The Evaluation, Comparison, Selection, and Implementation of Derivative Pricing Methods into a Trading Algorithm

JASON BOHNE

Faculty Supervisor: Jie Yang

B.S. in Mathematics
Honors College
The University of Illinois at Chicago

Abstract

The aim of this thesis is to analyze and apply the most common pricing methods for European and American options to real financial market data in order to determine the accuracy of each method. Moreover, by evaluating the accuracy across a variety of market conditions, we hope to determine whether there is a significant difference between each methods results and if so, which one is the most accurate across our tests. We will then implement our results into a trading algorithm to test over an out-of-sample data set to either confirm or not confirm our findings.

Contents

CHAPTER 1

Introduction

The history of derivatives began centuries ago when ancient Greek and Roman farmers used forward delivery contracts to protect themselves against uncertainty in future crop yields. While popular, these contracts were primarily used for hedging, not speculative purposes. Nowadays some common standardized derivatives include futures, options, and interest rate swaps. The focus of this thesis will be on analyzing and applying option pricing methods. **Chapter 1** will contain an introduction on option contracts along with some of their common properties. In **Chapter 2** we will theoretically discuss a variety of pricing methods for European and American options for which we will provide the scripts in **Chapter 3**. **Chapter 4** will describe the setup and results of our experiment. We will end our paper with a conclusion on our findings along with some possible steps moving forward in **Chapter 5**.

1.1 Background

That being said, it is crucial to define what an option is. Options are publicly traded financial instruments with a specific underlying asset attached to them, commonly equities. The holder of the option has the right to buy or sell the underlying asset at the strike price of the option. The right of the option is classified as either a call or a put which respectively allows the holder to buy or sell the underlying. Strikes are generally issued at fixed intervals above and below the current market price with the prescribed distance between strikes dependent on the market price.

In addition to strikes, each option has an expiration which is the final date a holder can exercise the contract on. Options can either be American or European Style. An American option allows the holder to exercise their right anytime before the expiration date while a

European option only allows the holder to exercise on the expiration date. The standard-issue expirations available to traders are weekly's, monthly's, quarterlies, and LEAPs (Long-term Equity Anticipation Securities). The Options Clearing Corporation (OCC), the clearinghouse of all listed options traded in the United States, produces a yearly calendar with expirations for contracts with the standard expirations noted above [13].

It is important to note the unique Option Symbol generated by the OCC for each option contract [19]. Given the underlying asset ticker, expiration date, option right, and strike price of a contract the corresponding option symbol is of the following form:
$"Ticker"$ $"Expiration"$ $"Right"$ $"Strike."$

We provide the following example below for clarification:

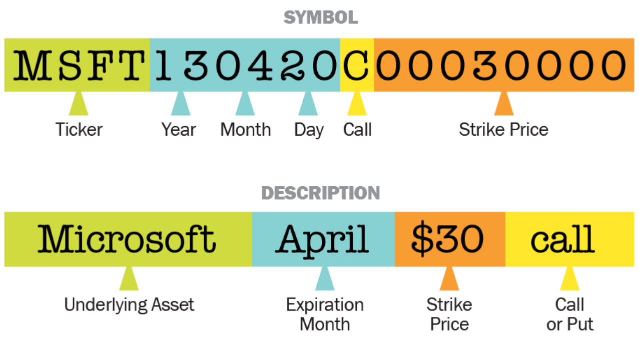


FIGURE 1.1: OCC Option Symbol Components
[14]

CHAPTER 2

Option Pricing Methods

As one of the purposes of this thesis is to analyze the most common pricing methods for European and American Options, we will provide a thorough analysis below. While the most common pricing methods are discussed by no means is the list exhaustive.

2.1 Analytic Formulas

The first class of option pricing methods we are going to examine are Analytic Methods. These methods give an explicit formula for the true value of a European or American option based on a set of input parameters. We will be discussing one such method, the Black-Scholes Pricing Formulas.

Black-Scholes Pricing Formulas

Introduced in the 1973 paper, *"The Pricing of Options and Corporate Liabilities"*[1] the Black Scholes Pricing Formulas are the solution to the Black Scholes Partial Differential Equation developed by Fischer Black and Myron Scholes. The PDE, as shown below, expresses the price of a European call or put option, V, as a function of the stock price S, the risk-free rate r, the volatility of the stock σ, and the time to expiration t.

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} + \frac{\partial V}{\partial t} - rV = 0$$

Solving the partial differential equation with European option boundary conditions results in the Black-Scholes Formulas, which one can use to calculate the expected value of a European option.

For the expected value of a European call option, $C(S,t)$ we note the solution is:

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

where

$$d_1 = \frac{\ln\frac{S}{K} + (r + \frac{\sigma^2}{2})(t)}{\sigma\sqrt{t}}$$

and

$$d_2 = \frac{\ln\frac{S}{K} + (r - \frac{\sigma^2}{2})(t)}{\sigma\sqrt{t}}$$

given the spot price S, the risk-free rate r, the volatility of the underlying σ, the time to expiration t and the strike price K. Note $N(\cdot)$ is the cumulative distribution function of the standard normal distribution.

With a similar analysis, it is not difficult to solve for the expected value of a European put option which is

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

where all variables represent equivalent values as when evaluating the European call option.

2.2 Analytic Approximations

The second class of option pricing methods we are going to examine are Analytic Approximation Methods. These methods give an approximation for the true value of an American option based on approximating the Black-Scholes PDE. While the Black-Scholes PDE with European option boundary conditions gives analytic solutions this is not the case with American option boundary conditions, hence the approximation. Nonetheless, we will be discussing two such methods, the Barone-Adesi-Whaley Method and the Bjerksund-Stensland Method.

Barone-Adesi-Whaley Method

Our analysis begins with the Barone-Adesi-Whaley Method. Described in their 1987 paper, *"Efficient Analytic Approximation of American Option Values"* [5], Barone-Adesi and Whaley describe a valuation technique for commodity options. Specifically taking into account the common early exercise feature which characterizes American options, they provide a Quadratic Approximation for the value of both American calls and puts.

Again we will not provide a formal derivation but instead an overview of the key points and formulas from this method. That being said, Barone and Whaley argue that assuming the Black-Scholes PDE applies to American options, then it also must apply to the early exercise premium of American options.

