Modeling Stock Market Trends with Geometric Brownian Motion

Prepared by:

Omerzahid Ali

Chieh-An Chang

Hua-Hsun Hsieh

Instructor: Dr. Omidali A. Jazi

University of Toronto Mississauga

December 3, 2024

Contents

1 Introduction

This project utilizes the principles of stochastic processes, with a focus on Geometric Brownian Motion (GBM), to model and predict stock prices. GBM is a foundational model in financial analysis, widely recognized for its ability to describe stock price dynamics. Its flexibility and adaptability make it a cornerstone for extending stochastic processes to more complex financial models.

In this paper, we will outline the theoretical foundation of GBM, define its key parameters, and demonstrate its application in stock market prediction. We incorporate Hamiltonian Monte Carlo (HMC) for parameter estimation to enhance the precision of our model. Our case study focuses on Johnson & Johnson (JNJ) stock data, utilizing GBM to simulate price trends and assess model performance. The paper concludes with a discussion of insights, limitations, and potential areas for future research.

2 Review of the Topic

The goal of this project is to model stock prices using GBM and evaluate its effectiveness in capturing the dynamics of financial markets. Stock prices are subject to random fluctuations influenced by various factors. While their behavior may resemble a simple random walk, such comparisons are limited by the assumptions of that model.

A simple random walk assumes price changes occur in fixed, equal-sized steps, independent of past movements. While the independence assumption aligns with market behavior, the fixed step size fails to capture the variable nature of real-world price fluctuations. As a result, the simple random walk is inadequate for accurately modeling stock price behavior.

To address these shortcomings, GBM extends the random walk model by incorporating continuous and variable price changes. By assuming that stock price returns follow a normal distribution, GBM introduces randomness in both the direction and magnitude of price movements. This makes it a more realistic model for financial markets.

In this project, we employ GBM to simulate stock price trends, using HMC for precise estimation of the model's drift and volatility parameters. This combined approach provides a robust framework for analyzing the variability and trends in stock prices, setting the stage for its practical application in predicting future stock behavior.

3 Definitions and Examples

3.1 Brownian Motion

Stock markets, foreign exchange markets, commodity markets, and bond markets are all assumed to follow Brownian motion.[1] In these markets, assets change continuously over very small intervals of time, and their states are altered by random amounts. This randomness is captured by Brownian motion, a stochastic process that models unpredictable, continuous fluctuations.

More importantly, the mathematical models describing Brownian motion form the foundation of financial asset pricing and derivatives pricing models.[2] These models provide a rigorous framework for understanding the random behavior of asset prices and are essential for the development of tools like Geometric Brownian Motion (GBM).

3.2 Importance of Geometric Brownian Motion

Building upon the concept of Brownian motion, GBM introduces a deterministic drift component (μ) and a volatility term (σ) scaled by Brownian motion. This makes GBM particularly well-suited for modeling stock prices, as it captures both the underlying price trend and the inherent volatility over time.[3] The assumption of log-normality is a widely accepted standard in the industry, though it is important to note that real-world data may not always fully align with this assumption.[4]

3.3 The Validity of Geometric Brownian Motion in Stock Market Modeling

The analysis of stock log returns demonstrates a high level of randomness, as indicated by elevated Shannon entropy.[5] This reflects the inherently unpredictable nature of stock market behavior. Additionally, predictive models such as Geometric Brownian Motion (GBM) preform well under metrics like Mean Absolute Percentage Error (MAPE), where a lower MAPE signifies improved predictive accuracy.[6] These findings underscore the utility of GBM as a robust framework for modeling and forecasting stock market dynamics, offering valuable insights despite the inherent challenges of financial prediction.

[1] Peng and Simon [8], 1242
[2] Ermogenous [2], 5
[3] Ermogenous [2], 6
[4] Mota [7], 56
[5] Brătian et al. [1], 15
[6] Brătian et al. [1], 13

3.4 Parameter Estimation Using Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) methods are used to estimate parameters when prior knowledge is unavailable. MCMC is particularly effective in capturing uncertainty in parameter estimates and generating posterior distributions.[7]

3.5 Application in Stock Price Estimation

The GBM model, represented by a stochastic differential equation (SDE), describes the evolution of stock prices over time. To estimate the key parameters μ (drift) and σ (volatility), we employ Hamiltonian Monte Carlo (HMC) within the MCMC framework. These estimated parameters are subsequently used to simulate future stock prices and construct predictive intervals.

4 Details of the topic

4.1 Introduction to Brownian Motion

Brownian motion, also called Wiener process, is a continuous-time stochastic process X_t with the following properties:[8]

1. $X_0 = 0$,
2. For $0 \leq t_1 < t_2 < \cdots < t_n$, the increments $X_{t_2} - X_{t_1}, \ldots, X_{t_n} - X_{t_{n-1}}$ are independent,
3. The increments are normally distributed: $X_t - X_s \sim N(0, (t-s)\sigma^2)$,
4. X_t has continuous paths, i.e., the function $t \to X_t$ is a continuous function of t.

