Module Name:

Financial Portfolio Analysis

Candidate Number:

277173

ABSTRACT

This report presents an in-depth financial risk analysis of five high-risk assets: Tesla Inc. (TSLA), Coinbase Global Inc. (COIN), Amazon.com Inc. (AMZN), iShares MSCI Emerging Markets ETF (EEM), and Moderna Inc. (MRNA). Utilizing advanced statistical techniques and financial models, the study aims to identify, measure, and mitigate the risks associated with these investments. Key analyses include the computation of daily returns, share prices, and the application of the Altman Z-score, Monte Carlo simulations, and credit risk models. Furthermore, the report explores the construction of the efficient frontier and optimal portfolio selection using utility functions. The findings highlight the significant volatility and potential returns of the selected assets, providing valuable insights for investors and financial analysts in managing portfolio risks and optimizing investment strategies.

INTRODUCTION

Financial risk analysis is a critical discipline within the broader scope of financial portfolio management, focusing on identifying, measuring, and mitigating risks associated with investments. In an increasingly complex financial landscape, investors and financial analysts must navigate through various types of risks, including market risk, credit risk, and operational risk. This report delves into the intricacies of financial risk analysis, employing advanced statistical techniques and financial models to evaluate and manage these risks effectively.

The primary objective of this report is to provide a comprehensive analysis of financial risk by examining a selection of five high-risk assets. These assets, chosen based on their market volatility, sector diversity, and growth potential, include Tesla Inc. (TSLA), Coinbase Global Inc. (COIN), Amazon.com Inc. (AMZN), iShares MSCI Emerging Markets ETF (EEM), and Moderna Inc. (MRNA). By analysing the daily returns, share prices, and statistical measures of these assets, this report aims to offer insights into their risk profiles and the broader implications for portfolio management.

This report also includes a thorough exploration of key financial concepts such as the Altman Z-score, Monte Carlo simulations, and credit risk analysis, alongside practical applications of these models. The analysis is further enriched by the computation of variance-covariance matrices, linear regression against market indices, and the construction of efficient frontiers. These methodologies are employed to assess the risk-return profiles of the selected assets and optimize portfolio performance.

MAIN BODY

What is “Risky asset”?

Risky assets are investments that offer a potential for higher returns, but come with a higher level of uncertainty and a greater chance of loss compared to more stable investments. This risk is typically associated with the variability or volatility of returns, meaning their prices can fluctuate significantly over short periods.

Some characteristics for a risky asset are:

1. High volatility
2. Potential for high return
3. Market sensitivity
4. Liquidity Concerns
5. Uncertainty and unpredictability.

Examples:

1. Equities
2. Cryptocurrencies, Etc.

Choosing five risky assets with proper justification:

For my dissertation, I will analyze the following five risky assets, considering factors like market volatility, sector diversity, growth potential, and market dynamics:

1. **Tesla, Inc. (TSLA) [10-5-2022 to 10-5-2023]**

Justification: Tesla's high volatility is driven by its innovative electric vehicle industry presence and CEO Elon Musk's public profile. The growth potential in renewable energy and automotive sectors makes TSLA ideal for studying fluctuations and risk.

2. **Coinbase Global, Inc. (COIN) [10-5-2022 to 10-5-2023]**

Justification: Coinbase is a leading cryptocurrency exchange, subject to significant volatility and regulatory uncertainties. The fluctuating value of cryptocurrencies and evolving regulatory landscape pose substantial risks, impacting operations and profitability.

3. **Amazon.com, Inc. (AMZN) [10-5-2022 to 10-5-2023]**

Justification: Amazon's diverse operations in e-commerce, cloud computing, and AI expose it to various market risks. Despite its size, AMZN's significant price movements provide rich data for risk-return analysis.

4. **iShares MSCI Emerging Markets ETF (EEM) [10-5-2022 to 10-5-2023]**

Justification: EEM offers exposure to large and mid-sized companies in emerging markets, which are generally more volatile. Analysing EEM allows for an examination of geopolitical risks, currency fluctuations, and different economic dynamics.

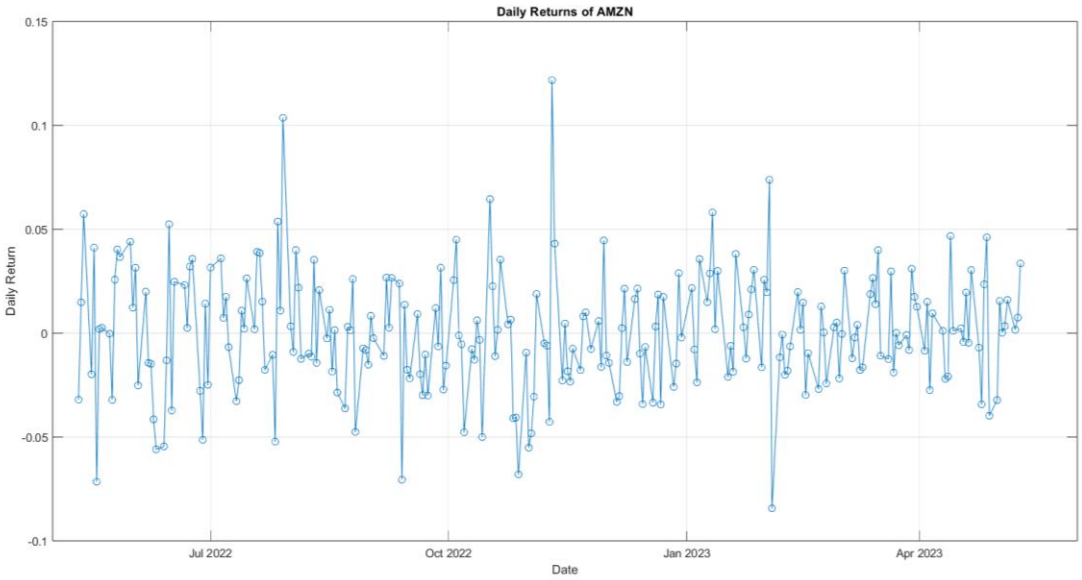
5. **Moderna, Inc. (MRNA) [10-5-2022 to 10-5-2023]**

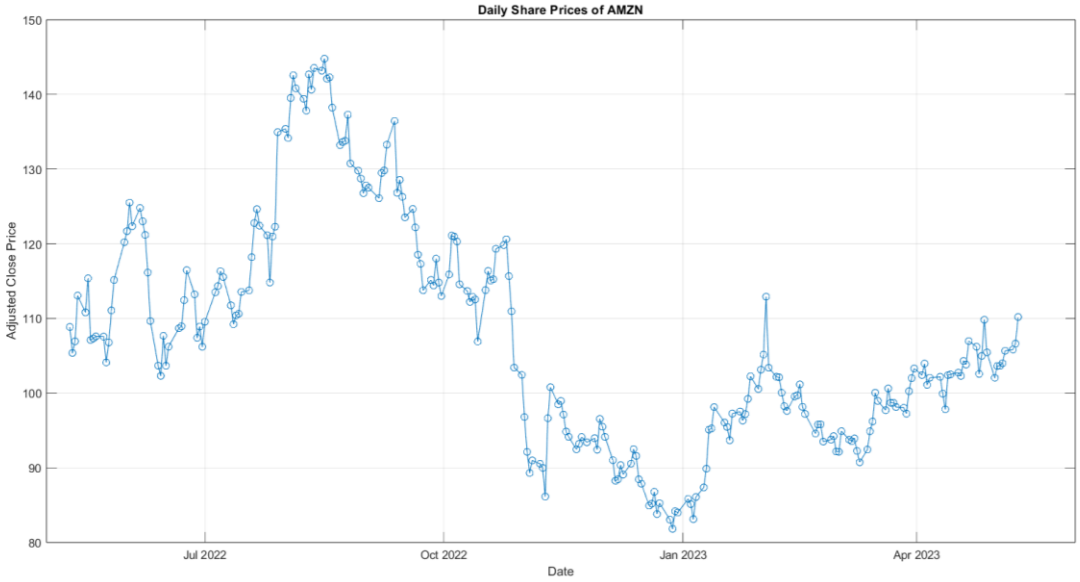
Justification: As a biotech company, Moderna's stock is highly sensitive to drug development news, regulatory approvals, and market adoption of products like its COVID-19 vaccine. The biotech sector's inherent risks and high growth potential make MRNA ideal for risk analysis.

ANALYSIS OF THE WHOLE DATA AND MAKING A PORTFOLIO:

PLOTTING DAILY RETURNS AND DAILY SHARE PRICES FOR EACH INDIVIDUAL ASSETS:

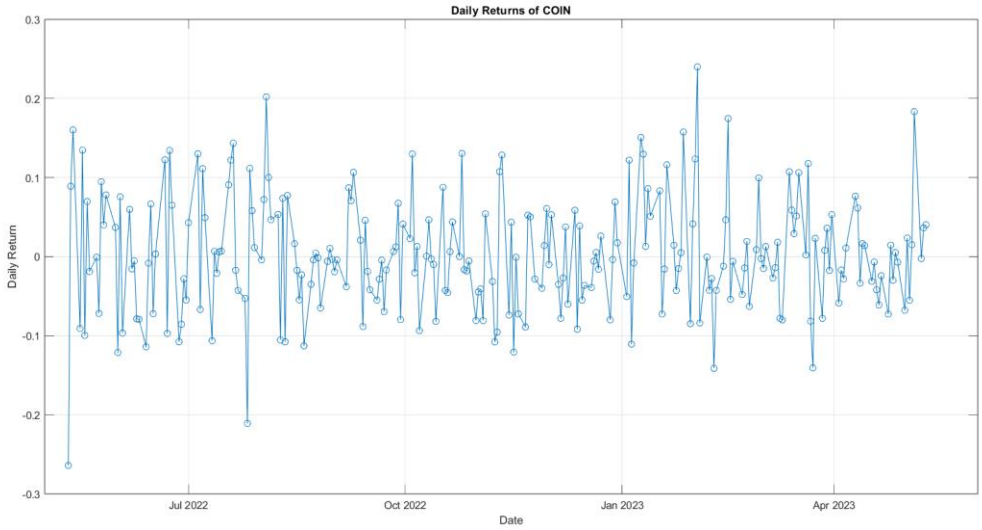
- **AMZN:**

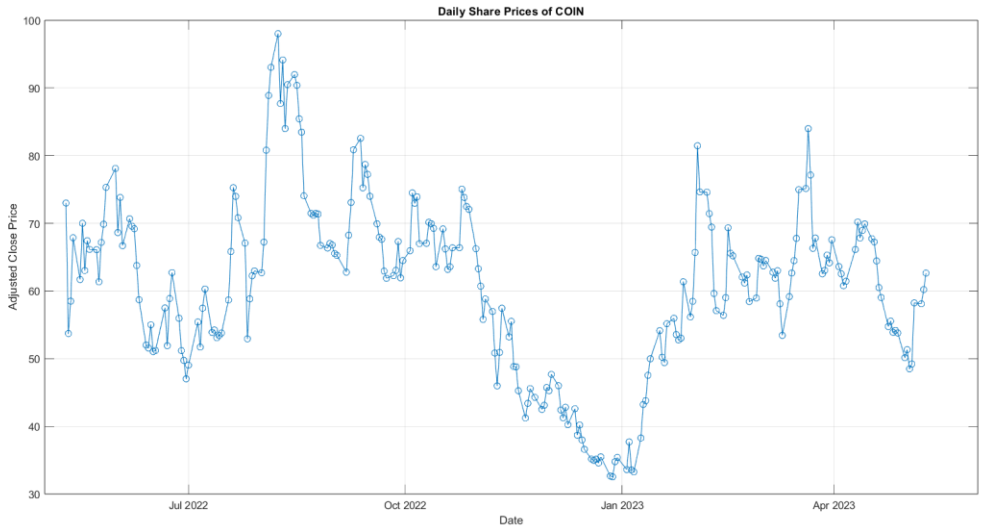




- **Daily Returns:** The returns show high volatility with frequent spikes, indicating significant price movements both upwards and downwards. This reflects the dynamic nature of Amazon's diverse business operations and market reactions to its performance and external factors.
- **Daily Share Prices:** The price trend shows substantial fluctuations over the period, with a notable decline and subsequent recovery. This trend reflects investor responses to Amazon's performance, market conditions, and external economic factors.

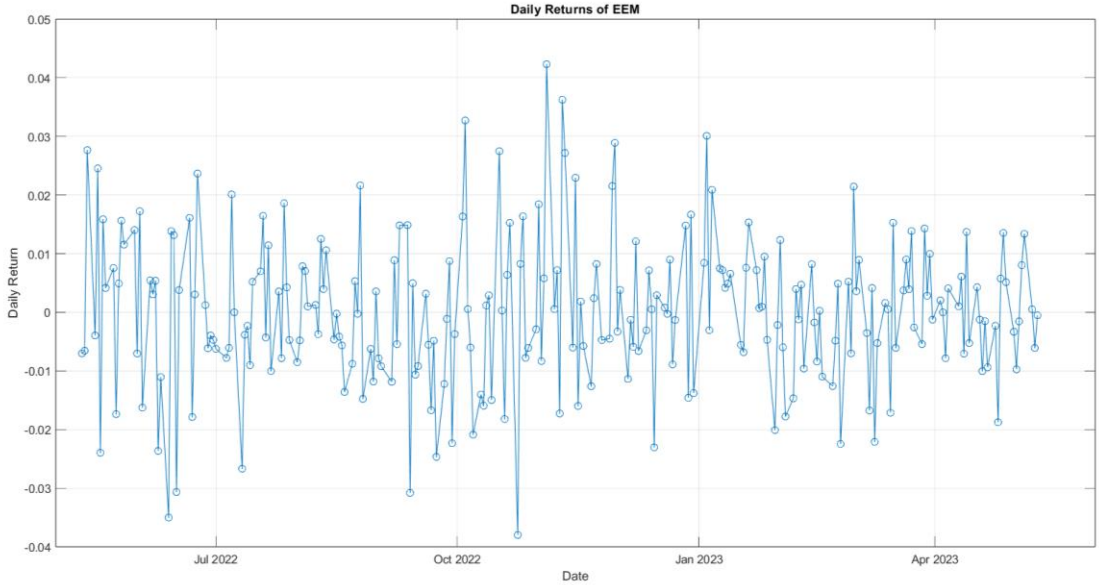
➢ **COIN:**

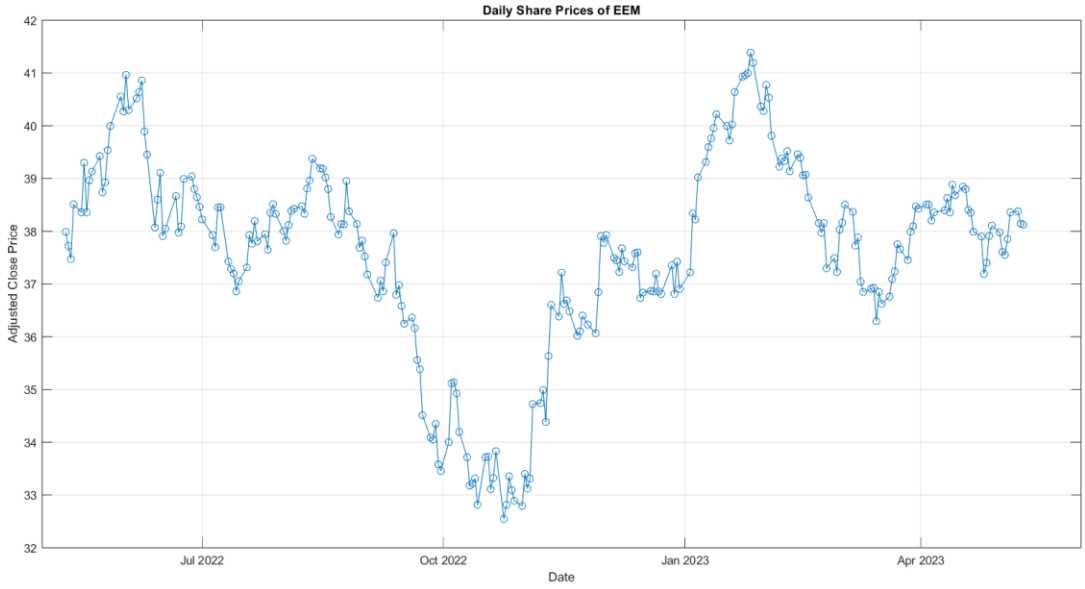




- **Daily Returns:** Coinbase exhibits extreme volatility with several large spikes, particularly higher positive and negative returns. This is typical for an asset tied closely to the highly volatile cryptocurrency market.
- **Daily Share Prices:** COIN's share price is highly volatile with sharp declines and rebounds. This is consistent with the performance of cryptocurrencies and regulatory news impacting the value of crypto-related stocks.

