GEORGIA INSTITUTE OF TECHNOLOGY

ISYE 6767 PROJECT 1

Delta Hedge Method for Options

Student Name:
Chongrui Wang

Student ID:
xxx

October 16, 2023

Contents

1 Interim Project: Implementing a Dynamic Delta Hedging Strategy

1.1 Background

Delta-hedging is a strategy used to replicate the value of a financial derivative (like a Call option) written on a traded asset. This is done by dynamically buying (or selling) an appropriate number of shares of the underlying asset and borrowing from (or lending to) a bank.

1.2 Description of the Delta-Hedging Process

1. The hedging period is from a start-date t_0 to an end-date t_N. Initially, there is a cash position of \$0.
2. At t_0, sell a European call option contract with an expiration date T and strike price K. The contract is for one share of stock and $t_0 < t_N < T$.
3. To hedge the short position in the European call, buy δ shares of the underlying stock at t_0, where δ is the rate of change of option value V with respect to changes in the underlying price S.
4. δ changes during the hedging period, so re-balance the portfolio daily to maintain a long position of δ_i shares of stock for each date t_i where $i = 1, 2, ...N$. δ_i should be calculated using implied volatility for each date.
5. For each date t_i, where $i = 1, 2, ...N$, calculate the cumulative hedging error till t_i.

1.3 Task

- Write a program that reads daily stock prices, risk-free rates, and option prices data for 2011, and computes the daily hedging error and PNL.
- Output a file `result.csv` containing stock price, option price, implied volatility, delta, hedging error, PNL, and PNL with hedge.

1.4 Data Provided

- `interest.csv`: Contains daily risk-free rates in 2011.
- `secGOOG.csv`: Contains adjusted closing stock prices in 2011 (assumption: no dividend).
- `opGOOG.csv`: Contains option prices data in 2011.

Note: The use of OOP (Object-Oriented Programming) or generic programming style is required for the project.

1.5 Mind Map for Solving the Problem

1. **Understanding the Problem**: Define key terms like delta, hedging error, and PNL.
2. **Data Parsing**: Read data from provided CSV files and ensure it's in a format that can be utilized.
3. **Model Development**:
   - **Delta Calculation**: Implement formula to compute delta daily.
   - **Hedging Error & PNL Calculation**: Develop a mechanism to compute the daily hedging error and PNL.
4. **Output Generation**: Create a system to export results to `result.csv`.
5. **Object-Oriented Design**: Build classes to represent key entities like Stock, Option, etc., to ensure an OOP approach.
6. **Unit Testing**: Create tests to ensure each component is working correctly.
7. **Report Writing**: Document the process, findings, and conclusions.

2 Report on Header Files

2.1 BSModel.h (Black-Scholes Model)

This header file is dedicated to the functionalities related to the Black-Scholes model.

Key Components:

- A constant for the mathematical value π.
- Functions for:
  - Computing the cumulative distribution and probability density functions of a standard normal distribution.
  - Calculating the Black-Scholes option price for a European call option.
  - Determining the Delta and Vega of an option using the Black-Scholes model.
  - Estimating implied volatility using a binary search method.

2.2 DataRow.h (Data Row)

This header file defines structures and functions to manage and access data rows related to option contracts.

Key Components:

- `DataRow` structure:
  - Fields for date, expiry date, call/put flag, and strike price of option contracts.
  - Constructor for initializing the structure.
- Functions for:
  - Constructing a vector of `DataRow` objects.
  - Binary search on a vector of `DataRow` objects.

2.3 Utils.h (Utilities)

This header file provides utility functions essential for various tasks in the project.

Key Components:

- Functions for:
  - Binary search on a vector of strings.
  - Writing multiple vectors of data into a CSV file, including vectors for dates, option prices, values, implied volatilities, option Deltas, hedging errors, and PNL values with and without hedging.

2.4 test.h (Test Suite and Implementations)

This file contains definitions and implementations related to simulating stock price paths, delta hedging, and the Black-Scholes model.

