Spring 2021 Project

Jeff Nguyen

01/04/2021

```r
library(quantmod)
```

```
## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: TTR

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

## Version 0.4-0 included new data defaults. See ?getSymbols.
```

```r
library(forecast)
library(aTSA)
```

```
##
## Attaching package: 'aTSA'

## The following object is masked from 'package:forecast':
##
##     forecast

## The following object is masked from 'package:graphics':
##
##     identify
```

University of Southern California
Marshall School of Business
FBE 543 Forecasting and Risk Analysis
Spring 2021
Directed by Professor Mohammad Safarzadeh

Table of Contents

Abstract

Introduction

We selected the following 5 securities to base our analysis of impact of COVID-19 on a CAPM model of 5 stocks upon.

Ticker	Security	Sector	Industry	Founded	Full Time Employees
MSFT	Microsoft Corporation	Technology	Software-Infrastructure	1975	163,000
GWPH	GW Pharmaceuticals PLC	Healthcare	Drug Manufacturers-General	1998	901
DIS	The Walt Disney Company	Communication Services	Entertainment	1923	223,000
CAT	Caterpillar INC	Industrials	Farm & Heavy Construction Machinery	1925	102,300
AMZN	Amazon.com INC	Consumer Cyclical	Internet Retail	1994	1,125,300

All information and data related to the securities are obtained from Yahoo Finance: MSFT, GWPH, DIS, CAT, and AMZN.

The objective of the study of the study is using the Modern Portfolio Theory to model a portfolio of five securities from different industries using adjusted closing price data from January 01, 2016 to December 31, 2018.

Methodology

1) Select at least five stocks from different industries.

2) Construct a portfolio of the selected stocks and graph the efficient frontier.

a. Find the optimum weights using MPT.

b. Using the optimum weights and monthly adjusted closing prices at the end of 2018 allocate $100.00 among the selected stocks. On 1/1/2019, the portfolio will have a value of 100 as an index.

c. Using the daily adjusted closing prices from 1/2/ 2019 to present calculate the holding values of the portfolio. Assume fixed holdings with no re-balancing taking place over time. Calculate the CAL equation and graph the CAL and the efficient frontier.

3) Do Naive, MA(5), MA(15), ES, Holt, and Holt-Winters forecasting of your portfolio returns and do a three-period-ahead forecasting of the portfolio returns for each forecast. Estimate the accuracy statistics.

4) Start with the regression analysis and forecasting of your portfolio returns. Use the CAPM and three-factor CAPM (Fama-French) models to estimate the coefficients of the models and use them for forecasting. Do a 10-days ex-post forecasting of the portfolio risk premiums and compare the forecasted value to actual ones. Do a three-period-ahead (ex-ante) forecasting of the portfolio risk premiums and write confidence intervals.
5) Do an ARIMA model of your portfolio returns and use it for three-period ahead forecasting of the returns to portfolio. Write confidence interval. Estimate the accuracy statistics.
6) Test your ARIMA model for the stability of the ARIMA coefficients.
7) Test your ARIMA model for the existence of ARCH and GARCH and do proper corrections, if needed.
8) Find different time-series measures of volatility for your portfolio returns (see the volatility file posted on Blackboard) and do a three-period ahead forecasting of the portfolio volatility. Compare the different measures of volatility with GARCH.
9) Use the accuracy statistics of the different forecasting techniques to decide which technique fits the data best.
10) Test whether your portfolio index conforms to the efficient market hypothesis.
11) Find 1% and 3% daily and monthly VaR of your portfolio.
12) Find 1% and 3% daily and monthly equity EVaR of your portfolio.
13) Graph the security Market Line (SML) of your portfolio and test whether you would add a stock of your own choice to the portfolio or not.
14) Do an intervention function analysis of the March 15th closing of US economy due to COVID19. Did the event have any effect on return to your portfolio.
15) Do a 2-variable VAR between your portfolio index and S&P500 index. Graph the Impulse response function of the VAR and comment on the relationship.

