Project #1: On returns and portfolios

Samuel Kalisch

2024-02-12

```
library(nimble)
```

Problem #1 (30 points)

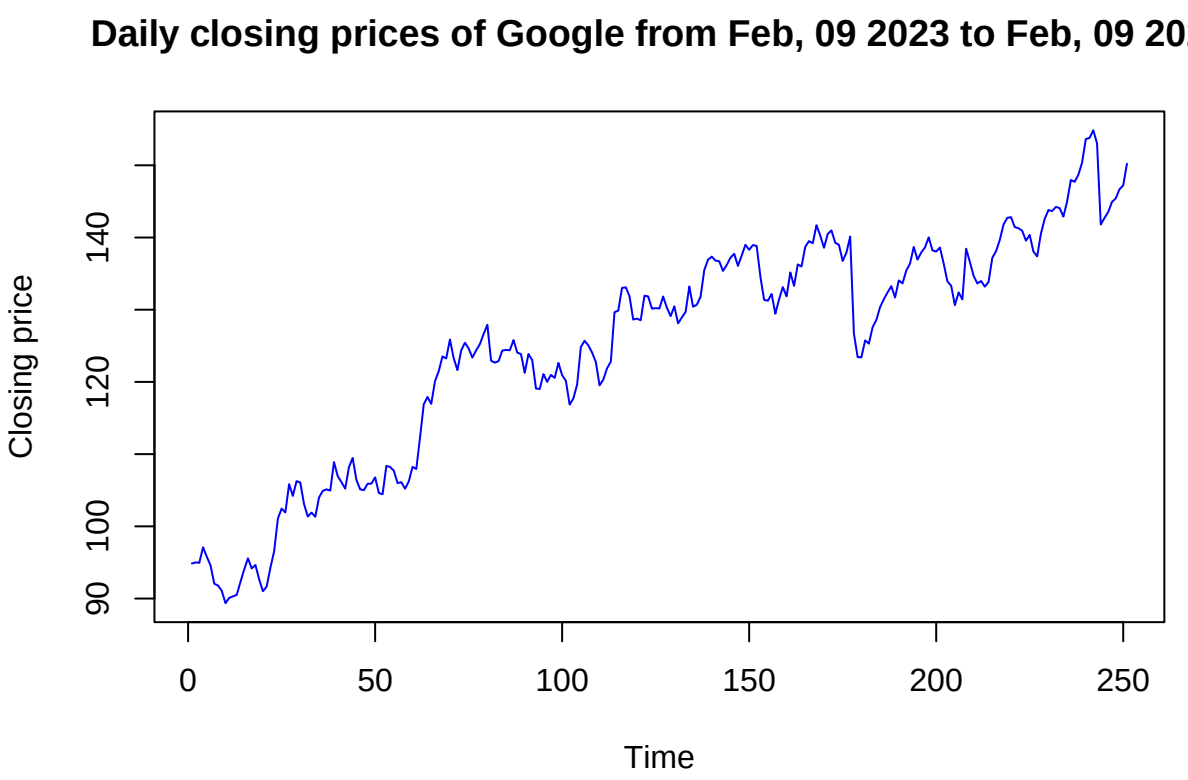
Go to Google historical data

Then, download the historical stock prices of *Alphabet, Inc.* for the last 252 (or so) trading days, i.e., for the last year.

```
goog<-read.csv("GOOG.csv")
#head(goog)
```

(5 points) Draw the time-plot of the evolution of the closing stock price (not the adjusted). You do **not** need to put the calendar days on the horizontal axis, but you **do** need to label your axes and give your time-plot a title indicating the dates.

```
close.price=goog$Close
head(close.price)
## [1] 94.86 95.00 94.95 97.10 95.78 94.59
plot(close.price,
     main="Daily closing prices of Google from Feb, 09 2023 to Feb, 09 2024",
     xlab= "Time",
     ylab = "Closing price",
     pch= 20,
     type="l",
     col = "blue")
```

