**Option Pricing Heston Model**

Dayne Cordray

May 6th, 2024

**Introduction**

For my final project in quantitative finance and deep learning, I chose to explore option pricing through the development of a Heston model. While the course covered a broad spectrum of topics, including contemporary methods like RNNs and LSTMs, I was drawn to the Heston Model for its rigorous mathematical foundation.

My decision to focus on the Heston Model stemmed from a desire to engage with a methodology that prioritized mathematical rigor over computational power. With previous experience in machine learning from other courses and internships, I sought to challenge myself with a model deeply rooted in mathematical theory.

Initially, my goal was to train the Heston model on American SPY options and leverage it for real-time price predictions. However, I encountered several challenges along the way, leading me to pivot towards European SPX options for training. The primary obstacle was the computational complexity associated with calibrating the model to American options, which involve solving partial differential equations to account for early exercise features—an endeavor that proved computationally prohibitive.

Despite this deviation from my original plan, the project offered a valuable opportunity to deepen my comprehension of option pricing models and the intricacies associated with financial derivatives. This exploration provided insights into both the theoretical foundations of quantitative finance and the practical hurdles encountered during model implementation in real-world scenarios.

**Methodology**

In this project, I employed the Heston model, implemented via QuantLib's **AnalyticHestonEngine**, to price options on the S&P 500 index. This model is chosen for its sophisticated approach to modeling stochastic volatility, an essential feature for capturing the dynamic nature of financial markets.

**Model Setup and Calibration:**

**Mathematical Framework of the Heston Model:**

* **Stock Price Dynamics:** The stock price 𝑆(𝑡) follows a stochastic differential equation (SDE):

𝑑𝑆(𝑡)=𝜇𝑆(𝑡)𝑑𝑡+𝑣(𝑡)𝑆(𝑡)𝑑𝑊𝑠(𝑡)*dS*(*t*)=*μS*(*t*)*dt*+*v*(*t*)​*S*(*t*)*dWs*​(*t*)

Here, 𝜇represents the risk-free rate, 𝑣(𝑡) is the instantaneous variance, and 𝑑𝑊𝑠(𝑡)*dWs*​(*t*) is a Wiener process influencing the stock price.

* **Volatility Dynamics:** The variance 𝑣(𝑡) adheres to a Cox-Ingersoll-Ross (CIR) process:

𝑑𝑣(𝑡)=𝜅(𝜃−𝑣(𝑡))𝑑𝑡+𝜎𝑣(𝑡)𝑑𝑊𝑣(𝑡)*dv*(*t*)=*κ*(*θ*−*v*(*t*))*dt*+*σv*(*t*)​*dWv*​(*t*)

In this model, 𝜅 is the rate of mean reversion to the long-term variance 𝜃, 𝜎 is the volatility of volatility, and 𝑑𝑊𝑣(𝑡)*dWv*​(*t*) is another Wiener process, correlated with 𝑑𝑊𝑠(𝑡)*dWs*​(*t*) via the correlation coefficient 𝜌.

**Implementation Using QuantLib:**

* **Heston Process Setup:** I configured the **HestonProcess** with up-to-date market data and optimized parameters to simulate the stock price and its volatility accurately.
* **Analytic Heston Engine:** This engine leverages the characteristic function derived from the Heston model's equations. By applying Fourier transform techniques, it efficiently computes option prices analytically, offering a substantial advantage in terms of speed and accuracy over numerical methods such as Monte Carlo simulations.

**Parameter Optimization:**

* **Objective Function:** I defined an objective function that measures the squared errors between my model's predicted prices and the actual market prices of options. This least squares method is critical for tuning the model parameters to reflect true market behaviors.
* **Optimization Technique:** Using **scipy.optimize.minimize** with the 'L-BFGS-B' method, I refined the model parameters to ensure the best fit to historical data. This optimization ensures that my model is realistic and robust for practical use in option pricing.

heston\_process = ql.HestonProcess(flat\_ts, dividend\_yield\_handle, spot\_handle, v0, kappa, theta, sigma, rho)  
heston\_model = ql.HestonModel(heston\_process)  
engine = ql.AnalyticHestonEngine(heston\_model)  
option\_ql.setPricingEngine(engine)

**Evaluation:**

To validate the model's effectiveness, I used statistical metrics such as RMSE and correlation coefficients. These metrics assess how well the model predicts actual market prices and gauge the degree of alignment between predicted and observed values. This quantitative evaluation helps ensure that my model is not only theoretically sound but also practically viable.

**Inputs**

**Option Data Acquisition and Preprocessing:**

* **Source:** I used the **yfinance** library to fetch data for S&P 500 index options (ticker symbol: SPX). The data included key details like strike prices, expiration dates, last traded prices, and trading volumes.
* **Spot Price:** The spot price, crucial for all subsequent calculations, represents the closing market price of the index and was retrieved for the latest trading day:

ticker = yf.Ticker(ticker\_symbol)  
spot = ticker.history(period="1d")['Close'].iloc[-1]

* **Volume and Price Filtering:** To ensure data quality and relevance, I filtered out options with low trading volumes and prices below $0.10, which are less likely to provide reliable signals for model calibration:

options\_df = options\_df[options\_df['volume'] > 10]

volSurfaceLong = volSurfaceLong[volSurfaceLong['lastPrice'] > 0.1]

**Handling of Expiration Dates:**

* **Standardization of Dates:** The expiration dates were standardized by converting them to numerical values representing the maturity in years, facilitating their use in mathematical models and curve fitting procedures:

options\_df['maturity'] = (options\_df['expiration'] - pd.Timestamp('now')).dt.days / 365.25

