Project on

Volatility Smiles and Stylized Facts in the Heston Model

Author:
Kamrul Hasan
M.Sc. in Actuarial and Financial Mathematics
Department of Mathematics

Supervisor:
Prof. Dr. Jörn Saß
Department of Mathematics

December 28, 2022

Abstract

In this project, the Black-Scholes and Heston models are observed. Option pricing(*call option*), simulations, Implied Volatility Smile and some stylized facts are presented here. In the Black-Scholes model, it is assumed that the volatility is constant, while the Heston model allows the stochastic volatility, which performs better, and is comparatively more preferable than the Black-Scholes model.

Acknowledgements

I would be delighted to thank my supervisor, *Professor Dr. Jörn Saß*, for giving an interesting topic of option pricing, supporting to complete the analysis, and advising me to write the project in LaTeX.
And I would also like to thank my family for their plenteous support.

Contents

1 Introduction

The Black-Scholes-Merton model (also called the Black-Scholes model) is proposed for option pricing in 1973 by Fischer Black, Myron Scholes and Robert Merton. It is the model which is the first analytic option pricing model[13]. Myron Scholes and Robert Merton (Black died already in 1995) received the 1997 Nobel Memorial Prize in Economic Sciences for their work, as stated in [7]. Although the Black-Scholes formula is often quite successful in explaining the stock option prices, it does have known biases [14]. It is important to have fair trading in a well-functioning market where portfolios with sure profits are not involved. The great disparity of the model is constant volatility.

On the other hand, to overcome the disparity of the Black-Scholes model, some stochastic volatility models are introduced by many researchers. Among them, Hull and White [11], Johnson and Shanno [15] and Wiggins [16] contributed a lot. Steven Heston proposed a stochastic volatility model as a generalization of the Black- Scholes model in 1993[3]. In this model, the randomness of the volatility of the underlying asset is considered. It describes stock options and other derivatives and it has also a closed-form solution. The solution of the technique is based on characteristic functions. In this model, a stock price S_t and variance v_t follow a Black-Scholes type stochastic process and a CIR process respectively.

In this project, we have discussed the European call option prices for the Black-Scholes model and Heston model. The simulation paths are observed on both models and the effects of changing the input parameters of option pricing are also presented. Besides, the Volatility smile in option pricing of the Black-Scholes model and Heston model are represented in the project. Furthermore, for log-returns in these models, we are assuming an arbitrage-free market since that is the only well-functioning type of market. Log returns for both models are also presented here. Additionally, for observing the skewness and kurtosis of the Heston model, we have figured out the effect of correlation(ρ) and the effect of volatility of variance(σ). These diagrams give us a better idea of the tails of distributions. Finally, returns, squared returns and absolute returns are observed for getting the idea that sequences are independent or not for both models.

2 Preliminary ideas

The stock price, S, follows a Geometric Brownian motion(GBM)(also known as exponential Brownian motion) and it is described by the following Stochastic Differential Equation(SDE),

$$dS = \mu S dt + \sigma S dW \tag{1}$$

where μ gives us the expected rate of return of the stock and σ shows the risk of the stock price under the probabiliy measure $\mathbb{P}$. And W is a process which is called Brownian motion(BM), as stated in [1].

The BM has two properties, which are continuous paths and independent increments. It is also considered as a Wiener process, which is a type of Markov process that is turned into a stochastic process to gain future value of a variable by using the current value of that variable. The discretization of the above equation (1) can be written as

$$\Delta S = \mu S \Delta t + \sigma S \sqrt{\Delta t} N \tag{2}$$

Here, $\mu \Delta t$ represents the expected value of the return and $\sigma \sqrt{\Delta t} N$ is the stochastic return in the small time period Δt, as follows from [4].

We can use the stock prices return for understanding the gain or loss of investment of an investor in the Black-Scholes model and Heston model in the short time period. To do that, we have to remember several important things which lead to the essential characters. To calculate the stochastic differential equations, Itô's lemma plays an important role. A market should be arbitrage-free because of a well-functioning market that ensures ***fair trading***. The *Risk-neutral pricing* and *Fundamental Theorem of Asset Pricing*, as defined in [5], show how a market can be *arbitrage-free* where a discounted asset is defined by,

$$\hat{S}_t = e^{-rt} S_t \tag{3}$$

where $\hat{S}_t$ are martingles under the probability measure $\mathbb{Q}$, which can be interpreted as risk-neutral trading and r is the risk-free interest rate under the risk-neutral measure $\mathbb{Q}$.

If we think at the time t, the price is S_t; which tells us that, at the beginning, the price will have fair, but that will not lead to an arbitrage opportunity market. So we can use characteristic functions, as defined in [5], by comparing the output of the models. The characteristic function is used in a different way in probability and is defined as the Fourier transform of the probability density function using Fourier transform parameters. *Girsanov's theorem*, as stated in [6], where *Novikov's condition* is satisfied, plays a massively important role in transforming the probability measure (historical measure) $\mathbb{P}$ to the risk neutral measure $\mathbb{Q}$.

3 The Black Scholes Model, PDE and Formula

3.1 The Black-Scholes Model(BSM)

In 1973, Fischer Black and Myron Scholes introduced a leading model for option pricing, as stated in [2]. The stock price is defined by the following equation,

$$dS_t = \mu S_t dt + \sigma S_t dW_t \tag{4}$$

Here, μ and σ are called expected return and volatility of the stock price receptively. These are measured under the probability measure $\mathbb{P}$.

A discrete approximation is given by,

$$\Delta S = \mu S \Delta t + \sigma S \Delta W \tag{5}$$

The solution of the stock price

$$S_t = S_0 e^{(\mu - \frac{\sigma^2}{2})t + \sigma W_t} \tag{6}$$

Our target is to obtain a risk-neutral process, to do this the pricing purposes would be under the risk-neutral measure $\mathbb{Q}$. So *Girsanov's theorem* is used to express the stock price S_t in the following form,

$$dS_t = rS_t dt + \sigma S_t d\tilde{W}_t \tag{7}$$

The Black-Scholes model will be compared with the Heston model. So we can consider the model under $\mathbb{Q}$ where μ is replaced by r which is called the risk-free interest rate. Here $\tilde{W}_t = W_t + \frac{\mu - r}{\sigma} t$. Since $\mu = r$, so $\tilde{W}_t = W_t$. And σ is the volatility of the risky asset(stock).

In the Black-Scholes model, the $\log return$ is normally distributed with mean $(r - \frac{\sigma^2}{2})\Delta t$ and variance $\sigma^2 \Delta t$ during the time interval $[t, t + \Delta t]$ and it is defined by the following way,

$$\ln\left(\frac{S_{t+\Delta t}}{S_t}\right) \sim N\left((r - \frac{\sigma^2}{2})\Delta t, \sigma^2 \Delta t\right)$$

3.1.1 Assumptions of the Black-Scholes model

There are several assumptions underlying the Black-Scholes model, as defined in [2]:

- ***Lognormal distribution***: Asset price can not take a negative value which is bounded by zero. The BSM assumes the stock prices follow log normal distribution.
- ***Risk free interest rate***: Risk free interest rate are assumed to be constant and also known.
- ***Efficient markets***: The model assumes that people can not predict consistantly the direction of the market or even an individual stock. The movement of a stock referred as a random walk.
- ***No dividends***: The BSM assumes that the stocks do not pay any dividends or returns.
- ***European-style options/Expiration date***: It is an European option.So it can be only exercised at maturity or expiraition date.
- ***No commissions and transaction costs***: There are no transaction costs for buying and selling options and stocks i.e. all securities are divisible.
- ***Liquidity***: It is possible to purchase or sell any amount of stock or options or their fractions at any given time. In short, markets are liquid.

3.2 Blsck-Scholes PDE

The PDE defined by the Black-Scholes model by using Itô's lemma for any underlying derivative f and current stock price S is

$$\frac{\partial f}{\partial t} + rS\frac{\partial f}{\partial S} + \frac{1}{2}\frac{\partial^2 f}{\partial S^2}S^2\sigma^2 = rf \tag{8}$$

An European call option must satisfy the equation (8) under the following conditions,

$$\begin{cases} f(S,T) = max(S-K,0) \\ f(0,t) = 0 \\ f(S \to \infty, t) = S - Ke^{-r(T-t)} \end{cases} \tag{9}$$

where K is the strike price or exercise price and T is the maturity time, r is the constant interest rate and σ is the volatility.

It is clear that in the equation (8) there is no μ. We can say that PDE is independent of risk, since μ rises with the risk level. So the expected rate of return will be a risk-free rate r.

3.3 The Black-Scholes Formula

The formula for the European call prices which gives the holder the right to purchase the asset at the strike price on the maturity date and it is defined by the following equation, as stated in [2] & [17],

$$\text{Call Option} = C(S_t, t) = S_t N(d_1) - N(d_2)Ke^{-r\tau} \tag{10}$$

where d_1 and d_2 are defined by,

$$d_1 = \frac{ln(\frac{S_t}{K}) + (r + \frac{\sigma^2}{2}\tau)}{\sigma\sqrt{\tau}}$$
$$d_2 = d_1 - \sigma\sqrt{\tau}$$

Here, $\tau = T - t$ is the time to maturity where T is the time at which an option can be exercised, r is the risk free interest rate, K is the exercise stock price and N is the standard normal cumulative distribution function.

S	r	T	σ
100	0.1	1	0.3

Table 1: Parameters used in the Black-Scholes Model.

4 Black-Scholes Call option price and effects of parameters

4.1 European call option price

To observe the ***European call option price***, the equation (10) is used where the parameters are taken from Table 1.