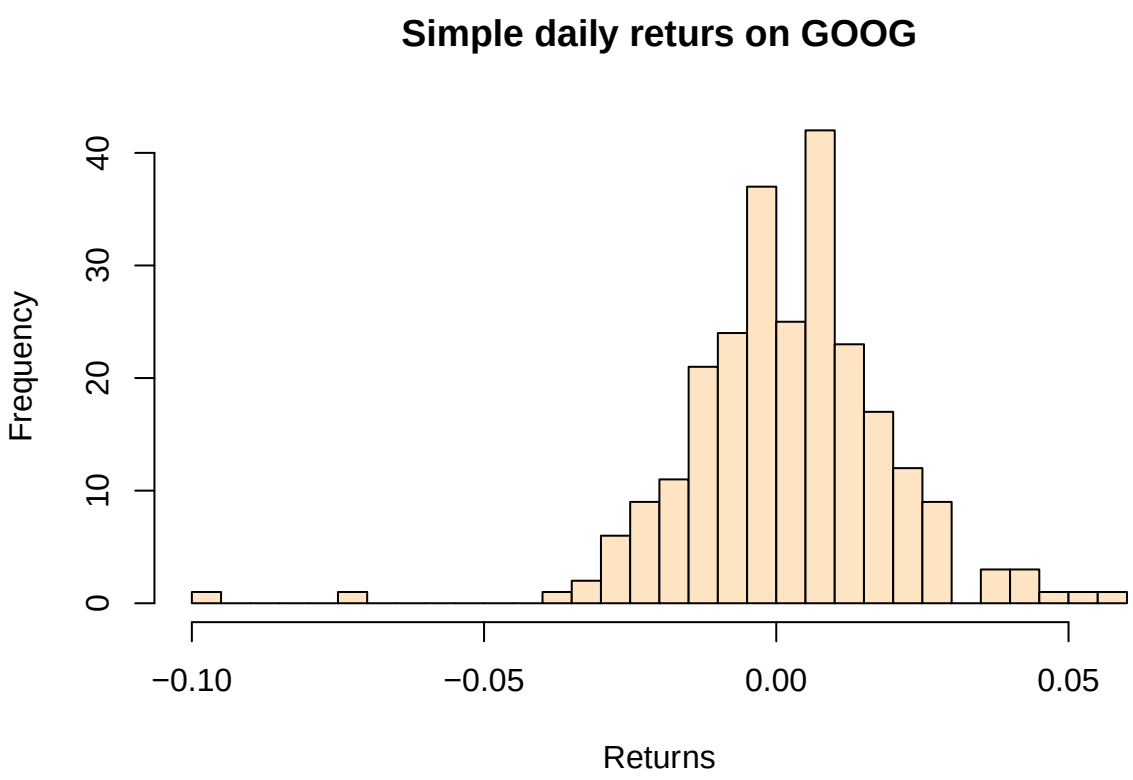


The **simple daily return** of the stock over a day indexed by t is defined as

$$\frac{\text{price at end of day } t - \text{price at end of day } (t-1)}{\text{price at end of day } (t-1)}$$

(5 points) Construct the vector of simple daily returns over the last year. Provide a visualization of the returns. What can you say about the characteristics of the distribution based on the above plot?

```
close.price.end=close.price[-1]
close.price.beg=close.price[-length(close.price)]
returns=(close.price.end-close.price.beg)/close.price.beg
#head(returns)
hist(returns,breaks=25,
     main="Simple daily returs on GOOG",
     xlab="Returns",
     col = "bisque")
```



