# How does the use of Monte Carlo integration sampling compare to the Black-Scholes model in terms of accuracy and reliability for stock option pricing?

# 1 Introduction

As a high school senior, and president of both the computer science and finance club, I have developed a strong background in computer science and a keen interest in finance. This has led me to explore traditional option pricing methods, such as the Black-Scholes model and Monte Carlo integration, to see how they compare to the actual value. Through this investigation, I hope to deepen my understanding of the field of finance, and its relation with the field of mathematics.

European call options are financial contracts prevalent in the stock market for hedging against potential losses. The Black-Scholes model, a widely accepted mathematical model in finance due to its computational simplicity, provides a solution for pricing European call options. However, this model relies on assumptions such as a constant risk-free rate and no dividends and may not always accurately reflect the complexity of the underlying stock distribution.

Monte Carlo integration, a numerical method for approximating definite integrals by randomly sampling points from the domain.  In this IA, I will gather historical stock data from the Stockholm 30 index using Yahoo Finance, model the stock using a lognormal distribution, generate random samples with Python, and price the European call option using both Black-Scholes and Monte Carlo integration. I will then compare the results using Logarithmic Standard Deviation, a statistical measure of the volatility of a stock or other financial asset over time. In summary, my aim is to compare the accuracy and reliability of the Black-Scholes model and Monte Carlo integration in pricing European call options.

# 2 Background

## 2.1 Black-Scholes

The Black-Scholes model, developed by Fischer Black, Myron Scholes and Robert Merton in 1973, is a widely used method for pricing options in financial markets. It is based on the following assumptions:

(1) The short term interest rate, r, is known and constant through time.

(2) The stock price follows Brownian Motion and the volatility is constant.

(3) The parameters $\mu$ and $\sigma$ are contingent on S, the stock price

(4) The stock does not pay dividends or other distributions

(5) Short selling is allowed

(6) There are no arbitrage opportunities and all security trading is continuous

(7) The option has a maturity at time period t (European Option)

The Black-Scholes model does not account for the ability to exercise an option before the expiration date. For this reason, this analysis will be conducted with European options, which unlike American options, can only be exercised on the expiration date. Despite these limitations, the Black-Scholes model remains a popular method for pricing options in financial markets, due to its computational and mathematical simplicity.

The Black-Scholes model uses the Ito calculus, which is a mathematical framework for dealing with stochastic processes. Ito's lemma is a key concept in this framework, allowing us to determine how different functions of the underlying asset change over time. The model uses this to determine the option price by taking into account the current price of the underlying asset, the volatility of the asset, the time to expiration of the option, and the risk-free interest rate.

“An Itˆo Process is an advanced technique that separates a stochastic process xt into a a sum of two integrals, one with respect to time and one with respect to Brownian Motion.” (Washburn and Mehmet, 2021)

$$d_t = d_0 + \int_0^t \sigma_t dW_t + \int_0^t u_t dt_t$$

Taking the ”derivative” of the Itˆo Process gives us the following differential equation:

$$dS(t) = \mu S(t)dt + \sigma S(t)dW(t)$$

If f is a twice differentiable function of the price of a call option or other derivative that depends on S, Itˆo's Lemma can be used to express the change in f, the value of an option, as df.

$$df = (\mu S\frac{\partial f}{\partial S} + \frac{\partial f}{\partial t} + \frac{1}{2}\frac{\partial^2 f}{\partial S^2}\sigma^2 S^2)dt + \frac{\partial f}{\partial S}\sigma S dz$$

We define Π as the value of our portfolio.

$$\Pi = -\Delta f + \frac{\delta f}{\delta S}\Delta S$$

We can sub in our previous equations to get the following:

$$\Delta\Pi = -[(\mu S\frac{\partial f}{\partial S} + \frac{\partial f}{\partial t} + \frac{1}{2}\frac{\partial^2 f}{\partial S^2}\sigma^2 S^2)\Delta t + \frac{\partial f}{\partial S}\sigma S\Delta z] + \frac{\partial f}{\partial S}[\mu S\Delta + \sigma S\Delta z]$$