- `Question1(...)`: Simulates stock price paths and implements delta hedging for each path.
- `Question2(...)`: Performs computations using the Black-Scholes model and reads data from CSV files.
- `tests()`: Serves as the test suite, running the 'Question1(...)' and 'Question2(...)' functions with predefined parameters.
- `test_BSModel()`: Test whether our calculation for the option price, delta, and implied volatility is accurate based on our implementation to BS Model.

Summary: The header files encapsulate the foundational structures and functions required for the project. `BSModel.h` focuses on the mathematical and financial aspects of option pricing using the Black-Scholes model. `DataRow.h` emphasizes data management, ensuring that option contract data is stored and accessed efficiently. Lastly, `Utils.h` contains auxiliary functions, crucial for tasks such as searching within data vectors and exporting results to a CSV file.

Together, these files lay the groundwork for implementing the dynamic delta hedging strategy, providing the necessary tools and abstractions to handle data and perform key computations.

3 Report on Implementation Files

3.1 BSModel.cpp (Black-Scholes Model Implementation)

This file provides the implementation details for the functions related to the Black-Scholes model.

- `norm_cdf(double x)`: Calculates the cumulative distribution function of a standard normal distribution.
- `norm_pdf(const double x)`: Computes the probability density function of a standard normal distribution.
- `BlackScholesCall(...)`: Implements the Black-Scholes formula for a European call option.
- `DeltaBS(...)`: Computes the Delta of an option.
- `vega(...)`: Calculates the Vega of an option.
- `implied_volatility_binary_search(...)`: Estimates the implied volatility using a binary search.

3.2 DataRow.cpp (Data Row Implementation)

This file offers implementation details for functions and structures related to managing data rows.

- `DataRow`: Constructor initializes the 'DataRow' object.
- `constructDataRows(...)`: Constructs a vector of 'DataRow' objects.
- `binary_search_row(...)`: Performs a binary search on a vector of 'DataRow' objects.

3.3 main.py (Python Visualization)

The script focuses on visualization tasks using Python libraries like pandas, matplotlib, and seaborn.

- `plotQ1paths()`: Reads data from 'Q1path.csv' and visualizes multiple stock price paths.
- `plotQ1HE()`: Reads data from 'Q1HE.csv' and visualizes the distribution of hedging errors using a histogram.

4 Implementation Guide for the Delta Hedging Strategy

4.1 Data Parsing and Representation

Files Involved: `DataRow.h`, `DataRow.cpp`

- **Purpose**: Store and manage data rows related to option contracts.
- **Implementation**:
  - The `DataRow` structure is used to encapsulate information about option contracts.
  - Functions like `constructDataRows(...)` and `binary_search_row(...)` help manage and search within these data rows.

4.2 Black-Scholes Model Computations

Files Involved: `BSModel.h`, `BSModel.cpp`

- **Purpose**: Compute option prices, deltas, and implied volatilities.
- **Implementation**:
  - Functions like `BlackScholesCall(...)`, `DeltaBS(...)`, and `implied_volatility_binary_search(...)` are used for respective computations.

4.3 Utility Functions

Files Involved: `utils.h`, `utils.cpp`

- **Purpose**: Provide auxiliary functionalities.
- **Implementation**:
  - Use the `binary_search(...)` function for efficient searching.
  - The `writeCSV(...)` function exports results to CSV files.

4.4 Main Logic and Testing

Files Involved: `test.h`, `main.cpp`

- **Purpose**: Test functionalities, simulate stock price paths, implement delta hedging, and export results.

- **Implementation**:
  - The `Question1(...)` and `Question2(...)` functions handle the main computations.
  - `tests()` serves as a testing suite.
  - `main.cpp` initiates testing and manages program flow.

4.5 Visualization

File Involved: `main.py`

- **Purpose**: Provide graphical representation of results.
- **Implementation**:
  - The `plotQ1paths()` and `plotQ1HE()` functions visualize stock price paths and hedging errors, respectively.