Data Analysis

1) Select at least five stocks from different industries.

```
# Set start date and end date of data
start_date <- "2016-01-01"
end_date <- "2018-12-31"

# Get data
getSymbols("MSFT", src = "yahoo", from = start_date, to = end_date)
```

```
## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be checked for
## alternate defaults.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.
```

```
## [1] "MSFT"
```

```r
getSymbols("GWPH", src = "yahoo", , from = start_date, to = end_date)
```

```
## [1] "GWPH"
```

```r
getSymbols("DIS", src = "yahoo", , from = start_date, to = end_date)
```

```
## [1] "DIS"
```

```r
getSymbols("CAT", src = "yahoo", , from = start_date, to = end_date)
```

```
## [1] "CAT"
```

```r
getSymbols("AMZN", src = "yahoo", , from = start_date, to = end_date)
```

```
## [1] "AMZN"
```

```r
getSymbols("^GSPC", src = "yahoo", , from = start_date, to = end_date)
```

```
## [1] "^GSPC"
```

```r
getSymbols("^TNX", src = "yahoo", from = start_date, to = end_date)
```

```
## Warning: ^TNX contains missing values. Some functions will not work if objects
## contain missing values in the middle of the series. Consider using na.omit(),
## na.approx(), na.fill(), etc to remove or replace them.
```

```
## [1] "^TNX"
```

```r
# Adjusted Prices
adjMSFT <- MSFT$MSFT.Adjusted
adjGWPH <- GWPH$GWPH.Adjusted
adjDIS <- DIS$DIS.Adjusted
adjCAT <- CAT$CAT.Adjusted
adjAMZN <- AMZN$AMZN.Adjusted

# Get adjusted returns data
rMSFT <- diff(log(to.monthly(MSFT)$MSFT.Adjusted))
rGWPH <- diff(log(to.monthly(GWPH)$GWPH.Adjusted))
rDIS <- diff(log(to.monthly(DIS)$DIS.Adjusted))
rCAT <- diff(log(to.monthly(CAT)$CAT.Adjusted))
rAMZN <- diff(log(to.monthly(AMZN)$AMZN.Adjusted))
rGSPC <- diff(log(to.monthly(GSPC)$GSPC.Adjusted))
rTNX <- (to.monthly(TNX)$TNX.Adjusted) / 1200 # Using monthly rate
```

```
## Warning in to.period(x, "months", indexAt = indexAt, name = name, ...): missing
## values removed from data
```

```r
# Calculate statistics
MSFT_return_mean <- mean(rMSFT, na.rm = TRUE)
GWPH_return_mean <- mean(rGWPH, na.rm = TRUE)
DIS_return_mean <- mean(rDIS, na.rm = TRUE)
CAT_return_mean <- mean(rCAT, na.rm = TRUE)
AMZN_return_mean <- mean(rAMZN, na.rm = TRUE)
GSPC_return_mean <- mean(rGSPC, na.rm = TRUE)
TNX_return_mean <- mean(rTNX, na.rm = TRUE)

MSFT_return_var <- var(rMSFT, na.rm = TRUE)
GWPH_return_var <- var(rGWPH, na.rm = TRUE)
DIS_return_var <- var(rDIS, na.rm = TRUE)
CAT_return_var <- var(rCAT, na.rm = TRUE)
AMZN_return_var <- var(rAMZN, na.rm = TRUE)
GSPC_return_var <- var(rGSPC, na.rm = TRUE)

# Excess Returns
reMSFT <- rMSFT - rTNX
reGWPH <- rGWPH - rTNX
reDIS <- rDIS - rTNX
reCAT <- rCAT - rTNX
reAMZN <- rAMZN - rTNX

# Information Tables:
pricTabl <- data.frame(MSFT, GWPH, DIS, CAT, AMZN)

# Creates data frame of asset prices
retTabl <- data.frame(rMSFT, rGWPH, rDIS, rCAT, rAMZN)

# Creates data frame of returns
EretTabl <- data.frame(reMSFT, reGWPH, reDIS, reCAT, reAMZN)

# Excess return data frame
retTabl <- retTabl[-1,]  # remove missing data due to lagging
EretTabl <- EretTabl[-1,]  # remove missing data due to lagging
priceMat <- matrix(c(MSFT, GWPH, DIS, CAT, AMZN), nrow= length(MSFT), ncol=5, byrow=TRUE)  # creates a 

# Variance/Covariance Matrix
asset.names <- c("MSFT", "GWPH", "DIS", "CAT", "AMZN")

# Create a list of row and col names for the var/cov matrix
VCV <- matrix(c(cov(retTabl)), nrow=5, ncol = 5, byrow=TRUE) # create a var/cov matrix by finding cov o
dimnames(VCV) <- list(asset.names, asset.names)  # assigns asset.names to the VCV matrix

#Calculate Returns
rm <- matrix(colMeans(retTabl, na.rm=TRUE))  # creates an average return matrix, omitting missing value
erm <- matrix(colMeans(EretTabl, na.rm=TRUE))  # creates an average excess return matrix, omitting miss
tnxy = mean((rTNX)[-1,])  # calculates the average bond yield excluding Jan (risk free rate)

#Create Return Table
retmat <- matrix(c(rm, erm), ncol=2)
dimnames(retmat) = list(asset.names, c("Return ", "Excess Return"))
```

First we want to look at the data statistics

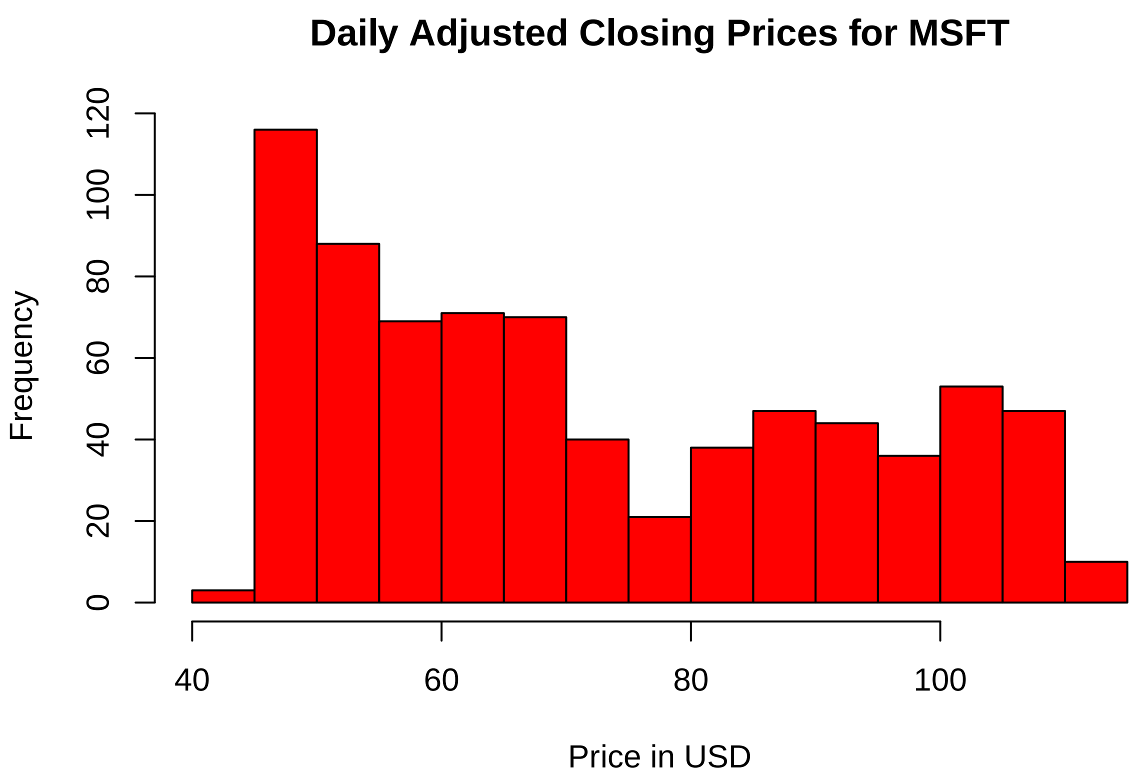
Instruments	Mean Returns	Variance of Returns	Beta (5Y Monthly)
MSFT	0.0190403	0.0027112	.87
GWPH	0.0183674	0.0299313	1.96
DIS	0.0045494	0.0017214	1.08
CAT	0.0223445	0.0058996	.98
AMZN	0.0263838	0.0062955	1.3