After distributing and cancelling like terms, we are left with the following equation:

$$\Delta\Pi = -\frac{\partial f}{\partial t}\Delta t - \frac{1}{2}\frac{\partial^2 f}{\partial S^2}\sigma^2 S^2\Delta t$$

The $\Delta z$ term that was cancelled out represented a stochastic variable. Without this variable, our portfolio is riskless given our assumptions. Therefore, our portfolio will earn instantaneous rates of return of time period $\Delta t$.

$$\Delta\Pi = r\Pi\Delta t$$

Once we substitute the previous equations, we can solve for the Black-Scholes equation as follows:

$$(-\frac{\partial f}{\partial t} - \frac{1}{2}\frac{\partial^2 f}{\partial S^2}\sigma^2 S^2)\Delta t = r(-f + \frac{\partial f}{\partial S}S)\Delta t$$

And when rearrange it becomes the Black-Scholes equation

$$\frac{\delta f}{\delta t} + rS\frac{\delta f}{\delta S} + \frac{1\delta^2 f}{2x^2}\sigma^2 S^2 = rf$$

The expected payoff for the black scholes model is given by the following equation:

$$C = SN(d_1) - Ke^{-rt}N(d_2), \tag{2.1.2}$$

with

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

$$d_2 = d_1 - \sqrt{\sigma^2 t}$$

where $C$ is the price of the call option, and N signifies a standard-normal distribution. The parameters in this formula are as follows:

$S$ - The spot price of the underlying asset.

$K$ - The strike price of the option.

$r$ - The risk-free interest rate.

$\sigma^2$ - The variance in the value of the underlying asset.

$t$ - Time in years to the date of maturity.

As mentioned, the Black-Scholes model assumes that the variance of the underlying asset's price is constant. The variance $\sigma^2$ is calculated using the following equation:

$$\sigma^2 = \frac{\sum_{i=1}^{n} (X_i - \bar{X})^2}{n-1} \tag{2.1.3}$$

where n represents the number of data points in the sample, $X_i$ are samples from the population, and $\bar{X}$ is the sample mean.

## 2.2 Monte Carlo Integration

The Monte Carlo method is a numerical approach for pricing options, opposed to the analytical approach seen in the Black-Scholes model. The Monte Carlo method is based on generating random samples from the probability distribution of the underlying asset. The option's value is then estimated by calculating the average payoffs generated from these samples. This process is repeated multiple times, with different sets of random samples, to obtain more accurate estimates of the option's value.

The advantage of the Monte Carlo method is that it can be applied to a wide range of assets and option types, making it a versatile tool for option pricing. However, it can be computationally intensive, requiring a large number of simulations to obtain accurate results. The Monte Carlo method has gained popularity in recent years due to the increasing computational power of modern computers.

This is expressed in the equation below:

$$S_t = S_0 e^{(r-\frac{1}{2}\sigma^2)t+\sigma W_t} \tag{2.2}$$

$W_t$ is defined as the square root of time to maturity multiplied by a random variable Z, where Z follows a normal distribution with a mean of 0 and a standard deviation of $\sqrt{t}$

S - The spot price of the underlying asset.

r - The risk-free interest rate.

$\sigma^2$ - The variance in the value of the underlying asset.

t - Time in years to the date of maturity.

## 2.3 Risk-Free Rate

To determine the risk-free rate for this analysis, a ten-year government bond rate was utilised. Utilising the ten-year risk-free rate is suitable for practical use, assuming that risk-free rates remain relatively constant across time (Damodaran, 2008).

| Date | Yield |
|---|---|
| 2019 Dec 31 | 1.3% |
| 2018 Dec 31 | 1.8% |
| 2017 Dec 31 | 1.7% |

From "Average Risk Free Rate Sweden." Statista, 16 Nov. 2022, https://www.statista.com/statistics/885803/average-risk-free-rate-sweden/#:~:text=This%20rate%20represents%20the%20minimum,in%20Sweden%20was%201.4%20percent.

