University of Liège

Financial Mathematics and Stochastic Calculus

Simulation of a stock price as a geometric brownian motion: project report

Author:
DUYSINX Antoine

Professor:
C. Paquay

April 2021

Contents

1 Introduction

In this work, a Geometric Brownian Motion (GBM) simulation will be done on future stock prices of the Procter & Gamble (ticker "PG") company, which sells a large range of consumer products all over the world and is quoted on the New-York Stock Exchange (NYSE) since January 13, 1978. After the simulation, we will provide an estimation of the expected stock price at a given point in time, as well as confidence interval to evaluate the precision of our forecast. Finally, we will comment how the results of the simulation are impacted if we change the value of some parameters, like the drift, the volatility rate or the forecasting horizon.

2 Data wrangling

2.1 Libraries

When writing the R script, two libraries were used, namely *randomcoloR* and *MASS*. These libraries must be installed and loaded before executing the code otherwise some errors will occur. To do this, the two following R commands can be used:

```r
install.packages("randomcoloR")
install.packages("MASS")
library(randomcoloR)
library(MASS)
```

2.2 Importing and getting to know the data

First of all, for the purpose of our simulation, we need past data to estimate the GBM parameters. As a consequence, one year of stock price data (from January 1, 2021 to December 31, 2021) is retrieved from Yahoo Finance as a CSV file. If needed raw data can be found here.
This data set contains 7 variables and 251 rows, which corresponds approximately to the opening days of the stock exchange. If we take a closer look to the variables we have,

- *Date*: Date of the trading day.
- *Open*: Share price at the opening of the stock exchange on a particular trading day.
- *High*: Highest value that the stock price took on a particular trading day.
- *Low*: Lowest value that the stock price took on a particular trading day.
- *Close*: Share price at the close of the stock exchange on a particular trading day.
- *Adj.Close*: Adjusted close is the closing price after adjustments for all applicable splits and dividend distributions.
- *Volume*: Number of a stock's shares that are traded on the stock exchange in a day.

Once familiar with the data description, we can import them in R. In the extent of this work, only adjusted close price and dates matter, therefore other columns will be dropped.

2.3 Evolution of the stock price

Adjusted close prices can be plot on an time series graph. It allows us to visualize how the stock price has evolved over the last year.

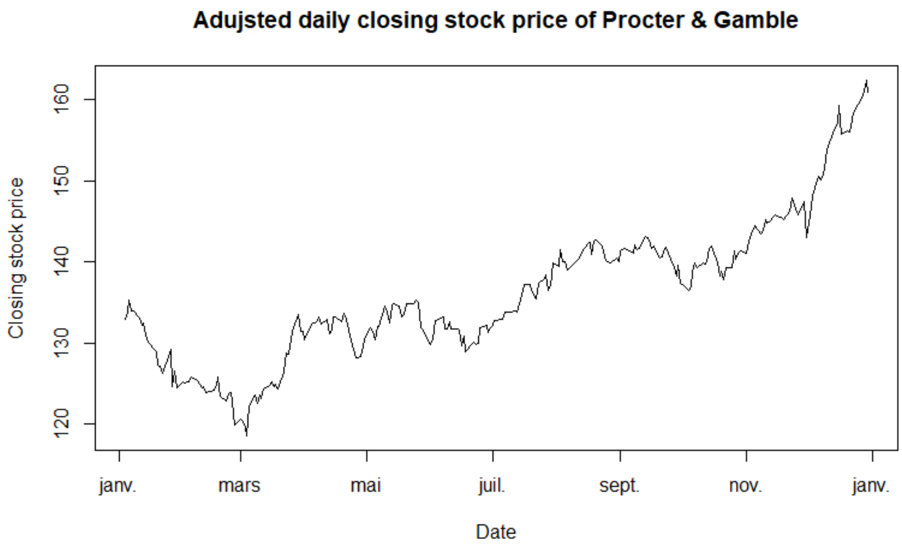

Figure 1: Adjusted daily closing price of PG from 01/01/2021 to 12/31/2021

As it can be easily seen, 2021 was clearly a thriving year for Procter & Gamble.