Brownian motion serves as a foundational model for randomness with respect to time.

4.2 Brownian Motion with Drift and Volatility

In financial modeling, stock prices are better captured by **Brownian motion with drift and volatility**, also called geometric Brownian motion (GBM). This process incorporates two critical features:

- A **drift term** μ, representing the deterministic trend or expected growth rate,
- A **volatility term** σ, representing the random fluctuations.

[7] Saadi, Ykhlef, and Guessoum [11]
[8] Lawler [6], 174

The stochastic differential equation (SDE) for GBM is:

$$dS_t = \mu S_t\, dt + \sigma S_t\, dW_t,$$

where S_t is the stock price, μ is the percentage drift, σ is the volatility, and W_t is a standard Brownian motion.[9] [10]

4.3 Stochastic Integration with Respect to Brownian Motion

Stochastic integration generalizes the classical integral to stochastic processes, allowing integration with respect to Brownian motion. Unlike classical Riemann or Lebesgue integration, stochastic integrals account for the randomness in the integrator. For a process X_t, the stochastic integral $\int_0^t X_s\, dW_s$ is well-defined if X_t is adapted and satisfies certain integrability conditions.[11]

4.4 Ito's Formula III

Ito's formula is an extension of the chain rule to stochastic processes. In its more advanced form, often referred to as Ito's Formula III[12], it establishes the dynamics of a function $f(t, Z_t)$, where Z_t is a stochastic process satisfying:

$$dZ_t = X_t\, dt + Y_t\, dW_t,$$

and W_t is a standard Brownian motion. If $f(t, x)$ has two continuous derivatives in x and one continuous derivative in t, then $f(t, Z_t)$ evolves according to:

$$f(t, Z_t) - f(0, Z_0) = \int_0^t \frac{\partial f}{\partial x}(s, Z_s)\, dW_s + \int_0^t \left(\frac{\partial f}{\partial t}(s, Z_s) + \frac{1}{2} \frac{\partial^2 f}{\partial x^2}(s, Z_s) Y_s^2 \right) ds.$$

4.4.1 Explanation of the Terms

1. **First Integral**: The term $\int_0^t \frac{\partial f}{\partial x}(s, Z_s)\, dW_s$ represents the stochastic part of the evolution, integrating with respect to the Brownian motion W_t. It captures the randomness inherent in the process.

2. **Second Integral**:

   The term $\int_0^t \left(\frac{\partial f}{\partial t}(s, Z_s) + \frac{1}{2} \frac{\partial^2 f}{\partial x^2}(s, Z_s) Y_s^2 \right) ds$ represents the deterministic contribution.

[9] Ermogenous [2], 6
[10] Lawler [6], 223
[11] Lawler [6], 200
[12] Lawler [6], 212

4.5 Solving the Stochastic Differential Equation Using Ito's Formula

If $f(t,x) = e^{at+bx}$, then applying Ito's formula gives us the following[13]:

$$Z_t = e^{at+bW_t} \quad \text{satisfies} \quad dZ_t = bZ_t\,dW_t + \left(a + \frac{b^2}{2}\right) Z_t\,dt.$$

This result comes directly from Ito's Formula III[14], where the stochastic and deterministic terms are handled explicitly.

For the SDE $dS_t = \mu S_t\,dt + \sigma S_t\,dW_t$, we recognize it has the same form as dZ_t. Setting $a = \mu - \sigma^2/2$ and $b = \sigma$, the solution follows:

$$S_t = S_0 \exp\left[\left(\mu - \frac{\sigma^2}{2}\right) t + \sigma W_t\right].$$

This implies that S_t is log-normally distributed, with $\mathbb{E}[S_t] = S_0 e^{\mu t},\ \mathrm{Var}[S_t] = S_0^2 e^{2\mu t}(e^{\sigma^2 t} - 1)$.[15]

Thus, the solution leverages the known structure of exponential functions under stochastic dynamics. By simulating paths of S_t or using the distributional properties of GBM, one can predict stock prices probabilistically.

4.6 Markov Chain Monte Carlo for Geometric Brownian Motion

In our analysis, we utilize Hamiltonian Monte Carlo (HMC) to estimate the parameters of the GBM model. Unlike traditional MCMC methods, such as Metropolis-Hastings, which propose new samples based on random steps, HMC uses the gradient of the log-posterior to inform the sampling process.[16] This allows HMC to take larger, more informed steps in the parameter space, leading to faster convergence and more efficient exploration of the posterior distribution.[17]

The HMC algorithm in our application works by constructing a Hamiltonian system, where the parameters of interest (drift and volatility) are treated as "particles" moving through a potential field. The algorithm uses the gradients of the log-likelihood to simulate the movement of these particles and generates new samples of the parameters that reflect their posterior distribution. This process is computationally efficient, making it well-suited for parameter estimation in models like GBM.[18]

[13] Lawler [6], 213
[14] Lawler [6], 212
[15] Wu and Buyya [14], 741
[16] Seiler [12], 201
[17] Yamada, Ohno, and Ohta [15], 14
[18] Gelman et al. [5], 307

5 Application

In this section, we will apply the Geometric Brownian Motion (GBM) model, along with Hamiltonian Monte Carlo (HMC) for parameter estimation, as discussed earlier. Our objective is to use the estimated parameters of drift and volatility from historical JNJ stock data to simulate future stock prices.