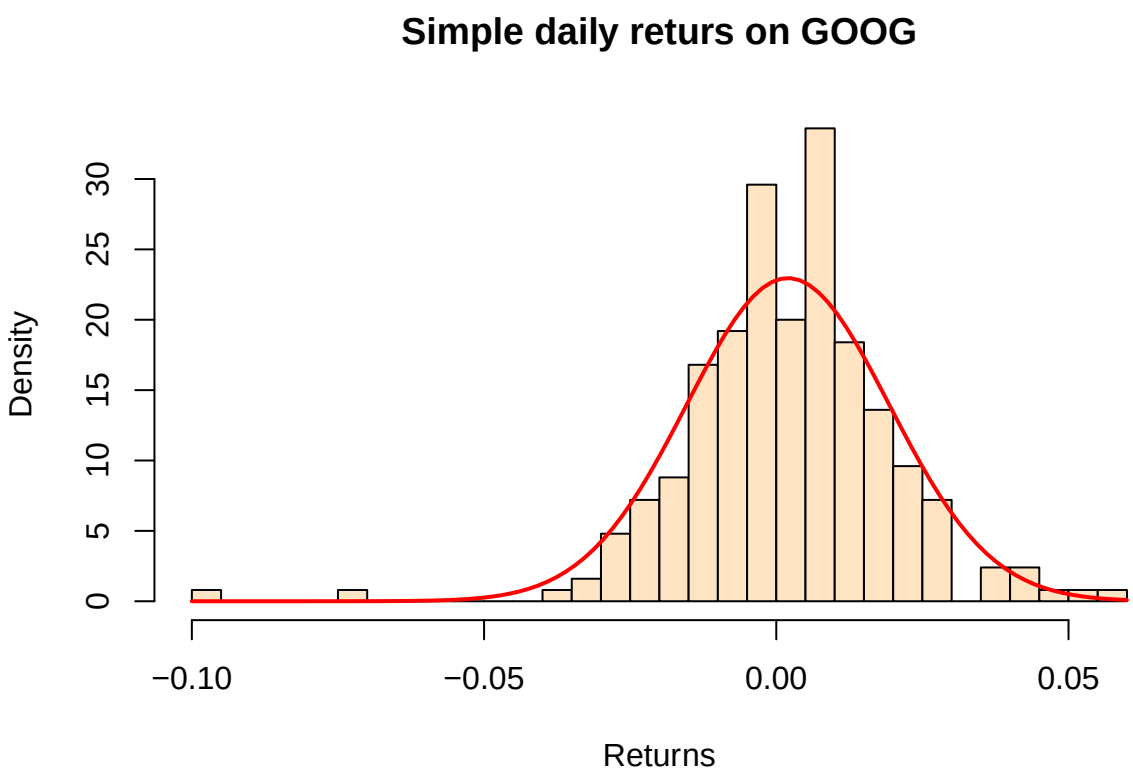
Based on the above plot, GOOG returns look fairly normally distributed with the a couple of outliers, especially in the negative side, this is expected as the movement of the stock is failty random on a given days with some days like earnings reports or where a news is announced that can shift this randomess into a big increase or decrease on the price of the stock.

(10 points) Fit the normal distribution to the above returns. Superimpose the appropriate graph for your model onto the appropriate graph of the data to convince your reader that your model is valid.

```
mu_hat=mean(returns)
sigma_hat= sd(returns)

hist(returns,breaks=25,
     main="Simple daily returs on GOOG",
     xlab="Returns",
     col = "bisque",
     prob=TRUE)

curve(dnorm(x, mean = mu_hat, sd = sigma_hat), col="red", lwd=2, add=TRUE)
```



(20 points) You will have to install a package to solve this part of your project. First, run the following in your console:

`install.packages('nimble')`

When that is finished, you need to uncomment

`library(nimble)`

from the first chunk in this document.

Next, you should learn more about the **Laplace (double exponential)** distribution. This is easily done by visiting:

Wikipedia: The Laplace distribution

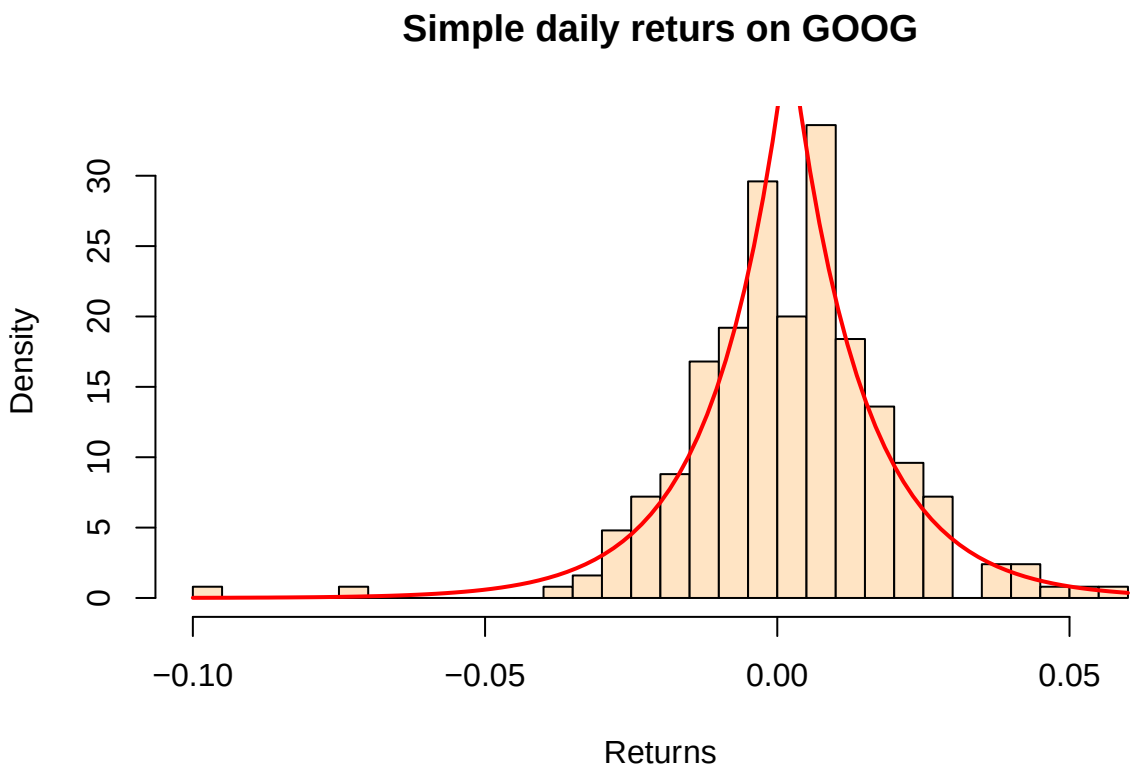
Now, you are equipped to fit the Laplace (double exponential) distribution to the above returns. To learn about the parametrization of the Laplace distribution in `R`, type `?ddexp` into the console in RStudio.

After you have completed the fit, superimpose the appropriate graph for your model onto the appropriate graph of the data.

```
b_hat= sigma_hat/sqrt(2)

hist(returns,breaks=25,
     main="Simple daily returs on GOOG",
     xlab="Returns",
     col = "bisque",
     prob=TRUE)
```

```r
curve(ddexp(x, location = mu_hat, scale = b_hat), col="red", lwd=2, add=TRUE)
```



Problem #2 (70 points)

Your initial wealth is exactly \$100. You are allowed to invest in shares of a particular stock. You are also allowed to both lend and borrow at the continuously compounded risk-free interest rate of 0.05. Keeping your money uninvested is **not allowed**.

You can rebalance your portfolio every morning, once you have observed the opening stock price. This means that you can change the number of shares you own (if you decide to do so) and accordingly adapt your risk-free investment.

You proceed to create a "rule'' according to which you will be rebalancing your portfolio. Here are some possible rules you can use:

i. **(10 points)** Always own exactly one share and keep the rest of my wealth invested at the risk-free rate. If needed, borrow at the continuously compounded risk-free interest rate.

ii. **(20 points)** Always keep exactly \$100 in the stock and the rest of my wealth invested at the risk-free rate. If needed, borrow at the continuously compounded risk-free interest rate.

iii. **(30 points)** Always keep exactly one half of my wealth invested in the stock and the rest of my wealth invested at the risk-free rate. If needed, borrow at the continuously compounded risk-free interest rate.