Parameters of indices:

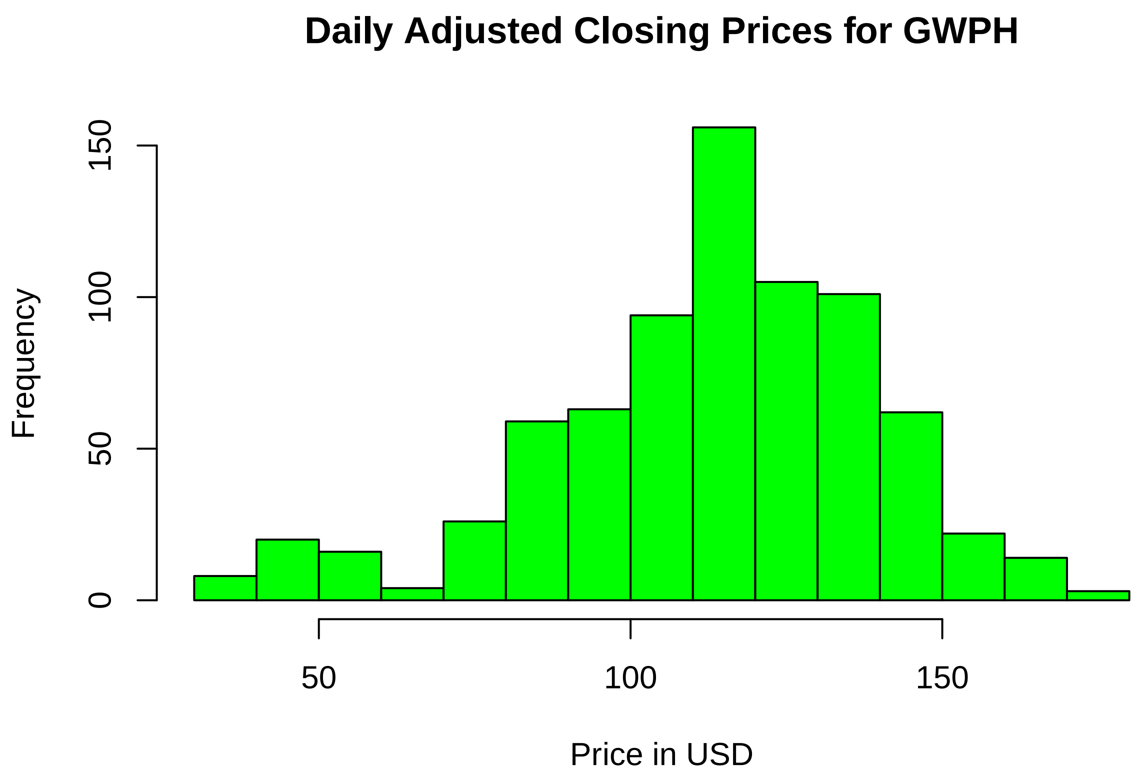
Instruments	Mean Returns	Variance of Returns	Beta
S&P 500	0.0070788	0.0010008	N/A
10-Year T-bill	0.0019565	0	N/A

We look at distribution of adjusted closing prices for each security:

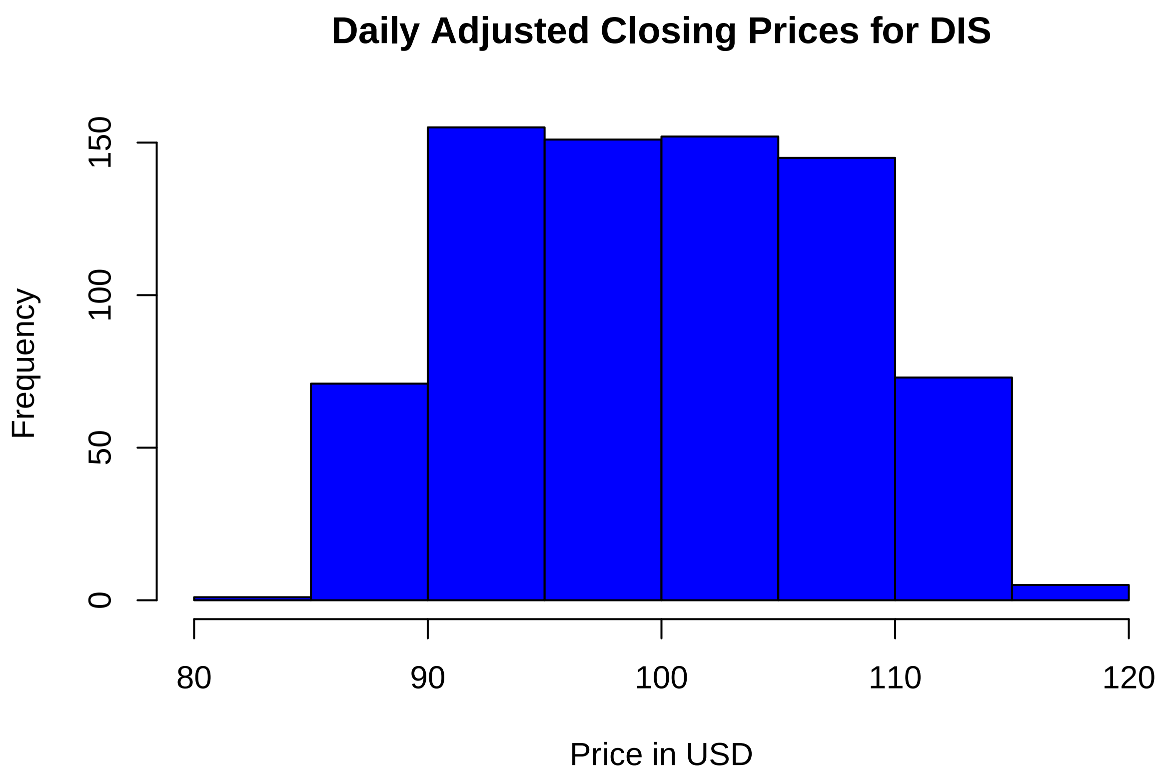
```
hist(adjMSFT,
     main='Daily Adjusted Closing Prices for MSFT',
     xlab='Price in USD',
     col='red',
    )
```