4.6 Steps to Accomplish the Task

1. **Data Collection**: Ensure all required data files are available.
2. **Compile and Run**: Compile the C++ program and run the executable to generate CSV results.
3. **Visualization**: Execute the `main.py` script for graphical representations.
4. **Analysis**: Inspect generated CSV files for results and insights.

5 BS Model Testing

This report showcases the testing results for our implementation of the Black-Scholes Model. The target values were obtained from an online Black-Scholes Model calculator, and they are compared with the values we obtained from our implemented model.

The test result is shown in Figure 1.

5.1 Discussion

The results obtained from our implemented Black-Scholes Model are in close agreement with the target values from the online calculator. Small discrepancies may be attributed to rounding differences or potential nuances in calculation methods.

5.2 Conclusion

Our implementation of the Black-Scholes Model produces accurate results in comparison to the online calculator. This consistency provides confidence in the reliability of our model for option pricing.

6 Output Analysis for Question 1

6.1 Output Breakdown

The sample implementation for solving problem 1 can be found at Figure 2.

1. **Program Initialization**:

   The program starts and displays a welcome message, signaling that it's ready for user input.

2. **Question Selection**:

   The user is prompted to select which question or functionality they'd like to run. An invalid choice is made initially, and the program notifies the user of the error.

3. **Question 1 Selection**:

   The user selects the first question, indicating their intention to proceed with the associated functionality.

4. **Parameter Input for Question 1**:

   The user is prompted to input various parameters essential for computations. After an initial invalid value, the user provides the necessary values for Spot Price, Strike Price, Time to Maturity, Interest Rate, Mu, and Sigma.

5. **Program Exit**:

   Post parameter input, the user is prompted again to select a question. Instead, the user opts to exit the program, which then gracefully concludes its execution.

6.2 Analysis

- The program offers an *interactive user interface*, allowing users to choose between distinct functionalities.
- *Error handling mechanisms* are evident, managing incorrect inputs effectively.
- For the first question, several parameters related to the Black-Scholes model are collected from the user.
- The provided interaction doesn't extend to the results or outputs of the first question.

6.3 Stock Price Paths

The graph of price path for problem 1 can be found at Figure 4. I have also tried another group of parameters to implement this function. The result is shown in Figure 5.

The another group of parameters I have tried for problem 1 is:

Table 1: Alternative Parameter values used for Question 1

Parameter	**Value**
Spot Price	100
Strike Price	110
Time to Maturity	0.2
Interest Rate	0.05
μ (mu)	0.1
σ (sigma)	0.3

Observations:

- *Initial Stock Price*: All the paths seem to start from a similar stock price around the 100 mark.
- *Variability*: The graph illustrates a high degree of variability in the stock price paths, with some paths reaching values as high as approximately 140 and some dipping to around 80.
- *General Trend*: While individual paths exhibit a lot of fluctuations, there doesn't appear to be a consistent upward or downward trend for the overall collection of price paths over the observed time horizon.
- *Convergence at the End*: Towards the 100th time step, many paths seem to converge to a narrower range, indicating reduced variability or uncertainty as time progresses.

6.4 Distribution of Hedging Errors

The graph of the distribution of hedging error for problem 1 can be found at Figure 6. I have also tried another group of parameters to implement this function. The result is shown in Figure 7.

Observations:

- *Normal Distribution*: The distribution of hedging errors appears to be approximately normally distributed around zero.
- *Centered Around Zero*: The peak of the distribution is centered around a hedging error value of zero, indicating that a significant number of hedging predictions were accurate or had minimal errors.

- *Spread*: Most of the hedging errors lie between -1 and 1, suggesting that the strategy was generally effective. However, there are instances where the hedging error reached values as low as -3 and as high as 2.
- *Tail Behavior*: The tails of the distribution, especially towards the negative side, show that there were a few instances where the hedging strategy was off by a significant margin.