Over the following 10 days, you observe the following stock prices for a non-dividend-paying stock:

Day	0	1	2	3	4	5	6	7	8	9	10
Stock price	100	80	64	80	64	80	100	80	64	80	100

As the time passes you follow investment rules above to rebalance your portfolio. Complete the following table describing your portfolio **just before and just after** the rebalancing is done. Even more precisely, for the 10 days, both for "before" and "after" the rebalancing, print out:

- the number of shares of stock in the portfolio,
- the balance of the risk-free investment,
- the wealth in the stock,
- the total wealth.

Solution

First take note of the continuously compounded, risk-free interest rate:

```
r=0.05
```

Then, I create a vectore containing the evolution of the stock price:

```
s= c(100, 80, 64, 80, 64, 80, 100, 80, 64, 80, 100)
```

Rule 1:

```
#set up vectors
pi.v=numeric(11)
cash=numeric(11)
wealth=numeric(11)

#initialize
wealth[1]=100
pi.v[1]= 1
cash[1]=wealth[1]-pi.v[1]*s[1]


#dynamics of the rule
rebalance<-function(s.price){
  return(1)
  }

#we move forward thorugh time with this rule
for(i in 1:10){
  cash[i+1]=cash[i]*exp(r/365)
  wealth[i+1]=cash[i+1]+pi.v[i]*s[i+1]

  #before re-balancing
  print(sprintf("Day=%d, %s", i, "before"))
  print(sprintf("shares=%6.4f, cash=%6.4f, wealth in stock=%6.4f, total wealth=%6.4f",
                pi.v[i], cash[i+1], pi.v[i]*s[i+1], wealth[i+1]))

  #now we re-balance
  pi.v[i+1]=rebalance(s[i+1])
  cash[i+1]=wealth[i+1]-pi.v[i+1]*s[i+1]
```

```
  #after re-balancing
  print(sprintf("Day=%d, %s", i, "after"))
  print(sprintf("shares=%6.4f, cash=%6.4f, wealth in stock=%6.4f, total wealth=%6.4f",
                pi.v[i+1], cash[i+1], pi.v[i+1]*s[i+1], wealth[i+1]))
}
## [1] "Day=1, before"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=80.0000, total wealth=80.0000"
## [1] "Day=1, after"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=80.0000, total wealth=80.0000"
## [1] "Day=2, before"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=64.0000, total wealth=64.0000"
## [1] "Day=2, after"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=64.0000, total wealth=64.0000"
## [1] "Day=3, before"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=80.0000, total wealth=80.0000"
## [1] "Day=3, after"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=80.0000, total wealth=80.0000"
## [1] "Day=4, before"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=64.0000, total wealth=64.0000"
## [1] "Day=4, after"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=64.0000, total wealth=64.0000"
## [1] "Day=5, before"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=80.0000, total wealth=80.0000"
## [1] "Day=5, after"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=80.0000, total wealth=80.0000"
## [1] "Day=6, before"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=100.0000, total wealth=100.0000"
## [1] "Day=6, after"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=100.0000, total wealth=100.0000"
## [1] "Day=7, before"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=80.0000, total wealth=80.0000"
## [1] "Day=7, after"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=80.0000, total wealth=80.0000"
## [1] "Day=8, before"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=64.0000, total wealth=64.0000"
## [1] "Day=8, after"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=64.0000, total wealth=64.0000"
## [1] "Day=9, before"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=80.0000, total wealth=80.0000"
## [1] "Day=9, after"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=80.0000, total wealth=80.0000"
## [1] "Day=10, before"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=100.0000, total wealth=100.0000"
## [1] "Day=10, after"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=100.0000, total wealth=100.0000"
```

