# Derivation of the Black-Scholes PDE and its implementation using Python

Aidyn Kadyr, Nurzhan Bekzhanov

April 27, 2021

Report for MATH 350 Research Methods, Spring 2021

*Department of Mathematics, School of Sciences and Humanities*

*Nazarbayev University*

**Supervisor:** Dongming Wei

---

**Abstract**

The Black-Scholes model was created by Myron Scholes and Fisher Black in 1968 and further developed by Robert C. Merton. It is a widely used model for estimating derivatives prices and facilitated options trading around the world. The model applies mathematical concepts such the Geometric Brownian Motion, Ito's Calculus and Partial Differential Equations. Moreover, it represents derivative prices under certain conditions, including no arbitrage and risk-neutral pricing principles. While, the Monte Carlo Simulation finds the expected present option value of all future cash flows. This work is aimed at deriving the Black-Scholes PDE and comparing it with Monte Carlo Simulation, albeit without the proof. In the end, there is a simple example of the model implemented using Python to estimate a particular option price for the Walmart 135 Strike, April 1 Call contract. Results reveal that both Black-Scholes model and Monte Carlo simulation are able to produce prices close to actual call option price

---

## 1 Introduction

The Black-Scholes PDE is derived using the Ito Calculus as well as the Geometric Brownian Motion. Most of the concepts and derivations are defined in the textbook by Marcel B. Finan (2016) - Discussion of Financial Economics in Actuarial Models. [1] The Black-Scholes Equation is given by the following Partial Differential Equation.

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

The solution for the PDE in terms of a price of a Call Option is given analytically by: (Proof Omitted)

$$C_0 = S_0 N(d_1) - K e^{-rT} N(d_2)$$

$$d_1 = \frac{ln\frac{S_0}{K} + (r + \frac{\sigma^2}{2})T}{\sigma\sqrt{T}}$$

$$d_2 = \frac{ln\frac{S_0}{K} + (r - \frac{\sigma^2}{2})T}{\sigma\sqrt{T}}$$

$N(x)$ - CDF of Standard Normal Distribution

$S_0$ - Stock Price

$K$ - Exercise Price (Strike Price)

$r$ - Risk-free Interest Rate

$T$ - Time to Expiration

$\sigma$ - Standard Deviation (Volatility)

# 2 Derivation

Assumptions regarding the Black-Scholes model are the following:

1. Stock prices follow the Geometric Brownian Motion described by the stochastic differential equation:

$$dS(t) = \alpha S(t)dt + \sigma S(t)dZ(t) \tag{2}$$

2. Short selling of stocks is allowed.
3. Stock are divisible.
4. No arbitrage principle holds: there is no risk-free ways of making money.
5. Constant risk-free interest rate.
6. Continuous trading.
7. Zero transaction costs and no commissions.

**Proof of Equation (1):**
Let $V = V(S,t)$ be the value of a call or put option, and $\Delta$ be the number of shares of price $S$. Assume the following portfolio of buying an option and buying $\Delta$ shares of stock:

$$\pi = V(S,t) + \Delta S$$

The value of the portfolio will change according to

$$d\pi = dV(S,t) + \Delta dS \tag{3}$$

where $dV$ is derived from the Ito's Lemma: (Proof omitted)

$$dV = (\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + \alpha S \frac{\partial V}{\partial S})dt + \sigma S \frac{\partial V}{\partial S} dZ(t) \tag{4}$$

where $dV$ is a differential of a time dependent stochastic process. It is derived by using the Taylor's series up to second derivative, since rest of the terms are close to zero.

Substituting $dV$and $dS$ into Eq. 3 we get $d\pi$:

$$d\pi = \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + \alpha S \frac{\partial V}{\partial S}\right) dt + \sigma S \frac{\partial V}{\partial S} dZ(t) + \Delta(\alpha S(t)dt + \sigma S(t)dZ(t))$$

Let $\Delta = -\frac{\partial V}{\partial S}$ s.t. stochastic part is cancelled:

$$d\pi = \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + \alpha S \frac{\partial V}{\partial S}\right) dt + \sigma S \frac{\partial V}{\partial S} dZ(t) + (-\frac{\partial V}{\partial S})(\alpha S(t)dt + \sigma S(t)dZ(t))$$

$$= \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + \alpha S \frac{\partial V}{\partial S}\right) dt + \sigma S \frac{\partial V}{\partial S} dZ(t) - \frac{\partial V}{\partial S}\alpha S(t)dt - \frac{\partial V}{\partial S}\sigma S(t)dZ(t)$$

$$= \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2}\right)$$

By the no arbitrage principle we need our position to equal to the bank deposit at a risk-free interest rate such that:

$$d\pi = r\pi dt = r(V + \Delta S)dt = r(V - \frac{\partial V}{\partial S}S)dt = \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2}\right)$$