6.5 Conclusion

The stock price simulation showcases the inherent unpredictability and volatility of stock prices over time, emphasizing the importance of robust hedging strategies. The hedging error distribution suggests that the employed hedging strategy was effective for a majority of scenarios, as evidenced by the errors clustering around zero. However, there were cases where significant discrepancies were observed, highlighting areas for potential improvement in the hedging strategy. Regular analysis and adjustments based on observed hedging errors can enhance the overall effectiveness of the hedging process.

7 Simulated Option Price Paths

Options are financial derivatives that give the buyer the right, but not the obligation, to buy or sell an asset at a specified price on or before a certain date. The plot analyzed in this document shows multiple simulated paths for option prices over time, calculated using a stochastic model.

The option price paths were simulated using the Monte Carlo method, which is commonly used for estimating the future value of options. The specific details of the model parameters, such as volatility, drift, and the risk-free rate, are essential for replicating this analysis.

As seen in Figure 8, the option price paths exhibit significant variation as time progresses. This is reflective of the inherent uncertainty in the pricing of options. Some paths show a steep increase in price, suggesting favorable market conditions, while others indicate a decline, representing less favorable conditions.

7.1 Statistical Analysis

A comprehensive statistical analysis would include the calculation of the mean, median, and standard deviation of the option prices at each time step. This would provide insight into the central tendency and dispersion of the prices over time.

$$Mean : \mu(t) = \frac{1}{N} \sum_{i=1}^{N} P_i(t)$$

$$Median : med(t) = Median\{P_1(t), P_2(t), \ldots, P_N(t)\}$$

$$StandardDeviation : \sigma(t) = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(P_i(t) - \mu(t))^2}$$

Where $P_i(t)$ is the price of the i-th simulated path at time t, and N is the total number of simulations.

7.2 Conclusion

The analysis of the simulated option price paths provides valuable insights into the potential future behavior of option prices. Understanding the distribution of these paths can help traders and investors make informed decisions regarding option trading strategies.

8 Output Analysis for Question 2

8.1 Output Breakdown

The sample implementation for solving problem 2 can be found at Figure 3.

1. **Program Initialization**:

   The program initiates and signals readiness for user interaction.

2. **Question Selection**:

   The user opts for the second functionality or question after being prompted.

3. **Parameter Input for Question 2**:

   The user is requested to enter various parameters, specifically dates and strike price. The program ensures the date format is adhered to, providing feedback for incorrect inputs.

4. **Program Exit**:

   Post input collection, the user is prompted to select a functionality again. Instead, they decide to terminate the program, which then concludes in an organized manner.

8.2 Analysis

- The program maintains an *interactive user interface*, guiding users through different functionalities.

- The system effectively manages *invalid inputs*, particularly regarding date format, offering clear corrective feedback.
- For the second question, parameters related to dates and strike price are gathered, essential for specific computations or analyses.
- The user interaction flow concludes gracefully, indicating well-structured design.

8.3 Result of Q2

This report provides an analysis of the hedging performance for a stock based on two datasets from different months. Specifically, we aim to understand the difference in the Profit and Loss (PNL) without hedging compared to the PNL with hedging.

The graph of the results for problem 2 can be found at Figure 9. I have also tried another group of parameters to implement this function. The result is shown in Figure 10.

The another group of parameters I have tried for problem 2 is:

Table 2: Alternative Parameter values used for Question 2

Parameter	**Value**
t_0 Date	2011-03-01
tn Date	2011-03-31
T Date	2011-06-18
Strike Price	500

8.4 Observations

Upon examining the data tables, we make the following observations:

1. The PNL with hedging tends to deviate further from zero in comparison to the normal PNL in one of the datasets.
2. The implied delta and hedging values vary over the period, which can influence the PNL values.

8.5 Analysis

The deviation of the PNL with hedging from zero in one dataset suggests a potential anomaly or a significant shift in market conditions. This deviation indicates the inefficacy of the hedging strategy during that particular period.

A potential reason for this observation could be that the returns for this stock do not adhere to the assumption of log-normal returns. This deviation from log-normal returns could lead to inaccurate option pricing and consequently result in imperfect hedging.

