Financial Options: History, Computational Analysis, and Future Significance

Corey Maxedon

Indiana State University

Acknowledgements

This research would not have been possible without Dr. Vin Isaia, a mathematics professors at Indiana State University. He helped me in developing a love for computer programming. I would have been lost in the mathematical theory behind this topic without him.

I would also like to thank Dr. Reza Houston, a finance professor at Ball State University, for supporting my financial research in this topic. Dr. Houston directed my research on the history and background of derivatives.

Abstract

This paper is directed toward a mathematical audience and looks at options, or derivatives, from a computational standpoint using computer programming with Python. A history of two options pricing formulas is given: the binomial pricing model and Black-Scholes' formula (Cox, Ross, & Rubinstein, 1979; Black & Scholes, 1973). The two formulas are compared against one another to see the benefits of each. The Black-Scholes model is by far the superior way of pricing options. The binomial pricing model has a limit of Black-Scholes formula. The Black-Scholes model is tested against real world data, but significant error is found. The error can give an implied volatility for the stock, which indicates current stock volatility rather than historic. Black-Scholes can also use the error to find arbitrage opportunities in the market. I created a new options trading strategy using Python that gives a list of ranked arbitrage opportunities for all options in an option chain. The program lets the user know the optimal strike price to buy and sell put and call options in order to benefit on the arbitrage opportunities at hand. The future for options trading and pricing has not lost its significance since finding the solution 45 years ago.

Keywords: Black-Scholes, option pricing, arbitrage, computer programming

Financial Options: History, Computational Analysis, and Future Significance

"In our view, however, derivatives are financial weapons of mass destruction, carrying dangers that, while now latent, are potentially lethal."

-Warren Buffett, Berkshire Hathaway Annual Report, 2002

Introduction

This paper is a computational analysis of options and dives into financial derivative pricing using computer programming with Python. The history of derivatives and option theory is included. Several definitions are given so no prior knowledge is assumed with the target audience of the mathematics community. The bulk of the methods and analysis will include two option-pricing models: the binomial pricing model and Black-Scholes formula. These mathematically challenging procedures are coded into a computer program in Python in order to price real options in the market.

A binomial pricing tree is used to price options of stock. Equations are given to produce this value. Python was used to speed up the calculation process, which is lengthy if the tree's number of nodes is scaled up. When the number of nodes in the pricing tree is pushed to infinity, Black-Scholes formula falls out in the form of a limit. This formula is discussed briefly with an overview of the derivation. Then, the binomial pricing tree is compared to the Black-Scholes continuous pricing formula. In theory, both formulas should produce the same answer for the value of the option.

The optimal formula is used to test the error between option value in today's market for options. The formula's output may or may not be equal to the real world value. The implications of this are discussed in the analysis section. The paper will end with a discussion on the current market fluctuations and its affect on the value of options. Since options are used as an insurance vehicle, one would expect the volume of options purchased to be higher in times of economic instability. The future of options may also be different than it is now.

The Great Unknown

Much of the formulas and papers written on options were published in the '60s and '70s. I need to learn what the option formulas are and see if one has a benefit over the other. Finding the most efficient (least computationally time consuming formula with the least error from real intrinsic price) way to price these options using Python. It was not clear what the optimal formula is when programming pricing models. It was unknown to me whether these formulas still have real world significance. It was interesting to see the error in what Black and Scholes' formula has from the real world, and why this error may occur or what benefits can come from it. The current market volatility could hurt the potential validity of the output of the formula. Also, an easy, intuitive way to price a large sets of option prices in order to find arbitrage opportunities has never been created.

Area of Interest. Options are well known and the formulas have been established, but it is unclear whether they still correlate to the actual market performance of options. In order to study option pricing, I must learn how these pricing models were created. Once the optimal method is deduced, a comparison between real and intrinsic values for options is studied. Due to this uncertainty, I will test the accuracy based on real world data from the market and compare it

to the output of the computer programs I built that use the formulas for pricing options. With this information, I hope to produce a trading strategy based on any arbitrage opportunities at hand.

I seek to learn more about options and how they are priced. I want to know how these formulas were found and if they are still relevant today. Once I find their relevance, I can project this on the future of the options market and create a user-friendly trading strategy for real investors to utilize in their trading activity.

Background

Defining Stock

A **stock** is a part of a company sold by the company in order to raise money for growth and expansion. A person who buys the stock is called a **shareholder** (Wilmott, Howison, & Dewynne, 1995). Thus, shareholders own a part, or share, of the company and have voting rights in company ventures. When a person buys stock, they have a feeling the company might be more valuable in the future than it is currently. Therefore, the stock might go up if it was more valuable than the current stock price suggests. The purchaser could then sell their stock to another buyer and make money on the difference between buying price and selling price.

Defining Option Jargon

An **option**, or derivative, gives a buyer the *option* to buy or sell an underlying stock, the **underlying**, at a later date, or the **expiry date** (Hull, 1997; Saunders & Cornett, 2019). Only stock options are analyzed in this paper, but currency and commodities, such as corn or oranges, could also be used as the underlying. In fact, several international companies and banks use options on currency to **hedge**, or remove risk from, currency exchange risk between different

countries (Hull & White, 1987; Kim & Kim, 2015). Stock options are bought and sold. When the option to sell the underlying is purchased, this is called a **put**. When the option to buy the underlying is purchased, this is called a **call**. The option has a premium, expiry date, and a strike price. The **strike price** is the price at **expiry**, or end time, a person would like to buy or sell the underlying stock for regardless of actual stock price. The **expiry date** is the date the option expires, and the buyer must make a decision to engage on the option or decline (Bodie, Kane, & Marcus, 2018; Saunders & Cornett, 2019; Hull, 1997; Wilmott, Howison, & Dewynne, 1995).

American options are options that have the ability to be executed on at any time before expiry. **European options** are options that only have execution on at expiry. The **premium** is a fair price required by the **writer**, or seller, of the option to purchase an option (Hull, 1997). When a person is the option seller, the individual makes money on the premium. If the premium goes up, the option buyer makes money. If they premium goes down, the option seller makes money. If the price is not fair, either the buyer or seller is taken advantage of. **Arbitrage** is buying something with the guarantee of profit when selling (Saunders & Cornett, 2019). Arbitragers act on mispriced options. Even a small misprice could create huge profits when scaled up.

Premium prices are immediately **sunk**, or lost without the ability of regaining, up front and will not be returned. If a premium did not exist, options would be the best investing vehicle ever. A person could use a strike price near zero on a call option. This would allow unlimited profit on all options purchased. The premium is in place to negate this way of investing, which is also considered a variation of arbitrage (Wilmott, Howison, & Dewynne, 1995).

Significance of the Option

Options exist in order to reduce the risk of investing. If a person buys stock, they are liable for 100% of the loss if the stock goes down. An option holder is only at risk of losing their premium if the stock goes below their strike price. The premium will never cost more than the stock itself and will usually cost much less. Therefore, the option is the best way to reduce risk in investing (Hull, 1997).

