THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

DEPARTMENTS OF MATHEMATICS AND FINANCE

PRICING METHODS FOR AMERICAN PUT OPTIONS

LAUREN VALDIVIA
SPRING 2014

A thesis
submitted in partial fulfillment
of the requirements
for baccalaureate degrees in Mathematics and Finance
with honors in Mathematics and Finance

Reviewed and approved* by the following:

Leonid Vaserstein
Professor of Mathematics
Thesis Supervisor

James Miles
Professor of Finance
Honors Adviser

Victoria Sadovskaya
Associate Professor of Mathematics
Honors Adviser

*Signatures are on file in the Schreyer Honors College.

ABSTRACT

Options valuation is an area of active research in financial mathematics. Fisher Black and Myron Scholes published the most popular closed-form solution for vanilla European options in their 1973 paper, "The Pricing of Options and Corporate Liabilities." The Black-Scholes equation still accurately prices vanilla European options and is still widely used. However, there is no closed-form solution to the pricing of American put options. As such, several numerical methods have been proposed to approximate the price of American put options. The purpose of this paper is to analyze these methods in terms of computational speed and flexibility in adaptation to non-vanilla options. In particular, this paper will focus on the binomial asset pricing model, finite difference schemes, and Monte Carlo methods.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

Options valuation is an ongoing area of research in financial mathematics. An option is a contract that gives the buyer the right, but not the obligation, to buy or sell an asset at a specified strike price on or before a specified date. The seller or writer of the option has the obligation to fulfill the transaction should the owner choose to "exercise" the option. Because this contract holds protection against downside risk for the stock, it holds value and the buyer must pay the seller a premium to enter this obligation.

Several different types of options are currently traded. A "call" option conveys the right of the owner to buy an asset a specified price, known as the "strike" price. A "put" option conveys the right to sell an asset. "European" options can only be exercised at a specified date, the "exercise" or "expiry" date. "American" options, however, may be exercised at any time on or before that date. Other, more complex options, such as the Bermudan or Asian options, have further constraints on payoff or execution.

The owner of a European call will only choose to exercise the option when the agreed upon strike price is less than the stock price at expiry. This would allow them to buy the stock from options contract for the strike price and sell the stock in the market for a greater price, garnering an instant gain. Should the strike price be less than the market stock price, the owner of the option would simply choose to let the option expire. Thus, the owner's payoff at time of expiry is known to be the following as shown in [4]:

$$max[0, S - K],$$

where

S is the price of the underlying stock at the expiry date, and

K is the strike price of the option.

Because the owner of the European call pays some premium to the seller of the contract, the owner's maximum profit is

$$S - K - C(S, t),$$

where

$C(S,t)$ is the price paid for the call option as a function of stock price, S
and time until expiration, t.

Similarly, the owner's minimum profit (maximum loss) is

$$0 - C(S,t).$$

The payoff for a European put can be derived in a similar manner. If the strike price is greater than the market stock price at exercise, the owner of the contract will buy the stock from the market for stock price S and sell the stock to the seller of the contact for the higher value of K. If the market stock price is less than the strike price, the owner will simply let the contract expire. Thus, the owner's payoff at time of expiry is known to be

$$max[0, K - S].$$

Because the owner of the European put pays some premium to the seller of the contract, the owner's maximum profit, as stated in [4] is

$$K - S - P(S,t)$$

where

$P(S,t)$ is the price paid for the put option.

Similarly, the owner's minimum profit (maximum loss) is

$$0 - P(S,t).$$

Puts for which the strike price is greater than the market price are said to be "in-the-money" options. If the strike price is equal to the market stock price, they are known as "at-the-money" options. Puts for which the strike price is less than the market price are known as "out-of-the-money" options.

Clearly, the structure provides protection against declines in the stock price as it provides a boundary for the maximum loss that is less than the value of the actual stock. The primary challenge in options theory is to determine how much should someone be willing to pay for this contract.

Although options contracts have been traded for centuries and options valuation has been studied since at least the nineteenth century, trading activity and academic interest increased when, in 1973, options were issued with standardized terms and traded with guaranteed terms through the Chicago Board Options Exchange.

Following the opening of the CBOE in 1973, Fisher Black and Myron Scholes made one of the most important contributions to modern finance theory with the publication of their paper, "The Pricing of Options and Corporate Liabilities." The paper gave a theoretical estimate of the price of European options, leading to a boom in options trading.

Their Black-Scholes equation is a partial differential equation that describes the price of a call over over time. The equation, as proved by [1] is

$$\frac{\partial f}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 f}{\partial S^2} + rS\frac{\partial f}{\partial S} - rf = 0, \tag{1}$$

where

S is the price of the underlying stock,

$f(S,t)$ is the price of the option as a function of time and stock price,

σ is the volatility of the stock's returns,

t is the time in years (now = 0, expiry = T), and

r is the annualized risk-free interest rate, continuously compounded.

Solving the equation at the terminal and boundary conditions gives the closed-form Black-Scholes formula proved in [1] for the price of a European Call, denoted as $C(S,t)$:

$$C(S,t) = N(d_1)S - N(d_2)Ke^{-rt},$$

where

$$\begin{aligned} d_1 &= \tfrac{1}{\sigma\sqrt{t}}[\ln(\tfrac{S}{K}) + (r + \tfrac{\sigma^2}{2})t], \\ d_2 &= \tfrac{1}{\sigma\sqrt{t}}[\ln(\tfrac{S}{K}) + (r - \tfrac{\sigma^2}{2})t], \\ &= d_1 - \sigma\sqrt{t}, \end{aligned}$$

and $N(*)$ is the cumulative normal distribution.

The formula above can easily be manipulated to solve for the price of a European put, denoted as $P(S,t)$, via the Put-Call Parity, which relates the values of a European call, put, and the underlying stock such that

$$C(t) - P(t) = S(t) - Ke^{-rt}.$$

From this relation, it is easily seen that

$$P(S,t) = N(-d_2)Ke^{-rt} - N(-d_1)S,$$

as shown in [4].

For American options, however, the Black-Scholes equation becomes the inequality:

$$\frac{\partial f}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 f}{\partial S^2} + rS\frac{\partial f}{\partial S} - rf \leq 0,$$

as stated in [1].

