigma W(t), \tag{2.1}$$

with r as the drift parameter, σ as the stock volatility, and W following a standard Brownian motion. The solution to Eq. (2.1) is given by:

$$S(T) = S_0 \exp([r - \tfrac{1}{2}\sigma^2]T + \sigma W(T)), \tag{2.2}$$

with S_0 as the initial stock price. Since standard normal distributions are more common than Brownian motion, the Black-Scholes solution can be rewritten to transform the distribution as follows:

$$S(T) = S_0 \exp([r - \tfrac{1}{2}\sigma^2]T + \sigma\sqrt{T}Z), \tag{2.3}$$

where Z is taken from the Standard Normal distribution with mean 0 and variance 1. The stock price can also be computed using the previous time period's price, instead of a function of the initial stock price. The following equation gives the stock price dependent on the previous period:

$$S(t+1) = S_t \exp([r - \tfrac{1}{2}\sigma^2]\Delta t + \sigma\sqrt{\Delta t}Z)\,. \qquad (2.4)$$

Using the Black-Scholes model, the option price can be computed if the stock is traded.

Using the Black-Scholes model, plain or vanilla options can be directly calculated. This holds particularly true for European and Asian options. These options generally do not have path dependent exercise prices, but have well-defined properties. Once path-dependent options or many stochastic variables are introduced, the options cannot be calculated using exact formulas. This occurs when techniques such as binomial trees, Monte Carlo simulations, and finite difference methods are used.

Although Monte Carlo simulations are simply based off random number generation, it has many practicalities in statistical, mathematical, and physical fields. Tilley (1993) was the first to propose a solution of using Monte Carlo simulations for pricing American options. Jackel (2002), Glasserman (2003), and Hull (2006) suggest using simulations when exact formulas are not available. Since American options have the added complexity of continuous exercise period, simulations can be used to generate the multiple exercise period. Simulation also allows for experimenting or "what if" questions, in addition to investigating different parameter values without physically buying the option. European, and other simple type options, do not require complex Monte Carlo simulations to determine if the option will be exercised, since they can be implicitly calculated using the Black-Scholes formula.

2.2 - American Call Non-existent

With any style of option, the investor has a choice between a call and a put. Hull (2006) determined that an American call option will never be exercised early. He presents this argument with the following example.

- American call option, non-dividend paying stock
- Expiration time: 1 month
- Current stock price: $50
- Strike price: $40

Since the option is currently in the money, the investor might consider exercising the option immediately. If the option is immediately exercised, then the investor can either hold the stock until after the expiration date or sell it immediately. The investor will lose the interest on the $40 if the stock is held until after the expiration date. The second option is to sell the stock directly,

but it is actually better if the investor sells the option instead of the stock. Other investors do not wish to hold the stock – otherwise the current stock price would not be at \$50 –therefore, the investor can sell the option for more than the intrinsic value of \$10.

To formalize the argument, consider the following parameters:

$S_0 \equiv$ Current stock price

$K \equiv$ Strike price

$T \equiv$ Time to expiration

$r \equiv$ Risk-free interest rate

$C \equiv$ Value of the American call option

$c \equiv$ Value of the European call option

Since the investor has a right not to exercise the option, the following must hold

$$c \geq \max\{S_0 - Ke^{-rT}, 0\}.$$

Because an investor holding an American option has the right to exercise over a continuous time period, the American call option value is required to be greater than the European call option, or $C \geq c$. Therefore,

$$C \geq S_0 - Ke^{-rT}.$$

Since $r > 0$, then $C > S_0 - K$ must also be true. If it was advantageous for the investor to exercise the option early, then $C = S_0 - K$; therefore Hull shows that it will never be profitable for an investor to exercise an American option early. With Hull's conclusion, the American call option becomes of little interest to investors, so there is no reason for practical simulation models to assess the American call options.

2.3 - American Put Early Exercise

In contrast to Section 2.2 regarding American call options, it can be optimal for American put options to be exercised early. Hull (2006) states that once an American put option is in the money it should always be exercised early. He suggests that not exercising the option early once the option is in the money results in one of two outcomes: less gain or less value. If the stock price rises, then the investor will have a lower gain once the option is exercised since the payoff is the strike price minus the current market value. Conversely, if the current market price

remains the same or slightly lower, then considering the time value of money it is better for the investor to have the value in today's dollars instead of the same value in future dollars.

Consider the same parameter definitions as Section 2.2 and the additional:

$P \equiv$ Value of the American put option

$p \equiv$ Value of the European put option

The optional exercise of an American put option creates the following

$$p \geq \max\{Ke^{-rT} - S_0, 0\}.$$

An American put option is worth more than a European put option, because there are times when the put option can be exercised for a profit. Since early exercise is a possibility, the stronger condition below must also hold

$$P \geq K - S_0.$$

While $r > 0$, an American put option can be equal to its intrinsic value $K - S_0$; therefore, there exist times when a European put option is worth less than its intrinsic value. Therefore, it is beneficial for the investor to exercise the American put option once it is in the money.

The rule of immediately exercising the American put option when it is in the money is utilized in Glasserman (2003), Achdou and Pironneau (2005), and others. The point at which the option becomes in the money is called the exercise boundary. An American put option price, given by the red line, and the corresponding exercise boundary is illustrated below.

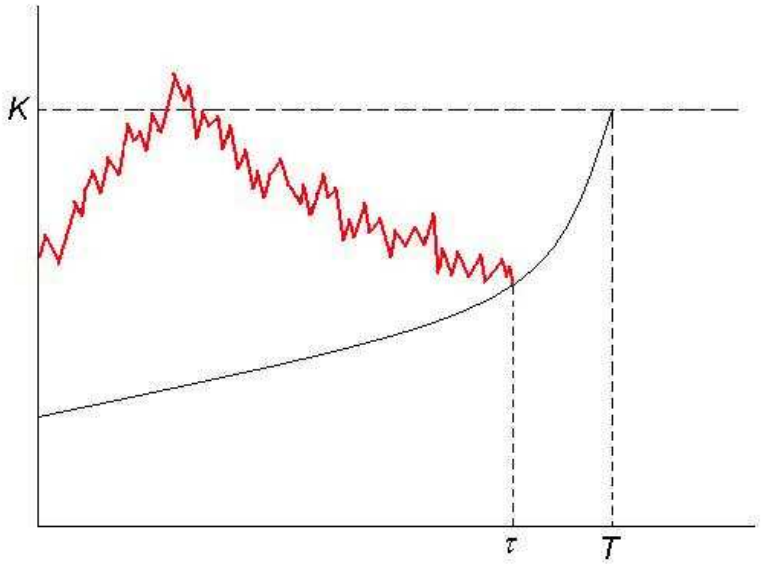

Figure 2.1 - Exercise Boundary for American Put Options

In Figure 2.1, the American put option would be exercised at τ since it is the first time the price of the underlying asset hits the exercise boundary. The area under the curve represents the exercise region, or where the option is in the money. The upward curvature of the exercise boundary in Figure 2.1 is typical of American put options; therefore, the exercise boundary approaches the strike price as the time approaches the expiration date.

2.4 - Stochastic Meshes

Generally, a Monte Carlo simulation will generate several paths based on different random numbers. The paths are discretized as far as the time periods used, and as the time period length decreases the paths approach the actual continuous exercise time.

The price of an option is given by the solution to a partial differential equation. For certain types of options a closed-form solution is possible, and for European options the solution is given by the Black-Scholes model, see Section 2.1 and Eq. (2.1) and (2.2). For option types that do have a closed-form solution, several approximation techniques can be employed. The methods of stochastic mesh, finite differences, and finite element analysis will be explored in the remaining parts of this section.

2.4.1 - Mesh Method

As an extension of using stochastic meshes to price American options, mesh methods were developed. Broadie and Glasserman (2004) introduces a stochastic mesh method for pricing high-dimensional American options when there is a finite – possibly large – number of exercise dates. Their method requires knowledge of the transition density of the underlying process of the asset prices and other state variables. However, Broadie, et al (2000) extend the original stochastic mesh method to applications where the transition density is either unknown or fails to exist. They choose mesh weights through a constrained optimization problem, so the transition density is unneeded. They also use information regarding the movements of the underlying state variables or the prices of easily computed European options to prevent the use of a transition density. Finally, Achdou and Pironneau (2005) propose a method of adaptive mesh refinement that is faster because it has fewer unknowns for a given accuracy than a given uniform mesh.

2.4.2 - Finite Difference Method

For option types that do have a closed-form solution, a technique from numerical analysis called finite difference method can be used. The finite difference method is based on the mathematical expression of the form:

$$f(x+b) - f(x+a). \qquad (2.5)$$

Finite differences and the finite difference method are one of the simplest ways to approximate a differential operator; therefore, they are very helpful for solving differential equations.

One of the first finite difference algorithms for pricing American put options comes from Brennan and Schwartz (1977). They propose an algorithm that discretizes the Black-Scholes model using finite differences. Achdou and Pironneau (2005) give the formal mathematical definitions of finite differences method used in financial engineering. Others, including Hull and White (1990), Wilmott (1998), and Hull (2006), discuss the use of finite differences to convert the underlying differential equation into a difference equation, either by explicit or implicit methods.

Shu, et al. (2005) propose a modification to the finite difference method by combining the advantages of a semi-analytical method with the sliced-fixed boundary finite difference method. Oosterlee, et al. (2004) aim to decrease the number of grid points by using fourth order finite differences, instead of the general quadratic model. They also employ grid stretching in space by means of an analytic coordinate transformation in order to decrease the number of grid points required to obtain the same accuracy. Also, Persson (2006) discusses the space-adaptive and time-adaptive finite difference methods. Finally, Zhao, et al. (2007) propose a compact finite difference method to convert the Black-Scholes partial differential equation into an ordinary differential equation.

2.4.3 - Finite Element Method

Similar to finite difference method discused in Section 2.4.2, the finite element method approximates a partial differential equation when a closed form solution is not possible. Where a finite difference method approximates the actual differential equation, the finite element method approximates its solution. The finite element methods are based on either eliminating the differential equation completely or rendering the partial differential equation into an equivalent

ordinary differential equation. If the later approach is used, then the ordinary differential equation can be solved using standard techniques, such as finite differences.

The general finite element method is to create a mesh or grid. This grid can consist of triangles, squares, or curvilinear polygons that discretize the infinite dimensional surface. Then a basis is chosen to complete the discretization. Basis functions can be either linear or piecewise polynomial.

The first person attributed with applying the finite element method to the financial sector was Courant (1943). Achdou and Pironneau (2005) propose methods employing finite element analysis to a parabolic boundary value problem, along with the Black-Scholes equation in one and two dimensions. Allegretto, et al. (2001) propose a new finite element method, but then show that the method presented in Allegretto, et al. (2003) actually produces more accurate results. Finally, Zhang (2005) proposes an adaptive mesh refinement to the finite element analysis approach to help reduce the size of the linear systems.