Approximating the resulting equation transforms the early exercise premium partial differential equation into a second-order ordinary differential equation below:

$$S^2 F_{SS} + NSF_S - \frac{M}{K}F = 0$$

This can be solved for two roots q_1, q_2. After applying the known boundary conditions, the following formulas appear:

The Quadratic Approximation of American call option, $C(S,T)$ is:

$$\begin{cases} C(S,T) = c(S,T) + A_2(\frac{S}{S^*})^{q_2} & S < S^* \\ C(S,T) = S - X & S \geq S^* \end{cases}$$

Such that $A_2(\frac{S}{S^*})^{q_2}$ is our approximation where $A_2 = \frac{S^*}{q_2}(1 - e^{(b-r)T}N[d_1(S^*)])$ and $c(S,T)$ represents the European call option value.

Where the spot price S, the critical spot price S^* the short term interest rate r, cost of carrying b, the time to expiration T and exercisable proceeds $S - X$. Additionally $N(d_1)$ is the same cumulative distribution function of the standard normal distribution from the Black-Scholes.

With a similar analysis, it is not difficult to solve for the Quadratic Approximation of an American put option which is

$$\begin{cases} P(S,T) = p(S,T) + A_1(\frac{S}{S^{**}})^{q_1} & S > S^{**} \\ P(S,T) = X - S & S \leq S^{**} \end{cases}$$

Such that $A_1 = -\frac{S^{**}}{q_1}(1 - e^{(b-r)T}N[-d_1(S^{**})]$, $p(S, T)$ is the European put option value, S^{**} is the updated critical spot price and all other variables are equivalent as in the American call approximation.

Bjerksund-Stensland Method

The next analytic approximation method of interest is the Bjerksund-Stensland Method. Referencing the work from their 1993 paper, *"Closed-form approximation of American options"* [7], the Bjerksund-Stensland method approximates American calls and puts by providing a lower bound to the option value.

Bjerksund and Stensland first present a class of general exercise strategies along with the corresponding option values. In each exercise strategy, there is a constant trigger price X which defines the price at which the holder will exercise the contract early.

For any exercise strategy that satisfies these requirements, the approximate value of the American call option is:

$$c = \alpha(X)S^{\beta} - \alpha(X)\phi(S, T|\beta, X, X) + \phi(S, T|1, X, X) -$$

$$\phi(S, T|1, K, X) - K\phi(S, T|0, X, X) + K\phi(S, T|0, K, X)$$

such that S is the spot price, T is the time to expiration, K is the strike price, X is the trigger price, σ is underlying volatility, b is cost to carry, r is risk-free interest rate, $\alpha(X) := (X - K)X^{-\beta}$ and $\beta := (\frac{1}{2} - \frac{b}{\sigma^2}) + \sqrt{(\frac{b}{\sigma^2} - \frac{1}{2})^2 + \frac{2r}{\sigma^2}}$

Additionally our function $\phi(S, T|\tau, H, X)$ evaluates the payoff at time T assuming the spot price $S_T < X \; \forall t \in [0, T)$ and the final spot price $S_T < H \leq X$. This intuitively makes sense as we want to know the option payoff assuming we did not exercise early, implying the spot never reached or exceeded the trigger. As the exact formulation of ϕ is quite lengthy we have omitted it.

From there they explore two specific strategies that vary the trigger price, X, of the exercise strategy. We will only describe the strategy used in our implementation. Here we define the optimal exercise price for a given contract as B_T and the trigger price as X_T where

$$X_T = B_0 + (B_\infty - B_0)(1 - e^h(T))$$

One can view our trigger price, X_T as a time-weighted average of the optimal exercise price between an infinite lived, B_∞ and an infinitesimal lived, B_0 American option. Incorporating this trigger in our approximation gives us the Bjerksund-Stensland Approximation for American call options.

As expected, a similar analysis can be completed to shown that the approximate value of the American put option is identical to that of the American call with the following change:

$$\beta := (\frac{1}{2} + \frac{b}{\sigma^2}) + \sqrt{(-\frac{b}{\sigma^2} - \frac{1}{2})^2 + \frac{2r-b}{\sigma^2}}$$

2.3 Numerical Methods

Next we are going to examine Numerical Methods. There are many different applications of numerical methods in option pricing from binomial trees to finite difference methods . Binomial trees are graphs, specifically trees, that allow for a finite number of outcomes at each time step. By working backward in time one can approximate the current true value of a European or American option. Finite Difference Methods approximate the solution to a differential equation using Taylor Series expansions of the derivatives in the underlying equation. We will be discussing three numerical methods, the Cox-Ross-Rubenstein Binomial Tree, the Jarrow-Rudd Binomial Tree, and the Crank-Nicolson Finite Difference Method.

Binomial Trees

As we will be discussing variations of Binomial Trees it would be helpful to first discuss the general idea. Let's assume a stock price can move either up or down over discrete time periods. Then the underlying asset which is price S at time t can move to either uS or dS at time $t+1$ with respective probabilities of $q,\ 1-q$, shown below:[18]

$$\begin{tikzcd}
 & \begin{matrix} uS \\ (O_u) \end{matrix} \\
\begin{matrix} S \\ (O) \end{matrix} \arrow[ur, "q"] \arrow[dr, "1-q"'] & \\
 & \begin{matrix} dS \\ (O_d) \end{matrix}
\end{tikzcd}$$

FIGURE 2.1: General Binomial Tree

While we provided a one-step Binomial Tree, it is natural to see how one can extend the tree to n steps where there are $n+1$ outcomes for the option at expiration. Once the underlying binomial tree is constructed one can use the payoffs at each of the $n+1$ outcomes to calculate the option prices backward in time. For European Options, the expected value of the option, E , at time-step t is

$$Ee^{r\triangle t} = O_u q + O_d(1-q)$$

such that $e^{r\triangle t}$ is our discount factor $O_u,\ O_d$ are the options directly above and below at $t+1$. Working backwards to $t=0$ gives us the present option value.

There is a caveat for American options as the holder can exercise the contract whenever. Therefore in our backward time-stepping we must compare the exercisable proceeds of the option to its current expected value. If at any node the proceeds are greater, we assume the option is exercised there and assign that node the value of the proceeds.

One last point is that as $\triangle t \to 0$ both the Cox-Ross-Rubinstein and Jarrow-Rudd Binomial Trees (along with all of the other major Binomial Trees) converge to the value computed by the Black-Scholes Pricing Formulas. This is shown rigorously in the original papers for each Binomial Tree, respectively [2] and [3].