5.1 Describe the Dataset

The dataset was obtained from Yahoo Finance.[19] Tables 1 and 2 below display the structure of the dataset, showcasing two sample rows.

Table 1: Columns 1 to 6 of JNJ Historical Data

ticker	ref_date	price_open	price_high	price_low	price_close
JNJ	2021-01-04	157.24	157.38	154.13	156.50
JNJ	2021-01-05	156.25	158.76	155.07	158.34

Table 2: Columns 7 to 11 of JNJ Historical Data

volume	price_adjusted	ret_adjusted_prices	ret_closing_prices	cumret_adjusted_prices
11765900	139.9339	NA	NA	1.000000
9602300	141.5791	0.0117573	0.0117572	1.011757

It contains 11 different variables[20] where:

1. ticker: represents the id name of a stock(e.g. Apple company will use the id name AAPL)
2. ref_date: represents the reference date of a stock.
3. price_open: reprsents the open price of a stock.
4. price_high: represents the highest price of a stock in a day.
5. price_low: represents the lowest price of a stock in a day.
6. price_close: represents the raw closing price of a stock in a day.
7. volumn: represents the number of tradings of a stock in a day.
8. price_adjusted: represents the adjustment price of a stock after event such as dividend or etc.

[19] Finance [3]
[20] Perlin [9], "The Data"

9. ret_adjusted_prices: represents the log return of a stock after price adjustment.
10. ret_closing_prices: represents the log return of a stock without price adjustment
11. curret_adjusted_prices: The accumulated arithmetic/log return for the period (starts at 100%)

In this application, our primary focus is on the variable *price_adjusted* within the dataset. Using this variable, we will explicitly compute the log returns based on the adjusted prices.

5.2 Verify the Normality of the Log Return.

As discussed in previous sections, financial institutions often utilize GBM for stock prediction due to its assumption that log returns approximate a normal distribution. This makes GBM particularly effective for modeling changes that follow a roughly normal distribution.

To assess whether our data approximately follow a normal distribution, we begin by visualizing it using a histogram. Subsequently, a Q-Q plot is employed for further analysis.

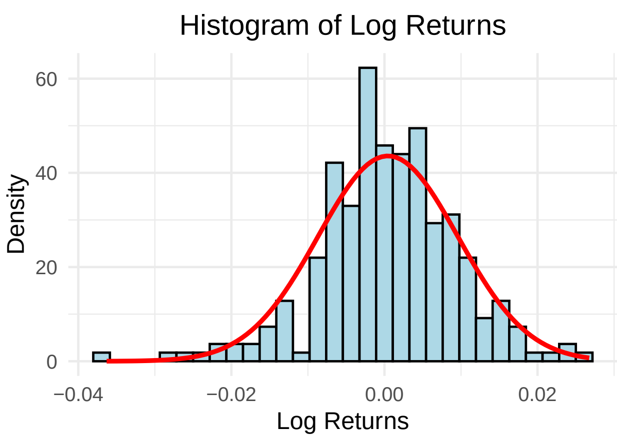

Figure 1a: Histogram of log returns.

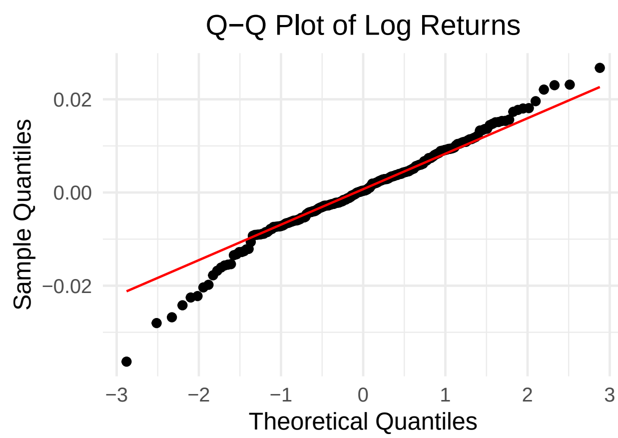

Figure 1b: Q−Q plot of log returns.

Figure 1: Normality Check of Log Returns

Figure 1a shows the histogram of log returns, resembling a bell curve with slight leftward skewness, likely due to an outlier. Excluding this outlier, the distribution appears approximately normal with a mean near zero.