2.5 - Least-Squares Approach

When assessing an American option it is necessary to evaluate the value of exercising and continuing at the early exercise points. Computing the value of exercising the option is relatively simple, and generally the Black-Scholes formula is used. However, computing the value of continuing to hold the option until a later time period can be tricky. Tsitsiklis and Van Roy (2001) proposed a method for evaluating complex American options using simulation and dynamic programming. Here simulation is used to create each successor state. The recursive function of the dynamic programming is used backward to determine if the American option should be exercised or held at each time period.

Broadie, et al. (2000) also use dynamic programming in their model, but also include weights involved in the American option prices. The conditional expectation of the dynamic programming recursion is estimated from the weighted averages of the simulated future values. Haugh (2004) discusses using approximate dynamic programming and simulation to compute the upper and lower bounds on the true option price.

Longstaff and Schwartz (2001) propose an approach using least-squares to estimate the value of continuation. This method entails first simulating the paths. Then, starting at the final time period, each path is evaluated to see if the option would be exercised, and the associate cash

flows are recorded. The algorithm then backs up one time period, and the paths are examined to see which are in the money. For each path that could be exercised, the algorithm performs a linear regression. The least-squares approach for the linear regression results in a function that relates the current option value to the value of continuing. Longstaff and Schwartz use a simple quadratic regression function contained below:

$$Continuing = \beta_0 + \beta_1 * exercise + \beta_2 * exercise^2 . \qquad (2.6)$$

The function is then evaluated for each path that is in the money, and compared with the value of immediate exercising. The cash flow matrix is then updated to reflect the which paths would be exercised in the current time period, and the algorithm proceeds to the next previous time period. Once all time periods have been examined, the stopping rule or location of each path can be compiled and the cash flow matrix will contain the gains realized by the American option. Unlike previous algorithms, Longstaff and Schwartz's method only performs the regression when the option is in the money; therefore, the algorithm is more efficient.

Glasserman and Yu (2004) also discuss a weighted Monte Carlo technique that is the same as the least-squares method performed at a later time. This method still requires a regression, but since the regression is done later, the estimates produced are less dispersed and an upper bound on the option price can be provided.

Chan, et al. (2004) also discuss using Monte Carlo simulations for pricing American options. Instead of generating the paths and then performing the optimal exercising problem backwards, they propose a simulation method that generates the paths backward. Given a particular initial stock price S_0, the solution Eq. (2.2) is modified for each path and defined as:

$$\begin{aligned} S_1 &= S_0 \exp([r - \tfrac{1}{2}\sigma^2]\Delta t + \sigma\sqrt{\Delta t}\varepsilon_N) \\ &\vdots \\ S_i &= S_0 \exp(i[r - \tfrac{1}{2}\sigma^2]\Delta t + \sigma\sqrt{\Delta t}(\varepsilon_N + \varepsilon_{N-1} + \ldots + \varepsilon_{N-i+1})) \\ &\vdots \\ S_N &= S_0 \exp(N[r - \tfrac{1}{2}\sigma^2]\Delta t + \sigma\sqrt{\Delta t}(\varepsilon_N + \varepsilon_{N-1} + \ldots + \varepsilon_1)) \end{aligned} \qquad (2.7)$$

where

$$\varepsilon \sim \mathrm{N}(0,1) .$$

Chan, et al.'s backward generation method generates the random numbers for each path at every time period using only an initial seed value. This allows the random numbers to be regenerated from the seed value as the algorithm is progressing backward through time. This means the

random numbers must be generated twice, but saves computer memory by not storing each value. Similar methods to reduce the amount of memory needed in the path generation are proposed in Chan, et al. (2002, 2006).

One of the simulated models contained in this work uses the backward path generation of Chan, et al. (2004) and the relatively simple least-squares regression found in Longstaff and Schwartz (2001).

CHAPTER 3 - LEAST-SQUARES METHOD

As seen is Section 2.5, American options can be estimated using least-squares approach with a backward path generation. Chan et al. (2004) presented a simulation method in which the paths are generated in a backward time sequence. This saves computer storage by only simulating the paths when they are needed; however, the price of generating the path is that each random number must be generated twice. In their paper, Chan et al. (2004) approximated the value of continuing using the least-squares scheme, which was originally presented in Longstaff and Schwartz (2001). In this chapter, the method using least-squares scheme is proposed using a discrete simulation implementation. First, the algorithm from Chan et al. (2004) with a slight modification is presented along with a run time analysis for the least-squares algorithm. This method uses the least-squares approach of Longstaff and Schwartz (2001), but the adaptation of the backward path generation allows for less memory storage required to complete the algorithm. The further modification from the algorithm presented by Chan et a. (2004) entails using the next time periods cash flow in the least-squares regression computations, instead of the maximum cash flow over all future time periods. Next, the implementation of the least-squares algorithm using discrete simulation is discussed. Finally, several different inputs are used to show the outputs of the least-squares simulation model, along with results of using two variance reduction techniques.

3.1 - Notation

The following notation will be used for the remaining portions of the thesis:

$S_0 \equiv$ Initial stock price

$r \equiv$ Risk-free interest rate

$K \equiv$ Strike price

$\sigma \equiv$ Stock volatility

$N \equiv$ Number of time periods

$\Delta t \equiv$ Length of each time period (in years)

$M \equiv$ Number of paths

$T \equiv$ Expiration time

$Z \sim$ Independently identically distributed from *N(0, 1)*

$\omega_i = Z_N + Z_{N-1} + \ldots + Z_{N-i+1}$

3.2 - Algorithm

An adaptation of the algorithm given in Chan et al. (2004) is implemented for one of the models of this research. The least-squares approach of Longstaff and Schwartz (2001) is used with the addition of backwardly generating paths. However, in the linear regression portion, the previous period's cash flows are used instead of the maximum cash flow from all future time periods. The algorithm is now presented as follows:

Step 0. System Inputs:

(a) Initial stock price (S_0)

(b) Risk-free interest rate (r)

(c) Strike price (K)

(d) Stock volatility (σ)

(e) Number of time periods (N)

(f) Length of each time period in years (T)

(g) Call or put option

Step 1. Initialization:

(a) Set the seed for the initial path to any positive integer.

(b) Generate the random numbers Z_j for each path j = 1, 2, … M and compute their sum ω_N.

Step 2. Compute S_N for the expiry date T using:

$$S_i = S_0 \exp(i[r - \tfrac{1}{2}\sigma^2]\Delta t + \sigma\sqrt{\Delta t}\omega_i)\,. \qquad (3.1)$$

Step 3. Compute the cash flows for each path using one of the following:

$$P(j) = \begin{cases} \max\{K - S_j(t), 0\} & \text{given put option} \\ \max\{S_j(t) - K, 0\} & \text{given call option} \end{cases} \quad (3.2)$$

Step 4. Backup one time period; set $i = i - 1$.

(a) Using the same seed sequence, extract Z_{N-i+1} and compute $\omega_{i-1} = \omega_i - Z_{N-i+1}$.

(b) Compute S_{i-1} by using (3.1).

(c) Extract the next seed value.

Step 5. Compute if the option is in the money for each path *j*. For each path:

(a) Let X be the vector containing asset prices S_i and Y be the vector containing the corresponding cash flows received at i+1 time period, which have been discounted backward to the i^{th} time period.

(b) Regress using least-squares approach to estimate the value of continuing using the Eq. (2.6). This will result in the conditional expectation function $\mathrm{E}[Y|X]$.

(c) Compute the value of continuing using $\mathrm{E}[Y|X]$ and the value of immediately exercising using Eq. (3.2).

(d) Determine whether to exercise the option immediately or hold the option until the next time period, based on which gives the higher expected value. Establish the current cash flows conditional on not exercising prior to time period *i* using:

$$C_i(j) = \begin{cases} \text{cash flow} & \text{if cash flow} \geq \mathrm{E}[Y|X] \\ 0 & \text{otherwise} \end{cases} \quad (3.3)$$

(e) Compute the present value of the cash flows $P_i(j)$ given by:

$$P_i(j) = \max\{C_i(j),\ e^{-r\Delta t} P_i(j)\}. \quad (3.4)$$

Step 6. If at time period one terminate, else go back to Step 4.

There are two main contributors to the computational effort required for the least-squares algorithm, which are computing the stock price and solving the least-squares regression equations. For the stock price computation of Step 2, a stock price must be computed for every path M and every time period N, so the running time is O(NM). To solve the least-squares regression equation of Step 5 a 3×3 matrix must be solved, and the least-squares regression equation might potentially be solved a total of N times, i.e. 3^2N. Therefore, the running time for the least-squares regression equation portion is O(N), and the running time for the entire least-squares algorithm is O(NM).

3.3 - Implementation and Computational Experiments

While Section 3.2 presented the least-squares algorithm with backwardly generated paths, this section discusses the implementation and computational experiments associated with this algorithm. Unlike most financial engineering algorithms, the program created here was using discrete event-based simulation software, Arena. After the model was created, multiple sets of inputs are explored to show the performance of the model.

3.3.1 - Least-Squares Simulation Model

Traditionally, financial engineering problems were simulated using math software, such as MATLAB, general purple programming language such as, C or C++; however, in this research a discrete event simulator software called Arena, from the Rockwell Automation, Inc (Rockwell Automation) is used to greatly simplify the programming efforts and complexity. The following parts will describe how the algorithm of least-squares is simulated.

The first step from the algorithm given in Section 3.2 (Step 0) is to enter the system inputs. These inputs need to be stored manually into the variables by assigning each variable an initial condition. Also, each entity in the model is simulating a unique path; e.g. 50 entities will create 50 sequences of possible stock prices over the planned horizon referred as the “paths”. The program was designed to simulate 200 paths, but can be easily modified to create fewer or more paths by changing the number of entities that are sent into the simulation system.

Next, a loop is created to complete Step 1. Each entity is given a number that represents the path number, so that each path can be distinguished in future computations. Then the random numbers from a normal distribution are generated with specific seed values. Two hundred pairs of seed values were used to initialize in the simulation system. The following equations give the

general form for normal distributions in Arena, and then the specific form used to generate the random number in the model.

$$\begin{aligned} Z_j &= Norm(\text{mean},\ \text{standard deviation},\ \text{seed value}) \\ &= Norm(0,\ 1,\ \text{path number}). \end{aligned} \tag{3.5}$$

Since the seed value specified as the path number, each entity has a unique random number sequence. Each entity will then add the generated Z_j value to ω and continue to loop until all time periods have been completed.

To complete Step 2 and Step 3, the model then computes the stock prices for each time period according to the formula given in Eq. (3.1). Then, the associated payoff is determined using Eq. (3.2) based on whether an American call or American put was specified initially.

Next, the model creates another loop for Step 4 through Step 6. Since this loop simulates the backing up of time periods, the next value of the seed sequence needs to be extracted. Remembering that the seeds are inputted into the model in pairs, the entity needs to call up the corresponding pair value for the second generation of the standard normal distribution. Since both the value of seed number 1 and number 201 are the same value, the second time generating the standard normal needs to call up the 201 value. The modified form of Eq. (3.5) is given below:

$$Z_j = Norm(0,\ 1,\ \text{path number} + 200). \tag{3.6}$$

For computing the least-squares estimators, three tandem Visual Basic for Applications (VBA) subroutines were used within the Arena simulation model. The first subroutine read and stored all input data into the variables or variable arrays. Then, all entities wait until each path has completed the first subroutine before allowing the second subroutine to fire once. The second subroutine calculates the estimate of the regression coefficients – β_0, β_1, and β_2 from Eq. (2.3) – and then, the value of continuation. Then, the decision on exercise the option immediately or hold until a later time period is evaluated by Eq. (3.4) and the present value of the cash flows is computed by Eq. (3.4). Finally, the third subroutine is used to store all necessary values back to the simulation model for use later in the program or for outputs of the program.