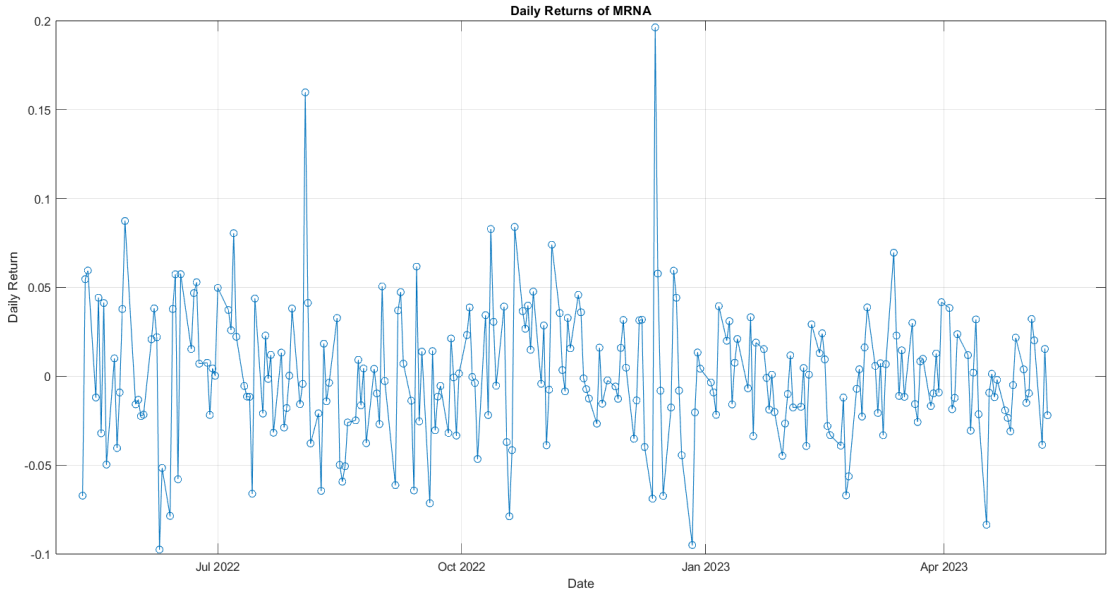
➢ **EEM:**

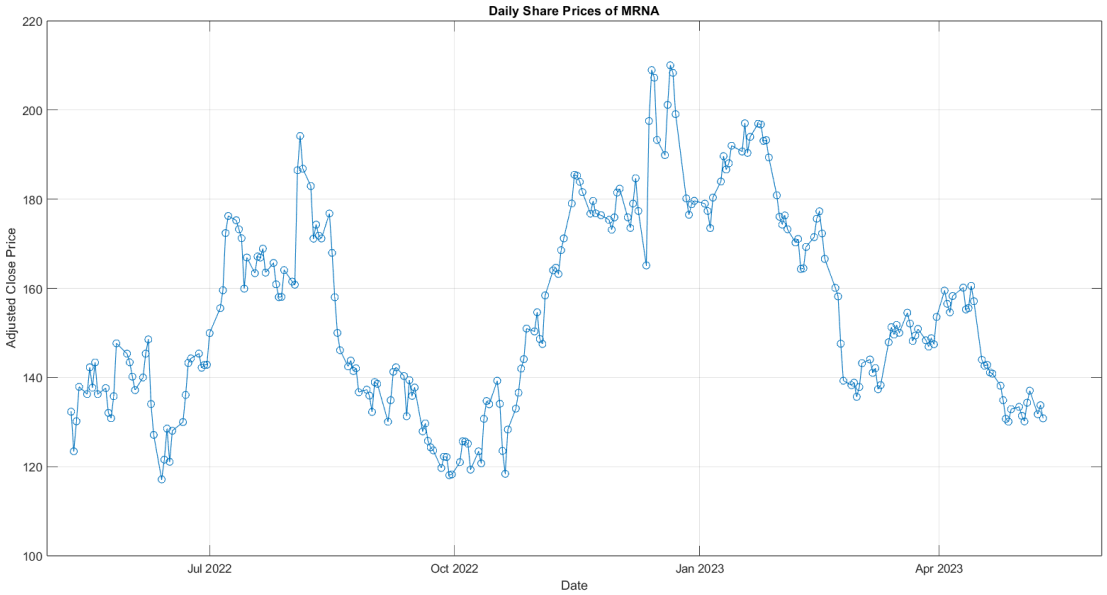




- **Daily Returns:** EEM shows consistent volatility, but the magnitude of daily returns is relatively lower compared to COIN. This reflects the inherent risks and movements in emerging markets, driven by geopolitical and economic factors.
- **Daily Share Prices**: EEM shows fluctuating trends with noticeable peaks and troughs, reflecting the volatility of emerging markets. The price movements indicate sensitivity to global economic conditions and geopolitical events.

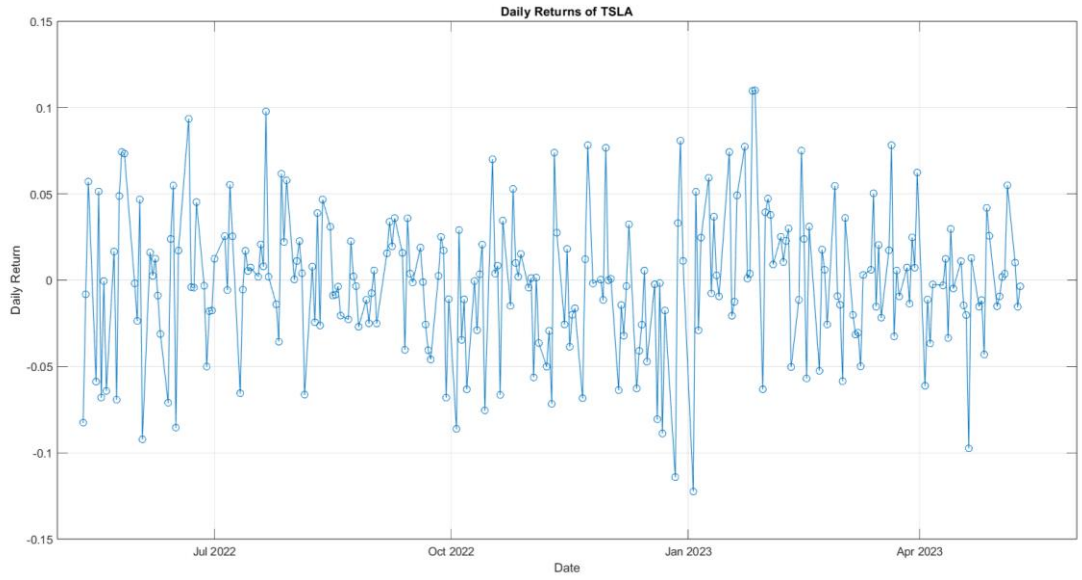
- **MRNA:**

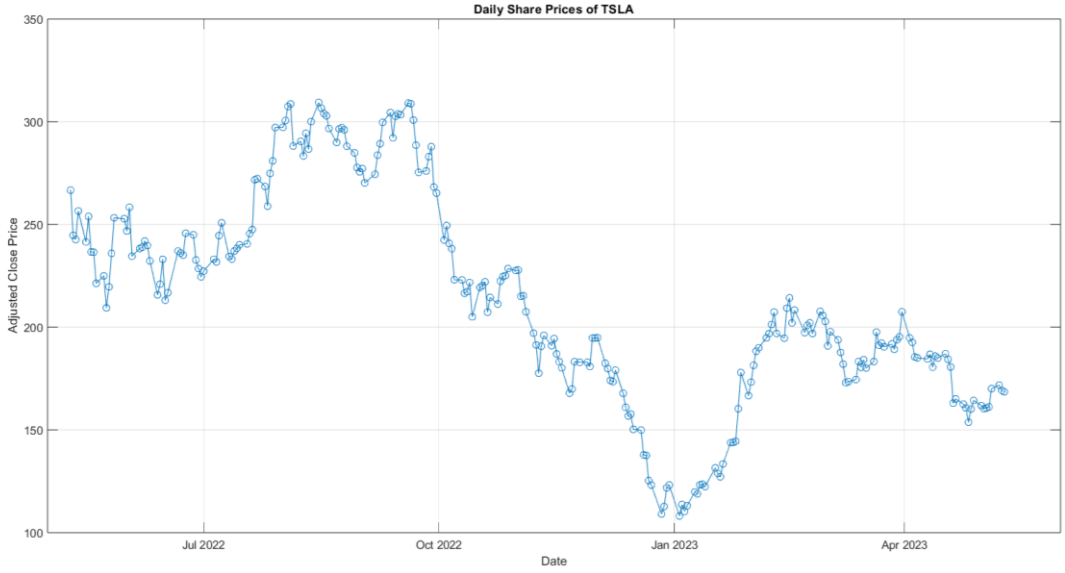




- **Daily Returns:** The returns of MRNA are characterized by sharp spikes, reflecting the biotech sector's sensitivity to news about drug development and approvals. Significant movements are noted, likely linked to announcements related to their COVID-19 vaccine and other products.
- **Daily Share Prices**: MRNA's share prices demonstrate significant peaks and troughs, reflecting the impact of news related to drug trials and approvals. The sharp movements highlight investor sentiment driven by the company's progress in biotechnology innovations.

- **TSLA:**





- **Daily Returns:** TSLA's returns show significant volatility with numerous spikes. This aligns with Tesla's nature as a high-growth company in the innovative electric vehicle sector, which experiences substantial market speculation and investor sentiment swings.
- **Daily Share Prices:** TSLA's share prices exhibit notable volatility, with substantial upward and downward trends. This reflects the company's growth trajectory, market expectations, and reactions to news related to innovation and production targets.

COMPUTATION OF SAMPLE MEAN, VARIANCE, STANDARD DEVIATION (SD) OF THE LOGARITHMIC OR LINEAR RETURNS OF THESE SHARES (IN ANNUAL TERMS)

AMZN:

The output through MATLAB code is:

```
Sample Mean of Daily Returns: 0.000448
Sample Variance of Daily Returns: 0.000806
Sample Standard Deviation of Daily Returns: 0.028383
```

Sample mean: 0.000448

Sample variance: 0.000806

Sample standard deviation: 0.028383

COIN:

The output through MATLAB code is:

```
Sample Mean of Daily Returns: 0.002051
Sample Variance of Daily Returns: 0.005359
Sample Standard Deviation of Daily Returns: 0.073204
```

Sample mean: 0.002051

Sample variance: 0.005359

Sample standard deviation: 0.073204

EEM:

The output through MATLAB code is:

```
Sample Mean of Daily Returns: 0.000094
Sample Variance of Daily Returns: 0.000159
Sample Standard Deviation of Daily Returns: 0.012611
```

Sample mean: 0.000094

Sample variance: 0.000159

Sample standard deviation: 0.012611

MRNA:

The output through MATLAB code is:

```
Sample Mean of Daily Returns: 0.000649
Sample Variance of Daily Returns: 0.001415
Sample Standard Deviation of Daily Returns: 0.037621
```

Sample mean: 0.000649

Sample variance: 0.001415

Sample standard deviation: 0.037621

TSLA:

The output through MATLAB code is:

```
Sample Mean of Daily Returns: -0.000993
Sample Variance of Daily Returns: 0.001665
Sample Standard Deviation of Daily Returns: 0.040806
```

Sample mean: -0.000993

Sample variance: 0.001665

Sample standard deviation: 0.040806

COMPUTATION OF THE CORRESPONDING VARIANCE-COVARIANCE MATRIX 'V' FOR THE RETURNS:

The output through MATLAB code is:

```
>> varcovar
Variance-Covariance Matrix:
0.00080557  0.00130902  0.00020484  0.00043217  0.00061687
0.00130902  0.00535883  0.00043904  0.00106823  0.00173984
0.00020484  0.00043904  0.00015904  0.00018588  0.00023714
0.00043217  0.00106823  0.00018588  0.00141536  0.00048523
0.00061687  0.00173984  0.00023714  0.00048523  0.00166516
```

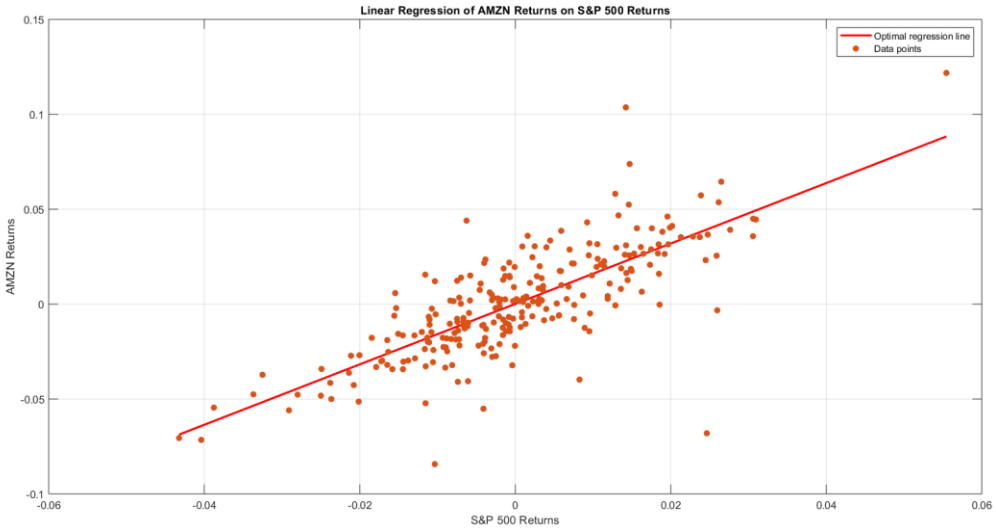
Variance-Covariance Matrix:

0.00080557	0.00130902	0.00020484	0.00043217	0.00061687
0.00130902	0.00535883	0.00043904	0.00106823	0.00173984
0.00020484	0.00043904	0.00015904	0.00018588	0.00023714
0.00043217	0.00106823	0.00018588	0.00141536	0.00048523
0.00061687	0.00173984	0.00023714	0.00048523	0.00166516

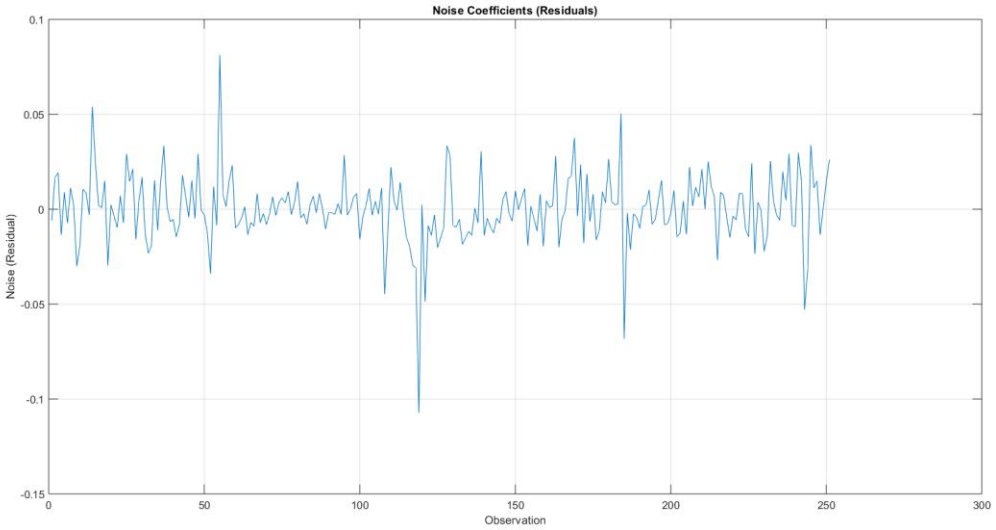
1. **Diagonal Elements (Variance):**
    - The diagonal elements represent the variance of each asset.
    - **COIN** has the highest variance (0.00535883), indicating it is the most volatile asset.
    - **EEM** has the lowest variance (0.00015904), reflecting lower volatility compared to the other assets.
2. **Off-Diagonal Elements (Covariance):**
    - The off-diagonal elements indicate the covariance between pairs of assets.
    - The covariances show that **COIN** has a relatively high covariance with other assets, especially with **AMZN** (0.00130902) and **TSLA** (0.00173984), suggesting that they tend to move together to some extent.
    - **EEM** generally has lower covariance with other assets, indicating more independent price movements compared to others.
3. **Interpretation:**
    - High covariance values between assets like **COIN** and **TSLA** or **AMZN** imply that these assets are likely to exhibit similar price movements under similar market conditions.
    - Lower covariance, such as between **EEM** and other assets, suggests diversification benefits as their price movements are less synchronized.

