ISSN 2181-3187

# UNLOCKING OPTION PRICING WITH R: LEVERAGING R FOR SIMPLIFIED CALCULATIONS

***Tojiboyeva Aziza***
*Master's student at University of World Economy and Diplomacy, Tashkent*

**Abstract:** This article offers a hands-on guide to implementing option pricing models in R. We introduce the Black-Scholes model, covering its theoretical basis and practical calculation using R's OptionPricing package. Readers will learn to price calls and puts, as well as explore the concept of implied volatility for market insights. This article is ideal for those seeking to apply R to quantitative finance or gain a deeper understanding of option pricing mechanisms.

**Keywords:** Option Pricing, Options, Quantitative Finance, Implied Volatility, Black-Scholes Model, R Programming, R for Finance, Rmetrics, OptionPricing, Data Analysis.

The statistical programming language R is a very powerful tool for statistical and graphical techniques to explore, illustrate, study, and model almost any imaginable mathematical scenario. It is also very popular as a tool for the study of quantitative finance. Financial professionals increasingly favor R for its adaptability and robust data analysis tools, making it a leading language in the field.

In the field of finance, options give investors the chance to create revenue through option writing strategies, speculate on future price fluctuations, and hedge their investments. However, it can be difficult to appropriately price options as well as understand how volatility operates. This is where R, a powerful programming language for statistical computing and graphics, comes into play. R provides a rich toolbox for volatility modeling and option pricing with its many libraries and packages tailored to the finance industry.

In this article I will try to take a look at the use of R to explore Black-Scholes option pricing model.

R, a strong programming language and environment for statistical computation, offers an extensive selection of packages and functions that help simplify the process of importing and cleaning financial data. In practice, obtaining data from several sources—such as databases, spreadsheets, or web APIs—is required for importing financial data into R. Several packages in R, such as "readr", "readxl", and "httr", make it easier to read data from various file formats or online services. For example, you may import financial data into R with a few lines of code using the "readxl" package if the data is saved in an Excel spreadsheet. Similarly, the "httr" package can assist you

ISSN 2181-3187

in retrieving data straight into your R environment if you need to get real-time stock prices from an API such as Alpha Vantage.

One of the rich tools available for quantitative finance in R is Rmetrics collection. Rmetrics is a free and open-source software project for teaching computational finance. Rmetrics is based primarily on the statistical R programming language[1]. The project was started in 2001 by Diethelm Wuertz, based at the Swiss Federal Institute of Technology in Zurich. The Rmetrics project is now supported by the nonprofit Rmetrics Association which also publishes several ebooks on, but not limited to, finance.

•**fBasics:** which is a collection of functions to explore and investigate basic properties of financial returns and related quantities. The fields covered include techniques of explorative data analysis and the investigation of distributional properties, including parameter estimation and hypothesis testing;

•**fOptions:** a library of function for the pricing of basic European and American put and call options;

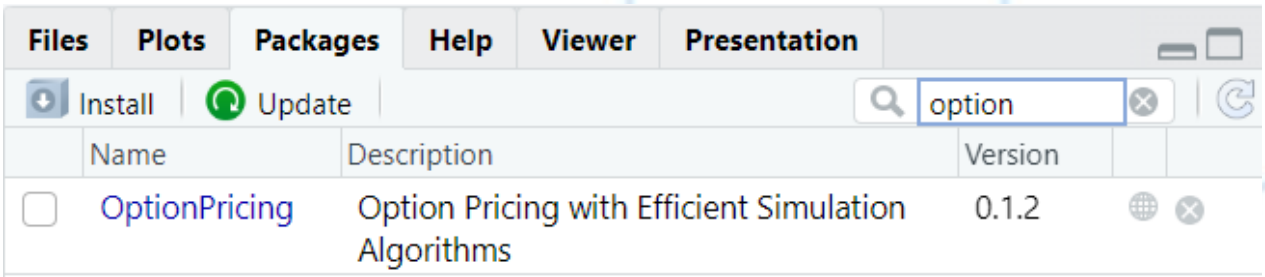
•**fAsianOptions:** includes different approximation methods for the pricing of Asian option in the Black and Scholes model;

•**fExoticOptions:** standard Asian option pricing, as well as barrier options, binary options, lookback options, etc.

• **timeDate:** a framework for chronological and calendar objects;

This small list is only a part of the suite specifically focused on option pricing. Unfortunately, fOptions package was removed from the CRAN repository on 2022-04-29 for misrepresentation of authorship and ownership of copyright[2]. So we use "OptionPricing" package in order to estimate the value of the options contract.

First, we need R and R Studio in order to set this up.




The required package is called OptionPricing, which can be completed by clicking on the packages tab in the interface, then "Install" and typing in "OptionPricing" as is. Before using, "OptionPricing" must be checked in the "Packages" tab to get it up and running for use. The OptionPricing package calculates

[1] https://en.wikipedia.org/wiki/Rmetrics
[2] https://cran.r-project.org/web/packages/fOptions/index.html

ISSN 2181-3187

the Price, Delta and Gamma for European options using the Black-Scholes formula. The price, Delta and Gamma for Asian call options under geometric Brownian motion are calculated using a very efficient Monte Carlo and randomized quasi-Monte Carlo algorithm.