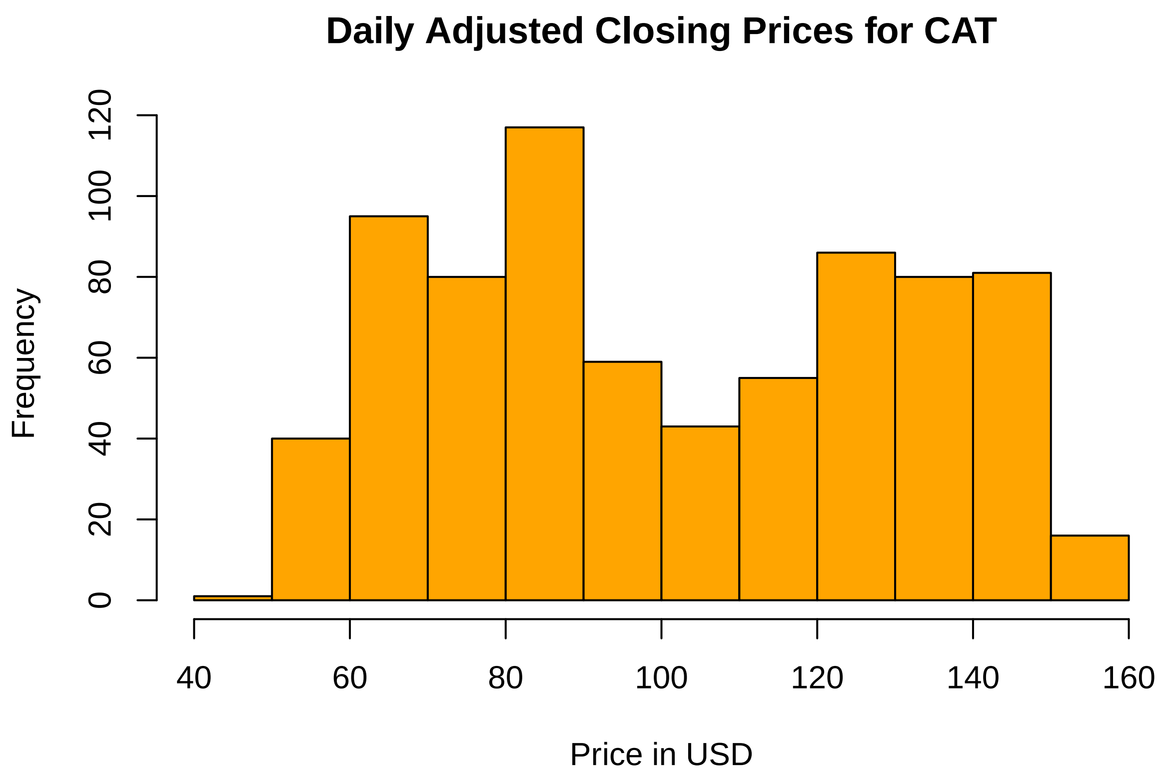

```
hist(adjGWPH,
     main='Daily Adjusted Closing Prices for GWPH',
     xlab='Price in USD',
     col='green',
    )
```