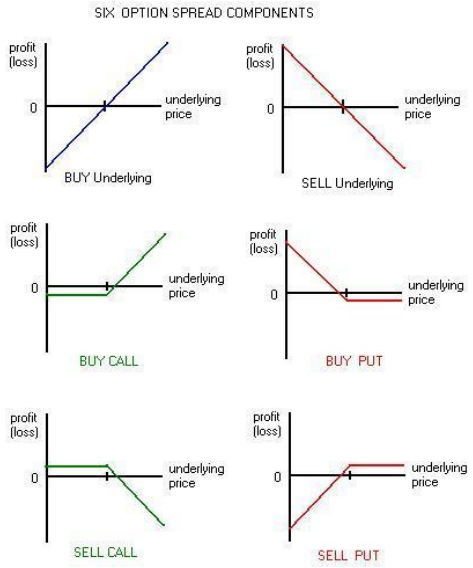
Options for Insurance Purposes

Suppose a person has 100 shares of XYZ Company. Unfortunately, CEO John Doe is found guilty of sexual harassment, which severely hurts the stock price of XYZ. Thankfully, the person had a put on XYZ Company's stock, which minimizes the loss felt by this surprise event. In this way, it acts exactly like an insurance premium on a car. Options are sometimes used as a way of insuring an individual's investments (Amram & Kulatilaka, 1998).

Put and Call Payoff Variations (Profit/Loss)

The top two graphs in Figure 1 show a profit/loss graph of buying and selling regular stock. When a person buys a stock, they make money when the stock goes up and lose money when the stock goes down. The opposite happens when the stock is sold.

Figure 1. Option Spread

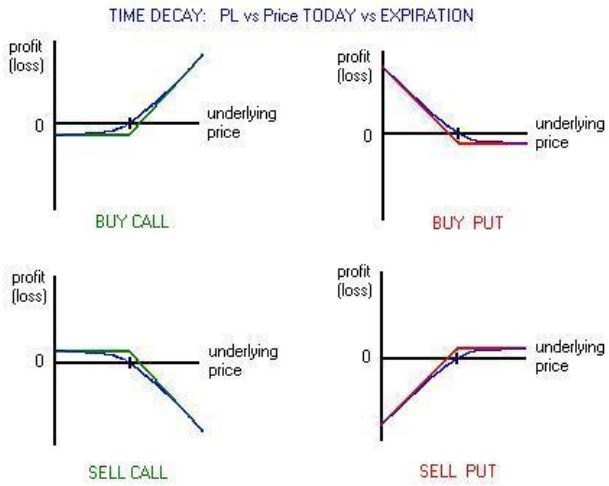


Source: Star Research, 2000

The buying and selling of puts and calls looks similar for about half of the graph, but levels out at a certain point. The reason this occurs is due to the option to not engage in a transaction with the stock. Until a person is "**in-the-money**", or able to turn a profit on the investment, a rational buyer of an option would not act on the option (Hull, 1997; Saunders & Cornett, 2019). At this point, the only loss is the premium price until the option is "in-the-money" where the graph begins to change slope. When selling options, the only profit is made on the premium. Money is lost when the seller must pay the buyer of the option (Bodie, Kane, & Marcus, 2018; Hull, 1997).

The value is affected by several variables including time. Instead of just a piecewise graph of two lines, time increases the price of the option as shown by Figure 2. As time goes by, the value of the option decreases all else being equal.

Figure 2. Time Component



Source: Star Research, 2000

Methods

In order to determine the best computation method for pricing options, the formulas need to be determined and explained. The paper for the binomial pricing model (Cox, Ross, & Rubinstein, 1979) came sometime after Black and Scholes paper (1973), but the idea of the binomial pricing model was thought of long before Black-Scholes formula. The description of how the binomial model works will lay a good groundwork for how Black-Scholes works. This paper is not a derivation of both formulas, but one needs to know how both came to be in order to understand the forces at play.

I will compare the formulas for each model once the formulas are known. Then, a comparison between the benefits of each is assessed. Each formula was programmed in Python to speed up the calculation time and provide a gateway into real world analysis. The last step will involve using the best method for pricing option premiums on real world data and analyzing it compared to the theoretical formula value. This process will also be completed using Python due

to the web connection abilities within Python. After these results are discussed, I will try to devise a way of pricing options in bulk in order to create a new trading strategy for investors in the option's market.

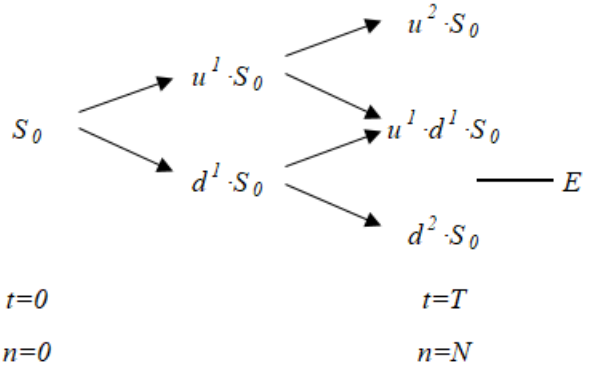
Analysis/Discussion

Binomial Pricing Tree

Figure 3 is a binomial pricing tree. The majority of this project is based on how the intrinsic premium for an option is found. Earlier, it was mentioned that the premium should be a fair price, but how is a fair price calculated and which method is most efficient or accurate? For American options, the premium is found easiest using a discrete binomial pricing tree. This allows option premium value to be given at all points in time. Option value can use a partial differential equation (PDE) continuous pricing model using a fair amount of stochastic calculus (calculus with random processes) and Brownian motion (random motion like that of stock price) (Karatzas & Shreve, 2012), which is often referred to as Black-Scholes formula. Since stock prices are updated so often, they are thought of as being continuous.

First, the binomial tree is analyzed for pricing call options. Binomial statistics problems have two outcomes (Triola & Lossi, 2018). In the case of a stock, one can assume its value may go up or down at any moment in time. Assuming the price changes between nodes, the tree allows for a wide range of possible prices. In the farthest column on the right at n=N, the tree shows the price of the stock at expiry. Over the option's life, the stock has several jumps up and/or down to get to expiry. For example, $u^3 \cdot S_0$ had three up jumps. Logically, the next step is to see where *u* and *d* come from (Cox, Ross, & Rubinstein, 1979; Wilmott, Howison, & Dewynne, 1995).

Figure 3. Binomial Pricing Tree



$S_0 = initial\ stock\ price$
$u = up\ jump\ probability$
$d = down\ jump\ probability$
$E = exercise\ price$
$T = total\ time\ until\ expirary$
$t = time\ at\ period\ n$
$N = number\ of\ horizontal\ nodes$
$n = step\ or\ node\ number\ in\ tree$
Note: The exponents show how many times an up or down jump have occurred.