The Black Scholes model estimates the value of a European call or put option by using the following parameters:

S = Stock Price

K = Strike Price at Expiration

r = Risk-free Interest Rate

T = Time to Expiration

sig = Volatility of the Underlying asset

Using R, we can write a function to compute the option price once we have the values of these 5 parameters. The BlackScholes function takes these parameters and returns the value of the call or put option.

```
# Step 1 – Load required package:
```{r}

require(OptionPricing)

# Step 2 –Build the pricing formula:

BlackScholes <- function(S, K, r, T, sig, type){

 if(type=="C"){
 d1 <- (log(S/K) + (r + sig^2/2)*T) / (sig*sqrt(T))
 d2 <- d1 - sig*sqrt(T)

 value <- S*pnorm(d1) - K*exp(-r*T)*pnorm(d2)
 return(value)}

 if(type=="P"){
 d1 <- (log(S/K) + (r + sig^2/2)*T) / (sig*sqrt(T))
 d2 <- d1 - sig*sqrt(T)

 value <-  (K*exp(-r*T)*pnorm(-d2) - S*pnorm(-d1))
 return(value)}
}

```
```

ISSN 2181-3187

This function first calculates the **d1** and **d2** parameters required for the Black-Scholes model and then uses ***pnorm()*** command of R which simulates a cumulative normal distribution. In order to use the Black-Scholes function to value a call and a put option, we can run the following lines:

```
# Step 3 – Calculate call and put options using the formula:

```{r}

call <- BlackScholes(110,100,0.04,1,0.2,"C")
show(call)

put <- BlackScholes(110,100,0.04,1,0.2,"P")
show(put)

[1] 16.96868
[1] 3.047622

```
```

Some basic assumption that this model has is that assumes that the volatility is constant over the life of the option. There are two ways to estimate the volatility that is plugged in the Black-Scholes model[3]. The first approach is by calculating the historical standard deviations of the stock returns and the second method uses the option market prices to find the Implied Volatility or the volatility that the market estimate for the stock. In the second approach to calculate volatility is to find the Implied Volatility value that the market estimates for a specific option. We can use the Black Scholes model to calculate the Implied Volatility by using known values of the stock price, strike price, time to expiration, interest rate and an array of standard deviation values that would allow us to find an option price which is the nearest price related to the option market price. Suppose that the parameters of the Black Scholes model are the following:

S <- 50
K <- 70
C <- 1.5 (Actual Call Price)
r <- 0
Time <- 1
sigma <- seq(0.01,0.5,0.005)

[3] https://financetrain.com/black-scholes-options-pricing-model-in-r

ISSN 2181-3187

The variable ***CallValueperSigma*** would store all the option prices that are calculated with the different values of sigma along the sigma vector.

```
# The variable CallValueperSigma would store all the option prices that are calculated with the different values of sigma along the
# sigma vector.

CallValueperSigma <- BlackScholes(S,K,r,Time,sigma)
```

The next step is to find which of these values is the nearest compared to the actual call price which is 1.5.

```
# IV is the the min value of the difference between each value of the
# CallValueperSigma vector and the call price.

IV <- which.min(abs(CallValueperSigma-C))/2

#The "which" command would return the index of the value in the sigma vector that #correspond to the option price derived from the BlackScholes function that is #nearest respect the call market price.

#This value is divided by two in order to get the specific Implied Volatility value from the sigma vector, because sigma starts from 0 to 0.5.
```

The Implied Volatility value from above is 32%. So we can conclude that the market assigns a 32 % of Implied Volatility to this contract option. In this case, the market anticipates the price of the underlying asset to experience relatively high volatility (standard deviation) over the course of a year, with an annualized standard deviation of approximately 32%.

In conclusion, this article explores R, a powerful statistical programming language, for option pricing in finance. It highlights the "OptionPricing" package within R that simplifies option valuation calculations. The Black-Scholes model, a popular method for European option pricing, is explained along with its reliance on implied volatility.

ISSN 2181-3187

**References:**


1. Black, Fischer., & Scholes, Myron. (1973). The Pricing of Options and Corporate Liabilities. Journal of Political Economy, 81(3), 637-654.
2. Hull, John C. (2018). Options, Futures, and Other Derivatives. Pearson Education.
3. Wilmott, Paul. (2006). Paul Wilmott on Quantitative Finance (2nd ed.). John Wiley & Sons.
4. Rmetrics Project: https://www.rmetrics.org/ [The official Rmetrics website, with documentation and resources for financial analysis with R]
5. OptionPricing package documentation: https://cran.r-project.org/web/packages/OptionPricing/OptionPricing.pdf
6. Wikipedia - https://en.wikipedia.org/wiki/Rmetrics
7. CRAN Packages – https://cran.r-project.org/web/packages/fOptions/index.html
8. https://financetrain.com/black-scholes-options-pricing-model-in-r