Financial Modeling for Option Pricing: Heston Model Implementation

1. Introduction & Objectives

I undertook this project as part of my exploration into quantitative finance, combining stochastic modeling with machine learning techniques for option pricing. The goal was to implement the Heston stochastic volatility model, simulate option prices using Monte Carlo methods, and calibrate the model to market data.

My primary objectives were:
1. Understand and implement the Heston model for option pricing.
2. Use Monte Carlo simulations to generate option price estimates.
3. Calibrate the model to real-world market data and analyze its accuracy.
4. Compare results with expected theoretical values and discuss discrepancies.

2. Theory: Heston Stochastic Volatility Model

The Heston model extends the Black-Scholes framework by incorporating stochastic volatility, making it more realistic for financial markets. The model is defined by the following stochastic differential equations (SDEs):

$$dS_t = \mu S_t dt + \sqrt{V_t} S_t dW_t^S$$

$$dV_t = \kappa (\theta - V_t) dt + \sigma \sqrt{V_t} dW_t^V$$

where:
- S_t is the asset price.
- V_t is the variance (volatility squared).
- μ is the drift.
- κ is the mean-reversion rate.
- θ is the long-run variance.
- σ is the volatility of volatility.
- W_t^S and W_t^V are two correlated Wiener processes with correlation ρ.

The Heston model allows volatility to fluctuate over time, making it more effective for modeling real-world option prices compared to the constant volatility assumption in Black-Scholes.

3. First Approach: Monte Carlo Simulation

Methodology

To generate option prices under the Heston model, I implemented a Monte Carlo simulation using the Euler-Maruyama discretization of the SDEs.

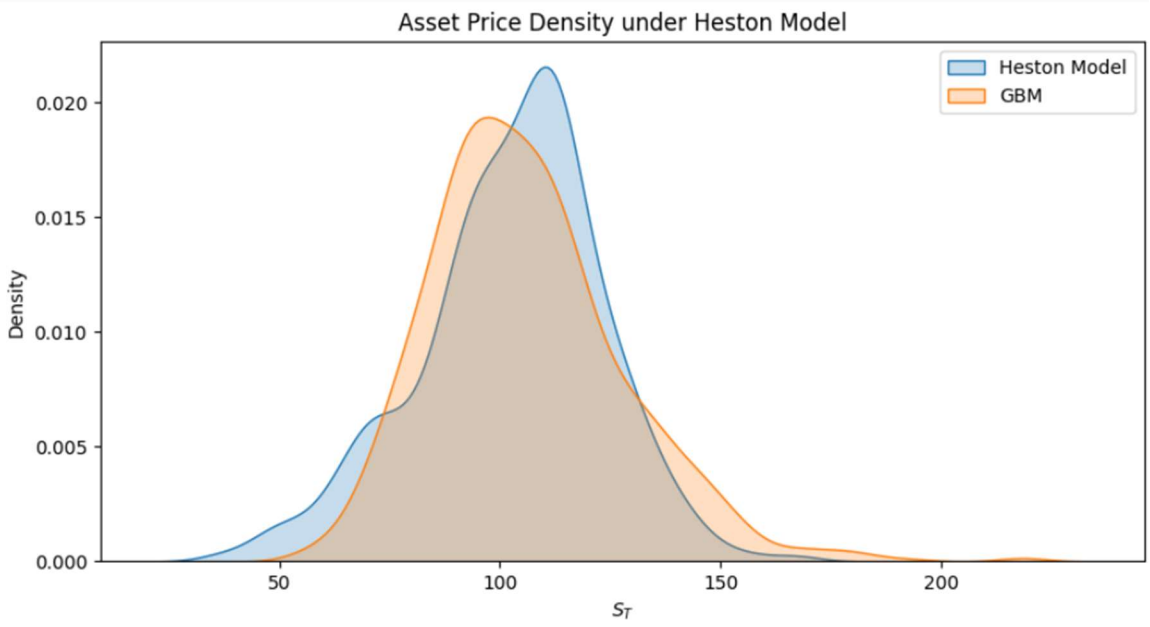
Steps involved:
1. Simulate paths for both stock price (S_t) and volatility (V_t) over discrete time steps.
2. Use correlated Wiener processes (W_t^S and W_t^V) to introduce randomness.
3. Compute the expected option payoff at maturity and discount it back to present value.
4. Average across multiple simulations to estimate the option price.