**Risk-Free Rates:**

* **Source and Use:** I scraped risk-free rates from the U.S. Treasury website, covering periods from 1 month to 30 years. These rates are crucial for financial models as they are used to discount future cash flows and for curve fitting to interpolate or extrapolate interest rates for various maturities:

clean\_rates, maturities = get\_risk\_free()

**Curve Fitting:**

* **Nelson-Siegel-Svensson Model:** I used the Nelson-Siegel-Svensson model to fit these rates, providing a continuous yield curve that is essential for deriving risk-free rates applicable to different option maturities:

curve\_fit, status = calibrate\_nss\_ols(clean\_rates, maturities)

**Parameter Estimation for the Heston Model:**

* **Initial Parameters and Bounds:** Before model calibration, initial parameters and their bounds were set based on typical market behaviors and stability considerations:

bounds = [  
 (0.001, 6), # kappa  
 (0.001, 0.15), # theta  
 (0.01, 2), # sigma  
 (-1, 1), # rho  
 (0.001, 0.5) # v0  
]

**Outputs**

**Summary of Results**

This section outlines the results obtained from applying the Heston model to predict S&P 500 index option prices. The model's performance was evaluated through a combination of visualizations and statistical measures to assess how closely the predictions align with actual market prices.

**Visual Comparisons of Predicted and Actual Prices**

**3D Visualization of Option Prices:**

A graph of a graph with lines and numbers

Description automatically generated with medium confidence

An interactive 3D scatter plot provides a comprehensive visual comparison between actual and predicted prices across different strikes and maturities. This graphical representation is instrumental in highlighting the model’s performance and identifying areas where it excels or falls short. The circles in red are predicted prices and the blue are actual prices. We can see options close to expiration around the 5000-strike price are consistently being overpriced while the rest of the predictions seems to be underpriced.

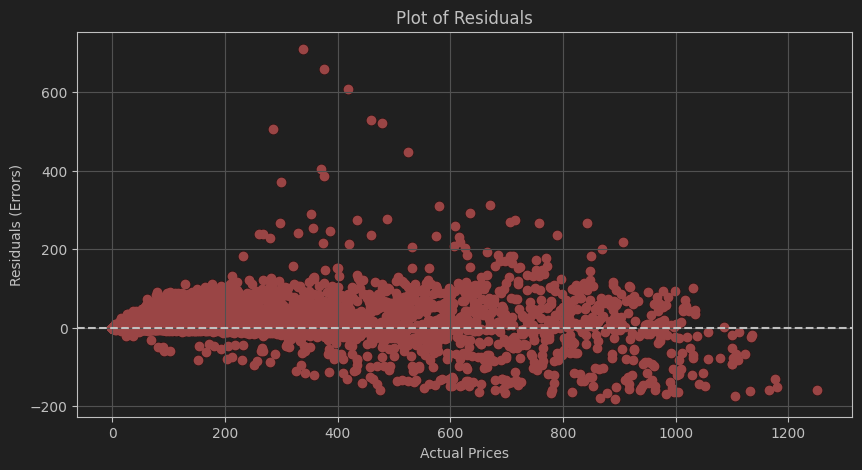
**Statistical Analysis**

**Quantitative Metrics:** I quantified the model's accuracy using several key metrics:

* **Mean Absolute Error (MAE):** The MAE of 26.51 indicates that, on average, the model's predictions deviate from the actual prices by approximately 26.51 points. This metric helps gauge the average magnitude of errors without considering their direction, providing a straightforward measure of prediction error.
* **Mean Squared Error (MSE):** With an MSE of 2,715.98, this metric underscores the average squared deviation between predicted and actual values, highlighting the impact of significant outliers on the model’s performance.
* **Root Mean Squared Error (RMSE):** The RMSE of 52.12 is particularly telling as it provides the error magnitude in the same units as the prices. A lower RMSE is preferable and indicates higher accuracy; in this context, an RMSE of 52.12 can be seen as relatively high, suggesting room for model improvement.
* **Correlation Coefficient:** A correlation coefficient of 0.981 is highly indicative of a strong linear relationship between the predicted and actual prices, affirming that the model predictions generally follow the true market trends closely.

| **Metric** | **Value** |
| --- | --- |
| Mean Absolute Error | 26.51 |
| Mean Squared Error | 2,715.98 |
| Root Mean Squared Error | 52.12 |
| Correlation Coefficient | 0.981 |

**Residual Analysis**

****

The residuals plot below shows the differences between actual and predicted prices. Ideally, residuals should be randomly dispersed around the horizontal axis (zero line), which would indicate that the model's errors are random and not biased.

**Analysis:**

* The majority of residuals are clustered near the zero line, which suggests that the model is accurate for most predictions.
* The presence of several outliers, particularly those far above and below the main cluster, points to instances where the model significantly deviates from actual prices. These outliers could indicate specific market conditions or option characteristics that are not adequately captured by the current model parameters.

**Conclusion**

A screen shot of a computer screen

Description automatically generated

The comprehensive evaluation of the Heston model using statistical metrics and visual analysis highlights its strengths in capturing the overall trends of the options market. However, the analysis also reveals areas where the model may require further refinement, such as improving its handling of outliers and extreme market conditions. The model is able to capture some features of the market very well (pictured above) but is overall lacking in my opinion and would not be something that could be used to drive decisions yet. Future work could focus on adjusting model parameters, incorporating additional market factors, or exploring more complex stochastic models to enhance predictive accuracy.