# Notes for the Write-Up

## Summary (250-300)

This project covers two main methods to determine the pricing of options contracts. These two methods were Monte Carlo Simulation and the Binomial Lattice Approach. The focus was on 4 types of options, primarily, Asian, Lookback and Floating lookback, and American options. In all 4 types of options, the call and put prices were assessed (American is assessed for put option only). These options are briefly described below:

Asian Options:

These are a type of exotic option in which payoff depends on the average underlying stock price over the option lifetime rather than the underlying stock price at maturity for regular European options.

Lookback options:

These options have a fixed strike price, but at maturity, the contract owner can look into the history of the underlying price action throughout the option lifetime and choose the most favorable price to exercise at.

Floating lookback options:

These options have to be exercised at the stock price at maturity time, but the strike price can be set to the most favorable price that the underlying asset reached throughout the option lifetime.

American Options:

American options are identical to the classical European options in every regard except they also have the ability to exercise prior to maturity. This makes the process for their valuation a bit more complicated than the options discussed above.

In terms of pricing methods Monte Carlo simulation provides a more random method to allow for the incorporation of the random nature of the underlying stock price. This process is based on the geometric Brownian motion. This process was designed so that as time increments become infinitesimally small, Monte Carlo Simulation becomes a continuous simulation of stock price path. The binomial lattice approach is based on discrete methodologies to compute the price of the stock along various possible paths using risk neutral probabilities.

## Primary inputs

No dividend payments assumed.

1. Present stock price = 100 (100$)

2. Risk-free rate = 0.02 (2%)

3. Volatility = 0.25 (25%)

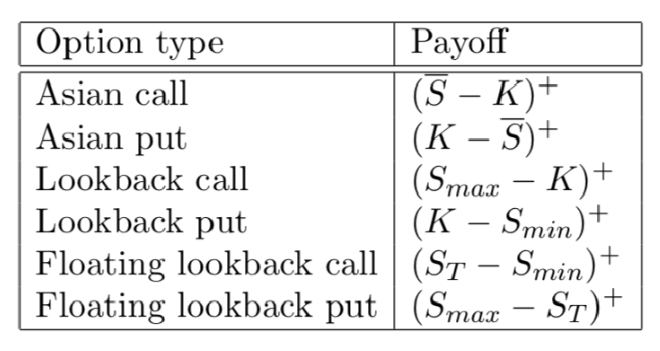
4. Defined incremental time interval = 1 week (1/52 in years)

5. Maturity of option: 2/12 (2 months)

6. Continuous compounding was assumed for processes pertaining to the Time Value of Money (i.e., discounting back in time and compounding forward in time).

## Pricing method for the options

Table is from the Pricing American Options Using Monte Carlo Simulation paper by Nairn McWilliams



## Monte Carlo Method (500-750)

The process of option pricing is done based on the price of the underlying asset. To price the underlying asset, in this case a stock, we utilize a process derived from the multiplicative model, called the geometric Brownian motion. When going from a discrete model like multiplicative model to continuous model like geometric Brownian Motion, Brownian Motion (which is based on the Wiener process) is used to convert the discrete model to continuous. Further conversion of the model is done with Ito's lemma to bring it to the formulation used in this case. This method uses the Wiener process as a basis for randomness.

The Monte Carlo Method is a method derived from utilizing the standard normal random variable eta (originally from the Wiener process) in the formula for the stock price path. We can generate a different eta for each step to generate each stock price along the path. Essentially, this is taking on the Markovian assumption for the stock price (where future stock price only depends on current stock price and does not need to consider historical stock price) and assumes a random walk.

This whole model utilizes the risk-free rate throughout the process (discounting, geometric Brownian motion etc.). This is possible due to the risk neutral assumption (assumption that the world is risk neutral i.e., investors don't care about risk).

How the process works:

This method requires the following inputs:

1. Present stock price

2. Risk-free rate

3. Volatility

4. Defined incremental time interval (which makes it somewhat discrete in nature but is a part of the accepted error in the solution)

5. Maturity of the option

It begins at the present and takes "random steps” which assumes a log-normal distribution that has a mean of (risk-free rate - volatility^2/2) multiplied by dt and a standard deviation of volatility multiplied by the square root of dt. The random steps each move forward by a time interval of dt until the maturity date of the option is reached. As dt approaches 0, the simulation becomes continuous, however, this project has a dt of 1 week, so the simulations were discrete (as mentioned before). Using these simulations for the underlying stock price, the option payoff is appropriately found and discounted accordingly to find the option price.

In the processes coded in this document, 1000 Monte Carlo Simulations (considering the project minimum of 250 simulations) (what are called 'paths') were created per option valuation to generate a large enough sample size to more accurately predict options pricing. The option price was taken as the arithmetic mean of the present value of the payoff at all paths.