Figure 1b presents the Q-Q plot, comparing the empirical data to theoretical quantiles. While most points on the right align with the theoretical quantiles, data points on the left deviate slightly, indicating a left skew.[21]

[21] kross2016qqplot, Plots 2 & 3: A Tale of Tails

Despite these deviations, the majority of data points align well with the theoretical quantiles, and the outliers' influence is limited. Thus, we conclude that it is reasonable to treat the log returns as approximately normally distributed based on both the histogram and Q-Q plot analyses.

5.3 Using MCMC to Estimate Parameters

In the Geometric Brownian Motion (GBM) model, predictions require not only the initial price but also the drift (expected log return), volatility (variance of log return), and time increment (dt).

With the normal distribution of log returns established, we can now proceed with making predictions. Using data from January 2021 to January 2022, we estimate the drift and volatility through Hamiltonian Monte Carlo (HMC) simulation. Our objective is to forecast the stock's trend from January 2022 to January 2023, leveraging the data from the preceding year. Upon applying HMC, we obtain the following estimates for the drift and volatility:

```
## The drift (mu) is: 0.1165554
```

```
## The volatility (sigma) is: 0.145965
```

The final piece of the configuration is determining the number of time steps for the forecast. Since we aim to make daily predictions, the time increment (dt) will be set to $\frac{1}{252}$, corresponding to the typical number of trading days in a year.

The variable W, representing the Brownian motion component, will follow a normal distribution with a mean of 0 and a standard deviation scaled by the time increment dt, as discussed earlier.

Since we have estimate all the parameters, we can make a prediction for the 2022 to 2023 stock price for stock JNJ through the GBM model.

5.4 Prediction and Simulation

The red line on both following figures represents the actual stock price from 2022 to 2023. The blue lines indicate the simulations based on the drift, volatility, and time increment we established in the previous section.

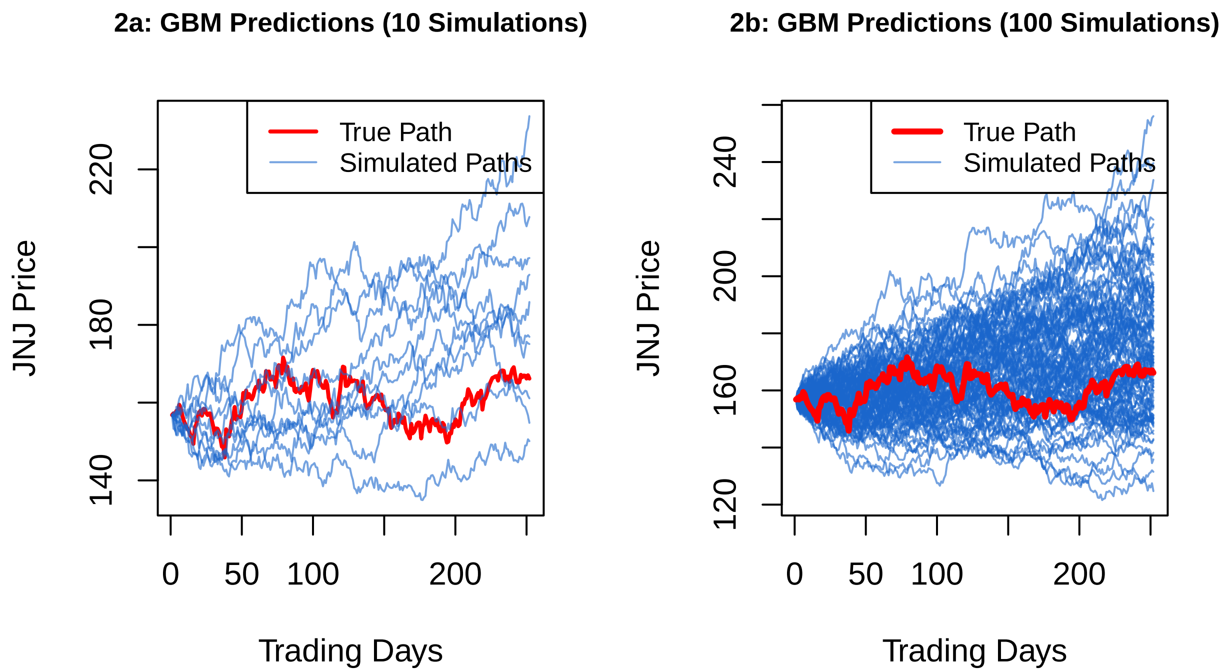

Figure 2: GBM Simulations of JNJ Stock Prices

Figure 2a shows the results of 10 simulations, where the variance appears reasonable for prediction periods of up to approximately 40 days, as the simulated price range closely follows the true path. However, over a longer horizon, the simulated price range widens significantly, spanning from around 140 to 230. This introduces concerns about the model's ability to accurately capture the true price path over a longer time frame.

To investigate whether increasing the number of simulations affects the variability in both the short and long term, we expanded the simulation to 100 paths, as shown in Figure 2b. In the longer term, the simulations now cluster primarily between 140 and 210 at the 252-day mark. However, in the short run, the paths continue to concentrate closely around the true price path, indicating minimal variation during this period. This suggests that Geometric Brownian Motion (GBM) provides a reliable method for short-term stock price prediction, as most of the simulation paths align with the actual price over the shorter horizon.