8.6 Conclusion

The discrepancies in the PNL with hedging from the normal PNL underscore the importance of regularly reviewing and adjusting hedging strategies. It is also crucial to ensure that the underlying assumptions, such as log-normal returns, hold for the stock in question. Further research and analysis are recommended to validate the cause of this deviation and develop more effective hedging techniques.

9 Closing Remark

In finance, as with many real-world applications, assumptions form the cornerstone of many models and strategies. The observed discrepancies in our hedging performance highlight the need for continuous validation of these assumptions and the importance of adaptability. While models provide a structured approach towards understanding markets and guiding decision-making, they are not infallible. The deviation observed in the PNL with hedging serves as a poignant reminder that the financial landscape is dynamic and ever-evolving. It emphasizes the importance of a regular review of our strategies and the need to be agile in our response to market intricacies. As we move forward, a holistic approach, combining both quantitative models and qualitative insights, will be instrumental in navigating the complexities of financial markets.

10 Appendix

```
Starting BS Model Testing ...
The Price of this Call Option by BS Model should be 13.63, get 13.6396
The Delta of this Call Option by BS Model should be 0.6265, get 0.626454
The Implied Volatility of this Call Option by BS Model should be 0.2, get 0.2
The Price of this Call Option by BS Model should be 2.696, get 2.69618
The Delta of this Call Option by BS Model should be 0.48, get 0.480302
The Implied Volatility of this Call Option by BS Model should be 0.1, get 0.1
Test Finished!
```

Figure 1: The Test of BS Model

```
Starting Project ...
Select the question you want to run (1/2) or type EOF to exit: 2
Enter t0 date (yyyy-mm-dd): 123
Invalid date format. Please enter in 'yyyy-mm-dd' format.
Enter t0 date (yyyy-mm-dd): wcr
Invalid date format. Please enter in 'yyyy-mm-dd' format.
Enter t0 date (yyyy-mm-dd): 2011/07/05
Invalid date format. Please enter in 'yyyy-mm-dd' format.
Enter t0 date (yyyy-mm-dd): 2011-07-05
Enter tn date (yyyy-mm-dd): 2011-07-29
Enter T date (yyyy-mm-dd): 2011-09-17
Enter Strike Price value: 2011-07-05
Failed to open the CSV file!Select the question you want to run (1/2) or type EOF to exit: Invalid choice. Please select 1 or 2.
Select the question you want to run (1/2) or type EOF to exit: ^Z
Exiting...
Finished!
```

Figure 2: The Sample Implementation of Q1

```
Starting Project ...
Select the question you want to run (1/2) or type EOF to exit: 2
Enter t0 date (yyyy-mm-dd): 123
Invalid date format. Please enter in 'yyyy-mm-dd' format.
Enter t0 date (yyyy-mm-dd): wcr
Invalid date format. Please enter in 'yyyy-mm-dd' format.
Enter t0 date (yyyy-mm-dd): 2011/07/05
Invalid date format. Please enter in 'yyyy-mm-dd' format.
Enter t0 date (yyyy-mm-dd): 2011-07-05
Enter tn date (yyyy-mm-dd): 2011-07-29
Enter T date (yyyy-mm-dd): 2011-09-17
Enter Strike Price value: 500
Select the question you want to run (1/2) or type EOF to exit: ^Z
Exiting...
Finished!
```