2.4 Daily log-returns

The next step in our data preparation is the computation of the daily log-returns. Indeed it is needed to estimate afterwards the GBM parameters. The log-returns at time t, denoted r_t, are defined as

$$r_t = ln(\frac{S_t}{S_{t-1}}) = ln(S_t) - ln(S_{t-1}) \tag{1}$$

where S_t and S_{t-1} are the closing prices of current and previous date respectively. Their computation is relatively straightforward in R. First, we have to log-transform the adjusted close price by applying the ln function. Then, we simply take the difference between the value at time t and the previous observation, also called the *first lag*. The time series of the log-returns that we finally obtain, can be plotted as shown below (figure 2). By looking at this chart, we can observe that the time series of the log-returns is quite stationary, even thought some pikes of volatility are still present.

It is often customary to assume that the log-returns are normally distributed. However, in practice it is not always a good assumption. Then, it might be worth inspecting the distribution of the log-returns. To this end, we have at our disposal several tools, like goodness-of-fit tests or graphical comparisons with theoretical known distributions. In this work, we will use the latter category.

Thus, we plot an histogram of the log-returns against a normal density which has same mean and standard deviation than the log-returns itself and a Q-Q plot (figure 3). On

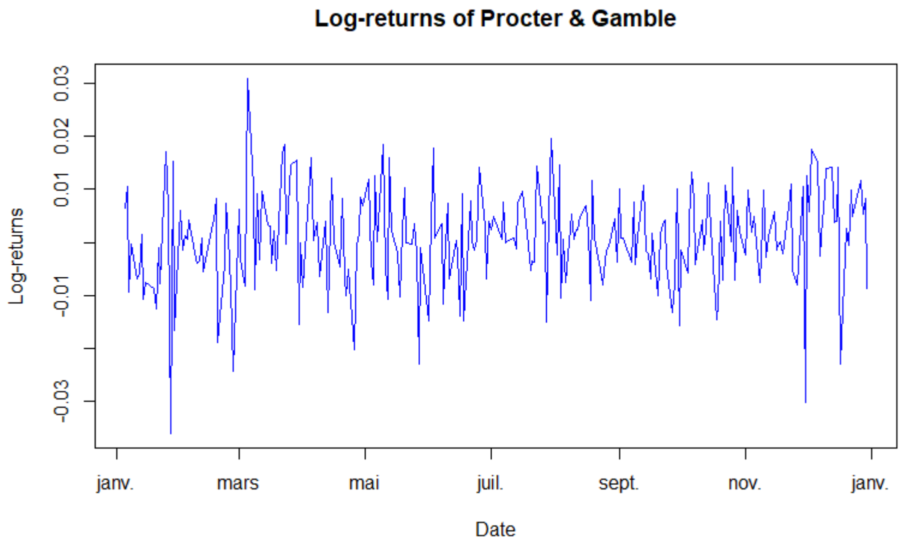

Figure 2: Time series of log-returns from 01/01/2021 to 12/31/2021

both, we can distinguish that the distribution of the log-returns is close to a normal distribution although it has a fatter left-tail and a higher concentration of the observations around the mean. In other words, it means that extreme losses or very negative returns are more likely than under a complete normality assumption.

As a conclusion, we can say that the returns are approximately normally distributed with a mean $\mu \Delta t$ and a standard deviation $\sigma \sqrt{\Delta t}$. Δt corresponding to the number of discrete periods between two points in time.