Cox-Ross-Rubinstein Binomial Tree

Cox, Ross, and Rubinstein (C-R-R) introduced one of the first references to option pricing via Binomial Trees in their 1979 paper, *"Option Pricing: A Simplified Approach"* [2]. A majority of their paper is dedicated to setting up the arbitrage-free argument along with the Binomial Tree framework for the underlying asset and option values. However, the main point which characterizes the C-R-R Binomial Tree is that the upwards multiplier is the inverse of the downwards multiplier s.t. $ud = 1$. Commonly referred to as an Equal Jumps Binomial Tree, up moves commute with down moves, simplifying the computation immensely. See below a one-step Cox-Ross-Rubinstein Binomial Tree

$$\begin{tikzcd} & \begin{array}{c} uS \\ (O_u) \end{array} \\ \begin{array}{c} S \\ (O) \end{array} \arrow[ur, "q"] \arrow[dr, "1-q"'] & \\ & \begin{array}{c} \frac{1}{u}S \\ (O_d) \end{array} \end{tikzcd}$$

FIGURE 2.2: C-R-R Binomial Tree

Denoting r as our compounding factor of the riskfree interest rate C-R-R provides the following probability formulas for a call option as the following:

$$q = \frac{r-d}{u-d}$$

$$1-q = \frac{u-r}{u-d}$$

Using these probabilities one can work backward in time and solve for the present value of the option.

Jarrow-Rudd Binomial Tree

Equally as famous as the Cox-Ross-Rubinstein Binomial Tree, Jarrow and Rudd described an Equal Probability Binomial Tree in their 1982 paper, *Approximate Option Valuation For Arbitrary Stochastic Processes"*[3]. While the general argument for the Binomial Tree is the same, the main difference occurs in the underlying asset Binomial Tree. In their formulation, a stock with price S at time t can move to either uS or dS at time step $t+1$ with equal probabilities of 50% as shown below: As the probabilities are known the size of the up and

$$\begin{tikzcd} & \begin{matrix} uS \\ (O_u) \end{matrix} \\ \begin{matrix} S \\ (O) \end{matrix} \arrow[ur, "50\%"] \arrow[dr, "50\%"'] & \\ & \begin{matrix} dS \\ (O_d) \end{matrix} \end{tikzcd}$$

down movement must be calculated. The formulas for both the up and down step size are the following[15]:

$$u = e^{(r-\frac{\sigma^2}{2})\triangle t+\sigma\sqrt{\triangle t}}$$

$$d = e^{(r-\frac{\sigma^2}{2})\triangle t-\sigma\sqrt{\triangle t}}$$

where r is riskfree rate, σ is the underlying volatility and $\triangle t$ is our change in time. Once we have constructed our underlying asset Binomial Tree we can work backward in time to solve for the present value of the option.

While we discussed two common Binomial Trees used in option pricing, there are many other variations including but not limited to the Leisen-Reimer Binomial Tree[8] and the Tian Binomial Tree[9].

Crank-Nicolson Finite Difference Method

There has been a great deal of literature devoted to approximating asset prices with explicit and implicit finite difference methods. Both Geske-Shastri [4] and Hull[6] describe finite difference methods for approximating value of derivatives. We will only be examining the Crank-Nicolson Finite Difference Method, which is commonly used to numerically solve diffusion equations. We first note the equation we wish to numerically solve is the time-dependent PDE Black-Scholes, shown below, on a grid of prices that our underlying asset can move to at some future time t.

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} + \frac{\partial V}{\partial t} - rV = 0$$

Therefore it is a natural extension to apply the Crank-Nicolson Finite Differences Method. Following a discretization on the Black-Scholes PDE similar to Goddard [16] about $f_{i+\frac{1}{2},j}$ we perform a central approximation for the first derivative in time and a central approximation for the first/second derivative in space resulting in:

$$-a_j f_{i-1,j-1} + (1+2b_j)f_{i-1,j} + c_j f_{i-1,j+1} = a_j f_{i,j-1} + (1-2b_j)f_{i,j} + c_j f_{i,j+1}$$

Such that $a_j = c_j = \delta_t \frac{\sigma^2 j^2 - rj}{4}$, $b_j = \delta_t \frac{\sigma^2 j^2 + r}{4}$, σ= volatility of underlying, and r is the riskfree rate.

Once we create a tridiagonal matrix from our scheme, we solve the implicit method working backward in time. In the context of our problem we examine the possible payoffs of our option at the expiration date, $t = t_{final}$ and work backward to $t = 0$.

A great benefit of using finite difference methods is that the approach can be used to approximate both European and American options as long as the boundary conditions are adjusted accordingly. Given American options, one would include the early exercise premium of the contract into the boundary conditions of our PDE to account for the holder's ability to exercise prematurely. Additionally, we achieve stability with any mesh spacing i, j and any time step t along with second-order accuracy in space and time.[10]

CHAPTER 3

Pricing Method Scripts

In chapter 2, we theoretically analyzed common methods used for pricing options. This chapter will include the scripts needed for our analysis with real financial data. Using the QuantLib library of pricing engines we are able to calculate the theoretical price of a specific contract for each one of our desired methods[1]. For conciseness, we will only include a subset of the QuantLib Script to highlight the mathematics.

Barone-Adesi-Whaley Analytic Approximation [11]

We start off with a script that calculates the Barone-Adesi-Whaley Analytic Approximation. Note the theoretical value for the American call or put is calculated in the switch control statement, with the cases depending on the right of the option. As we exhibited in chapter 2 when $S < S^*$ we approximate the American call using the European call value along with a function of S^* and when $S^* \geq S$ we apply our boundary conditions to approximate the American call as the exercisable proceeds of the difference between strike and spot. A similar argument applies to the American put.

```
        switch (payoff->optionType()) {
            case Option::Call:
            //Quadratic Approximation for American call
                Q = (-(n-1.0) + std::sqrt(((n-1.0)*(n-1.0))+4.0*K))/2.0;
                a =  (Sk/Q) * (1.0 - dividendDiscount * cumNormalDist(d1));
                if (spot<Sk) {
                // European call value + function of S*
                    results_.value = black.value() +
                        a * std::pow((spot/Sk), Q);
                     }
                else {
                    results_.value = spot - payoff->strike();
                    //Imposed by Boundary Conditions
```

[1]We will not be including the Black-Scholes Pricing Formulas in our scripts or experiment. While an American call on a non-dividend issuing stock can be evaluated with the Black-Scholes Pricing Formula, the formulas cannot be applied to American puts.

```
                    }
                break;
            case Option::Put:
                Q = (-(n-1.0) - std::sqrt(((n-1.0)*(n-1.0))+4.0*K))/2.0;
                a = -(Sk/Q) *
                         (1.0 - dividendDiscount * cumNormalDist(-d1));
                if (spot>Sk) {
                    results_.value = black.value() +
                             a * std::pow((spot/Sk), Q);
                    }
                else {
                    results_.value = payoff->strike() - spot;
                    }
                break;
            }
        }
```