This inequality does not have a closed-form solution. It has been shown that it is never optimal to exercise an American call with no dividends before the expiry date. Therefore, an American call without dividends can be priced using the Black-Scholes formula. This does not hold true, however, for American put options [8]. Although the majority of traded options are American options, their valuation remains a topic of active research. Due to the lack of a closed form solution, a large variety of approximation schemes have been utilized to price American put options. In this paper, I will describe and analyze several different tree, finite difference, and Monte Carlo pricing methods to identify the optimal methods in terms of computational speed, convergence, and flexibility of implementation.

Chapter 2

The Binomial Asset Pricing Method

J. Cox, S. Ross, and M. Rubinstein were the first to formulate the binomial asset pricing model in their 1979 paper, "The Valuation of American Put Options." The methodology begins with the assumption that the stock price follows a multiplicative binomial process over discrete time periods. The rate of return on the stock over each period can have two possible values: $u-1$ with probability q or $d-1$ with probability $1-q$. As q is a probability, it is necessary that $0 \leq q \leq 1$. Thus, if the current stock price is S, the stock price at the end of one period will be either uS or dS as shown by [3]:

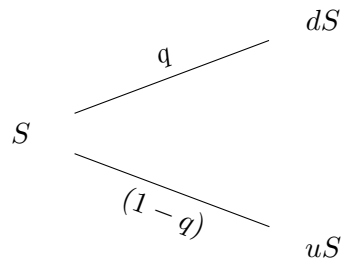

Figure 1. One period binomial tree for S

S is the stock price today.

q is the probability of a price increase.

u is the factor by which the price increases.

d is the factor by which the price decreases.

The interest rate is assumed constant with r denoting 1 plus the riskless interest rate over one period with the requirement that $u > r > d$. There are no taxes, transaction costs, or margin requirements. Any security may be shorted. First, a call on a stock with one period left to expiration is considered. Let C be the current value of the call, C_u be its value at the end of the period if the stock price rises to uS, and C_d be the value at the end of the period if the stock price drops to dS as shown in the tree from [8] below:

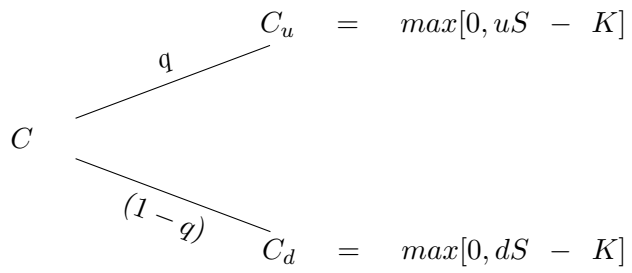

Figure 2. One period binomial tree for C

The arbitrage pricing theory is used by replicating the option by trading in the stock and money market instruments. A portfolio is formed containing Δ shares of stock and B in riskless bonds for a total cost of $\Delta S + B$. At the end of the period, the value of the portfolio will be the following as demonstrated in [8]:

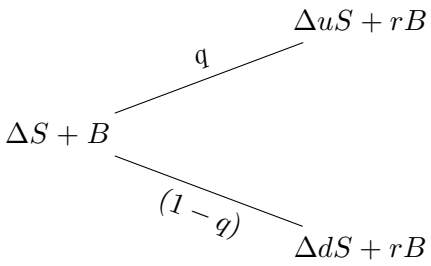

Figure 3. Arbitrage pricing theory tree

By selecting Δ and B so that the value of the portfolio and the call are equal at the end of the period, [3] shows:

$$\Delta uS + rB = C_u \text{ and } \Delta dS + rB = C_d.$$

Cox, Ross, and Rubinstein solved the equations below in [3], showing:

$$\Delta = \tfrac{C_u - C_d}{(u-d)S} \text{ and } B = \tfrac{uC_d - dC_u}{(u-d)r}.$$

Assuming there are no riskless arbitrage opportunities, the current value of the call must be equal to the current value of the portfolio. It can then be assumed as in [3] that:

$$C = \Delta S + B = \frac{C_u - C_d}{u-d} + \frac{uC_d - dC_u}{(u-d)r} = \frac{\frac{r-d}{u-d}C_u + \frac{u-r}{u-d}C_d}{r}$$

if the value is greater than $S - K$. If the value of the call is not greater than $S - K$,

$$C = S - K.$$

The equation was simplified by defining $p = \frac{r-d}{u-d}$ and $1 - p = \frac{u-r}{u-d}$ so that the formula for a call becomes:

$$C = \frac{pC_u + (1-p)C_d}{r}.$$

Considering the next simplest situation, a call with two periods remaining before expiration, the tree for the stock value is:

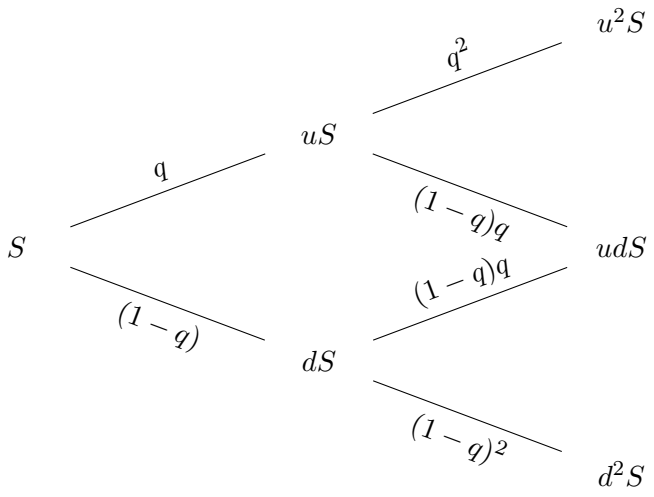

Figure 4. Two period binomial tree for S

and the call's payoffs can be modeled with the tree,

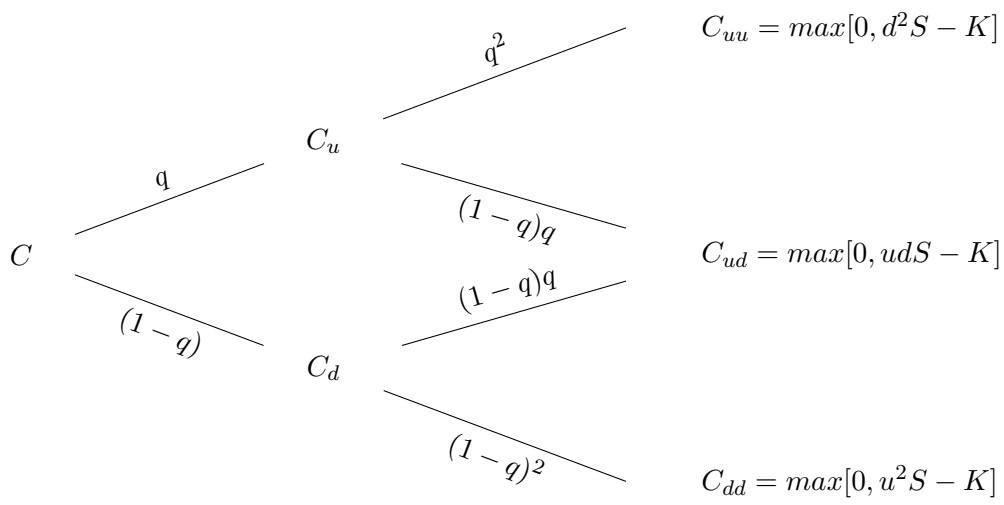

Figure 5. Two period binomial tree for C

At one period left, $C_u = \frac{pC_{uu}+(1-p)C_{ud}}{r}$ and $C_d = \frac{pC_{du}+(1-p)C_{dd}}{r}$.

Using substitutions and noting that $C_{du} = C_{ud}$, the formula for the value of a European call becomes:

$$C = \frac{p^2C_{uu}+2p(1-p)C_{ud}+(1-p)^2C_{dd}}{r^2}$$
$$= \frac{p^2max[0,u^2S-K]+2p(1-p)max[0,duS-K]+(1-p)^2max[0,d^2S-K]}{r^2}.$$

as shown in [3].

For any n time steps:

$$C = \frac{\sum_{j=0}^{n} \binom{n}{j} p^j(1-p)^{n-j}max[0,u^jd^n-jS-K]}{r^n}.$$

The same analysis can be used for pricing puts. Letting P denote the current price of a put with one period to expiration, the binomial tree is:

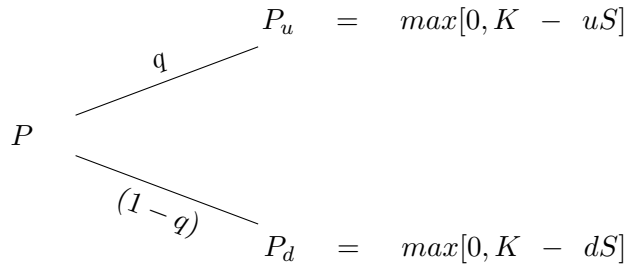

Figure 6. One period binomial tree for P

The price of the put is $P = max[\frac{pP_u+(1-p)P_d}{r}, K - S]$.

Note that discrete time can be exchanged for continuous time by substituting all incidences of r^n with e^{rn} [3]. The remainder of this thesis will operate in continuous time.

Any values can be used for the u and d with the constraint, $u > r > d$. Some values, however, are more optimal than others and lead to faster convergence. Cox, Ross, and Rubinstein offer the most common convention for choosing u, d, and p. Jarrow and Rudd also provide a well known convention for choosing the parameters via their equal-probability model [5].

Cox-Ross-Rubinstein Convention

Cox, Ross, and Rubinstein used the convention in [3] that:

$$\text{p} = \frac{e^{rt}-d}{u-d},\ d = 1/u, \text{ and } u = e^{\sigma\sqrt{(\frac{t}{n})}}.$$

Jarrow-Rudd Convention

Jarrow and Rudd suggest a different choice for the parameters. They utilize the convention in [5]:

$$p = \tfrac{1}{2},\ u = e^{r-\frac{1}{2}\sigma^2 t+\sigma\sqrt{t}}, \text{ and } d = e^{r-\frac{1}{2}\sigma^2 t-\sigma\sqrt{t}}.$$

Chapter 3

Finite Difference Methods

Finite difference methods are useful tools in mathematics for approximating the solutions to differential equations. These tools can be used to price options by approximating the derivatives in the continuous-time Black-Scholes-Merton differential equation that describes how an option price changes over time by a set of discrete-time difference equations. The discrete difference equations are then solved iteratively to calculate a price for the option.

The following finite difference methods use the following lattice of potential future prices of the underlying asset:

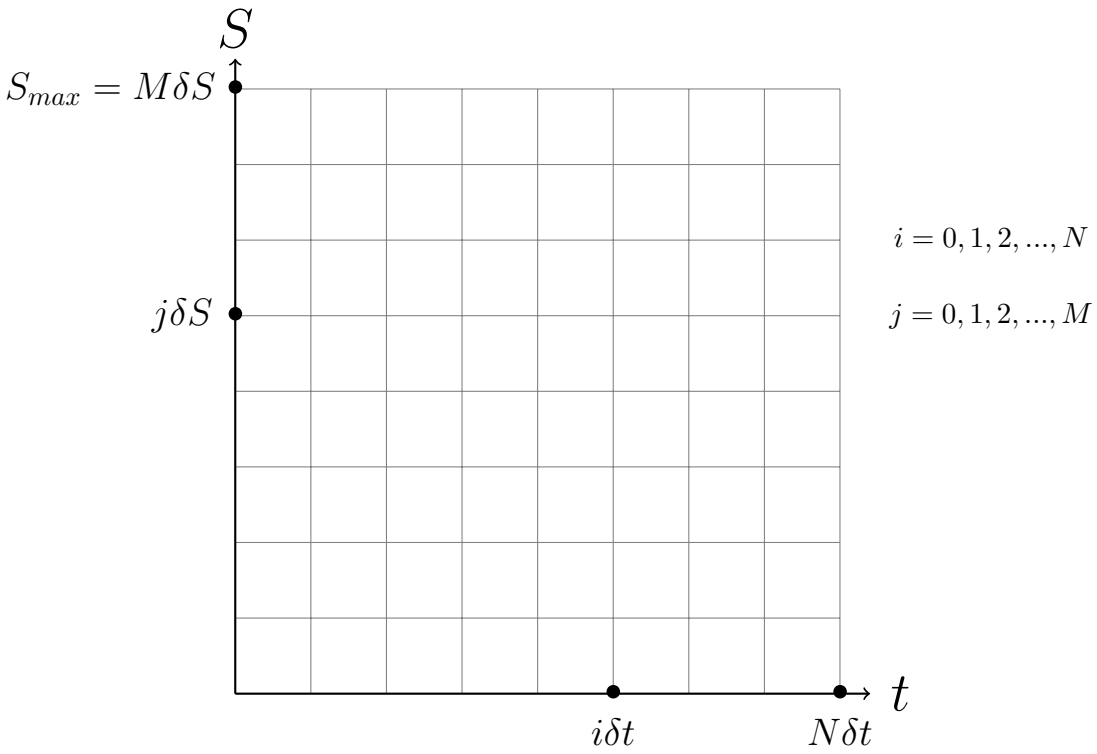

Figure 7. Option pricing lattice

The lattice is generated by discretizing the time between today and expiry into M equal time steps and the underlying stock price S into N equal steps. The boundary condition for the time is the expiration date located on the grid at time $N\delta t$. A boundary condition, $S_{max} = M\delta S$, is chosen such that the current market