LINEAR REGRESSION ON THE DATA USING AN APPROPRIATE INDEX AS A PROXY (S&P 500) [10-5-2022 to 10-5-2023] FOR THE MARKET PORTFOLIO. [INCLUDING ALPHA, BETA AND NOISE COEFFICIENTS]

AMZN:



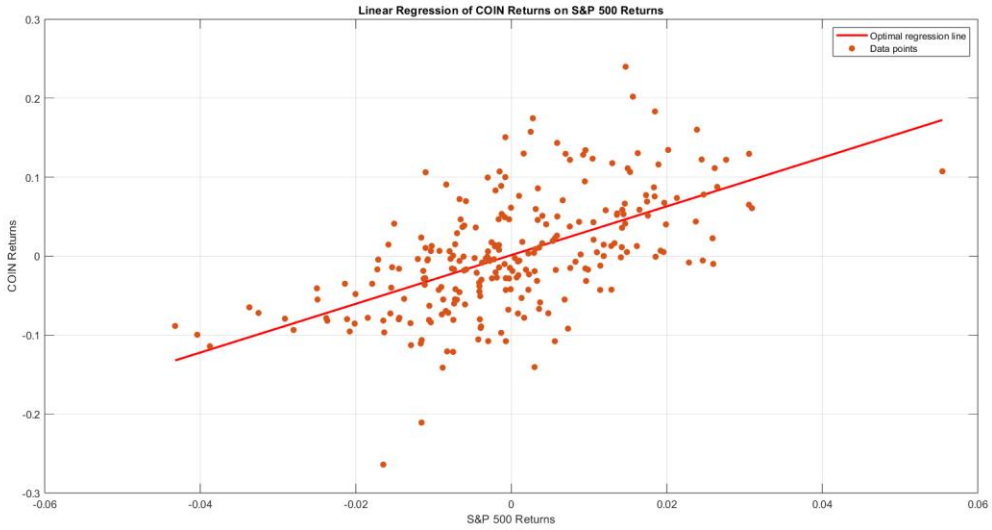
Analysis: The regression plot for AMZN returns against the S&P 500 returns shows a positive relationship. The regression line indicates that as the S&P 500 returns increase, AMZN returns tend to increase as well. This suggests that AMZN is positively correlated with the broader market, albeit with some scatter indicating individual volatility.



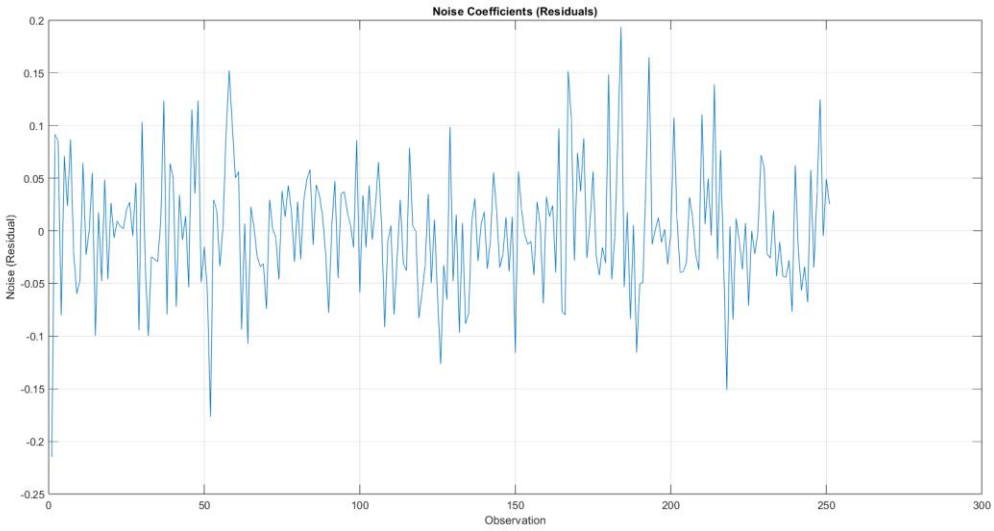
```
Alpha coefficient:
   8.4951e-05

Beta coefficient:
    1.5920
```

COIN:



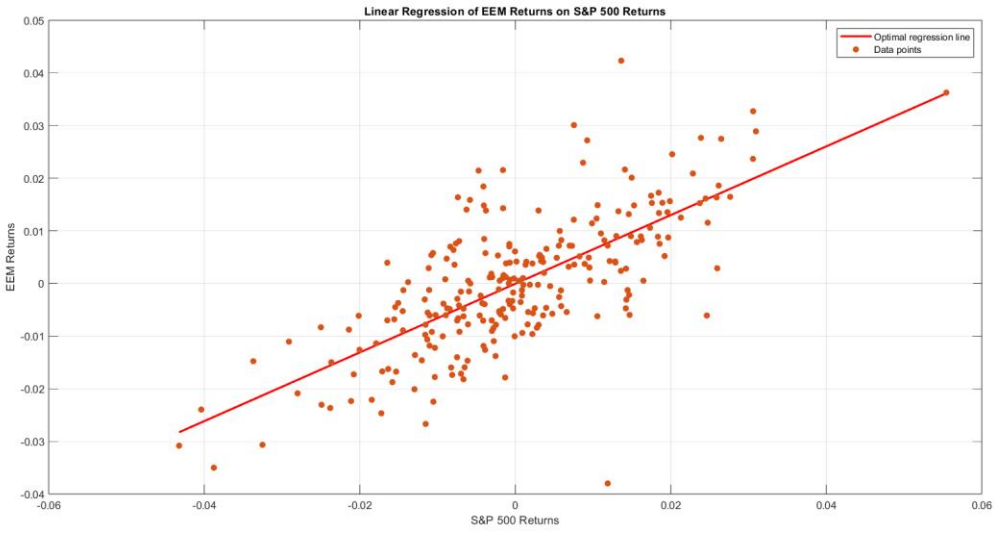
Analysis: The regression plot for COIN returns against the S&P 500 returns also shows a positive relationship but with a wider scatter of data points. This indicates a weaker but positive correlation with the market. The high scatter reflects the inherent volatility and market-specific risks associated with cryptocurrency-related assets.



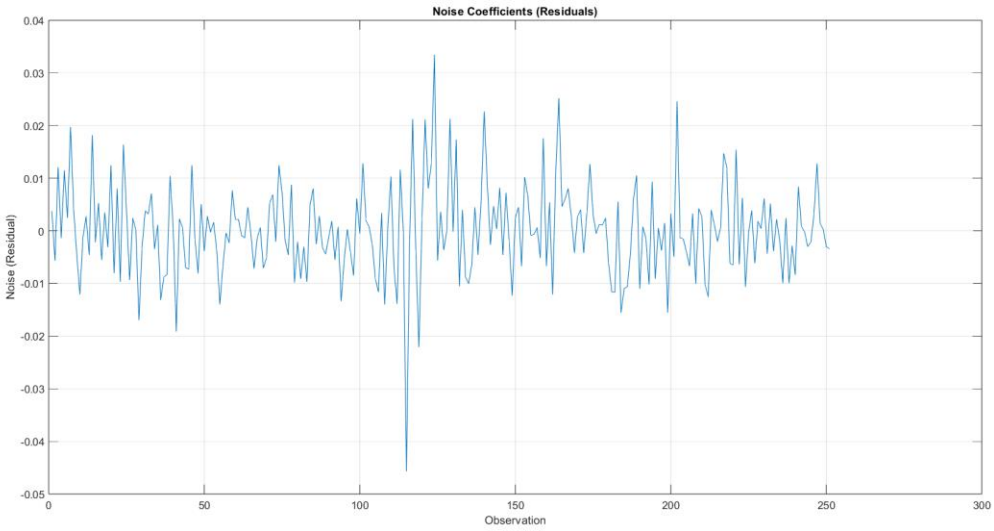

```
Alpha coefficient:
    0.0013

Beta coefficient:
    3.0865
```

EEM:



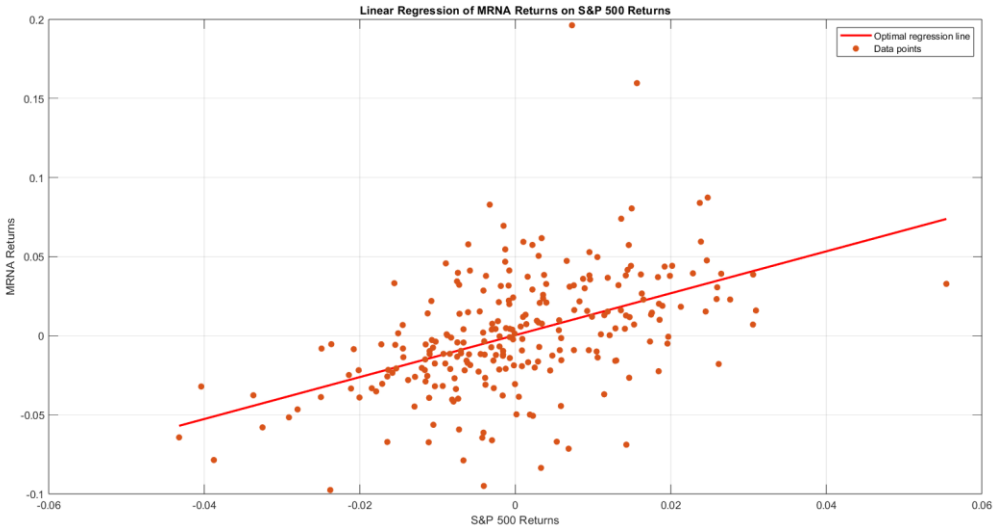
Analysis: The regression plot for EEM returns against the S&P 500 returns indicates a moderate positive correlation. The regression line shows that EEM tends to move in the same direction as the broader market, but the scatter suggests the influence of additional factors specific to emerging markets.



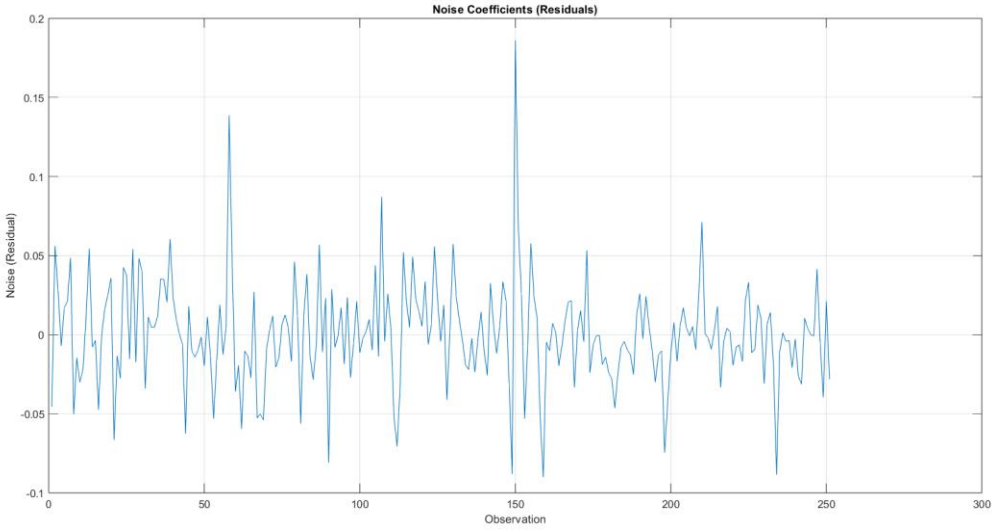

```
Alpha coefficient:
  -5.5135e-05

Beta coefficient:
    0.6525
```

MRNA:



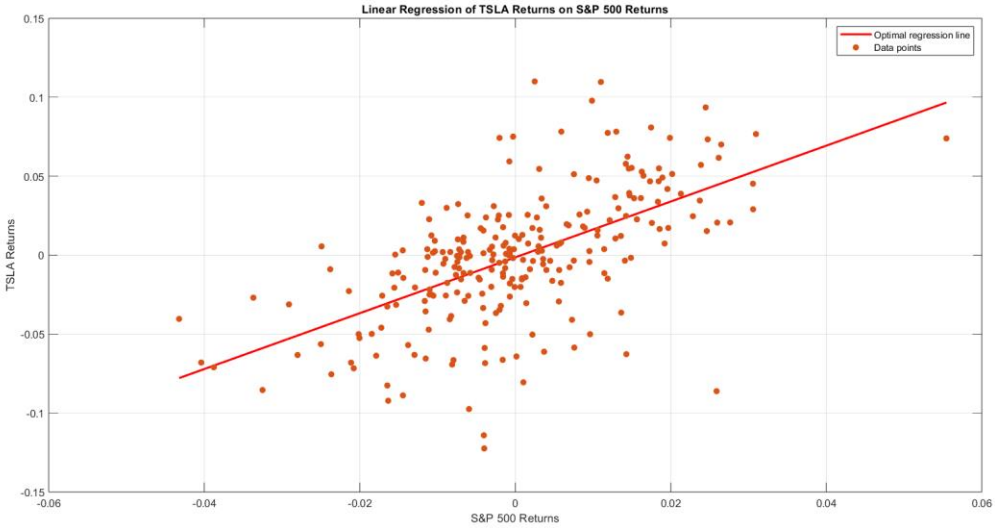
Analysis: The regression plot for MRNA returns against the S&P 500 returns demonstrates a positive correlation, though the scatter is more pronounced. This reflects MRNA's sensitivity to broader market trends while also being influenced significantly by sector-specific news and events, typical in the biotech industry.



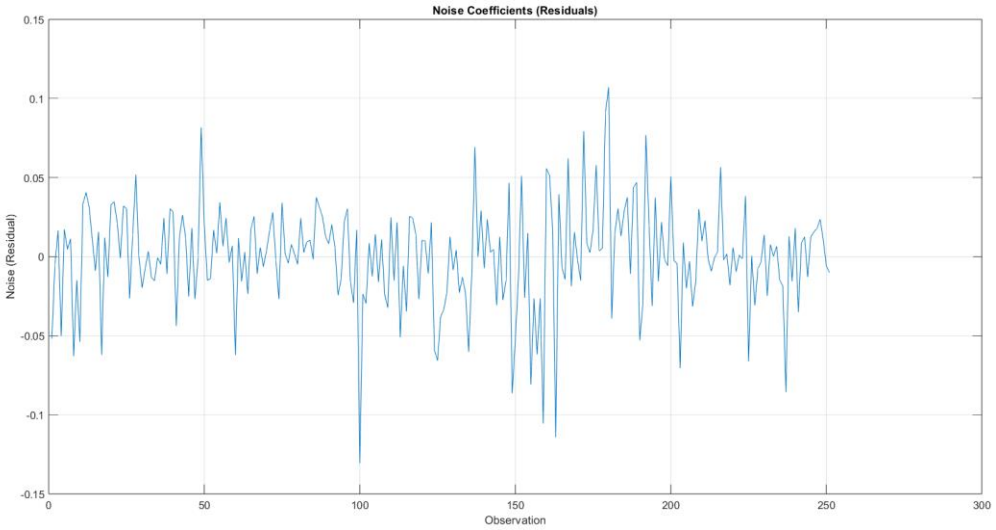

```
Alpha coefficient:
   3.4751e-04

Beta coefficient:
    1.3252
```

TSLA:



Analysis: The regression plot for TSLA returns against the S&P 500 returns shows a positive correlation with some degree of scatter. This indicates that TSLA generally moves with the market but also experiences significant individual volatility, likely due to its innovative nature and market speculation.