Since the loop simulates Step 4 through Step 6, once the first time period is completed the loop is completed. Therefore, the simulation has started with the final time period, evaluated each time period in a backward fashion, and ended with the first time period. Since there is no

termination condition – for example, simulation time or number of paths that have been exercised – the simulation is considered non-terminating. Hence, the simulation stops when all paths have executed through the logic.

Once the simulation is completed, the outputs of the model are collected and written to output files, although this is not necessary using Arena. The user could decide to use Arena internally to compute required statistics – such as averages, minimum, or maximum – for the users, but here the raw data was written to files for use in graphs and tables. Figure ?? contains the simulation model used for the least-squares method. The discussions on the computational results generated from the proposed discrete event simulation model are presented in Section 3.3.2.

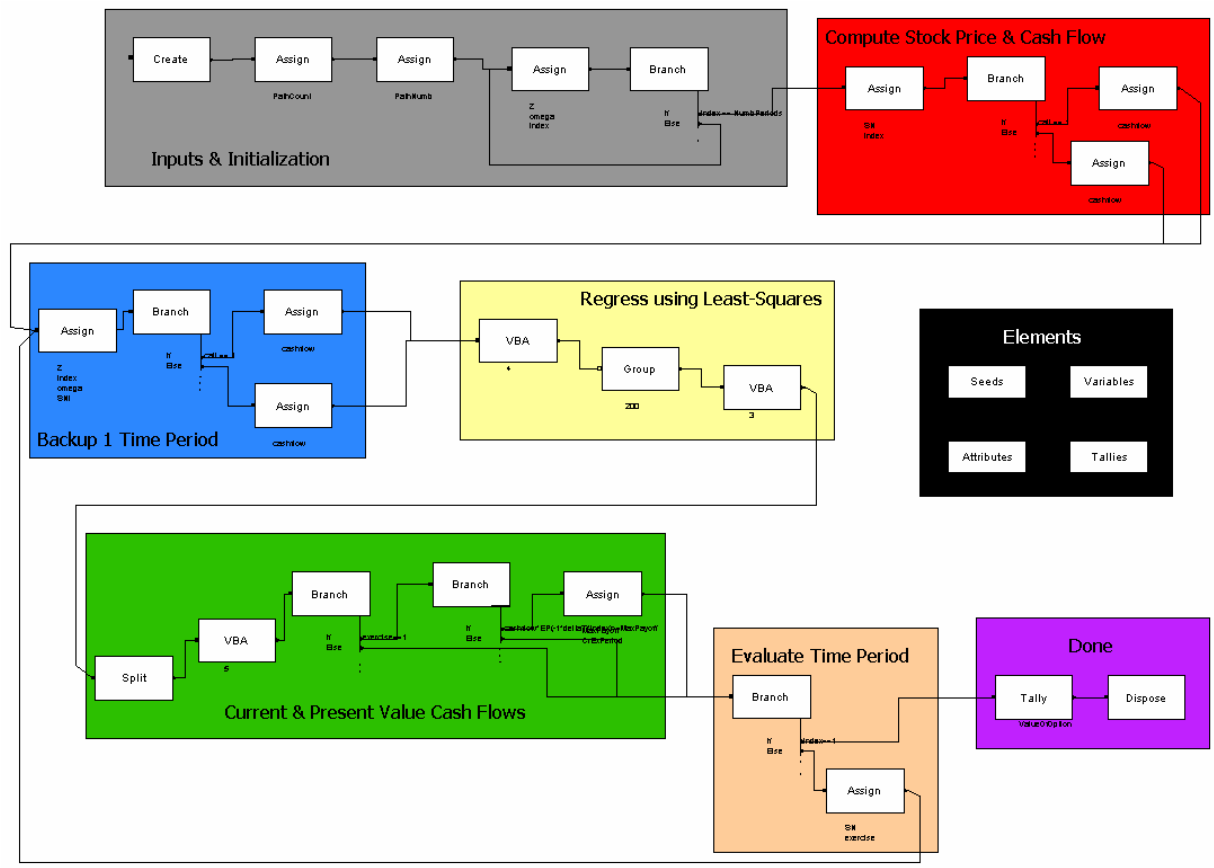

Figure 3.1 - Least-Squares Simulation Model

3.3.2 - Computational Results

Using the least-squares simulation model several sets of inputs were tested to show the performance of the model. As explained in Section 2.2, American call options will rarely be exercised early without dividends, therefore only American put options were considered when testing the simulation models. This does not imply the model cannot accommodate an American call option, the American put was simply chosen to use for the computational tests. The first section gives four examples and their corresponding graphs to show the influence of inputs on the stock prices that are generated by the model. The next section gives the resulting cash flows and optimal exercise time that are generated by the model. Finally, Example 6 and Example 7 show how antithetic variate and control variate techniques can be used to reduce the variability in the model.

3.3.2.1 - Example 1 to Example 4

Example 1 through Example 4 has two different pricing inputs and two different time period inputs. The two pricing inputs were chosen with a small verses large difference in price. The small difference is only $2, while the relatively large difference is $10. The stock volatility also is given either a high value of 0.20 or a low value of 0.05, corresponding to 20% and 5% variability, respectively. Similar to the pricing inputs, the time period inputs were given a wide range for the two inputs chosen. One set of inputs used a large planning horizon of a year and a small number of time period, resulting in a long period length of 73 days. Conversely, the second set of time period inputs use a moderate planning horizon of a quarter of a year, along with a modest time period length of a week. The following table gives the input combinations for each run of the model for the first four examples.

Table 3.1 - Least-Squares Model Inputs for Four Test Cases

Parameter	Symbol	Input 1	Input 2	Input 3	Input 4
Initial Stock Price	S_0	$100	$100	$125	$125
Strike Price	K	$112	$112	$135	$135
Stock Volatility	σ	0.20	0.20	0.05	0.05
Interest Rate	r	0.08	0.08	0.08	0.08
Length of Each Time Period	Δt	0.2000	0.0192	0.2000	0.0192
Number of Time Periods	N	5	13	5	13

Each set of inputs from Table 3.1 were run using the least-squares simulation model. Each set of inputs was run using 200 paths to simulate the behavior of the real stock prices. Note all models used the risk-free interest rate of 8% and the length of the time periods are reported in years. Therefore, Input 1 and Input 3 are simulated for one year, while Input 2 and Input 4 are simulated for a quarter of a year and each time period corresponds to a week. The stock prices generated from each path were then graphed verses time, and the resulting four graphs are following.

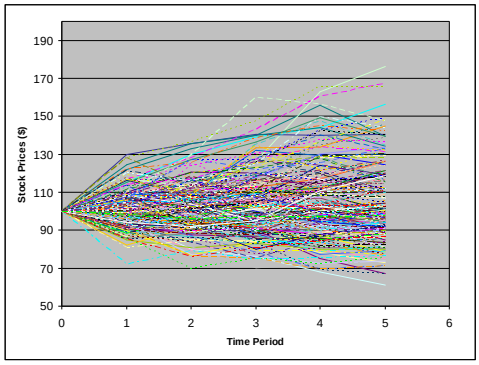

Figure 3.2 - Stock Prices with Input 1

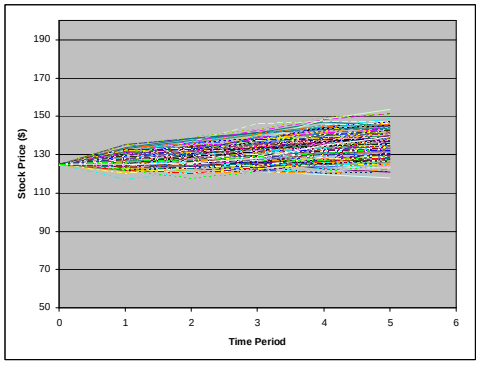

Figure 3.4 - Stock Prices with Input 3

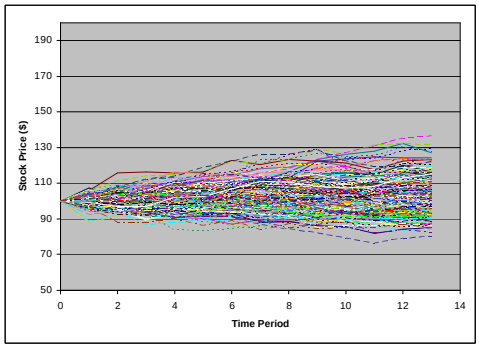

Figure 3.3 - Stock Prices with Input 2

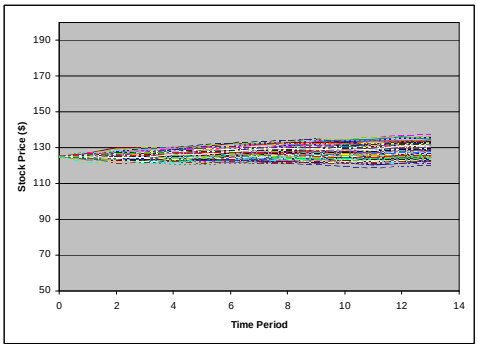

Figure 3.5 - Stock Prices with Input 4

In comparison of the graphs, the vertical axis is held constant from $50 to $200, while the horizontal axis depends on the length of time being simulated. The stock prices follow the general shape of increasing variability with increasing time, and the variability is symmetric about the initial stock price. This confirmed the idea that patterns of the stock prices approximate the Brownian motion. The input of the stock price volatility determines the amount of variability that is prescribed in the simulation model. In these examples, the graphs

corresponding to the stock volatility of 0.20 – Figure 3.1 and Figure 3.2 – have noticeably wider spreads at the final time period than the graphs depicted in Figure 3.3 and 3.4 with the stock volatility of 0.05. In contrast to the variability, the shape of the graph is not dependent on the input of initial stock price (another input parameter in the simulations). If the stock price in Input 1 was lowered from $100 to $80, the graph would simply shift down in the vertical axis, but the general shape would not change since the shape is dedicated by the stock volatility.

3.3.2.2 - Cash Flow and Optimal Exercise Times (Example 5)

Example 5 gives the present value cash flows and associated optimal exercise times that are generated by the least-squares model. Using the inputs from Example 4 – which are given under Input 4 in Table 3.1 – ten paths were simulated. Table 3.2 gives the cash flows for each path of the American put option. The current values of the cash flows $P_i(j)$ are calculated by Eq. (3.2) or Eq.(3.4) depending on the time period. Therefore, it is either the value of the cash flows conditional on not exercising prior to time period i, or the previous cash flow value $P_{i+1}(j)$ discounted back to the current time period i.