price is near the middle of the grid axis. This leads to the above grid with $M + 1$ time points and $N + 1$ price points. The interior nodes represent the price of the option at time $j\delta t$ and stock price $i\delta S$. If the value of the option is known at the boundary conditions, especially at expiry, the value for all interior nodes can be calculated iteratively. For American options, the payoff at each node is compared to the intrinsic value of the option that would be gained by holding the option to maturity. The final goal of the following methods is to find the price at time 0 [4].

Explicit Euler Scheme

E. Schwartz provided a numerical solution for valuing options on dividend-paying stocks. In addition, he derived the optimal strategy for exercising American options [2]. Again, revisiting Equation (1) as shown by Black and Scholes, the differential equation governing the value of the option between dividend payment dates becomes the following difference equation as seen as [1]:

$$\frac{\delta f}{\delta t} + rS\frac{\delta f}{\delta S} + \frac{1}{2}\sigma^2 S^2 \frac{\delta^2 f}{\delta S^2} = rf.$$

The backward difference approximation is used for $\frac{\delta f}{\delta t}$:

$$\frac{\delta f}{\delta t} = \frac{f_{i,j} - f_{i-1,j}}{\delta t}.$$

To obtain a better approximation of $\frac{\delta f}{\delta S}$, Schwartz used the average between the forward difference and the backward difference, known as the central approximation,

$$\frac{\delta f}{\delta S} = \frac{f_{i,j+1} - f_{i,j-1}}{2\delta S}.$$

A standard approximation is used for $\frac{\delta^2 f}{\delta S^2}$:

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

Substituting these values into Equation (1), we obtain:

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

$$a_j = \frac{1}{2}\delta t(\sigma^2 j^2 - rj)$$

$$b_j = 1 - \delta t(\sigma^2 j^2 + r)$$

$$c_j = \frac{1}{2}\delta t(\sigma^2 j^2 + rj)$$

as seen in [7].

The Explicit Euler Method can be visualized by the following lattice as shown in [9]:

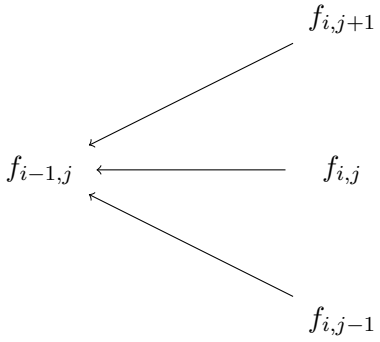

Figure 8. Explicit Euler lattice

Implicit Euler Scheme

The implicit method operates in a similar manner as the explicit method, but it uses a forward approximation for $\frac{\delta f}{\delta t}$ in [10]:

$$\frac{\delta f}{\delta t} = \frac{f_{i+1,j} - f_{i,j}}{\delta t}.$$

As in the explicit Euler method, the implicit method uses a central approximation for $\frac{\delta f}{\delta S}$ and a standard approximation for $\frac{\delta^2 f}{\delta S^2}$. From [10], substituting these approximations into Equation (1) gives:

$$f_{i+1,j} = a_j^* f_{i,j-1} + b_j^* f_{i,j} + c_j^* f_{i,j+1}$$
$$a_j^* = \tfrac{1}{2}\delta t(rj - \sigma^2 j^2 j)$$
$$b_j^* = 1 + \delta t(\sigma^2 j^2 + r)$$
$$c_j^* = \tfrac{1}{2}\delta t(-\sigma^2 j^2 - rj)$$

The Implicit Euler Method can be visualized by the following lattice as demonstrated in [9]:

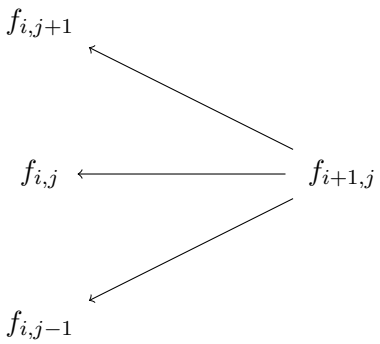

Figure 9. Implicit Euler Lattice

The lattice shows that working backward in time, three unknowns must be calculated from only one known value, $f_{i+1,j}$. When all the equations at a given time point are written simultaneously, there are $M-1$ equations in $M-1$ unknowns. Hence, the value for f at each node can be calculated [8].

Crank-Nicolson Scheme

The Crank - Nicolson finite difference method represents an average of the implicit and explicit method. Instead of using a backward approximation for $\frac{\delta f}{\delta t}$ as in the implicit and explicit methods, it uses a central approximation:

$$\frac{\delta f_{i-1/2,j}}{\delta t} = \frac{f_{i,j}-f_{i-1,j}}{\delta t} + O(\delta t^2).$$

It uses a central approximation for $\frac{\delta f}{\delta S}$:

$$\frac{\delta f_{i-1/2,j}}{\delta S} = \frac{\delta f_{i-1/2,j}}{\delta S} = \frac{1}{2}\Big[\frac{f_{i-1,j}}{\delta S} + \frac{\delta f_{i,j}}{\delta S}\Big]$$
$$= \frac{1}{2}\Big[\frac{f_{i-1,j+1}-f_{i-1,j-1}}{2\delta S} + \frac{f_{i,j+1}-f_{i,j-1}}{2\delta S}\Big] + O(\delta S^2).$$

A standard approximation is used for $\frac{\delta^2 f}{\delta S^2}$:

$$\frac{\delta^2 f_{i-1/2,j}}{\delta S^2} = \frac{1}{2}\Big[\frac{\delta^2 f_{i-1,j}}{\delta S^2} + \frac{\delta^2 f_{i,j}}{\delta S^2}\Big]$$
$$=\frac{1}{2}\Big[\frac{f_{i-1,j+1}-2f_{i-1,j}+fi-1,j-1}{\delta S^2} + \frac{f_{i,j+1}-2f_{i,j}+fi,j-1}{\delta S^2}\Big] + O(\delta S^2)$$

as shown in [10].