The average risk-free yield of the years 2017-2019 was used to determine the risk-free rate as it provides an accurate estimate of the rate while taking into account the fluctuations in the rate over the period of the stock data collection. The risk-free rate for this analysis is 1.6%.

## 2.4 Strike Prices

The strike prices used in the analysis were determined by calculating a range of ±20% from the last closing price of the stock of interest. The ±20% range was arbitrarily chosen as it provides a reasonable range of strike prices that are likely to be representative of the market value of the underlying asset. This is expressed in the equation below:

$$\text{Strike price range} = [0.8 \cdot S_0, 1.2 \cdot S_0] \tag{2.4}$$

## 2.5 Option Contracts

An option contract is a financial agreement between the buyer and the seller, where the buyer has the right, but not the obligation, to buy or sell an asset at a predetermined price (strike price) before or at a specified date (expiration date).

Options can be classified into two types: American-type and European-type options. American-type options can be exercised at any time before the expiration date, whereas European-type options can only be exercised on the expiration date. This paper uses European call options, where the buyer has the right to buy the underlying asset at the strike price on the expiration date. European call options are widely used in the financial markets and their closed-form solutions make them computationally easy to price.

A time to maturity of 1 month was used in this analysis because it is a common time frame for options trading. Many traders will use options with a time to maturity of 1 month as a way to speculate on short-term price movements or to hedge against potential price changes. Additionally, using a time to maturity of 1 month allows for a more direct comparison between the Black-Scholes and Monte Carlo simulations, as both methods are based on different assumptions and calculations. By using a shorter time frame, it is easier to see how well each method is able to predict the actual value of the call options.

$$1\ month = \frac{1\ year}{12\ months} \tag{2.5}$$

t = 0.8

# 3 Data Collection

The data in this paper was sourced from yahoo finance. The stocks selected for analysis are those listed on the OMX 30 Stockholm Index (OMXS30), a market index comprising the 30 stocks with the highest trading volume. Some stocks were excluded because they did not have enough historical data or they were not traded during the specified time frame.  The data was collected over a period of 2017-01-01 to 2020-01-01, a time frame chosen for its proximity to the present while also not containing any major events such as COVID.

## 3.1 Volatility

According to equation 2.1.3, the volatility of each individual stock can be determined. For further information on the calculation process, please consult the code provided in reference (5) in the appendix section.

Sample Calculation for ABB for Volatility:

$$\sigma^2 = \frac{169344.8956}{751}$$

$$\sigma = 15.02$$

**Table 3.1**: Stocks listed on OMXS30, their corresponding volatility, mean and sum of variance

| Company | Volatility | Mean | $\sum_{i=1}^{n} (x_i - \bar{x})^2$ |
|---|---|---|---|
| ABB Ltd | 15.02 | 198.0706 | 169344.8956 |
| Alfa Laval | 22.71 | 200.6234 | 387355.7659 |
| Assa Abloy B | 17.73 | 187.3054 | 236171.9475 |
| Atlas Copco A | 8.73 | 65.5127 | 57186.0324 |
| Atlas Copco B | 7.33 | 58.6391 | 40362.3292 |
| Boliden | 30.91 | 256.1815 | 717717.9447 |
| Ericsson B | 14.47 | 68.9773 | 157286.2508 |
| Essity B | 28.69 | 247.7626 | 525300.0808 |
| Evolution Gaming | 49.24 | 129.3079 | 1820706.3527 |
| Getinge B | 23.36 | 118.5984 | 409948.9200 |
| Hexagon AB | 7.85 | 64.3825 | 46309.3350 |
| Investor B | 10.03 | 101.4625 | 75510.6238 |
| Kinnevik B | 25.33 | 252.5999 | 481808.2474 |
| Nordea Bank | 14.77 | 89.2866 | 163942.4928 |
| Sandvik | 14.92 | 147.2388 | 167138.0021 |
| Sinch | 3.94 | 11.2695 | 11649.4934 |
| SEB A | 7.30 | 94.6124 | 39998.1452 |
| Skanska B | 21.51 | 179.6161 | 347351.8920 |
| SKF B | 14.24 | 169.2759 | 152321.5636 |
| SCA B | 13.88 | 79.7231 | 144703.6823 |
| Svenska Handelsbanken A | 12.29 | 107.9342 | 113391.6166 |
| Swedbank A | 31.46 | 188.2070 | 743109.2411 |
| Swedish Match | 7.06 | 37.7050 | 37381.3772 |
| Tele2 B | 18.68 | 109.7345 | 261973.7201 |
| Telia Company | 2.32 | 39.9968 | 4025.9961 |
| Volvo B | 13.39 | 142.9871 | 134669.5668 |