Specifically for the American Put Option, an Optimal Exercise Boundary was generated using backwards induction in order to compute the pricing for the American Put Option. To do this, all price actions at all incremental time steps for all possible Monte Carlo paths were considered for early exercise, and how the resulting payoff would compare with exercising later in the future, perhaps at a more "optimal" time. Adjusted for time value of money.

The visualizations for the methods are included in the appropriate section either Exotic or American. These help the understanding of the simulation process.

## Lattice Method (500-750)

The Lattice method is based on the multiplicative model. At each node there are 2 paths to process every change in time increment: up or down in stock price. These prices are determined by the u and d multipliers. The multiplicative model has some advantages over an additive model, the two main advantages being superior computational efficiency compared to the additive model and the stock price being unable to drop below 0, which is a realistic restriction for the stock price.

Multipliers u and d are determined by the 2 moments of the multiplicative model, the expected value and the variance being equated to the Geometric Brownian Motion moments to allow for a discrete model to be as continuous as possible. This allows us to find the following formulas for the u and d multipliers.

u = exp(sigma \* sqrt(lattice\_unit\_time))

d = 1/u

Where lattice\_unit\_time represents an individual time increment in years, and sigma represents volatility.

Using this model, we can also note that u is always greater than 1 and d is always less than 1 for the process to be executed properly. Additionally, the distance between u and d from 1 are identical.

The lattice model is primarily discrete due to the nature of its structure, as the price actions in-between each time increment are binomial.

A logical assumption to make for options pricing methods is to assume a risk-neutral world. Therefore, physical probabilities defined for classical methods of discretizing by Binomial Tree are not applicable in options pricing. To remain consistent with this assumption, the utilization of the risk-neutral probabilities fit for this scenario. The upper and lower probabilities are determined by the following formula:

p = (exp(r\*lattice\_unit\_time)-d)/(u-d)

Where p represents the risk-neutral probability of the stock going up in the next time increment, and continuous compounding was assumed, so an exponential of Euler’s number was used in the numerator of the risk-neutral probability rather than discrete compounding assumptions.

At each fork of the lattice structure, the subsequent time increment stock values are calculated using the probability and the u and d multipliers. This process is repeated until maturity and so it is fit to use a recursive feature in this lattice model code.

Once the final stock price for each path is obtained, it is possible to compute the payoff for the option using the table indicated at the beginning of the notebook under “Pricing method for the options”. Each incremental path multiplies its previous stock price by either u or d, and also multiplies its previous probability by either p or (1-p). Both the probabilities and the stock prices are kept track of throughout the process. To compute the exotic options pricing, the payoff of each path was multiplied by their own respective probabilities and summed together to find the expected payoff of the option at the maturity date. It is then discounted at the risk-free rate to present time to find the final options price.

For the American Put pricing, the lattice method described up to this point was all applicable, but insufficient to complete the options pricing process. In order to account for the early exercise option, backwards induction was required at each time increment to determine the best time to exercise the option. Starting at the maturity date and moving backwards in time, each time increment compared the optimal payoff from a future date, discounted to the current time increment, with the option of exercising at the current time increment. Whichever gave the highest payoff became the new optimal payoff.

##Visualizations (150-300)

This first visual helps us understand how the paths look in the Monte Carlo simulation. These paths are considered individually for payoff then considered all together for the final picture of the call or put valuation. The timeline should be noted as taken in weeks, 9 weeks in 2 months (rounding is involved for integer values).

This second visualization is for the optimal exercise boundary for an American put option. This shows how the best time to exercise the bond i.e., when the value of holding the bond is no longer worth it in comparison to the payoff that is possible at this point. We can see over time this boundary grows till the strike price itself, in the case of the put considered.

This table is a sample of 3 paths in the lattice randomly selected to show the process the stock price goes through over the 9-week process. Considering the initial price of $100, we can see the prices go up and down and in the end, be above, below or at the initial value. This table is a good overview of the intermediate steps the lattice goes through for the main underlying stock valuation.

## Comparison Between the Two Methods and Discussion (250-300)

The two methods both provide a good way to price options. We see that options priced by the lattice structure are a bit different than the options priced by Monte Carlo structure. In this we have the same unit time so the normally discrete structure of the lattice method versus the more continuous structure of the Monte Carlo method aren’t too much of a concern.

The Monte Carlo method is more accurate in the sense that it is stochastic in nature, so it takes into account more of the randomness involved in the stock price. We can possibly see this as more realistic to the real world if the assumptions like Markov process being true are as well. It is able to cover the spectrum of the stock price cases well, in comparison to the lattice approach. This makes sense since at each point it is not limited to the up or down multipliers. The stochastic nature of the Monte Carlo Simulation creates uncertainty that needs to be accounted for when computing the final output. As a result, it is recommended to include a confidence interval when determining options pricing with stochastic processes.