Rule 2:

```
#set up vectors
pi.v=numeric(11)
cash=numeric(11)
wealth=numeric(11)
```

```r
#initialize
wealth[1]=100
pi.v[1]= 100/s[1]
cash[1]=wealth[1]-pi.v[1]*s[1]


#dynamics of the rule
rebalance<-function(s.price){
  return(100/s.price)
  }

#we move forward thorugh time with this rule
for(i in 1:10){
  cash[i+1]=cash[i]*exp(r/365)
  wealth[i+1]=cash[i+1]+pi.v[i]*s[i+1]

  #before re-balancing
  print(sprintf("Day=%d, %s", i, "before"))
  print(sprintf("shares=%6.4f, cash=%6.4f, wealth in stock=%6.4f, total wealth=%6.4f",
                pi.v[i], cash[i+1], pi.v[i]*s[i+1], wealth[i+1]))

  #now we re-balance
  pi.v[i+1]=rebalance(s[i+1])
  cash[i+1]=wealth[i+1]-pi.v[i+1]*s[i+1]

  #after re-balancing
  print(sprintf("Day=%d, %s", i, "after"))
  print(sprintf("shares=%6.4f, cash=%6.4f, wealth in stock=%6.4f, total wealth=%6.4f",
                pi.v[i+1], cash[i+1], pi.v[i+1]*s[i+1], wealth[i+1]))
}
## [1] "Day=1, before"
## [1] "shares=1.0000, cash=0.0000, wealth in stock=80.0000, total wealth=80.0000"
## [1] "Day=1, after"
## [1] "shares=1.2500, cash=-20.0000, wealth in stock=100.0000, total wealth=80.0000"
## [1] "Day=2, before"
## [1] "shares=1.2500, cash=-20.0027, wealth in stock=80.0000, total wealth=59.9973"
## [1] "Day=2, after"
## [1] "shares=1.5625, cash=-40.0027, wealth in stock=100.0000, total wealth=59.9973"
## [1] "Day=3, before"
## [1] "shares=1.5625, cash=-40.0082, wealth in stock=125.0000, total wealth=84.9918"
## [1] "Day=3, after"
## [1] "shares=1.2500, cash=-15.0082, wealth in stock=100.0000, total wealth=84.9918"
## [1] "Day=4, before"
## [1] "shares=1.2500, cash=-15.0103, wealth in stock=80.0000, total wealth=64.9897"
## [1] "Day=4, after"
## [1] "shares=1.5625, cash=-35.0103, wealth in stock=100.0000, total wealth=64.9897"
## [1] "Day=5, before"
## [1] "shares=1.5625, cash=-35.0151, wealth in stock=125.0000, total wealth=89.9849"
## [1] "Day=5, after"
## [1] "shares=1.2500, cash=-10.0151, wealth in stock=100.0000, total wealth=89.9849"
## [1] "Day=6, before"
## [1] "shares=1.2500, cash=-10.0164, wealth in stock=125.0000, total wealth=114.9836"
## [1] "Day=6, after"
```

```
## [1] "shares=1.0000, cash=14.9836, wealth in stock=100.0000, total wealth=114.9836"
## [1] "Day=7, before"
## [1] "shares=1.0000, cash=14.9856, wealth in stock=80.0000, total wealth=94.9856"
## [1] "Day=7, after"
## [1] "shares=1.2500, cash=-5.0144, wealth in stock=100.0000, total wealth=94.9856"
## [1] "Day=8, before"
## [1] "shares=1.2500, cash=-5.0151, wealth in stock=80.0000, total wealth=74.9849"
## [1] "Day=8, after"
## [1] "shares=1.5625, cash=-25.0151, wealth in stock=100.0000, total wealth=74.9849"
## [1] "Day=9, before"
## [1] "shares=1.5625, cash=-25.0185, wealth in stock=125.0000, total wealth=99.9815"
## [1] "Day=9, after"
## [1] "shares=1.2500, cash=-0.0185, wealth in stock=100.0000, total wealth=99.9815"
## [1] "Day=10, before"
## [1] "shares=1.2500, cash=-0.0185, wealth in stock=125.0000, total wealth=124.9815"
## [1] "Day=10, after"
## [1] "shares=1.0000, cash=24.9815, wealth in stock=100.0000, total wealth=124.9815"
```