## 3.2 Black Scholes

According to equation 2.1.2, the value of the call option for each individual stock can be determined using Black Scholes. For further information on the calculation process, please consult the code provided in reference (8) in the appendix section.

Sample Calculation for ABB for Black Scholes Call Option:

$$d_1 = \frac{ln(\frac{225.1000}{258.8650}) + (0.016 + \frac{225.6004}{2})0.08}{\sqrt{225.6004 \cdot 0.08}}$$

$$d_1 = 2.1104$$

$$d_2 = 2.1104 - \sqrt{225.6004 \cdot 0.08}$$

$$d_2 = -2.1379$$

In order to determine the value of d1 and d2, the python library numpy was used to calculate the cumulative distribution function of the standard normal distribution. This is represented by the Greek letter phi (ϕ). As shown in the python screenshot (Figure 8), the norm.cdf of the values of d1 and d2, which are 0.98173 and 0.0155 respectively.

$$N(d_1) = 0.9817$$

$$N(d_2) = 0.0155$$

$$C = 225.1000 \cdot 0.9817 - (258.8650)e^{-(1.016)(0.08)} 0.01553,$$

$$C = 216.9735$$

### 3.3 Monte Carlo

According to equation 2.2, the value of the call option for each individual stock can be determined using Monte Carlo Integration. The first step in this implementation is to define the number of simulations (nSim) to be run. In this case, the number of simulations is set to 1,000,000. Next, a random normal variable (Z) is generated with a mean of 0 and a standard deviation of 1, with the same size as the number of simulations. This variable is used to generate the simulated terminal stock price (ST) at maturity. Once the simulated terminal stock price is generated, the simulated call payoffs are calculated by using the maximum of the difference between the terminal stock price and the strike price, and zero. The final step is to calculate the call price by averaging the simulated payoffs and discounting it to the present value using the risk-free rate. For further information on the calculation process, please consult the code provided in reference (9) in the appendix section.

$$S_t = S_0 e^{(r - \frac{1}{2}\sigma^2)t + \sigma W_t} \tag{3.3}$$

The Monte Carlo simulation is computationally intensive and cannot be performed by hand due to several reasons. Firstly, the complexity of the calculations required for each simulation, and the need for a large number of simulations to obtain accurate results, in this case 1 million. Additionally, the simulation relies heavily on the generation of random numbers, which is difficult to replicate manually.

**Table 3.3**: Stocks and their corresponding Black Scholes, Monte Carlo and Actual Call options

| Company | Black Scholes Call Option Price | Monte Carlo Call Option Price | Actual Call Option Price |
|---|---|---|---|
| ABB Ltd | 216.9735 | 152.4805 | 226.2000 |
| Alfa Laval | 235.5969 | 11.0405 | 243.2000 |
| Assa Abloy B | 216.4093 | 205.0875 | 232.1000 |
| Atlas Copco A | 73.6446 | 71.8165 | 87.5250 |
| Atlas Copco B | 57.5594 | 57.9268 | 76.6500 |
| Boliden | 248.4970 | 0.1095 | 233.6000 |
| Ericsson B | 229.8963 | 0.0266 | 234.1000 |
| Essity B | 78.3273 | 66.8322 | 76.1800 |
| Evolution Gaming | 301.7854 | 0.0864 | 306.8000 |
| Getinge B | 173.8385 | 15.9691 | 164.0000 |
| Hexagon AB | 55.5143 | 55.6223 | 75.3714 |
| Investor B | 108.3615 | 104.6682 | 133.5000 |
| Kinnevik B | 228.9240 | 1.6342 | 230.9000 |
| Nordea Bank | 72.9389 | 59.0063 | 75.5900 |
| Sandvik | 176.4450 | 140.2393 | 180.8000 |
| Sinch | 12.5392 | 12.5399 | 30.2000 |
| SEB A | 62.1716 | 62.2215 | 94.6000 |
| Skanska B | 211.2145 | 459.5042 | 222.9000 |
| SKF B | 181.2828 | 123.1254 | 178.4500 |
| SCA B | 90.4070 | 97.0497 | 93.1400 |
| Svenska Handelsbanken A | 92.8163 | 139.6660 | 94.0200 |
| Swedbank A | 139.4488 | 0.0138 | 143.8500 |
| Swedish Match | 33.3092 | 33.8149 | 54.5200 |
| Tele2 B | 134.7573 | 45.6263 | 146.7000 |
| Telia Company | 11.1282 | 11.0792 | 41.1100 |
| Volvo B | 147.9977 | 115.8952 | 163.9000 |