This way gives the Black-Scholes partial differential equation: [1]

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

# 3 Python implementation of the Black-Scholes PDE

As for March 26 the current price of Walmart stock (WMT) is 135.13. A Call option expiring at April 30, 2021 has a strike price of 135. Option data are derived from Yahoo Finance

To calculate the price of the call option we need the following parameters: $K$ (Strike Price), $S$ (Current Price), $r$ (risk-free rate), $T$ (Time to Maturity) and $\sigma$ (Volatility). While $K$ is given and $T$ can be easily calculated to be 35 days, $r$ is derived from the latest 10-year Treasury Bill yield, which is considered risk-free security, to equal 1.660.

The challenging part is to deduce the value of $\sigma$ - volatility of the stock prices. It is constantly changing. Here, the approach is to use historical data to take the standard deviation of the logreturns for the past year and multiple by the number of days the financial market functioned.

Plugging in the values result in the Call price of 4.115553. While the real Call price on Yahoo Finance is 2.9.

Note, that the difference in prices is accounted for ideal market conditions and assumptions that are not satisfied in the real world.

```
import numpy as np
import pandas as pd
from scipy.stats import norm
import pandas_datareader.data as web

def d1(S, K, T, r, sigma):
    d1 = (np.log(S/K) + (r + sigma**2 / 2) * T) / (sigma * np.sqrt(T))
    return d1
def d2(S, K, T, r, sigma):
    d2 = (np.log(S/K) + (r - sigma**2 / 2) * T) / (sigma * np.sqrt(T))
    return d2
def Call(S, K, T, r, sigma):
    Call = S * norm.cdf(d1(S, K, T, r, sigma)) - K * np.e ** (-r*T)
    * norm.cdf(d2(S, K, T, r, sigma))
    return Call

df = web.DataReader('WMT', 'yahoo','2020-03-24', '2021-03-26') #Walmart stock price data
df['logreturn'] = np.log (df['Close'] / df['Close'].shift(1))

sigma = np.sqrt(252) * df['logreturn'].std()
TB = web.DataReader('^TNX', 'yahoo','2020-03-24', '2021-03-26' ) #Treasury Bil yield data

r = TB.Close.iloc[-1] / 100 # risk free interst rate
S = df.Close.iloc[-1] # Current Price == Last Close Price
K = 135  # Strike Price
T = 35/365 # Option Expires at April 1 --> 35 days
print('By the Black-Scholes Model, price of the Call Option is:', end = ' ')
print(Call(S, K, T, r, sigma))
Return: By the Black-Scholes Model, price of the Call Option is: 4.115553973100759
```

# 4 Python implementation of the Monte Carlo Simulation

Monte Carlo Option Pricing method is widely used in mathematical finance to calculate the value of an option with multiple sources of uncertainties and random features, such as changing interest rates, stock prices or exchange rates. This method is called Monte Carlo simulation, naming after the city of Monaco, which is noted for its casinos. This method relies on repeated random sampling and statistical analysis in order to calculate the result. Monte Carlo simulation can generate various different price paths for the underlying share for options on equity. To obtain the value of an option the average value of payoffs should be discounted [3]. In this section, Python programming language is used to create a sequence of random paths to calculate the call option price. The time-0 Monte Carlo option price is defined by:

$$V(S_0, 0) = \frac{1}{n} e^{-rT} \sum_{i=1}^{n} V(S_T^i, T) \tag{5}$$

$V(S, T)$ - Option Payoff at time T

$S_T^1, S_T^2, ..., S_T^n$ - n Randomly Drawn Stock Prices

$K$ - Exercise Price (Strike Price)