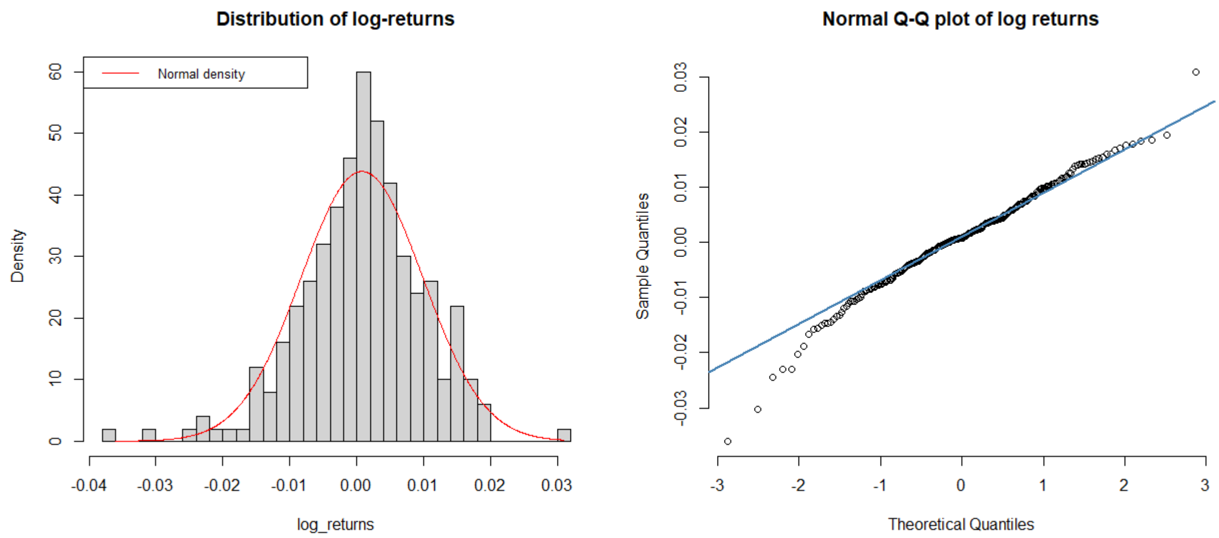

Figure 3: Distribution of log-returns

3 Geometric Brownian Motion

First of all, it seems important to recall how a Geometric Brownian Motion is defined. The stochastic process S_t (here the stock price) follows a Geometric Brownian Motion if it satisfies the following stochastic differential equation:

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

Where W_t is a *Wiener process* or a *standard brownian motion*, μ is called *the drift* and σ is called *the volatility*. The right-hand side term $\mu S_t dt$ controls the *trend* in the trajectory while the left-hand side term $\sigma S_t W_t$ is the *"random noise"* or the unpredictable effect in the path's trajectory. In fact, it is this stochastic differential equation that is defined as a Geometric Brownian Motion.

The solution of this differential equation can be computed and has the following form:

$$S_t = S_0 exp((\mu - \frac{\sigma^2}{2})t + \sigma W_t) \tag{3}$$

If we estimate or provide a value for the constant μ and σ, we are able to produce a Geometric Brownian Motion solution throughout time interval. In this work, we are going to estimate μ with the historical mean returns and σ with the historical standard deviation of the returns.
The mean return realized over a period of time of length T is given by:

$$\mu = \frac{1}{T}\sum_{t=1}^{T} r_t \tag{4}$$

While, the usual estimate of the standard deviation of the returns over a period of time of length T is given by:

$$\sigma = \sqrt{\frac{1}{T-1}\sum_{t=1}^{T}(r_t - \mu)^2} \tag{5}$$

In R, we can use the *mean()* and *sd()* functions. Thus it returns,

- $\mu = 0.07\%$ per day or $\mu \times 252 = 19.27\%$ per annum (assuming that there is 252 trading days per year).
- $\sigma = 2.13\%$ per day or $\sigma\sqrt{252} = 14.44\%$ per annum.

4 Simulation of Geometric Brownian Motion paths

In this section, we are going to simulate 15 possible paths for the future stock prices over the coming trading month. The time horizon considered is therefore $1/12 \times 252 = 21$ days. Given the solution of the stochastic differential equation (equation 3), we are able to simulate or generate function of it representing future paths. As mentioned earlier, we can split this equation into two parts. On the one hand, the trend, $(\mu - \frac{\sigma^2}{2})t$, which is constant over time whatever the path, and on the other hand, the random part, σW_t, which needs to be simulated by sampling repeatedly W_t and hence always varies.

By definition of a Wiener process, we know that $W_{t+\Delta t} - W_t \sim N(0, \Delta t)$. Therefore, if we consider $\Delta t = 1$ for intervals of 1 day, the increments ΔW_t can be generated as a realization of a normal distribution $N(0, 1)$. We generate as many increments as necessary to reach the time horizon considered, that is in this case 21. These increments represents the random variations in the stock price and without them, the stock price would be entirely controlled by the trend and all the paths would surely heading toward the same direction. Using,

$$S_t = S_0 exp((\mu - \frac{\sigma^2}{2})t + \sigma W_t) \qquad \forall t \in \{1, 2, ..., 21\}$$

and by injecting the proper value of W_t, we can estimate the stock price for each day. The simulation can be summarized in the following table:

Time	Increments	Wt	St
0		$W_0 = 0$	$S_0 = 160.99$
1	$\Delta W_1 = -0.2734$	$W_1 = W_0 + \Delta W_1$	$S_1 = S_0 exp((\mu - \frac{\sigma^2}{2}) \times 1 + \sigma W_1) = 159.61$
2	$\Delta W_2 = -0.4705$	$W_2 = W_1 + \Delta W_2$	$S_2 = S_0 exp((\mu - \frac{\sigma^2}{2}) \times 2 + \sigma W_2) = 156.29$
...	ΔW_t	$W_t = W_{t-1} + \Delta W_t$	$S_t = S_0 exp((\mu - \frac{\sigma^2}{2})t + \sigma W_t)$

Once we got a simulated vectors of stock prices, we can plot them to visualize different paths:

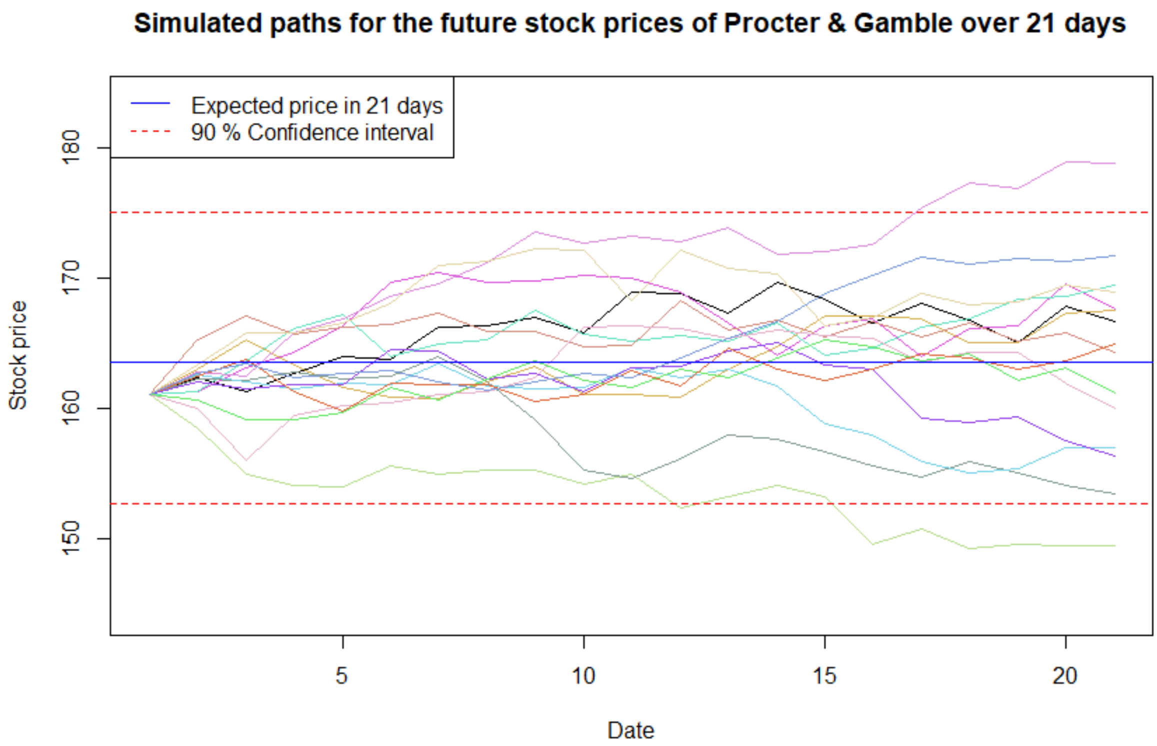

Figure 4: Simulated GBM paths over 21 days

5 Expected price after one month

From equation 3, if we apply ln on both side, we know that prices are normally distributed.

$$ln(S_t) = ln(S_0) + (\mu - \frac{\sigma^2}{2})t + \sigma W_t \sim N(ln(S_0) + (\mu - \frac{\sigma^2}{2})t, \sigma^2 t) \tag{6}$$

As a consequence, we can compute their expectation:

$$E(S_t) = S_0 exp(ln(S_0) + (\mu - \frac{\sigma^2}{2})t + \frac{\sigma^2 t}{2})$$

$$E(S_t) = S_0 exp(\mu t) \tag{7}$$

By simply plugging the value of μ, t and S_0 into this relation we easily obtain the expected stock price in one month.

$$E(S_{21}) = 160.99 \times exp(0.07\% \times 21) = \$163.59$$

Therefore, the expected stock price of Procter & Gamble in 21 days is \$163.59.