# 4 Results and Discussion

The use of logarithmic standard deviation (LSD) is a common practice in financial modelling for evaluating and comparing the accuracy of option pricing models. This is because the logarithmic scale compresses the range of large values, making it easier to compare the results of methods that generate a wide range of values. LSD is a symmetric measure that compares relative errors and is suitable for cases where relative error is more important than absolute, and when the variance of the dependent variable is the same for all data.

Sample Calculation for Monte Carlo logarithmic Standard Deviation for ABB

$$LSD = \sqrt{\frac{\sum_{i=1}^{n} (0.5(15.02)^2 - ln(\frac{216.9735}{226.2000}))}{n-1}}$$

$$LSD = \sqrt{\frac{\sum_{i=1}^{n} (112.8002 + 0.0416)}{n-1}}$$

$LSD = 0.0191$

For further information on the calculation process, please consult the code provided in reference (12) in the appendix section. This is done using the numpy log() function, which applies the natural logarithm to each element of the call price for each method. The logarithm is applied in order to compress the range of large values, making it easier to compare the results of the two models. Next, the standard deviation of the logarithmically transformed call prices for each model is calculated using the numpy std() function. In addition, consult the code to see the overall average of each method's LSD in reference (13).

**Table 4:** The average LSD error for the two methods with a maturity of 1 month

| Company | Black Scholes Error | Monte Carlo Error |
|---|---|---|
| ABB Ltd | 0.0191 | 0.1859 |
| Alfa Laval | 0.0159 | 1.6071 |
| Assa Abloy B | 0.0350 | 0.0719 |
| Atlas Copco A | 0.0863 | 0.1007 |
| Atlas Copco B | 0.1432 | 0.1441 |
| Boliden | 0.0309 | 3.4986 |
| Ericsson B | 0.0091 | 6.1531 |
| Essity B | 0.0139 | 0.2102 |
| Evolution Gaming | 0.0082 | 1.8237 |
| Getinge B | 0.0291 | 1.8563 |
| Hexagon AB | 0.1529 | 0.1480 |
| Investor B | 0.1043 | 0.0455 |
| Kinnevik B | 0.0043 | 1.6877 |
| Nordea Bank | 0.0179 | 0.1680 |
| Sandvik | 0.0122 | 0.1515 |
| Sinch | 0.4395 | 0.4375 |
| SEB A | 0.2099 | 0.2099 |
| Skanska B | 0.0269 | 1.3447 |
| SKF B | 0.0079 | 0.0261 |
| SCA B | 0.0149 | 0.0368 |
| Svenska Handelsbanken A | 0.0064 | 0.0921 |
| Swedbank A | 0.0155 | 3.7790 |
| Swedish Match | 0.2464 | 0.2477 |
| Tele2 B | 0.0425 | 0.2035 |
| Telia Company | 0.6534 | 0.6541 |
| Volvo B | 0.0510 | 0.0577 |
| **Overall Average** | **0.0922** | **0.9820** |