$r$ - Risk-free Interest Rate

$T$ - Time to Expiration

$\sigma$ - Standard Deviation (Volatility)

```
import numpy as np
import pandas as pd
from scipy.stats import norm
import pandas_datareader.data as web

r = TB.Close.iloc[-1] / 100 # risk free interst rate
S = df.Close.iloc[-1] # Current Price == Last Close Price
K = 135  # Strike Price
T = 35/365 # Option Expires at April 1 --> 35 days
print('By the Black-Scholes Model, price of the Call Option is:', end = ' ')
Nsimulations= 5000
Nsteps= 250
dt= T/Nsteps

df = web.DataReader('WMT', 'yahoo','2020-03-24', '2021-03-26') #Walmart stock price data
df['logreturn'] = np.log (df['Close'] / df['Close'].shift(1))

sigma = np.sqrt(252) * df['logreturn'].std()
```

```
TB = web.DataReader('^TNX', 'yahoo','2020-03-24', '2021-03-26' ) #Treasury Bil yield data

drift= (r-(sigma**2)/2)*dt
a= sigma*np.sqrt(dt)
x= np.random.normal(0,1,(Nsimulations, Nsteps))
#Normal random variable X with mean 0 and variance 1

Smat= np.zeros((Nsimulations, Nsteps))
#Matrix of stock prices with Nsimulations rows and Nsteps columns
Smat[:, 0] += S #The first column is filled with today's stock price

for i in range(1, Nsteps):
  Smat[:, i] += Smat[:, i-1]*np.exp(drift +a*x[:, i])
#Generating random stock prices

c= Smat[:, -1]- K       #Call = Stock price - Strike price
for i in range(len(c)):
  if c[i]< 0:           #Call price cannot be negative
    c[i]= 0
  else:
    c[i]= c[i]

payoff_call= np.mean(c)

MCS_Call = payoff_call*np.exp(-r*T)

print('By the Monte Carlo Simulation, price of the Call Option is:', end = ' ')
print(MCS_Call)
Return: By the Monte Carlo Simulation, price of the Call Option is: 4.1043723300656225
```

The graphs of the generated random stock prices.

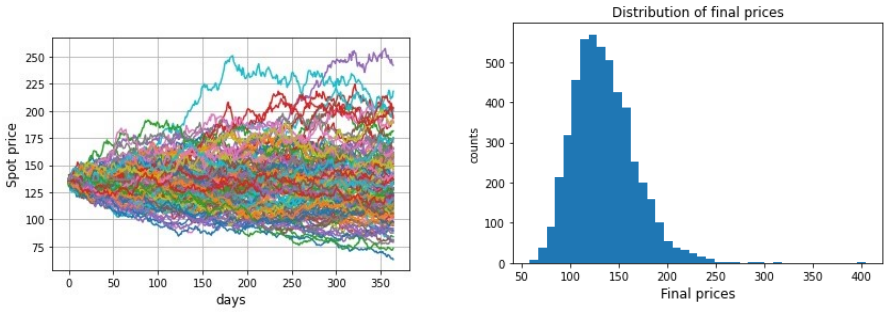


Figure 1: Plot of the stock price paths

# 5 Conclusion

The Black-Scholes equation is the foundation for the modern Option Pricing Theory. Even though it has several assumption that lead to an ideal market situation, it sets a benchmark for option prices existing in real life.

Moreover, the Black-Scholes PDE can be used not only to price an Option, but also any derivative that depends on the price of the underlying asset and time.

Therefore, it has a significant contribution to the field of financial mathematics and economics as well as the way financial markets function.

Obtained results show that the call option price calculated with the Black-Scholes model is equal to 4.11. On the other hand, the Monte Carlo simulation measures that the call option to be equal to 4.10. While the real Call price on Yahoo Finance is 2.9. We can conclude that the Black-Scholes model is supported by the Monte Carlo simulation. Yet results can be improved by examining non-constant volatility.

# References


[1] Marcel B. Finan. *A Discussion of Financial Economics in Actuarial Models.* Arkansas Tech University, 2016.

[2] Qiwu Jiang. *Comparison of Black-Scholes Model and Monte-Carlo Simulation on Stock Price Modeling.* 2019 International Conference on Economic Management and Cultural Industry (ICEMCI 2019), 2019.

[3] Samik Raychaudhuri. *Introduction to monte carlo simulation.* 2008 Winter simulation conference, pp. 91-100. IEEE, 2008.

[4] Terence Tao. *The Black-Scholes Equation.* Terry Tao Wordpress, 2008.

[5] Evan Turner. *The Black-Scholes Model and Extensions.* The University of Chicago, 2010.