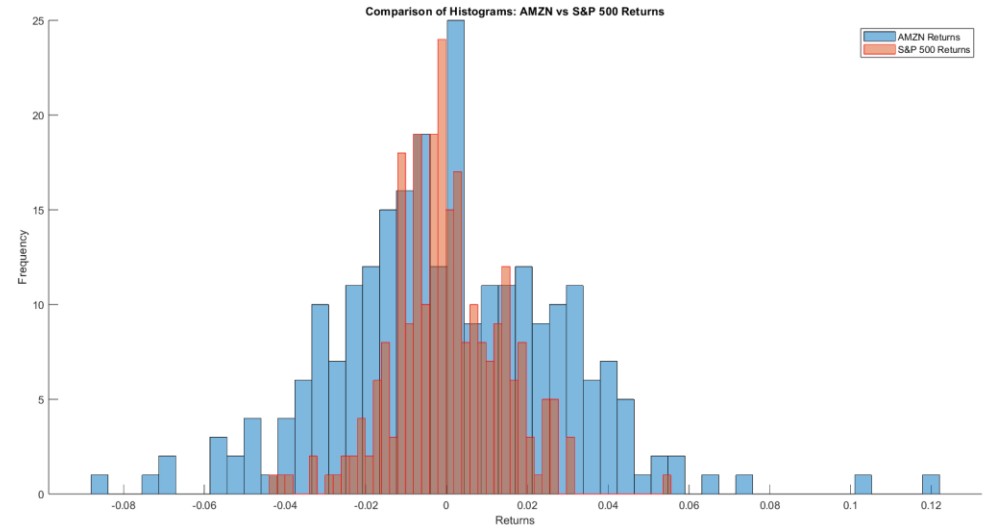



```
Alpha coefficient:
   -0.0014

Beta coefficient:
    1.7684
```

HISTOGRAM AND SCATTER PLOTS:

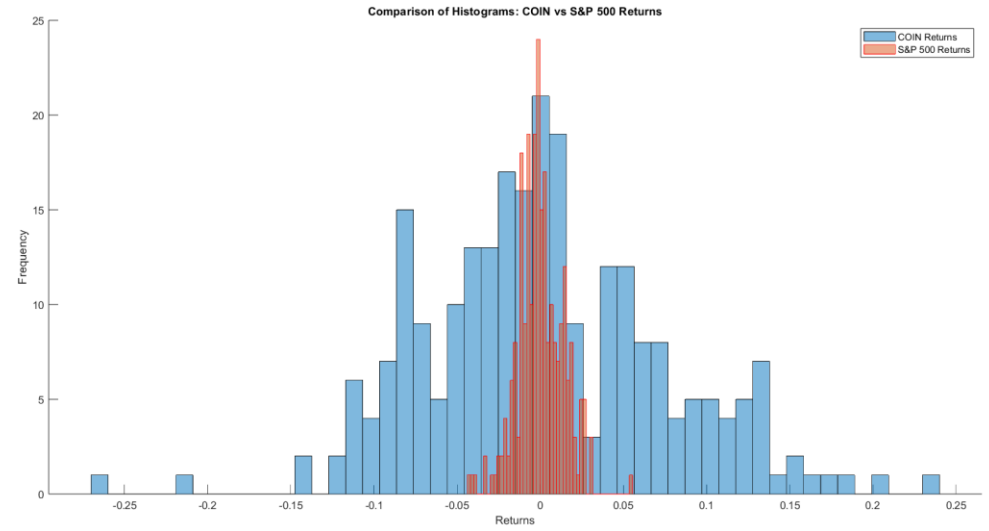
AMZN:

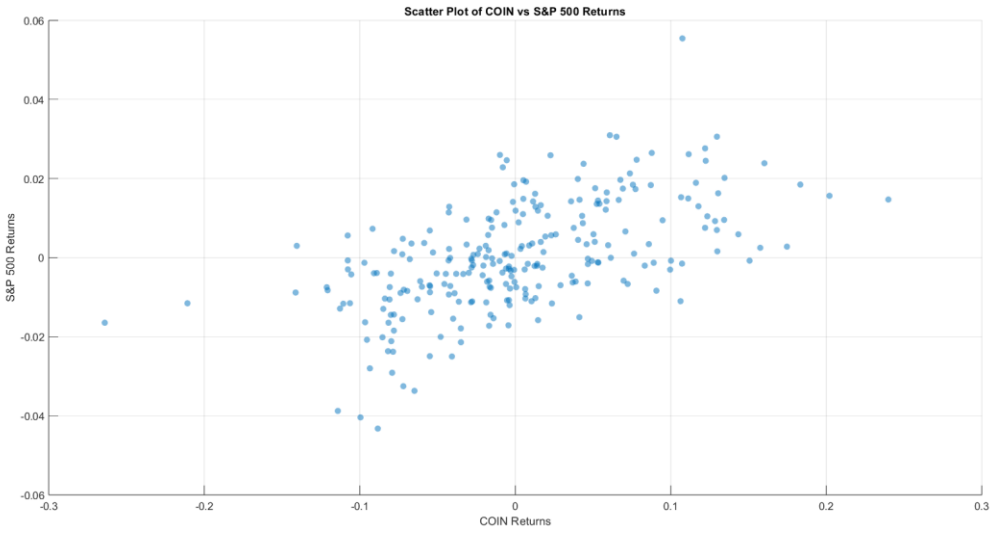


AMZN (Amazon) vs S&P 500 Returns

- **Histogram**: Amazon shows a wider range of returns compared to the S&P 500, indicating higher volatility. The peak is slightly shifted to the right, suggesting higher positive returns more frequently.
- **Scatter Plot**: A positive correlation is observed, indicating that Amazon's returns move in tandem with the S&P 500, though with higher variability.

COIN:

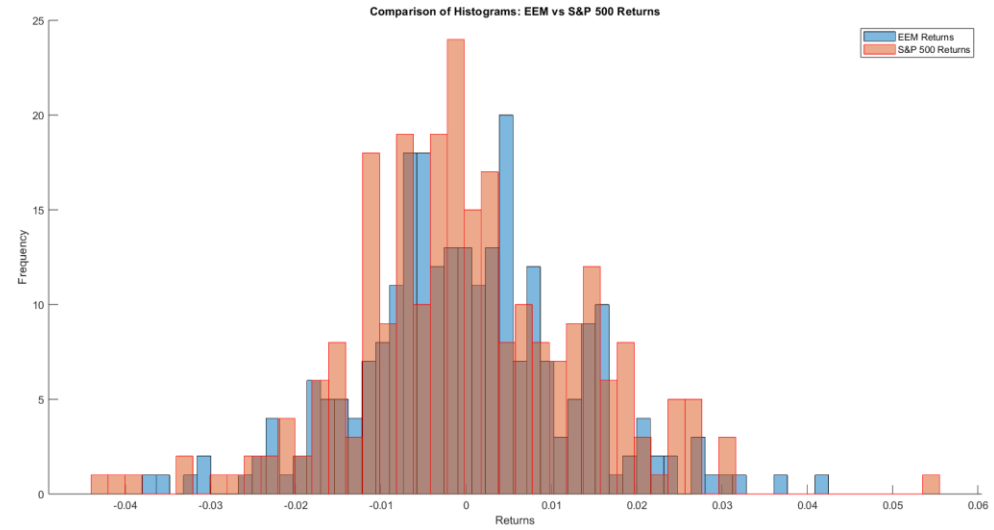


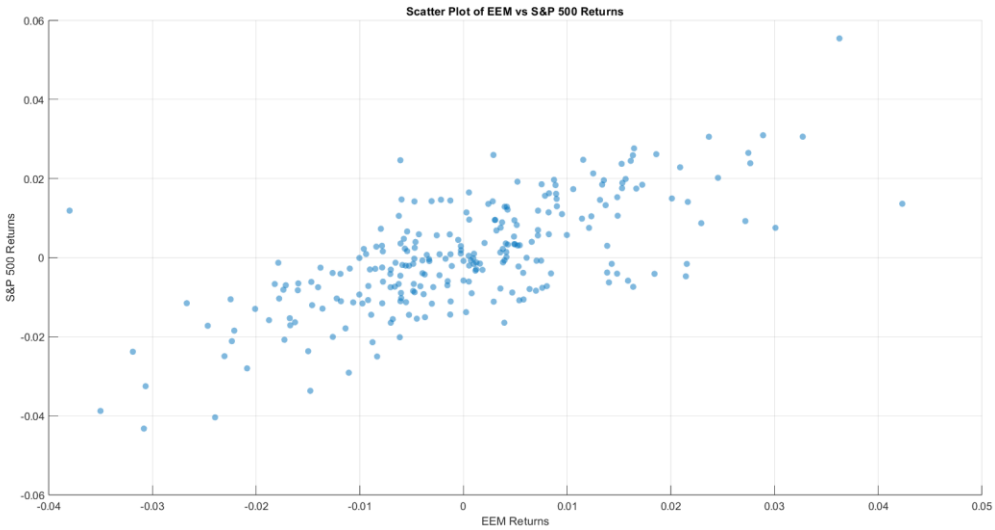


COIN (Coinbase) vs S&P 500 Returns

- **Histogram**: Coinbase has a much wider spread and higher peaks at the extreme negative and positive returns, showcasing higher volatility compared to the S&P 500. The distribution is more spread out, indicating less consistency in returns.
- **Scatter Plot**: Coinbase returns show a positive correlation but with significant outliers, indicating that while it often moves with the S&P 500, it can deviate significantly.

EEM:

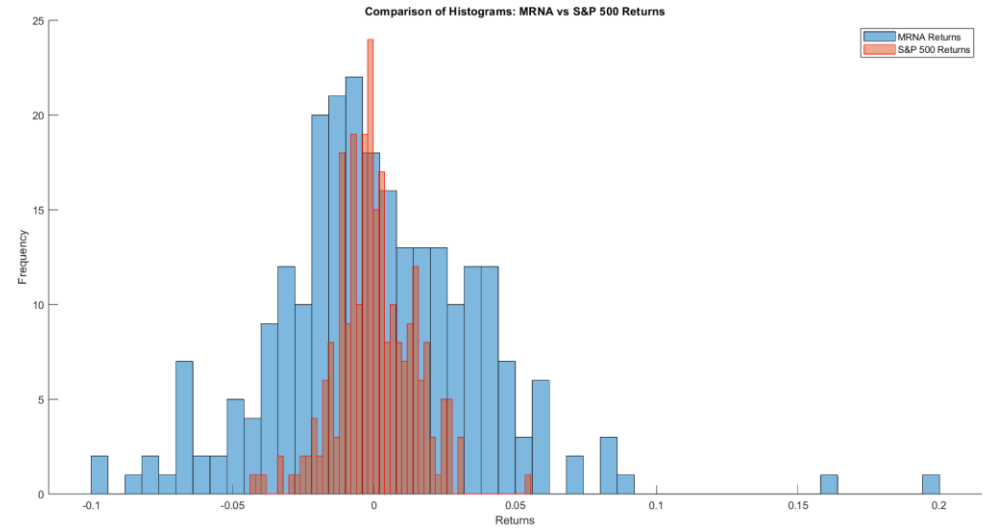


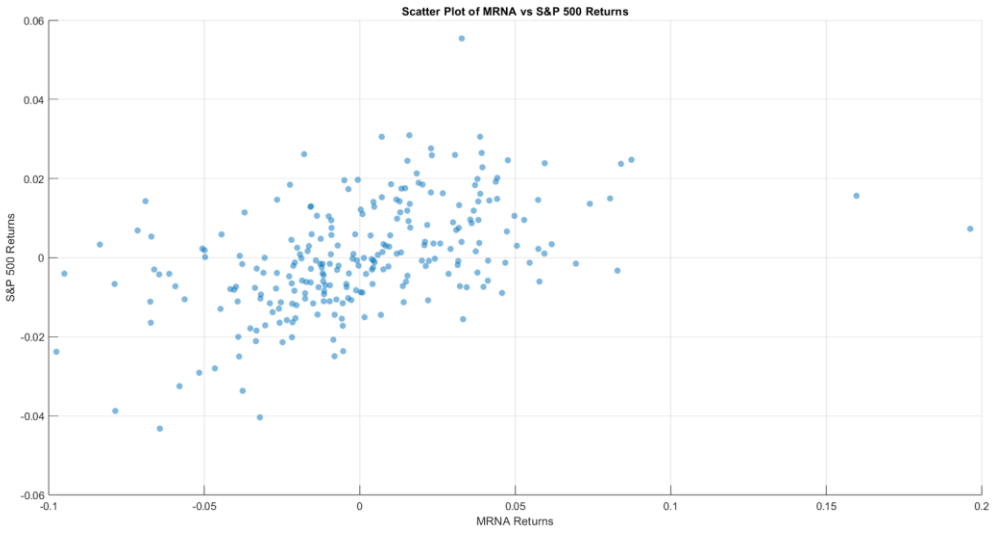


EEM (Emerging Markets ETF) vs S&P 500 Returns

- **Histogram**: Emerging Markets ETF has a distribution closer to the S&P 500 but shows slightly higher peaks on the positive side, indicating some higher positive returns but with similar volatility.
- **Scatter Plot**: Emerging Markets ETF shows a positive correlation with the S&P 500, indicating aligned movements with some variability.

MRNA:

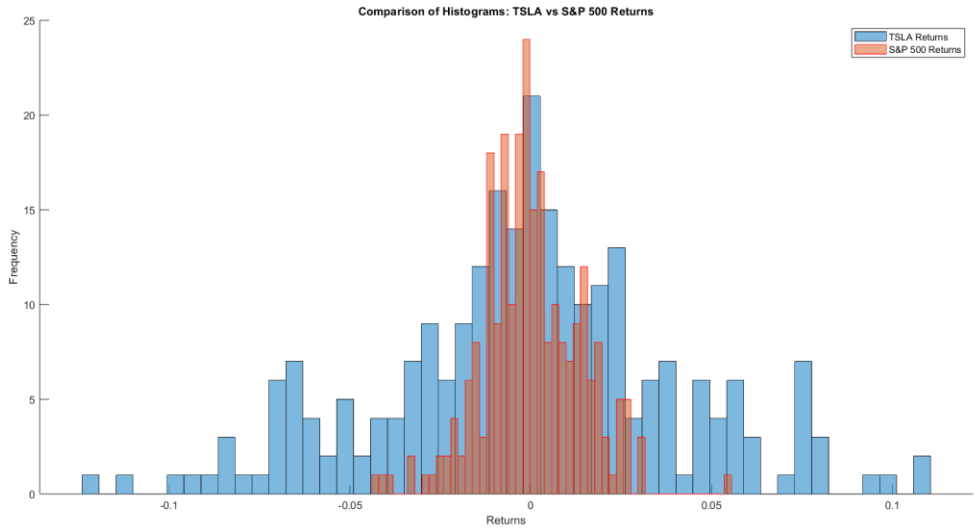


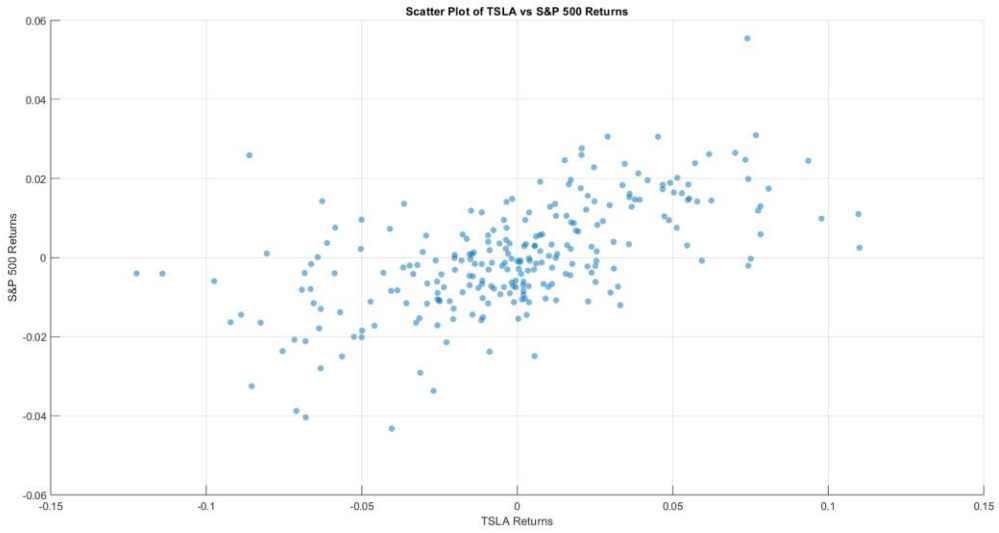


MRNA (Moderna) vs S&P 500 Returns

- **Histogram**: Moderna displays a broader distribution, indicating higher volatility with peaks slightly more pronounced on the positive side compared to the S&P 500.
- **Scatter Plot**: Moderna returns display a positive correlation with the S&P 500 but with higher variance, indicating that while it generally follows the market trend, its movements can be more extreme.

TSLA:

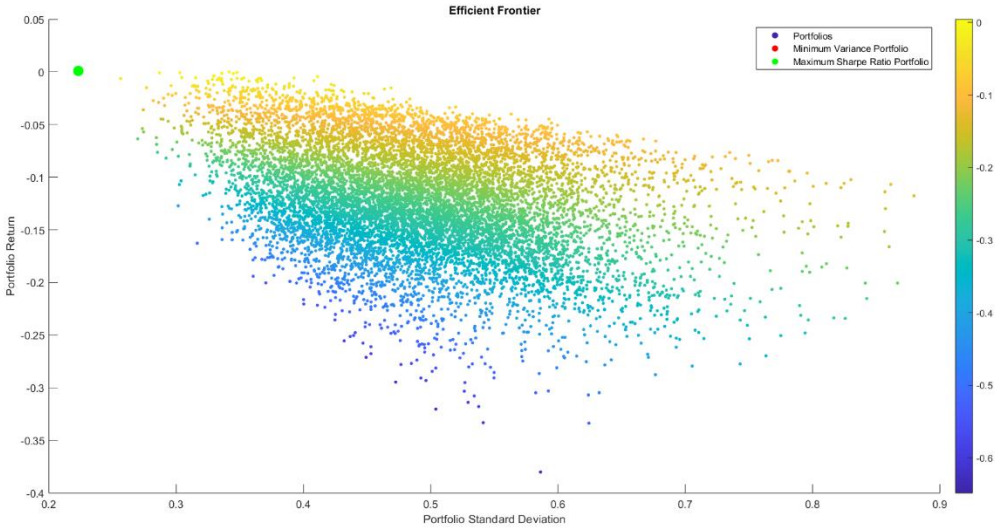




SLA (Tesla) vs S&P 500 Returns

- **Histogram**: Tesla shows a wider distribution, especially on the positive side, suggesting high volatility and frequent high positive returns compared to the S&P 500.
- **Scatter Plot**: Tesla shows a positive correlation with the S&P 500, with notable variance, indicating high sensitivity to market movements and higher volatility.