Bjerksund-Stensland Analytic Approximation [11]

Examining the script for the Bjerksund-Stensland Analytic Approximation we see $variance$ corresponds with an underlying volatility estimator. In terms of the pricing method, the theoretical value for the American call (or likewise American put by put-call Parity) is returned by $americanCallApproximation()$. We see the Bjerksund-Stensland calculates the optimal trigger price as an average between an infinite, B_∞ and an infinitesimal, B_0 lived American call. Given the trigger price, we calculate the approximation likewise.

```

    Real americanCallApproximation(Real S, Real X,
                                        Real rfD, Real dD, Real variance) {
        // bt: our optimal exercise price at time t
        Real bT = std::log(dD/rfD);
        Real rT = std::log(1.0/rfD);
        Real beta = (0.5 - bT/variance) +
                 std::sqrt(std::pow((bT/variance - 0.5), Real(2.0))
                           + 2.0 * rT/variance);
        // BInfinity: optimal exercise price for an infinite lived contract
        Real BInfinity = beta / (beta - 1.0) * X;

        //B0= optimal exercise price for infinitesimal lived contract
        Real B0 = std::max(X, rT / (rT - bT) * X);
        Real ht = -(bT + 2.0*std::sqrt(variance)) * B0 / (BInfinity - B0);
        // I our trigger price
        Real I = B0 + (BInfinity - B0) * (1 - std::exp(ht));
        QL_REQUIRE(I >= X,
                        "Bjerksund-Stensland approximation not applicable "
                        "to this set of parameters");
        if (S >= I) {
            return S - X;}
        else {
            // Using our phi function we calculate the approximation
            return (I - X) * std::pow(S/I, beta)
                         *(1 - phi(S, beta, I, I, rT, bT, variance))
                     +    S *  phi(S,  1.0, I, I, rT, bT, variance)
                     -    S *  phi(S,  1.0, X, I, rT, bT, variance)
                     -    X *  phi(S,  0.0, I, I, rT, bT, variance)
                     +    X *  phi(S,  0.0, X, I, rT, bT, variance);
            }
        }
    }
```

Below we show our implementation for both the Binomial Cox-Ross-Rubinstein Method and the Binomial Jarrow-Rudd Method. We note the scripts are both instances of the base Binomial Tree class. We are going to enforce that both Binomial Trees have 100 nodes.

Cox-Ross-Rubinstein Binomial Tree [11]

We first start with the Cox-Ross-Rubinstein Binomial tree which is an equal jump tree. For any node S at time t, S can move to either S_{up} or S_{down} at time $t{+}1$ such that $|up| = |down|$. Note as $CoxRossRubinstein$ is an instance of $EqualJumpsBinomialTree$ all we need to calculate is the probability of an up or down move at each node. Including a non-zero drift term would imply the use of the Cox-Ross-Rubinstein Tree with drift which can account for out-of-the-money options.

```
//Base EqualJumpsBinomialTree Class
  class EqualJumpsBinomialTree : public BinomialTree<T> {
      public:
        EqualJumpsBinomialTree(
                        const ext::shared_ptr<StochasticProcess1D>& process,
                        Time end,
                        Size steps)
        : BinomialTree<T>(process, end, steps) {}
        Real underlying(Size i, Size index) const {
            BigInteger j = 2*BigInteger(index) - BigInteger(i);
            // equal jumps centered around x_0
            return this->x0_*std::exp(j*this->dx_);
        }
        Real probability(Size, Size, Size branch) const {
            return (branch == 1 ? pu_ : pd_);
        }
    };
                                    ...
CoxRossRubinstein::CoxRossRubinstein(
                        const ext::shared_ptr<StochasticProcess1D>& process,
                        Time end, Size steps, Real)
    : EqualJumpsBinomialTree<CoxRossRubinstein>(process, end, steps) {

        dx_ = process->stdDeviation(0.0, x0_, dt_);
        pu_ = 0.5 + 0.5*driftPerStep_/dx_;; //P(u)
        //Non-zero driftPerStep_/dx implies we are using Cox-Ross-Rubinstein with drift
        pd_ = 1.0 - pu_; //P(d)
```

Jarrow-Rudd Binomial Tree [11]

The Jarrow-Rudd Binomial Tree is an equal probability tree where for any node S at time t, S can move to either S_{up} or S_{down} at time $t+1$ with equal probabilities such that $P(up) = P(down)$. Note as $JarrowRudd$ is an instance of $EqualProbabilitiesBinomialTree$ all we need to calculate is the size of an up move.

```
// Base EqualProbabilitiesBinomialTree Class
   class EqualProbabilitiesBinomialTree : public BinomialTree<T> {
      public:
        EqualProbabilitiesBinomialTree(
                         const ext::shared_ptr<StochasticProcess1D>& process,
                         Time end,
                         Size steps)
        : BinomialTree<T>(process, end, steps) {}
        Real underlying(Size i, Size index) const {
            BigInteger j = 2*BigInteger(index) - BigInteger(i);
            return this->x0_*std::exp(i*this->driftPerStep_ + j*this->up_);
            //if we chose will incorporate drift term
        }
        Real probability(Size, Size, Size) const { return 0.5; }
    };
                        ........
 JarrowRudd::JarrowRudd(
                         const ext::shared_ptr<StochasticProcess1D>& process,
                         Time end, Size steps, Real)
     : EqualProbabilitiesBinomialTree<JarrowRudd>(process, end, steps) {
         // drift removed
         up_ = process->stdDeviation(0.0, x0_, dt_);
     }
```

Crank-Nicolson Finite Difference Method [11]