Table 3.2 - Present Value Cash Flows for Example 5

Path j	$P_0(j)$	$P_1(j)$	$P_2(j)$	$P_3(j)$	$P_4(j)$	$P_5(j)$
1	2.91	2.96	3.00	3.05	0.00	0.00
2	4.54	4.62	4.69	4.77	4.84	3.91
3	3.49	3.54	3.60	3.66	3.72	5.79
4	0.00	0.00	0.00	0.00	0.00	0.00
5	11.73	11.92	12.11	12.31	12.51	14.37
6	3.55	3.61	3.67	3.73	3.79	5.79
7	0.00	0.00	0.00	0.00	0.00	0.00
8	6.70	6.81	6.92	0.00	0.00	0.00
9	5.07	5.16	5.24	5.32	5.41	4.48
10	14.46	14.70	14.93	15.18	15.42	17.23
Average	**5.25**					

As Table 3.2 shows, the American put option is never in the money for Path 4 and Path 7, while it is in the money for at least one time period for all other simulated paths. Since the present value of the cash flows is either the next time period's payoff discounted back to the current time period or the value of exercising in the current time period, a non-zero value does

not specify which of these scenarios exist for each time period along the path. Therefore, the program also reports the time period when the value of immediately exercising is greater than the value of continuation. The exercise points for Example 5 are given in Table 3.3 below.

Table 3.3 - Exercise Period for Example 5

Path j	$T1$	$T2$	$T3$	$T4$	$T5$
1	0	0	X	0	0
2	0	0	0	X	0
3	0	0	0	X	0
4	0	0	0	0	0
5	0	0	0	X	0
6	0	0	0	X	0
7	0	0	0	0	0
8	0	X	0	0	0
9	0	0	0	X	0
10	0	0	0	X	0

Using the information about when each period is exercised and the cash flow value at the point of exercising, the value of an option can be computed. The value of an American put option is given by:

$$V = \frac{\sum_i e^{-r*\Delta t} \text{ cash flow}_i}{\text{Number of paths}}. \qquad (3.7)$$

Regardless of the exercise time, the cash flows are discounted back to period zero and then averaged to obtain the value of the option. The discount factor for the cash flows is dependent on which time period the put option is exercised. Table 3.2 also gives the discounted cash flow values for each path and the associated value of the American option is $5.25. The value (or the price) of the option allows users to estimate how much they should be willing to pay upfront to buy the option. Since the initial price of most options is relatively small compared to the price and high volume of trading, the mathematical model does not include this initial cost. The computed value of Example 5 tells a user that if they found an American put option with the specified inputs and the underwriters were offering the option for $3.50, the user should buy the option since the average value of the option is $5.25. However, if the underwriters were offering the option for $6.00, the user would know that the average value of the option is below the initial

price; therefore, it is not advantageous to buy the option or the user themselves can offer (underwrite) another put option with lower than $6.00 price to guarantee a profit.

3.3.2.3 - Antithetic Variate Variance Reduction Technique (Example 6)

Antithetic variate technique is a variance reduction technique that can reduce the variability of the simulation outputs. A summary discussion of the antithetic variate method was presented in Section 1.1.3. In general, the antithetic variate technique can be used for any model output that is dependent on a random number generator. For Example 6, the variance in cash flow of the American put option is reduced. However, the variance could have also been reduced in other outputs that were dependent on the Z_i or ω values, the average stock price for example.

The inputs for Example 6 are given in Table 3.1 under Input 3, and 10 paths were used to generate the example. The model was run for 20 replications for each system. The resulting cash flow payoffs are given in Table 3.4 below.

Table 3.4 - Antithetic Variate on Least-Squares Model

Replication	Antithetic Variate Payoff		Base System Payoff
	Raw Data	Paired Data	
1	6.96		6.96
2	6.45	6.70	4.01
3	4.01		4.44
4	5.79	4.90	3.15
5	4.44		4.18
6	7.19	5.81	5.42
7	3.15		4.90
8	6.20	4.67	2.73
9	4.18		6.74
10	5.96	5.07	4.51
11	5.42		3.28
12	8.00	6.71	7.00
13	4.90		4.47
14	9.77	7.34	4.73
15	2.73		5.63
16	10.06	6.39	7.34
17	6.74		3.88
18	6.88	6.81	8.76
19	4.51		7.19
20	6.58	5.54	4.56
Average over Replications		**5.99**	**5.19**
Variance in Replications		**0.853**	**2.664**
F-statistic		**0.3201**	
Reject?		**Yes**	

According to Eq. (1.2), the payoffs from each pair in the antithetic variate technique are averaged together. The average over the 10 replications is then computed and compared to the straight average cash flow payoff from the base system. The sample variance is also computed for each system. As Table 3.4 reports, the average cash flow for the antithetic variate method is \$5.99, and the average cash flow for the base system is \$5.19; therefore, both systems produced similar average cash flow values. In contrast, the variance in the replications between the systems is considerably different. The variance in the base system is 2.664, while the variance in the antithetic variate system is 0.853, a 68% reduction.

A formal hypothesis test was conducted to test to see if there was a difference in the variances between the two systems. The null hypothesis is that the variances are equal, $\sigma_1^2 = \sigma_2^2$, verses the alternative of unequal variances $\sigma_1^2 \neq \sigma_2^2$. The *F*-test determined the variance in the replications between the antithetic variate and base system is significant at a 95% confidence level. Also, the corresponding confidence interval on the difference in system means is (–0.16, 1.76) at 95% confidence level; therefore the simulated average stock prices are not significantly different at a 95% confidence level.

Since the number of paths in the above example is relatively small, Example 6 was repeated using an increasing number of paths. The resulting average stock prices and variance in stock prices, along with results from the *F*-test at 95% confidence level are reported in Table 3.5 below. The degrees of freedom for the *F*-test are reported for the numerator (antithetic variate system) then the denominator (base system).

Table 3.5 - Antithetic Variate Results Varying the Number of Paths

# of Paths	Antithetic Variate System		Base System		*F*	Deg. of Freedom	Reject?
	Average	Variance	Average	Variance			
10	5.99	0.853	5.19	2.66	0.3201	9, 19	Yes
15	5.72	0.699	5.09	1.631	0.4285	9, 19	No
25	5.50	0.501	5.15	0.945	0.5303	9, 19	No
50	5.31	0.241	5.14	0.436	0.5533	9, 19	No
200	5.40	0.083	5.43	0.112	0.7370	9, 19	No

As Table 3.5 shows, the number of paths does not affect the average stock price greatly, while higher number of paths affects the variance between the replications. Even increasing the number of paths by five causes the F-test to fail to reject the null hypothesis. Therefore, there is no proof that the antithetic variate technique produces significantly lower variances when the number of paths is above 10.

3.3.2.4 - Control Variate Variance Reduction Technique (Example 7)

The final example for the least-squares model uses the control variate technique to reduce the variability in the stock price. For an overview of the control variate method please see Section 1.1.4. In particular, the control variate technique uses two simulated prices and one

theoretical value to remove the unnecessary variance the Monte Carlo simulation produces. The formula for the adjusted American stock price is given by Eq. (1.6) in Section 1.1.4, and states the simulated European option price and the theoretical or Black-Scholes value can be combined to contribute nothing to the expected value of the American option price $E(f_A)$, but reduce the variability, see Eq. (1.4) and (1.5). The value of the parameter a that minimized the variability in the adjusted value is given in Eq. (1.3). It divides the covariance between the original option and the control variate, with the variance of the control variate. In this case, the original option is the simulated American option and the control variate is the simulated European option.

The inputs for Example 7 are given in Table 3.1 under Input 3 and the simulation was run using 200 paths. The first ten paths are given in Table 3.6; however, all paths are used to calculate the summary statistics given at the bottom of the table.

Table 3.6 - Control Variate on Least-Squares Model

Rep.	Sim. American f_A	Sim. European f_E	Black-Scholes f_{BS}	*Adjusted American* f_C
1	131.95	139.27	135.54	129.63
2	130.16	131.09	135.54	132.93
3	131.28	129.21	135.54	135.22
4	122.49	120.63	135.54	131.77
5	131.21	129.21	135.54	135.15
6	129.59	13.052	135.54	132.72
7	119.58	117.77	135.54	130.64
8	123.72	123.18	135.54	131.41
9	131.43	130.89	135.54	134.32
10	124.25	124.11	135.54	131.36
Average	**129.87**			**132.02**
Variance	**13.630**			**3.687**
F-statistic	**3.696**			
Reject?	**Yes**			

The covariance between the simulated American option price and the simulated European option prices is 15.774, and the variance for the European option prices is $25.345. Based on the covariance and variance, the parameter a was computed as 0.622. As Table 3.6 shows, the variance difference is large between the original American option and the adjusted American option value. Again, an F-test was conducted to test if there was a difference in the variances

between the two systems. The null hypothesis is that the variances are equal, $\sigma_1^2 = \sigma_2^2$, verses the alternative of unequal variances $\sigma_1^2 \neq \sigma_2^2$. The resulting *F-statistics* is 3.696, so the variances are significantly different at 95% confidence level. Therefore, the control variate technique significantly reduces the variance in the American options for Example 7.

CHAPTER 4 - FINITE DIFFERENCE METHOD

Section 2.4 introduces another method of evaluating American options using a stochastic mesh and either finite differences or finite element techniques. Achdou and Pironneau (2005) propose an algorithm that creates the exercise boundary by finite difference method. They also provide a C++ code to compute their Front-Tracking algorithm. In this chapter, the finite difference method is first discussed. Then, the method using the Front-Tracking algorithm from Achdou and Pironneau (2005) to create the exercise boundary and the simulation model is presented, along with a running time analysis for the algorithm. Finally, the outputs of both the Front-Tracking algorithm and the simulation model are presented.

4.1 - Background

Regardless of which solution procedure is used, all algorithms of this type require a stochastic mesh. A stochastic mesh is a grid in which points correspond to a time period and a stock price. Generally, the time and stock price intervals are equally spaced, so $\Delta t = T/N$ and $\Delta S = S_{max}/Q$; therefore, there are a total of $N+1$ time periods and $Q+1$ stock prices. The time points and stock price points create a grid consisting of a total of $(N+1)(Q+1)$ points, and an example is given in Figure 4.1 below.

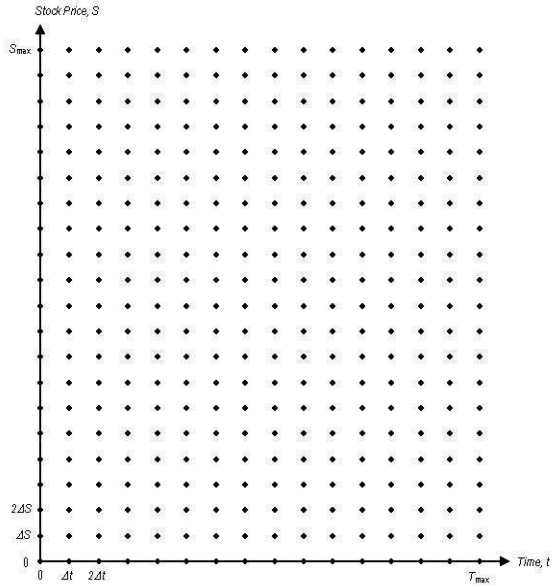

Figure 4.1 - Stochastic Mesh

For a given point (i ,j), the corresponding time period is $i\Delta T$ and the stock price is $j\Delta S$, and then the value of the option at the same point (i, j) would be $f_{i,j}$.