Substituting these approximations into Equation (1) reduces it to the following from [10]:

$$-\bar{a_j} f_{i-1,j-1} + (1-\bar{b_j}) f_{i-1,j} - \bar{c_j} f_{i-1,j+1} = \bar{a_j} f_{i,j-1} + (1+\bar{b_j}) f_{i,j} + \bar{c_j} f_{i,j+1}$$
$$\bar{a_j} = \tfrac{\delta t}{4}(\sigma^2 j^2 - rj)$$
$$\bar{b_j} = -\tfrac{\delta t}{2}(\sigma^2 j^2 + r)$$
$$\bar{c_j} = \tfrac{\delta t}{4}(\sigma^2 j^2 + rj)$$

The Crank-Nicolson Method can be visualized as the following lattice from [9]:

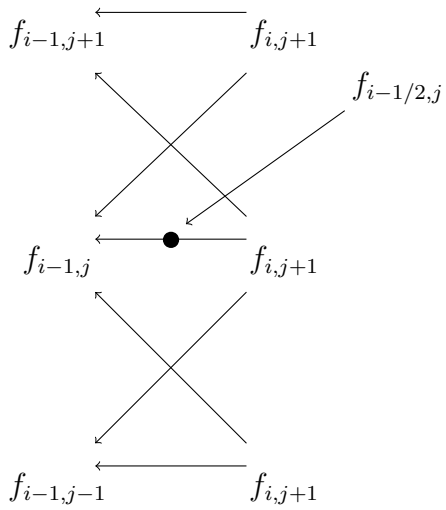

Figure 10. Crank-Nicolson lattice

The Crank-Nicolson method prices all three of the left side nodes of Figure 10 based on the values of the three right side nodes. When the difference equation is written for all values of i, it leads to a set of $M-1$ equations with $M-1$ unknowns. Hence, the value for f at each node can be calculated by solving this system of equations.

Chapter 4

Monte Carlo Methods

Another commonly used tool in options pricing is Monte Carlo simulation. Monte Carlo simulations generate a large number of random possible price paths for the underlying asset via simulation. The methods then calculate the associated payoff of the option for each price path and discount the payoff to present, giving the value of the option [6].

Least Squares Method

One of the most popular Monte Carlo simulations is the Least Squares Method, as suggested by Francis Longstaff and Eduardo Schwartz in 2001 [6]. The method begins by simulating all potential stock price paths for the underlying asset.

For each path, the approach discretizes the interval into time steps:

$$t_0 = 0, t_1 = \frac{T}{N}, t_2 = 2\frac{T}{N}, ..., t_N = T.$$

The optimal stoping policy is analyzed at each exercise date t_k in a path. In practice, American options are continuously exercisable and the LSM algorithm can be used to approximate the value by taking a sufficiently large N [6].

At each exercise time point in a path, option holders compare the payoff for immediate exercise with the expected payoff if they continued to hold the option. If the payoff for immediate exercise is higher, then they exercise the options. Otherwise, they will continue to hold the options. The expected payoff for continuation is conditional on the information available at that time step [6].

The LSM approach uses least squares to approximate this conditional expectation at $t_{N-1}, t_{N-2}, ..., t_1$. Once the conditional expectation function at time t_{N-1} is estimated, it can be determined whether early exercise at time t_{N-1} is optimal for "in-the-money" stock price paths. Once the exercise decision is identified, the option cash flow paths can then be approximated. The recursion then proceeds back to time t_{N-2} and repeats the procedure until the exercise decision at each time along the path has been determined. The American option is then valued by starting at time 0 and moving forward along each path until the first

stopping time occurs, discounting the cash flow from exercise back to 0, and taking the average of all possible paths [6].

Chapter 5

Analysis of Methods

The strength of a method, as evaluated in this paper, will depend on three main factors: speed of convergence, computational speed, and flexibility in adapting the method to other types of options.

Convergence

Below is a table of an "in-the-money" American put options prices solved using the 6 methods. The option was based on the following assumptions: the initial market price of the underlying asset was $50, the strike price was $52, volatility was 20%, time until expiration was 6 months, and the risk free interest was 1%. The methods were executed according to the MATLAB codes in Appendix A for 50 to 1000 time steps.

#Steps	Cox-Ross-Rubinstein	Jarrow-Rudd	Explicit	Implicit	Crank-Nicolson	LSM
50	3.83875160632631	3.84726069835730	3.84377615442609	3.82643096984480	3.83467144551772	3.99489875939005
100	3.85505808523183	3.85523095688777	3.85074756600852	3.84196258547927	3.84621140989956	3.99226331028885
150	3.85169289497368	3.84701013131968	3.84377615442609	3.83862173773549	3.84158494843894	4.23262980137297
200	3.84640606840612	3.84986923357919	3.84869389535543	3.84405619020964	3.84630191382340	4.10042216825743
250	3.84936828162124	3.85118014721803	3.84941504394197	3.84570229191665	3.84750848053154	4.16811880264316
300	3.85059832756608	3.84991507889147	3.84825605779171	3.84514558164352	3.84666494415969	3.87632164080351
350	3.85023852642592	3.84766877059214	3.84642900550802	3.84369972081095	3.84502826484124	4.06813450582395
400	3.84912624704810	3.84837696705458	3.84817692356075	3.84578232556754	3.84694634606534	3.89735918605097
450	3.84767573045003	3.84954901489488	3.84884319584083	3.84671730881887	3.84775462064074	3.74502586173402
500	3.84817502338039	3.84974600165473	3.84880780101859	3.84689397305967	3.84783033824438	3.98912790648058
550	3.84908550261524	3.84933580551359	3.84834292599961	3.84660111145239	3.84745517396499	4.10045607181261
600	3.84943855777898	3.84855014124744	3.84762130496055	3.84601476864477	3.84680338956144	3.88358306554478
650	3.84941314331118	3.84755427624181	3.84771849652152	3.84621343575205	3.84694559880015	3.82579151058907
700	3.84912255686826	3.84839556706877	3.84833123424629	3.84693666836307	3.84761703177683	4.15574856289967
750	3.84865192046633	3.84895590275214	3.84861101316335	3.84731129190983	3.84794708067739	3.95532623797887
800	3.84806155938239	3.84917059741765	3.84864504935349	3.84742697607103	3.84802415530884	3.85531380382262
850	3.84792968476761	3.84912551600990	3.84849628889884	3.84734926847631	3.84791258649144	3.82040069105163
900	3.84847643943449	3.84888603637499	3.84821161144504	3.84712691190937	3.84766043214653	4.00297627705022
950	3.84880916923662	3.84849915789042	3.84782864574437	3.84679648223174	3.84730409182349	3.93611695159565
1000	3.84897106415889	3.84800790635033	3.84785941760112	3.84686882589521	3.84735240142405	4.05933326714502

Table 1. American put prices by method

As shown, the binomial tree and finite difference methods were far superior to the least squares method in speed of convergence.