Figure 3: The Sample Implementation of Q2

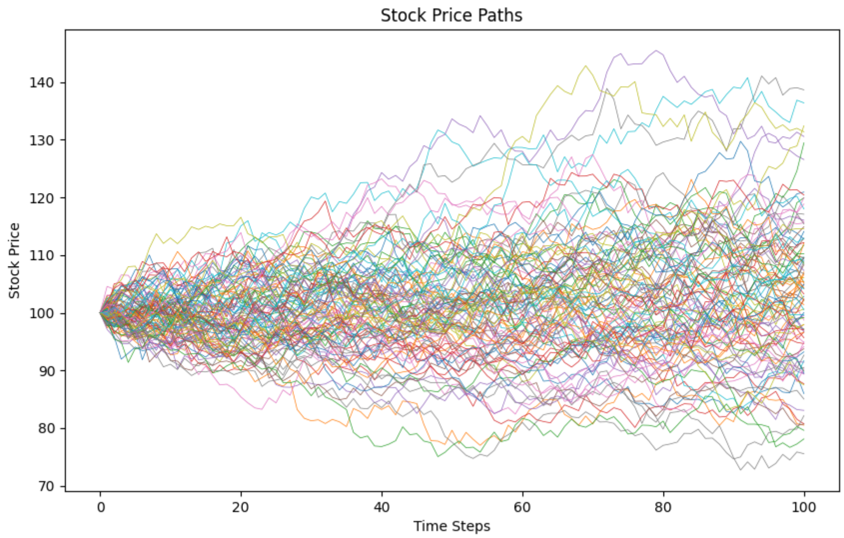

Figure 4: The Price Path for Q1

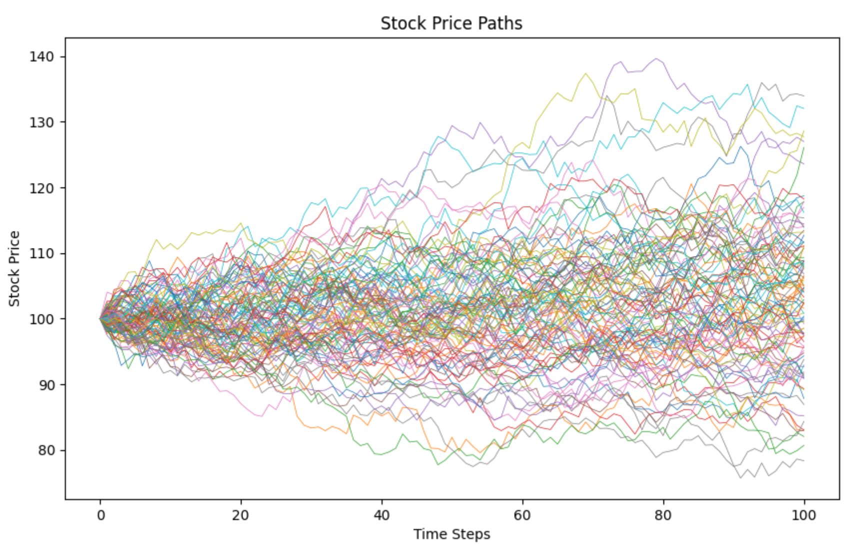

Figure 5: The Price Path with Another Group of Parameters

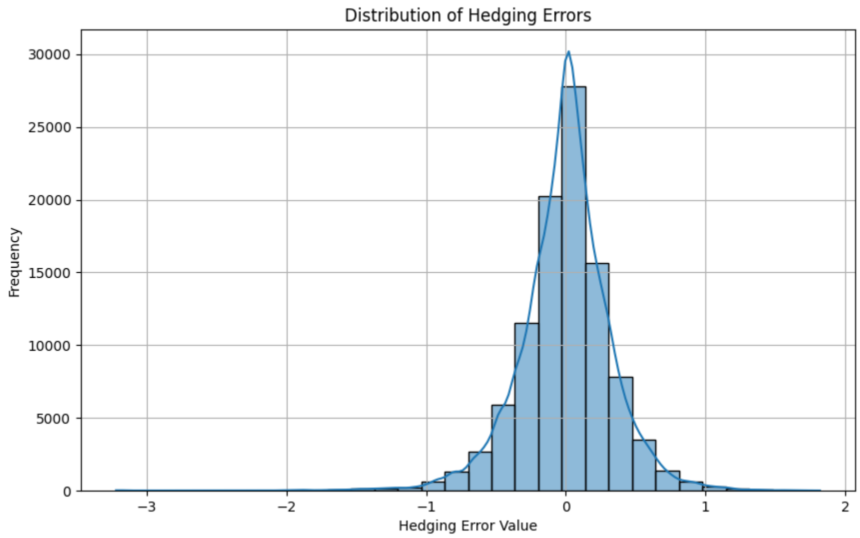

Figure 6: The Distribution of Hedging Error for Q1

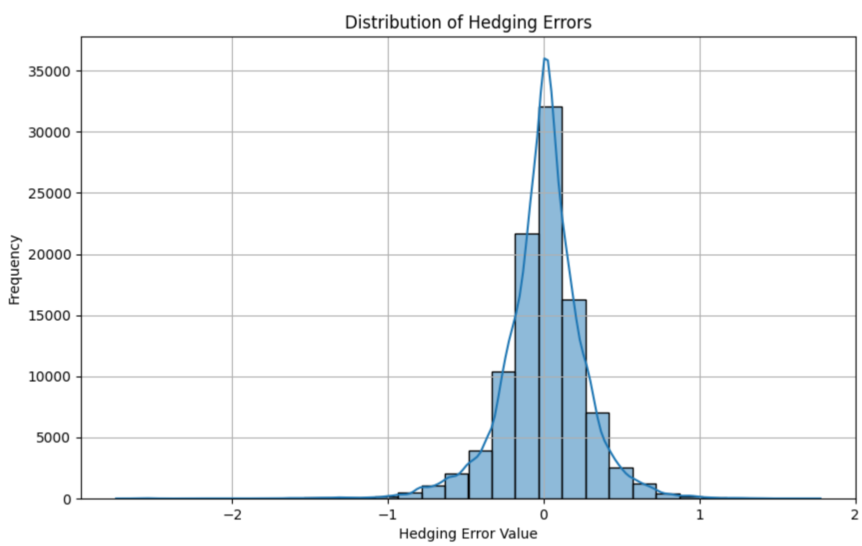

Figure 7: The Distribution of Hedging Error with Another Group of Parameters

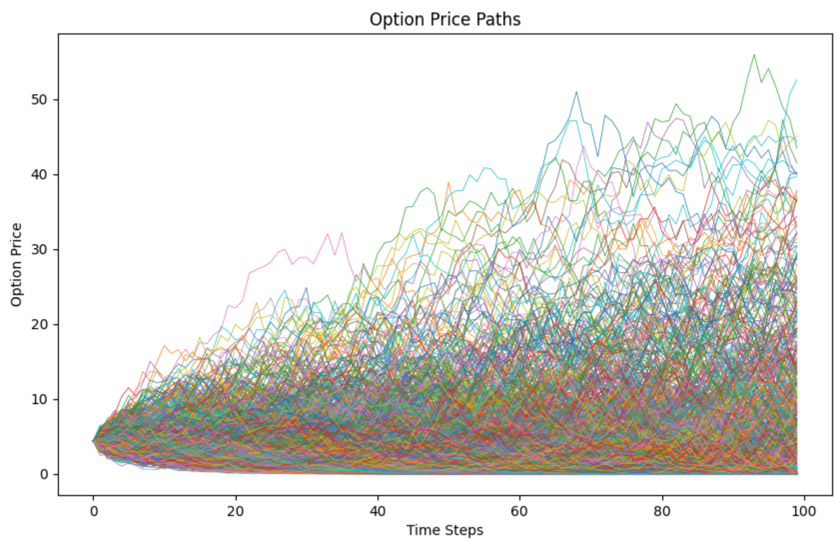

Figure 8: The Option Price Path

date	S	V	Implied v	delta	hedging e	PNL	PNL (with hedge)
2011/7/5	532.44	44.2	0.259686	0.722654	0	0	0
2011/7/6	535.36	46.9	0.269337	0.733318	-0.591473	-2.7	-0.591473
2011/7/7	546.6	55.3	0.269861	0.787576	-0.750633	-11.1	-1.34211
2011/7/8	531.99	43.95	0.268563	0.719396	-0.908904	0.25	-2.25101
2011/7/11	527.28	41	0.27595	0.691509	-1.34887	3.2	-3.59988
2011/7/12	534.01	46.4	0.286788	0.722767	-2.09656	-2.2	-5.69645
2011/7/13	538.26	49.3	0.287137	0.745056	-1.92643	-5.1	-7.62288
2011/7/14	528.94	41.15	0.272178	0.706903	-0.722036	3.05	-8.34491
2011/7/15	597.62	99.65	0.28149	0.940744	-10.6735	-55.45	-19.0184