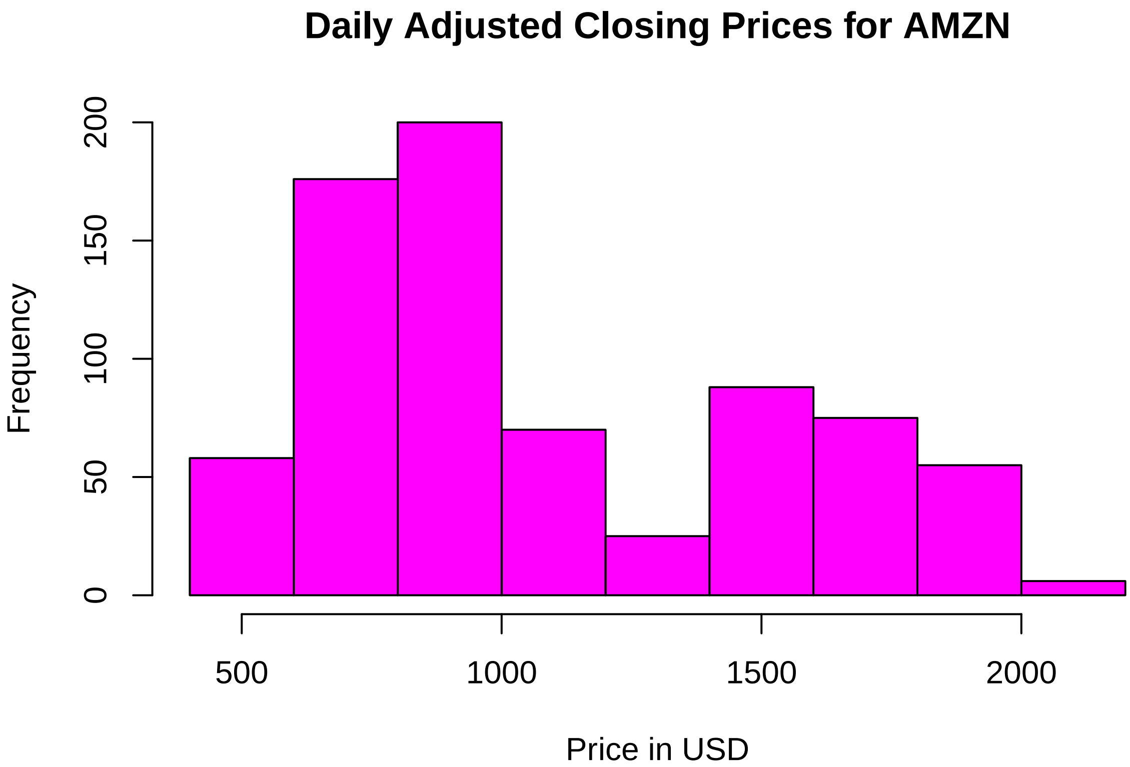

```r
hist(adjDIS,
     main='Daily Adjusted Closing Prices for DIS',
     xlab='Price in USD',
     col='blue',
    )
```




```r
hist(adjCAT,
     main='Daily Adjusted Closing Prices for CAT',
     xlab='Price in USD',
     col='orange',
    )
```




```
hist(adjAMZN,
     main='Daily Adjusted Closing Prices for AMZN',
     xlab='Price in USD',
     col='magenta',
    )
```



CAPM Portfolio Construction

2a) Find the optimum weights using MPT

Since the investor's objective is to minimize risk subjected to a minimum return of the risk free asset–US Treasury Bill, in this case–we solve the constrained optimization problem.
Let x_i denotes the weight of the investment in asset i ($i = 1, 2, 3, 4, 5$), and assume all money is invested in i, meaning $\sum x_i = x_1 + x_2 + x_3 + x_4 + x_5 = 1$.

$$
\begin{aligned}
\text{The returns of the portfolio is:} & \\
R_{p,x} &= x_1 * r_1 + x_2 * r_2 + x_3 * r_3 + x_4 * r_4 + x_5 * r_5 \\
\text{The expected returns on the portfolio is:} & \\
\mu_{p,x} &= E[R_{p,x}] \qquad (1) \\
&= x_1 * \mu_1 + x_2 * \mu_2 + x_3 * \mu_3 + x_4 * \mu_4 + x_5 * \mu_5 \\
\text{The variance of the portfolio returns is:} & \\
\sigma^2_{p,x} &= var(R_{p,x})
\end{aligned}
$$

Formulating the Markowitz portfolio problem:

The investor's objective is:

$$\begin{aligned}
&max \quad \mu_p = w' * \mu \quad \text{s.t.} \\
&\sigma_p^2 = w' * (\textstyle\sum) * w \quad \text{and} \quad w' * I = 1 \\
&\text{where:} \\
&w = \quad \text{matrix of asset weights in the portfolio} \\
&w' = \quad \text{transpose matrix of asset weights in the portfolio} \\
&\mu = \quad \text{matrix of mean returns of asset in the portfolio} \\
&\sum = \quad \text{Variance-covariance matrix of asset returns in the the portfolio} \\
&w' * I = \sum_{i=1}^{n} w_n \quad \text{or the sum weights of the asset in the portfolio, I is notation for identity matrix}
\end{aligned} \tag{2}$$

Let $\mu_{p,0}$ denotes a target expected return level. Formulate the problem:

$$\begin{aligned}
min \quad \sigma_{p,w}^2 = w' * (\textstyle\sum) * w \quad \text{s.t.} \\
\mu_p = w' * \mu = \mu_{p,0}, \quad \text{and} \quad w' * I = 1
\end{aligned} \tag{3}$$

To solve this, form the Lagrangian function:

$$L(w, \lambda_1, \lambda_2) = w' * \sum * w + \lambda_1 * (w' * \mu - \mu_{p,0}) + \lambda_2 * (w' * I - 1) \tag{4}$$

Because there are two constraints ($w' * \mu = \mu_{p,0}$ and $w'1 = 1$) there are two Langrange multipliers λ_1 and λ_2. The first order condition for a minimum are the linear equations:

$$\begin{aligned}
&\frac{\partial L(w, \lambda_1, \lambda_2)}{\partial w} = \frac{\partial(\sum * w^2)}{\partial w} + \frac{\partial(\lambda_1 * (w' * \mu - \mu_{p,0}))}{\partial w} + \frac{\lambda_2 * (w' * I - 1)}{\partial w} = 0 \\
&\frac{\partial L(w, \lambda_1, \lambda_2)}{\partial \lambda_1} = 0 \\
&\frac{\partial L(w, \lambda_1, \lambda_2)}{\partial \lambda_2} = 0
\end{aligned} \tag{5}$$

Simplify, we have:

$$\begin{aligned}
&\frac{\partial L(w, \lambda_1, \lambda_2)}{\partial w} = 2 * \sum * w + \lambda_1 * \mu + \lambda_2 * I = 0 \\
&\frac{\partial L(w, \lambda_1, \lambda_2)}{\partial \lambda_1} = w' * \mu - \mu_{p,0} = 0 \\
&\frac{\partial L(w, \lambda_1, \lambda_2)}{\partial \lambda_2} = w' * I - 1 = 0
\end{aligned} \tag{6}$$