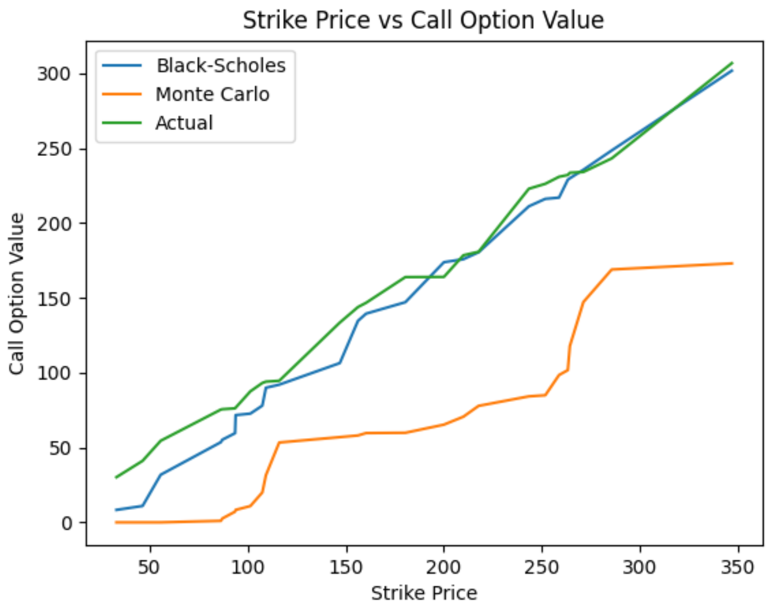


**Figure 4.1** line chart comparing the Monte Carlo method, Black Scholes model and actual call option price

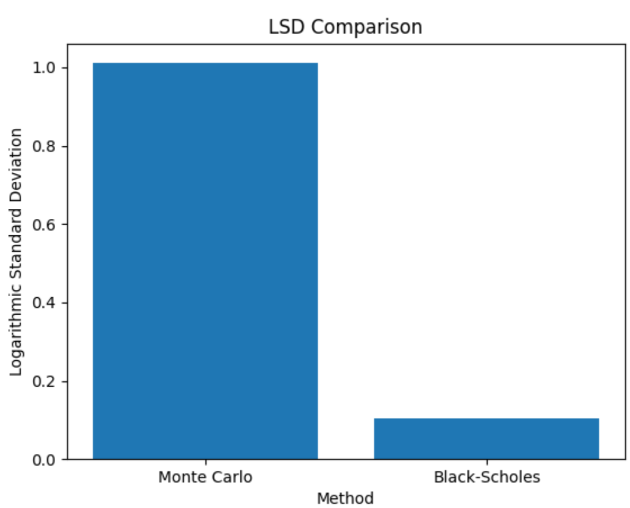


**Figure 4.2** bar chart comparing the Logarithmic Standard Deviation of the Monte Carlo method to that of the Black Scholes model

According to these findings, the Monte Carlo method has a higher level of error and therefore less precision than the Black Scholes model. This comparison is based on the average LSD, and it may be useful to take a step further and analyse the LSD values for stock more carefully. The discrepancy between  these two methods could be a result of some of the factors not taken into account for this paper, such as general market trends, and the industries of each stock in the OSMX30.

One possible explanation for the performance disparity is that the Monte Carlo method is based on simulation, unlike the closed-form solution offered by the Black Scholes model. Monte Carlo method generates estimates through random sampling, which can result in a wider range of possible outcomes and greater uncertainty in the results. It should be noted, however, that the Monte Carlo method is more useful than the Black Scholes model in situations where the underlying process cannot be easily modelled with closed-form solutions, such as when the underlying process is non-linear or has complex dependencies.

Overall, this comparison emphasises the significance of selecting an appropriate option pricing method based on the specific requirements of the problem at hand. While the Black Scholes model is more precise, it may not be appropriate for certain types of problems, where the Monte Carlo method may be more robust. More research is needed to fully understand the trade-offs between different option pricing methods, as well as how to choose the best method for a given problem.