2011/7/18	594.94	97.65	0.300413	0.926444	-11.197	-53.45	-30.2154
2011/7/19	602.55	103.8	0.266228	0.96029	-10.299	-59.6	-40.5144
2011/7/20	595.35	97.8	0.299707	0.931925	-11.2154	-53.6	-51.7297
2011/7/21	606.99	108.15	0.275666	0.964257	-10.72	-63.95	-62.4497
2011/7/22	618.23	118.7	0.24842	0.985999	-10.434	-74.5	-72.8837
2011/7/25	618.98	119.95	0.294243	0.971588	-10.9469	-75.75	-83.8307
2011/7/26	622.52	123.25	0.28697	0.978582	-10.8099	-79.05	-94.6405
2011/7/27	607.22	108.65	0.304805	0.957691	-11.1845	-64.45	-105.825
2011/7/28	610.94	112.1	0.302652	0.964977	-11.0742	-67.9	-116.899
2011/7/29	603.69	106.8	0.370045	0.924709	-12.7726	-62.6	-129.672

Figure 9: The Result of Q2

date	S	V	Implied v	delta	hedging e	PNL	PNL (with hedge)
2011/3/1	600.76	106.65	0.291056	0.889954	0	0	0
2011/3/2	600.79	107.45	0.304978	0.881965	-0.775339	-0.8	-0.775339
2011/3/3	609.56	115.55	0.311335	0.895536	-1.14252	-8.9	-1.91786
2011/3/4	600.62	108.15	0.322066	0.873319	-1.75067	-1.5	-3.66853
2011/3/7	591.66	99.4	0.310346	0.863367	-0.827604	7.25	-4.49614
2011/3/8	592.31	99.5	0.305592	0.86937	-0.368379	7.15	-4.86452