Strike price	Call price
98	17.7943
100	16.7341
102	15.7213

The following code is used for the call prices:

Code Listing 1: Black Scholes call price

```
import numpy as np
from scipy.stats import norm
import matplotlib.pyplot as plt
N = norm.cdf

def BS_CALL(S, K, T, r, sigma):
d1 = (np.log(S/K) + (r + sigma**2/2)*T) / (sigma*np.sqrt(T))
d2 = d1 - sigma * np.sqrt(T)
return S * N(d1) - K * np.exp(-r*T)* N(d2)
```

4.2 Effects of changing the input parameters

The following observations are represented here, by changing the input parameters, as follows from [17]. We get a good idea about the European call option price. Among the Figures, Figure 2 is very important and it tells us that there is a unique value of σ.

4.2.1 Current price effects on the Black Scholes call price

All the variables are constant except the current stock price S. Below the diagram represents how the call price changes corresponding to the S.

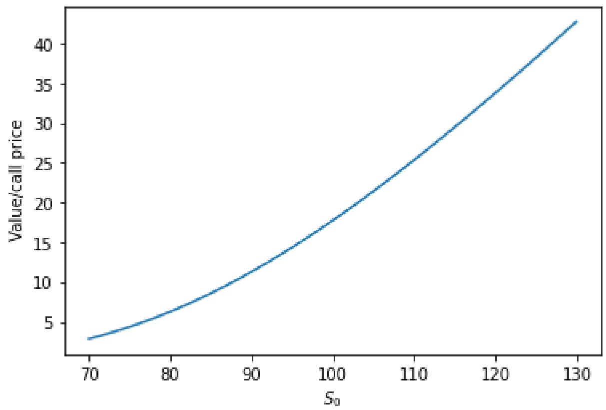


Figure 1: Changing the call price with S

The following code is used:

Code Listing 2: Black Scholes call price for different value of S

```
K = 98
r = 0.1
T = 1
sigma = 0.3

S = np.arange(70,130,0.1)

calls_for_changes_S = [BS_CALL(s, K, T, r, sigma) for s in S]

plt.xlabel('$S_0$')
plt.ylabel(' Value/call price')
plt.plot(S, calls_for_changes_S)
```

4.2.2 Sigma effect on Black-Scholes value

All parameteres are constant without sigma.

The following code is used:

Code Listing 3: Black Scholes call price for different value of S

```
  K = 98
  r = 0.1
  T = 1
  Sigmas = np.arange(0.1, 1.5, 0.01)
  S = 100
calls_for_changes_sigma = [BS_CALL(S, K, T, r, sig) for sig in Sigmas]
  plt.xlabel('$\sigma$')
  plt.ylabel(' Value/Call price')
  plt.plot(Sigmas, calls_for_changes_sigma)
```

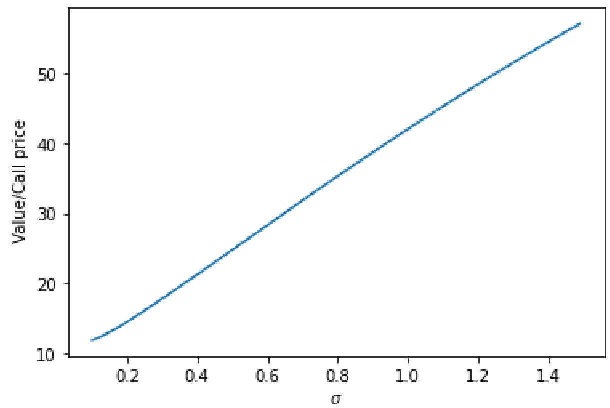


Figure 2: Changing the call price with σ

4.2.3 Effect of Time on Black-Scholes value

All parameteres are constant without Time, T.

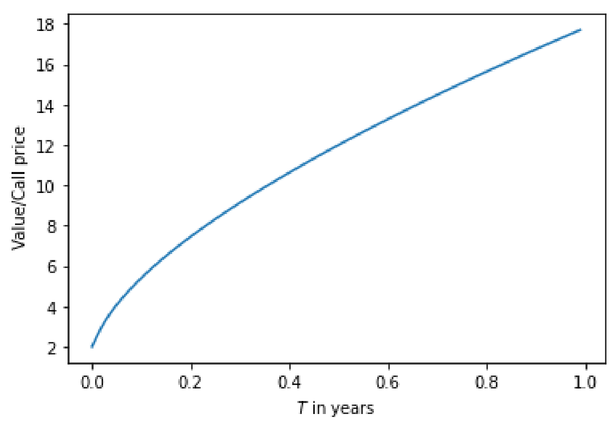


Figure 3: Changing the call price with T

The following code is used:

Code Listing 4: Black Scholes call price for different value of T

```
K = 98
r = 0.1
T = np.arange(0, 1, 0.01)
sigma = 0.3
S = 100

calls_for_changes_T = [BS_CALL(S, K, t, r, sigma) for t in T]
plt.xlabel('$T$ in years')
plt.ylabel(' Value/Call price')
plt.plot(T, calls_for_changes_T)
```

5 Option pricing by Monte Carlo Simulation of the Black Scholes Model

5.1 Simulation paths of the Black Scholos Model

The underlying asset, i.e. Stock price , follows a Brownian motion or Wiener-process. So these stochastic processes by simulating, we can calculate the price of financial instruments(option), as defined in [2].

$$dS_t = \mu S_t dt + \sigma S_t dW_t \tag{11}$$

We can take the $\log S(t)$ because the stock price can not take negative value. By using It$\hat{o}$'s lemma, we get

$$d \log S_t = (\mu - \frac{1}{2}\sigma^2)dt + \sigma dW \tag{12}$$

$$\log S_t = \log S_0 + (\mu - \frac{1}{2}\sigma^2)t + \sigma \int_0^t dW \tag{13}$$

Wiener process can be seen as a continuous-time version of a random walk with mean 0 and variance t i.e. $N(0,t)$. It can be also written as $\sqrt{t}N(0,1)$

Risk-neutral assumption: If we make a risk-neutral assumption, the μ becomes the risk- free interest rate(r). So

$$S_T = S_0 \exp[(r - \frac{1}{2}\sigma^2)t + \sigma\sqrt{T}N] \tag{14}$$

which defines the stock price at T maturity. We generate a large number of stock price estimates with this equation in Monte Carlo simulation.Option price, which is the expected value of a pay-off function. Then we have to use a discount factor because of the time value of money. *For the following values, we get the below simulated paths*

$S = 100, K = 98, T = 1, r = 0.1, \sigma = 0.3, steps = 252, N = 1000$

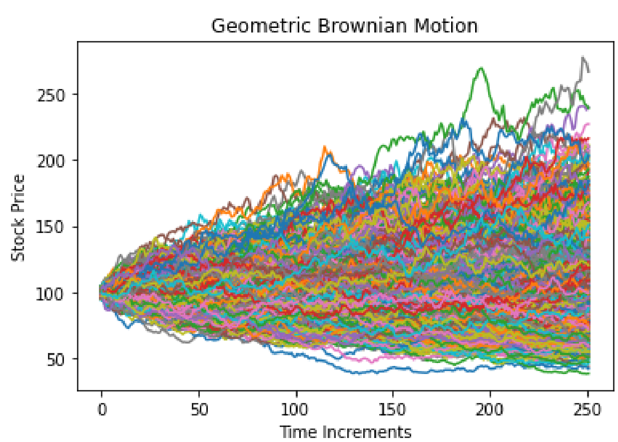


Figure 4: Simulation paths for $N = 1000$

The following code is used:

Code Listing 5: Black Scholes simulation paths for $N = 1000$

```
import numpy as np
from scipy.stats import norm
import matplotlib.pyplot as plt
N = norm.cdf

def geo_paths(S, T, r, sigma, steps, N):
dt = T/steps
#S_{T} = ln(S_{0})+\int_{0}^T(\mu-\frac{\sigma^2}{2})dt+\int_{0}^T \sigma dW(t)
ST = np.log(S) +  np.cumsum(((r - sigma**2/2)*dt +\
sigma*np.sqrt(dt) * \
np.random.normal(size=(steps,N))),axis=0)
return np.exp(ST)

S = 100 #stock price S_{0}
K = 98 # strike
T = 1 # time to maturity
r = 0.1 # risk free risk in annual %
sigma = 0.3 # annual volatility in %
steps = 252 # time steps
N = 1000 # number of trials

paths= geo_paths(S,T,r,sigma,steps,N)

plt.plot(paths);
plt.xlabel("Time Increments")
plt.ylabel("Stock Price")
plt.title("Geometric Brownian Motion")
```

If we take the number of trials for $N = 10000$, the simulated paths shows the following figure.

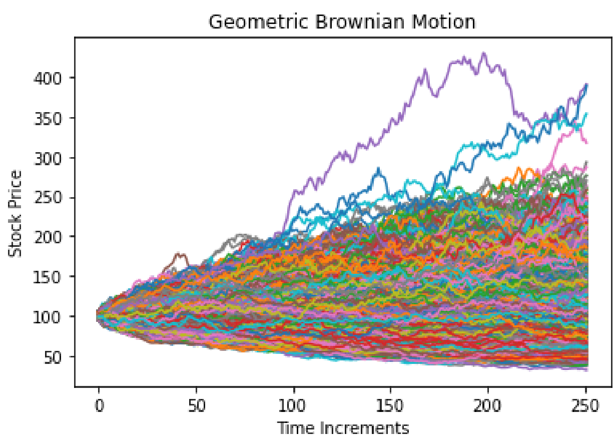


Figure 5: Simulation paths for $N = 10000$

The following python code is used:

Code Listing 6: Black Scholes simulation paths for $N = 10000$

```
import numpy as np
from scipy.stats import norm
import matplotlib.pyplot as plt
N = norm.cdf

def geo_paths(S, T, r, sigma, steps, N):
dt = T/steps
#S_{T} = ln(S_{0})+\int_{0}^T(\mu-\frac{\sigma^2}{2})dt+\int_{0}^T \sigma dW(t)
ST = np.log(S) +  np.cumsum(((r - sigma**2/2)*dt +\
sigma*np.sqrt(dt) * \
np.random.normal(size=(steps,N))),axis=0)
return np.exp(ST)
```

```
S = 100 #stock price S_{0}
K = 98 # strike
T = 1 # time to maturity
r = 0.1 # risk free risk in annual %
sigma = 0.3 # annual volatility in %
steps = 252 # time steps
N = 10000 # number of trials

paths= geo_paths(S,T,r,sigma,steps,N)

plt.plot(paths);
plt.xlabel("Time Increments")
plt.ylabel("Stock Price")
plt.title("Geometric Brownian Motion")
```

6 Compare Black-Scholes call price with simulation results

6.1 Black Scholes call price and simulated call price for N=1000

Black Scholes call price and simulated call price are given below for $N = 1000$ and $\sigma = 0.3$.