After the grid or stochastic mesh is created, the finite difference method can be preformed to determine the exercise boundary. There are two forms of the finite difference method, which are distinguished by whether the values are generated forward or backward in time. A graphical depiction of the difference between the two methods is given in Figure 4.2 below.

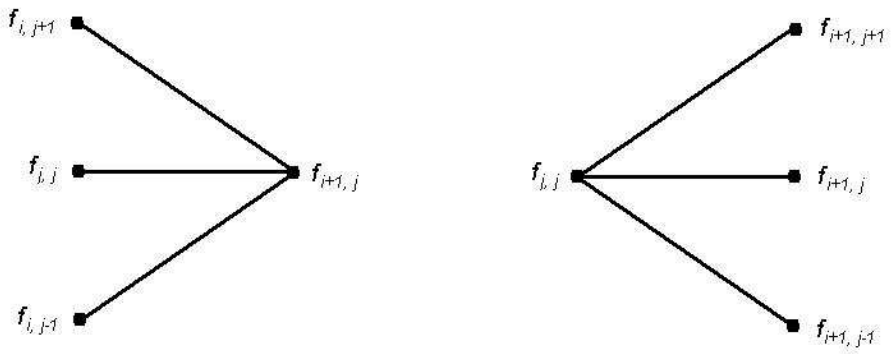

Figure 4.2 - Difference Between Implicit (left) and Explicit (right) Methods

As the figure shows, the implicit finite difference method is the forward approximation and explicit is the backward method. For a symmetrical approximation, an average of the high and low point are used to obtain the next value in the implicit method, the expression is as follows:

$$\frac{\partial f}{\partial S} = \frac{f_{i,j+1} - f_{i,j-1}}{2\Delta S}. \qquad (4.1)$$

Similarly, the equation relating the next value at $i\Delta t$ and the previous value $(i+1)\Delta t$:

$$\frac{\partial f}{\partial t} = \frac{f_{i+1,j} - f_{i,j}}{\Delta t}. \qquad (4.2)$$

This leads to a finite difference approximation for $\partial^2 f / \partial S^2$ at the point (i, j) as follows:

$$\frac{\partial^2 f}{\partial S^2} = \frac{f_{i,j+1} + f_{i,j-1} - 2f_{i,j}}{\Delta S^2}. \qquad (4.3)$$

These values can then be plugged into the differential equation that the American put option must satisfy, Eq. (4.4), and be simplified into Eq. (4.5).

$$\frac{\partial f}{\partial t} + rS\frac{\partial f}{\partial S} + 0.5\sigma^2 S^2 \frac{\partial^2 f}{\partial S^2} = rf \tag{4.4}$$

$$a_j f_{i,j-1} + b_j f_{i,j} + c_j f_{i,j+1} = f_{i+1,j} \tag{4.5}$$

where

$$a_j = 0.5rj\Delta t - 0.5\sigma^2 j^2 \Delta t,$$

$$b_j = 1 + \sigma^2 j^2 \Delta t + r\Delta t, \text{ and}$$

$$c_j = -0.5rj\Delta t - 0.5\sigma^2 j^2 \Delta t.$$

Eq. (4.5) defines all interior points for the grid. The points along the edges of the grid are defined by the following boundary conditions:

- The value at time T (ie. Upon expiration), $f_{N,j} = \max(K - j\Delta S, 0)$
- The value when the stock price is zero, $f_{i,0} = K$
- The put option being worth zero when $S = S_{max}$.

From Eq. (4.5), Q-1 simultaneous equations must be solved in order to calculate the point $f_{i,j}$ as $j = 1, 2, \ldots Q\text{-}1$ for all $i = 1, 2, \ldots N$. Therefore, the simultaneous equations will need to be solved a total of N times. Since solving a system of linear equations using LU decomposition has a running time of $O(N^3)$, the running time for solving the system of linear equations is $O(Q^{N^3})$.

For the explicit finite difference method, the assumption that $\partial f / \partial S$ and $\partial^2 f / \partial S^2$ are the same at point (i, j) as at point $(i+1, j)$. Then Eq. (4.1) and (4.3) can be simplified and results in the following modification of Eq. (4.5):

$$f_{i,j} = a_j^* f_{i+1,j-1} + b_j^* f_{i+1,j} + c_j^* f_{i+1,j+1} \tag{4.6}$$

where

$$a_j^* = \frac{1}{1+r\Delta t}\left[-0.5rj\Delta t + 0.5\sigma^2 j^2 \Delta t\right],$$

$$b_j^* = \frac{1}{1+r\Delta t}\left[1 - \sigma^2 j^2 \Delta t\right], \text{ and}$$

$$c_j^* = \frac{1}{1+r\Delta t}\left[0.5rj\Delta t + 0.5\sigma^2 j^2 \Delta t\right].$$

Upon examining the implicit method, Eq. (4.5) relates three different values at time $i\Delta t$ and one value at time $(i+1)\Delta t$. Conversely, the explicit method given in (4.6) relates one value at $i\Delta t$ and

three different values at time $(i+1)\Delta t$. The advantage of the implicit finite difference method is that it is very robust and always converges to the solution of the differential equation as ΔS and Δt approach zero. As stated above, the implicit method requires solving M-1 simultaneous equations, but the additional assumptions for the explicit method allows for computational simplification.

In Achdou and Pironneau's (2005) Front-Tracking algorithm, the explicit finite difference method described above is used. The difference between the generic explicit finite difference method and their Front-Tracking algorithm is the method to solve the simultaneous equations.

4.2 - Algorithm

The algorithm for the second model of this research uses a stochastic mesh and finite difference method to determine the optimal exercise time for American put options. It first calculates the exercise boundary, then simulates the American option price to see if the current price in the exercise region. The same notation presented in Section 3.1 is used here, with the addition of the following:

$P \equiv$ American option payoff

$\gamma \equiv$ Function leading to the exercise boundary

$n \equiv$ Partition on interval $[0, T]$ into subintervals $[t_{n-1}, t_n]$ where $1 \le n \le N$

$h \equiv$ Maximum size of the subintervals on S

$(0, \bar{S}) \equiv$ localized interval for the problem

The algorithm is as follows:

Step 0. System Inputs:

(a) Initial stock price (S_0)

(b) Risk-free interest rate (r)

(c) Strike price (K)

(d) Stock volatility (σ)

(e) Number of time periods (N)

(f) Length of each time period (Δt)

Step 1. Create exercise boundary using Front-Tracking Algorithm (Achdou and Pironneau, 2005)

(a) Choose *k* such that $\gamma_h^{n-1} = S_k$.

(b) Solve discrete finite difference problem corresponding to:

$$\frac{P^n - P^{n-1}}{\Delta t_n} - \frac{\sigma^2(S,t_n)S^2}{2}\frac{\partial^2 P^n}{\partial S^2} - rS\frac{\partial P^n}{\partial S} + rP^n = 0 \quad \text{for } \gamma_h^n < S < \bar{S}$$

$$P^n = P_0 \quad \text{for } 0 \le S \le \gamma_h^n$$

and $P^n(\bar{S}) = 0$.

(c) Stop if P^n satisfies $P^n \ge P_0$ and the following:

$$\forall v, \quad \left(P^n - P^{n-1}, v - P^n\right) + \Delta t_n a_{t_n}\left(P^n, v - P^n\right) \ge 0$$

else, shift the point γ_h^n to the next node on the mesh left/right according to which of the constraints is violated by P^n.

Step 2. Generate simulated stock prices using Black-Scholes Eq. (2.4), which uses the standard normal parameter adaptation of the Brownian motion distribution.

Step 3. Compare the current stock price with the exercise boundary. If the stock price is in the exercise region, then discontinue the path generation since the American option would be exercised. Else, continue to the next time step *t*+1 and go back to Step 2.

For the running time analysis of the entire algorithm, consider two portions individually first. For the Front-Tracking algorithm (Step 1), recall that solving a system of linear equations using LU decomposition has a running time of $O(N^3)$, the running time for solving the system of linear equations is $O(Q^{N^3})$. When analyzing the simulation portion (Step 2 and 3), the stock prices might be searched *N* times for each path, resulting in $O(NM)$. This loop would need to be completed a maximum of *N* times, so the running time for the simulation portion is $O(N^2M)$. For the entire finite difference algorithm, the running time is $O(Q^{N^3} + N^2M)$.

4.3 - Implementation and Computational Experiments

This section presents the implementation of a stochastic mesh and implicit finite difference method, along with experimental results associated with the simulation model. As with the previous chapter, the discrete event-based simulation software, Arena, was used to create the simulated model; however, the Front-Tracking Algorithm from Achdou and Pironneau (2005) was written in C++ code. The Front-Tracking Algorithm was used to calculate the exercise boundary, and then used in the simulation model. The rest of this section shows the experimental results of the model.

4.3.1 - Finite Difference Model

Unlike the previous least-square model, there are two separate programs that need to be run to complete the mesh and finite difference model. First, the Front-Tracking Algorithm from Achdou and Pironneau (2005) needs to be run using a C++ code complier, and then the created discrete event simulation in Arena can simulate the American option. The following paragraphs will describe both the C++ code and simulation model components.

Although the Front-Tracking Algorithm of Achdou and Pironneau (2005) is not used until Step 1 of the mesh method, the code must be run before the simulation model can be started. Since Achdou and Pironneau (2005) provided all coding components for their Front-Tracking Algorithm, the program simply needs to be compiled and ran. The interest rate, stock volatility, and the strike price are the inputs into the code, while the maximum stock price is defined as $200 and the time to expiration is defined over one year. The maximum stock price and time to expiration can also be changed if the user would like. However, the parameters only effect how big of a mesh is created. The maximum stock price needs to be sufficiently above the simulated values to ensure the mesh is large enough, and for the examples it remained at $200. The program is also specifies the time period of one year is broken down into 400 intervals (N) and the stock price is into 200 intervals (M). These parameters can also be modified in the code if the user feels that they cause intervals to be too wide, but for all examples in this section the intervals were kept as the default values.

The first routine of the code creates the stochastic mesh, and then the value of the option is evaluated at each point. The code only allows for American put options to be considered since the payoff value is always equal to the strike price minus the stock price on the grid. Once the

payoff for each point is calculated, the second subroutine determines if the stock price would be exercised at the corresponding time period and stock price. Step 1(c) determines if the American put option would be evaluated, and shifts the function parameter γ to the next grid point according to which constraint is violated. Once the exercise boundary subroutine is finished, the code creates two output files: stochastic mesh and exercise boundary. The Front-Tracking Algorithm is now complete and preparations for the Arena model can then be preformed.

The exercise boundary output from the Front-Tracking Algorithm is saved as a text file, but Arena primarily uses VBA code to interact with Microsoft Excel, not plain text files. Therefore, the input file for the Arena code is an Excel file with the exercise boundary points and corresponding stock prices imported into the first two columns. With the input file completed, the Arena model can be executed.