# 5 Future Improvements

In order to further improve the analysis presented in this paper, several key areas could be considered for future research. One key area would be to incorporate stochastic calculus when modelling volatility. The biggest assumption made in this paper is that volatility of stocks is constant, however, it can change over time, and therefore it is important to use a model that can take this into account. The use of stochastic calculus would allow for the modelling of volatility as a random process, which would make the model more realistic and accurate, reducing the relative error.

Another area for future research would be to use importance sampling with Monte Carlo method. Importance sampling is a technique that allows to more efficiently generate samples from a probability distribution. By using importance sampling, it is possible to reduce the number of simulations required to achieve a given level of accuracy. This can greatly improve the computational efficiency of the Monte Carlo integration sampling and make it more practical to use in a real-world setting.

Additionally, it would be beneficial to compare the results of the Monte Carlo integration sampling and Black-Scholes model over different market periods. This would help to understand how the models perform in different market conditions such as bear, bull, and stagnant markets. Furthermore, it would be useful to carry out the analysis over several different maturities, providing a more comprehensive view of the model performance over different time horizons.

# 6 Conclusion

Based on the above logarithmic standard deviation values, it appears that the Monte Carlo method has a higher deviation from the actual value of the call option compared to the Black-Scholes method. This suggests that the Black-Scholes method may be more accurate in predicting the value of the call option. However, it is important to note that these results are based on a single stock and a time to maturity of 1 month. Further analysis should be conducted to determine if this trend holds true for other stocks and different time to maturities.

This conclusion cannot be presented for all different stocks and maturities, however the same pattern was observed for the majority of all stocks. This result suggests that simulating a stock's price movements using a Geometric Brownian Motion process is more accurate than using Monte Carlo integration.

# References

WASHBURN, Brandon, and Mehmet DİK. "Derivation of Black-Scholes Equation Using Itô's Lemma." *Proceedings of International Mathematical Sciences*, 15 June 2021, https://doi.org/10.47086/pims.956201.

Meding, Isak, and Viking Zandhoff Westerlund. "Pricing European Options with the Black-Scholes and Monte Carlo Methods: A Comparative Study." *Home*, 7 Apr. 2022, https://gupea.ub.gu.se/handle/2077/71238.

Pipis, George, et al. "Pricing of European Options with Monte Carlo." *Predictive Hacks*, 28 Dec. 2020, https://predictivehacks.com/pricing-of-european-options-with-monte-carlo/.

Agarwal, Kushal. "What Can the Monte Carlo Simulation Do for Your Portfolio?" *Investopedia*, Investopedia, 13 July 2022, https://www.investopedia.com/articles/investing/112514/monte-carlo-simulation-basics.asp#:~:text=Monte%20Carlo%20is%20used%20in,under%20analysis%20and%20its%20volatility.

Hayes, Adam. "Black-Scholes Model: What It Is, How It Works, Options Formula." *Investopedia*, Investopedia, 10 Jan. 2023, https://www.investopedia.com/terms/b/blackscholes.asp.

Damodaran, Aswath. *Risk-Free Rate*. Stern School of Business, New York University, Dec. 2008, http://people.stern.nyu.edu/adamodar/pdfiles/papers/riskfreerate.pdf.

# Appendix

**(1)**

```python
import math
from scipy.stats import norm
import yfinance as yf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

**(2)**

```python

omxs30_tickers = ["ABB.ST", "ALFA.ST",
"ASSA-B.ST", "ATCO-A.ST", "ATCO-B.ST", "BOL.ST",
"ELUX-B.ST", "ERIC-B.ST", "ESSITY-B.ST", "GETI-B.ST",
"HEXA-B.ST", "INVE-B.ST", "KINV-B.ST",
"NDA-SE.ST", "SAND.ST", "SINCH.ST", "SEB-A.ST", "SKA-B.ST", "SKF-B.ST", "SCA-B.ST",
"SHB-A.ST", "SWED-A.ST", "SWMA.ST", "TEL2-B.ST", "TELIA.ST", "VOLV-B.ST"]

```

**(3)**

```python
stock_data = {}
volatility = {}
price_list_MC = []
price_list_BS = []
price_list_Actual = []
lsd_list_MC = []
lsd_list_BS = []
strike_prices = []
```

**(4)**