Call price for $N = 1000$ and $\sigma = 0.3$	
Black Scholes price	17.794
Simulated call price	16.275

The following code is used:

Code Listing 7: Black Scholes call price and simulated call price for $N = 1000$ and $\sigma = 0.3$

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


payoffs = np.maximum(paths[-1]-K, 0)
option_price = np.mean(payoffs)*np.exp(-r*T) #discounting back to present value

bs_price = BS_CALL(S, K, T, r, sigma)

print(f"Black Scholes Price is {bs_price}")
print(f"Simulated price is {option_price}")
```

6.2 Black Scholes call price and simulated call price for N=10000

Now, if we increase the number of N, suppose $N = 10000$, we get simulated call price with $\sigma = 0.3$ that is almost same as the Black-Scholes call price with $\sigma = 0.3$ and $N = 1000$.

Call price for $N = 10000$ and $\sigma = 0.3$	
Black Scholes price	17.794
Simulated call price	17.728

As we increase N towards infinity, the price approaches the Black-Scholes price, due to the Central Limit Theorem.

7 The Heston Model, PDE, Formula and Characteristics functions

7.1 The Heston Model

In 1993, Steven Heston, an American mathematician, economist, financier, and naturist, proposed a model which is a generalization of the Black-Scholes model, as stated in [3]. The risky asset, such as Stock, under the Heston model is derived from the following SDEs in article [3], [8] , [9]& [21].

$$dS_t = \mu S_t dt + \sqrt{v_t} S_t dz_1(t) \tag{15}$$

where v_t is the stochastic variance and μ is the drift of the process of the stock.

The instantaneous variance v_t follows a Cox-Ingersoll-Ross(CIR) process and it is defined by,

$$dv_t = \kappa(\theta - v_t)dt + \sigma\sqrt{v_t} dz_2(t) \tag{16}$$

In the equation (15) and (16), z_1 and z_2 are Wiener processes with correlation ρ which is equivalently with covariance ρdt as follows,

$$dz_1(t)dz_2(t) = \rho dt \tag{17}$$

In the above equations (15), (16) & (17), S_t is the underlying stock price which follows a geometric Brownian motion, κ is the mean-reversion rate, θ is the long-term variance, σ is the volatility of volatilty, z is a Wiener process and ρ is the correlation between the Brownian motions. And two initial parameters S_0 and v_0, which are non-negative.

In the equation (15) and (16), the stock price S_t and the variance v_t are measured under the probability measure $\mathbb{P}$. We can measure the S_t under the risk neutral measure $\mathbb{Q}$ by using Girsanov's theorem by the following expression,

$$dS_t = rS_t dt + \sqrt{v_t} S_t d\tilde{z_1}(t) \tag{18}$$

Equations (15) and (18) are exactly same under the probability measure $\mathbb{Q}$. Here $\tilde{z_1}(t) = z_1(t) + \frac{\mu - r}{\sqrt{v_t}} t$. Since $\mu = r$, so $\tilde{z_1}(t) = z_1(t)$.

The risk neutral **log price process**, $\ln S_t$, in equation(15) or (18) in defined by,

$$\ln S_t = \left(r - \frac{1}{2}\right) dt + \sqrt{v_t} dz_1(t) \tag{19}$$

The process v_t is strictly positive if the Feller condition is satisfied. It is given by,

$$2\kappa\theta > \sigma^2$$

According to CIR(1985), the mean and variance of v_t conditional on the value of $v_s (t > s)$ are respectively given below, as stated in [9] & [10]

$$Mean = E[v_t|v_s] = \theta + (v_s - \theta)e^{-\kappa(t-s)}$$

$$Variance = Var[v_t|v_s] = \frac{v_s \sigma^2 e^{-\kappa(t-s)}}{\kappa}(1 - e^{-\kappa(t-s)}) + \frac{\theta\sigma^2}{2\kappa}(1 - e^{-\kappa(t-s)})^2$$

The mean reversion rate κ is the important parameter here. When $\kappa \mapsto 0$, Mean $\mapsto v_s$ and Variance will be undefined. When $\kappa \mapsto \infty$, Mean $\mapsto \theta$ and Variance approaches 0. Variance, σ, could not take the value 0 because C_j and D_j will be undefined in equation (24).

The volatility of the variance parameter(σ) describes the tails of the probability distribution i.e. the parameter σ controls the ***Kurtosis***, which is defined by the follwing way,

$$Kurt[X] = E[(\tfrac{X-\mu}{\sigma})^4]$$

where mean μ and varinace σ^2 for random variable X. Kurtosis, which is the effect of the volatility clustering.

On the other hand, ***Skewness*** is defined by the formula,

$$Skew[X] = E[(\tfrac{X-\mu}{\sigma})^3]$$

The skewness of the density of $\ln(S_T)$ is controled by the parameter ρ and it is the correlation between Brownian motions $dz_1(t)$ and $dz_2(t)$.

7.2 The PDE of the Heston model

Under the Heston model, options are calculated by a certain PDE. The derivation in the Black-Scholes PDE and Heston PDE are the same. But in the Heston model we need another derivative in the portfolio because of the hedge of the volatility.
However, the lead of the PDE under the Heston model is given below in the articles [3], [9], [10] & [11]

$$\left(\frac{\partial f}{\partial t} + \frac{1}{2}vS^2\frac{\partial^2 f}{\partial S^2} + \sigma\rho vS\frac{\partial^2 f}{\partial S\partial v} + \frac{1}{2}v\sigma^2\frac{\partial^2 f}{\partial v^2}\right) - rf + rS\frac{\partial f}{\partial S} + \{\kappa(\theta - v) - \lambda(S, v, t)\}\frac{\partial f}{\partial v} = 0 \quad (20)$$

The market price of volatility risk is λ. The above PDE (20) satisfies the following boundary conditions for the European call option,

$$\begin{cases} f(S, v, T) = max(S - K, 0) \\ f(0, v, t) = 0 \\ \frac{\partial f}{\partial S}f(S \to \infty, v, t) = 1 \\ \frac{\partial f}{\partial t}(S, 0, t) + rS\frac{\partial C}{\partial S}(S, 0, t) + \kappa\theta\frac{\partial f}{\partial v}(S, 0, t) - rf(S, 0, t) = 0 \\ f(S, v \to \infty, t) = S \end{cases} \quad (21)$$

Now, equation (20) can be written as the following way, defined in the article [9],

$$\frac{\partial C}{\partial t} + AC - rC = 0$$

where $A = rS\frac{\partial}{\partial S} + \frac{1}{2}vS^2\frac{\partial^2}{\partial S^2} + \sigma\rho Sv\frac{\partial^2}{\partial S\partial v} + \frac{1}{2}\sigma^2 v\frac{\partial^2}{\partial v^2} + \{\kappa(\theta - v) - \lambda v\}$

The generator of the Heston model is defined by A, where first two terms as the generators of the Black-Scholes model with $v = \sqrt{\sigma}$.

7.3 The formula and characteristics functions of the Heston model

By analogy with the Black-Scholes formula, the solution of the form for European call option under the Heston model, as stated in [3], [9], [10]& [21].

$$C(S, v, t) = SP_1 - KP(t, T)P_2 \quad (22)$$

The price at time t of a unit discount bond that matures at time $t + \tau$ is

$$P(t, t + \tau) = e^{-r\tau}$$

where $P_j, j = 1, 2$ are risk neutral probabilities obtained by inverting the characteristic function f_j.

Now,

$$P_j(x, v, T; ln(K)) = Pr(\ln S_t > \ln K) = \frac{1}{2} + \frac{1}{\pi}\int_0^\infty Re\Big[\frac{e^{-i\phi\ln(K)}f_j(x, v, T; \phi)}{i\phi}\Big]d\phi \quad (23)$$

In the above equation,

$$
\begin{cases}
x = \ln[S] \\
f_j = \exp(C_j + D_j v(t) + i\phi \ln S(t) \\
C_j = r\phi i\tau + \frac{\kappa\theta}{\sigma^2}[(b_j - \rho\sigma\phi i + d_j)\tau - 2\ln(\frac{1-g_j \exp(d_j\tau)}{1-g_j})] \\
D_j = \frac{b_j - \rho\sigma\phi i + d_j}{\sigma^2}\frac{1-\exp(d_j\tau)}{1-g_j \exp(d_j\tau)} \\
g_j = \frac{b_j - \rho\sigma\phi i + d_j}{b_j - \rho\sigma\phi i - d_j} \\
d_j = \sqrt{(\rho\sigma\phi i)^2 - \sigma^2(2u_j\phi i - \phi^2)}
\end{cases}
\tag{24}
$$

For $j = 1, 2$ where $u_1 = 1/2, u_2 = -1/2, a = \kappa\theta, b_1 = \kappa + \lambda - \rho\sigma, b_2 = \kappa + \lambda$. The parameter λ is the market price of volatility risk.

We can implement the function to price call options. We price the option at time $t = 0$, therefore, $\tau = T$.

S_0	K	r	v_0	κ	θ	λ	σ	T
100	98	0.1	0.04	2	0.04	0	0.3	1

Table 2: Parameters used in the Heston Model.

8 European call option price under the Heston model

By using the equations (23) & (24), we get the equation (22) which is the formula of the Heston model and it is calculated below by using the parameters of Table 2.

Correlation between Brownian motions (ρ)	European call option price
0.1	14.328
-0.1	14.452

The following code is used:

Code Listing 8: Heston Call Price