Computational Speed

The methods were also analyzed according to speed of computation for the same "in-the-money" put option with initial market price of the underlying asset of $50, strike price of $52, volatility of 20%, time until expiration of 6 months, and the risk free interest rate of 1%. The following table of computational speeds (in seconds) was generated from the MATLAB codes in Appendix A.

# Steps	Cox-Ross-Rubinstein	Jarrow-Rudd	Explicit	Implicit	Crank-Nicolson	LSM
1	0.000251	0.000209	0.000124	0.000828	0.000138	NA
10	0.000579	0.001514	0.000126	0.037895	0.000185	0.047674
100	0.005290	0.004481	0.002392	0.139840	0.005816	0.119086
1000	0.092882	0.089921	0.280340	9.047076	0.651630	1.053513
10000	4.481435	3.769200	27.410012	903.010911	120.083392	171.872077

Table 2. Computational Speed (seconds) by method

Clearly, the binomial methods have a far faster computational time than all other methods. The Implicit Euler method stands out as having the slowest computation time. In this method, the inverse of a matrix must be calculated, which is an expensive calculation for a computer to perform.

Flexibility

Dividends

One of the main tests of flexibility of the aforementioned methods will be how well they handle the dividend payments on the underlying stock.

As lattice methods, the binomial tree methods and the finite difference methods can easily handle dividend payments. When a dividend is payed, the underlying stock price must be reduced by the dividend amount at the node at which it was payed.

The least squares method can similarly be adjusted to account for annual dividend payments. As the simulation spawns potential paths of stock prices, these paths can be adjusted to account for the percentage of stock paid out as dividends.

Exotic Options

Another concern when evaluating the usefulness of the methods, is how well they adapt to options with more complicated payoff structures. To examine this criterion, consider a special type of option contract, the Asian option.

Asian options are one of the most basic forms of exotic options. Their payoff is determined by the average underlying price over some pre-set period of time.

Conceptually, lattice-based methods like binomial tree methods and finite difference methods can easily adapt to this exotic payoff. At each node of the payoff tree, the payoff simply depends on the average of the stock price along the branch. However, as the number of nodes on a tree grows, so does the number of averages which must be taken, especially in the central nodes. The number of averages taken is exponentially related to the number of possible asset prices. Thus, computationally, pricing Asian options via lattice-based methods is very inefficient.

Due to Asian option's dependence on the entire path of the underlying, Asian options are particularly suited to Monte Carlo simulation. Since many possible paths are generated in the least squares method, it is easy to compute payoffs that are dependent on the path of the underlying.

Chapter 6

Conclusion

Each of the binomial, finite difference, and least squares methods were able to value American options with reasonable accuracy with a relatively fast computational speed. At a smaller number of steps, the binomial tree methods and the Explicit and Crank-Nicolson methods had comparable computation speeds. The Implicit Method was comparable to that of the Least Squares method at smaller number of steps. As the number of time steps increased, the binomial methods exhibited a greater computational speed than the other methods. The implicit method revealed itself has having the slowest computation speed.

As shown, all the lattice methods had a clear advantage in the number of steps required until the price converged to a value. The Least Squares method had the worst convergence rate. In fact, even at 1000 time steps, the price given by the method was still fluctuating by about $0.15.

While the Least Squares method ranked lowest in terms of speed and accuracy, it is the most flexible method. By nature, Monte Carlo simulations are easily adjusted to handle dividends and exotic payoff functions. As such, because Monte Carlo simulations are less accurate and slower than other numerical methods, the simulations are best suited as a last resort method when payoff functions are too complicated to compute efficiently using other numerical methods.

For future research, it would be useful to compare the Cox-Ross-Rubinstein and Jarrow-Rudd conventions to other conventions for choosing the parameters in the binomial methods. For example, the Tian method matches the first three moments of the binomial model to the first three moments of a lognormal distribution. The Leisen-Reimer method relic on approximating the normal distribution used in the Black-Scholes model.

Appendix A

MATLAB Code

Below is the MATLAB code for the methods detailed in this thesis. Code was written with reference to software available on P. Goddard's site [9].

Cox-Ross-Rubinstein Binomial Method

```
function oPrice = BinomialTreeCRR(X,S0,r,sig,T,N,oType,earlyExercise)
% Calculate the Cox Ross Rubinstein model parameters
dt = T/N;
a = exp(r*dt);
u = exp(sig*sqrt(dt));
d = 1/u;
p = (a-d)/(u-d);

    % Loop over each node and calculate the Cox Ross Rubinstein underlying price tree priceTree =
nan(N+1,N+1);
priceTree(1,1) = S0;
for idx = 2:N+1
    priceTree(1:idx-1,idx) = priceTree(1:idx-1,idx-1)*u;
    priceTree(idx,idx) = priceTree(idx-1,idx-1)*d;
end

    % Calculate the value at expiry
valueTree = nan(size(priceTree));
switch oType
    case 'PUT'
        valueTree(:,end) = max(X-priceTree(:,end),0);
    case 'CALL'
        valueTree(:,end) = max(priceTree(:,end)-X,0);
end

    % Loop backwards to get values at the earlier times N = size(priceTree,2)-1;
for idx = N:-1:1
    valueTree(1:idx,idx) = ...
    exp(-r*dt)*(p*valueTree(1:idx,idx+1) ...
    + (1-p)*valueTree(2:idx+1,idx+1));
    if earlyExercise
        switch oType
            case 'PUT'
```

```
                    valueTree(1:idx,idx) = ...
                    max(X-priceTree(1:idx,idx),valueTree(1:idx,idx));
                case 'CALL'
                    valueTree(1:idx,idx) = ...
                    max(priceTree(1:idx,idx)-X,valueTree(1:idx,idx));
            end
        end
end

        % Output the option price
oPrice = valueTree(1);
end
```