Equations for *A*, *u*, and *d*

$$A = \frac{e^{-r \cdot dt}}{2} + e^{(r+v^2) \cdot dt}$$
$$u = A + \sqrt{A^2 - 1}$$
$$d = A - \sqrt{A^2 - 1}$$
$$dt = \frac{T}{N-1}$$
$r = interest\ rate\ on\ a\ 1 - yr.\ treasury\ bill\ (T - Bill)$
$v = annualized\ historical\ volatility\ (standard\ deviation\ of\ return\ on\ stock\ price)$
$dt = time\ passed\ from\ node\ to\ node$

The key is to find the potential final stock prices at expiry and roll back a premium price to initial time zero. This is done by computing the probability of an up and down jump at each point, or node. v is a percentage, because daily returns are shown in a percentage form using this equation:

$$stock\ return = \frac{S_1}{S_0} - 1$$

S_1 is stock price at $time = n$. S_0 is stock price at $time = n - 1$. *N* is the total number of horizontal nodes, which is equal to the number of vertical nodes in the last column. *A* is used to find the magnitude of the up and down jump at each node as seen in the equations for *u* and *d*. Once this calculation is complete, it is used in a future stock price equation. $u^3 \cdot S_0$ is an example of the equation at work. The equation for the last column in the tree is:

$$S(N, i) = u^{i-1} \cdot d^{N-i} \cdot S_0$$

i is a counter that starts at 0. The payout (*P*) at each node is calculated as the difference between the price of the stock at expiry (*S(N,i)*) and the strike price, or exercise price (*E*). It is easiest to imagine S and P as *N*×*N* matrices thus the *S(N,i)* format. To signify the option not being exercised for negative payouts, the equation for payoff on a call option is:

$$P(N, i) = \max(S(N, i) - E, 0)$$

A put option just needs a reversal of the subtraction. Now, using a form of discounting, expected return calculation, and self financing, the tree is rolled back without the addition of any extra money into the **portfolio**, or set, of options (Bodie, Kane, & Marcus, 2018; Wilmott, Howison, & Dewynne, 1995). This is where α and β are introduced. α is the number of stock purchased in order to provide for the possible payoff for the buyer if the option ends up being enacted. β is the number of treasury bills purchased to store the cash until more α is purchased. Since the government contracts these bills, they have almost no risk of **default**, or inability to pay back at maturity. They are also highly **liquid**, or easy to get the cash back (Saunders & Cornett, 2019). Treasury bills are a bank account for the cash needed in the case of more stock being purchased. Their equations are as follows (Wilmott, Howison, & Dewynne, 1995):

$$\alpha(n-1, i) = \frac{P(n, i+1) - P(n, i)}{S(n, i+1) - S(n, i)}$$

$$\beta(n-1,i) = \frac{P(n,i+1) - \alpha(n-1,i) \cdot S(n,i+1)}{1 + (r * dt)}$$

n is a counter starting at *N* and reducing by 1 in each run. α is dividing the difference of payout between two consecutive nodes and stock price between the same nodes in the column ahead of α and β. In this case, the final column values are being used for *S* and *P*. β is the difference between payout of the up jump and a multiplication of α and the stock price of the up jump. The denominator is used as a discounting function. Now, the stock price is calculated again for the *N-1* column. It is:

$$S(n-1,i) = u^{i-1} \cdot d^{n-1-i} \cdot S_0$$

The payout equation is now different because it uses the calculated α and β. It looks like:

$$P(n-1,i) = \alpha(n-1,i) \cdot S(n-1,i) + \beta(n-1,i)$$

The process is now complete. α and β have created a portfolio of stocks and bonds used to provide the payout at expiry. At this point, α and β is calculated to find *S* and *P* for the next column in decreasing order. This is repeated until *S(0,0),* or S_0, has been reached. The premium price is *P(0,0)* or the current value on the option.

Black-Scholes Formula

Once the number of nodes in the binomial model reaches infinity, Black-Scholes formula is created. The mathematical limit of the binomial model is Black-Scholes formula as seen in Figure 4. Fischer Black, mathematician and physicist, and Myron Scholes, an economist at the University of Chicago, published the solution in a 1973 paper called, "The pricing of options and corporate liabilities." In this paper, the black-box formula for option value was found. Several authors had been close to the formula. In fact, Case Sprenkle created a formula quite similar to the final Black-Scholes' formula. Sprenkle's formula (1961) is as follows:

Sprenkle's Formula for Call Option Value

$$Option\ Value = kxN(b_1) - k^*cN(b_2)$$

$$b_1 = \frac{\ln\left(\frac{kx}{c}\right) + \frac{1}{2}v^2(t^* - t)}{v\sqrt{t^* - t}}$$

$$b_2 = \frac{\ln\left(\frac{kx}{c}\right) - \frac{1}{2}v^2(t^* - t)}{v\sqrt{t^* - t}}$$

$k, k^* = unknown\ parameters$
$x = stock\ price$
$c = exercise\ price$
$t^* = maturity\ date$
$t = current\ date$
$v = annual\ volatility\ of\ return\ on\ stock$
$ln = natural\ logarithm$
$N(b) = cumulative\ normal\ density\ function$

This equation is used for a European call option only. Also, Sprenkle was not able to find the solutions for k and k^*. The other inputs are similar to the binomial pricing model, which makes sense because this equation is the limit of the binomial model. An equation with unknown parameters is not easy to solve and not convenient for finance professionals who are the target audience for such an equation. Black and Scholes took a different approach in their solution. The partial differential path lines up well with a well-known physics problem called the heat equation. This equation solves the change in temperature at any point in a subspace at any point in time (Bluman & Cole, 1969). This overlap between physics and finance could be the reason it took a physicist and an economist to complete the solution.

Another nice, well-known solution is the cumulative normal density function learned in statistics (Bowling, Khasawneh, Kaewkuekool, & Cho, 2009; Triola & Lossi, 2018). The heat equation also includes this type of distribution, but the finance PDE needs ran in the opposite direction to the heat equation PDE since final time in the heat equation is the starting time for

Black-Scholes. Black and Scholes were able to solve the PDE and came to a solution without the unknown constants found in Sprenkle's equation. Black-Scholes formula (1973) is as follows:

Black and Scholes Formula for European Call and Put Options

$$Call: w(x,t) = xN(d_1) - ce^{t-t^*}N(d_2)$$
$$Put: w(x,t) = ce^{t-t^*}N(-d_2) - xN(-d_1)$$

$$d_1 = \frac{\ln\left(\frac{x}{c}\right)+(r+\frac{1}{2}v^2)(t^*-t)}{v\sqrt{t^*-t}}$$
$$d_2 = \frac{\ln\left(\frac{x}{c}\right)+(r-\frac{1}{2}v^2)(t^*-t)}{v\sqrt{t^*-t}}$$

$x = stock\ price$
$c = exercise\ price$
$t^* = maturity\ date$
$t = current\ date$
$r = interest\ rate\ on\ a\ 1-yr.\ treasury\ bill\ (T-Bill)$
$v = annual\ volatility\ of\ return\ on\ stock$
$ln = natural\ logarithm$
$N(d) = cumulative\ normal\ density\ function$

Assumptions

A key difference between the Sprenkle and Black-Scholes formula is Black-Scholes inclusion of r, or the rate on a one-yr. T-Bill. In order to achieve this solution, Black and Scholes had to make a list of seven assumptions (1973):

1. **The T-Bill is constant throughout time.** Although this is not true, it does not fluctuate much. Even with the fluctuation, the answer does not change drastically.

2. **Stock price follows a predictable model.** This is not true. Unexpected systematic events, or market events, may happen that the model has no way of predicting, such as the Great Depression in 1929.