Rewrite in matrix form:

$$\begin{pmatrix} 2*\sum & \mu & I \\ \mu' & 0 & 0 \\ I' & 0 & 0 \end{pmatrix} * \begin{pmatrix} w \\ \lambda_1 \\ \lambda_2 \end{pmatrix} = \begin{pmatrix} 0 \\ \mu_{p,0} \\ I \end{pmatrix} \tag{7}$$

or

$$\begin{aligned} & A * z_w = b_0 \\ & \text{where} \\ & A = \begin{pmatrix} 2*\sum & \mu & I \\ \mu' & 0 & 0 \\ I' & 0 & 0 \end{pmatrix} \\ & z_w = \begin{pmatrix} w \\ \lambda_1 \\ \lambda_2 \end{pmatrix} \\ & b_0 = \begin{pmatrix} 0 \\ \mu_{p,0} \\ I \end{pmatrix} \end{aligned} \tag{8}$$

The solution for z_w is:

$$z_w = A^{-1} * b_0 \tag{9}$$

The variance-covariance matrix is as follow:

```
VCV
```

```
##              MSFT        GWPH          DIS          CAT         AMZN
## MSFT 0.0027112031 0.003255723 0.0004124035 0.0014986837 0.0025785122
## GWPH 0.0032557231 0.029931273 0.0020363050 0.0058326612 0.0064760106
## DIS  0.0004124035 0.002036305 0.0017214342 0.0009435589 0.0005948209
## CAT  0.0014986837 0.005832661 0.0009435589 0.0058996262 0.0023212073
## AMZN 0.0025785122 0.006476011 0.0005948209 0.0023212073 0.0062955209
```

The monthly risk-free rate is: 0.0019664

```
# Optimum Portfolio
ZOPT <- solve(VCV,erm)  # multiply inverse of VCV to excess return to find z
WOPT <- ZOPT/sum(ZOPT)  # calculates weights
dimnames(WOPT) <- list(asset.names, "Weights") #label the weight matrix

# Calculate stats
ROPT <- t(WOPT)%*%rm  # calculate optimal portfolio's return
VOPT <- t(WOPT)%*%VCV%*%WOPT  # calculate optimal portfolio's variance
SDOPT <- VOPT^0.5  # calculate optimal portfolio's std dev
SRatio <-(ROPT-tnxy)/(SDOPT)  # calculate optimal portfolio's Sharpe ratio

# Create Optimal Stats Table
```

```r
PTBL <- matrix(c(ROPT, VOPT, SDOPT, SRatio), nrow = 4)  # create a matrix of return, variance, std dev,
optstat.names <- c("Return", "Variance", "Std Dev", "Sharpe")  # labels for PTBL matrix

dimnames(PTBL) <- list(optstat.names, "Opt. Portfolio")  # label the optimal portfolio matrix values
```

The optimal portfolio weights are as follow:

```r
WOPT
```

```
##          Weights
## MSFT  0.53181782
## GWPH -0.11209766
## DIS  -0.08535048
## CAT   0.34599848
## AMZN  0.31963184
```

The statistics of the optimal portfolio is:

```r
PTBL
```

```
##          Opt. Portfolio
## Return      0.023842997
## Variance    0.003055134
## Std Dev     0.055273267
## Sharpe      0.395790177
```

2b) Allocate $100.00 among the selected stocks using adjusted closing prices at 2018M12. 2019M1 will have a value of 100 as an index.

```r
# Set start date and end date of data
start_date1 <- "2018-12-01"
end_date1 <- "2020-08-31"

# Get data
getSymbols("MSFT", src = "yahoo", from = start_date1, to = end_date1)
```

```
## [1] "MSFT"
```

```r
getSymbols("GWPH", src = "yahoo", , from = start_date1, to = end_date1)
```

```
## [1] "GWPH"
```

```r
getSymbols("DIS", src = "yahoo", , from = start_date1, to = end_date1)
```

```
## [1] "DIS"
```

```r
getSymbols("CAT", src = "yahoo", , from = start_date1, to = end_date1)
```

```
## [1] "CAT"
```

```r
getSymbols("AMZN", src = "yahoo", , from = start_date1, to = end_date1)
```

```
## [1] "AMZN"
```

```r
getSymbols("^GSPC", src = "yahoo", , from = start_date1, to = end_date1) # S&P 500
```

```
## [1] "^GSPC"
```

```r
getSymbols("^TNX", src = "yahoo", from=start_date1, to=end_date1) # TNX (10-year T-bill)
```

```
## Warning: ^TNX contains missing values. Some functions will not work if objects
## contain missing values in the middle of the series. Consider using na.omit(),
## na.approx(), na.fill(), etc to remove or replace them.
```

```
## [1] "^TNX"
```

```r
rMSFT1 <- diff(log(to.monthly(MSFT)$MSFT.Adjusted))
rGWPH1 <- diff(log(to.monthly(GWPH)$GWPH.Adjusted))
rDIS1 <- diff(log(to.monthly(DIS)$DIS.Adjusted))
rCAT1 <- diff(log(to.monthly(CAT)$CAT.Adjusted))
rAMZN1 <- diff(log(to.monthly(AMZN)$AMZN.Adjusted))
rGSPC1 <- diff(log(to.monthly(GSPC)$GSPC.Adjusted))
rTNX1 <- to.monthly(TNX)$TNX.Adjusted /1200  # Using monthly rate
```

```
## Warning in to.period(x, "months", indexAt = indexAt, name = name, ...): missing
## values removed from data
```

```r
rTNX1 <- rTNX1[-1,]  # remove missing data due to lagging
mean_rTNX1 <- mean(rTNX1, na.rm=TRUE)

# Adjusted Prices
adjMSFT1 <- MSFT$MSFT.Adjusted
adjGWPH1 <- GWPH$GWPH.Adjusted
adjDIS1 <- DIS$DIS.Adjusted
adjCAT1 <- CAT$CAT.Adjusted
adjAMZN1 <- AMZN$AMZN.Adjusted
adjGSPC <- GSPC$GSPC.Adjusted

investedAmount <- 100

sharesMSFT <- as.numeric(investedAmount * WOPT[1] / adjMSFT1[1])
sharesGWPH <- as.numeric(investedAmount * WOPT[2] / adjGWPH1[1])
sharesDIS <- as.numeric(investedAmount * WOPT[3] / adjDIS1[1])
sharesCAT <- as.numeric(investedAmount * WOPT[4] / adjCAT1[1])
sharesAMZN <- as.numeric(investedAmount * WOPT[5] / adjAMZN1[1])
```

```r
holdings <- data.frame("Holding Value"=sharesMSFT*adjMSFT1 +
                                       sharesGWPH*adjGWPH1 +
                                       sharesDIS*adjDIS1 +
                                       sharesCAT*adjCAT1 +
                                       sharesAMZN*adjAMZN1)
names(holdings)[1] <- "Port. Holdings Val"  # rename column
```