COMPUTING THE EFFICIENT FRONTIER AND PLOTTING IT, USING THE APPROPRIATE ANNUALIZATION PROCEDURE OUTLINED IN CHAPTER SEVEN OF THE LECTURE NOTES:



Efficient Frontier Diagram Analysis

The Efficient Frontier diagram displays optimal portfolios offering the highest expected return for a given level of risk (standard deviation) or the lowest risk for a given return.

Key Points:

- Portfolios: Blue dots represent various asset combinations based on their expected returns and standard deviations.
- Minimum Variance Portfolio: Highlighted in red, this portfolio has the lowest risk.
- Maximum Sharpe Ratio Portfolio: Marked in green, this portfolio offers the best risk-adjusted return.

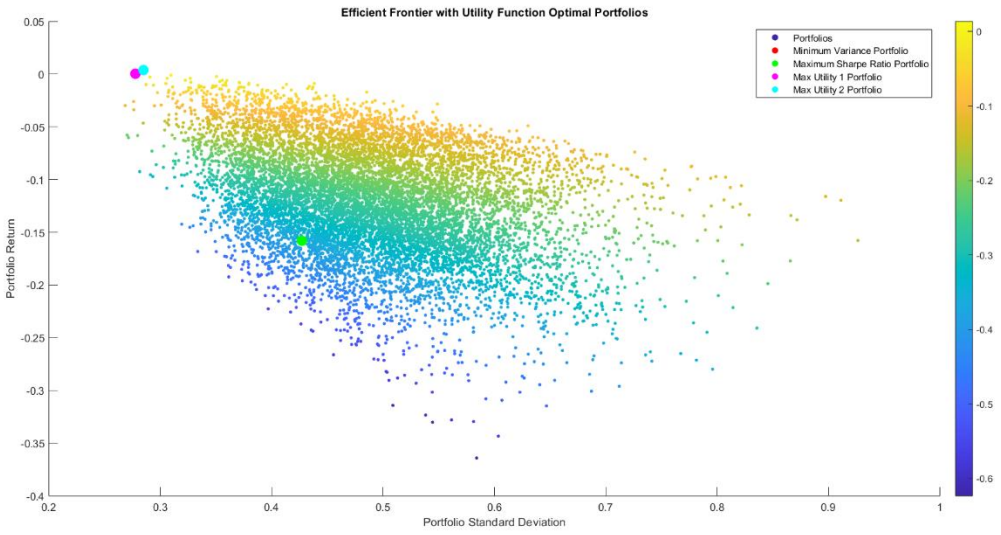
The colour gradient shows the range of returns, with warmer colours (yellow) indicating higher returns and cooler colours (blue) indicating lower returns.

USING OF BOTH UTILITY FUNCTION:

$$[u(\mu, \sigma) = \mu - \alpha\sigma^2]$$

$$[u(\mu, \sigma) = \frac{\mu}{\sigma^2}]$$

TO SELCT THE OPTIMAL PORTFOLIO.



Efficient Frontier with Utility Function Optimal Portfolios

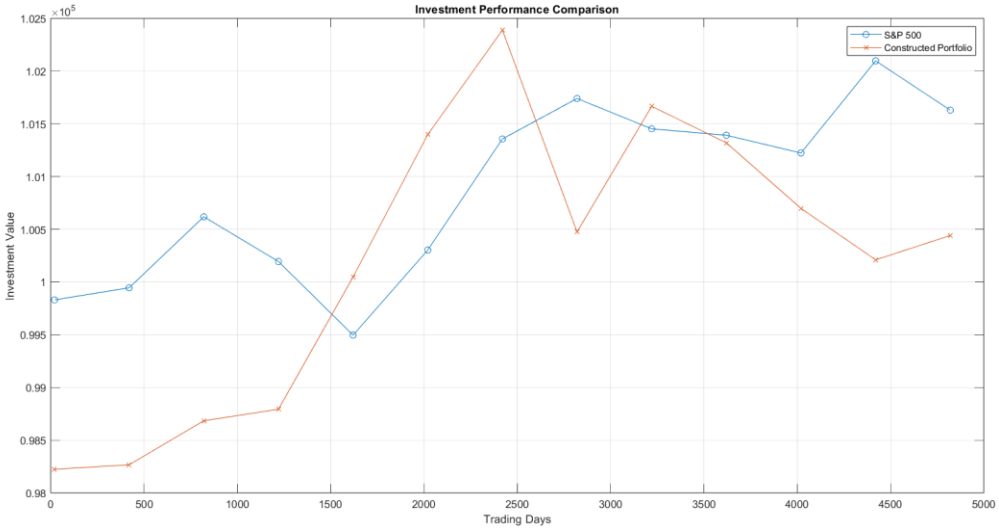
This graph shows the Efficient Frontier and optimal portfolios based on different utility functions:

Key Points:

1. **Portfolios:** Blue dots represent various asset combinations with their expected returns and standard deviations.
2. **Minimum Variance Portfolio:** Red dot, indicating the lowest risk portfolio.
3. **Maximum Sharpe Ratio Portfolio:** Green dot, showing the highest risk-adjusted return.
4. **Max Utility Portfolios:**
    - **Max Utility 1 Portfolio:** Magenta dot, optimal for the first utility function.
    - **Max Utility 2 Portfolio:** Cyan dot, optimal for the second utility function.

The colour gradient shows the range of returns, with yellow indicating higher returns and blue indicating lower returns.

COMPARING INVESTMENT PERFORMANCE FOR PORTFOLIO VS INDEX PERFORMANCE AT SOME FUTURE POINT [S&P500 910-5-2023 TO 10-5-2024)]



Investment Performance Comparison

The graph compares the performance of the S&P 500 (blue line) with a constructed portfolio (orange line) over a series of trading days.

- **S&P 500**: The investment value generally increases with some fluctuations, indicating overall growth with periods of volatility.
- **Constructed Portfolio**: Initially underperforms compared to the S&P 500, showing a gradual increase with a significant spike around the midpoint, followed by more fluctuations.

This comparison highlights the constructed portfolio's more volatile performance relative to the steady growth of the S&P 500.

STUDYING IF ASSET PROTECTION WOULD HAVE BEEN USEFUL

```
Asset: C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\TSLA.xls
Call Price: 50.55
Put Price: 12.01

Asset: C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\AMZN_1.xlsx
Call Price: 7.23
Put Price: 27.04

Asset: C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\COIN_1.xlsx
Call Price: 7.21
Put Price: 74.57

Asset: C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\EEM_1.xlsx
Call Price: 0.00
Put Price: 91.88

Asset: C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\MRNA_1.xlsx
Call Price: 22.00
Put Price: 21.17
```

Asset: TSLA
Call Price: 50.55
Put Price: 12.01

Asset: AMZN
Call Price: 7.23
Put Price: 27.04

Asset: COIN
Call Price: 7.21
Put Price: 74.57

Asset: EEM
Call Price: 0.00
Put Price: 91.88

Asset: MRNA
Call Price: 22.00
Put Price: 21.17

```
Asset: C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\TSLA.xls
Total Profit (Long Call, Short Put): -34.68

Asset: C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\AMZN_1.xlsx
Total Profit (Long Call, Short Put): 0.67

Asset: C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\COIN_1.xlsx
Total Profit (Long Call, Short Put): -4.63

Asset: C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\EEM_1.xlsx
Total Profit (Long Call, Short Put): 0.07

Asset: C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\MRNA_1.xlsx
Total Profit (Long Call, Short Put): -0.74
```

Asset: TSLA
Total Profit (Long Call, Short Put): -34.68

Asset: AMZN
Total Profit (Long Call, Short Put): 0.67

Asset: COIN
Total Profit (Long Call, Short Put): -4.63

Asset: EEM
Total Profit (Long Call, Short Put): 0.07

Asset: MRNA
Total Profit (Long Call, Short Put): -0.74

From the provided data, it appears that the use of call and put options for asset protection varies across different assets. Here's a brief summary:

- **TSLA (Tesla)**: The total profit from a long call and short put strategy is -34.68, indicating a significant loss. This suggests that asset protection strategies were not useful in this case.
- **AMZN (Amazon)**: The total profit is 0.67, which is positive but minimal. This suggests a marginal benefit from asset protection.
- **COIN (Coinbase)**: The total profit is -4.63, indicating a loss. This suggests that asset protection strategies were not useful.
- **EEM (Emerging Markets ETF)**: The total profit is 0.07, which is very small. This suggests a negligible benefit from asset protection.
- **MRNA (Moderna)**: The total profit is -0.74, indicating a slight loss. This suggests that asset protection strategies were not useful.

Additional Concepts

1. Short Selling Constraints: These limitations restrict selling borrowed shares, aiming to buy them back at lower prices. They can reduce market efficiency by limiting corrections of overvalued stocks. For example, constraints could increase the volatility of Tesla (TSLA) and Coinbase (COIN), both prone to speculative trading.
2. Detection of Outliers: Outliers can skew portfolio optimization and risk assessment. Detecting and managing them ensures robust analysis. For instance, addressing the extreme volatility in Coinbase's returns using robust statistical techniques like winsorization can stabilize the portfolio.
3. Robust Optimization: This technique creates portfolios resilient to market uncertainties. Considering the high volatility of assets like COIN and TSLA, methods such as the Black-Litterman model or scenario analysis ensure the portfolio remains optimal under adverse conditions.
4. Black-Litterman Model: This model combines market returns with investor views, adjusting asset weights based on both. Applying it allows for a nuanced portfolio, balancing the high returns of TSLA and the risks of COIN with the stability of EEM (iShares MSCI Emerging Markets ETF).

Impact on Portfolio Selection

Example Portfolio Analysis:

- Statistical Analysis:
    - Sample Means and Variances: Highlight expected return and risk for each asset.
    - Variance-Covariance Matrix: Essential for understanding asset return interrelationships and diversification.
- Advanced Concepts Application:
    - **Short Selling Constraints Impact:** Reduced short selling in TSLA and COIN could lead to higher volatility.
    - **Detection of Outliers:** Smoothing COIN’s returns reduces skewness, aiding predictive accuracy.
    - **Robust Optimization**: Techniques like stress testing manage high volatility, ensuring portfolio resilience.
    - **Black-Litterman Model**: Adjusts returns based on market views and investor beliefs, leading to a more balanced portfolio despite COIN’s volatility.

Findings and Conclusion of Portfolio Analysis

Findings:

1. Volatility and Returns:
    - TSLA: High volatility; long call and short put strategy resulted in -34.68.
    - AMZN: Significant fluctuations; minimal positive return of 0.67.
    - COIN: Extreme volatility; resulted in a loss of -4.63.
    - EEM: Consistent lower volatility; negligible positive return of 0.07.
    - MRNA: High sensitivity; slight loss of -0.74.
2. Correlation with Market Index (S&P 500):
    - Most assets show a positive correlation, but the degree varies.
    - AMZN and TSLA: Higher variability in correlation.
    - COIN: Weaker correlation with high volatility.
    - EEM: Moderate correlation influenced by emerging markets.
    - MRNA: Positive correlation with sector-specific influence.
3. Efficient Frontier and Optimal Portfolios:
    - Minimum Variance Portfolio: Lowest risk.
    - Maximum Sharpe Ratio Portfolio: Best risk-adjusted return.
    - Utility function-based portfolios cater to different risk tolerances.
4. Performance Comparison:
    - The constructed portfolio initially underperformed the S&P 500, showing higher volatility.
5. Asset Protection Strategies:
    - Generally ineffective, resulting in losses or minimal gains.

Conclusion:

- High volatility assets like TSLA and COIN pose significant risks.
- Stable assets like AMZN and EEM benefit minimally from protection strategies.
- The constructed portfolio's higher volatility compared to the S&P 500 highlights its sensitivity to market conditions.
- Efficient Frontier and utility function analyses provide valuable insights for optimal portfolio selection based on risk-return preferences.
- Asset protection strategies were not useful, indicating the need for more robust risk management techniques tailored to each asset.

APPENDIX

CODES AND THEIR EXPLANATIONS:

CODE FOR FINDING DAILY RETURNS AND PLOTTING DAILY RETURNS AND SHARE PRICES, FINDING SAMPLE MEAN, VARIANCE, STANDARD DEVIATIONS;

AMZN:

```
% Define the file path
file_path = 'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\AMZN_1.xlsx';

% Read the Excel file into a table
data = readtable(file_path);

% Display the first few rows of the table
disp(head(data));

% Convert the 'Date' column to datetime format
% Assuming the date format is 'yyyy-MM-dd'
data.Date = datetime(data.Date, 'InputFormat', 'yyyy-MM-dd');

% Compute the daily returns if not already present
if any(ismember(data.Properties.VariableNames, 'Daily_returns'))
    % Daily returns column exists, check for missing values
    if all(isnan(data.Daily_returns))
        data.Daily_returns = [NaN; diff(data.AdjClose) ./ data.AdjClose(1:end-1)];
    end
else
    % Daily returns column does not exist, compute it
    data.Daily_returns = [NaN; diff(data.AdjClose) ./ data.AdjClose(1:end-1)];
end

% Add the daily returns column to the data table
data.Daily_returns = data.Daily_returns;

% Save the updated table back to the Excel file
writetable(data, file_path);

% Plot the daily share prices
figure;
plot(data.Date, data.AdjClose, '-o');
title('Daily Share Prices of AMZN');
xlabel('Date');
ylabel('Adjusted Close Price');
grid on;
```

```
% Plot the daily returns
figure;
plot(data.Date, data.Daily_returns, '-o');
title('Daily Returns of AMZN');
xlabel('Date');
ylabel('Daily Return');
grid on;

% Compute sample mean, variance, and standard deviation of the daily returns
mean_daily_returns = mean(data.Daily_returns, 'omitnan');
variance_daily_returns = var(data.Daily_returns, 'omitnan');
std_daily_returns = std(data.Daily_returns, 'omitnan');

% Display the computed statistics
fprintf('Sample Mean of Daily Returns: %.6f\n', mean_daily_returns);
fprintf('Sample Variance of Daily Returns: %.6f\n', variance_daily_returns);
fprintf('Sample Standard Deviation of Daily Returns: %.6f\n', std_daily_returns);
```

COIN:

```
% Define the file path
file_path = 'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\COIN_1.xlsx';

% Read the Excel file into a table
data = readtable(file_path);

% Display the first few rows of the table
disp(head(data));

% Convert the 'Date' column to datetime format
% Assuming the date format is 'yyyy-MM-dd'
data.Date = datetime(data.Date, 'InputFormat', 'yyyy-MM-dd');

% Compute the daily returns if not already present
if any(ismember(data.Properties.VariableNames, 'Daily_returns'))
    % Daily returns column exists, check for missing values
    if all(isnan(data.Daily_returns))
        data.Daily_returns = [NaN; diff(data.AdjClose) ./ data.AdjClose(1:end-1)];
    end
else
    % Daily returns column does not exist, compute it
    data.Daily_returns = [NaN; diff(data.AdjClose) ./ data.AdjClose(1:end-1)];
end

% Add the daily returns column to the data table
data.Daily_returns = data.Daily_returns;

% Save the updated table back to the Excel file
writetable(data, file_path);

% Plot the daily share prices
figure;
plot(data.Date, data.AdjClose, '-o');
title('Daily Share Prices of COIN');
xlabel('Date');
ylabel('Adjusted Close Price');
grid on;
```

```
% Plot the daily returns
figure;
plot(data.Date, data.Daily_returns, '-o');
title('Daily Returns of COIN');
xlabel('Date');
ylabel('Daily Return');
grid on;

% Compute sample mean, variance, and standard deviation of the daily returns
mean_daily_returns = mean(data.Daily_returns, 'omitnan');
variance_daily_returns = var(data.Daily_returns, 'omitnan');
std_daily_returns = std(data.Daily_returns, 'omitnan');

% Display the computed statistics
fprintf('Sample Mean of Daily Returns: %.6f\n', mean_daily_returns);
fprintf('Sample Variance of Daily Returns: %.6f\n', variance_daily_returns);
fprintf('Sample Standard Deviation of Daily Returns: %.6f\n', std_daily_returns);
```