```python
for ticker in omxs30_tickers:
    stock_data[ticker] = yf.download(ticker, start='2017-01-01', end='2020-01-01')

```

**(5)**

```python

    #Calculate Variance
    data = stock_data[ticker]['Close']
    mean = data.mean()
    variance = data.apply(lambda x: (x - mean)**2)
    sum_of_squared_variance = variance.sum()
    volatility[ticker] = (sum_of_squared_variance/(len(data)-1))**(1/2)
```

**(6)**

```python
    #Set Variables
    r = 0.016 #risk free rate
    sigma = volatility[ticker] #variance
    T = 0.08 #time to maturity
    S0 = stock_data[ticker]['Close'][-1] #spot price

```

(7)

```
    #range of strike price +/- 20%
    for k in range(80, 120):
        K=k/100*S0 #strike price
        strike_prices.append(K)
```

(8)

```
        #Calculate Black-Scholes Call Option
        d1 = (np.log(S0/K) + (r + sigma**2/2) * T)/(sigma * np.sqrt(T))
        d2 = d1 - sigma * np.sqrt(T)
        phid1 = norm.cdf(d1)
        phid2 = norm.cdf(d2)
        call_price_BS = S0 * phid1 - K * math.exp(-r * T) * phid2
        print(call_price_BS)
```

(9)

```
        #Calculate Monte-Carlo Call Option
        nSim=1000000
        Z = np.random.normal(loc=0, scale=1, size=nSim)
        WT = np.sqrt(T) * Z
        ST = S0*np.exp((r - 0.5*sigma**2)*T + sigma*WT)
        simulated_call_payoffs = np.exp(-r*T)*np.maximum(ST-K,0)
        call_price_MC = np.mean(simulated_call_payoffs)
        print(call_price_MC)
```

(10)

```
        #Get Actual Value Call Option
        stock_data[ticker] = yf.download(ticker, start='2020-01-01', end='2020-01-31')
        call_price_Actual = stock_data[ticker]['Close'][-1]
        print(call_price_Actual)
```

(11)

```
        #Save all Prices
        price_list_MC.append(call_price_MC)
        price_list_BS.append(call_price_BS)
        price_list_Actual.append(call_price_Actual)
```

(12)

```
        # Calculate logarithmic standard deviation
        log_call_price_BS = np.log(call_price_BS)
        log_call_price_MC = np.log(call_price_MC)
        log_call_price_Actual = np.log(call_price_Actual)
        log_stdBS = np.std([log_call_price_BS, log_call_price_Actual])
        log_stdMC = np.std([log_call_price_MC, log_call_price_Actual])
        lsd_list_BS.append(log_stdBS)
        lsd_list_MC.append(log_stdMC)
```

**(13)**

```
#Average Logarithmic Standard Deviation
average_lsd_MC = sum(lsd_list_MC) / len(lsd_list_MC)
print("MC:",average_lsd_MC)
average_lsd_BS = sum(lsd_list_BS) / len(lsd_list_BS)
print("BS:",average_lsd_BS)

```

**(14)**

```
#Sort Lists
strike_prices.sort()
price_list_BS.sort()
price_list_MC.sort()
price_list_Actual.sort()
```

**(15)**

```
#Plot Line Graph of MC, BS and Actual Call Option Values vs Strike Price
plt.plot(strike_prices, price_list_BS, label='Black-Scholes')
plt.plot(strike_prices, price_list_MC, label='Monte Carlo')
plt.plot(strike_prices, price_list_Actual, label='Actual')
plt.legend()
plt.xlabel('Strike Price')
plt.ylabel('Call Option Value')
plt.title('Strike Price vs Call Option Value')
plt.show()
```

**(16)**

```
#Plot Bar Graph of Logarithmic Standard Deviation
x = ['Monte Carlo', 'Black-Scholes']
y = [average_lsd_MC, average_lsd_BS]

plt.bar(x, y)
plt.xlabel('Method')
plt.ylabel('Logarithmic Standard Deviation')
plt.title('LSD Comparison')
plt.show()
```