Jarrow-Rudd Binomial Method

```
function oPrice = BinomialTreeJR(X,S0,r,sig,T,N,oType,earlyExercise)
dt = T/N;
p = 0.5;
u = exp((r-sig*sig/2)*dt + sig*sqrt(dt));
d = exp((r-sig*sig/2)*dt - sig*sqrt(dt));

    % Loop over each node and calculate the JR underlying price tree
priceTree = nan(N+1,N+1);
priceTree(1,1) = S0;
for idx = 2:N+1
    priceTree(1:idx-1,idx) = priceTree(1:idx-1,idx-1)*u;
    priceTree(idx,idx) = priceTree(idx-1,idx-1)*d;
end

    % Calculate the value at expiry
valueTree = nan(size(priceTree));
switch oType
    case 'PUT'
        valueTree(:,end) = max(X-priceTree(:,end),0);
    case 'CALL'
        valueTree(:,end) = max(priceTree(:,end)-X,0);
end

    % Loop backwards to get values at the earlier times
N = size(priceTree,2)-1;
for idx = N:-1:1
    valueTree(1:idx,idx) = ...
    exp(-r*dt)*(p*valueTree(1:idx,idx+1) ...
    + (1-p)*valueTree(2:idx+1,idx+1));
    if earlyExercise
        switch oType
            case 'PUT'
                valueTree(1:idx,idx) = ...
                max(X-priceTree(1:idx,idx),valueTree(1:idx,idx));
            case 'CALL'
                valueTree(1:idx,idx) = ...
                max(priceTree(1:idx,idx)-X,valueTree(1:idx,idx));
        end
    end
end
% Output the option price
oPrice = valueTree(1);
end
```

Explicit Euler Method

```
function AmerPutPrice= ExplicitEuler(T, Spot, K, v, r,q, N, M)
dt = T/N; % Time increment
mu = r - q - v2/2; % Drift for stock process
dx = v*sqrt(3*dt); % Increment for stock price

pu = dt*(v2/2/dx2 + mu/2/dx); % Up probability
pm = 1 - dt*v2/dx2 - r*dt; % Middle probability
pd = dt*(v2/2/dx2 - mu/2/dx); % Down probability

S = zeros(2*M+1, N+1); % Initialize stock price
V = zeros(2*M+1, N+1); % Initialize option price

I = [0:1:N]; % Indices for time step
J = [M:-1:-M]'; % Indices for stock price step

% Stock price at maturity
S = Spot*exp(J.*dx);

      % Call price at maturity
V(:,end) = max(K - S, 0);

      % Work backwards through the lattice
for j=N:-1:1
      for i=2:2*M
            V(i,j) = pu*V(i-1,j+1) + pm*V(i,j+1) + pd*V(i+1,j+1);
      end
      % Lower boundary
      V(2*M+1,j) = V(2*M,j) + (S(2*M) - S(2*M+1));

      % Upper boundary
      V(1,j) = V(2,j);

      % Early exercise check
      for i=1:2*M+1
            V(i,j) = max(K - S(i), V(i,j));
      end
end

      AmerPutPrice = V(M+1,1);
end
```

Implicit Euler Method

```
function AmerPutPrice = ImplicitEuler(T,Spot,K,v,r,q,PutCall,EuroAmer, N,M)
dt = T/N; % Time increment
mu = r - q - v2/2; % Drift for stock process
dx = v*sqrt(3*dt); % Increment for stock price

    % Probabilities.
pu = -dt/2*(v2/dx2 + mu/dx); % Up probability
pm = 1 + dt*v2/dx2 + r*dt; % Middle probability
pd = -dt/2*(v2/dx2 - mu/dx); % Down probability

    % Initialize stock price and option value.
S = zeros(2*M+1, N+1);
V = zeros(2*M+1, N+1);

    % Indices for stock price step.
J = [M:-1:-M]';

    % Stock price at maturity
S = Spot*exp(J.*dx);
clear J

    % Option price at maturity is intrinsic value.
if strcmp(PutCall,'P')
    V(:,end) = max(K - S, 0);
else
    V(:,end) = max(S - K, 0);
end

    % Upper and lower boundaries for the grid.
if strcmp(PutCall,'P')
    lambda_L = max(0,K - S(2*M+1));
    lambda_U = 0;
elseif strcmp(PutCall,'C')
    lambda_L = 0;
    lambda_U = max(0,S(1) - K);
end

    for j=N:-1:1
    % Create the 'C' vector.
    C = [lambda_U; V(2:2*M,j+1); lambda_L];
    % Solve the triangular system of equations.
    V(:,j) = SolveTriangular(C,pu,pm,pd,lambda_L,lambda_U);
    % Apply early exercise to American options.
    if strcmp(EuroAmer,'A')
```

```
            for i=1:2*M+1;
                switch PutCall
                    case 'P'
                    V(i,j) = max(V(i,j), K - S(i)); % Puts.
                case 'C'
                    V(i,j) = max(V(i,j), S(i) - K); % Calls.
                end
            end
        end
end
clear i j

        % Output the price and compare to the trinomial tree.
AmerPutPrice = V(M+1,1)
end
```

Crank-Nicolson Method

```
function Put = CrankNicolson(T,Spot,K,v,r,q,N,M)
dt = T/N;
mu = r-q-v2^/2;
dx = v*sqrt(3*dt);

        pu = -1/4*dt*(v2^/dx2^ + mu/dx);
pm = 1+ dt*v2^/2/dx2^+r*dt/2;
pd = -1/4*dt*(v2^/dx2^-mu/dx);

        S = zeros(2*M+1,N+1);
V = zeros(2*M+1,N+1);

        J = [M:-1:-M]';
S = Spot*exp(J.*dx);
V(:,end) = max(K-S,0);

        pmp = zeros(2*M+1,N+1);
pp = zeros(2*M+1,N+1);
C = zeros(2*M+1,N+1);

        for j = N+1:-1:2
        pmp(2*M,j) = pd+pm;
        for i =2*M-1:-1:2
                pmp(i,j) = pm-pd/pmp(i+1,j)*pu;
        end
end

        lambda_L = S(2*M+1) - S(2*M);
lambda_U = 0;

        for j=N+1:-1:2
        for i =2*M:-1:2
                if i==2*M
                        pp(i,j) = -pu*V(i-1,j)-(pm-2)*V(i,j) - pd*V(i+1,j) +pd*lambda_L;
                else
                        pp(i,j) = -pu*V(i-1,j)-(pm-2)*V(i,j)- pd*V(i+1,j)-pd/pmp(i+1,j)*pp(i+1,j);
                end
        end
        j=j-1;
        for i=1:2*M+1
                if i==1
                        C(i,j) = (pp(i+1,j+1) + pmp(i+1,j+1)*lambda_U) / (pmp(i+1,j+1) + pu);
                        V(i,j) = max(K - S(i), C(i,j));
                elseif i<2*M+1
```

```
                C(i,j) = (pp(i,j+1) - pu*C(i-1,j))/pmp(i,j+1);
                V(i,j) = max(K - S(i), C(i,j));
            else
                C(i,j) = C(i-1,j) - lambda_L;
                V(i,j) = max(K - S(i), C(i,j));
            end
        end
        j=j+1;
end

% Output the value matrix and the price of the American put
Put = V(M+1,1);
end
```