```
import numpy as np
from scipy.integrate import quad

# Heston call price
def Heston_call_price(S0, v0, K, T, r, kappa, theta, sigma, rho, lambda):
p1 = p_Heston(S0, v0, K, r, T, kappa, theta, sigma, rho, lambda, 1)
p2 = p_Heston(S0, v0, K, r, T, kappa, theta, sigma, rho, lambda, 2)
return S0 * p1 - K * np.exp(-r*T) * p2

# Heston probability
def p_Heston(S0, v0, K, r,T, kappa, theta, sigma, rho, lambda, j):
integrand = lambda phi: np.real(np.exp(-1j * phi * np.log(K)) \
* f_Heston(phi, S0, v0, T, r, kappa, theta, sigma, rho, lambda, j) \
/ (1j * phi))
integral = quad(integrand, 0, 100)[0]
return 0.5 + (1 / np.pi) * integral

# Heston characteristic function
def f_Heston(phi, S0, v0, T, r, kappa, theta, sigma, rho, lambda, j):

if j == 1:
u = 0.5
b = kappa + lambda - rho * sigma
else:
u = -0.5
b = kappa + lambda

a = kappa * theta
d = np.sqrt((rho * sigma * phi * 1j - b)**2 - sigma**2 * (2 * u * phi * 1j - phi**2)
)
g = (b - rho * sigma * phi * 1j + d) / (b - rho * sigma * phi * 1j - d)
C = (r) * phi * 1j * T + (a / sigma**2) \
* ((b - rho * sigma * phi * 1j + d) * T - 2 * np.log((1 - g * np.exp(d * T))/(1 - g)
))
D = (b - rho * sigma * phi * 1j + d) / sigma**2 * ((1 - np.exp(d * T)) / (1 - g * np
.exp(d * T)))

return np.exp(C + D * v0 + 1j * phi * np.log(S0))
```

```
# Parameters
T = 1        # maturity
S0 = 100     # spot price
K = 98      # strike price
r = 0.1     # risk-free interest rate
v0 = 0.04   # initial variance
rho = 0.1  # correlation between Brownian motions
kappa = 2    # mean reversion rate
```

```
theta = 0.04 # Long term mean of variance
sigma = 0.3  # volatility of volatility
lambda = 0    # market price of volatility risk

# Option values
Vc = Heston_call_price(S0, v0, K, T, r,kappa, theta, sigma, rho, lambda) # call

print('Call price: ' + str(round(Vc, 3)))
```

Call price: 14.328

```
# Parameters
T = 1         # maturity
S0 = 100      # spot price
K = 98       # strike price
r = 0.1      # risk-free interest rate
v0 = 0.04    # initial variance
rho = -0.1  # correlation between Brownian motions
kappa = 2     # mean reversion rate
theta = 0.04 # Long term mean of variance
sigma = 0.3  # volatility of volatility
lambda = 0    # market price of volatility risk

# Option values
Vc = Heston_call_price(S0, v0, K, T, r,kappa, theta, sigma, rho, lambda) # call

print('Call price: ' + str(round(Vc, 3)))
```

Call price: 14.452

9 Simulation of the Heston Model

The Monte Carlo procedure for the Heston model is similar to that of the Black-Scholes model. The key difference is that we need to simulate the variance paths. The variance values are then used to generate the stock price paths. Also, we need to generate correlated standard normal random variables. Additionally, it is necessary to generate the full paths whereas we can directly simulate the stock price at maturity with Black-Scholes. Let's define some parameters and implement the Monte Carlo procedure.

9.1 Simulated Call Option price of the Heston Model for small number of paths

For the following parameters (*see the below coding area*), **the call price shows *14.340***.

```
import numpy as np
import matplotlib.pyplot as plt

# Parameters
T = 1            # maturity
S0 = 100          # spot price
K = 98         # strike price
r = 0.1         # risk-free interest rate
v0 = 0.04        # initial variance
rho = 0.1       # correlation between Brownian motions
kappa = 2        # mean reversion rate
theta = 0.04     # Long term mean of variance
sigma = 0.3      # volatility of volatility
n_steps = 252    # number of time steps
n_paths = 1000   # number of paths
n_blocks = 2000 # number of blocks
dt = T/n_steps   # time step

# Initialize arrays
Vc_list = np.zeros(n_blocks) # call array


for j in range(n_blocks):
# Correlated normal random variables
W1, W2 = np.random.multivariate_normal([0,0], [[1, rho], [rho, 1]], (n_steps,
                                                      n_paths)).T

# Initialize array for variance
v = np.zeros((n_steps + 1, n_paths)).T
v[:, 0] = v0

# Initialize array for stock
S = np.zeros((n_steps + 1, n_paths)).T
S[:, 0] = S0

# Compute the paths
for i in range(1, n_steps + 1):
S[:, i] = S[:, i-1] * np.exp((r  - 0.5*v[:, i-1])*dt \
+ np.sqrt(v[:, i-1])*np.sqrt(dt)*W2[:, i-1])
v[:, i] = np.abs(v[:, i-1] + kappa*(theta - v[:, i-1])*dt \
+ sigma*np.sqrt(v[:, i-1])*np.sqrt(dt)*W1[:, i-1])

# Compute the discounted option price for the block
Vc_list[j] = np.exp(-r*T)*np.mean(np.maximum(S[:,-1] - K, 0))

# Final option price (mean of the prices from each block)
Vc = np.mean(Vc_list)


print('Call price: ' + str(round(Vc, 3)))
```

9.2 Price paths and variance paths

The following code is used for price paths:

```
fig = plt.figure(figsize=(10, 7))
ax = fig.add_subplot(111)
for i in range(20):
ax.plot(S[i, :])
ax.set_xlabel(r'$t$')
ax.set_ylabel(r'$S_t$');
```

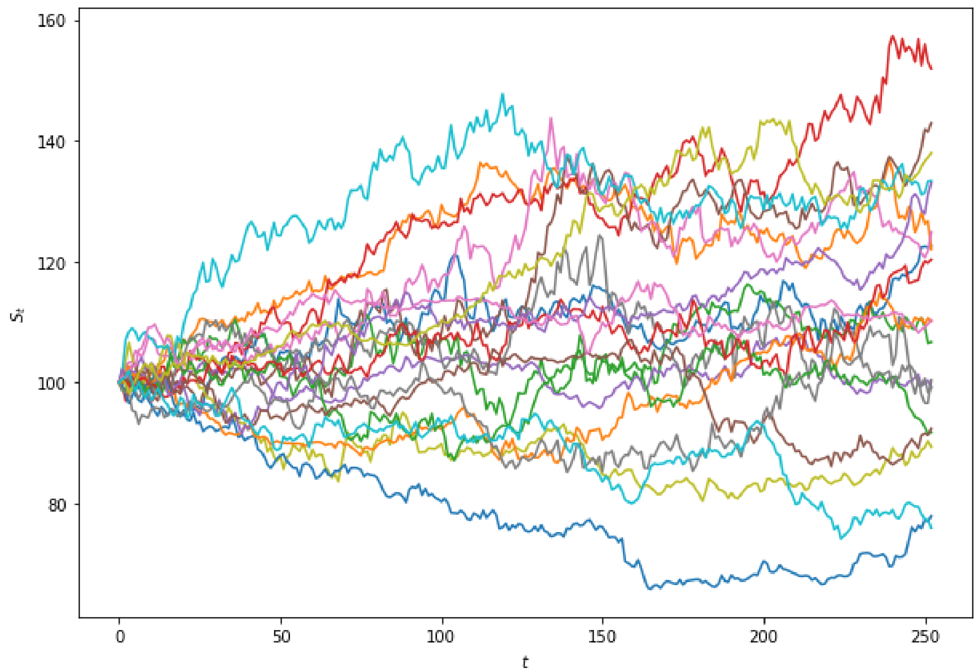


Figure 6: Price simulation paths for $N = 1000$

The following code is used for variance paths:

```
fig = plt.figure(figsize=(10, 7))
ax = fig.add_subplot(111)
for i in range(20):
ax.plot(v[i, :])
ax.set_xlabel(r'$t$')
ax.set_ylabel(r'$v_t$');
```

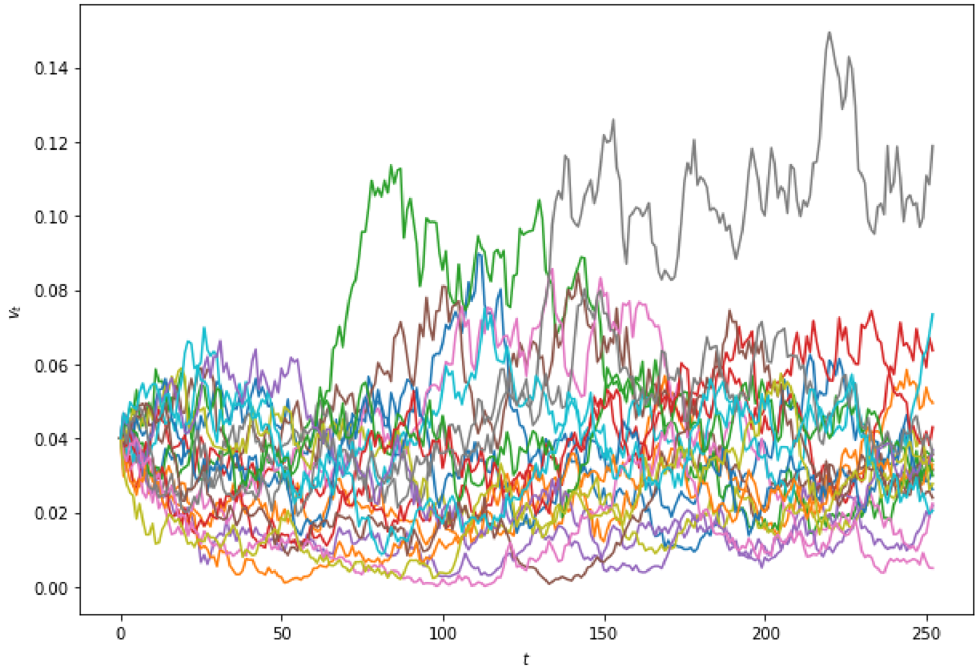


Figure 7: Variance simulation paths for $N = 1000$

9.3 Simulated Call Option price of the Heston Model for large number of paths

If we take the number of path for **10000** in the above code, **the call option price shows *14.331***.

Number of paths (N)	European call option price
1000	14.340
10000	14.331

Although we have increased the number of paths, the call price is almost same here.

9.4 Distribution of the stock price and variance

The following code is used for the distribution of Stock price:

```
import scipy.stats as ss

fig = plt.figure(figsize=(16,5))
ax = fig.add_subplot(121)

ax.hist(S[:,-1], density=True, alpha = 0.55, color = 'red',  bins=100,label="
                                            frequencies of $S_T$")
ax.legend(); ax.set_title("Stock price distribution"); ax.set_xlabel("$S_T$"); ax.
                                            set_ylabel("Density")

plt.show()
```

The following code is used for the distribution of variance:

```
import scipy.stats as ss

fig = plt.figure(figsize=(16,5))
ax = fig.add_subplot(121)

ax.hist(v[:,-1], density=True,alpha = 0.55, color = 'red', bins=100,label="frequencies
                                             of $v_T$")
```

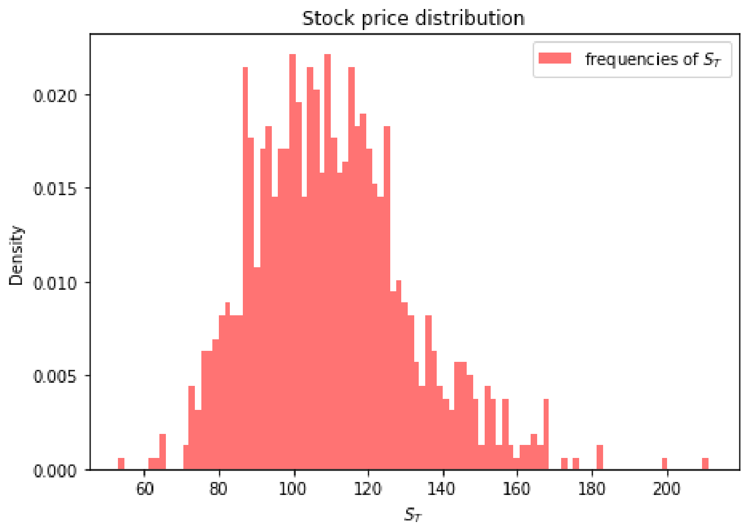


Figure 8: Stock price at maturity

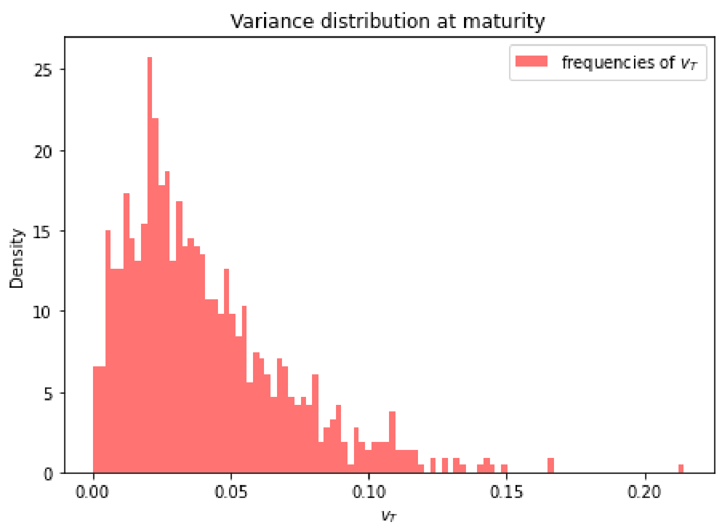


Figure 9: Variance at maturity

```
ax.legend(); ax.set_title("Variance distribution at maturity"); ax.set_xlabel("$v_T$")
                                              ;ax.set_ylabel("Density")
plt.show()
```

10 Volatililty Smile in Option Pricing

10.1 Implied volatility Smile from the Black-Scholes Model

In the *section 4*, we have observed European call option prices. Now we are going to demonstrate the **Implied volatility**. Imagine, we know everything except the volatility of the call price equation. The form of the equation will be, as defined in [18]:

$$17.794 = 100N\left(\frac{\ln(\frac{100}{98}) + (0.1 + \frac{\sigma^2}{2})1}{\sigma\sqrt{1}}\right) - N\left(\frac{\ln\frac{100}{98} + (0.1 + \frac{\sigma^2}{2})1}{\sigma\sqrt{1}} - \sigma\sqrt{1}\right) 98\exp(-0.1 * 1) \quad (25)$$

The σ parameter above is the volatility at which the Black-Scholes formula would return a value of 17.7940. So, essentially, once we solve for σ in the equation above, we have the implied volatility of the option price. Since the formula above cannot be solved explicitly, we must resort to iterative measures.

10.1.1 Implied volatility by Newton Raphson Algorithm

The Newton Raphson method is a widely used algorithm for calculating the implied volatility of an option. For solving the implied volatility, we can apply the following steps, as stated in [19]:

1. $f(\sigma) = V_{BS_\sigma} - V_{market}$
2. Initial guess $\sigma_0 = 0.3$ (Chosen the covariance)
3. Iterate as follows $\sigma_{n+1} = \sigma_n - \frac{V_{BS_\sigma} - V_{market}}{\frac{\partial V_{BS_\sigma}}{\partial \sigma}}$
4. If $|V_{BS_\sigma} - V_{market}| < \epsilon$, return σ_n

The above function computes the partial derivative of the value of a call option with respect to volatility. Normally, the partial derivative is referred to as Vega.

$$Vega = S\sqrt{T}N'(d_1) \quad (26)$$

Where d_1 is the same as we get from the Black-Scholes formula and N' is the probability density function for a standard normal. So we have a function of *Vega*. By making another function to implement the Newton Raphson method, we can find the implied volatility.

10.1.2 Implied volatility as a minimization problem

To compute the implied volatility, as defined in [18], we minimize the absolute difference between the market price and the Black-Scholes price. We use Scipy's *minimize_scalar* function that uses Brent's method. The minimization is bounded, such as volatility is less than 300%.

$$\text{lmplied volatility} = \arg\min_{\sigma} \left| V_{market} - V_{BS}(S, K, T, r, \sigma) \right| \quad (27)$$

Where V is the value of a European call or put option.
This is the more robust method than the Newton-Raphson Algorithm because of *vega*. In the Out of Money options, $Vega$ is closed to be zero.

10.1.3 Implied volatility by using the MC option price and py_vollib_vectorized

At different strike prices for capturing the volatility smile, strikes are taken by using numpy arange. We can calculate these prices using the Monte-Carlo approach. After that, we have to take the discounted expectation of the payoff. Then the implied volatility is computed from py_vollib_vectorized, as follows from [20]. This is the very simplest and dynamic way to compute the implied volatility. Here, this technique is used to discuss the implied volatility:

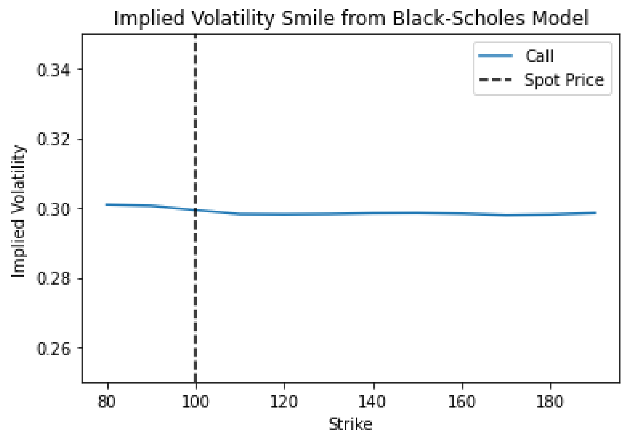


Figure 10: Implied volatility Vs Strike price

The following code is used:

```
from py_vollib_vectorized import vectorized_implied_volatility as implied_vol

# Set strikes and complete MC option price for different strikes
K = np.arange(80,200,10)

#number of trials N = 10000
calls = np.array([np.exp(-r*T)*np.mean(np.maximum(paths[-1]-k,0)) for k in K])

