Introduction

This report showcases the results of a python program that analyzed six stocks via the stochastic model for predicting future asset prices utilizing Brownian motion. The program retrieves stock data from yahoo finance and implements system of SDEs to simulate future price trajectories.

I selected the following stocks for my analysis (a):

1. Apple (AAPL)
2. Google (GOOGL)
3. Microsoft (MSFT)
4. Amazon (AMZN)
5. Tesla (TSLA)
6. Meta (META)

Daily stock data was collected for a one-year period using the yfinance library in python:

```
tickers = ['AAPL', 'GOOGL', 'MSFT', 'AMZN', 'TSLA', 'META']

start_date = '2023-12-01'

end_date = '2024-12-01'

data = yf.download(tickers, start=start_date, end=end_date)['Adj Close']
```

Programming Debrief

The drift and volatility statistics were calculated using the normalized log returns of the securities, as required in the guidelines. The covolatility matrix was also computed to capture the correlations between different stocks (b):

```
log_returns = np.log(data / data.shift(1))

mu = log_returns.mean() / hn

sigma_squared = ((log_returns - log_returns.mean()) ** 2).sum() / (n * hn)

Sigma = log_returns.cov() / hn
```

Cholesky decomposition was used to get the square root of the covolatility matrix to ensure the correlation is accurate (c):

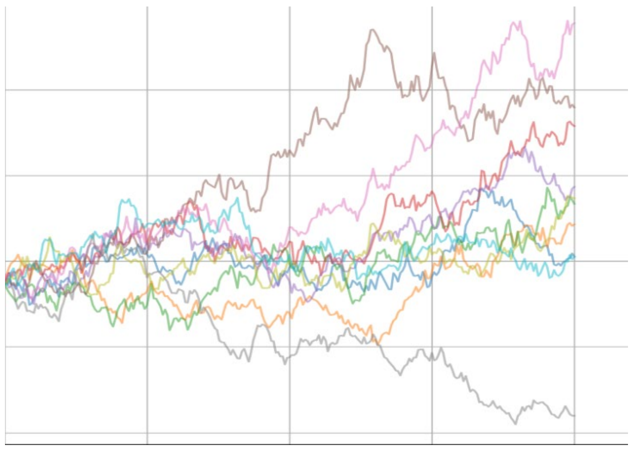
```
sqrt_Sigma = cholesky(Sigma, lower=True)
```

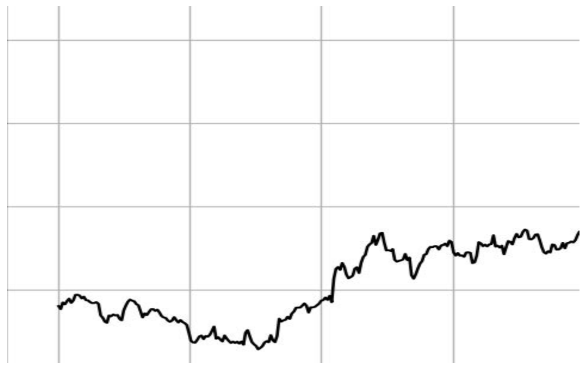
I then used the Euler method to simulate 10 trajectories of the SDE system for each stock. This implementation was simple in python and works great for Ito processes using recursion. The simulation used a daily time step of 1/252 of a year and ran for one year, generating our paths for the trajectories (d):

```
def simulate_gbm(S0, mu, sigma, Sigma, T, dt, num_paths):
    num_steps = int(T / dt)
    paths = np.zeros((num_paths, N, num_steps + 1))
    paths[:, :, 0] = S0

    for i in range(1, num_steps + 1):
        Z = np.random.normal(0, 1, (num_paths, N))
        dW = np.sqrt(dt) * (sqrt_Sigma @ Z.T).T
        paths[:, :, i] = paths[:, :, i-1] * np.exp((mu - 0.5 * sigma) * dt + dW)

    return paths
```