This clustering of simulations around the true path also leads us to the concept of constructing a confidence interval for the predicted prices, allowing us to quantify the uncertainty in our forecast.

5.5 95% Confidence Interval.

With 100 simulated paths, we construct a confidence interval around the forecasted stock prices for each day. The grey shaded area in Figure 3 below represents the 95% confidence interval, indicating that approximately 95% of the simulations are expected to capture the true stock price on any given day.

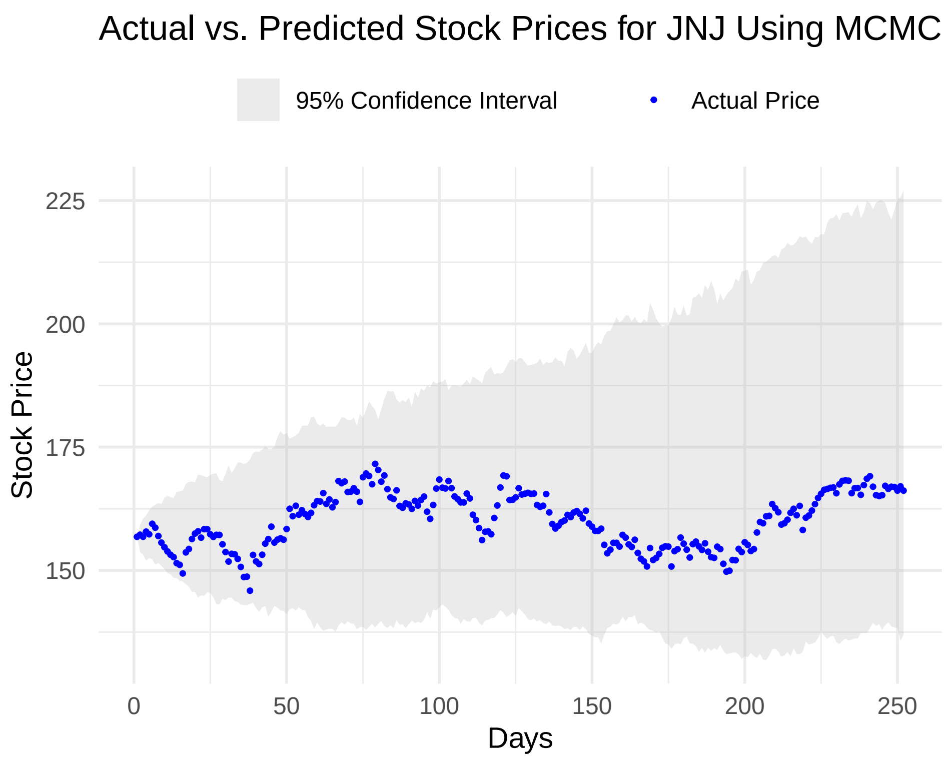

Figure 3: The 95% confidence interval for the results of the GBM model.

Throughout the 252-day period, if we cut a vertical line on a specific day, the grey line represents the recalculated 95% confidence interval on that day.

The true stock price in each date consistently falls within the specified interval each day, which confirms the reliability of our predictions.

One interesting observation is that as the number of days increases, the area of the 95% confidence interval also expands. This aligns with our previous analysis, which concluded that geometric Brownian motion (GBM) tends to exhibit less variability in the short term.

6 Conclusion

In this project, we explored the application of Geometric Brownian Motion (GBM) to stock market analysis, specifically focusing on the stock "JNJ." Using Hamiltonian Monte Carlo (HMC) for parameter estimation, we successfully applied GBM to model the behavior of stock prices based on historical data. Our results show that GBM can effectively capture the underlying trends of stock price movements and generate meaningful predictions, particularly when assessing the 95% confidence interval.

While the model provided valuable insights, we observed that the range of the 95% confidence interval increased as the forecast horizon extended. This is likely due to the inherent limitations of GBM, which assumes constant drift and volatility over time. In reality, these parameters can fluctuate due to market conditions, macroeconomic factors, and political events, leading to potential outliers in the data. This is a well-known challenge when using any model to predict stock prices.

Despite these limitations, GBM remains a widely used and foundational model in financial analysis, offering a solid starting point for stock price predictions. It serves as a baseline that can be extended with more sophisticated models, such as the Black-Scholes model for options pricing, which could be a natural next step in deepening our understanding of financial modeling beyond GBM.

Appendix

```
# Connects to Yahoo Finance's API.
library(yfR)
# Helps with extracting data from datasets.
library(dplyr)

# Libraries for creating tables using kable
library(knitr)
library(kableExtra)

# Libraries for plotting
library(ggplot2)
library(gridExtra)
library(grid)

# Library for Bayesian inference using HMC
library(rstan)
```

```
# Define the stock ticker symbol for Johnson & Johnson (JNJ)
tickers <- "JNJ"

# Set the date range for stock data
start_date <- as.Date("2021-01-01") # Start date: January 1, 2021
end_date <- as.Date("2022-01-01")   # End date: January 1, 2022

# Fetch historical stock data for JNJ from Yahoo Finance using the 'yf_get' function
stock_data <- yf_get(tickers = tickers, first_date = start_date, last_date = end_date)
```

The following code will utilize the knitr and kableExtra libraries.[22]

```
# Display a sample of JNJ historical data (first 2 rows)

# Extract and format the first 6 columns for preview
```

[22] Zhu [16]

```r
table1 <- head(stock_data[1:6], n = 2)  # Select the first 6 columns and first 2 rows
# Add a caption for the table
kable(table1, caption = "Columns 1 to 6 of JNJ Historical Data") %>%
  # Style the table to hold its position in LaTeX
  kable_styling(latex_options = "hold_position")

# Extract and format columns 7 to 11 for preview
table2 <- head(stock_data[7:11], n = 2)  # Select columns 7 to 11 and first 2 rows
# Add a caption for the table
kable(table2, caption = "Columns 7 to 11 of JNJ Historical Data") %>%
  # Style the table to hold its position in LaTeX
  kable_styling(latex_options = "hold_position")
```

```r
# Compute the log returns of JNJ stock from 2021 to 2022

# Extract the adjusted closing prices of the stock

# 'price_adjusted' contains adjusted closing prices
stock_price <- stock_data$price_adjusted

# Calculate the log returns
log_returns <- diff(log(stock_price))  # Use the difference of log to compute log returns

# The log return for a stock is calculated as:
# log_return[t] = log(stock_price[t] / stock_price[t-1])
# Using 'diff' with 'log' achieves this computation for the entire series.
```

```r
# Generate a histogram with a normal curve overlay for log returns
histogram_plot <- ggplot(data = data.frame(log_returns = log_returns),
                         aes(x = log_returns)) +
  geom_histogram(aes(y = ..density..), fill = "lightblue", color = "black") +
  # The histogram uses density instead of raw counts so it matches the scale of
  # the normal curve, allowing them to be displayed together.
```

```r
  stat_function(fun = dnorm, args = list(mean = mean(log_returns), sd = sd(log_returns)),
                color = "red", size = 1) +
  # Overlays a normal curve using the mean and standard deviation of the log returns
  ggtitle("Histogram of Log Returns") +
  xlab("Log Returns") +
  ylab("Density") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
  # Centers the title above the plot

# Generate a Q-Q plot to assess the normality of log returns
qq_plot <- ggplot(data.frame(log_returns = log_returns), aes(sample = log_returns)) +
  stat_qq() +
  # Plots sample quantiles against theoretical quantiles from a normal distribution
  stat_qq_line(color = "red") +
  # Adds a red reference line to visualize deviations from normality
  ggtitle("Q-Q Plot of Log Returns") +
  xlab("Theoretical Quantiles") +
  ylab("Sample Quantiles") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))
```

Since we are using the ggplot2 library, we cannot use par(mfrow = c(1, 2)) to combine two plots into the same box. Instead, we use the grid library to combine the two plots. .[23]

```r
# Arrange the histogram and Q-Q plot side by side with individual captions
combined_plot <- grid.arrange(
  arrangeGrob(
    histogram_plot,
    bottom = textGrob("Figure 1a: Histogram of log returns.",
                      gp = gpar(fontsize = 10), hjust = 0.5) # Caption for histogram
  ),
  arrangeGrob(
```

[23]Rygel [10]

```
    qq_plot,
    bottom = textGrob("Figure 1b: Q-Q plot of log returns.",
                      gp = gpar(fontsize = 10), hjust = 0.5) # Caption for Q-Q plot
  ),
  ncol = 2 # Arrange plots in two columns
)
```

```
# Download JNJ historical data from 2021 to 2023
tickers <- "JNJ"
start_date <- as.Date("2021-01-01")
end_date <- as.Date("2023-01-01")
stock_data <- yf_get(tickers = tickers, first_date = start_date, last_date = end_date)

# Extract the adjusted closing prices for analysis
adjusted_prices <- stock_data$price_adjusted

# Select the last 252 trading days (approximately 1 year of data for 2022-2023)
prices_2022_2023 <- tail(adjusted_prices, 252)

# Get the starting price (first price in the series for simulation)
S0 <- as.numeric(first(prices_2022_2023))

# Compute log returns for the period excluding the last 252 trading days
log_returns <- na.omit(diff(log(adjusted_prices)))[1:(length(adjusted_prices) - 252)]
```

The following code will utilize the rstan package to help estimate drift and volatility using HMC.[24]

```
# Prepare data for the Stan model
stan_data <- list(
  N = length(log_returns),    # Number of log returns
  y = as.vector(log_returns), # Log-returns vector (log(S_t / S_(t-1)))
  dt = 1 / 252                # Time step, assuming 252 trading days in a year
)
```

[24] Team [13]

```
# Define the Stan model for Geometric Brownian Motion (GBM)
stan_model_code <- "
data {
  int<lower=0> N;          // Number of observations
  real y[N];               // Log-returns: log(S_t / S_(t-1))
  real dt;                 // Time step (dt)
}
parameters {
  real mu;                 // Drift (mean return)
  real<lower=0> sigma;     // Volatility (standard deviation of returns)
}
model {
  // Likelihood: log-returns are roughly normally distributed
  for (i in 1:N) {
    y[i] ~ normal(mu * dt, sigma * sqrt(dt));
  }
}
"

# Compile the Stan model
model <- stan_model(model_code = stan_model_code)

# Fit the model using MCMC sampling
fit <- sampling(model, data = stan_data, iter = 2000, chains = 4)

# Extract posterior samples
posterior_samples <- extract(fit)

# Compute the posterior mean of mu and sigma
mu <- mean(posterior_samples$mu)
cat("The drift (mu) is:", mu, "\n")
```

```r
sigma <- mean(posterior_samples$sigma)
cat("The volatility (sigma) is:", sigma, "\n")
```

```r
# Generate plots for GBM predictions
par(mfrow = c(1, 2))  # Set up a 1x2 plotting area for side-by-side plots


# --- First Plot: 10 Simulations ---
set.seed(2024)
n <- 252  # Number of trading days (1 year)
dt <- 1 / n  # Time step
simulations <- 10  # Number of simulation paths
# Brownian increments
W <- matrix(rnorm(n * simulations, mean = 0, sd = sqrt(dt)), nrow = n, ncol = simulations)
S <- matrix(0, nrow = n, ncol = simulations)  # Matrix for simulated paths
S[1, ] <- S0  # Initialize paths with starting price


# Generate GBM paths
for (i in 2:n) {
  S[i, ] <- S[i-1, ] * exp((mu - 0.5 * sigma^2) * dt + sigma * W[i, ])
}


# Plot the true price path and simulated paths
plot(as.numeric(prices_2022_2023), type = "l", col = "red", lwd = 2,
     ylim = range(S, prices_2022_2023), main = "2a: GBM Predictions (10 Simulations)",
     cex.main = 0.8, xlab = "Trading Days", ylab = "JNJ Price")
matlines(S, type = "l", lty = 1, col = rgb(0.1, 0.4, 0.8, 0.6))  # Add simulated paths
legend("topright", legend = c("True Path", "Simulated Paths"),
       col = c("red", rgb(0.1, 0.4, 0.8, 0.6)), lty = 1, lwd = c(2, 1), cex = 0.8)


# --- Second Plot: 100 Simulations ---
set.seed(2024)
simulations <- 100  # Increase the number of simulation paths
# Brownian increments
```

```
W <- matrix(rnorm(n * simulations, mean = 0, sd = sqrt(dt)), nrow = n, ncol = simulations)
S <- matrix(0, nrow = n, ncol = simulations)  # Matrix for simulated paths
S[1, ] <- S0  # Initialize paths with starting price

# Generate GBM paths
for (i in 2:n) {
  S[i, ] <- S[i-1, ] * exp((mu - 0.5 * sigma^2) * dt + sigma * W[i, ])
}

# Plot the simulated paths first
# Set up the plot without drawing the red line yet
plot(as.numeric(prices_2022_2023), type = "n",
     ylim = range(S, prices_2022_2023), main = "2b: GBM Predictions (100 Simulations)",
     cex.main = 0.8, xlab = "Trading Days", ylab = "JNJ Price")
matlines(S, type = "l", lty = 1, col = rgb(0.1, 0.4, 0.8, 0.6))  # Add simulated paths
# Ensure it overlays the blue lines
lines(as.numeric(prices_2022_2023), type = "l", col = "red", lwd = 3)

# Add the legend
legend("topright", legend = c("True Path", "Simulated Paths"),
       col = c("red", rgb(0.1, 0.4, 0.8, 0.6)), lty = 1, lwd = c(3, 1), cex = 0.8)
```

```
# Reset plotting area to default
par(mfrow = c(1, 1))
```

To generate the area of the 95% confidence interval, we use ggplot with geom_ribbon.[25]

```
# Calculate the 95% confidence intervals for predicted stock prices
conf_intervals <- apply(S, 1, function(prices) {
  quantile(prices, probs = c(0.025, 0.975))  # 2.5% and 97.5% quantiles
})

# Extract the lower and upper bounds
```

[25] GeeksforGeeks [4]

```
lower_bound <- conf_intervals[1, ]
upper_bound <- conf_intervals[2, ]

# Prepare data for plotting
plot_data <- data.frame(
  Day = 1:length(lower_bound),  # Day indices
  Lower_Bound = lower_bound,    # Lower bound of confidence interval
  Upper_Bound = upper_bound     # Upper bound of confidence interval
)

# Prepare historical data for comparison
historical_data <- data.frame(
  Day = 1:length(prices_2022_2023),  # Day indices
  Actual_Price = prices_2022_2023    # Actual observed prices
)

# Generate the plot
ggplot() +
  # Plot the confidence interval as a shaded area
  geom_ribbon(
    data = plot_data,
    aes(x = Day, ymin = Lower_Bound, ymax = Upper_Bound,
        fill = "95% Confidence Interval"),
    alpha = 0.3
  ) +
  # Overlay the actual historical prices as points
  geom_point(
    data = historical_data,
    aes(x = Day, y = Actual_Price, color = "Actual Price"),
    size = 0.5
  ) +
  # Customize the legend for fill and color
```

```
scale_fill_manual(name = "", values = c("95% Confidence Interval" = "gray")) +
scale_color_manual(name = "", values = c("Actual Price" = "blue")) +
# Add titles and labels
labs(
  title = "Actual vs. Predicted Stock Prices for JNJ Using MCMC",
  x = "Days",
  y = "Stock Price"
) +
# Apply a minimal theme for simplicity
theme_minimal() +
# Center the plot title and position the legend at the top
theme(
  plot.title = element_text(hjust = 0.5),
  legend.position = "top"
)
```

References

[1] Vasile Brătian, Ana-Maria Acu, Diana Marieta Mihaiu, and Radu-Alexandru Șerban. “Mathematical Modeling and Analysis of XYZ Phenomena”. In: *Mathematics* 10.3 (2022), pp. 9–15. DOI: 10.3390/math10030309. URL: https://www.mdpi.com/2227-7390/10/3/309.

[2] Angeliki Ermogenous. “Brownian Motion and its applications in the Stock Market”. In: (2006).

[3] Yahoo Finance. *Johnson & Johnson (JNJ) Stock Historical Prices & Data*. Accessed: 2024-11-29. 2024. URL: https://finance.yahoo.com/quote/JNJ/history/.

[4] GeeksforGeeks. *Shading Confidence Intervals Manually with ggplot2 in R*. Accessed: 2024-12-03. 2024. URL: https://www.geeksforgeeks.org/shading-confidence-intervals-manually-with-ggplot2-in-r/.

[5] Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. *Bayesian data analysis*. Chapman and Hall/CRC, 1995.

[6] Gregory F Lawler. *Introduction to stochastic processes*. Chapman and Hall/CRC, 2018.

[7] Pedro Mota. “Normality assumption for the log-return of the stock prices”. In: *Discussiones Mathematicae Probability and Statistics* 32.1-2 (2012), pp. 47–58.

[8] Chelsea Peng and Colette Simon. “Financial Modeling with Geometric Brownian Motion”. In: *Open Journal of Business and Management* 12.2 (2024), pp. 1240–1250.

[9] Marcelo Perlin. *yfR: Downloads and Organizes Financial Data from Yahoo Finance*. Accessed: 2024-11-30. rOpenSci. 2024. URL: https://docs.ropensci.org/yfR/.

[10] Axel Skaven Rygel. *egg: Extensions for ’ggplot2’: Ecosystem Vignette*. Accessed: 2024-12-03. CRAN. 2024. URL: https://cran.r-project.org/web/packages/egg/vignettes/Ecosystem.html.

[11] H Ait Saadi, F Ykhlef, and A Guessoum. “Mcmc for parameters estimation by bayesian approach”. In: *Eighth International Multi-Conference on Systems, Signals & Devices*. IEEE. 2011, pp. 1–6.

[12] Christof Seiler. “Bayesian statistics in computational anatomy”. In: *Statistical Shape and Deformation Analysis*. Elsevier, 2017, pp. 193–214.

[13] Stan Development Team. *RStan: The R Interface to Stan*. Accessed: 2024-12-03. CRAN. 2024. URL: https://cran.r-project.org/web/packages/rstan/vignettes/rstan.html.

[14] Caesar Wu and Rajkumar Buyya. “Chapter 18 - Real Option Theory and Monte Carlo Simulation”. In: *Cloud Data Centers and Cost Modeling*. Ed. by Caesar Wu and Rajkumar Buyya. Morgan Kaufmann, 2015, pp. 707–772. ISBN: 978-0-12-801413-4. DOI: https://doi.org/10.1016/B978-0-12-801413-4.00018-0. URL: https://www.sciencedirect.com/science/article/pii/B9780128014134000180.

[15] Taisuke Yamada, Keitaro Ohno, and Yusaku Ohta. "Comparison between the Hamiltonian Monte Carlo method and the Metropolis–Hastings method for coseismic fault model estimation". In: *Earth, Planets and Space* 74.1 (2022), p. 86.

[16] Hao Zhu. *kableExtra: A package for elegant and complex tables in R Markdown.* Accessed: 2024-12-03. 2024. URL: https://github.com/haozhu233/kableExtra?tab=readme-ov-file.