The first step of the Arena model is to input the system parameters into the appropriate variables. Next, Step 2 is completed for the first time period using either (2.3) or (2.4), since Eq. (2.4) reduces to (2.3) when t is equal to zero. The simulated American option prices were then checked to see if it has reached the exercise boundary yet. The VBA code looks up the exercise boundary price at the current time period, and then determines if the simulated price is above or below exercise boundary price. If the simulated price is less than or equal too the exercise boundary price, then the value of the option is calculated and the path is discontinued. Conversely, if the simulated price is above the exercise boundary price, then the time period is incremented by one and the next stock price is calculated back in Step 2. The loop stops when either the American put option is exercised or reaches the expiration time. Similar to the least-squares Arena model, the value of the option is written to an output file for reporting purposes.

4.3.2 - Computational Results

Using similar inputs as the least-squares Arena model, the Front-Tracking Algorithm followed by the mesh Arena model were tested to show the performance of the method. As Section 4.3.1 explains, only American put options can be tested, since the Front-Tracking Algorithm only computes the payoff for put options. Only evaluating put options is also consistent with the discussion found in Section 2.3. The experiment is given by Example 8, and entails the basic outputs from the method.

4.3.2.1 - Example 8

For the first example using the mesh method, the outputs from the Front-Tracking Algorithm are presented. The American put option has a strike price of $112 and a stock volatility of 0.20, while the risk-free interest rate is 8%. The stock price maximum was kept at $200, and the grid was kept at 200 divisions for the stock price and 400 intervals for the one year time period. After the stochastic mesh is created, the payoff at each point is also computed. Since the option is a put, the payoff is the difference between the strike price and the stock price. Therefore, as the stock price increases, the stock payoff decreases. A portion of the stock payoff grid that is created from the inputs is contained in the figure below.

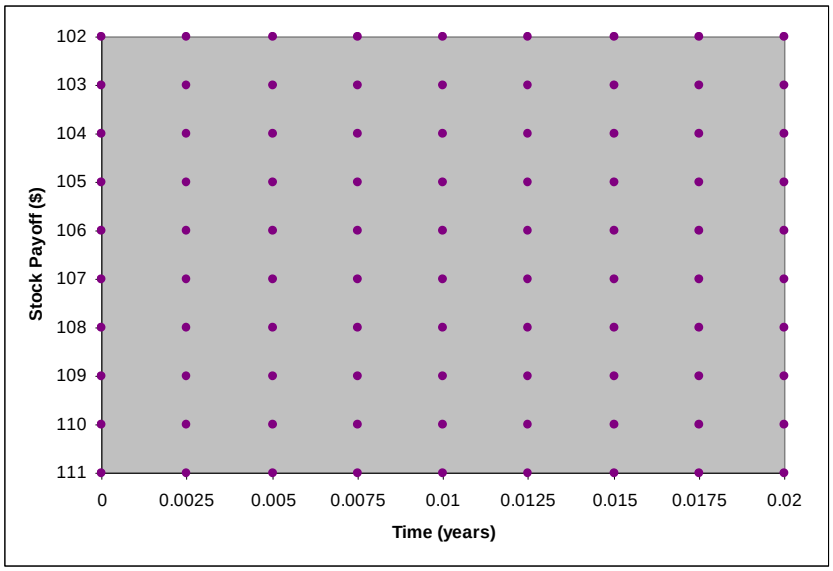

Figure 4.3 - Portion of the Payoff Grid for Example 8

The grid that is generated from the Front-Tracking Algorithm has equally spaced intervals for the stock prices and the time periods, although the length of each interval is different for each axis. With 400 intervals over a one year time period, each interval is 0.0025 years in length, while 200 intervals for $200 gives each interval a length of $1. In Figure 4.3, a portion of the grid from Example 8 is given. The grid is shown for the time period 0 to time period 0.02 and the stock payoff of $102 to $111. The stock payoff axis might appear reversed; however, a $111 payoff value corresponds to $1 stock price value on the grid. Therefore, the lowest stock price value would be $1 and the lowest time period value would be 0, so the payoff of the option at this point (0, 1) would be the $111 shown in Figure 4.3.

In addition to creating the stochastic mesh, the Front-Tracking Algorithm also determines the exercise boundary for the given American put option. The exercise boundary that is created from the inputs for Example 8 is given in Figure 4.4 below. The strike price is the constant red line at $112, while the exercise boundary is given by the blue line. The general shape of the exercise boundary is the same as in Figure 2.1, however the lowest exercise price is much higher than the general picture. The higher exercise price at time zero creates a larger exercise region; therefore, the option in Example 8 would have a higher likelihood of exercising.

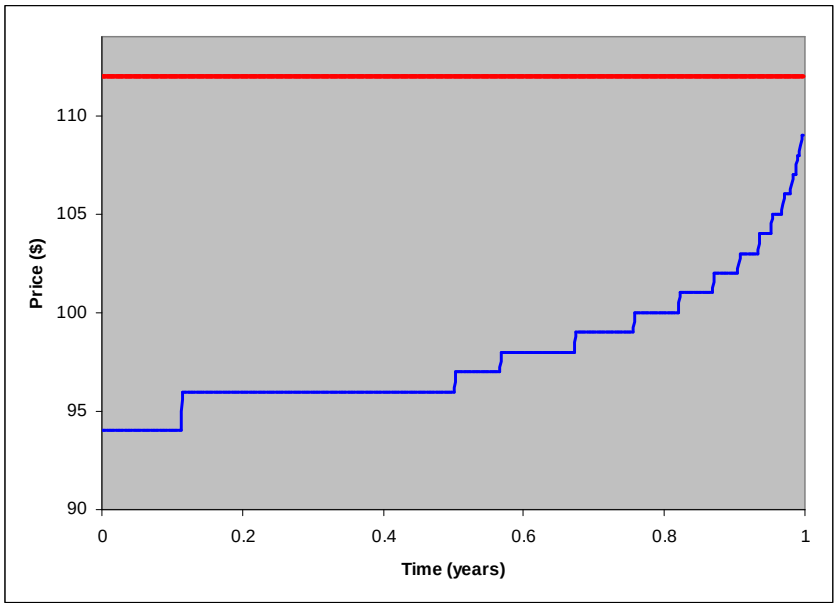

Figure 4.4 - Exercise Boundary for Example 8

Finally, the simulation model can be executed and the corresponding cash flows and value of the American option can be computed. Although 200 paths were simulated using this model, the output for the first 15 paths are given in the following table. Table 4.1 below gives the cash flow value and the exercise time period for the American put option.

Table 4.1 - Finite Difference Method Payoffs

Path #	Cash Flow	Exercise Time Period
1	17.54	2
2	17.08	1
3	24.04	1
4	20.44	1
5	21.00	1
6	0.00	N/A
7	0.00	N/A
8	21.77	2
9	18.72	1
10	0.00	N/A
11	19.74	5
12	17.31	1
13	0.49	5
14	0.00	N/A
15	20.06	4
Overall Average	**12.14**	

The cash flow for each path is equal to the payoff when the option is exercised or zero if the option is never exercised. If the option is exercised the period number is listed in the table, else the exercise time period is not applicable. The overall average in Table 4.1 is the value of the American put option using all 200 paths. Using Eq. (3.7), the value of the option in Example 8 is $12.14.

4.4 - Conclusion

The finite difference method approximates the actual differential equation, compared to finite element which approximates the differential equation's solution. There are two implementations of the finite difference method, which differ by whether the values are generated forwards (implicit) or backwards (explicit). The Front-Tracking algorithm from Achdou and Pironneau (2005) uses the implicit finite difference method to calculate the exercise boundary. The advantage of the implicit method is that it is very robust and always converges to the solution of the differential equation as ΔS and Δt approach zero. The tradeoff for this robust

method is the large memory requirements and extensive computational efforts in solving a system of linear equations in each time period. Since a complete stochastic mesh must be created before the finite difference method can approximate the differential equation, the $(N+1)(Q+1)$ grid must be stored into the computer memory. In addition to the large memory requirements, the finite difference method has extensive computational efforts. The running time analysis resulted in $O(Q^{N^3} + N^2 M)$ for the entire finite difference algorithm. In conclusion, the implicit finite difference method might be robust, but the computer memory required to store the grid and the computational efforts are large obsticals for the algorithm.

CHAPTER 5 - COMPARISON OF COMPUTATIONAL RESULTS

Simulating American options is relatively easy with the current computing power, but determining when to exercise the option can be computationally difficult. In current financial modeling, two theoretical methods of determining when to exercise American options are used; however, a previous comparison has not been completed. Both methods have been theoretically proven by Broadie and Glasserman (2004), Achdou and Pironneau (2005), Longstaff and Schwartz (2001), Chan, et al. (2004), and others, and numerical results are provided in some instances.

In this section, both the least-squares method and the finite difference method programs are run using the same inputs, and the outputs from the system are compared. The first section describes the formal comparison test methods. The next two sections detail two specific comparisons, while the third section reports the results of all ten comparison tests. Finally, the conclusions of comparing between the systems will be discussed.

5.1 - Formal Comparison Tests

For each of the ten comparisons, the least-squares model and the finite difference method model were each run for 30 replications with 200 paths simulated per replication. The value of the American put option given by Eq. (3.7) was collected and averaged over each replication.

The first step of the analysis was to determine if the two systems had equal variances. The null hypothesis is that $\sigma_{\text{LS}}^{\ 2} = \sigma_{\text{FD}}^{\ 2}$ verses the alternative hypothesis $\sigma_{\text{LS}}^{\ 2} \neq \sigma_{\text{FD}}^{\ 2}$ where the legend LS in the depicted tables denotes the least-squares algorithm and the FD represents the finite difference method. The corresponding *F*-test statistic and *p*-value are then calculated. The formal test is then evaluated at 95% confidence level to determine if the variances are significantly different or not.

The second stage compares the means of the two systems using the paired *T*-tests. The null hypothesis is that $\mu_{\text{LS}} - \mu_{\text{FD}} = 0$ verses the alternative hypothesis of $\mu_{\text{LS}} - \mu_{\text{FD}} \neq 0$. The calculated *T*-test statistic and the 95% confidence interval for the difference in means are then reported. As a result of the paired *T*-test, it can be determined if the two methods are statistically

different at a 95% confidence level. If the computational results of the two methods are statistically different, then the higher value of the American put option results represent the better system, since an owner would like to maximize their profits (present values of the payoffs) on the stock market.

5.2 - Comparison with Input 3

The first comparison of the two methods uses the input data from Input 3 in Table 3.1 both models. The average value of the American put options for 30 replications of simulation runs with 200 simulated paths in each replication are used to compare the least-squares method and the finite difference method. The resulting data for each model is given in Table 5.1 at the end of the section.

The first step of the analysis was to determine if the two systems had similar variances. Using the null hypothesis of equal variances verses the alternative of unequal variances, the resulting *F-statistic* was calculated to be 0.72 with a *p*-value of 0.381; therefore, the test fails to reject the null hypothesis that the variances are not significantly different at 95% confidence level.

