

Project- Calculate Implied Volatility

Quantitative Finance Cohort

Team Members:

1. Somanshu
2. Aryan
3. Mb Vamshi
4. Nitin kumar

Introduction

The project calculates the implied volatility of call option from Black Scholes partial Differential Equation using Newton Raphson Method and Bisection Method using python language . Also we create a python-based GUI interface application for taking necessary input from frontend and return implied volatility using these two methods.

Feature

- **Black Scholes Model:**

The Black-Scholes Model (also known as the Black-Scholes-Merton model) is a mathematical framework for pricing European-style options. The model provides a theoretical estimate of the price of a call or put option by considering factors such as the stock price(S), strike price(K), time to expiration(T), risk-free interest rate(r), and volatility(sigma).

Black-Scholes Model Formula

In mathematical notation:

$$C = S_t N(d_1) - K e^{-rt} N(d_2)$$

where:

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

and

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

C	Call option price
N	CDF of the normal distribution
S _t	Spot price of an asset
K	Strike price
r	Risk-free interest rate
t	Time to maturity
σ	Volatility of the asset
T= (T ₁ -t)	Time left till maturity(in years)

We all know that volatility measure expected price variation in the underlying And play essential role in setting the price in the option of the stock. On the other hand, Implied Volatility(IV) is computed from option market prices and dependent on the market's expectations for future price fluctuations. As we know the call option price and the other values in the black Scholes model, the formula can be used to derive implied volatility by iteratively adjusting sigma until theoretical option price is close to observed market price till certain accuracy.

- **Newton-Raphson Method:**

The **Newton-Raphson method** is an iterative numerical technique used to find the roots of a real-valued function. It is widely used because of its fast convergence properties when the initial guess is close to the true root.

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

The method leverages the derivative of the function to iteratively refine the estimate of the root. This process is repeated until the difference between successive approximations is below a predefined tolerance level, ensuring convergence to a solution. Here we iterate the formula until the difference of (n+1)th term and nth term is less 0.00001. Here, the function $f(x(n))$ represents the difference between the theoretical Black-Scholes option price and the observed market price.

- **Bisection Method:**

The **Bisection Method** is a simple and reliable numerical technique used to find the root of a continuous function within a given interval. It is based on the **Intermediate Value Theorem**, which states that if a function $f(x)$ is continuous in the interval $[a,b]$ and $f(a)$ and $f(b)$ have opposite signs, then there must be at least one root x within that interval where $f(x)=0$. The method works by repeatedly dividing the interval in half and selecting the subinterval where the function changes sign. This process continues until the interval becomes sufficiently small, ensuring an accurate approximation of the root.

- **GUI Interface**

A **Graphical User Interface (GUI)** for the Black-Scholes model allows users to input necessary parameters such as stock price (S_0), strike price (K), risk-free rate (r), time to expiration (T), and observed option price in an intuitive and user-friendly manner. The GUI typically consists of input fields, dropdown menus, and buttons for users to select the desired method—Newton-Raphson or Bisection—for computing the implied volatility. Additionally, the interface can display results dynamically, showing real-time calculations and error handling messages if convergence issues arise. A well-designed GUI enhances accessibility, allowing even non-technical users to compute implied volatility efficiently.

Once the user enters the required inputs and selects a numerical method, the GUI triggers a backend computation where the Newton-Raphson method (for fast convergence) or the Bisection method (for robustness) is applied to estimate the implied volatility. The calculated value is then displayed on the interface, along with details like the number of iterations, execution time, or graphical representations of convergence. The GUI can also include additional features such as exporting results, error handling messages, and visualization of option price behavior under different volatilities. Python libraries like Tkinter, PyQt, or Streamlit can be used to build such a GUI, making financial modelling more interactive and efficient.

Advantages

- **Robustness:** Including both numerical methods Newton-Raphson and Bisection method ensures reliable results under different scenarios. If one method fails to converge, the other may succeed.
- **User-Friendly Interface:** The GUI simplifies inputting parameters and interpreting results, making the tool accessible even to non-technical users.
- **Flexibility:** The model supports a wide range of input parameters, such as varying spot prices, strike prices, and maturities, allowing for diverse option-pricing scenarios.

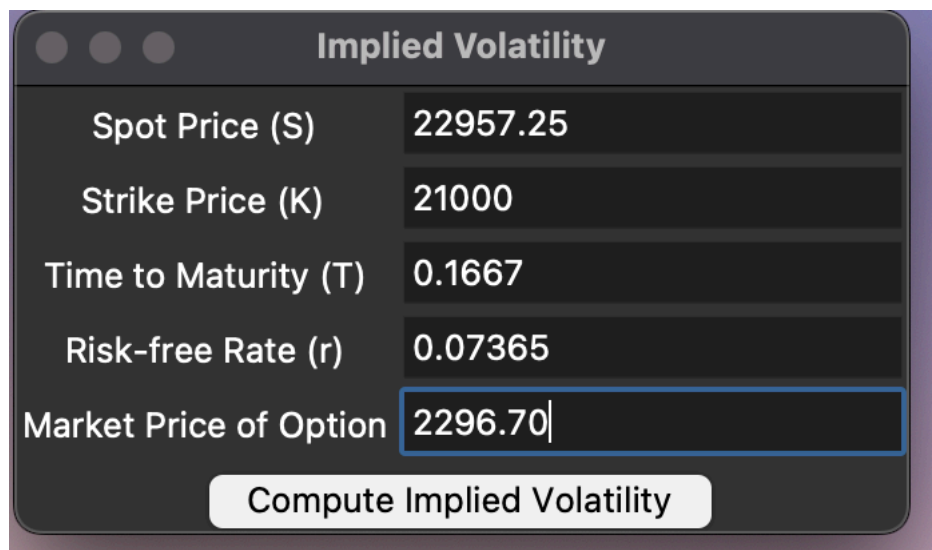
Disadvantages

- **European Option Assumption:** The Black-Scholes model is only applied to European-style options, which cannot be exercised before maturity. This limits its application for American-style options.
- **Model Assumptions:** The model assumes constant volatility and risk-free rates, which might not hold true in real-world markets, leading to inaccurate results.
- **Numerical Method Limitations:**
 - **Newton-Raphson Method:** may fail to converge if the derivative (vega) is near zero or if the initial guess is poor.
 - **Bisection Method:** slower convergence compared to Newton-Raphson method and requires predefined bounds for the volatility.
- **Market Conditions:** The model does not account for external factors such as liquidity constraints, transaction costs, or sudden market shifts, which can impact actual option prices.

Conclusion

This project illustrates the use of the Black-Scholes model alongside numerical methods to compute implied volatility, providing traders and financial analysts with a practical tool supported by a robust Python implementation and an intuitive graphical user interface. By integrating both the Newton-Raphson and Bisection methods, the application ensures flexibility and reliability across various scenarios. While it has some limitations, such as reliance on initial guesses and the idealised assumptions of the Black-Scholes model, the tool serves as a valuable resource for both educational and practical purposes in financial markets.

Here is the input:



Implied Volatility	
Spot Price (S)	22957.25
Strike Price (K)	21000
Time to Maturity (T)	0.1667
Risk-free Rate (r)	0.07365
Market Price of Option	2296.70

Compute Implied Volatility

The output:

