Modelling Stock Market Trends As Brownian Motion Using Monte Carlo Simulations

PHY-412, Computational Physics
Course Instructor: Dr. Sunil Pratap Singh

Anirudh Kalla, 18037

Smit Kanawade, 18267

Department of Physics
IISER Bhopal

Preface

The choice of this research project originally stemmed from our passion for utilizing computational algorithms and techniques we learnt in the course to understand quantitative behaviour of physical phenomenon that can be modeled like physical systems. Modeling the share prices in a given stock market index came as a natural choice, since it provides all the basic quantitative nuances one would expect from theoretical descriptions of physical phenomenon and systems and yet does not exceed complexity in a way that renders our computational algorithms unviable/unusable. As computational and programming resources increase manifold, utilizing our algorithms (Monte Carlo, in this case) to tackle real life physics problems becomes increasingly lucrative due to higher precision and accuracy. We strive to not only use, but to create powerful tools that push the boundaries of computational physics using intelligent inputs and utilizing intuition wherever possible, giving us a definite edge over raw brute-force driven approaches.

The completion of this assignment gives us much pleasure. We would like to show our gratitude to Dr. Sunil Pratap Singh, Course Instructor, IISER Bhopal for giving us an opportunity to work on such an enriching assignment with good guidelines throughout the course of this semester.

In addition, a thank you to Mr Ayush Mathur from Economics Department, who guided us at crucial junctures of this project, and whose prompt help gave us the insight we needed to tackle the economics/finance component of the project.

We also thank the Indian Institute for Science Education and Research, Bhopal, for providing a conducive environment to us, greatly facilitating our project.

We would also like to expand our deepest gratitude to all those who have directly and indirectly guided us in writing this assignment.

The Authors

Contents

1 Introduction **3**

2 Essential Concepts/Theory **3**
2.1 Monte Carlo Simulations 3
2.2 Brownian Motion Model For Stock Market 3
2.3 Drift 3
2.4 Volatility 4

3 Qualitative Description/Equations **4**

4 Method and Code Implementation **5**
4.1 Code 7

5 Results/Plots/Analysis **9**

6 Appendix **9**
6.1 Capital Asset Pricing Model 9
6.2 Beta (Measure of Volatility) 10
6.3 Sharpe Ratio (Quantifiable Measure of Volatility) 10

7 Bibliography **11**
7.1 Monte Carlo Simulations 11
7.2 Brownian Motion Model 11
7.3 Apendix/Glossary 11

1 Introduction

The Stock Market is one of the most interesting examples of systems that can be modeled using Brownian Motion due to their heavily random trends and high volatility, affected deeply by the smallest of stimulus/initial conditions. We here attempt to analyse stock prices of individual stocks, along with the S&P500 index (Market-Capitalization-Weighted index of 500 leading publicly traded companies in the U.S) for reference, using Brownian motion and applying Monte Carlo simulations for the same. We shall also compute important quantities like Sharpe ratio and Beta-Scale (volatility) of a given share using MC Simulations for a more in depth analysis of the stock and whether it's a worthwhile investment.

2 Essential Concepts/Theory

To go further into the project, we need to go over a few important concepts we will repeatedly encounter further into the project. These include Monte Carlo Simulations, Brownian Motion Model For Stock Market, Drift, and Volatility.

2.1 Monte Carlo Simulations

An MC Simulation is an incredibly powerful tool that is used to predict the probabilities of different outcomes when there is some random variable/s present. It essentially works on **Repeated Random Sampling** to obtain Accurate Results. Quantitatively, MC Simulations involve drawing a large number of pseudo-random variables from a given distribution of a set interval, one at a time, or once at many different times, and assigning values to them based on the threshold we define.

2.2 Brownian Motion Model For Stock Market

In this model we assume that our assets (Stocks, in this case (may be replaced with SGBs etc.)) have continuous prices evolving continuously in time and are driven by Brownian motion processes. Brownian Motion is a pattern of motion that is characterized by random fluctuations in certain properties of a given particle/asset/variable. Within such an asset/-particle distribution, there exists no preferential direction of flow of the Brownian motion parameters (stock price, in this case). A detailed quantitative analysis of the same is given in the section titled **Qualitative Description/Equations** below.

2.3 Drift

The specific model of Brownian motion we are using here is called **Geometric Brownian Motion**. This kind of BM is characterized by two quantities, Drift and Volatility.
Drift is essentially a trend or growth rate. If the drift is positive, the trend is going up over time. If the drift is negative, the trend is going down. An index is said to drift if it does not return to unity when prices in the current period return to their levels in the base period. Chain indices may drift when prices fluctuate over the periods they cover.

2.4 Volatility

Geometric Brownian Motion is a continuous-time stochastic process in which the logarithm of the randomly varying quantity follows a Brownian motion with drift and volatility.The meaning of volatility is a variation or the spread of distribution. The value of volatility is always positive (or zero) because it is actually related to standard deviation of the distribution. Simply put, it is the standard deviation of the return of an asset (i.e Time -Series Distribution of the asset).

3 Qualitative Description/Equations

As mentioned before, Geometric Brownian Motion Equations contain additional terms like Drift and Volatility. The entire Equation Scheme of the same is listed below for reference. These equations formed the underlying basis of our code implementation.

$$\begin{gathered}\text{Drift } = \mu - \frac{1}{2}\sigma^2 \\ \text{Volatility } = \sigma Z[\text{Rand}(0;1)] \\ r = \left(\mu - \frac{1}{2}\sigma^2\right) + \sigma Z[\text{Rand}(0;1)] \\ S_t = S_{t-1} * e^{\left(\mu - \frac{1}{2}\sigma^2\right) + \sigma Z[\text{Rand}(0;1)]}\end{gathered} \tag{1}$$

The notation used is defined as follows:-

- S_t represents the stock price at time, t.
- S_{t-1} represents the stock price at timestamp preceding t, i.e (t-1).
- μ represents the mean of the distribution.
- σ represents the standard deviation of the distribution.

More generally, for any process to follow a Geometric Brownian Motion Pattern, it must satisfy the following equation:-

$$dS_t = \mu S_t dt + \sigma S_t dW_t \tag{2}$$

Here:-

- W_t is a Wiener process or Brownian motion
- μ is the percentage drift (Constant)
- σ is the percentage volatility (Constant)

4 Method and Code Implementation

In the Brownian motion model mentioned above, the stock price at the next time step depends on the drift and volatility. These values we get from the history of the stocks. The mean and the standard deviation of the historical returns gives the drift and volatility constants required in the model. So, we start by analyzing the past trends in the stock prices to get an idea about their dynamics. The stock price history over the last five years of the companies is shown in Figure 1 below.

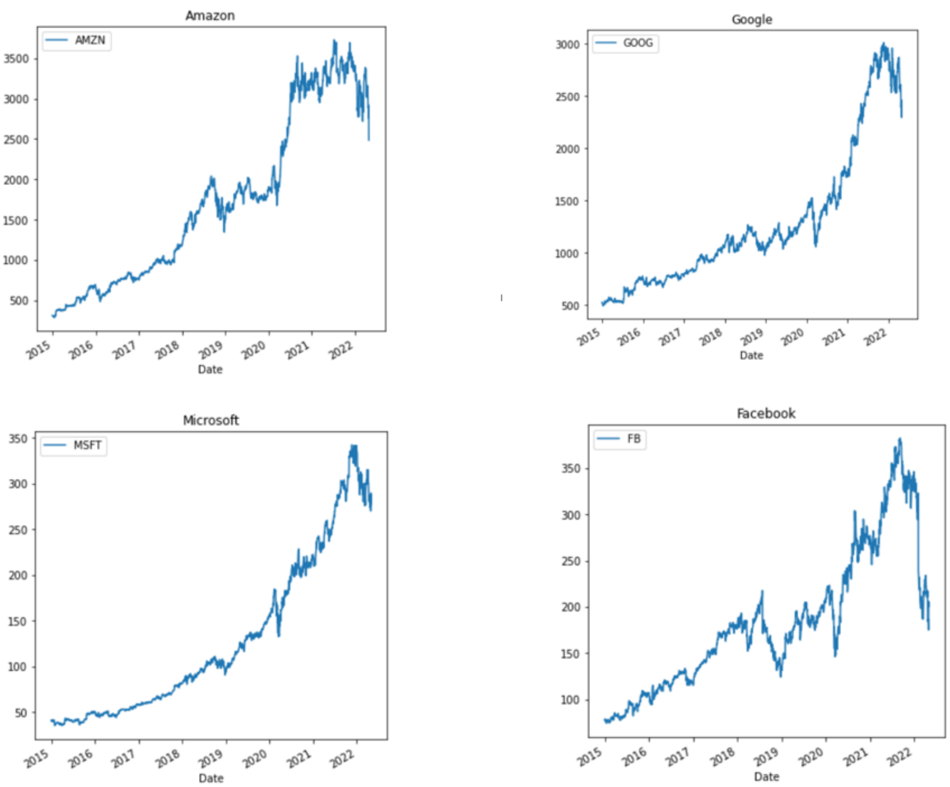

Figure 1: Stock price of analyzed companies over last 5 years

Monte-Carlo simulation is essentially getting random numbers from a probability distribution and sampling these values to project the target variable. To get the probability distribution, one requires the probability distribution of the data, specifically the distribution of the daily returns of the stocks. The density plot of daily returns of one particular stock is shown in Figure2.

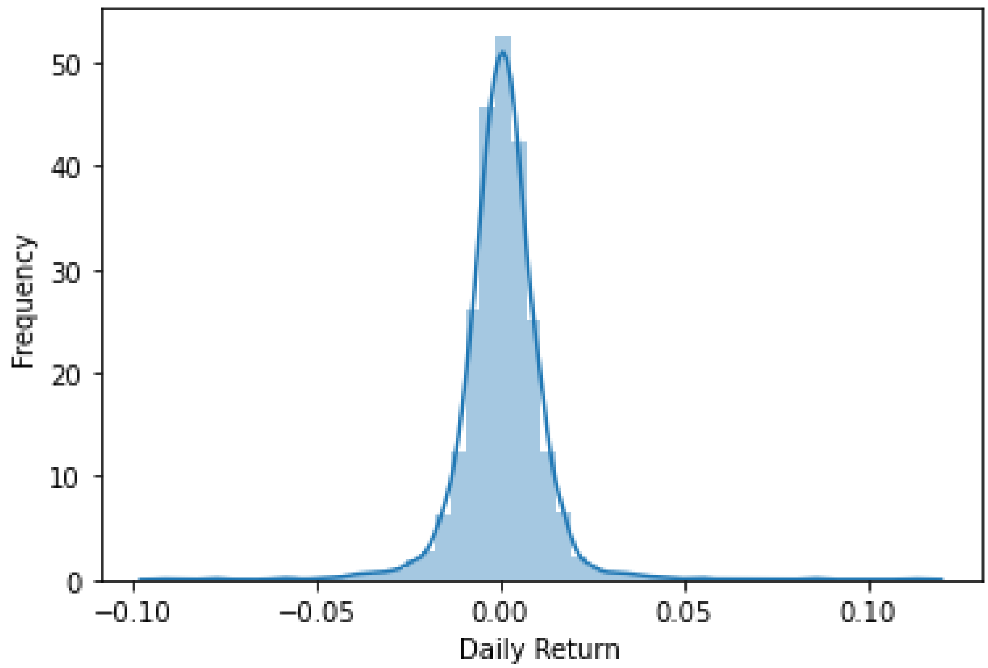

Figure 2: Density plot of daily returns

From the plot, we see that the distribution of the daily returns follows normal distribution; this gives the distribution from which the random numbers are chosen. Choosing n random numbers from the distribution gives the idea of how the stock would probably behave over the next n days. However, this by itself isn't particularly useful since this is just one out of the virtually infinite possible price series for this stock. The actual probability that the stock would follow this price series in the future is very close to zero. Critical information about the behavior and possible outcomes is obtained by running many(thousands or even more) random walk simulations with little insight from the historical data. For example, the random price curves for 60 simulations for 100 days in the future are shown in figure 3 below.

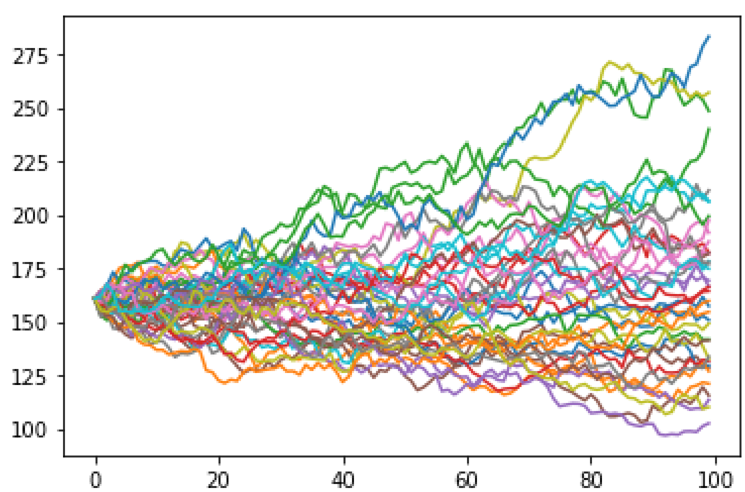

Figure 3: 40 simulations for a period of next 100 days

4.1 Code

The drift calculation is shown below. The pct_change is used to get the percentage changes of the adjusted close prices of the current day compared with the previous day. The resulting percentage of price changes are known as stock returns

```
def log(data):
 return (np.log(1+data.pct_change()))
```

```


def driftc(data, return_type='logarithmic'):
if return_type=='logarithmic':
    lr = log(data)
elif return_type=='simple':
    lr = simple_returns(data)
mu = lr.mean()
var = lr.var()
drift = mu-(0.5*var)
try:
    return drift.values
except:
    return drift
```

```
def daily_returns(data, n_days, iters, return_type='logarithmic'):
drift_v = driftc(data, return_type)
if return_type == 'logarithmic':
    try:
        st_val = log(data).std().values
    except:
        st_val = log(data).std()
elif return_type=='simple':
    try:
        st_val = simple_returns(data).std().values
    except:
        st_val = simple_returns(data).std()

s = np.exp(drift_v + st_val * stats.norm.ppf(np.random.rand(n_days,
iters)))
return s
```

```
def mc_sim(data, n_days, iters, output_type='logarithmic', plot=True):
# Generate daily returns
dreturns = daily_returns(data, n_days, iters, output_type)
# Create empty matrix
p_list = np.zeros_like(dreturns)
# Put the last actual price in the first row of matrix.
p_list[0] = data.iloc[-1]
# Calculate the price of each day
for i in range(1,n_days):
    p_list[i] = p_list[i-1]*dreturns[i]

# Plot Option
if plot == True:
    x = pd.DataFrame(p_list).iloc[-1]
```

```
        fig, ax = plt.subplots(1,2, figsize=(14,4))
        sns.distplot(x, ax=ax[0])
        sns.distplot(x, hist_kws={'cumulative':True},kde_kws={'cumulative'
    :True},ax=ax[1])
        plt.xlabel("Stock Price")
        plt.show()

    #CAPM and Sharpe Ratio

    # Printing information about stock
    try:
        [print(name) for name in data.columns]
    except:
        print(data.name)
    print(f"Days: {n_days-1}")
    print(f"Expected Value: ${round(pd.DataFrame(p_list).iloc[-1].mean()
    ,2)}")
    print(f"Return: {round(100*(pd.DataFrame(p_list).iloc[-1].mean()-
    p_list[0,1])/pd.DataFrame(p_list).iloc[-1].mean(),2)}%")
    print(f"Probability of Breakeven: {threshold_prob(pd.DataFrame(p_list)
    ,0, on='return')}")


    return pd.DataFrame(p_list)
```

```
def threshold_prob(predicted_data, threshold, on = 'value'):

    if on == 'return':
        pred_0 = predicted_data.iloc[0,0]
        pred = predicted_data.iloc[-1]
        p_list = list(pred)
        above = [(i*100)/pred_0 for i in p_list if ((i-pred_0)*100)/pred_0
    >= threshold]
        below = [(i*100)/pred_0 for i in p_list if ((i-pred_0)*100)/pred_0
    < threshold]
    elif on == 'value':
        pred = predicted_data.iloc[-1]
        p_list = list(pred)
        above = [i for i in p_list if i >= threshold]
        below = [i for i in p_list if i < threshold]
    else:
        print("'on' must be either value or return")
    probability=(len(above)/(len(above)+len(below)))
    return probability
```

5 Results/Plots/Analysis

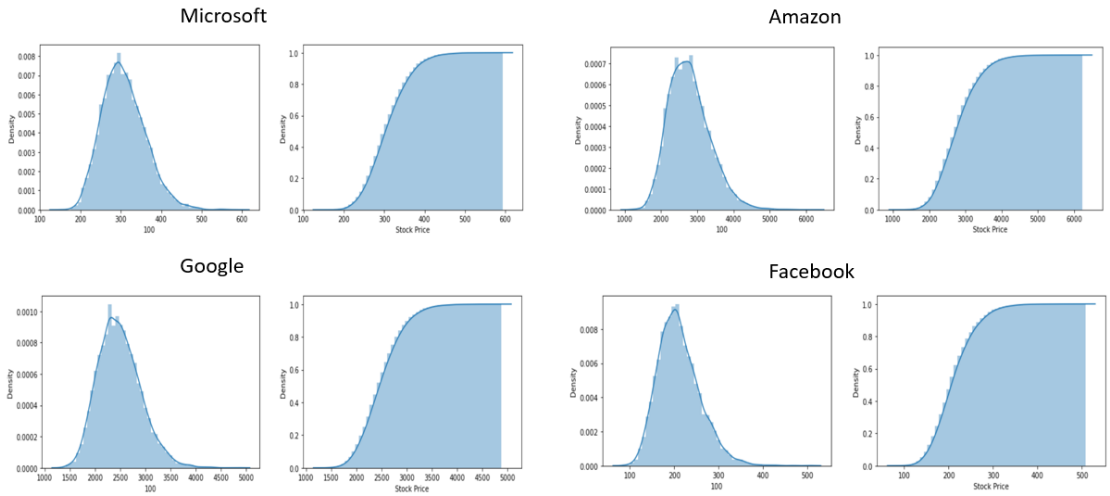

Figure 4: 40 simulations for a period of next 100 days

6 Appendix

6.1 Capital Asset Pricing Model

The Capital Asset Pricing Model (CAPM) helps to calculate investment risk and what ROI one should expect as an investor. CAP Model deals with quantification of Systematic Risks, using the below mentioned formula. Sysematic risks, for reference, are market risks that cannot be diversified away. Severe Calamities/Pandemic, Recessions, and Wars are examples of systematic risks.

$$R_a = R_{rf} + \beta_a * (R_m - R_{rf})$$

where:

$$\begin{aligned} R_a &= \text{ Expected return on a security} \\ R_{rf} &= \text{ Risk-free rate} \\ R_m &= \text{ Expected return of the market} \\ \beta_a &= \text{ The beta of the security} \\ (R_m - R_{rf}) &= \text{ Equity market premium} \end{aligned}$$

CAPM's starting point is the risk-free rate–typically a 10-year government bond yield-/Sovereign Gold Bonds etc. A premium is added, one that equity investors demand as compensation for the extra risk they accrue. This equity market premium consists of the expected return from the market as a whole, less the risk-free rate of return. The equity risk premium is multiplied by a coefficient called β. The details of what β is, are listed below.

6.2 Beta (Measure of Volatility)

It measures a stock's relative volatility–that is, it shows how much the price of a particular stock jumps up and down compared with how much the entire stock market jumps up and down. If a share price moves exactly in line with the market, then the stock's beta is 1. A stock with a beta of 1.5 would rise by 15% if the market rose by 10 % and fall by 15% if the market fell by 10%.

Beta is found by statistical analysis of individual, daily share price returns in comparison with the market's daily returns over precisely the same period.

6.3 Sharpe Ratio (Quantifiable Measure of Volatility)

The Sharpe ratio is used to help investors understand the return of an investment compared to its risk. The ratio is the average return earned in excess of the risk-free rate per unit of volatility or total risk. Volatility is a measure of the price fluctuations of an asset or portfolio. The formulaic details of the Sharpe Ratio are given below:-

Sharpe Ratio = $\frac{R_p - R_f}{\sigma_p}$ where:

$$R_p = \text{ return of portfolio}$$
$$R_f = \text{ risk-free rate}$$
$$\sigma_p = \text{ standard deviation of the portfolio's excess return}$$

7 Bibliography

7.1 Monte Carlo Simulations

- Kroese, D. P.; Brereton, T.; Taimre, T.; Botev, Z. I. (2014). "Why the Monte Carlo method is so important today". WIREs Comput Stat. 6 (6): 386–392. doi:10.1002/wics.1314. S2CID 18521840

- Spall, J. C. (2003), Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control, Wiley, Hoboken, NJ. `http://www.jhuapl.edu/ISSO`

- V. Prokaj. Proposal selection for mcmc simulation. Applied Stochastic Models and Data Analysis, pp. 61–65, 2009

- J. S. Dagpunar. Simulation and Monte Carlo. John Willey & Sons Ltd., Great Britain, Chippenham, Willtshire, 2007.

7.2 Brownian Motion Model

- Tsekov, Roumen (2013). "Brownian Markets". Chin. Phys. Lett. 30 (8): 088901. arXiv:1010.2061. Bibcode:2013ChPhL..30h8901T. doi:10.1088/0256-307X/30/8/088901. S2CID 18675919

- Karatzas, Ioannis; Shreve, Steven E. (1991). Brownian motion and stochastic calculus. New York: Springer-Verlag. ISBN 0-387-97655-8.

- Ross, Sheldon M. (2014). "Variations on Brownian Motion". Introduction to Probability Models (11th ed.). Amsterdam: Elsevier. pp. 612–14. ISBN 978-0-12-407948-9.

- Ross, Sheldon M. (2014). "Variations on Brownian Motion". Introduction to Probability Models (11th ed.). Amsterdam: Elsevier. pp. 612–14. ISBN 978-0-12-407948-9.

- Karl Sigman. "Simulating Brownian motion (BM) and geometric Brownian motion (GBM)". `http://www.columbia.edu/~ks20/4404-Sigman/4404-Notes-sim-BM.pdf`

7.3 Apendix/Glossary

- `https://www.investopedia.com/articles/06/capm.asp`

- `https://www.investopedia.com/terms/b/beta.asp`

- `https://www.investopedia.com/articles/financial-theory/08/risk-free-rate-return.asp`

- Sharpe, William F. (1994). "The Sharpe Ratio". The Journal of Portfolio Management. 21 (1): 49–58

- I Antoniou et al. "On the log-normal distribution of stock market data". In: Physica A: Statistical Mechanics and its Applications 331.3 (2004), pp. 617–638