6 Confidence interval

Since prices are normally distributed, we can infer some confidence intervals. Again, consider the same time horizon, namely 21 days or 1 month. However, before that we need to compute the normal parameters for that time horizon:

- mean $= ln(S_0) + (\mu - \frac{\sigma^2}{2})t = ln(160.99) + (0.07\% - \frac{(2.13\%)^2}{2}) \times 21 = 5.0965$
- standard deviation (sd) $= \sqrt{\sigma^2 t} = \sqrt{(2.13\%)^2 \times 21} = 0.0416$

Therefore, $ln(S_{21}) \sim N(5.0965, 0.0416^2)$

Let's then construct a 90%-confidence interval for the stock price of Procter & Gamble after 21 days.
The confidence interval can be easily obtained with the following formula:

$$mean - q_{1-\alpha/2} \times sd < ln(S_{21}) < mean + q_{1-\alpha/2} \times sd$$

Where $q_{1-\alpha/2}$ is the quantile $1 - \alpha/2$ of the standard normal distribution.
If we replace the different elements by their corresponding value, we get:

$$5.0965 - 1.6448 \times 0.0416 < ln(S_{21}) < 5.0965 + 1.6448 \times 0.0416$$

Finally, by applying the exponential function, we end up with:

$$152.6227 < S_{21} < 175.0612$$

Thus, there is a probability of 90% that the stock price of Procter & Gamble will lie between \$152.6227 and \$175.0612 in 21 days. This two-sided confidence interval can be visualized on figure 4.

7 Impact of the time horizon considered

If we had considered another time horizon for the past data, the parameters of the GBM would have had significant different values than they do for 1 year. For example, if we reduce the past data to only the last six months of 2021, the drift and the volatility would have higher value than for 1 year. Most of the time, when we reduce the time horizon of past data, volatility increases, while the mean return either rises or decreases depending on whether the recent performances of the stock were good or not. However, in the long run, mean return has often a more stable value.
As a consequence of a higher volatility and a higher drift, we get larger confidence intervals and a higher expected price. Therefore, it clearly reduces the precision of the prediction.

8 Impact of forecasting prices over the coming 3 months

Predicting stock prices over 3 months instead of a single one, leads to different results.
First of all, the expected stock price 63 days is now \$168.94 instead of \$163.59. This increase results from the effect of the positive trend and the longer time horizon considered. Then, the confidence interval is also modified. Now, the stock price will lie between 149.63 and 189.75 with a probability of 90%. As we can notice, a longer forecast horizon widens the confidence interval. It comes from the fact that the distribution parameters of the stock price are dependant and increase with the time horizon considered (see equation 6). In other words, stock prices have non stationary normal distributions than vary with time, leading to different confidence intervals.
To conclude, it is worth noting that the further we forecast in time, the less precise we are.