3. **The stock pays no dividends.** Robert Merton (1973) uses Black and Scholes' derivation to incorporate a dividend. The effect is minor so for convenience, this point will not be included.

4. **The option is European.** Merton (1973) also proved the value of an American option without a dividend will have the same value as a European option. It is always optimal hold options until maturity on stocks not paying a dividend.

5. **Buying and selling stock or options incurs no transaction cost.** Discounts and benefits are given to high frequency traders. Transaction costs may be minimal. Otherwise, an infinite amount of transactions (a requirement with infinite time steps) would have ill effects on profits.

6. **Fractions of stocks and T-Bills are possible to hold.** This is not true, but a fraction of stock could always be scaled up to a whole number.

7. **Short selling has no consequences.** These consequences have small effects on the intrinsic value of the option.

These assumptions do not have a large effect on the difference of the value Black-Scholes finds. Many of the assumptions were explained and make little difference in option value. For these reasons, the original Black-Scholes formula is used to evaluate the model's efficiency.

The Optimal Formula

With the equations of the binomial pricing model and Black-Scholes completely described, they can both be coded into Python. The programming methods from Black-Scholes are not as involved as the binomial pricing model. Since Black-Scholes is a black-box version, few lines of coded are needed, no matrices are needed, and no loops are required. The raw Python code for both models is found in Appendices 1-2.

In order to analyze the error in the binomial pricing model compared to Black-Scholes, I created a graph of the binomial model at several different nodes. The input for both models have the exact same arbitrary values. The binomial model approaches a limit which is Black-Scholes answer given the same input variables. One model is more beneficial based on error and computation time required. Table 1 shows the comparison of Black-Scholes and the pricing tree with a different amounts of total nodes.

Table 1.

Black-Scholes v. Binomial Pricing Model at Increasing Nodes

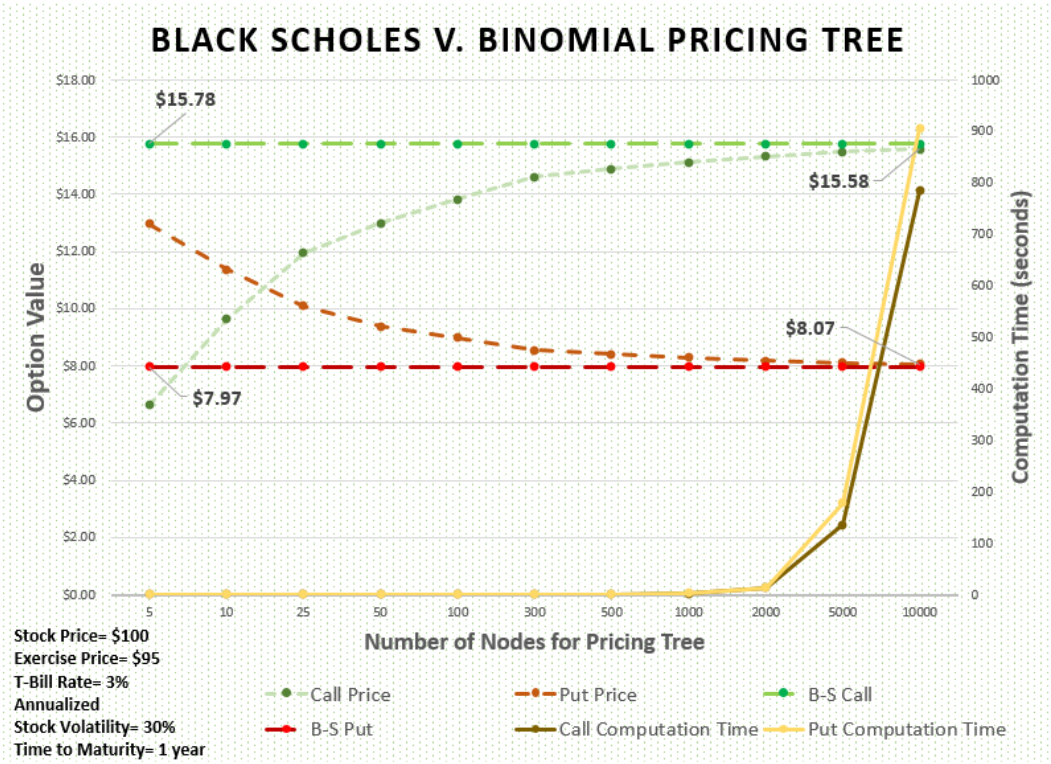
Initial Constants	
Stock Price	$100.00
Exercise Price	$95.00
Time to Maturity	1 year
T-Bill Rate	3%
Annualized Stock Volatility	30%

Type	Nodes	Call	Error	Put	Error	Call Time (s)	Put Time (s)	Matrix Setup (s)
Black-Scholes	Infinite	$15.7766	-	$7.9689	-	0.0020	0.0010	-
Binomial Pricing Tree	5	$6.6329	57.96%	$12.9697	62.75%	-	-	-
	10	$9.6397	38.90%	$11.3929	42.97%	0.0010	0.0010	-
	25	$11.9599	24.19%	$10.1033	26.78%	0.0040	0.0040	-
	50	$12.9837	17.70%	$9.3753	17.65%	0.0120	0.0150	-
	100	$13.8331	12.32%	$8.9971	12.90%	0.0578	0.0678	-
	300	$14.6296	7.27%	$8.5422	7.19%	0.2773	0.2463	0.0020
	500	$14.8978	5.57%	$8.4243	5.71%	0.8637	0.8856	-
	1000	$15.1513	3.96%	$8.2884	4.01%	3.6433	3.9933	-
	2000	$15.3344	2.80%	$8.1955	2.84%	14.1662	14.1313	0.0020
	5000	$15.4965	1.78%	$8.1122	1.80%	135.4645	178.2803	0.0040
	10000	$15.5778	1.26%	$8.0697	1.26%	785.0947	907.2761	0.0160

Source: Primary

Black-Scholes is taken to be the real value in the error formula since it is the solution with infinite nodes. The computation time of Black-Scholes is very low. The time for computation of the binomial tree is almost zero for trees with a small number of total nodes, but the error is significant. In order to reach a manageable error (under 5%), the time to complete the calculation is far more than Black-Scholes. Figure 4 shows the relationship between Black-Scholes and nodes in the binomial tree with computation time needed at different amounts of total nodes.

Figure 4.



Source: Primary

Black-Scholes is $15.78, and the binomial model approaches this value. Black-Scholes is in fact the limit of the binomial pricing model. The benefit of Black-Scholes formula is the speed of calculation and the true intrinsic value given.

Real World Comparison

With the most efficient equation for pricing option premium, real world market premium prices for options are studied and compared to what Black-Scholes gives for real world inputs. Appendix 2 shows the code used for finding and computing option value using real world data. All data was taken on November 11, 2018. The program is able to go to Yahoo Finance and take

the last five years of daily stock price data. This data is used to calculate historical volatility. The data is also where the program gets the current stock price.