#call_ivs

y= implied_vol(calls, S, K, T, r, flag='c', return_as='numpy', on_error='ignore')


plt.ylim([0.25, 0.35])

plt.plot(K, y, label=r'Call')

plt.ylabel('Implied Volatility')
plt.xlabel('Strike')
plt.axvline(S, color='black',linestyle='--',
label='Spot Price')

plt.title('Implied Volatility Smile from Black-Scholes Model')
plt.legend()
plt.show()
```

10.2 Implied volatility Smile from the Heston Model

To get the volatility smile, we can follow the technique where MC option price and py_vollib_vectorized are used. This method has been discussed above to find the volatility simile from the Black-Scholes model. Now, If we replace the Monte-Carlo approch for the Heston model in the above technique instead of the Black-Scholes model , we can easily find out the implied volatility smile from the Heston model.

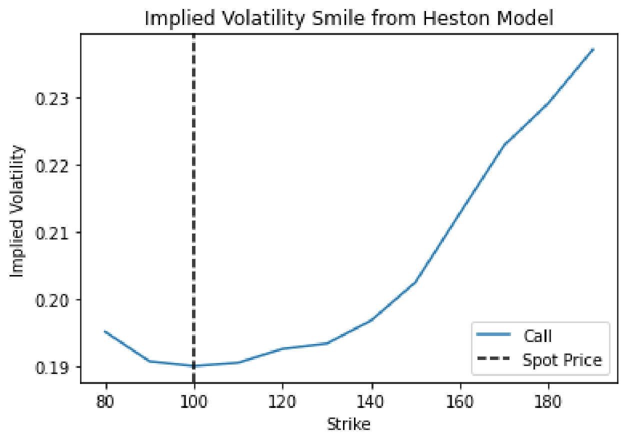


Figure 11: Implied volatility Vs Strike price

The following code is used:

```
from py_vollib_vectorized import vectorized_implied_volatility as implied_vol

#rho = 0.1
# Set strikes and complete MC option price for different strikes

K = np.arange(80,200,10)

calls = np.array([np.exp(-r*T)*np.mean(np.maximum(S[:,-1]-k,0)) for k in K])


call_ivs = implied_vol(calls, S0, K, T, r, flag='c', return_as='numpy', on_error='
                                                    ignore')

plt.plot(K, call_ivs, label=r'Call')


plt.ylabel('Implied Volatility')
plt.xlabel('Strike')
plt.axvline(S0, color='black',linestyle='--',
label='Spot Price')
plt.title('Implied Volatility Smile from Heston Model')
plt.legend()
plt.show()
```

11 Observing stylized facts

11.1 Log return in the Black-Scholes model

In the section 4, we have seen that,

$$S_T = S_0 \exp\{(r - \frac{1}{2}\sigma^2)t + \sigma\sqrt{T}N\} \tag{28}$$

The **log-return** is defined as $\ln(\frac{S_t}{S_0})$ and it can be expressed as,

$$\ln\left(\frac{S_t}{S_0}\right) = \left(r - \frac{1}{2}\sigma^2\right)t + \sigma\sqrt{T}N \tag{29}$$

which is follows the *Normal distribution* i.e. $\ln(\frac{S_t}{S_0}) \sim N((r - \frac{1}{2}\sigma^2), \sigma\sqrt{T})$. It is a stochastic process that is normally distributed around its (time-dependent) mean and it has a (time-dependent) standard deviation.

The following graph shows the log return distribution of the Black Scholes model:

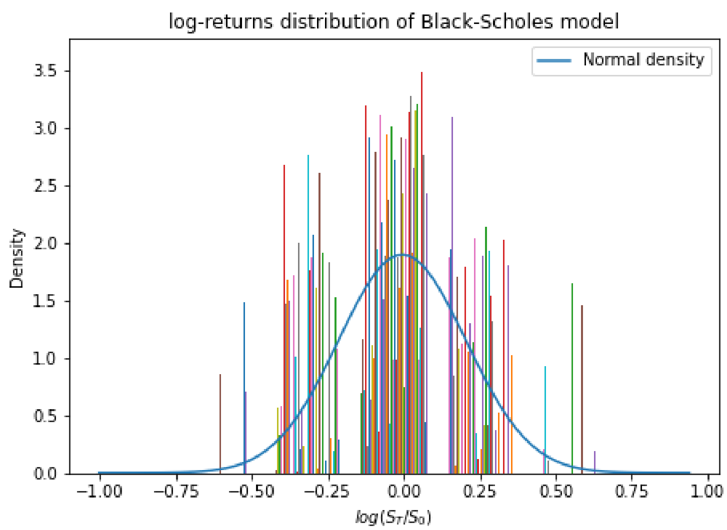


Figure 12: Log return distribution of the Black-Scholes model

The following code is used:

```
import numpy as np
from scipy.stats import norm
import matplotlib.pyplot as plt
N = norm.cdf

def log_return(S, T, r, sigma, steps, N):
dt = T/steps

return (r*dt)-(0.5*sigma**2)*dt+np.cumsum(sigma*np.sqrt(dt)*np.random.normal(size=(
                                        steps, N)), axis=0)


S = 100 #stock price S_{0}
K = 98 # strike
T = 1 # time to maturity
r = 0.1 # risk free risk in annual %
sigma = 0.3 # annual volatility in %
steps = 252 # time steps
N = 1000 # number of trials
```

```
log_R= log_return(S,T,r,sigma,steps,N)

import scipy.stats as ss
x = np.linspace(log_R.min(), log_R.max(), 100)
y = np.linspace(0.00001, 0.3, 500)

fig = plt.figure(figsize=(16,5))
ax = fig.add_subplot(121)

ax.plot(x, ss.norm.pdf(x, log_R.mean(), log_R.std(ddof=0)),label="Normal density")
ax.hist(log_R, density=True, bins=7)
ax.legend(); ax.set_title("log-returns distribution of Black-Scholes model"); ax.
                                            set_xlabel("$log(S_T/S_0)$")
plt.show()
```

11.2 Log return in the Heston model

In the section 7, we have seen the stochastic differential equations of the stock price process and volatility process. In equation (17), where ρ represents the correleation of the Brownian motions. The solution of the price process S_t on the interval d_t is log-normal and it is known to be,

$$S_{t+dt} = S_t \exp\{(\mu_t - \frac{1}{2}v_t)dt + \sqrt{v_t}W_{dt}\} \tag{30}$$

Dividing by S_t and taking log on equation (30), the **log return process** is given by,

$$Return = \log\left(\frac{S_{t+dt}}{S_t}\right) = \left(\mu_t - \frac{1}{2}v_t\right)dt + \sqrt{v_t}W_{dt} \tag{31}$$

which implies that the distribution of the log returns,

$$Return = \log\left(\frac{S_{t+dt}}{S_t}\right) \sim N\left(\left(\mu_t - \frac{1}{2}v_t\right)dt, v_t dt\right)$$

The following graph shows the log return distribution of the Heston model:

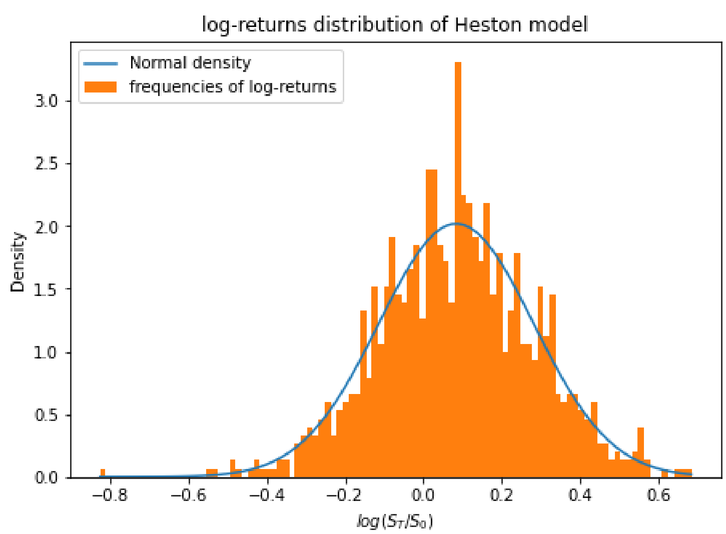
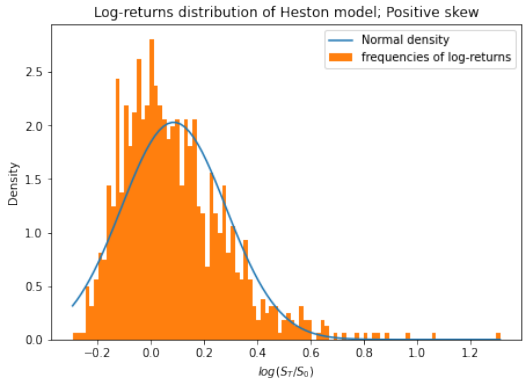


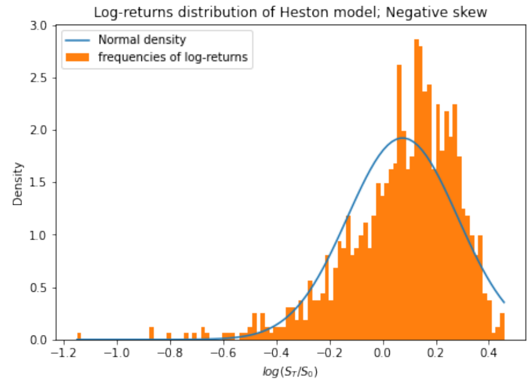
Figure 13: Log return distribution of the Heston model

The following code is used:

```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as ss

# Parameters
T = 1            # maturity
S0 = 100          # spot price
K = 98         # strike price
r = 0.1         # risk-free interest rate
v0 = 0.04        # initial variance
rho = 0.1       # correlation between Brownian motions
kappa = 2        # mean reversion rate
theta = 0.04     # Long term mean of variance
sigma = 0.3      # volatility of volatility
n_steps = 252    # number of time steps
n_paths = 1000   # number of paths
n_blocks = 2000 # number of blocks
dt = T/n_steps   # time step

# Initialize arrays
Vc_list = np.zeros(n_blocks) # call array


for j in range(n_blocks):
# Correlated normal random variables
W1, W2 = np.random.multivariate_normal([0,0], [[1, rho], [rho, 1]], (n_steps,
                                          n_paths)).T

# Initialize array for variance
v = np.zeros((n_steps + 1, n_paths)).T
v[:, 0] = v0

# Initialize array for stock
S = np.zeros((n_steps + 1, n_paths)).T
S[:, 0] = S0

# Compute the paths
for i in range(1, n_steps + 1):
S[:, i] = S[:, i-1] * np.exp((r - 0.5*v[:, i-1])*dt \
+ np.sqrt(v[:, i-1])*np.sqrt(dt)*W2[:, i-1])
v[:, i] = np.abs(v[:, i-1] + kappa*(theta - v[:, i-1])*dt \
+ sigma*np.sqrt(v[:, i-1])*np.sqrt(dt)*W1[:, i-1])