Rule 3:

```
#set up vectors
pi.v=numeric(11)
cash=numeric(11)
wealth=numeric(11)

#initialize
wealth[1]=100
pi.v[1]= (wealth[1]/2)/s[1]
cash[1]=wealth[1]-pi.v[1]*s[1]


#dynamics of the rule
rebalance<-function(wealth,s.price){
  return((wealth/2)/s.price)
  }

#we move forward thorugh time with this rule
for(i in 1:10){
  cash[i+1]=cash[i]*exp(r/365)
  wealth[i+1]=cash[i+1]+pi.v[i]*s[i+1]

  #before re-balancing
  print(sprintf("Day=%d, %s", i, "before"))
  print(sprintf("shares=%6.4f, cash=%6.4f, wealth in stock=%6.4f, total wealth=%6.4f",
                pi.v[i], cash[i+1], pi.v[i]*s[i+1], wealth[i+1]))

  #now we re-balance
  pi.v[i+1]=rebalance(wealth[i+1],s[i+1])
  cash[i+1]=wealth[i+1]-pi.v[i+1]*s[i+1]

  #after re-balancing
  print(sprintf("Day=%d, %s", i, "after"))
  print(sprintf("shares=%6.4f, cash=%6.4f, wealth in stock=%6.4f, total wealth=%6.4f",
```

```
                pi.v[i+1], cash[i+1], pi.v[i+1]*s[i+1], wealth[i+1]))
}
## [1] "Day=1, before"
## [1] "shares=0.5000, cash=50.0068, wealth in stock=40.0000, total wealth=90.0068"
## [1] "Day=1, after"
## [1] "shares=0.5625, cash=45.0034, wealth in stock=45.0034, total wealth=90.0068"
## [1] "Day=2, before"
## [1] "shares=0.5625, cash=45.0096, wealth in stock=36.0027, total wealth=81.0123"
## [1] "Day=2, after"
## [1] "shares=0.6329, cash=40.5062, wealth in stock=40.5062, total wealth=81.0123"
## [1] "Day=3, before"
## [1] "shares=0.6329, cash=40.5117, wealth in stock=50.6327, total wealth=91.1444"
## [1] "Day=3, after"
## [1] "shares=0.5697, cash=45.5722, wealth in stock=45.5722, total wealth=91.1444"
## [1] "Day=4, before"
## [1] "shares=0.5697, cash=45.5785, wealth in stock=36.4578, total wealth=82.0362"
## [1] "Day=4, after"
## [1] "shares=0.6409, cash=41.0181, wealth in stock=41.0181, total wealth=82.0362"
## [1] "Day=5, before"
## [1] "shares=0.6409, cash=41.0237, wealth in stock=51.2726, total wealth=92.2964"
## [1] "Day=5, after"
## [1] "shares=0.5769, cash=46.1482, wealth in stock=46.1482, total wealth=92.2964"
## [1] "Day=6, before"
## [1] "shares=0.5769, cash=46.1545, wealth in stock=57.6852, total wealth=103.8397"
## [1] "Day=6, after"
## [1] "shares=0.5192, cash=51.9199, wealth in stock=51.9199, total wealth=103.8397"
## [1] "Day=7, before"
## [1] "shares=0.5192, cash=51.9270, wealth in stock=41.5359, total wealth=93.4629"
## [1] "Day=7, after"
## [1] "shares=0.5841, cash=46.7314, wealth in stock=46.7314, total wealth=93.4629"
## [1] "Day=8, before"
## [1] "shares=0.5841, cash=46.7378, wealth in stock=37.3852, total wealth=84.1230"
## [1] "Day=8, after"
## [1] "shares=0.6572, cash=42.0615, wealth in stock=42.0615, total wealth=84.1230"
## [1] "Day=9, before"
## [1] "shares=0.6572, cash=42.0673, wealth in stock=52.5769, total wealth=94.6441"
## [1] "Day=9, after"
## [1] "shares=0.5915, cash=47.3221, wealth in stock=47.3221, total wealth=94.6441"
## [1] "Day=10, before"
## [1] "shares=0.5915, cash=47.3285, wealth in stock=59.1526, total wealth=106.4811"
## [1] "Day=10, after"
## [1] "shares=0.5324, cash=53.2406, wealth in stock=53.2406, total wealth=106.4811"
```