The internet is also scraped, or analyzed and retrieved, for the current T-Bill rate, which is found on the United States Department of Treasury website (USDT, 2018). Making the code take inputs for only the essential information makes the user's experience with the program convenient and friendly. The program currently only needs the stock's ticker (abbreviated market name), expiry date, and strike price in order to present a value. Table 2 shows the outputs of the code in Appendix 2 for three different companies.

Table 2.

Real World Comparison of Black-Scholes Calculated Premium Value and Actual Market Premium

Constant Values	
T-Bill Rate	2.73%
Expiry Date	11/18/2019
Days to Maturity	68

Company	Ticker	Current Stock Price	Volatility	Strike	B-S Price-Call	Market Price-Call	Error	B-S Price-Put	Market Price-Put	Error
General Mills, Inc.	GIS	$ 45.31	18.47%	$32.50	$ 12.9749	$ 10.90	19.04%	$ -	$ 0.07	-100.00%
General Mills, Inc.	GIS	$ 45.31	18.47%	$45.00	$ 1.7174	$ 2.06	-16.63%	$ 1.1791	$ 1.85	-36.26%
General Mills, Inc.	GIS	$ 45.31	18.47%	$47.50	$ 0.6978	$ 1.00	-30.22%	$ 2.6468	$ 4.59	-42.34%
The Coca-Cola Co.	KO	$ 49.68	13.60%	$23.00	$ 26.7967	$ 21.54	24.40%	$ -	$ 0.02	-100.00%
The Coca-Cola Co.	KO	$ 49.68	13.60%	$49.00	$ 1.6756	$ 1.51	10.97%	$ 0.7470	$ 0.91	-17.91%
The Coca-Cola Co.	KO	$ 49.68	13.60%	$50.00	$ 1.1311	$ 0.94	20.33%	$ 1.1974	$ 1.36	-11.96%
Verizon Communications, Inc.	VZ	$ 58.46	16.74%	$23.00	$ 35.5767	$ 32.00	11.18%	$ 0.0016	$ 0.01	-84.00%
Verizon Communications, Inc.	VZ	$ 58.46	16.74%	$57.50	$ 2.3666	$ 2.31	2.45%	$ 1.1149	$ 1.46	-23.64%
Verizon Communications, Inc.	VZ	$ 58.46	16.74%	$60.00	$ 1.1554	$ 1.05	10.03%	$ 2.3910	$ 2.86	-16.40%

Source: Primary; Data Found on: (GIS, 2018; KO, 2018; VZ, 2018; USDT, 2018)

For each company, three different strike prices were accessed on the **option chain**, or list of option (call and puts) prices at different strike prices (Hull, 1997). One was deep "in-the-money", or at a profitable strike price, for a call. The other two were right on the line of "in-the-money" and "out-of-the-money" for a call. The time to expiry is 68 days, which is divided by 365 to put in terms of years; the form Black-Scholes formula needs. The volatility of all stocks was low, but the error was significant between the Black-Scholes model and the real world price.

At first, one may think this error makes Black-Scholes an invalid way to price options. Some error was as high as 100%, but this does not make Black-Scholes useless. The theory is analyzed and explained in two main ways. One way finance professionals benefit by this error is the introduction of **implied volatility** (Bodie, Kane, & Marcus, 2018). It is a well-known idea in finance that past performance is not equal to future performance. Thus, using historical data to calculate volatility is problem some. This error in Black-Scholes is used to find what the market is implying the current volatility of the stock may be.

In order to solve for this, the equation is manipulated to find volatility while making the option value equal to the current market value. If an individual tries to solve this, it is quite hard and time consuming. The person would need to get volatility out of the normal distribution calculation. Instead, one could change the input of volatility until the output of premium price matches the value given by Black-Scholes formula. For this reason, finance professionals use Black-Scholes model today in industry.

A separate thought about the error between Black-Scholes and market price is that in the future the market would reach Black-Scholes price. **Efficient market hypothesis** believes that markets are efficient and represent all current data (Saunders & Cornett, 2019). Therefore, if Black-Scholes quotes a different price, the market should reach that price over time. Now, it is clear there is an arbitrage opportunity in some situations. If Black-Scholes gives a quote higher than the market value, the market should reach this value. Therefore, if a person buys an option at a low price now and it is almost guaranteed to go up, this is an arbitrage opportunity.

Now, look back at Table 2. Any part that has a positive error represents an arbitrage opportunity. The higher the error relates to more profit. Since realistically, these prices may not end up at Black-Scholes value, it is best to act on prices that are significantly above the market

price. At the very least, one can expect these options to have the greatest probability of going up with all else being equal. The same arbitrage is captured in **selling**, or writing, options (Hull, 1997). When a person is the option seller, he or she makes money one the premium. If they premium goes down, the option seller makes money. Therefore, when the error is negative on Table 2, there is an arbitrage opportunity to sell the option at that specific strike price.

Introducing the Arbitrage Option Chain

If this calculation was completed for every strike price on the option chain, a list of the best prices to buy and sell options in order to make money using arbitrage is created. Unfortunately, that would be very time consuming. It would be nice to devise a way to price several options at once. In Appendix 3, I was able to create a program that does just that. Table 3 shows that programs output.

This new, innovative approach at pricing options is very helpful as an options trading strategy. The four sub-tables within Table 3 give the rank of arbitrage opportunities in buying and selling put and call options for a particular stock at a certain date in the future. I used the Coca-Cola option chain for January 18, 2019 (KO, 2018). My program only needs two inputs from the user. It needs a stock ticker to access and a date for a specific option chain. All other information is scraped from the internet (KO, 2018 USDT, 2018) including the option chain being accessed.

One may expect this program to take a considerable amount of time due to the retrieval of so much data from the internet, but I was able to process the program on my computer in 7.53 seconds for this particular option chain. This time includes operations separate from just

calculation for call and put. This time could have been lower with a computer with more processing power and better internet speeds.

Table 3.

KO Arbitrage Option Chain for Buying and Selling

Constant Values	
T-Bill Rate	2.73%
Expiry Date	1/18/2019
Days to Maturity	66
Current Stock Price	$ 49.75
Volatility	13.60%

Buy these **CALL** options at these strike prices:

Rank	Strike	B-S Premium	Real Premium	Diff
1	$ 44.00	$ 5.98	$ 5.75	$ 0.23
2	$ 50.00	$ 1.15	$ 0.97	$ 0.18
3	$ 49.00	$ 1.70	$ 1.55	$ 0.15
4	$ 48.00	$ 2.39	$ 2.24	$ 0.15
5	$ 47.00	$ 3.19	$ 3.05	$ 0.14
6	$ 45.00	$ 5.01	$ 4.89	$ 0.12
7	$ 46.00	$ 4.07	$ 3.95	$ 0.12
8	$ 40.00	$ 9.95	$ 9.90	$ 0.05
9	$ 55.00	$ 0.06	$ 0.03	$ 0.03

Sell these **CALL** options at these strike prices:

Rank	Strike	B-S Premiu	Real Prem	Diff
1	$ 42.00	$ 7.96	$ 8.20	$ (0.24)
2	$ 41.00	$ 8.95	$ 9.15	$ (0.20)
3	$ 43.00	$ 6.97	$ 7.00	$ (0.03)

Buy these **PUT** options at these strike prices:

Rank	Strike	B-S Premium	Real Premium	Diff
None				

Sell these **PUT** options at these strike prices:

Rank	Strike	B-S Premiu	Real Prem	Diff
1	$ 55.00	$ 5.04	$ 5.65	$ (0.61)
2	$ 49.00	$ 0.71	$ 0.99	$ (0.28)
3	$ 48.00	$ 0.40	$ 0.66	$ (0.26)
4	$ 47.00	$ 0.20	$ 0.45	$ (0.25)
5	$ 50.00	$ 1.15	$ 1.39	$ (0.24)
6	$ 46.00	$ 0.09	$ 0.31	$ (0.22)
7	$ 45.00	$ 0.04	$ 0.21	$ (0.17)
8	$ 44.00	$ 0.01	$ 0.15	$ (0.14)
9	$ 43.00	$ 0.00	$ 0.12	$ (0.12)
10	$ 42.00	$ 0.00	$ 0.09	$ (0.09)
11	$ 41.00	$ -	$ 0.08	$ (0.08)
12	$ 40.00	$ -	$ 0.07	$ (0.07)

Source: Primary; Data Found on: (KO, 2018; USDT, 2018)

Note: Values enclosed in parentheses are negative.

The left of Table 3 organizes buying puts and calls while the right sorts selling puts and calls. The right is ranked on most negative difference between Black-Scholes and market price

while the left is ranked on the most positive. This is due to the fact talked about earlier: buyers profit when the premium goes up; sellers profit when the premium goes down.

For example, if John wants to buy an option on Coca-Cola and wants the best chance for arbitrage, he can buy a call on KO at a strike price of $44 for $5.75 right now. Black-Scholes says this price will move to $5.98 at some point. John could hold this call until the premium reaches $5.98 and profit by the amount in the Diff column, or $0.23. On the other hand, if Sally wants to sell an option on Coca-Cola and wants the best chance for arbitrage, she can sell a put on KO at a strike price of $42 for $5.65 right now. Black-Scholes says this price will move to $5.04 at some point. Sally keep this put for sale until the premium reaches $5.98 and profits by the amount in the Diff column, or $0.61 even though $0.61 is listed as a negative number.

Conclusion

The world of options and pricing options is dense. Finance, in general, has a lot of jargon and lingo involved. The same goes for the options world. The quest to find and learn the process behind pricing option premiums is complete. In the search, two formulas were found that are used to price options. The formulas were much different, but had many important similarities. In the beginning, the goal was to determine which formula is best and most accurate. I also wanted to see if these formulas could still be used in the real world. The equations were found nearly 50 years ago. Some analysis was done back then, but recently, not as much robustness testing has been done. The ultimate goal was to derive a strategy for the real world options market from Black and Scholes' formula.

Summary of Findings

Through this experience, the background and history of pricing options was found. How the formulas came to be was not a quick process. Several people contributed to the formula Fischer Black and Myron Scholes found for option pricing. One person, Case Sprenkle, had the formula almost completely solved. The form of Black and Scholes' formula was almost the exact same as Sprenkle's formula. The solution is far better than the binomial pricing model based on computation time and error incurred. Since the time the formula was first found, the validity and usefulness of Black-Scholes has not been lost. There was significant error found when Black-Scholes was compared to real world data, but this error is used to benefit the market and its investors.

Finance professionals are able to use the equations in at least two useful ways: implied volatility and potential arbitrage. Implied volatility is an important indicator of the current state of a stock's volatility. Since historical performance is not the best indicator of future performance, implied volatility gives investors a better sense of how volatile the stock currently is. Arbitrage opportunity is another direction the error is taken. Investors could look at Black-Scholes intrinsic value for premium and compare it to the market price. If Black-Scholes formula finds the option to be undervalued, an investor could profit on this option.

Throughout this computation comparison and real world comparison, I was able to code these equations into Python for quick calculations and real world input. I was able to produce code that is simple for others to use and very practical in use for options trading strategies. An arbitrage option chain has never been created, and it is a very useful tool to use. The programs were also able to run quickly even with all the calculations being made.

Future Study and Last Thoughts

More analysis needs to be completed on the arbitrage option chain. Another topic to look into is the performance of this strategy in the market. Overall, the evidence of a new way to price bulk option premiums and produce a trading strategy using this information is very exciting. The formulas used in pricing options have not lost their significance to the real world. Several industries still use options for many areas. Banks, insurance companies, and individual investors all benefit in the trade of options. Options are usual vehicles to hedge investments against risk (Amram & Kulatilaka, 1998). Sometimes arbitragers can also benefit off the options market. This is where my trading strategy tries to gain an advantage. This strategy needs further testing.

References

Amram, M., & Kulatilaka, N. (1998). Real options: Managing strategic investment in an uncertain world. *OUP Catalogue*.

Black, F., & Scholes, M. (1973). The pricing of options and corporate liabilities. *Journal of political economy*, *81*(3), 637-654.

Bluman, G., & Cole, J. (1969). The General Similarity Solution of the Heat Equation. *Journal of Mathematics and Mechanics*, *18*(11), 1025-1042. Retrieved from http://www.jstor.org/stable/24893142

Bodie, Z., Kane, A., & Marcus, A. J. (2018). *Investments* (11th ed.). New York, NY: McGraw-Hill Education.

Bowling, S. R., Khasawneh, M. T., Kaewkuekool, S., & Cho, B. R. (2009). A logistic approximation to the cumulative normal distribution. *Journal of Industrial Engineering and Management*, *2*(1).

Buffett, W. E. (2003, March 6). 2002 Annual Report [Letter to Shareholders]. Omaha, NE.

Cox, J. C., Ross, S. A., & Rubinstein, M. (1979). Option pricing: A simplified approach. *Journal of financial Economics*, *7*(3), 229-263.

GIS : Summary for General Mills, Inc. (2018, November 10). Retrieved November 11, 2018, from https://finance.yahoo.com/quote/GIS?p=GIS

Hull, J., & White, A. (1987). Hedging the risks from writing foreign currency options. *Journal of International money and Finance*, *6*(2), 131-152.

Hull, J. C. (1997). *Options, futures and other derivative securities*. Upper Saddle River, NJ: Prentice Hall.

Karatzas, I., & Shreve, S. (2012). *Brownian motion and stochastic calculus* (Vol. 113). Springer Science & Business Media.

Kim, K. A., & Kim, S. H. (2015). *Global corporate finance: A focused approach* (2nd ed.). Singapore: World Scientific.

KO : Summary for Coca-Cola Company (The). (2018, November 10). Retrieved November 11, 2018, from https://finance.yahoo.com/quote/KO?p=KO

Merton, R. C. (1973). Theory of rational option pricing. *The Bell Journal of economics and management science*, 141-183.

Saunders, A., & Cornett, M. M. (2019). *Financial markets and institutions*. New York, NY: McGraw-Hill Education.

Sprenkle, C. M. (1961). Warrant prices as indicators of expectations and preferences. *Yale economic essays*, *1*(2), 179-231.

Star Research Inc. (2000). Profit (loss) vs price graphs: A simple and powerful way to understand options. Retrieved November 8, 2018, from http://www.optionstar.com/art/art2.htm

Triola, M. F., & Lossi, L. (2018). *Elementary statistics* (13th ed.). United States: Pearson.