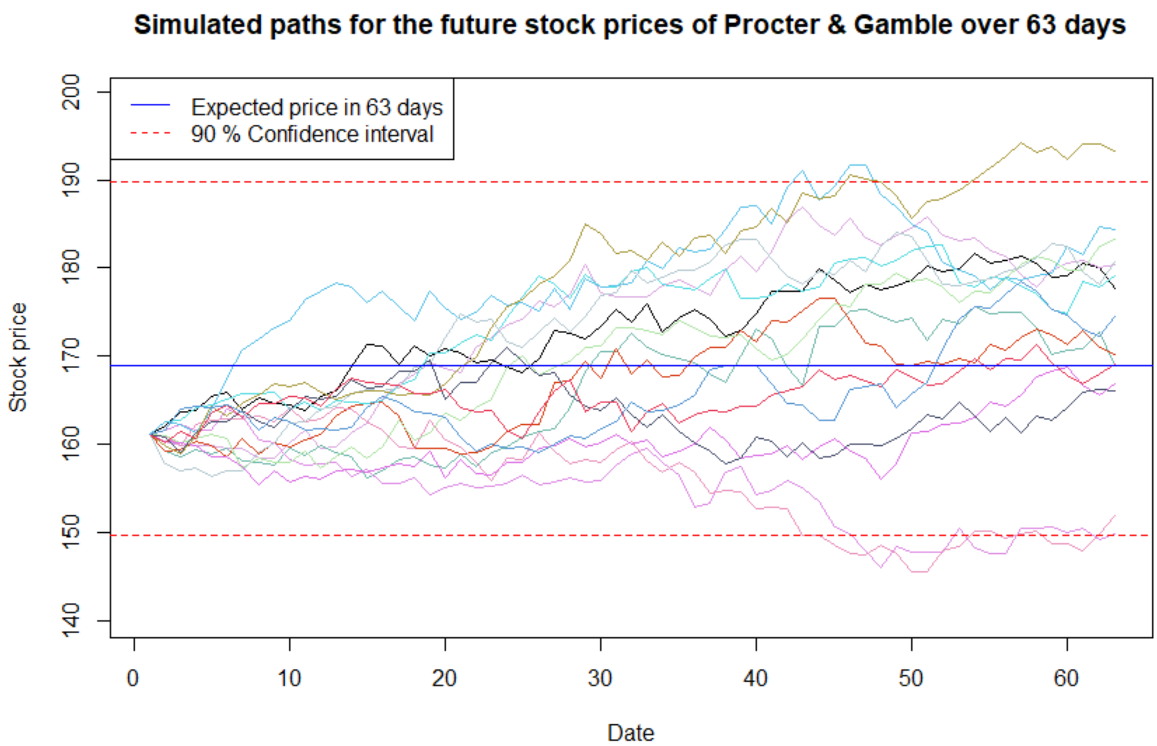

Figure 5: Simulated GBM over 63 days

9 Conclusion

As a conclusion for this report, it might be really interesting to compare what was the real stock price at January 31, 2022 versus the forecasted price by our model.

According to the Yahoo Finance website, the stock price of Procter & Gamble was 160.45. This value is comprised into our confidence interval but is a little bit lower than our expected value. Obviously, since the Geometric Brownian Motion is based on past data, it does not take into account the tough economic conditions that we face today. This is where the weakness of the model lies.