The second stage compares the means for the results from the two methods via a paired *T*-test. The null hypothesis is the difference in means is equal to zero verses the alternative hypothesis of not equal to zero. The calculated *T-statistic* was calculated to be 2.48 and the 95% confidence interval for the difference in means is (0.0215, 0.2014). As a result, the results from the two methods are statistically different at a 95% confidence level, and the *T*-test shows that the average payoff from the least-squares method is significantly higher than that from the finite difference method.

Table 5.1 - Average Option Value for Comparison 1

Replication	Least-Squares	Finite Difference
1	6.6973	7.0871
2	6.5093	6.4932
3	6.6285	6.4787
4	6.9104	6.6846
5	6.6806	6.6245
6	6.8168	6.5605
7	6.6486	6.6148
8	6.6389	6.4686
9	6.5334	6.6855
10	6.6353	6.2396
11	6.7195	6.5700
12	6.6721	6.5716
13	6.8778	6.4425
14	6.9421	6.3353
15	6.5787	6.6239
16	7.0870	6.8999
17	6.7823	6.5663
18	6.6885	6.4954
19	6.3965	6.5496
20	6.6669	6.4849
21	6.6681	6.4136
22	6.3833	6.6481
23	6.8022	6.1518
24	6.7010	6.4123
25	6.3455	6.8641
26	6.7097	6.6054
27	6.7649	6.4124
28	6.6672	6.6499
29	6.6329	6.7104
30	6.6260	6.7233
Average	**6.6804**	**6.5689**
Reject *F*-Test?	**No**	
Reject *T*-Test?	**Yes**	

5.3 - Comparison with New Parameter Set

The second comparison example uses a new data set as the inputs into the models. The new inputs for the American put option are the following:

- S_0 = \$95
- K = \$100
- σ = 0.10
- r = 0.08
- Δt = 0.00274 year ≈ 1 days
- N = 30

Again, both the least-squares model and the finite difference model were simulated for 30 replications with 200 randomly generated paths for each replication. Also, the values (present value of the payoffs) of the American put option were averaged over each replication for comparison purposes. The resulting simulation outputs are contained in Table 5.2 at the end of this section.

Similar to the previous section, the variances of each system were tested first. The null hypothesis is the variances are equal verses the alternative not equal, just as in Section 5.1. The resulting *F*-test statistic is 4.48 and the *p*-value is less than 0.001. Consequently, the test rejects the null hypothesis concluding that the variances are significantly different at 99% confidence level.

With unequal variances, the paired *T*-test can be preformed to determine if the average payoffs generated form the two models are statistically different. Again, the null hypothesis is the means are equal verses the alternative hypothesis of not equal. The computed *T-statistic* is 69.64 and the corresponding *p*-value is less than 0.001. The 95% confidence interval for the differences in means is (1.1545, 1.2235). Therefore, at 95% confidence level, the two methods are statistically different and the mean (average payoffs) for the finite difference method is significantly lower than the mean from the least-squares method.

Table 5.2 - Average Option Value for Comparison 2

Replication	Least-Squares	Finite Difference
1	6.2320	5.0805
2	6.1684	4.9553
3	6.1366	4.9518
4	6.1397	4.9955
5	6.1595	4.9846
6	6.2565	4.9699
7	6.0866	4.9826
8	6.2986	4.9490
9	6.2445	4.9974
10	6.3015	4.9024
11	6.1545	4.9725
12	6.2302	4.9714
13	6.1249	4.9463
14	6.2783	4.9226
15	6.0305	4.9846
16	6.0913	5.0430
17	6.1195	4.9718
18	6.1775	4.9576
19	6.1216	4.9690
20	6.1559	4.9526
21	6.2659	4.9385
22	6.0876	4.9895
23	6.1156	4.8836
24	6.1463	4.9390
25	5.9997	5.0354
26	6.1298	4.9788
27	6.2414	4.9371
28	6.2571	4.9888
29	6.0870	5.0034
30	5.9916	5.0059
Average	**6.1610**	**4.9720**
Reject *F*-Test?	**Yes**	
Reject *T*-Test?	**Yes**	

5.4 - Results of the Ten Comparisons

Following the same procedure presented in Section 5.2 and Section 5.3, the remaining ten comparison tests were preformed. Each test consisted of running both the least-squares method and finite difference method for 30 replications of 200 paths each. As stated in Section 5.1, an *F*-test is first preformed to compare the variances, and then a paired *T*-test is conducted to test the difference in means. The null hypotheses for each test are given below.

***F*-Test**	***T*-Test**
H_0: $\sigma_{LS}^2 = \sigma_{FD}^2$	H_0: $\mu_{LS} - \mu_{FD} = 0$
vs. H_a: $\sigma_{LS}^2 \neq \sigma_{FD}^2$	vs. H_a: $\mu_{LS} - \mu_{FD} \neq 0$

Each of the ten pairs of comparisons has a different set of inputs, corresponding to the six parameters required for each method. Table 5.3 below summarizes the six input parameters, the *T-statistic*, and the conclusion on whether there exists a difference in means at a 95% confidence level. If the paired *T*-test rejects the null hypothesis at 95% confidence level, then the system which produced the higher average value for the American put option is also reported.

Table 5.3 - Results of All 10 Comparisons

Input	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
S_0	$100	$100	$125	$125	$95	$173	$173	$158	$125	$95
K	$112	$112	$135	$135	$100	$180	$165	$170	$175	$80
σ	.20	.20	.05	.05	.10	.30	.30	.35	.15	.25
Δt	.2000	.0192	.2000	.0192	.00274	.0025	.0025	.015	.01	.01
N	5	13	5	13	30	30	14	20	85	85
r	.08	.08	.08	.08	.08	.05	.05	.05	.05	.10
T-stat.	11.54	49.52	2.48	15.20	69.64	139.19	8.90	69.47	24.14	19.09
Reject?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Higher	LS	LS	LS	LS	LS	LS	LS	LS	LS	LS

5.5 - Comparison Summary

Following the formal tests given in Section 5.1, each of the ten comparison test runs were first tested for equal variances and then a paired *T*-test was preformed to test for the difference in

means between the least-squares and finite difference methods. As the formal comparisons show, all ten comparisons result in a significant difference in means at 95% confidence. For zero out of the ten test comparisons, the least-squares method produces higher American option value. For an owner, a higher option value relates to a higher cash flow payoff. Even though none of the ten comparison cases proved the finite difference method produces higher values, the least-squares is not guaranteed to produce a better option value. However, in general the least-squares method produces a higher average American put option value than the finite difference method.

CHAPTER 6 - COMPARISON OF MODELS

Presently, two theoretical methods for solving the optimal exercise problem of American options exist, but a numerical comparison between the models has previously not been preformed. In this section, several factors will be used to compare the two methods. First, the results from the ten comparison tests will be analyzed. Next, the ease of implementation and the degree each system is streamlined are evaluated. Finally, how fast each program executes and the efficiency of memory storage for each system is compared.

6.1 - Formal Statistical Tests

As Chapter 5 shows, the least-squares program and the finite difference method produce significantly different average value of the American put option. From all ten comparisons, the least-squares method was found to have a higher average value for an American put option value than the finite difference method. Therefore, an investor would be willing to pay a higher initial price to buy the option from using the least-squares method program. Although, the least-squares method is not guaranteed to produce a better option value. Thus, the two methods produce statistically different American put option values; however, in general the least-squares method produces better values for American put options.

6.2 - Ease of Implementation

In addition, the two methods of determining the optimal exercise time also differ in ease of implementation. Since both programs were created using the discrete simulation software Arena, neither program involved low and complicated levels programming to simulate the stock. However, the finite difference method program created for this research did not require low levels of programming since the exercise boundary was already created by the program published in Achdou and Pironneau (2005). To program the entire finite difference method algorithm would require solving the Q-1 simultaneous equations by L U decomposition, matrix inversion, or other means. These methods are not generally accomplished using a higher levels of programming; therefore, to create a self contained implementation that simulates the entire algorithm would require a much lower level of programming.

Compared to the finite difference method, the least-squares model requires a higher level of programming to complete the entire algorithm. The most complex programming component of the least-squares method is solving the least-squares equation, found in Step 5 of the algorithm (Section 3.2). The regression coefficients are solved from matrix computations: multiplication, inversion, and transpose. These matrix computations do require some higher knowledge of any programming language, but solving a system of linear equations requires an even higher level of programming. Hence, the finite difference method is easier to implement than the finite difference method.

6.3 - Degree Streamlined

After the algorithms are implemented, the degree to which each program is streamlined can be examined. In the least-squares model, seven parameter inputs need to be inserted into the model, along with the number of paths to be simulated. Then the model can run to completion.

Conversely, the finite difference method program has two sets of inputs, three for the Front-Tracking Algorithm and six for the simulation model. After the inputs are placed into the programs, the exercise boundary program runs to completion. The resulting exercise boundary is then also an input into the simulation model, and the second program can finally be run to completion.

Since the least-squares model requires fewer inputs and only executes a single program, the least-squares program is more streamlined for the user than the finite difference method double programs.

6.4 - Running Time Analysis

Another difference between the models is the amount of time to execute each program. A running time analysis for the least-squares method is discussed in Section 3.2, while the presentation of the running time analysis for the finite difference method is found in Section 4.2. The running time for the entire least-squares method is $O(NM)$, and the running time for the finite difference method is $O(Q^{N^3} + N^2 M)$, where N denotes the number of time periods, M is the number of paths simulated, and Q represents the number of stock price intervals.

Comparing the running times for the two proposed methods, the worse case scenario for the finite difference method grows much faster than the worse case for the least-squares method.

For an investor who is looking for real-time information, the least-squares model has the potential to provide more timely information when the number of time periods N, or the number of paths simulated M is large.

6.5 - Efficiency of Memory Storage

Lastly, the efficiency of memory storage for each model can be explored. For this discussion, only the required storage will be discussed. The storage used to collect statistics and information not vital for the program to run to completion not will examined. Many approaches for simulating American options are very costly in terms of computer memory requirements since the entire path needs to be generated and stored before the optimal exercise point can be determined. Hence, the memory storage is the number of time periods times the number of paths, or $N \times M$.

However, the least-squares model generates the paths backwards. It does require the seed values for the random number streams to be stored and the standard normal random numbers to be generated twice, but the backward generation eliminates the need to generate and store all the simulated paths. The memory required for the least-squares model is the number of paths M that are going to be simulated. Also, the least-squares regression requires both the current stock price and the next period's cash flows to be stored, resulting in $2M$ computer memory. All other calculated values can be replaced when the program transitions between time periods. The total computer memory required for the least-squares method is $3M$. Therefore, the least-squares method requires far less memory storage than other methods with forward generated paths to simulate American options.

Unlike the least-squares model, the finite difference method requires the entire stochastic mesh to be stored into memory for the calculations of the exercise boundary. So, the computer memory used to store the stochastic mesh is equal to the number of time periods points $N+1$ multiplied by the number of stock price points $Q+1$, i.e. $(N+1)(Q+1)$. Although the simulation does not require the entire path to be stored, the entire mesh used to create the exercise boundary still requires a considerable amount of memory. Also, the exercise boundary must be stored in order to use it as an input into the simulation; therefore, another $(N+1)$ of computer memory is required. The total memory required for the finite difference method is $(N+1)(Q+2)$.