EEM:

```
% Define the file path
file_path = 'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\EEM_1.xlsx';

% Read the Excel file into a table
data = readtable(file_path);

% Display the first few rows of the table
disp(head(data));

% Convert the 'Date' column to datetime format
% Assuming the date format is 'yyyy-MM-dd'
data.Date = datetime(data.Date, 'InputFormat', 'yyyy-MM-dd');

% Compute the daily returns if not already present
if any(ismember(data.Properties.VariableNames, 'Daily_returns'))
    % Daily returns column exists, check for missing values
    if all(isnan(data.Daily_returns))
        data.Daily_returns = [NaN; diff(data.AdjClose) ./ data.AdjClose(1:end-1)];
    end
else
    % Daily returns column does not exist, compute it
    data.Daily_returns = [NaN; diff(data.AdjClose) ./ data.AdjClose(1:end-1)];
end

% Add the daily returns column to the data table
data.Daily_returns = data.Daily_returns;

% Save the updated table back to the Excel file
writetable(data, file_path);

% Plot the daily share prices
figure;
plot(data.Date, data.AdjClose, '-o');
title('Daily Share Prices of EEM');
xlabel('Date');
ylabel('Adjusted Close Price');
grid on;
```

```
% Plot the daily returns
figure;
plot(data.Date, data.Daily_returns, '-o');
title('Daily Returns of EEM');
xlabel('Date');
ylabel('Daily Return');
grid on;

% Compute sample mean, variance, and standard deviation of the daily returns
mean_daily_returns = mean(data.Daily_returns, 'omitnan');
variance_daily_returns = var(data.Daily_returns, 'omitnan');
std_daily_returns = std(data.Daily_returns, 'omitnan');

% Display the computed statistics
fprintf('Sample Mean of Daily Returns: %.6f\n', mean_daily_returns);
fprintf('Sample Variance of Daily Returns: %.6f\n', variance_daily_returns);
fprintf('Sample Standard Deviation of Daily Returns: %.6f\n', std_daily_returns);
```

MRNA:

```
% Define the file path
file_path = 'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\MRNA_1.xlsx';

% Read the Excel file into a table
data = readtable(file_path);

% Display the first few rows of the table
disp(head(data));

% Convert the 'Date' column to datetime format
% Assuming the date format is 'yyyy-MM-dd'
data.Date = datetime(data.Date, 'InputFormat', 'yyyy-MM-dd');

% Compute the daily returns if not already present
if any(ismember(data.Properties.VariableNames, 'Daily_returns'))
    % Daily returns column exists, check for missing values
    if all(isnan(data.Daily_returns))
        data.Daily_returns = [NaN; diff(data.AdjClose) ./ data.AdjClose(1:end-1)];
    end
else
    % Daily returns column does not exist, compute it
    data.Daily_returns = [NaN; diff(data.AdjClose) ./ data.AdjClose(1:end-1)];
end

% Add the daily returns column to the data table
data.Daily_returns = data.Daily_returns;

% Save the updated table back to the Excel file
writetable(data, file_path);

% Plot the daily share prices
figure;
plot(data.Date, data.AdjClose, '-o');
title('Daily Share Prices of MRNA');
xlabel('Date');
ylabel('Adjusted Close Price');
grid on;
```

```
% Plot the daily returns
figure;
plot(data.Date, data.Daily_returns, '-o');
title('Daily Returns of MRNA');
xlabel('Date');
ylabel('Daily Return');
grid on;

% Compute sample mean, variance, and standard deviation of the daily returns
mean_daily_returns = mean(data.Daily_returns, 'omitnan');
variance_daily_returns = var(data.Daily_returns, 'omitnan');
std_daily_returns = std(data.Daily_returns, 'omitnan');

% Display the computed statistics
fprintf('Sample Mean of Daily Returns: %.6f\n', mean_daily_returns);
fprintf('Sample Variance of Daily Returns: %.6f\n', variance_daily_returns);
fprintf('Sample Standard Deviation of Daily Returns: %.6f\n', std_daily_returns);
```

TSLA:

```
% Define the file path
file_path = 'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\TSLA.xls';

% Read the Excel file into a table
data = readtable(file_path);

% Display the first few rows of the table
disp(head(data));

% Convert the 'Date' column to datetime format
% Assuming the date format is 'yyyy-MM-dd'
data.Date = datetime(data.Date, 'InputFormat', 'yyyy-MM-dd');

% Compute the daily returns if not already present
if any(ismember(data.Properties.VariableNames, 'Daily_returns'))
    % Daily returns column exists, check for missing values
    if all(isnan(data.Daily_returns))
        data.Daily_returns = [NaN; diff(data.AdjClose) ./ data.AdjClose(1:end-1)];
    end
else
    % Daily returns column does not exist, compute it
    data.Daily_returns = [NaN; diff(data.AdjClose) ./ data.AdjClose(1:end-1)];
end

% Add the daily returns column to the data table
data.Daily_returns = data.Daily_returns;

% Save the updated table back to the Excel file
writetable(data, file_path);

% Plot the daily share prices
figure;
plot(data.Date, data.AdjClose, '-o');
title('Daily Share Prices of TSLA');
xlabel('Date');
ylabel('Adjusted Close Price');
grid on;
```

```
% Plot the daily returns
figure;
plot(data.Date, data.Daily_returns, '-o');
title('Daily Returns of TSLA');
xlabel('Date');
ylabel('Daily Return');
grid on;

% Compute sample mean, variance, and standard deviation of the daily returns
mean_daily_returns = mean(data.Daily_returns, 'omitnan');
variance_daily_returns = var(data.Daily_returns, 'omitnan');
std_daily_returns = std(data.Daily_returns, 'omitnan');

% Display the computed statistics
fprintf('Sample Mean of Daily Returns: %.6f\n', mean_daily_returns);
fprintf('Sample Variance of Daily Returns: %.6f\n', variance_daily_returns);
fprintf('Sample Standard Deviation of Daily Returns: %.6f\n', std_daily_returns);
```

GEERAL EXPLANATIONS OF THE CODES:

1. **Define File Path:** Specifies the location of the Excel file containing stock data.
2. **Read Data:** Reads the stock data from the Excel file into a MATLAB table and displays the first few rows.
3. **Convert Date Column:** Converts the 'Date' column to datetime format for proper date handling.
4. **Compute Daily Returns:** Calculates daily returns based on adjusted close prices, adding a 'Daily_returns' column to the data.
5. **Save Updated Data:** Writes the updated table, including daily returns, back to the Excel file.
6. **Plot Data:** Creates plots of the daily adjusted close prices and daily returns.
7. **Calculate and Display Statistics:** Computes and prints the mean, variance, and standard deviation of the daily returns, formatted to eight decimal places.

CODE FOR FINDING VARIANCE-COVARIANCE MATRIX:

```
% Define the file paths
file_paths = {
    'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\AMZN_1.xlsx', ...
    'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\COIN_1.xlsx', ...
    'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\EEM_1.xlsx', ...
    'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\MRNA_1.xlsx', ...
    'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\TSLA.xls'
};

% Initialize an empty cell array to store the daily returns
returns = [];

% Loop through each file, read the data, and compute the daily returns
for i = 1:length(file_paths)
    % Read the Excel file into a table
    data = readtable(file_paths{i});

    % Convert the 'Date' column to datetime format if it's not already
    if ~isdatetime(data.Date)
        data.Date = datetime(data.Date, 'InputFormat', 'dd-MM-yyyy');
    end

    % Compute the daily returns
    if any(ismember(data.Properties.VariableNames, 'Daily_returns'))
        if all(isnan(data.Daily_returns))
            daily_returns = [NaN; diff(data.AdjClose) ./ data.AdjClose(1:end-1)];
        else
            daily_returns = data.Daily_returns;
        end
    else
        daily_returns = [NaN; diff(data.AdjClose) ./ data.AdjClose(1:end-1)];
```

```
    % Compute the daily returns
    if any(ismember(data.Properties.VariableNames, 'Daily_returns'))
        if all(isnan(data.Daily_returns))
            daily_returns = [NaN; diff(data.AdjClose) ./ data.AdjClose(1:end-1)];
        else
            daily_returns = data.Daily_returns;
        end
    else
        daily_returns = [NaN; diff(data.AdjClose) ./ data.AdjClose(1:end-1)];
    end

    % Append the daily returns to the returns array
    returns = [returns, daily_returns];
end

% Remove rows with NaN values (these occur due to the initial NaN in diff)
returns = returns(~any(isnan(returns), 2), :);

% Compute the variance-covariance matrix
cov_matrix = cov(returns);

% Display the variance-covariance matrix with eight decimal places
disp('Variance-Covariance Matrix:');
[nRows, nCols] = size(cov_matrix);
for row = 1:nRows
    for col = 1:nCols
        fprintf('%.8f\t', cov_matrix(row, col));
    end
    fprintf('\n');
end
```

EXPLANATIONS OF THE CODES:

1. **Define File Paths:** Specifies the locations of multiple Excel files containing stock data.
2. **Initialize Returns Array:** Creates an empty array to store daily returns.
3. **Loop Through Files:** Reads each Excel file, converts the 'Date' column to datetime format, and computes daily returns based on adjusted close prices.
4. **Handle Daily Returns:** Checks if 'Daily_returns' column exists and computes it if necessary.
5. **Append Returns:** Adds the computed daily returns to the returns array.
6. **Remove NaN Rows:** Eliminates rows with NaN values from the returns array.
7. **Compute Covariance Matrix:** Calculates the variance-covariance matrix of the daily returns.
8. **Display Covariance Matrix:** Prints the variance-covariance matrix with values formatted to eight decimal places.

CODES FOR FINDING LINEAR REGRESSION AND FOR FINDING ALPHA, BETA AND NOISE COEFFICIENTS:

AMZN:

```
% Define the file paths
sp500_file_path = 'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\s&p500_historical_data_past.xlsx';
amzn_file_path = 'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\AMZN_1.xlsx';

% Read the Excel files into tables
sp500_data = readtable(sp500_file_path);
amzn_data = readtable(amzn_file_path);

% Convert the 'Date' columns to datetime format
sp500_data.Date = datetime(sp500_data.Date, 'InputFormat', 'MM/dd/yyyy');
amzn_data.Date = datetime(amzn_data.Date, 'InputFormat', 'MM/dd/yyyy');

% Extract the adjusted close prices
sp500_close = sp500_data.AdjClose; % Assuming 'AdjClose' is the column name for adjusted close prices
amzn_close = amzn_data.AdjClose; % Assuming 'AdjClose' is the column name for adjusted close prices

% Compute daily returns
sp500_returns = [NaN; diff(sp500_close) ./ sp500_close(1:end-1)];
amzn_returns = [NaN; diff(amzn_close) ./ amzn_close(1:end-1)];

% Align the data by dates
[common_dates, idx_sp500, idx_amzn] = intersect(sp500_data.Date, amzn_data.Date);
sp500_returns = sp500_returns(idx_sp500);
amzn_returns = amzn_returns(idx_amzn);

% Remove NaN values resulting from alignment
valid_idx = ~isnan(sp500_returns) & ~isnan(amzn_returns);
sp500_returns = sp500_returns(valid_idx);
amzn_returns = amzn_returns(valid_idx);

% Compute the expected return vector and the variance-covariance matrix
mu = mean([sp500_returns, amzn_returns]); % Vector of expected returns for each asset
V = cov(sp500_returns, amzn_returns); % Covariance matrix

mu_m = mu(1); % Expected return of the market (S&P 500)
mu_a = mu(2); % Expected return of the asset (AMZN)
var_m = V(1, 1); % Variance of the market
var_a = V(2, 2); % Variance of the asset
sig12 = V(1, 2); % Covariance between the market and the asset
```

```
% Estimate the optimal regression coefficients
beta = sig12 / var_m; % Beta coefficient
alpha = mu_a - beta * mu_m; % Alpha coefficient

% Plot the linear regression
figure;
Rm = linspace(min(sp500_returns), max(sp500_returns), 1000);
Ri = alpha + beta * Rm; % Linear regression line

plot(Rm, Ri, 'r', 'LineWidth', 2); % Plot regression line
hold on;
scatter(sp500_returns, amzn_returns, 'filled'); % Scatter plot of the returns
xlabel('S&P 500 Returns');
ylabel('AMZN Returns');
legend('Optimal regression line', 'Data points');
title('Linear Regression of AMZN Returns on S&P 500 Returns');
grid on;
hold off;

% Compute noise coefficients (residuals)
noise = amzn_returns - (alpha + beta * sp500_returns);

% Display alpha, beta, and noise coefficients
disp('Alpha coefficient:');
disp(alpha);
disp('Beta coefficient:');
disp(beta);
disp('Noise coefficients:');
disp(noise);

% Plot the noise coefficients
figure;
plot(noise);
xlabel('Observation');
ylabel('Noise (Residual)');
title('Noise Coefficients (Residuals)');
grid on;
```

COIN:

```
% Define the file paths
sp500_file_path = 'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\s&p500_historical_data_past.xlsx';
coin_file_path = 'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\COIN_1.xlsx';

% Read the Excel files into tables
sp500_data = readtable(sp500_file_path);
coin_data = readtable(coin_file_path);

% Convert the 'Date' columns to datetime format
sp500_data.Date = datetime(sp500_data.Date, 'InputFormat', 'MM/dd/yyyy');
coin_data.Date = datetime(coin_data.Date, 'InputFormat', 'MM/dd/yyyy');

% Extract the adjusted close prices
sp500_close = sp500_data.AdjClose; % Assuming 'AdjClose' is the column name for adjusted close prices
coin_close = coin_data.AdjClose; % Assuming 'AdjClose' is the column name for adjusted close prices

% Compute daily returns
sp500_returns = [NaN; diff(sp500_close) ./ sp500_close(1:end-1)];
coin_returns = [NaN; diff(coin_close) ./ coin_close(1:end-1)];

% Align the data by dates
[common_dates, idx_sp500, idx_coin] = intersect(sp500_data.Date, coin_data.Date);
sp500_returns = sp500_returns(idx_sp500);
coin_returns = coin_returns(idx_coin);

% Remove NaN values resulting from alignment
valid_idx = ~isnan(sp500_returns) & ~isnan(coin_returns);
sp500_returns = sp500_returns(valid_idx);
coin_returns = coin_returns(valid_idx);

% Compute the expected return vector and the variance-covariance matrix
mu = mean([sp500_returns, coin_returns]); % Vector of expected returns for each asset
V = cov(sp500_returns, coin_returns); % Covariance matrix

mu_m = mu(1); % Expected return of the market (S&P 500)
mu_a = mu(2); % Expected return of the asset (COIN)
var_m = V(1, 1); % Variance of the market
var_a = V(2, 2); % Variance of the asset
sig12 = V(1, 2); % Covariance between the market and the asset
```

```
% Estimate the optimal regression coefficients
beta = sig12 / var_m; % Beta coefficient
alpha = mu_a - beta * mu_m; % Alpha coefficient

% Plot the linear regression
figure;
Rm = linspace(min(sp500_returns), max(sp500_returns), 1000);
Ri = alpha + beta * Rm; % Linear regression line

plot(Rm, Ri, 'r', 'LineWidth', 2); % Plot regression line
hold on;
scatter(sp500_returns, coin_returns, 'filled'); % Scatter plot of the returns
xlabel('S&P 500 Returns');
ylabel('COIN Returns');
legend('Optimal regression line', 'Data points');
title('Linear Regression of COIN Returns on S&P 500 Returns');
grid on;
hold off;

% Compute noise coefficients (residuals)
noise = coin_returns - (alpha + beta * sp500_returns);

% Display alpha, beta, and noise coefficients
disp('Alpha coefficient:');
disp(alpha);
disp('Beta coefficient:');
disp(beta);
disp('Noise coefficients:');
disp(noise);

% Plot the noise coefficients
figure;
plot(noise);
xlabel('Observation');
ylabel('Noise (Residual)');
title('Noise Coefficients (Residuals)');
grid on;
```

EEM:

```
% Define the file paths
sp500_file_path = 'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\s&p500_historical_data_past.xlsx';
eem_file_path = 'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\EEM_1.xlsx';

% Read the Excel files into tables
sp500_data = readtable(sp500_file_path);
eem_data = readtable(eem_file_path);

% Convert the 'Date' columns to datetime format
sp500_data.Date = datetime(sp500_data.Date, 'InputFormat', 'MM/dd/yyyy');
eem_data.Date = datetime(eem_data.Date, 'InputFormat', 'MM/dd/yyyy');

% Extract the adjusted close prices
sp500_close = sp500_data.AdjClose; % Assuming 'AdjClose' is the column name for adjusted close prices
eem_close = eem_data.AdjClose; % Assuming 'AdjClose' is the column name for adjusted close prices

% Compute daily returns
sp500_returns = [NaN; diff(sp500_close) ./ sp500_close(1:end-1)];
eem_returns = [NaN; diff(eem_close) ./ eem_close(1:end-1)];

% Align the data by dates
[common_dates, idx_sp500, idx_eem] = intersect(sp500_data.Date, eem_data.Date);
sp500_returns = sp500_returns(idx_sp500);
eem_returns = eem_returns(idx_eem);

% Remove NaN values resulting from alignment
valid_idx = ~isnan(sp500_returns) & ~isnan(eem_returns);
sp500_returns = sp500_returns(valid_idx);
eem_returns = eem_returns(valid_idx);

% Compute the expected return vector and the variance-covariance matrix
mu = mean([sp500_returns, eem_returns]); % Vector of expected returns for each asset
V = cov(sp500_returns, eem_returns); % Covariance matrix

mu_m = mu(1); % Expected return of the market (S&P 500)
mu_a = mu(2); % Expected return of the asset (EEM)
var_m = V(1, 1); % Variance of the market
var_a = V(2, 2); % Variance of the asset
sig12 = V(1, 2); % Covariance between the market and the asset
```

```
% Estimate the optimal regression coefficients
beta = sig12 / var_m; % Beta coefficient
alpha = mu_a - beta * mu_m; % Alpha coefficient

% Plot the linear regression
figure;
Rm = linspace(min(sp500_returns), max(sp500_returns), 1000);
Ri = alpha + beta * Rm; % Linear regression line

plot(Rm, Ri, 'r', 'LineWidth', 2); % Plot regression line
hold on;
scatter(sp500_returns, eem_returns, 'filled'); % Scatter plot of the returns
xlabel('S&P 500 Returns');
ylabel('EEM Returns');
legend('Optimal regression line', 'Data points');
title('Linear Regression of EEM Returns on S&P 500 Returns');
grid on;
hold off;

% Compute noise coefficients (residuals)
noise = eem_returns - (alpha + beta * sp500_returns);

% Display alpha, beta, and noise coefficients
disp('Alpha coefficient:');
disp(alpha);
disp('Beta coefficient:');
disp(beta);
disp('Noise coefficients:');
disp(noise);

% Plot the noise coefficients
figure;
plot(noise);
xlabel('Observation');
ylabel('Noise (Residual)');
title('Noise Coefficients (Residuals)');
grid on;
```