10 Appendix

A R Code

```
rm(list = ls())

#loading the needed packages
library(randomcoloR)
library(MASS)

#Setting up the active directory. Put your path here!
setwd("~/Unif/MASTER 1 IG/Financial mathematics & stochastic
   calculus/Project")

##### TASK 1 #####

#Importing the data
PG <- read.csv(file = 'PG.csv', header = TRUE, sep=",")

#Inspecting the first lines of the data set
head(x = PG, n = 6)

#Isolating the adjusted close price and the dates
prices <- as.data.frame(PG[,c('Date','Adj.Close')])

colnames(prices) <- c("Date","Adj.Close")

prices$Date <- as.Date(prices$Date)

head(x = prices, n = 6)

##### TASK 2 #####

#plotting the adjusted stock prices
plot(
  x = prices$Date,
  y = prices$Adj.Close,
  main = "Daily adjusted closing stock price of Procter & Gamble",
  ylab = "Closing adjusted stock price",
  xlab = "Date",
  type = "l"
)

##### TASK 3 #####

#Computing the log-returns
log_prices <- log(x = prices$Adj.Close)
log_returns <- diff(x = log_prices, lag=1)

#drop the first line since the log-return is not available
log_returns <- na.omit(log_returns)
```

```r

#plot time series of log-returns
plot(x = prices[2:dim(prices)[1],1],
     y=log_returns,
     type = 'l',
     col = 'blue',
     main = "Log-returns of Procter & Gamble in 2021",
     ylab = "Log-returns",
     xlab = "Dates")

#Histogram of log-returns and study their distribution
x<-seq(from=min(log_returns),to=max(log_returns),length.out = 1000)

par(mfrow=c(1,2))

hist(log_returns,
     nclass = 30,
     probability = TRUE,
     main='Distribution of log-returns')

legend("topleft",
       legend = c("Normal density"),
       col=c("red"),
       lty=1,
       cex=0.8)

lines(x=x,
      y=dnorm(x,mean(log_returns),sd(log_returns)),
      col="red")

#Building a QQ plot for log-returns
qqnorm(log_returns,pch=1,frame=FALSE, main='Normal Q-Q plot of log returns')

qqline(log_returns, col="steelblue", lwd=2)

par(mfrow=c(1,1))

##### TASK 4 #####

#Estimating parameters of the GBM
mu <- mean(log_returns)#expected rate of return
sigma <- sd(log_returns)#volatility

##### TASK 5 #####

#Simulating paths
#Creation of a general function to simulate stock prices path during n days
simuGBMpath <- function(n,p,s_zero,drift,volatility){

  # p = number of path
```

```
  # n = number of days
  # s_zero = numeric value = last stock price value

  w ← matrix(data = 0, nrow = n+1, ncol = p) #empty matrix to stock
      simulated Wj
  s ← matrix(data = NA, nrow = n, ncol = p) #empty matrix to stock
     simulated prices
  random ← rep(x = NA, times=n) # empty vector to stock random
     numbers
  my_col_names ← rep(x = NA, times=p)


  for(i in 1:ncol(w)){
    random ← rnorm(n=n, mean = 0, sd = sqrt(1))
    my_col_names[i] ← paste("Path",i,sep="")
    for(j in 1:(nrow(w)-1)){
      w[j+1,i] ← w[j,i] + random[j]
      s[j,i] ← s_zero*exp((mu-((sigma^2)/2))*j+sigma*w[j,i])
    }
    colnames(s) ← my_col_names
    colnames(w) ← my_col_names
  }

  #Plotting simulated paths
  colors ← distinctColorPalette(k=n-1) #generating n-1 different
     colors

  plot(x=seq(1,n, by=1), y=s[,1],
       ylim=c(min(s)-5,max(s)+10),
       type = 'l',
       xlab = "Date",
       ylab = "Stock price",
       main = paste("Simulated paths for the future stock prices of
           Procter & Gamble over", n, "days"))

  for(i in 2:ncol(s)){
    lines(x=seq(1,n, by=1), s[,i], col=colors[i])
  }
}

#calling the function to simulate stock prices over 1 month
s_zero ← as.numeric(prices[dim(prices)[1],dim(prices)[2]])
p ← 15 #number of paths
n ← 252*(1/12) #number of simulation days

simuGBMpath(n = n, p = p, s_zero = s_zero, drift = mu, volatility =
    sigma)

##### TASK 6 #####

#Definition of a function to calculate the expected stock price
   after n days
```

```
ExpectedPrice ← function(drift, n, s_zero){
  x ← s_zero * exp(drift*n)
  abline(h = x, col='blue') #adding a horizontal line on the previous plot
  return(print(paste("The expected price after", n, "days is", x)))
}

#Expected price after one month
ExpectedPrice(drift = mu, n = n, s_zero = s_zero)


##### TASK 7 #####

#Definition of a function to calculate confidence intervals
CI ← function(drift, volatility, s_zero, n, conflevel){

 #Parameters of the normal distribution that the stock prices follow at time t=n
 mean ← log(s_zero)+(mu-((sigma)^2)*0.5)*n
 variance ← (sigma^2)*n
 sd ← sqrt(variance)

 #Computing confidence interval
 alpha ← 1-conflevel

 quantile ← qnorm(1-alpha/2, mean = 0, sd = 1) #quantile 1-alpha/2 of the standard normal distribution

 lower_bound ← exp(mean-quantile*sd)

 upper_bound ← exp(mean+quantile*sd)

 print(paste("There is a probability of",conflevel*100,"% that the stock price will lie between",
             lower_bound,"and",upper_bound, "in", n, "days"))

 #adding the upper and lower bound on the previous plot
 abline(h = lower_bound, col="red",lty=2)
 abline(h = upper_bound, col="red",lty=2)

 legend(x="topleft",legend=c(paste("Expected price in",n,"days"), paste(conflevel*100,"%","Confidence interval")), col=c("blue", "red"), lty= c(1,2), lwd(5,1))
}

#Confidence interval at level 0.9 for 21 days of simulation
conf ← 0.90 #confidence level
CI(drift = mu, volatility = sigma, s_zero = s_zero, n = n, conflevel = conf)

##### TASK 8 #####
```

```r
log_returns2 <- log_returns[125:250]
mu2 <- mean(log_returns2)
sigma2 <- sd(log_returns2)

simuGBMpath(n = 21, p = p, s_zero = s_zero, drift = mu2, volatility = sigma2)
ExpectedPrice(drift = mu2, n = 21, s_zero = s_zero)
CI(drift = mu2, volatility = sigma2, s_zero = s_zero, n = 21, conflevel = conf)
#clearly we see larger volatility and thus larger confidence interval
##### TASK 9 #####

#Forecasting prices over the coming 3 months or 63 days
simuGBMpath(n = 63, p = p, s_zero = s_zero, drift = mu, volatility = sigma)
ExpectedPrice(drift = mu, n = 63, s_zero = s_zero)
CI(drift = mu, volatility = sigma, s_zero = s_zero, n = 63, conflevel = conf)
```