In the previous chapter we discussed the formulation of the Crank-Nicolson Scheme. and its versatility in approximating American and European options. As we want to approximate American Options we will have to adjust our boundary conditions accordingly as seen below. Additionally we approximate with 100 time steps and 99 equally spaced grid points.

```
        prices_ = intrinsicValues_;
        controlPrices_ = intrinsicValues_;
        //predefined Finite Difference Operator
        controlOperator_ = finiteDifferenceOperator_;
        //Our Boundary Conditions
        controlBCs_[0] = BCs_[0];
        controlBCs_[1] = BCs_[1];

        operatorSet.push_back(finiteDifferenceOperator_);
        operatorSet.push_back(controlOperator_);
        arraySet.push_back(prices_.values());
        arraySet.push_back(controlPrices_.values());
        bcSet.push_back(BCs_);
        bcSet.push_back(controlBCs_);
        conditionSet.push_back(stepCondition_);
        conditionSet.push_back(ext::shared_ptr<StandardStepCondition>(
                                           new NullCondition<Array>));
        model_type model(operatorSet, bcSet);
       // Approximate our PDE given a certain number of timesteps
        model.rollback(arraySet, getResidualTime(),
                       0.0, timeSteps_, conditionSet);
        prices_.values() = arraySet[0];
        controlPrices_.values() = arraySet[1];
                           ...

        BlackCalculator black(striked_payoff, forwardPrice,
                              std::sqrt(variance), riskFreeDiscount);

        results->value = prices_.valueAtCenter()
            - controlPrices_.valueAtCenter()
            + black.value(); //approximation for value of option
        results->delta = prices_.firstDerivativeAtCenter()
            - controlPrices_.firstDerivativeAtCenter()
            + black.delta(spot); //approximation for delta
        results->gamma = prices_.secondDerivativeAtCenter()
            - controlPrices_.secondDerivativeAtCenter()
            + black.gamma(spot); //approximation for gamma
        results->additionalResults["priceCurve"] = prices_;
```

CHAPTER 4

Analysis of Results

In our first subsection, we will describe the general setup for our experiment. After performing our tests, we will describe broad observations on the accuracy of each pricing method over a variety of time frames. From there we will include a thorough comparison of each pricing method's accuracy along with an analysis on whether a significant difference occurred between the methods. We will then summarize and implement our results into a trading algorithm to either confirm or not confirm our findings on an out-of-sample data set.

4.1 Universe Setup

It is worth noting moving forward that we will denote U is the universe of options contracts we will be analyzing. For each option contract, C_i we will be examining the following properties.

- Strike Price
- Underlying Asset
- Expiration Date
- Expiration Type
- Right
- Bid
- Ask
- Theoretical Price
- Volume

Let’s begin to construct our universe, U. While there are multiple underlying assets one can analyze we chose SPY (SPDR S& P 500 ETF) as it is consistently the highest traded underlying for all options contracts measured in daily volume[12]. Additionally, all option contracts we have access to are American-Style so we enforce that our contracts are American-Style. That being said our analysis of real market data will be restricted to pricing methods for American options. We chose to limit ourselves to monthly contracts however one could analyze different expiration dates such as weekly’s, quarterlies, and LEAPs. It is worth noting exchanges commonly organize the upcoming monthly contracts into three distinct cycles, four months in advance. Monthly contracts for SPY however are regularly offered six months before expiration.

As our goal is to determine the performance of the pricing methods in actuality we choose to complete our tests over different market conditions that occur in real life. For simplicity, however, we will generalize the market into 3 distinct categories; Bullish, Bearish, and Stable Markets. Therefore we will be testing over the following dates in our experiment.

Stable Market

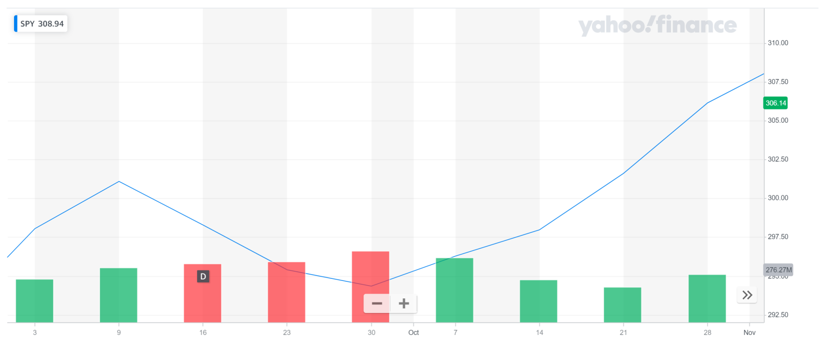


FIGURE 4.1: 9/20/2019 - 10/18/2019

During this month, SPY increased by a total of 1 %. As this time period comprises a relatively stable change in recent times, we classify this as our Stable Market.

Bearish Market

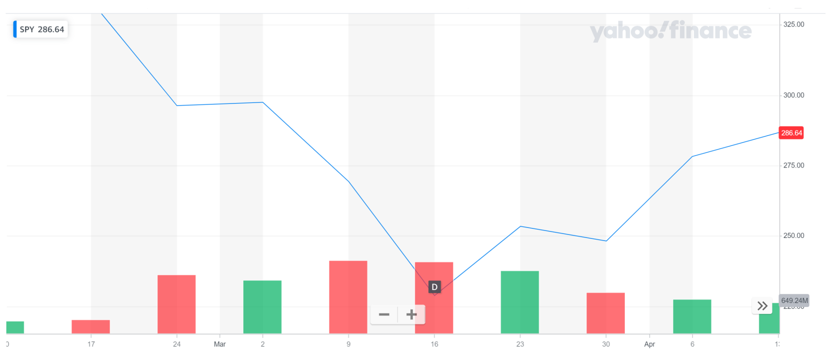


FIGURE 4.2: 2/20/2020 - 3/20/2020

During this month, SPY decreased by a total of 25 %. As this time period comprises of one of the largest declines in SPY in recent times, we classify this as our Bearish Market.

Bullish Market

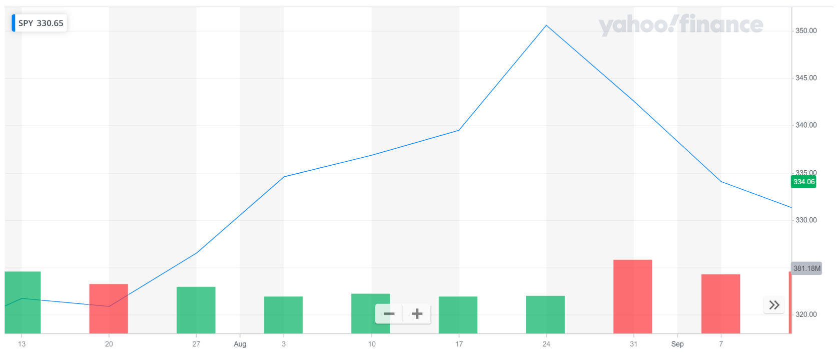


FIGURE 4.3: 7/20/20 - 8/21/2020

During this month, SPY increased by a total of 8 %. As SPY, along with the general market, experienced consistent growth we classify this as our Bullish Market.

One-month time frames for our underlying market trends were chosen with the intention to study SPY option contracts for the entirety of the month leading up to its' expiration date. Therefore, the monthly SPY option contracts we will be analyzing in our tests will have expiration dates of **October 2019, March 2020, and August 2020**.