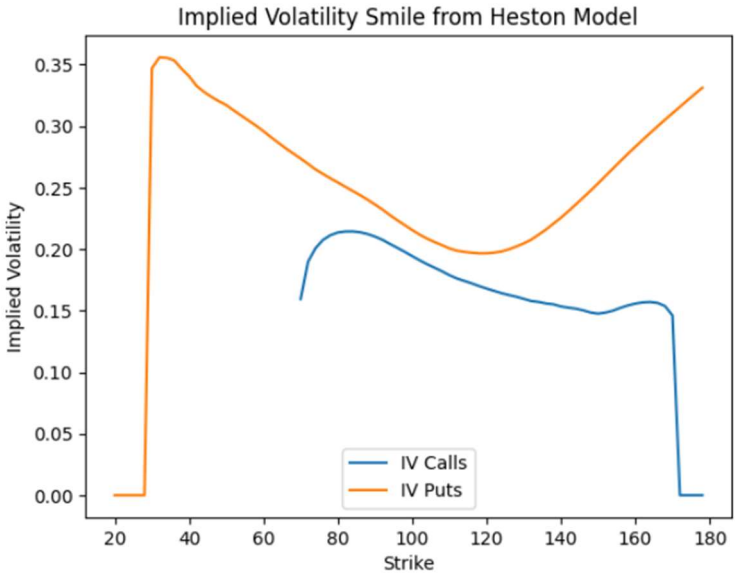
Resources Used

• Python (NumPy, SciPy, Matplotlib)
• Monte Carlo simulation with Euler-Maruyama discretization
• Generated synthetic option price paths

Results Obtained

The Monte Carlo simulation produced option prices, but initial results deviated significantly from expected theoretical values. The main sources of error were:
- High variance in stochastic paths, requiring more simulations for convergence.
- Discretization bias from the Euler-Maruyama scheme.
- Issues with negative variance in the model, sometimes requiring an adjusted scheme (e.g., Milstein or reflection methods).





4. Second Approach: Model Calibration

Methodology

After implementing the Heston model and Monte Carlo simulations, I focused on calibrating the model to real-world data. Calibration involved optimizing model parameters (κ, θ, σ, ρ) to fit observed market prices.

Algorithm & Resources Used

- Optimization algorithms: Least Squares, Differential Evolution
- Market data: Historical option prices
- Programming tools: Python (SciPy, pandas, NumPy)

Steps involved:

1. Defining an objective function to minimize the difference between market and model prices.

2. Using an optimization algorithm (e.g., least squares) to adjust the Heston parameters.

3. Computing a volatility surface to evaluate performance.

Results Obtained

Maturity	Strike	Market Price	Rate	Heston Price
0.0876	140.0	102.125	0.0013	9.7085
0.2026	140.0	103.15	0.0039	12.5477
0.4517	140.0	104.95	0.0085	16.508

These values were not as expected, showing a significant deviation from market prices.

Possible Reasons for Deviation

Upon analyzing my results, I identified several possible reasons for this deviation:
1. Inaccurate parameter estimates: Calibration might not have converged to optimal values.
2. Monte Carlo sampling error: More simulations may be needed to reduce variance.
3. Market factors: Real-world option prices are affected by liquidity, transaction costs, and jumps in prices, which the basic Heston model does not capture.
4. Discretization bias: The Euler-Maruyama method may have introduced numerical errors, suggesting the need for a more accurate discretization scheme.

5. Summary

This project allowed me to gain hands-on experience with stochastic models in finance, implementing the Heston model and exploring Monte Carlo simulations and calibration techniques.

- The Monte Carlo simulation helped approximate option prices, but initial results had high variance.
- Model calibration improved accuracy but still showed deviations from expected values.
- The Euler-Maruyama scheme may not be the best choice, and alternative methods (Milstein scheme, Quadratic Exponential discretization) could improve accuracy.
- Further refinements, such as using machine learning for parameter estimation, could enhance model performance.

Despite these challenges, this project has deepened my understanding of quantitative finance, option pricing, and numerical methods in stochastic processes. I shall work further on refining calibration methods and improving numerical accuracy.