# Compute the discounted option price for the block
Vc_list[j] = np.exp(-r*T)*np.mean(np.maximum(S[:,-1] - K, 0))

import scipy.stats as ss

log_R = np.log(S[:, i]/S[:, 0])
x = np.linspace(log_R.min(), log_R.max(), 100)
y = np.linspace(0.00001, 0.3, 500)

fig = plt.figure(figsize=(16,5))
ax = fig.add_subplot(121)

ax.plot(x, ss.norm.pdf(x, log_R.mean(), log_R.std(ddof=0)), label="Normal density")
ax.hist(log_R, density=True, bins=100,label="frequencies of log-returns")
ax.legend(); ax.set_title("log-returns distribution of Heston model"); ax.set_xlabel
                                          ("$log(S_T/S_0)$"); ax.set_ylabel("Density"
                                          )
plt.show()
```



(a) Log return distribution of the Heston model for $\rho = 0.9$



(b) Log return distribution of the Heston model for $\rho = -0.9$

Figure 14: Log return distribution of the Heston model

If we increase the correlation between Brownian motions, we get a positively skewed log return distribution of the Heston model. And for the negative value of ρ, it gives us a negatively skewed log return distribution of the model. Suppose for, receptively, $\rho = 0.9$ and $\rho = -0.9$, the above diagrams represent the positive and negative skewed.

The skewness for $\rho = 0.9$	1.188
The skewness for $\rho = -0.9$	-1.232

S_0	K	r	v_0	κ	θ	λ	T	σ	ρ
100	98	0.1	0.04	2	0.04	0	1	0.3	0

Table 3: Parameters used for observing skewness and kurtosis under the Heston Model.

11.3 Effect of correlation(ρ) and volatility of variance(σ)

The $\ln S_t$ i.e.(log stock price) under the Heston model follows non-Gaussian distribution, as stated in [9] &.[12]. By using the $\ln S_t$, the skewness and kurtosis at maturity are shown on depending the choice of parameters. Table 3 is used for observising the skewness and kurtosis on the log stock value distribution for the different values of ρ and σ.

Figure 15: The effect of correlation on Heston Density Function.

Effect of correlation (ρ):

The correlation(ρ) between the stock price process and the volatility process controls the heaviness of tails, which is the skewness of the density of $\ln S_T$. Figure 15 represents the effect of the correlation on Heston density function, where Table 3 is used for different values of ρ.

When $\rho = 0$, the skewness will be close to 0. It is represented in the middle diagram of the figures. For $\rho > 0$, the probability densities are positively skewed(upper-left diagram) i.e. the variance rises with the increase in stock prices. It wides the right tail and squeezes the left tail of the distribution. In the case of Out of the Money(OTM) call, where the strike price is larger than the stock price($S_T < K$), lies in the right tail. In short, we can say the OTM calls using the Heston model should be more costly than the Black-Scholes model call price. On the contrary, strike price is smaller than stock price ($S_T > K$) in In-The-Money(ITM) call options, i.e. a strike price lying in the left tail. So, the ITM calls of the Heston model should be cheaper than the Black-Scholes model call price.

Furthermore, for $\rho < 0$, the probability densities are negatively skewed, i.e. the volatility rises if stock prices decreases. In the OTM and ITM call options, counter behavior is observed.

Effect of Volatility of Variance(σ):

Figure 16 represents the effect of volatility of variance on Heston PDF by using Table 3. The σ effects the peak, which is the kurtosis of the distribution of the log stock price under the Heston model. For $\sigma = 0$ in equation (16), the variance process v_t is dropped out. That is why for $\sigma = 0$, the log stock price is normally distributed, which is totally clear in the left diagram of Figure 16. The volatility of variance of last two diagrams (middle and right) are $\sigma = 0.4$ and 0.8 respectively. It becomes clear that increasing σ rises the peaks of the distribution and also create heavy tails on bothsides.

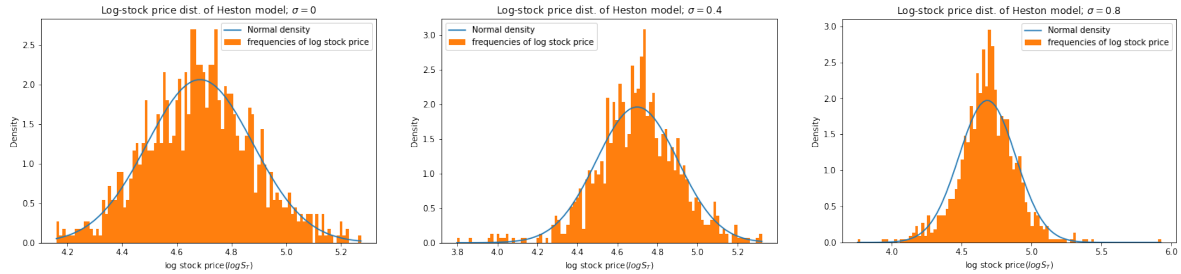


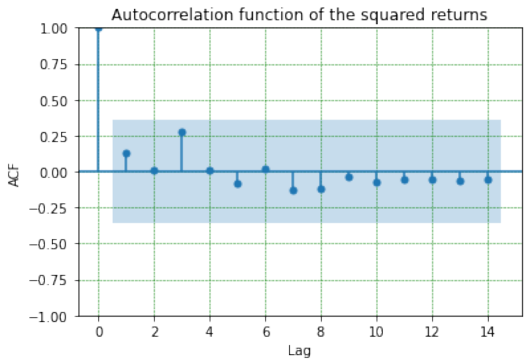
Figure 16: The effect of Volatility of Variance on Heston Density Function.

11.4 Autocorrelation and partial autocorrelation functions of the Black-Scholes model

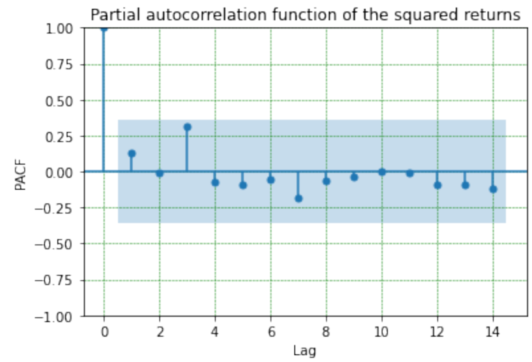
Lags in the squared return, absolute return and return in both functions of the Black-Scholes model go into different direction (Figure;17-19), so an interpretation is very difficult.

11.5 Autocorrelation and partial autocorrelation functions of the Heston model

In this model, all lags of the squared returns R_t^2, absolute returns $|R_t|$ and returns R_t in both functions are close to zero in Figure: 20-22, so there is no autocorrelation.

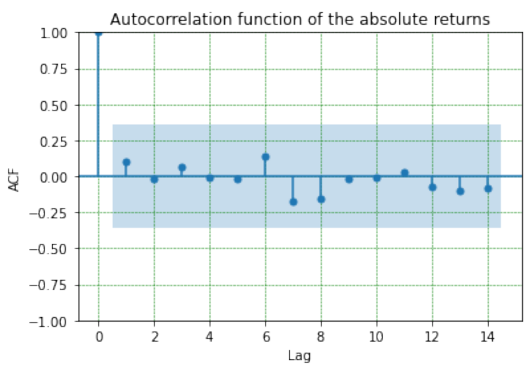


(a) Autocorrelation function

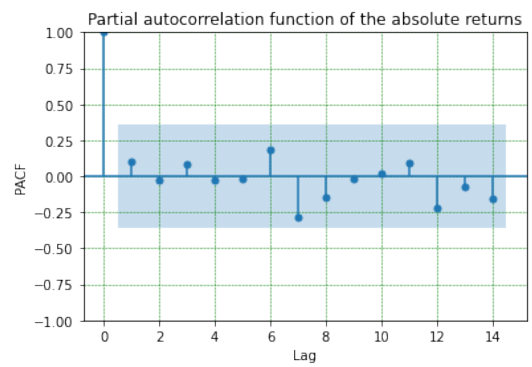


(b) Partial autocorrelation function

Figure 17: Autocorrelation and partial autocorrelation function of squared return (R_t^2)of the Black-Scholes model

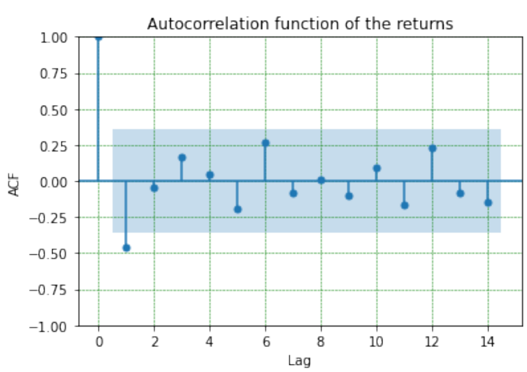


(a) Autocorrelation function

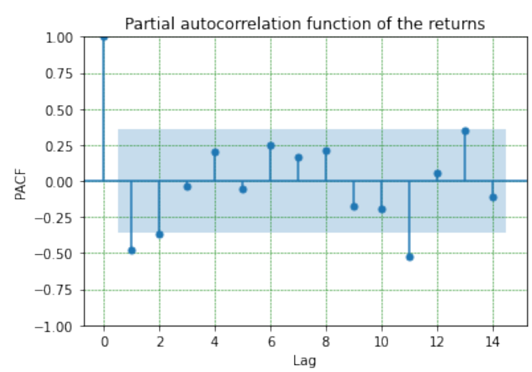


(b) Partial autocorrelation function

Figure 18: Autocorrelation and partial autocorrelation function of absolute return $|R_t|$of the Black-Scholes model

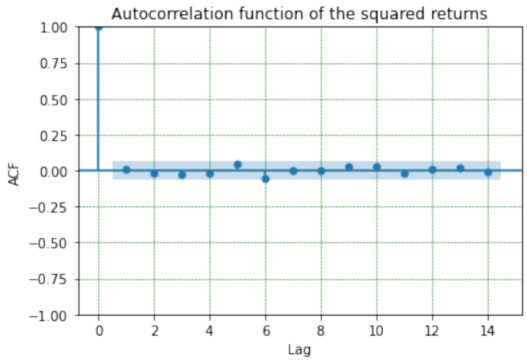


(a) Autocorrelation function

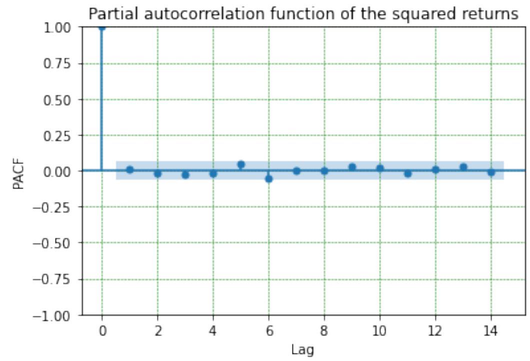


(b) Partial autocorrelation function

Figure 19: Autocorrelation and partial autocorrelation function of return R_t of the Black-Scholes model

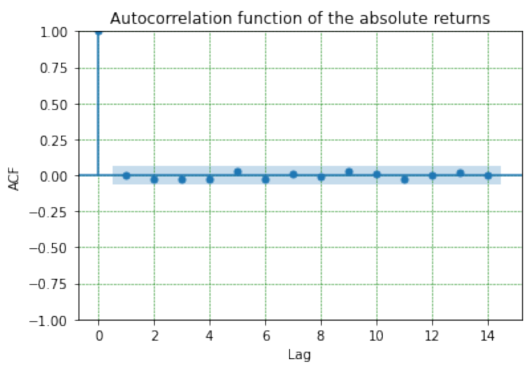


(a) Autocorrelation function

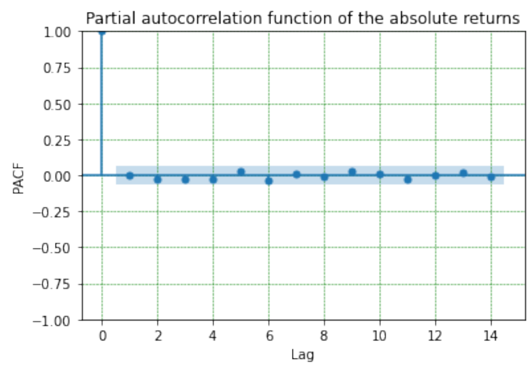


(b) Partial autocorrelation function

Figure 20: Autocorrelation and partial autocorrelation function of squared return (R_t^2)of the Heston model

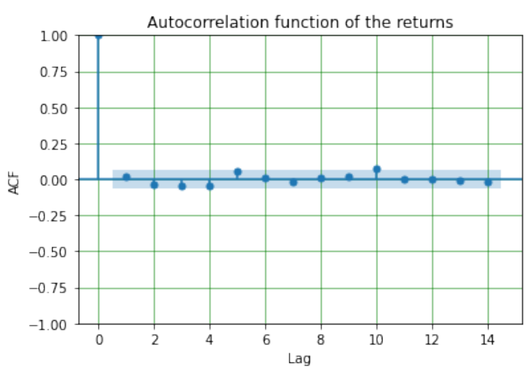