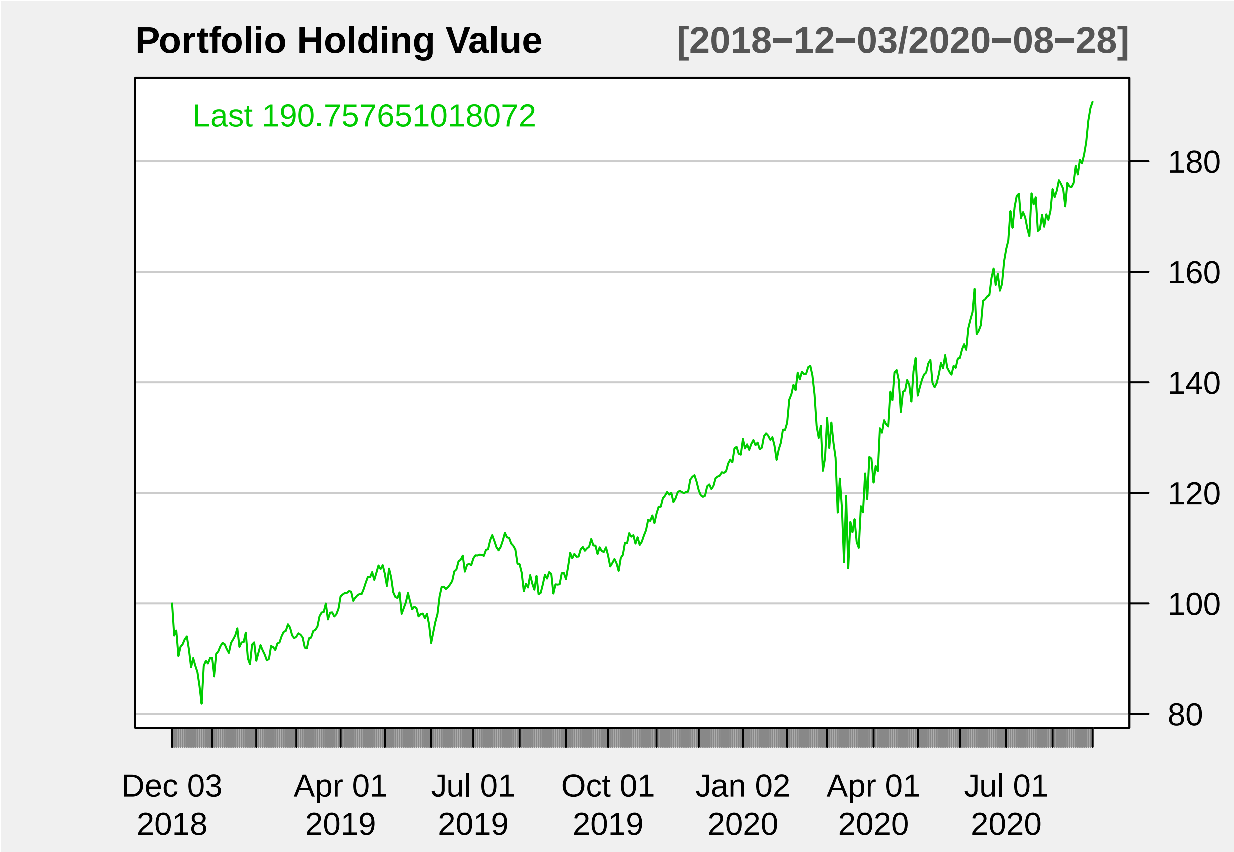
Based on the optimal weighting, to allocate $100 to the portfolio, we would be purchase the following amount of each security:

Ticker	Weights	Stock to purchase
MSFT	0.5318178	0.4876926
GWPH	-0.1120977	-0.0887902
DIS	-0.0853505	-0.0752235
CAT	0.3459985	0.2663836
AMZN	0.3196318	0.0180343

2c) Using the adjusted closing prices from 2018M12 to 2020M8 calculate the holding values of the portfolio (assume fixed holdings with no re-balancing taking place over time).

We can then observe the fluctuations in the holding value of the portfolio from the period starting December 01 2018 to August 31, 2020 as follow.

```r
chartSeries(holdings, name="Portfolio Holding Value", type="line", theme=chartTheme("white"))
```



By inspection we can see the portfolio experience a sharp sell off of almost 20% in December 2018, coincide with the broad U.S.market selloff due to a combination of the FED hiking the federal funds rate by 25 basis points to a targeted range of 2.25% to 2.5% (JeffCoxCNBCcom) and corporations followed suit by cutting profit forecasts and try temper expectations for earnings growth in 2019 after a big 2018 (Moyer).