It is worth noting the restrictions we have specified so far to establish our universe of contracts. Before we can move any further we need one more requirement. Denote for each contract $UnderlyingPrice_{C_i}$ as the underlying's current price. Then $\forall C_i \in U$ we enforce the following

$$Strike(C_i) \in [UnderlyingPrice_{C_i} - 2\sigma_{UP},\ UnderlyingPrice_{C_i} + 2\sigma_{UP}]$$

such that σ_{UP} is the standard deviation of the underlying price.

All together we implement this into our script as such:

```
self.option = algorithm.AddOption(symbol)
self.option.SetFilter(lambda universe: universe.Strikes(-2, 2).Expiration(timedelta(0), timedelta
     (180)))
```

with the additional expiration date check of the following:

```
if str(contract.Expiry.date())!='Target-Expiration_Date':
    continue
    ...
```

4.2 Pricing Method Setup

In our real data evaluation the option pricing method is set in the initialization of each option contract into our universe, as shown below:

```
elif security.Type == SecurityType.Option:
    security.PriceModel = OptionPriceModels.BjerksundStensland()
    #Instance of QLOptionPricingModel
```

Each option pricing method requires universal estimators to be specified:
The first estimator is the underlying volatility estimator. We will use the annualized sample standard deviation of daily returns for the last 63 trading days (which is equivalent to the last 3 months), as shown below:

```
if security.Type == SecurityType.Equity:
    #Perform history call for underlying volatility model
        security.VolatilityModel = StandardDeviationOfReturnsVolatilityModel(60)
        history = self.History(security.Symbol, 61, Resolution.Daily)
                                             ....
#Underlying volatiltiy estimator for each contract is the underlying volatility model
```

```
Option.ConstantQLUnderlyingVolatilityEstimator.Estimate(...)=option.Underlying.VolatilityModel.
     Volatility;

```

For each pricing method the underlying dividend estimator is the constant dividend yield estimator which returns a flat estimate of zero dividend yield, as shown below:

```
   Option.ConstantQLDividendYieldEstimator.Estimate(...)=0
```

For each pricing method the underlying riskfree rate estimator is the constant risk-free rate estimator which returns a flat estimate of the risk-free rate, as shown below:

```
    Option.ConstantQLRiskFreeRateEstimator.Estimate(...)=0.01
```

Note one could alter the estimators by introducing different look-back periods, flat estimates, or stochastic processes.

4.3 Real Data Evaluation

Our next step is to run a backtest over our time frame, B_i where we start at $B_{i,0}$ and iterate until we reach $B_{i,j}$, such that j is the final minute of the day at which our chosen option contract expires at.

For each contract C_i we calculate the mid-spread (MS) at time j as the following:

$$MS(C_{i(j)}) = \frac{Bid(C_{i(j)}) + Ask(C_{i(j)})}{2}$$

Assuming we have already calculated the theoretical value of the contract (Th) we can also calculate relative error as the following:

$$error_{C_i(j)} = \frac{MS(C_{i(j)}) - Th(C_{i(j)})}{MS(C_{i(j)})}$$

Now that we have the relative error at time step k for each contract we can extend this to the full month.

Below one can see the results from our real data evaluation. For conciseness, we have only included the three calls and puts with the largest amount of trading volume over the chosen time frames. Each entry represents the average relative error over the time frame between the mid-spread and theoretical value, calculated with the respective pricing method.

We first remark that it is clear we have a downward bias with our default estimators. In both calls and puts, we have underestimated the theoretical value of the contract leading to a positive error value. We also point out that this bias is prevalent in all of our time frames, however more significant over the Bearish time frame. As of right now, it is not clear whether the bias arises from our volatility, risk-free rate, or dividend yield estimator.

FIGURE 4.4: **Average Relative Error of Option Contracts**

Contract	Vol	B-A-W	B-S	C-N	C-R-R	J-R
Stable Market						
$9/20/19 \rightarrow 10/18/19$						
191018C00302000	63376905	2.678623792	2.678623792	2.678623792	2.67437879	2.674454546
191018C00301000	59370390	2.012332278	2.012332278	2.012332278	2.012688603	2.012431568
191018C00300000	27230486	2.474583479	2.474583479	2.474583479	2.473374215	2.473545016
191018P00296000	49791381	4.312775181	4.312987549	4.312442347	4.311959884	4.311950649
191018P00295000	19021435	4.333976181	4.334243502	4.333624632	4.333363144	4.333300709
191018P00290000	17972131	4.219581335	4.219811531	4.219360764	4.218265365	4.218304149
Bearish Market						
$2/20/20 \rightarrow 3/20/20$						
200320C00288000	151195022	18.95265	18.95265	18.95265	18.95318	18.95332
200320C00280000	41217042	15.14999	15.14999	15.14999	15.14947	15.14960
200320C00317000	40335972	8.08967	8.08967	8.08967	8.08864	8.08863
200320P00312000	121745412	12.17169	12.17186	12.17146	12.17068	12.17081
200320P00327000	68618846	13.73884	13.73908	13.73828	13.73820	13.73826
200320P00230000	36530004	14.09581	14.09581	14.09580	14.09590	14.09577
Bullish Market						
$7/20/20 \rightarrow 8/21/20$						
200821C00341000	43059413	0.39444	0.39444	0.39444	0.39173	0.39185
200821C00340000	38731467	2.0782	2.0782	2.0782	2.0785	2.07909
200821C00339000	21222348	2.26745	2.26745	2.26745	2.26686	2.26741
200821P00320000	103406088	6.45580	6.45610	6.45557	6.45705	6.45693
200821P00335000	45367644	5.08538	5.08553	5.08508	5.08412	5.08437
200821P00320000	40093401	6.22273	6.22297	6.22241	6.2200	6.22015

Vol: Total Traded Volume
B-A-W: Barone-Adesi-Whaley Analytic Approximation [5]
B-S: Bjerksund-Stensland Analytic Approximation [7]
C-N : Crank-Nicolson Finite Difference Method [4][6]
C-R-R: Cox-Ross-Rubinstein Binomial Tree [2]
J-R: Jarrow-Rudd Binomial Tree [3]

We now test whether there is a significant difference between our five chosen pricing methods. Performing a One Way ANOVA Test[17] such that our null hypothesis is that no difference exists between the pricing methods has the following results:

FIGURE 4.5: **ANOVA Test for Stable, Bearish, and Bullish Markets**

Time Frame	F Statistic	Prob	Conclusion
Stable Market	$1.5 * 10^{-6}$	1	Do Not Reject H_0
Bearish Market	$1.7 * 10^{-7}$	1	Do Not Reject H_0
Bullish Market	$4.3 * 10^{-7}$	1	Do Not Reject H_0

From this, it is clear there does not exist a difference in our pricing methods at a significant level. In other words, we are indifferent to the method used to calculate the theoretical value for an arbitrary option contract.