U.S. Department of the Treasury. (2018, November 9). Retrieved November 11, 2018, from https://www.treasury.gov/resource-center/data-chart-center/interest-rates/Pages/TextView.aspx?data=yield

VZ : Summary for Verizon Communications Inc. (2018, November 10). Retrieved November 11, 2018, from https://finance.yahoo.com/quote/VZ?p=VZ

Wilmott, P., Howison, S., & Dewynne, J. (1995). *Mathematical models of financial derivative products: A student introduction*. New York: Cambridge University Press.

Appendix 1

Binomial Pricing Model – raw Python code: dsp.py

```
import numpy as np
from scipy.stats import norm
import time
import pandas as pd
from pandas_datareader import data as pdr
import datetime
import fix_yahoo_finance as yf

def bs(ticker,exp,E):
    """
    Corey's Black-Scholes Option Pricer\n
    bs(ticker,Expiry Date (exp),Exercise price(E))\n
    ticker is stock ticker (ex. AAPL)\n
    exp is mm/dd/yyyy format\n
    """
    start=time.time()
    start1=time.time()

    yf.pdr_override()
    slist = (str(datetime.datetime.today()).split()[0]).split("-")
    s = str(datetime.date(int(slist[0])-5,int(slist[1]),int(slist[2])))

    while True:
        try:
            data = pdr.get_data_yahoo(ticker, start = s)
            break
        except ValueError:
            time.sleep(5)
            continue
        except OverflowError:
            slist = s.split("-")
            slist[0]=int(slist[0])+1
            s = str(datetime.date(int(slist[0]),int(slist[1]),int(slist[2])))
            continue
        break

    print('Done Data Collection',time.time()-start1)
    start1=time.time()

    data=data[['Close']].copy()
    vl=float(data.pct_change().std())*np.sqrt(252)
```

```
    data=data[::-1]
    price=data.iat[0,0]
    s=price

    print('Done Volatility',time.time()-start1)
    start1=time.time()

    #1 year T-Bill Bond rate
    url='https://www.treasury.gov/resource-center/data-chart-center/interest-
rates/Pages/TextView.aspx?data=yield'
    tbill_df=pd.read_html(url)[1]
    bond=float(tbill_df[5].iloc[-1])/100
    r=bond

    print('Done Bond',time.time()-start1)
    start1=time.time()

    uexp=int(time.mktime(datetime.datetime.strptime(exp,
"%m/%d/%Y").timetuple()))-18000
    today=time.strftime("%m/%d/%Y")
    utoday=int(time.mktime(datetime.datetime.strptime(today,
"%m/%d/%Y").timetuple()))-18000
    diffday=(uexp-utoday)/86400
    t=diffday/365

    d1=(np.log(s/E)+(r+vl**2/2)*t)/(vl*np.sqrt(t))
    d2=d1-(vl*np.sqrt(t))
    callpremium=s*norm.cdf(d1)-norm.cdf(d2)*E*np.e**(-r*t)

    print('Done Call',time.time()-start1)
    start1=time.time()

    putpremium=norm.cdf(-d2)*E*np.e**(-r*t)-s*norm.cdf(-d1)
    put=callpremium+(E/(1+r)**t)-s

    print('Done Put',time.time()-start1)
    end=time.time()-start

    print()
    print('The premium of the call option is $%(cp)4f.' %{'cp':callpremium})
    print('The premium of the put option is $%(pp).4f.' %{'pp':putpremium})
    print('Put-Call parity says the put is $%(put).4f.' %{'put':put})
    print('The current price is $%(price).2f.' %{'price':price})
    print('This took %(end).2f seconds.' %{'end':end})
    print('Volatility= '+str(vl))
```

```
    print('T-bill rate= '+str(r))
    print('Days to Maturity= '+str(diffday))
    return ''

ticker=input('Ticker: ')
strike=float(input('Strike Price: '))
exp=input('Expiry Date (mm/dd/yyyy): ')

print(bs(ticker,exp,strike))
```

Appendix 2

Black-Scholes' formula – raw Python Code: bs.py

```
import numpy as np
import time
import pandas as pd
from pandas_datareader import data as pdr
import datetime
import fix_yahoo_finance as yf

def call(N,u,d,E,s,r,dt,S,P,alpha,beta):
    start1=time.time()

    for i in range(0,N):
        S[N,i]=u**(i)*d**(N-i)*s
        P[N,i]=max(S[N,i]-E,0)

    for n in range(N,0,-1):
        for i in range(0,n):
            alpha[n-1,i]=(P[n,i+1]-P[n,i])/(S[n,i+1]-S[n,i])
            beta[n-1,i]=(P[n,i+1]-alpha[n-1,i]*S[n,i+1])/(1+(r*dt))
            S[n-1,i]=u**(i-1)*d**(n-1-i)*s
            P[n-1,i]=alpha[n-1,i]*S[n-1,i]+beta[n-1,i]
    premium_price=P[0,0]

    print('Done Call Premium',time.time()-start1)
    return premium_price

def put(N,u,d,E,s,r,dt,S,P,alpha,beta):
    start1=time.time()

    for i in range(0,N):
        S[N,i]=u**(i)*d**(N-i)*s
        P[N,i]=max(E-S[N,i],0)

    for n in range(N,0,-1):
        for i in range(0,n):
            alpha[n-1,i]=(P[n,i+1]-P[n,i])/(S[n,i+1]-S[n,i])
            beta[n-1,i]=(P[n,i+1]-alpha[n-1,i]*S[n,i+1])/(1+(r*dt))
            S[n-1,i]=u**(i-1)*d**(n-1-i)*s
            P[n-1,i]=alpha[n-1,i]*S[n-1,i]+beta[n-1,i]
    premium_price=P[0,0]

    print('Done Put Premium',time.time()-start1)
```

```
    return premium_price

def dsp(ticker,exp,E,N=2000):
    """
    Corey's Discrete Option Pricer\n
    dsp(ticker,time(T),nodes on time(N),Exercise price(E))\n
    ticker is stock ticker (ex. AAPL)\n
    time in days\n
    more nodes gives more precise value\n
    """
    start=time.time()
    start1=time.time()

    yf.pdr_override()
    slist = (str(datetime.datetime.today()).split()[0]).split("-")
    s = str(datetime.date(int(slist[0])-5,int(slist[1]),int(slist[2])))

    while True:
        try:
            data = pdr.get_data_yahoo(ticker, start = s)
            break
        except ValueError:
            time.sleep(5)
            continue
        except OverflowError:
            slist = s.split("-")
            slist[0]=int(slist[0])+1
            s = str(datetime.date(int(slist[0]),int(slist[1]),int(slist[2])))
            continue
        break

    print('Done Data Collection',time.time()-start1)
    start1=time.time()

    data=data[['Close']].copy()
    vl=float(data.pct_change().std())*np.sqrt(252)
    data=data[::-1]
    price=data.iat[0,0]
    s=price

    print('Done Volatility',time.time()-start1)
    start1=time.time()

    #1 year T-Bill rate
```

```
    url='https://www.treasury.gov/resource-center/data-chart-center/interest-
rates/Pages/TextView.aspx?data=yield'
    tbill_df=pd.read_html(url)[1]
    r=float(tbill_df[5].iloc[-1])/100

    print('Done T-Bill',time.time()-start1)
    start1=time.time()

    uexp=int(time.mktime(datetime.datetime.strptime(exp,
"%m/%d/%Y").timetuple()))-18000
    today=time.strftime("%m/%d/%Y")
    utoday=int(time.mktime(datetime.datetime.strptime(today,
"%m/%d/%Y").timetuple()))-18000
    diffday=(uexp-utoday)/86400
    T=diffday/365
    dt=T/(N-1)
    A=(.5*(np.exp(-r*dt)+np.exp((r+vl**2)*dt)))
    u=A+np.sqrt(A**2-1)
    d=A-np.sqrt(A**2-1)

    print('Done A,u,d',time.time()-start1)
    start1=time.time()

    S=np.zeros((N,N))
    P=np.zeros((N,N))
    alpha=np.zeros((N,N))
    beta=np.zeros((N,N))
    N=N-1

    print('Done Matrix Setup',time.time()-start1)

    callp=call(N,u,d,E,s,r,dt,S,P,alpha,beta)
    putp=put(N,u,d,E,s,r,dt,S,P,alpha,beta)

    end=time.time()-start

    print()
    print('The premium of the call option is $%(premium).4f.'
%{'premium':callp})
    print('The premium of the put option is $%(premium).4f.'
%{'premium':putp})
    print('The current price is $%(price).2f.' %{'price':price})
    print('This took %(end).2f seconds.' %{'end':end})
    return ''
```

```
ticker=input('Ticker: ')
strike=float(input('Strike Price: '))
exp=input('Expiry Date (mm/dd/yyyy): ')
nodes=float(input('Steps in tree: '))

print(dsp(ticker,exp,strike,N=nodes))
```