MRNA:

```
% Define the file paths
sp500_file_path = 'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\s&p500_historical_data_past.xlsx';
mrna_file_path = 'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\MRNA_1.xlsx';

% Read the Excel files into tables
sp500_data = readtable(sp500_file_path);
mrna_data = readtable(mrna_file_path);

% Convert the 'Date' columns to datetime format
sp500_data.Date = datetime(sp500_data.Date, 'InputFormat', 'MM/dd/yyyy');
mrna_data.Date = datetime(mrna_data.Date, 'InputFormat', 'MM/dd/yyyy');

% Extract the adjusted close prices
sp500_close = sp500_data.AdjClose; % Assuming 'AdjClose' is the column name for adjusted close prices
mrna_close = mrna_data.AdjClose; % Assuming 'AdjClose' is the column name for adjusted close prices

% Compute daily returns
sp500_returns = [NaN; diff(sp500_close) ./ sp500_close(1:end-1)];
mrna_returns = [NaN; diff(mrna_close) ./ mrna_close(1:end-1)];

% Align the data by dates
[common_dates, idx_sp500, idx_mrna] = intersect(sp500_data.Date, mrna_data.Date);
sp500_returns = sp500_returns(idx_sp500);
mrna_returns = mrna_returns(idx_mrna);

% Remove NaN values resulting from alignment
valid_idx = ~isnan(sp500_returns) & ~isnan(mrna_returns);
sp500_returns = sp500_returns(valid_idx);
mrna_returns = mrna_returns(valid_idx);

% Compute the expected return vector and the variance-covariance matrix
mu = mean([sp500_returns, mrna_returns]); % Vector of expected returns for each asset
V = cov(sp500_returns, mrna_returns); % Covariance matrix

mu_m = mu(1); % Expected return of the market (S&P 500)
mu_a = mu(2); % Expected return of the asset (MRNA)
var_m = V(1, 1); % Variance of the market
var_a = V(2, 2); % Variance of the asset
sig12 = V(1, 2); % Covariance between the market and the asset
```

```
% Estimate the optimal regression coefficients
beta = sig12 / var_m; % Beta coefficient
alpha = mu_a - beta * mu_m; % Alpha coefficient

% Plot the linear regression
figure;
Rm = linspace(min(sp500_returns), max(sp500_returns), 1000);
Ri = alpha + beta * Rm; % Linear regression line

plot(Rm, Ri, 'r', 'LineWidth', 2); % Plot regression line
hold on;
scatter(sp500_returns, mrna_returns, 'filled'); % Scatter plot of the returns
xlabel('S&P 500 Returns');
ylabel('MRNA Returns');
legend('Optimal regression line', 'Data points');
title('Linear Regression of MRNA Returns on S&P 500 Returns');
grid on;
hold off;

% Compute noise coefficients (residuals)
noise = mrna_returns - (alpha + beta * sp500_returns);

% Display alpha, beta, and noise coefficients
disp('Alpha coefficient:');
disp(alpha);
disp('Beta coefficient:');
disp(beta);
disp('Noise coefficients:');
disp(noise);

% Plot the noise coefficients
figure;
plot(noise);
xlabel('Observation');
ylabel('Noise (Residual)');
title('Noise Coefficients (Residuals)');
grid on;
```

TSLA:

```
% Define the file paths
sp500_file_path = 'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\s&p500_historical_data_past.xlsx';
tsla_file_path = 'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\TSLA.xls';

% Read the Excel files into tables
sp500_data = readtable(sp500_file_path);
tsla_data = readtable(tsla_file_path);

% Convert the 'Date' columns to datetime format
sp500_data.Date = datetime(sp500_data.Date, 'InputFormat', 'MM/dd/yyyy');
tsla_data.Date = datetime(tsla_data.Date, 'InputFormat', 'MM/dd/yyyy');

% Extract the adjusted close prices
sp500_close = sp500_data.AdjClose; % Assuming 'AdjClose' is the column name for adjusted close prices
tsla_close = tsla_data.AdjClose; % Assuming 'AdjClose' is the column name for adjusted close prices

% Compute daily returns
sp500_returns = [NaN; diff(sp500_close) ./ sp500_close(1:end-1)];
tsla_returns = [NaN; diff(tsla_close) ./ tsla_close(1:end-1)];

% Align the data by dates
[common_dates, idx_sp500, idx_tsla] = intersect(sp500_data.Date, tsla_data.Date);
sp500_returns = sp500_returns(idx_sp500);
tsla_returns = tsla_returns(idx_tsla);

% Remove NaN values resulting from alignment
valid_idx = ~isnan(sp500_returns) & ~isnan(tsla_returns);
sp500_returns = sp500_returns(valid_idx);
tsla_returns = tsla_returns(valid_idx);

% Compute the expected return vector and the variance-covariance matrix
mu = mean([sp500_returns, tsla_returns]); % Vector of expected returns for each asset
V = cov(sp500_returns, tsla_returns); % Covariance matrix

mu_m = mu(1); % Expected return of the market (S&P 500)
mu_a = mu(2); % Expected return of the asset (TSLA)
var_m = V(1, 1); % Variance of the market
var_a = V(2, 2); % Variance of the asset
sig12 = V(1, 2); % Covariance between the market and the asset
```

```
% Estimate the optimal regression coefficients
beta = sig12 / var_m; % Beta coefficient
alpha = mu_a - beta * mu_m; % Alpha coefficient

% Plot the linear regression
figure;
Rm = linspace(min(sp500_returns), max(sp500_returns), 1000);
Ri = alpha + beta * Rm; % Linear regression line

plot(Rm, Ri, 'r', 'LineWidth', 2); % Plot regression line
hold on;
scatter(sp500_returns, tsla_returns, 'filled'); % Scatter plot of the returns
xlabel('S&P 500 Returns');
ylabel('TSLA Returns');
legend('Optimal regression line', 'Data points');
title('Linear Regression of TSLA Returns on S&P 500 Returns');
grid on;
hold off;

% Compute noise coefficients (residuals)
noise = tsla_returns - (alpha + beta * sp500_returns);

% Display alpha, beta, and noise coefficients
disp('Alpha coefficient:');
disp(alpha);
disp('Beta coefficient:');
disp(beta);
disp('Noise coefficients:');
disp(noise);

% Plot the noise coefficients
figure;
plot(noise);
xlabel('Observation');
ylabel('Noise (Residual)');
title('Noise Coefficients (Residuals)');
grid on;
```

GENERAL EXPLANATIONS FOR THE CODE:

1. **Define File Paths:** Specifies the locations of the S&P 500 and TSLA Excel files.
2. **Read Data:** Reads the stock data from the Excel files into tables.
3. **Convert Date Columns:** Converts the 'Date' columns to datetime format.
4. **Extract Adjusted Close Prices:** Extracts the adjusted close prices for both S&P 500 and TSLA.
5. **Compute Daily Returns:** Calculates daily returns for both indices.
6. **Align Data by Dates:** Aligns the data based on common dates.
7. **Remove NaN Values:** Removes rows with NaN values after alignment.
8. **Compute Expected Returns and Covariance Matrix:** Calculates the mean returns and the variance-covariance matrix.
9. **Estimate Regression Coefficients:** Computes the alpha and beta coefficients for the regression.
10. **Plot Linear Regression:** Plots the regression line and scatter plot of the returns.

11. **Compute Residuals:** Calculates the noise coefficients (residuals).
12. **Display Coefficients and Residuals:** Displays the alpha, beta, and noise coefficients.
13. **Plot Residuals:** Plots the noise coefficients to visualize the residuals.

CODES FOR PLOTTING SCATTER PLOTS AND HISTOGRAM:

AMZN:

```
% Load the data from Excel files
amzn_data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\AMZN_1.xlsx');
sp500_data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\s&p500_historical_data_past.xlsx');

% Ensure the data is sorted by date
amzn_data = sortrows(amzn_data, 'Date');
sp500_data = sortrows(sp500_data, 'Date');

% Compute daily returns
amzn_data.Return = [NaN; diff(amzn_data.Close) ./ amzn_data.Close(1:end-1)];
sp500_data.Return = [NaN; diff(sp500_data.Close) ./ sp500_data.Close(1:end-1)];

% Drop the NaN values resulting from the shift
amzn_returns = amzn_data.Return(~isnan(amzn_data.Return));
sp500_returns = sp500_data.Return(~isnan(sp500_data.Return));

% Create histograms in a single figure
figure;
hold on;

% Histogram for AMZN returns
histogram(amzn_returns, 50, 'EdgeColor', 'k', 'FaceAlpha', 0.5);
% Histogram for S&P 500 returns
histogram(sp500_returns, 50, 'EdgeColor', 'r', 'FaceAlpha', 0.5);

title('Comparison of Histograms: AMZN vs S&P 500 Returns');
xlabel('Returns');
ylabel('Frequency');
legend('AMZN Returns', 'S&P 500 Returns');
hold off;


% Create scatter plot
figure;
scatter(amzn_returns, sp500_returns, 'filled', 'MarkerFaceAlpha', 0.5);
title('Scatter Plot of AMZN vs S&P 500 Returns');
xlabel('AMZN Returns');
ylabel('S&P 500 Returns');
grid on;
```

COIN:

```
% Load the data from Excel files
coin_data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\COIN_1.xlsx');
sp500_data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\s&p500_historical_data_past.xlsx');

% Ensure the data is sorted by date
coin_data = sortrows(coin_data, 'Date');
sp500_data = sortrows(sp500_data, 'Date');

% Compute daily returns
coin_data.Return = [NaN; diff(coin_data.Close) ./ coin_data.Close(1:end-1)];
sp500_data.Return = [NaN; diff(sp500_data.Close) ./ sp500_data.Close(1:end-1)];

% Drop the NaN values resulting from the shift
coin_returns = coin_data.Return(~isnan(coin_data.Return));
sp500_returns = sp500_data.Return(~isnan(sp500_data.Return));

% Create histograms in a single figure
figure;
hold on;

% Histogram for COIN returns
histogram(coin_returns, 50, 'EdgeColor', 'k', 'FaceAlpha', 0.5);
% Histogram for S&P 500 returns
histogram(sp500_returns, 50, 'EdgeColor', 'r', 'FaceAlpha', 0.5);

title('Comparison of Histograms: COIN vs S&P 500 Returns');
xlabel('Returns');
ylabel('Frequency');
legend('COIN Returns', 'S&P 500 Returns');
hold off;

% Create scatter plot
figure;
scatter(coin_returns, sp500_returns, 'filled', 'MarkerFaceAlpha', 0.5);
title('Scatter Plot of COIN vs S&P 500 Returns');
xlabel('COIN Returns');
ylabel('S&P 500 Returns');
grid on;
```

EEM:

```
% Load the data from Excel files
eem_data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\EEM_1.xlsx');
sp500_data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\s&p500_historical_data_past.xlsx');

% Ensure the data is sorted by date
eem_data = sortrows(eem_data, 'Date');
sp500_data = sortrows(sp500_data, 'Date');

% Compute daily returns
eem_data.Return = [NaN; diff(eem_data.Close) ./ eem_data.Close(1:end-1)];
sp500_data.Return = [NaN; diff(sp500_data.Close) ./ sp500_data.Close(1:end-1)];

% Drop the NaN values resulting from the shift
eem_returns = eem_data.Return(~isnan(eem_data.Return));
sp500_returns = sp500_data.Return(~isnan(sp500_data.Return));

% Create histograms in a single figure
figure;
hold on;

% Histogram for EEM returns
histogram(eem_returns, 50, 'EdgeColor', 'k', 'FaceAlpha', 0.5);
% Histogram for S&P 500 returns
histogram(sp500_returns, 50, 'EdgeColor', 'r', 'FaceAlpha', 0.5);

title('Comparison of Histograms: EEM vs S&P 500 Returns');
xlabel('Returns');
ylabel('Frequency');
legend('EEM Returns', 'S&P 500 Returns');
hold off;

% Create scatter plot
figure;
scatter(eem_returns, sp500_returns, 'filled', 'MarkerFaceAlpha', 0.5);
title('Scatter Plot of EEM vs S&P 500 Returns');
xlabel('EEM Returns');
ylabel('S&P 500 Returns');
grid on;
```

MRNA:

```
% Load the data from Excel files
mrna_data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\MRNA_1.xlsx');
sp500_data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\s&p500_historical_data_past.xlsx');

% Ensure the data is sorted by date
mrna_data = sortrows(mrna_data, 'Date');
sp500_data = sortrows(sp500_data, 'Date');

% Compute daily returns
mrna_data.Return = [NaN; diff(mrna_data.Close) ./ mrna_data.Close(1:end-1)];
sp500_data.Return = [NaN; diff(sp500_data.Close) ./ sp500_data.Close(1:end-1)];

% Drop the NaN values resulting from the shift
mrna_returns = mrna_data.Return(~isnan(mrna_data.Return));
sp500_returns = sp500_data.Return(~isnan(sp500_data.Return));

% Create histograms in a single figure
figure;
hold on;

% Histogram for MRNA returns
histogram(mrna_returns, 50, 'EdgeColor', 'k', 'FaceAlpha', 0.5);
% Histogram for S&P 500 returns
histogram(sp500_returns, 50, 'EdgeColor', 'r', 'FaceAlpha', 0.5);

title('Comparison of Histograms: MRNA vs S&P 500 Returns');
xlabel('Returns');
ylabel('Frequency');
legend('MRNA Returns', 'S&P 500 Returns');
hold off;

% Create scatter plot
figure;
scatter(mrna_returns, sp500_returns, 'filled', 'MarkerFaceAlpha', 0.5);
title('Scatter Plot of MRNA vs S&P 500 Returns');
xlabel('MRNA Returns');
ylabel('S&P 500 Returns');
grid on;
```

TSLA:

```
% Load the data from Excel files
tsla_data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\TSLA.xls');
sp500_data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\s&p500_historical_data_past.xlsx');

% Ensure the data is sorted by date
tsla_data = sortrows(tsla_data, 'Date');
sp500_data = sortrows(sp500_data, 'Date');

% Compute daily returns
tsla_data.Return = [NaN; diff(tsla_data.Close) ./ tsla_data.Close(1:end-1)];
sp500_data.Return = [NaN; diff(sp500_data.Close) ./ sp500_data.Close(1:end-1)];

% Drop the NaN values resulting from the shift
tsla_returns = tsla_data.Return(~isnan(tsla_data.Return));
sp500_returns = sp500_data.Return(~isnan(sp500_data.Return));

% Create histograms in a single figure
figure;
hold on;

% Histogram for TSLA returns
histogram(tsla_returns, 50, 'EdgeColor', 'k', 'FaceAlpha', 0.5);
% Histogram for S&P 500 returns
histogram(sp500_returns, 50, 'EdgeColor', 'r', 'FaceAlpha', 0.5);

title('Comparison of Histograms: TSLA vs S&P 500 Returns');
xlabel('Returns');
ylabel('Frequency');
legend('TSLA Returns', 'S&P 500 Returns');
hold off;

% Create scatter plot
figure;
scatter(tsla_returns, sp500_returns, 'filled', 'MarkerFaceAlpha', 0.5);
title('Scatter Plot of TSLA vs S&P 500 Returns');
xlabel('TSLA Returns');
ylabel('S&P 500 Returns');
grid on;
```

GENERAL EXPLANATIONS FOR THE CODES:

1. **Load Data:** Reads TSLA and S&P 500 data from Excel files into tables.
2. **Sort Data by Date:** Ensures the data is sorted chronologically by date.
3. **Compute Daily Returns:** Calculates daily returns for TSLA and S&P 500.
4. **Remove NaN Values:** Removes rows with NaN values resulting from return calculations.
5. **Create Histograms:** Plots histograms of TSLA and S&P 500 returns in a single figure for comparison.
6. **Create Scatter Plot:** Plots a scatter plot to visualize the relationship between TSLA and S&P 500 returns.