The Lattice method is a deterministic process, which means the pricing does not change every time the program is run. This is an advantage of the lattice method but also a disadvantage. Since there is an inherent uncertainty and randomness associated with the process, the lattice structure is too basic with only 2 forks at every node and set probabilities and multipliers. This process does not cover as much of the extreme cases as the Monte Carlo method does, but the final outcomes are close and within a given interval of each other. This means that either pricing process given that the dictated assumptions hold, will work.

## Lessons Learned (Need 250 at least)

The pricing of the American put in either case is very tedious and complicated due to the ability to exercise the option earlier than the maturity date. It involves optimizing the timing of when the payoff is optimal by comparing different payoff options at each time increment. The exotic options on the other hand are quite simple by comparison as there is only one option contract exercise occurrence at the maturity date. This simplifies the problem as it does not require comparisons with other exercise options to find the optimal.

The Monte Carlo method is very intuitive but needs a large number of computations to be considered ‘accurate’ to a certain degree. As a result, a large number of paths with a large number of time steps is required to ensure that the process is as close to continuous as possible. This is especially the case in pricing American put options as many Monte Carlo Simulation paths with infinitesimally small time-increments are required to achieve convergence, which ends up resulting in a feasible and consistent Optimal Exercise Boundary. Challenges with attempting to create Monte Carlo Simulations with a large number of paths and time steps involve lack of memory and high computational time and effort. This trade-off between accuracy and computational time needs to be considered. In computing the American put option using Monte Carlo Simulation for this particular project, it was observed that because the time increments were 1 week in length, and there were only 9 total increments until maturity, the plot of the Optimal Exercise Boundary was inconsistent and fragmented with many noticeable kinks.

**## Explanation of Code (500 -750)**

The pricing is done by a function. The inputs are the given initial conditions. The OPtype variable allows for the user to input the type of the option, so the proper payoff method will be used. A plot of paths is also optional.

First step is to calculate the price according to the underlying stock price which is dictated by the geometric Brownian motion (GBM) after applying Ito's lemma to it. We are assuming a risk neutral world, and so the application of r in the geometric Brownian motion and the discounting process, is done so with this understanding. The double loop in the function through each step point (inner loop) for every path (outerloop)

After the underlying stock price is determined according to GBM, we can calculate important properties of the path like average price, minimum and maximum price. These are utilized according to the type of option being priced. For example, the Asian option utilizes the average stock path price in comparison to the strike price to determine the payoff.

The discounting of this payoff of the path using the risk-free rate is done (with continuous discounting utilized). The average of all paths present value of the payoff determines the put and call value for the options.

The below code creates a custom function that performs Monte Carlo simulations given initial conditions dictated previously.

This was done using a nested for loop where each path is created in the inner loop in list form, and nested into another list on the outer loop to act similarly to a MATLAB array.

Additionally, a plot of the Simulation was created for visualization purposes.

This custom function acts as a pipeline for Monte Carlo Simulations and will be used in another custom function that ties all of the American Put Option pricing computation together.

The below code is a custom function that determines the Optimal Exercise Points for a single given time. This will be used in another custom function that ties all of the American Put Option pricing computation together inside a loop that will iterate for all time increments, and the Optimal Exercise Points will be stored for each time increment to create an Optimal Exercise Boundary.

This is done through using backwards induction by comparing the payoff of Optimal Exercising at a future time versus exercising at a previous time increment.

This code follows the pseudo-code steps provided in page 15 of "Pricing American Options using Monte Carlo Simulation" by Nairn McWilliams provided in the course. Only the steps relevant to pricing were followed as the question only asked for price, not mean optimal stopping times or confidence intervals.

This is a custom function that ties the other functions together to determine the American Put Option pricing. It begins by performing Monte Carlo Simulation, and then initializes the backwards induction process by determining the stock prices, put prices, and optimal exercise price for all paths at maturity date. Subsequently, the code performs the backward induction process by looping over the second last time increment until time = 0 and every loop uses the custom function directly above. Finally, the function plots the Optimal Exercise Boundary.

The binomial lattice method is done simply by following the lattice paths. The probabilities of the upper or lower path are determined by the risk neutral probabilities (not the physical probabilities calculated for a typical lattice) since this is an option pricing case done in a risk neutral world. The u and d scaling values at each fork are determined by the basic lattice model, which takes into account the GBM moments to allow the model to be as continuous as possible despite being discrete.

The main function follows the idea that if the time is below the maturity prescribed, then the function will call onto itself appending the time and probability for the upper and lower parts accordingly. The options are priced accordingly, where the probability for each path is taken into account.

The final wrapper function is to discount the values (continuous discounting to keep it as consistent with the previous methods) and grab the proper values for each option for output.