4.4 Implementation

Lastly, we present an implementation of our findings into a trading algorithm. Our out-of-sample data is from November 2020 to January 2021 which spans a long enough time frame to experience bullish, bearish, and stable points in time. Running a backtest over our time frame, we produce the following relative average errors.

FIGURE 4.6: **Average Relative Error of Option Contracts**

Contract	Vol	B-A-W	B-S	C-N	C-R-R	J-R
Out-of-Sample data set						
$11/15/20 \rightarrow 1/15/21$						
210115C00380000	172887852	3.118747568	3.118747568	3.118747568	3.120401288	3.120199295
210115C00381000	105375385	2.953607075	2.953607075	2.953607075	2.954763296	2.954575036
210115C00382000	79773400	3.059143999	3.059143999	3.059143999	3.059199757	3.059308856
210115P00368000	52089251	10.63383405	10.6341782	10.63347105	10.63354459	10.63332738
210115P00375000	29945288	7.0013056371	7.001534286	7.000945899	6.999511168	6.999479611
210115P00365000	25779894	10.77798391	10.77834935	10.77764943	10.77656601	10.7765485

Vol: Total Traded Volume
B-A-W: Barone-Adesi-Whaley Analytic Approximation [5]
B-S: Bjerksund-Stensland Analytic Approximation [7]
C-N : Crank-Nicolson Finite Difference Method [4][6]
C-R-R: Cox-Ross-Rubinstein Binomial Tree [2]
J-R: Jarrow-Rudd Binomial Tree [3]

Again we test whether there is a significant difference between our five chosen pricing methods. We perform a One Way ANOVA Test[17] such that our null hypothesis is that no difference exists between the pricing methods:

FIGURE 4.7: **ANOVA Test for Out-of-Sample data set**

Time Frame	F Statistic	Prob	Conclusion
Out-of-Sample data set	$5.8 * 10^{-7}$	1	Do Not Reject H_0

It is clear there does not exist a difference in our pricing methods at a significant level for our out of sample data set. However, it is worth noting that the errors once again exhibit a downward bias with our chosen default estimators. That being said our trading algorithm would consistently buy all of the options in our universe as the theoretical price is lower than the current market price. As our universe consisted of both puts and calls of varying strikes, it is unlikely for this to be a profitable strategy. Nonetheless, we have confirmed our findings that we are indifferent to the method used to calculate the theoretical value for an arbitrary option contract.

CHAPTER 5

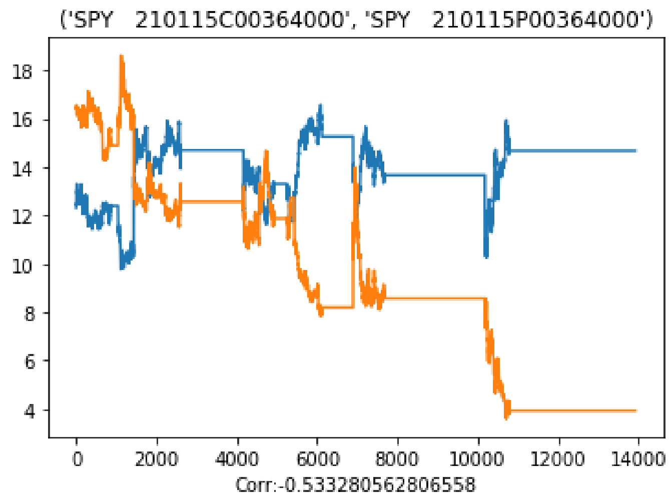
Conclusion

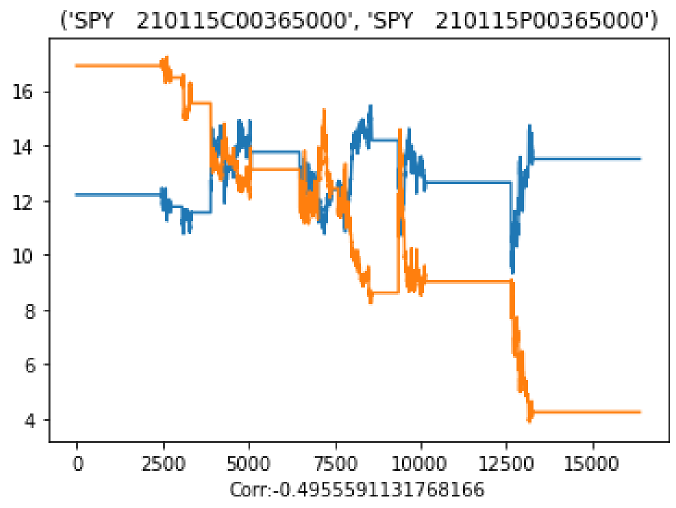
This thesis analyzed and applied the most common pricing methods for European and American options to real financial market data in order to determine the accuracy of each method. We started off by describing what an options contract is along with some common terminology. In **Chapter 2** we theoretically discussed the methods and relevant mathematics. Moving on to **Chapter 3** we provided the scripts that we implemented into our experiment in **Chapter 4**. From our tests, we concluded in both an in-sample and out-of-sample data set that the pricing methods are not significantly different from one another with respect to the accuracy of calculating the theoretical value.

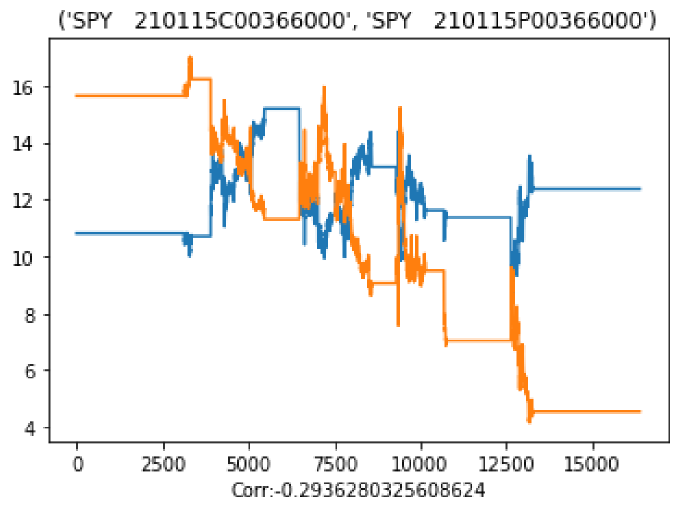
5.1 Looking Ahead

Looking ahead there is still a lot more to do. For one, in all of our tests, we encountered a significant downward bias in our estimations of the theoretical value, which need further justification and improvement as part of my future work. As this bias occurred across different pricing methods, we assume it is related to the default estimators we specified in **Chapter 4.2**. While we used average and flat rate estimators for convenience, these are not necessarily the best choices. For instance, both GARCH models and stochastic processes are often used to estimate the volatility of the underlying. One point to make is if the bias is minimized we might conclude there is a significant difference between the accuracy of the pricing methods. However, as the difference between theoretical prices is usually smaller than 0.01, the smallest increment to trade options, we do not think it is relevant to pursue this further.

