# Assignment4: Monte Carlo Option Pricing

## Monte Carlo Option Pricing

**By: Sarah Lazio-Maimone**

### Introduction

Options are contracts that allow for an asymmetric buy or sell of an asset at a predetermined fixed price. The execution of buying an option is considered a 'call', whereas, selling an option is a 'put'. Due to the asymmetrical nature of these options, it can be difficult to truly understand and determine the appropriate value. Many investments and corporate finance firms use a computational approach to assist in deeming fair prices. There are many different techniques to implement option value methods, including: binomial trees, closed form solutions, algorithms and Monte Carlo simulations. Binomial trees are a way of calculating potential paths of stock prices using a discrete-time model.

First function is a stock price generator. Given inputs of: stock price, maturity, steps, sigma, interest rate, iterations and programming seed, the function will return an output containing: a matrix of stock prices with rows forming the price path, mean ending price, mean max price, and mean min price for the porfolio.

Second function provides an option price estimation on top of the outputs from the previous function. The option price estimation returns a string 'callorput', the strike price, and another string 'exotic' that identifies which option type was selected. There are 5 different types of options available to determine stock payoffs: 'floatlookback' is the difference between the last price and the min or max price, 'fixedlookback' is the difference between the min or max price and the exercise price, 'asianarithmetic' is the arithmetic average, 'asiangeometric' is the geometric mean, and "assetornothing' is the fixed payoff if the asset is below the strike price.

#### Load necessary packages

*psych* - "A general purpose toolbox for personality, psychometric theory and experimental psychology. Functions are primarily for multivariate analysis and scale construction using factor analysis, principal component analysis, cluster analysis and reliability analysis, although others provide basic descriptive statistics."

```
#install/load packages
pacman::p_load(psych)
```

### Stock Price Generation

Develop function to formulate given inputs that will return matrix of stock prices with rows forming price path, mean ending price, mean max price, and mean min price.

```
main <- function(price, maturity, steps, sigma, int, iter, seed){
#pricing inputs
S0 <- price #current stock price
r <- int #interest rate
sig <- sigma #standard deviation/sigma
T <- maturity #time in yrs
```

```
m <- steps #steps
t <- T/m #time to maturity

#generate matrix of N(0,1) random numbers
n <- iter
nvars <- m*n
set.seed(seed)
e <- as.vector(rnorm(n=nvars, m=0, sd=1))
E <- matrix(e,nrow = n,ncol = m)

#convert random variables to multiplicative factor
fac <- exp((r-.5*sig^2)*t+sig*sqrt(t)*E)

#create stock price path
fac1 <- t(apply(fac,1,cumprod)) #cumulative returns

St <- fac1*S0 #multiplied by price


#create summary means
x <- as.numeric(ncol(St))
meanend <- mean(St[,x])
meanmax <- mean(apply(St, 1, max, na.rm=TRUE))
meanmin <- mean(apply(St, 1, min, na.rm=TRUE))

#return final output as list with St hidden
final <- list("First Two Price Paths"=head(St, n=2), "Mean Ending Price" = meanend,
              "Mean Max Price" = meanmax , "Mean Min Price" = meanmin)

return(final)
}
```

Baseline inputs: (default value in parentheses): Stock price (20), Maturity (0.25), Steps (20), Sigma (0.4), interest rate (0.03), iterations (1000), seed (12)

The output below will display the first two price paths and the summary measures

```
#inputs
df1 <- main(20, 0.25, 20, 0.4, 0.03, 1000, 12)
#output
df1
```

```
## $`First Two Price Paths`
##          [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]     [,8]
## [1,] 18.70693 16.70236 17.43784 16.95179 19.34127 18.78898 18.94030 18.97158
## [2,] 21.44819 21.93510 23.12699 22.27332 22.35005 20.57222 19.18193 20.07497
##          [,9]    [,10]    [,11]    [,12]    [,13]    [,14]    [,15]    [,16]
## [1,] 20.53006 21.30370 19.69631 18.01130 16.09071 16.65925 15.79453 15.67380
## [2,] 20.68644 20.83912 20.27498 18.71168 17.89366 15.90486 16.30661 15.50645
##         [,17]    [,18]    [,19]    [,20]
## [1,] 14.92605 14.56841 13.32745 14.19706
## [2,] 16.94518 16.98628 16.41634 16.42661
##
## $`Mean Ending Price`
## [1] 20.20476
##
## $`Mean Max Price`
```

```
## [1] 22.8527
##
## $`Mean Min Price`
## [1] 17.55858
```