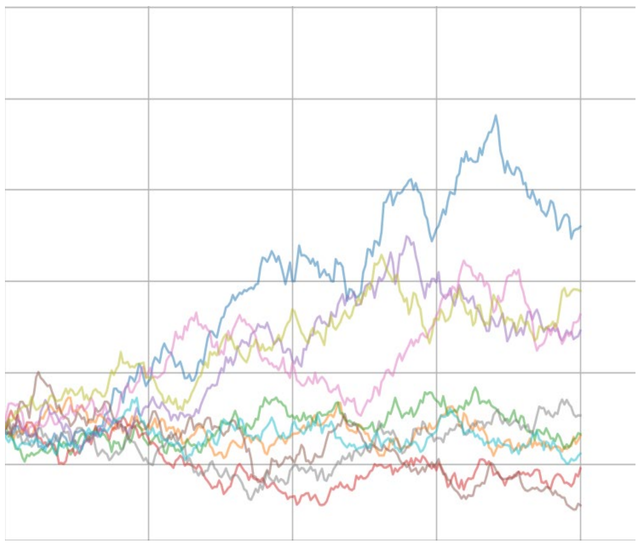
The trajectories were then graphed, with the stock price on the vertical axis and the time course over one year on the horizontal axis. The graphs to the right of those are the historical data.

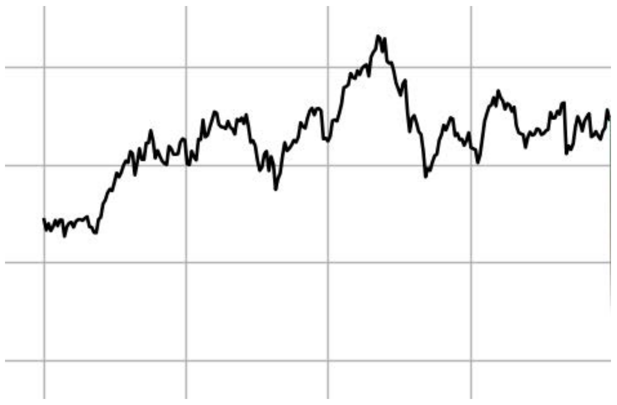
Apple:



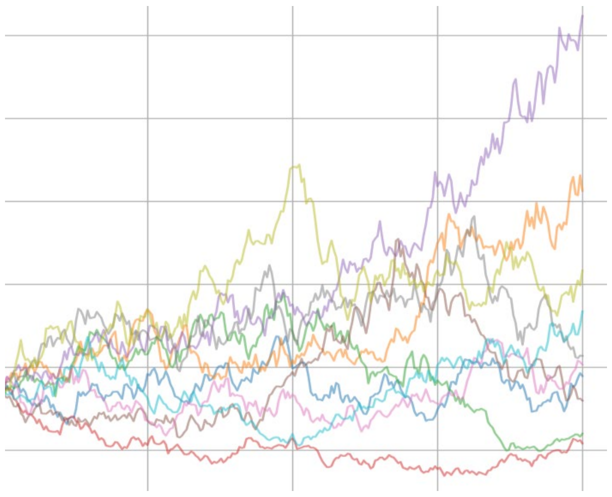


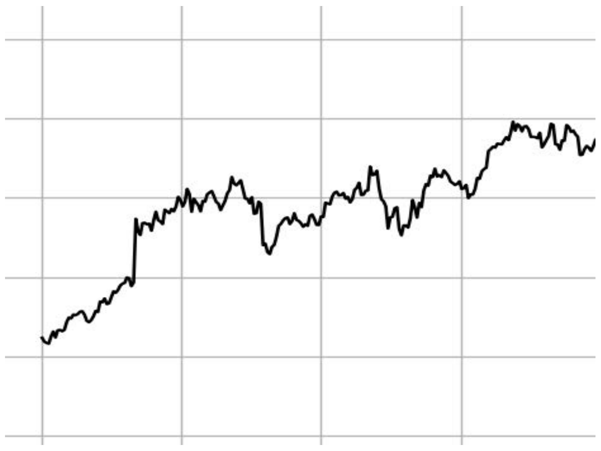
Microsoft:



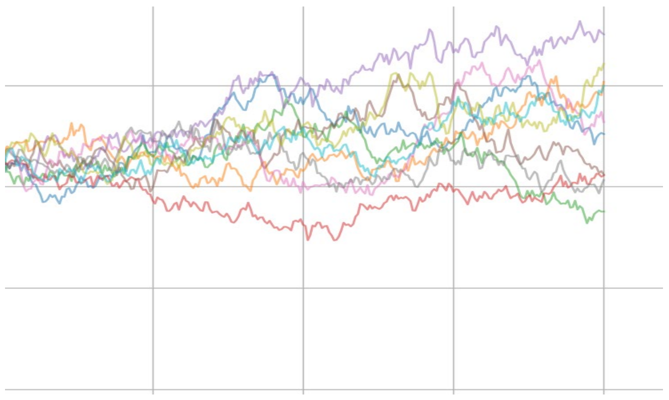


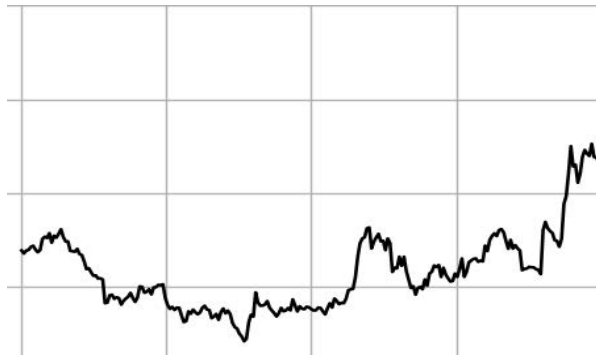
Meta:



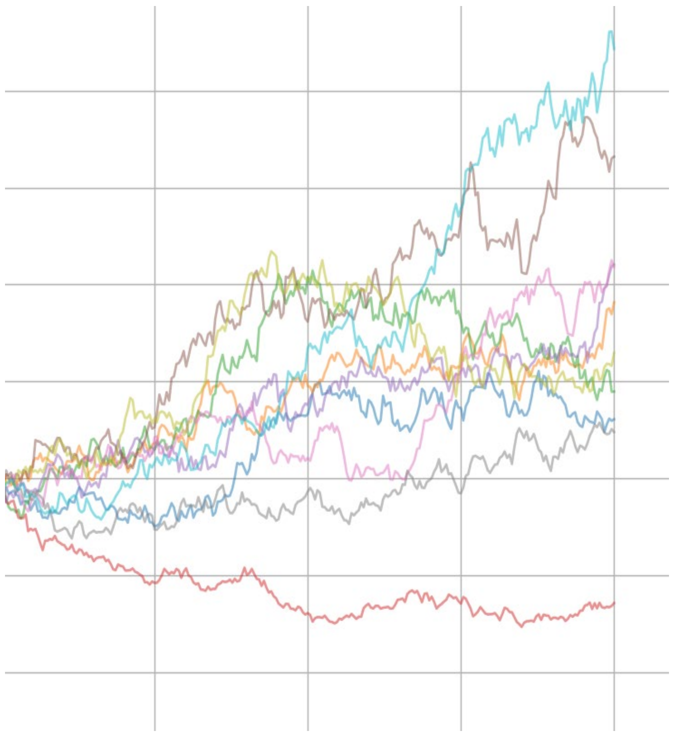


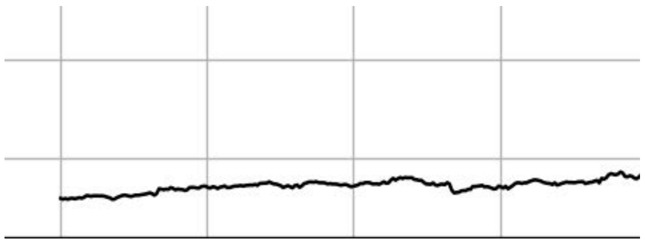
Tesla:



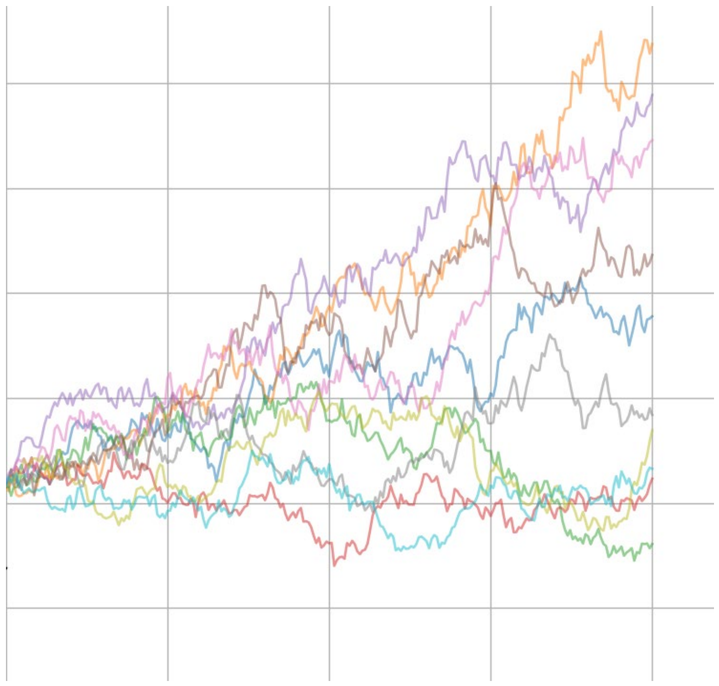


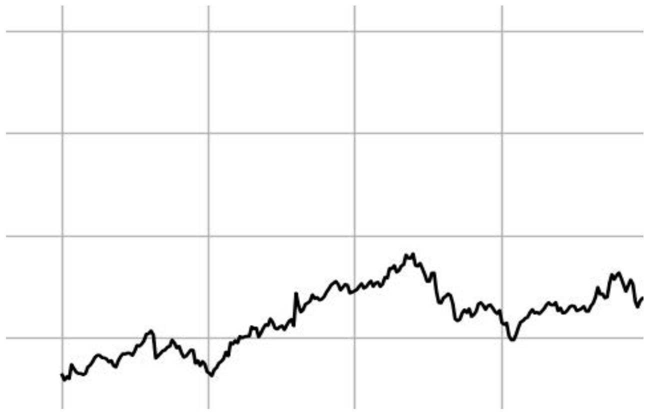
Amazon:





Google:





Observing the Graphs

- The simulated paths show wildly different levels of volatility for different stocks, reflecting their historical price movements. One example is how Apple shows wider price ranges in its simulated paths compared to more stable stocks like Amazon.
- Some of the paths follow past data trends, while others diverge, showcasing the somewhat unpredictable nature of those stocks. For example, with Amazon and Apple paths show notable growth while others indicate potential declines.
- The use of the covolatility matrix seemed to ensure that the stock paths reflect their historical correlations. I noted specifically similar movements of related stocks like Google and Meta, where the performance of each corporation may affect the other.
- Perhaps stocks with wider 'spreads', such as Google, indicate higher uncertainty and potential for both substantial gain or loss.
- Some stocks show a tendency in the long run to maintain their average value based on past data, notably in stocks with recent changes like Tesla after the election.

Conclusion/Some Reflections:

- The portfolio contains only tech stocks, which are not necessarily diverse, but provide interesting data with high volatility but potential for high growth.
- Long-term investments, as expected, appear to be safer and more stable in comparison to short term where the volatility in a smaller period can present a scenario of more risk.
- The model is limited not only by the lack of capability to simulate real world events that are the largest determinants of stock performance, but also the calculations themselves, specifically:
    - The assumption of log-normal distribution
    - Constant drift and volatility
- All things considered however, the results provide insight into the potential future behavior of these tech stocks utilizing Brownian motion.
- Utilizing only tech stocks I reached an interesting conclusion. I believe more than six stocks would allow this program to be more usable in real world decision making, as it would provide a comprehensive overview of a market as a whole rather than strictly analyzing portfolio performance, as obviously machine learning would need to be implemented for modern analysis.