Least Squares Method

```
function [Price,CF,S,t] = LeastSquares(S0,K,r,T,sigma,N,M,type)
% S0 Initial asset price
% K Strike Price
% r Interest rate
% T Time to maturity of option
% sigma Volatility of underlying asset
% N Number of points in time grid to use (minimum is 3, default is 50)
% M Number of points in asset price grid to use (minimum is 3, default is 50)
% type True (default) for a put, false for a call
dt = T/N;
t = 0:dt:T;
t = repmat(t',1,M);

    R = exp((r-sigma^2/2)*dt+sigma*sqrt(dt)*randn(N,M));
S = cumprod([S0*ones(1,M); R]);

    ExTime = (M+1)*ones(N,1);

    CF = zeros(size(S)); % Cash flow matrix

    CF(end,:) = max(K-S(end,:),0); % Option only pays off if it is in the money

    for ii = size(S)-1:-1:2
    if type
        Idx = find(S(ii,:) < K); % Find paths that are in the money at time ii
    else
        Idx = find(S(ii,:) > K); % Find paths that are in the money at time ii
    end
    X = S(ii,Idx)'; X1 = X/S0;
    Y = CF(ii+1,Idx)'*exp(-r*dt); % Discounted cashflow
    R = [ ones(size(X1)) (1-X1) 1/2*(2-4*X1-X1.^2)];
    a = R\Y; % Linear regression step
    C = R*a; % Cash flows as predicted by the model
    if type
        Jdx = max(K-X,0) > C; % Immediate exercise better than predicted cashflow
    else
        Jdx = max(X-K,0) > C; % Immediate exercise better than predicted cashflow
    end
    nIdx = setdiff((1:M),Idx(Jdx));
    CF(ii,Idx(Jdx)) = max(K-X(Jdx),0);
    ExTime(Idx(Jdx)) = ii;
    CF(ii,nIdx) = exp(-r*dt)*CF(ii+1,nIdx);
end
```

```
        Price = mean(CF(2,:))*exp(-r*dt);
end
```

BIBLIOGRAPHY

[1] F. Black, M. Scholes, The Pricing of Options and Corporate Liabilities, *Journal of Political Economy*, **81**, 637-654 (1973).

[2] M. Brennan, E. Schwartz, The Valuation of American Put Options, *The Journal of Finance*, **32**, 449-462 (1977).

[3] J. Cox, S. Ross, M. Rubinstein, Option Pricing: A Simplified Approach, *Journal of Financial Economics*, **7**, 229-263 (1979).

[4] J. Hull, *Futures and Options Markets, 8th edition*, Pearson, 2013.

[5] R. Jarrow, A. Rudd, Approximate Option Valuation for Arbitrary Stochastic Processes, *Journal of Financial Economics*, **10**, 347-369 (1982).

[6] F. Longstaff, E. Schwartz, Valuing American Options by Simulation: A Simple Least-Squares Approach, *The Review of Financial Studies*, **14**, 113-147 (2001).

[7] E. Schwartz, The Valuation of Warrants: Implementing a New Approach, *Journal of Financial Economics*, **4**, 79-93 (1977).

[8] S. Shreve, *Stochastic Calculus for Finance I: The Binomial Asset Pricing Model*, Springer, 2004.

[9] P. Turner, *Guide to Scientific Computing, 2nd edition*, CRC Press, 2000.

[10] P. Goddard, Financial Engineering: Option Pricing, http://www.goddardconsulting.ca/financial-engineering.html.

LAUREN N. VALDIVIA

Cell: 717-348-6376 • lnv103@psu.edu • 215 Poplar Road • Lewistown, PA 17044

EDUCATION

Pennsylvania State University, University Park, PA — Expected **May 2014**
Schreyer Honors College, Smeal College of Business, Eberly College of Science

- Dual Majors in Finance and Mathematics (Systems Analysis)
- Core competencies include advanced quantitative analytical skills and mathematics (matrix theory, linear programming, and statistics)
- Basic programming experience in MATLAB, VBA, and C++
- Minor in Spanish
- Dean's List Fall 2010, Spring 2011, Fall 2011, Spring 2012, Fall 2012, Spring 2013, Fall 2014

Study Abroad, University of the Balearic Islands, Palma de Mallorca, Spain — **Summer 2012**

- Home stay experience with a Spanish family
- Two courses on the use of Spanish language and culture in business and tourism
- Extensive field work conducting studies of the tourism market in Palma

Lewistown Area Senior High School, Lewistown, PA — **June 2010**

- Class Valedictorian
- GPA: 4.00/4.00
- National Advanced Placement (AP) Scholar

EMPLOYMENT

PNC Bank, Pittsburgh, PA — **June 2013 – August 2013**
Summer Analyst – Asset Backed Finance Group

- Learned and implemented complex securitization concepts in a 10-week period
- Structured a borrowing base model to support a $200 million securitization deal that generated $2 million in first year revenue

Penergy Solutions, Huntingdon, PA — **August 2011 – Present**
Financial Planning Assistant

- Coauthored strategic business plan and five-year financial projections for start-up company
- Updated financials and reports for the company

Penn State Tennis Center**, University Park, PA* ***January 2011 – Present
Tennis Instructor – Professional Tennis Registry (PTR) Certified

- Taught clinics to players of all ages and ability levels
- Coached students in private lessons

SPECIAL PROJECTS

Thesis Research — **September 2013 - Present**
Pricing American Put Options

- Analyzed numerical methods for approximating the price of American put options
- Utilized Matlab and C++ to conduct of a comparison of Monte Carlo, Finite Difference, and Tree techniques for approximating the Black Scholes pricing model

LEADERSHIP

Penn State Club Tennis September 2010 – Present
Travel Team Captain

- Directed team selections and practices
- Organized travel arrangements and tournament entries