2011/3/9	591.77	98.1	0.293568	0.877792	0.560181	8.55	-4.30434
2011/3/10	580.3	89.55	0.314532	0.837815	-0.960092	17.1	-5.26443
2011/3/11	576.71	86.05	0.310318	0.832536	-0.469741	20.6	-5.73417
2011/3/14	569.99	80.55	0.313308	0.813266	-0.566263	26.1	-6.30043
2011/3/15	569.56	80.3	0.316806	0.811329	-0.667793	26.35	-6.96822
2011/3/16	557.1	72.65	0.34487	0.759302	-3.12877	34	-10.097
2011/3/17	561.36	74.15	0.326439	0.783722	-1.39583	32.5	-11.4928
2011/3/18	561.06	72.5	0.311886	0.792569	0.0173046	34.15	-11.4755
2011/3/21	576.5	84.3	0.302566	0.846335	0.452793	22.35	-11.0227
2011/3/22	577.32	84.6	0.299127	0.852552	0.844868	22.05	-10.1779
2011/3/23	582.16	88.15	0.291861	0.871881	1.41928	18.5	-8.75858
2011/3/24	586.89	91.9	0.286723	0.888699	1.79129	14.75	-6.96729
2011/3/25	579.74	86.1	0.2968	0.865865	1.23505	20.55	-5.73224
2011/3/28	575.36	82.25	0.298109	0.854961	1.29059	24.4	-4.44165
2011/3/29	581.73	87	0.287341	0.881263	1.98475	19.65	-2.45691
2011/3/30	581.84	87.75	0.301874	0.873079	1.32967	18.9	-1.12724
2011/3/31	586.76	91.8	0.299475	0.888445	1.57322	14.85	0.445982

Figure 10: The Result of Q2 with Another Group of Parameters