(a) Autocorrelation function

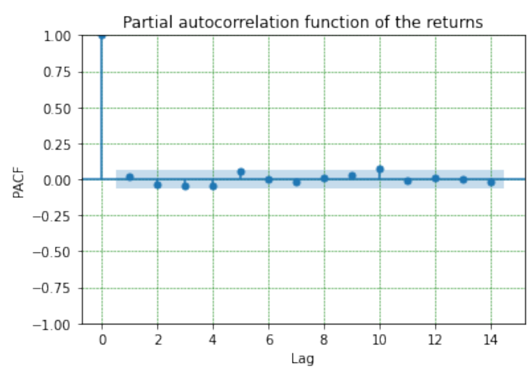


(b) Partial autocorrelation function

Figure 21: Autocorrelation and partial autocorrelation function of absolute return $|R_t|$ of the Heston model



(a) Autocorrelation function



(b) Partial autocorrelation function

Figure 22: Autocorrelation and partial autocorrelation function of return R_t of the Heston model

12 Conclusion

In the Black-Scholes model, the European call option prices in Figure 2 can be presented as a stickily increasing functions of the implied volatility σ. That tells us that there is a unique value of σ. It is shown in Figure 10, which is the smile effect(Implied volatility smile) of the model that is almost the same, everywhere. On the other hand, under the Heston model, the volatility is graphically presented in Figure 11, which is the smile effect of the model. It is clear that the volatility is stochastic in this model.

We have observed the European call option price and simulated call option price under the Black-Scholes model. It is obvious that simulated call prices are going to close to the Black-Scholes call price by increasing the number of paths(N). Under the Heston model, European call option prices and simulated call prices are almost the same although we have risen the number of paths(N).

The log-return distributions in the Black-Scholes model and Heston model are represented in Figure 12 and Figure 13 respectively. It becomes apparent that the log return of the Black-Scholes model is more flatter than the log return of the Heston model. In Figure 14, when we increase the correlation between Brownian motions(ρ), the log-return distribution is positively skewed (left diagram) under the Heston model. On the other side, we get a negatively skewed (right diagram) log-return distribution for negative correlation(ρ) under the model. Furthermore, Figure 15 and Figure 16 under the Heston model show the effect of correlation(ρ), which represents the skewness of the density of $\ln S_T$, and the effect of the Volatility of Variance(σ) that shows the kurtosis of the distribution of the log stock price, respectively.

In the both models, the autocorrelation function and partial autocorrelation function are observed. Squared returns R_t^2 for the Black-Scholes model and Heston model are not an independent sequence. Besides, absolute returns in the Black-Scholes model are independet sequence i.e. absolute returns $|R_t|$ are uncorrelated. On the contrary, in the Heston model the absolute returns are correlated. Furthermore, returns R_t in the Black-Scholes model and Heston model are uncorrelated and correlated respectively.

In short, we can say that the Heston model should produce more expensive In-The-Money(ITM) call price than the Black-Scholes call price. Under the Heston model, Out of the Money(OTM) call price should be less expensive.

References

[1] Stefano M. Iacus. Simulation and Inference for Stochastic Differential Equations, Springer-Verlag New York, 2008.

[2] Fisher Black, Myron Scholes. *The Pricing of Options and Corporate Liabilities, Journal of Political Economy*, Volume 3, 1973.

[3] Steven L. Heston. *A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options*,The Review of Financial Studies, Volume 6, 1993.

[4] John Hull. *Options, futures, and other derivatives*,Pearson Education, New York, 2018.

[5] Rama Cont, Peter Tankov. *Financial Modeling With Jump Processes*, Champan & Hall/CRC, 2004.

[6] Bernt Øksendal. *Stochastic Differential Equations*, Springer-Verlag Berlin Heidelberg,2003.

[7] Jörn Sass. *Lecture notes in Financial Mathematics*, University of Kaiserslautern, Summer (2022).

[8] Ser-Huang Poon. *A Practical Guide to Forecasting Financial Market Volatility*, John Wiley and Sons limited (2005).

[9] Tahmid Tamrin Suki, A B M Shahadat Hossain. *A comparative analysis of the Black-Scholes-Merton model and the Heston stochastic volatility model, J. Bangladesh Math. Soc.*, Vol. 39 (2019) 127-140.

[10] Fabrice Douglas Rouah. *The Heston Model and Its Extensions in Matlab and C++*, John Wiley and Sons, Inc., Hoboken, New Jersey (2013).

[11] Hull J. and White A. *The Pricing of Options on Assets with Stochastic Volatilities. Journal of Finance*, 42 (1987), 281-300.

[12] Mitun Kumar Mondal, Md. Abdul Alim, Md. Faizur Rahman, Md. Haider Ali Biswas, *Mathematical Analysis of Financial Model on Market Price with Stochastic Volatility. Journal of Mathematical Finance*, 07(02):351-365

[13] John C. Hull, *Options, Futures and Other Derivatives*, Global edition, Eighth edition, Pearson Education Limited (2015-2016)

[14] Rubinstein, M., *Non parametric Tests of Alternative Option Pricing Models Using All Reported Trades and Quotes on the 30 Most Active CBOE Option Classes from August 23, 1976 through August 31, 1978, Journal of Finance*, 40, 455-480.

[15] Johnson H. and Shanno D., *Option Pricing when the Variance Is Changing. Journal of Finance and Quantitative Analysis*, 22 (1987), 143-151.

[16] Wiggins J. B., *Option Values under Stochastic Volatility: Theory and Empirical Estimates. Journal of Financial Economics*,19 (1987), 351-372.

[17] John, *Black Scholes Model Python*, https://www.codearmo.com/python-tutorial/options-trading-black-scholes-model, December 22, 2020.

[18] John, *Calculating the Volatility Smile*, https://www.codearmo.com/python-tutorial/calculating-volatility-smile, January 03, 2021.

[19] John, *Implied Volatility for European Call with Python*, https://www.codearmo.com/blog/implied-volatility-european-call-python, September 08, 2020.

[20] *Implied Volatility*, https://py-vollib-vectorized.readthedocs.io/en/latest/pkg_ref/iv.html.

[21] Robin-Guilliou,*Option pricing*, Heston model, Updated on 11 Jan, 2022.