Even when the number of paths is moderately large, the least-squares method generally requires less memory than the finite difference method. In the examples presented here, the number of time periods and stock prices that defined the grid was constant, 200 and 400 respectively. So, the memory required for the finite difference method is (201)(402) = 80,802 double precision floating point variables. The memory required for the least-squares method was generally 200 paths which transpires to 200 integer variables. The total computer memory required for the least-squares method is 3(200) = 600. Therefore, the least-squares model requires far less memory (0.74%) to execute than the finite difference method.

CHAPTER 7 - CONCLUSION

In conclusion, the results of comparing the proposed least-squares algorithm and the finite difference method are summarized, along with remarks regarding future research.

7.1 - Final Remarks

From Chapter 5, there are significant differences in the value of the American put options produced by each method. Using the ten comparison examples, the proposed least-squares method generally produced a higher value for the American put option. Hence, the least-squares method produced a better estimate for the American put option value than the finite difference method for most cases.

In addition to producing a better estimate for the value of American put options, there are several other factors to consider. The least-squares method is easier to implement and is a more streamlined process than the finite difference method. Also, the least-squares algorithm running time is generally much faster than the finite difference method. Finally, the least-squares method only requires M amount of memory storage, compared to the $(N+1)(Q+1)$ memory required for the finite difference method. Overall, the least-squares method is faster, more concise, easier to implement, and requires less computer memory than the finite difference method.

In conclusion, the proposed least-squares method produces a better estimate of the American put option value at a faster speed, with an easier implementation, and requiring less computer memory. Therefore, the least-squares method is superior to the finite difference method for solving the optimal exercise problem for American put options.

7.2 - Future Research

There are two areas in which the research presented in this paper can be extended. The first involves the relationship equation used in the least-squares regression. Currently, a quadratic relationship is used; however, a higher order relationship or linear combination relationship might be more appropriate. Also, the proposed least-squares algorithm includes all paths, including the paths in which the cash flow is zero. Including these paths results in the regression equation depended on the zero values, but the actual relationship might be better

represented if these paths were not included in the linear regression computations. The second area of extension involves the parameters used in the calculations for the value of continuing. The next time period's cash flow is used in the proposed least-squares algorithm, and the maximum of all future cash flows is used in the algorithm presented in Chan et al. (2004). A better estimation of the value of an American put option might be obtained using a combination of the two values or even the stock prices, instead of the cash flows. Therefore, the relationship equation and the parameters used in the least-squares algorithm are possible future research areas.

CHAPTER 8 - REFERENCES

Achdou, Yves, and Olivier Pironneau. 2004. "A Numerical Procedure for the Calibration of the Volatility with American Options." *Applied Mathematical Finance* 12(3).

Achdou, Yves, and Olivier Pironneau. 2005. *Computational Methods for Option Pricing*. Philadelphia: Society for Industrial and Applied Mathematics.

Allegretto, Walter, Yanping Lin, and Hongtao Yang. 2001. "A Finite Element Method for Pricing American Put Options on Zero-Coupon Bonds." Paper presented at European Financial Management Association 2001 Lugano Meetings in Lugano, Switzerland.

Allegretto, Walter, Yanping Lin, and Hongtao Yang. 2003. "Numerical Pricing of American Put Options on Zero-Coupon Bonds." *Applied Numerical Mathematics* 46(2): 113-134.

AitSahlia, Farid, and Peter Carr. 1997. "American Options: A Comparison of Numerical Methods." In *Numerical Methods in Finance*, ed. L. C. G. Rogers and D. Talay, 67-87. Cambridge, United Kingdom: Cambridge University Press.

Belomestny, Denis and Grigori Milstein. 2006. "Adaptive Simulation Algoithms for Pricing American and Bermudan Options by Local Analysis of Financial Market." Paper presented at SFB 649 Discussion Paper 2006-038, in Berlin, Germany.

Black, Fischer, and Myron Scholes. 1973. "The Pricing of Options and Corporate Liabilities." *Journal of Political Economy* 81(3): 637-654.

Brennan, Michael and Eduardo Schwartz. 1976. "The Valuation of American Put Options." *Journal of Finance* 32(2): 449-462.

Brennan, Michael, and Eduardo Schwartz. 1977. "Finite Difference Methods and Jump Processes Arising in the Pricing of Contingent Claims: A Synthesis." *Journal of Financial and Quantitative Analysis* 13: 461-474.

Broadie, Mark, Paul Glasserman, and Zachary Ha. 2000. "Pricing American Options by Simulation Using a Stochastic Mesh with Optimized Weights." In *Probabilistic Constrained Optimization: Methodology and Applications*, Ed. S. Uryasev, 32-50. Norwell, MA: Kluwer Academic Publishers.

Broadie, Mark, and Paul Glasserman. 2004. "A Stochastic Mesh Method for Pricing High-Dimensional American Options." *Journal of Computational Finance*. 7(4): 35-72.

Chang, Raymond, Yong Chen, and Kit-Ming. Yeung. 2003. “A Memory Reduction Method in Pricing American Options.” *Journal of Statistical Computation and Simulation* 74(7): 501-511.

Chang, Raymond, Chi-Yan Wong, and Kit-Ming. Yeung. 2002. “Pricing American-Style Options by Monte Carlo Methods without Storing all the Intermediate Asset Prices.” Chinese University of Hong Kong Mathematics Department Research Report #2002-11.

Chang, Raymond, Chi-Yan Wong, and Kit-Ming Yeung. 2006. “Pricing Multi-Asset American-Style Options by Memory Reduction Monte Carlo Methods.” *Applied Mathematics and Computation* 179(2): 535-544.

Coleman, Thomas, Yuying Li, and Arun Verma. 2002. “A Newton Method for American Option Pricing.” *Journal of Computational Finance* 5(3): 51-78.

Cont, Rama and Ekaterina Voltchkova. 2005. “A Finite Difference Scheme for Option Pricing in Jump Diffusion and Exponential Levy Models.” *SIAM Journal of Numerical Analysis* 49(4): 1596-1626.

Courant, Richard. 1943. “Variational Methods for the Solution of Problems of Equilibrium and Vibrations.” *Bulletin of the American Mathematical Society* 46: 1-23.

Forsyth, Peter and Kenneth Vetzal. 2001. “Quadratic Convergence for Valuing American Options Using a Penalty Method.” *SIAM Journal of Scientific Computing*. 23(6): 2095-2122.

Glasserman, Paul. 2003. *Monte Carlo Methods in Financial Engineering*. New York: Springer-Verlag.

Glasserman, Paul and Bin Yu. 2004. “Simulation for American Options: Regression Now or Regression Later?” In *Proceedings of the Monte Carlo and Quasi-Monte Carlo Methods*, Ed. H. Niederreiter, 213-226. Berlin: Springer-Verlag.

Haugh, Martin. 2004. “Approximate Dynamic Programming (ADP) and Dual Methods for Pricing American Options.” http://www.columbia.edu/~mh2078/MCS04/MCS_american.pdf (accessed October 16, 2007).

Huang, Jacqueline and Jong-Shi Pang. 1998. “Option Pricing and Linear Complementarily.” *Journal of Computational Finance*. 2(3): 1-31.

Hull, John. 2006. *Options, Futures, and Other Derivatives*. Upper Saddle River, NJ: Pearson Education, Inc. 6th Ed.

Hull, John and Alan White. 1990. "Valuing Derivative Securities Using the Explicit Finite Difference Method." *Journal of Financial and Quantitative Analysis* 25: 87-100.

Ikonen, Samuli and Jari Toivanen. 2004. *Pricing American Options Using LU Decomposition*. Technical Report B 4/2004, Department of Mathematical Information Technology, University of Jyväsklä.

Indragoby, Govindaraj, and Olivier Pironneau. 2004. "Calibration for Option Pricing with Automatic Differentiation." Rapport de recherché LJLL-UPMC.

Jaillet, Patrick, Damien Lamberton, and Bernard Lapeyer. 1990. "Variational Inequalities and the Pricing of American Options." *Acta Applicandae Mathematicae* 21: 263-289.

Kerman, Jouni. 2002. "Numerical Methods for Option Pricing: Binomial and Finite-difference Approximations." Diss., Courant Institute of Mathematical Sciences, New York University.

Longstaff, Francis, and Eduardo Schwartz. 2001. "Valuing American Options by Simulation: A Simple Least-Squares Approach." *The Review of Financial Studies* 14(1): 113-147.

McLeish, Don. 2005. *Monte Carlo Simulation and Finance*. Hoboken, NJ: John Wiely and Sons, Inc.

Merton, Robert. 1973. "The Theory of Rational Option Pricing." *Bell Journal of Economics and Management Science* 4(1): 141-183.

Oosterlee, Cornelis, Coenraad Leentvaar, and Xinzheng Huang. 2004. "Pricing Options with Discrete Dividends by High Order Finite Differences and Grid Stretching." Paper presented at the European Congress on Computational Methods in Applied Sciences and Engineering, July 24-28, in Jyväskylä, Finland.

Papatheodorou, T, M Koulisianis, and P Hadjidoukas. 2000. "Numerical Methods for the American Option Valuation Problem." Paper presented at the 16th IMACS World Congress on Scientific Computation, Applied Mathematics, and Simulation, August 21-25, in Lausaunne, Switzerland.

Persson, Jonas. 2006. "Accurate Finite Difference Methods for Option Pricing." *Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology*. PhD diss., Uppsala University.

Persson, Jonas. 2007. “Pricing American Options Using a Space-Time Adaptive Finite Difference Method.” In Technical Report 2007-004 from Department of Information Technology, Uppsala University.

Pooley, David. 2003. “Numerical Methods for Nonlinear Equations in Option Pricing.” PhD diss., University of Waterloo.

Rockwell Automation . Arena Simulation Software. http://www.arenasimulation.com (accessed December 11, 2007).

Ross, Sheldon. 2003. *Introduction to Probability Models*. San Diego: Academic Press. 8th Ed.

Shu, Jiwu, Yonggeng Gu, Xiaotie Deng, and Weimin Zheng. 2005. “A Sliced-Finite Difference Method for the American Option.” *IIE Transactions* 37: 939-944.

Tilley, James. 1993. “Valuing American Options in a Path Simulation Model.” *Transactions of the Society of Actuaries* 45: 83-104.

Tkitsiklis, John and Benjamin Van Roy. 2001. “Regression Methods for Pricing Complex American-Style Options.” *IEEE Transactions on Neural Networks* 12(4): 694-703.

Wilmott, Paul. 1998. *Derivatives: The Theory and Pricing of Financial Engineering*. Chichester: Wiley.

Zhang, Chensong. 2005. “Pricing American Options by Adaptive Finite Element Method.” Paper presented at the Spotlight on Graduate Research Competition, December 15, at University of Maryland.

Zhao, Jichao, Robert Corless, and Matt Davison. 2007. “Compact Finite Difference Method for American Option Pricing.” *Journal of Computational and Applied Mathematics* 206(1): 306-321.