Appendix 3

Black-Scholes Arbitrage Option Chain – raw Python code: bsauto.py

```
import numpy as np
from scipy.stats import norm
import time
import pandas as pd
from pandas_datareader import data as pdr
import datetime
import fix_yahoo_finance as yf

def vlpr(ticker):
    yf.pdr_override()
    slist = (str(datetime.datetime.today()).split()[0]).split("-")
    s = str(datetime.date(int(slist[0])-5,int(slist[1]),int(slist[2])))

    while True:
        try:
            data = pdr.get_data_yahoo(ticker, start = s)
            break
        except ValueError:
            continue
        except OverflowError:
            slist = s.split("-")
            slist[0]=int(slist[0])+1
            s = str(datetime.date(int(slist[0]),int(slist[1]),int(slist[2])))
            continue
        break

    data=data[['Close']].copy()
    vl=float(data.pct_change().std())*np.sqrt(252)
    data=data[::-1]
    s=data.iat[0,0]

    #1 year T-Bill rate
    url='https://www.treasury.gov/resource-center/data-chart-center/interest-
rates/Pages/TextView.aspx?data=yield'
    tbill_df=pd.read_html(url)[1]
    r=float(tbill_df[5].iloc[-1])/100

    return vl,s,r

def bs(t,E,vl,s,r):

    d1=(np.log(s/E)+(r+vl**2/2)*t)/(vl*np.sqrt(t))
    d2=d1-(vl*np.sqrt(t))
    callpremium=s*norm.cdf(d1)-norm.cdf(d2)*E*np.e**(-r*t)
    putpremium=norm.cdf(-d2)*E*np.e**(-r*t)-s*norm.cdf(-d1)

    return callpremium,putpremium

def striker(ticker,exp):

    uexp=int(time.mktime(datetime.datetime.strptime(exp,
"%m/%d/%Y").timetuple()))-18000
```

```
    today=time.strftime("%m/%d/%Y")
    utoday=int(time.mktime(datetime.datetime.strptime(today,
"%m/%d/%Y").timetuple()))-18000
    diffday=(uexp-utoday)/86400
    t=diffday/365


url='https://finance.yahoo.com/quote/%(ticker)s/options?p=%(ticker)s&date=%(u
date)s' %{'ticker':ticker,'udate':uexp}
    optionscall=pd.read_html(url)[0]
    optionsput=pd.read_html(url)[1]

    vlpr1=vlpr(ticker)
    vl=vlpr1[0]
    s=vlpr1[1]
    r=vlpr1[2]

    pricerange=s*.2
    low=s-pricerange
    high=s+pricerange

    #Call Option Spread
    optionscall=optionscall[['Strike','Strike','Last Price']].copy()
    optionscall.columns=['Strike','BS Premium','Real Premium']

    optionscall = optionscall[optionscall['BS Premium'].between(low, high,
inclusive=True)]
    for x in range(0,optionscall.shape[0]):
        optionscall.iat[x,1]=round(bs(t,optionscall.iat[x,1],vl,s,r)[0],3)
    optionscall['Diff']=optionscall['BS Premium']-optionscall['Real Premium']
    optionscallbuy=optionscall
    optionscallsell=optionscall
    optionscallbuy = optionscall[optionscall['Diff'].gt(0)]
    optionscallsell = optionscall[optionscall['Diff'].lt(0)]

    #Put Option Spread
    optionsput=optionsput[['Strike','Strike','Last Price']].copy()
    optionsput.columns=['Strike','BS Premium','Real Premium']

    optionsput = optionsput[optionsput['BS Premium'].between(low, high,
inclusive=True)]
    for x in range(0,optionsput.shape[0]):
        optionsput.iat[x,1]=round(bs(t,optionsput.iat[x,1],vl,s,r)[1],3)
    optionsput['Diff']=optionsput['BS Premium']-optionsput['Real Premium']
    optionsputbuy=optionsput
    optionsputsell=optionsput
    optionsputbuy = optionsput[optionsput['Diff'].gt(0)]
    optionsputsell = optionsput[optionsput['Diff'].lt(0)]

    calltablebuy=optionscallbuy.sort_values(by=['Diff'],
ascending=False).reset_index(drop=True)
    calltablesell=optionscallsell.sort_values(by=['Diff'],
ascending=True).reset_index(drop=True)
    puttablebuy=optionsputbuy.sort_values(by=['Diff'],
ascending=False).reset_index(drop=True)
    puttablesell=optionsputsell.sort_values(by=['Diff'],
ascending=True).reset_index(drop=True)
```

```
    #Printing
    print()
    print('%(ticker)s Arbitrage Option Chain' %{'ticker':ticker})
    print('Buy these call options at these strike prices: ')
    print(calltablebuy)
    print('Buy these put options at these strike prices: ')
    print(puttablebuy)
    print()
    print('Sell these call options at these strike prices: ')
    print(calltablesell)
    print('Sell these put options at these strike prices: ')
    print(puttablesell)
    print()
    print('Current Stock Price= '+str(s))
    print('Volatility= '+str(vl))
    print('T-bill rate= '+str(r))
    print('Days to Maturity= '+str(diffday))

    return ''

ticker=input('Ticker: ')
exp=input('Expiry Date (mm/dd/yyyy): ')

striker(ticker,exp)
```