Another surprise we discovered were inverse relationships in the relative error of option straddles. An option straddle consists of a call and put contract with the same strike and expiration. Let's denote C as the time series of relative error for an arbitrary call contract at a minute j and likewise P for the time series of the put contract. Then plotting the option straddle we noticed the error to be inversely correlated, shown below.







This observation persisted across all of our chosen time frames with contracts of varying liquidity. Examining further we saw that the relationship between the straddle seemed to be more inverted in times of lower volatility than in times of higher volatility. In times of extreme volatility, some straddles even displayed positive correlation ratios. Intuitively, we believe this makes sense as during times of low volatility, there is less uncertainty, keeping the relationship between the price of puts and calls balanced. In times of higher volatility however, people are unsure of the market direction therefore, reflecting in the prices of puts and calls.

Bibliography

[1] Fischer Black and Myron Scholes. 'The Pricing of Options and Corporate Liabilities'. In: *Journal of Political Economy* 81.3 (1973), pp. 637–654. ISSN: 00223808, 1537534X. URL: http://www.jstor.org/stable/1831029.

[2] John C. Cox, Stephen A. Ross and Mark Rubinstein. 'Option pricing: A simplified approach'. In: *Journal of Financial Economics* 7.3 (1979), pp. 229–263. ISSN: 0304-405X. DOI: https://doi.org/10.1016/0304-405X(79)90015-1. URL: https://www.sciencedirect.com/science/article/pii/0304405X79900151.

[3] Robert Jarrow and Andrew Rudd. 'Approximate option valuation for arbitrary stochastic processes'. In: *Journal of Financial Economics* 10.3 (1982), pp. 347–369. ISSN: 0304-405X. DOI: https://doi.org/10.1016/0304-405X(82)90007-1. URL: https://www.sciencedirect.com/science/article/pii/0304405X82900071.

[4] Robert Geske and Kuldeep Shastri. 'Valuation by Approximation: A Comparison of Alternative Option Valuation Techniques'. In: *The Journal of Financial and Quantitative Analysis* 20.1 (1985), pp. 45–71. ISSN: 00221090, 17566916. URL: http://www.jstor.org/stable/2330677.

[5] Giovanni Barone-Adesi and Robert E. Whaley. 'Efficient Analytic Approximation of American Option Values'. In: *The Journal of Finance* 42.2 (1987), pp. 301–320. ISSN: 00221082, 15406261. URL: http://www.jstor.org/stable/2328254.

[6] John Hull and Alan White. 'Valuing Derivative Securities Using the Explicit Finite Difference Method'. In: *The Journal of Financial and Quantitative Analysis* 25.1 (1990), pp. 87–100. ISSN: 00221090, 17566916. URL: http://www.jstor.org/stable/2330889.

[7] Petter Bjerksund and Gunnar Stensland. ‘Closed-form approximation of American options’. In: *Scandinavian Journal of Management* 9 (1993), S87–S99. ISSN: 0956-5221. DOI: https://doi.org/10.1016/0956-5221(93)90009-H. URL: https://www.sciencedirect.com/science/article/pii/095652219390009H.

[8] Dietmar Leisen and Matthias Reimer. ‘Binomial Models for Option Valuation—Examining and Improving Convergence’. In: *Applied Mathematical Finance* 3 (Mar. 1995), pp. 319–346. DOI: 10.1080/13504869600000015.

[9] Yisong “Sam” Tian. ‘A flexible binomial option pricing model’. In: *Journal of Futures Markets* 19.7 (1999), pp. 817–843. DOI: https://doi.org/10.1002/(SICI)1096-9934(199910)19:7<817::AID-FUT5>3.0.CO;2-D. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/%28SICI%291096-9934%28199910%2919%3A7%3C817%3A%3AAID-FUT5%3E3.0.CO%3B2-D. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291096-9934%28199910%2919%3A7%3C817%3A%3AAID-FUT5%3E3.0.CO%3B2-D.

[10] Randall Leveque. *Finite Difference Methods for Ordinary and Partial Differential Equations*. SIAM, 2007. ISBN: 978-0-89871-629-0.

[11] Apache Software Foundation. *QuantLib*. Version 1.21. 15th Mar. 2021. URL: https://www.quantlib.org/.

[12] Barchart. *Most Active Etfs Options*. URL: https://www.barchart.com/options/most-active/etfs. (accessed: 03.09.2021).

[13] CBOE. *Exchange Rule Book*. URL: https://markets.cboe.com/us/options/membership/#options-exchange-rules. (accessed: 03.09.2021).

[14] Jeff Clark. *Option Symbol Decomposition*. URL: https://www.jeffclarktrader.com/glossary/. (accessed: 03.09.2021).

[15] Phil Goddard. *Option Pricing - Alternative Binomial Models*. URL: https://www.goddardconsulting.ca/option-pricing-binomial-alts.html#jr. (accessed: 03.09.2021).

[16] Phil Goddard. *Option Pricing Using The Crank-Nicolson Finite Difference Method*. URL: https://www.goddardconsulting.ca/about.html. (accessed: 03.09.2021).

[17] Will Kenton. *Analysis of Variance (ANOVA)*. URL: https://www.investopedia.com/terms/a/anova.asp. (accessed: 03.09.2021).

[18] Macro Option. *How Binomial Trees Work in Option Pricing*. URL: https://www.macroption.com/binomial-trees/. (accessed: 03.09.2021).

[19] Wikipedia. *Option Symbol*. URL: https://en.wikipedia.org/wiki/Option_symbol. (accessed: 03.09.2021).

1 Software

The software for my tests can be found on this GitHub Repository.

LEAN is the open-souce algorithmic trading engine used for the research and back-testing environment. The data for options was provided through **QuantConnect**, originally provided by **Algoseek**. LEAN's License is here.

QuantLib was the open-source library used for the pricing method scripts presented in **Chapter 3** along with in the original code created to execute out tests. QuantLib's License is here.