## Option Price Estimation

Develop function using all inputs from previous function plus additional inputs identifying call or put, strike price, and identifying option type.

```
main2 <- function(price, maturity, steps, sigma, int, iter, seed, callorput, strike,
         exotic){
  X <- strike

  #pricing inputs
  S0 <- price #current stock price
  r <- int #interest rate
  sig <- sigma #standard deviation/sigma
  T <- maturity #time in yrs
  m <- steps #steps
  t <- T/m #time to maturity

  #generate matrix of N(0,1) random numbers
  n <- iter
  nvars <- m*n
  set.seed(seed)
  e <- as.vector(rnorm(n=nvars, m=0, sd=1))
  E <- matrix(e,nrow = n,ncol = m)

  #convert random variables to multiplicative factor
  fac <- exp((r-.5*sig^2)*t+sig*sqrt(t)*E)

  #create stock price path
  fac1 <- t(apply(fac,1,cumprod)) #cumulative returns

  St <- fac1*S0 #multiplied by price


  #Float Look Back
  #call is max(St-min(St),0)
  #put is max(max(St)-St,0)
  if(exotic=="floatlookback"){
    if(callorput=="call"){
      f <- ifelse(St>apply(St,1,min),St-apply(St,1,min),0)
      pv <- mean(exp(-r*T)*f)
    }
    if(callorput=="put"){
      f <- ifelse(apply(St,1,max)>St,apply(St,1,max)-St,0)
      pv <- mean(exp(-r*T)*f)
    }
  }

  #Fixed Look Back
  #call is max(max(St)-K,0)
  #put is max(K-min(St),0)
  if(exotic=="fixedlookback"){
    if(callorput=="call"){
      f <- ifelse(apply(St,1,max)>X,apply(St,1,max)-X,0)
```

```
      pv <- mean(exp(-r*T)*f)
    }
    if(callorput=="put"){
      f <- ifelse(X>apply(St,1,min),X-apply(St,1,min),0)
      pv <- mean(exp(-r*T)*f)
    }
  }

  #Asian Arithmetic
  #call is max(0,mean(St)-X)
  #put is max(0, X-mean(St))
  if(exotic=="asianarithmetic"){
    MeanSt <- apply(St,1,mean)
    if(callorput=="call"){
          f<- ifelse(MeanSt>X, MeanSt-X, 0)
          pv <- mean(exp(-r*T)*f)
      }
    if(callorput=="put"){
          f<- ifelse(X>MeanSt, X-MeanSt, 0)
          pv <- mean(exp(-r*T)*f)
      }
  }


  #Asian Geometric
  #call is max(0,geomean(St)-X)
  #put is max(0, X-geomean(St))
  if(exotic=="asiangeometric"){
    MeanSt <- apply(St,1,geometric.mean)
    if(callorput=="call"){
          f<- ifelse(MeanSt>X, MeanSt-X, 0)
          pv <- mean(exp(-r*T)*f)
      }
    if(callorput=="put"){
          f<- ifelse(X>MeanSt, X-MeanSt, 0)
          pv <- mean(exp(-r*T)*f)
      }
  }

  #Asset or Nothing
  #call is e^(-r(T-t)) E[St|St>K]
  #or      e^(-r(T-t)) ln(S0) +(r-sig^2/2)T
  #put is fixed payout of market price
  if(exotic=="assetornothing"){
    x <- as.numeric(ncol(St))
    if(callorput=="call"){
          f<- ifelse(St[,x]>X, St[,x], 0)
          pv <- mean(exp(-r*T)*f)
      }
    if(callorput=="put"){
          f<- ifelse(X>St[,x], X, 0)
          pv <- mean(exp(-r*T)*f)
      }
  }

  return(pv)
}
```

Baseline inputs: All inputs from function 1 PLUS a (i) a string 'callorput' (call), (ii) strike (20), and (iii) a string 'exotic'

The output below will display tabulated present value results for each of the five option types as both call or put type, plus additional input values

```
#all inputs from function above, can be changed as needed
price <- 20
maturity <- 0.25
steps <-  20
sigma <- 0.4
int <- 0.03
iter <- 1000
seed <- 12
#new inputs for new function
exotic <- c("floatlookback", "fixedlookback", "asianarithmetic", "asiangeometric",
         "assetornothing")
callorput <- c("call", "put")
strike <- 20
output <- matrix(nrow=10, ncol=4)
pv <- 0
y <- 1

#loop through each exotic type and call
for (i in seq_along(exotic)){
  output[y,1] <- exotic [i]
  output[y,2] <- callorput[1]
  output[y,3] <- strike
  output[y,4] <- main2(price, maturity, steps, sigma, int, iter, seed, callorput[1],
         strike, exotic[i])
  output[y+1,1] <- exotic [i]
  output[y+1,2] <- callorput[2]
  output[y+1,3] <- strike
  output[y+1,4] <- main2(price, maturity, steps, sigma, int, iter, seed, callorput[2],
         strike, exotic[i])
  y <- y+2
}

#create dataframe & rename columns
df3 <- data.frame(output)
colnames(df3) <- c('Exotic', 'Call or Put', 'Strike', 'Option Price')
#output
df3
```

```
##             Exotic Call or Put Strike      Option Price
## 1    floatlookback        call     20   2.5101539729006
## 2    floatlookback         put     20  2.74441071454103
## 3    fixedlookback        call     20  2.91661212073614
## 4    fixedlookback         put     20  2.49285184917642
## 5  asianarithmetic        call     20  1.01511826435883
## 6  asianarithmetic         put     20 0.928143975504205
## 7   asiangeometric        call     20 0.979476290354559
## 8   asiangeometric         put     20 0.958006882367453
## 9   assetornothing        call     20  11.6760466929507
## 10  assetornothing         put     20  9.86572886490224
```

Present value is determined for each exotic price depending on each model's resident formula. The highest option value is with asset-or-nothing model, this is because it allows the investor to obtain the asset at a lower predetermined rate if the strike price is higher than the market price at the time of expiration. Normally, put options on an asset-or-nothing type do not pay the difference between the two since they will not actually receive the asset. fixed-lookback and float-lookback were the next highest option price. The method for these two options requires simulation. For the floating lookback, the team analyses the maximum or minimum price (depending on call or put) over the maturity period and takes the difference from the ending simulated value. Similarly, the fixed lookback will take either the minimum or maximum price from the simulation and compare it to the strike price. Asian arithmetic and Asian geometric are also closed-form options that apply simulation to understand present value. The main difference for these is that the payoff is calculated by the average prices over the simulated lifespan of the asset. As a result, the Asian or Monte Carlo payoff will be less expensive than lookbacks or asset-or-nothing options. The difference between Asian Arithmetic and Asian Geometric is only the type of average applied, where arithmetic is mean by sum of total values divided by number of values and geometric is the average rate of return of a set of products.

## Conclusion

The main intent of this modeling project is to help determine how much an option is worth. The team provided different numerical solutions that are actually only approximate answers in an attempt to understand the likely path from many different samples, as well as closed-form solutions that can provide an exact answer without using a simulation. European and Binomial/Black-Scholes models mostly used closed form solutions, where Asian and Monte Carlo methods are numerical solutions. An advantage of using Binomial methods is that they are computationally inexpensive, but typically harder to code. Monte Carlo models are the opposite where they are simpler to code, but are computationally expensive. Asset-or-nothing options are a numerical method that is considered a simplified risk hedge because the investor can actually receive the stock. Both fixed lookback and floating lookback methods are closed-form and require simulation since they payoff is determined by the maximum or minimum price over the life of the option. Asian and Monte Carlo options are also closed-form models that require simulation in order to provide a hypothetical average for the life until the options maturity.

### Considerations

In this analysis one should realize that all of the methods in this report assume that the volatility (sigma) of each asset is constant of the life of each option. In real life there are many additional external factors that would most likely cause the asset's volatility to increase or decrease, especially if the maturity for the option is an extended lifetime. Another important note is that dividend yield was not taken into account, which could impact the overall price of the investment. Cash dividends tend to drop the price of the asset and can impact the call or put premiums.

This assignment helped to seriously reinforce my understanding of the differences in exotic options. Since most of the modeling required outside research, it really forced me to find multiple sources and fully read and understand the information to ensure the accuracy of my report. My personal limitation was the inability to fully validate my own coding since the simulations are hard to reproduce in a formal web search. I feel as though I was able to piece together enough of the concepts that it still helped validate the code to an extent, whereas, certain methods should hypothetically have a higher present value based on their definition.