CODES FOR FINDING EFFICIENT FRONTIER AND UTILITY OPTIMAL PORTFOLIO:

```
% Load data from Excel files
amzn_data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\AMZN_1.xlsx');
coin_data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\COIN_1.xlsx');
eem_data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\EEM_1.xlsx');
mrna_data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\MRNA_1.xlsx');
tsla_data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\TSLA.xls');

% Extract the adjusted closing prices
amzn_prices = amzn_data.AdjClose;
coin_prices = coin_data.AdjClose;
eem_prices = eem_data.AdjClose;
mrna_prices = mrna_data.AdjClose;
tsla_prices = tsla_data.AdjClose;

% Compute daily returns
amzn_returns = diff(log(amzn_prices));
coin_returns = diff(log(coin_prices));
eem_returns = diff(log(eem_prices));
mrna_returns = diff(log(mrna_prices));
tsla_returns = diff(log(tsla_prices));

% Combine returns into a matrix
returns = [amzn_returns, coin_returns, eem_returns, mrna_returns, tsla_returns];

% Compute annualized mean returns and covariance matrix
mean_returns = mean(returns) * 252;
cov_matrix = cov(returns) * 252;

% Number of assets
num_assets = size(returns, 2);
```

```
% Generate random portfolio weights
num_portfolios = 10000;
results = zeros(3, num_portfolios);
for i = 1:num_portfolios
    weights = rand(num_assets, 1);
    weights = weights / sum(weights);

    portfolio_return = dot(weights, mean_returns);
    portfolio_std_dev = sqrt(weights' * cov_matrix * weights);

    results(1, i) = portfolio_return;
    results(2, i) = portfolio_std_dev;
    results(3, i) = portfolio_return / portfolio_std_dev;
end

% Extract the results
portfolio_returns = results(1, :);
portfolio_std_devs = results(2, :);
sharpe_ratios = results(3, :);

% Identify the portfolio with the maximum Sharpe ratio
[max_sharpe_ratio, max_sharpe_idx] = max(sharpe_ratios);
max_sharpe_weights = rand(num_assets, 1);
max_sharpe_weights = max_sharpe_weights / sum(max_sharpe_weights);

% Identify the minimum variance portfolio
[min_var_std_dev, min_var_idx] = min(portfolio_std_devs);
min_var_weights = rand(num_assets, 1);
min_var_weights = min_var_weights / sum(min_var_weights);
```

```
% Plot the efficient frontier
figure;
scatter(portfolio_std_devs, portfolio_returns, 10, sharpe_ratios, 'filled');
colorbar;
xlabel('Portfolio Standard Deviation');
ylabel('Portfolio Return');
title('Efficient Frontier');
hold on;
scatter(portfolio_std_devs(min_var_idx), portfolio_returns(min_var_idx), 100, 'r', 'filled');
scatter(portfolio_std_devs(max_sharpe_idx), portfolio_returns(max_sharpe_idx), 100, 'g', 'filled');
legend('Portfolios', 'Minimum Variance Portfolio', 'Maximum Sharpe Ratio Portfolio');
hold off;

%Utility

% Parameters
alpha = 3; % Risk aversion coefficient

% Generate random portfolio weights
num_portfolios = 10000;
results = zeros(4, num_portfolios); % Adding an additional row for utility values
for i = 1:num_portfolios
    weights = rand(num_assets, 1);
    weights = weights / sum(weights);

    portfolio_return = dot(weights, mean_returns);
    portfolio_std_dev = sqrt(weights' * cov_matrix * weights);

    results(1, i) = portfolio_return;
    results(2, i) = portfolio_std_dev;
    results(3, i) = portfolio_return / portfolio_std_dev;
    results(4, i) = portfolio_return - alpha * (portfolio_std_dev ^ 2); % Utility function 1
end
```

```
% Utility function 1: max u(μ, σ) = μ - ασ2
[max_utility1, max_utility1_idx] = max(results(4, :));
max_utility1_weights = rand(num_assets, 1);
max_utility1_weights = max_utility1_weights / sum(max_utility1_weights);

% Utility function 2: max u(μ, σ) = μ / σ2
utility2_values = results(1, :) ./ results(2, :).^2;
[max_utility2, max_utility2_idx] = max(utility2_values);
max_utility2_weights = rand(num_assets, 1);
max_utility2_weights = max_utility2_weights / sum(max_utility2_weights);

% Plot the efficient frontier
figure;
scatter(results(2, :), results(1, :), 10, results(3, :), 'filled');
colorbar;
xlabel('Portfolio Standard Deviation');
ylabel('Portfolio Return');
title('Efficient Frontier with Utility Function Optimal Portfolios');
hold on;
scatter(results(2, min_var_idx), results(1, min_var_idx), 100, 'r', 'filled'); % Minimum Variance Portfolio
scatter(results(2, max_sharpe_idx), results(1, max_sharpe_idx), 100, 'g', 'filled'); % Maximum Sharpe Ratio Portfolio
scatter(results(2, max_utility1_idx), results(1, max_utility1_idx), 100, 'm', 'filled'); % Utility Function 1 Optimal Portfolio
scatter(results(2, max_utility2_idx), results(1, max_utility2_idx), 100, 'c', 'filled'); % Utility Function 2 Optimal Portfolio
legend('Portfolios', 'Minimum Variance Portfolio', 'Maximum Sharpe Ratio Portfolio', 'Max Utility 1 Portfolio', 'Max Utility 2 Portfolio');
hold off;
```

EXPLANATIONS FOR THE CODES:

1. **oad Data:** Reads historical stock data from Excel files for AMZN, COIN, EEM, MRNA, and TSLA.
2. **Extract Prices:** Extracts the adjusted closing prices for each stock.
3. **Compute Daily Returns:** Calculates daily logarithmic returns for each stock.

4. **Combine Returns:** Combines the daily returns into a matrix.
5. **Annualized Metrics:** Computes annualized mean returns and the covariance matrix.
6. **Generate Portfolios:** Generates random portfolio weights and calculates the expected return, standard deviation, and Sharpe ratio for each portfolio.
7. **Identify Optimal Portfolios:** Identifies the portfolios with the maximum Sharpe ratio and minimum variance.
8. **Plot Efficient Frontier:** Plots the efficient frontier and highlights the minimum variance and maximum Sharpe ratio portfolios.
9. **Utility Functions:** Calculates portfolios maximizing two utility functions considering risk aversion, and identifies optimal portfolios for each utility function.
10. **Plot Utility Optimal Portfolios:** Plots the efficient frontier again, highlighting the utility function optimal portfolios along with the previously identified portfolios.

CODES FOR COMPARING THE PORTFOLIO WITH THE FUTURE INDEX:

```
% Load the S&P 500 historical data
data = readtable('C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\s_p500_historical_data.xlsx');

% Extract relevant columns
dates = data.Date;
close_prices = data.Close;

% Calculate daily returns for S&P 500
sp500_returns = diff(close_prices) ./ close_prices(1:end-1);

% Define initial investment amount
initial_investment = 100000; % Example amount

% Portfolio allocations (20% in each asset)
allocations = [0.20, 0.20, 0.20, 0.20, 0.20];

% Historical data for the constructed portfolio (Example data)
% Replace this with the actual data for TSLA, COIN, AMZN, EEM, MRNA
tsla_returns = randn(length(sp500_returns), 1) * 0.02;
coin_returns = randn(length(sp500_returns), 1) * 0.03;
amzn_returns = randn(length(sp500_returns), 1) * 0.01;
eem_returns = randn(length(sp500_returns), 1) * 0.015;
mrna_returns = randn(length(sp500_returns), 1) * 0.025;

% Combine the returns
portfolio_returns = [tsla_returns, coin_returns, amzn_returns, eem_returns, mrna_returns];

% Calculate weighted portfolio returns
weighted_portfolio_returns = portfolio_returns * allocations';

% Initialize investment values
sp500_investment = initial_investment;
portfolio_investment = initial_investment;
```

```
% Store investment values for each time point
sp500_investment_values = [];
portfolio_investment_values = [];

% Calculate investment values at each 20 trading day interval
interval = 20;
for i = 1:interval:length(sp500_returns)
    % Update investment values
    sp500_investment = sp500_investment * (1 + sp500_returns(i));
    portfolio_investment = portfolio_investment * (1 + weighted_portfolio_returns(i));

    % Store the values
    sp500_investment_values = [sp500_investment_values; sp500_investment];
    portfolio_investment_values = [portfolio_investment_values; portfolio_investment];
end

% Plot the investment values
time_points = (1:interval:length(sp500_returns)) * interval;
figure;
plot(time_points, sp500_investment_values, '-o', 'DisplayName', 'S&P 500');
hold on;
plot(time_points, portfolio_investment_values, '-x', 'DisplayName', 'Constructed Portfolio');
xlabel('Trading Days');
ylabel('Investment Value');
title('Investment Performance Comparison');
legend;
grid on;
```

EXPLANATIONS OF THE CODES IN DETAIL:

1. **Load Historical Data:**
    - The code reads the S&P 500 historical data from an Excel file and extracts the relevant columns for dates and closing prices.
2. **Calculate Daily Returns:**
    - It computes the daily returns for the S&P 500 using the difference between consecutive closing prices.
3. **Define Initial Investment:**
    - An initial investment amount (e.g., $100,000) is specified.
4. **Portfolio Allocations:**
    - The portfolio is allocated equally among five assets, with 20% of the investment in each asset.
5. **Historical Returns Data:**
    - Example historical return data for TSLA, COIN, AMZN, EEM, and MRNA is generated. In a real scenario, this should be replaced with actual return data for these assets.
6. **Combine Returns:**
    - The individual returns of the five assets are combined into a single matrix.
7. **Calculate Weighted Portfolio Returns:**

- The code computes the weighted returns of the portfolio by multiplying the individual asset returns with their respective allocations.

8. **Initialize Investment Values:**
    - Both the S&P 500 and the constructed portfolio start with the initial investment amount.
9. **Store Investment Values:**
    - Arrays are initialized to store the investment values at each time point.
10. **Calculate Investment Values:**
    - For every 20 trading days, the investment values are updated based on the returns, and the values are stored in the respective arrays.
11. **Plot Investment Values:**
    - The investment values of both the S&P 500 and the constructed portfolio are plotted over time to compare their performance.

This process allows you to visualize how an equally weighted portfolio of five assets would perform compared to an investment in the S&P 500 over a given period. The use of 20 trading day intervals provides a clearer long-term view of investment performance.

CODES FOR CHECKING THE ASSET ASSET PROTECTION WOULD HAVE BEEN USEFUL:

```
% Import the historical price data for each asset from XLSX files
assets = {'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\TSLA.xls', ...
          'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\AMZN_1.xlsx', ...
          'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\COIN_1.xlsx', ...
          'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\EEM_1.xlsx', ...
          'C:\Users\MAYUKH\OneDrive\Desktop\FPA Dissertation\MRNA_1.xlsx'};
num_assets = length(assets);
prices = cell(num_assets, 1);
dates_all = cell(num_assets, 1);
stats = struct();

for i = 1:num_assets
    data = readtable(assets{i});
    dates_all{i} = datetime(data.Date, 'InputFormat', 'yyyy-MM-dd');
    prices{i} = data.AdjClose;

    % Calculate daily log returns
    log_returns = diff(log(prices{i}));

    % Calculate mean and variance of log returns
    mean_return = mean(log_returns);
    var_return = var(log_returns);

    % Annualize the mean and variance
    annualized_mean = mean_return * 252; % 252 trading days in a year
    annualized_var = var_return * 252;

    % Store results
    stats(i).mean = annualized_mean;
    stats(i).variance = annualized_var;
    stats(i).std_dev = sqrt(annualized_var);
end
```

```
% Compute the option prices for each asset
strike_price = 130;
expiry_time = 1/2;
risk_free_rate = 0;

for i = 1:num_assets
    current_price = prices{i}(end);
    volatility = stats(i).std_dev;

    [call_price, put_price] = blsprice(current_price, strike_price, risk_free_rate, expiry_time, volatility);

    fprintf('Asset: %s\n', assets{i});
    fprintf('Call Price: %.2f\n', call_price);
    fprintf('Put Price: %.2f\n\n', put_price);
end

% Example: Long Call and Short Put
% This strategy profits if the asset price increases
for i = 1:num_assets
    current_price = prices{i}(end);
    volatility = stats(i).std_dev;

    [call_price, put_price] = blsprice(current_price, strike_price, risk_free_rate, expiry_time, volatility);

    % Simulate future price scenario
    simulated_price = current_price * exp(stats(i).mean * expiry_time); % Simplified simulation

    % Calculate profits
    profit_call = max(simulated_price - strike_price, 0) - call_price;
    profit_put = put_price - max(strike_price - simulated_price, 0);
    total_profit = profit_call + profit_put;

    fprintf('Asset: %s\n', assets{i});
    fprintf('Total Profit (Long Call, Short Put): %.2f\n\n', total_profit);
end
```

```
% Function to compute Black-Scholes price
function [call, put] = blsprice(S, E, r, T, sigma)
    d1 = (log(S/E) + (r + 0.5*sigma^2)*T) / (sigma*sqrt(T));
    d2 = d1 - sigma*sqrt(T);

    call = S * normcdf(d1) - E * exp(-r*T) * normcdf(d2);
    put = E * exp(-r*T) * normcdf(-d2) - S * normcdf(-d1);
end
```

EXPLANATION OF THE CODES:

Importing Data and Calculating Statistics

1. **Initialize Assets and Storage:**
    - A cell array **assets** contains the file paths for the Excel files of five assets (TSLA, AMZN, COIN, EEM, MRNA).
    - The number of assets is determined, and cell arrays **prices** and **dates_all** are initialized to store adjusted close prices and dates, respectively.
    - A structure **stats** is initialized to store statistical data for each asset.
2. **Read Data and Calculate Returns:**

- A loop iterates through each asset file:
    - Reads the data from the Excel file.
    - Converts the 'Date' column to datetime format.
    - Extracts the adjusted close prices.
    - Computes the daily logarithmic returns.
    - Calculates the mean and variance of the daily log returns.
    - Annualizes the mean and variance using 252 trading days.
    - Stores the annualized mean, variance, and standard deviation in the **stats** structure.

Option Pricing Using Black-Scholes Model

3. **Define Option Parameters:**
    - Defines the strike price, expiry time (in years), and risk-free rate.
4. **Calculate Option Prices:**
    - A loop iterates through each asset:
        - Retrieves the current price of the asset.
        - Uses the annualized standard deviation as the volatility.
        - Calls the custom **blsprice** function to calculate the call and put option prices using the Black-Scholes formula.
        - Prints the calculated call and put prices for each asset.

Example Strategy: Long Call and Short Put

5. **Strategy Simulation:**
    - A loop iterates through each asset to simulate a long call and short put strategy:
        - Retrieves the current price and volatility.
        - Calls the **blsprice** function to get the call and put prices.
        - Simulates a future asset price using a simplified exponential growth model based on the annualized mean return.
        - Calculates the profit from the long call and short put positions:
            - **Long Call Profit:** If the simulated price is above the strike price, the profit is the difference minus the call price. Otherwise, the profit is negative the call price.
            - **Short Put Profit:** If the simulated price is below the strike price, the profit is the put price minus the difference. Otherwise, the profit is the put price.
            - The total profit is the sum of the profits from the long call and short put positions.

- Prints the total profit for each asset.

Custom Black-Scholes Function

6. **Black-Scholes Pricing Function (blsprice):**
    - The function **blsprice** calculates the call and put prices using the Black-Scholes formula:
        - Computes d_1 and d_2 based on the inputs (current price S, strike price E, risk-free rate r, time to expiry T, and volatility σ).
        - Uses the cumulative distribution function **normcdf** to calculate the probabilities.
        - Computes the call and put prices based on the Black-Scholes equations.

Summary

- The code processes historical stock data for five assets, calculates statistical measures (annualized mean and variance of returns), and uses the Black-Scholes model to price call and put options.
- It also simulates a trading strategy involving a long call and a short put to estimate potential profits under simplified assumptions. The results are printed for each asset.
- The custom **blsprice** function encapsulates the Black-Scholes pricing logic, facilitating option price calculations within the main script.

REFERENCES

Reference Number	Description	URL
1	Tesla Inc. (TSLA) [10-5-2022 to 10-5-2023]	https://www.tesla.com
2	Coinbase Global Inc. (COIN) [10-5-2022 to 10-5-2023]	https://www.coinbase.com
3	Amazon.com Inc. (AMZN) [10-5-2022 to 10-5-2023]	https://www.amazon.com
4	iShares MSCI Emerging Markets ETF (EEM) [10-5-2022 to 10-5-2023]	https://www.ishares.com/us/products/239726/
5	Moderna Inc. (MRNA) [10-5-2022 to 10-5-2023]	https://www.modernatx.com