The second visibly sharp sell off of the portfolio holding value also coincides with the broad market sell off in the mid March 2020 with investors raising cash in a risk-on environment when COVID-19 lockdowns started going into effects in the U.S.

Find the tangency point of the Capital Allocation Line (CAL) and the efficient frontier.

The tangency point of the Capital Allocation Line is the point where the weights of the portfolio is optimal, represented by the point (σ_p, r_p) which is $(0.0552733, 0.023843)$.

Calculate the CAL equation and graph CAL and the efficient frontier.

The efficient frontier is the portfolio possibility curve represented by the equation: $CAL = 0.0019664 + 0.3957902 * \sigma_p$

```r
# Efficient Frontier and CAL
j <- 0  # set value for iterative loop variable t
return_p <- rep(0, 50000)
sd_p <- rep(0, 50000)
```

```r
# create a matrix of 0 to fill later with sd of different weights
vect_0 <- rep(0, 50000)

# create a matrix of 0
fractions <- matrix(vect_0, 10000, 5)

# create a matrix of 0 to fill with weights
# iterate through weights for asset 1-5 from -20% to 100% by 10%
for (a in seq(-.2, 1, 0.1))
  {
  for (b in seq(-.2, 1, 0.1))
    {
    for (c in seq(-.2, 1, 0.1))
      {
      for (d in seq(-.2, 1, 0.1))
        {
        for (e in seq(-.2, 1, 0.1))
          {
          #test that the weights are equal to 1
          if (a+b+c+d+e==1)
            {
            # increment j by 1 if a+b+c+d+e is equal to 1 (valid weights)
            j=j+1
            # load a,b,c,d,e values into row j of the matrix
            fractions[j,] <- c(a,b,c,d,e)
            # calculate the std dev of the portfolio at a given weight of assets
            sd_p[j] <- (t(fractions[j,])%*%VCV%*%fractions[j,])^.5
            # calculate the return of the portfolio at a given weight of assets
            return_p[j] <- fractions[j,]%*%rm
            }
          }
        }
      }
    }
  }
# assign filled vector spots in return_p to the R_p matrix to omit empty spots
Rport <- return_p[1:j]

# assign filled vector spots in sd_p to the sigma_p matrix to omit empty spots
StdDev_p <- sd_p[1:j]

# Create Capital Asset Line
# Create x-coordinates for CAL points
f <- seq(0,.24, .24)

# Calculate corresponding y-coordinates
CAL <- tnxy + SRatio * f
```

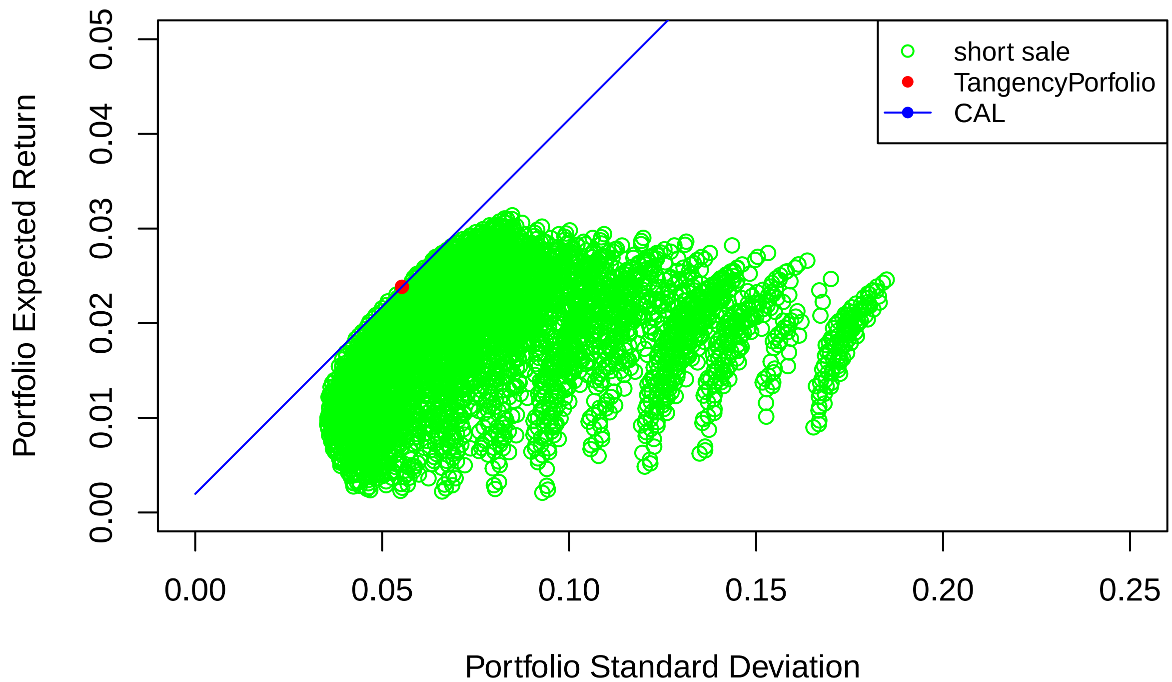
```
## Warning in SRatio * f: Recycling array of length 1 in array-vector arithmetic is deprecated.
##   Use c() or as.vector() instead.
```

```r
#Plot the portfolio possibilities curve:
plot(StdDev_p, Rport, col="green1", xlab="Portfolio Standard Deviation", ylab= "Portfolio Expected Retu
```

```r
#Plot of tangency point in red
points(SDOPT, ROPT, col= "red", pch=16, bg="red")

#Plot of CAL in blue
points(f, CAL, col= "blue", type="l")

legend("topright",c("short sale", "TangencyPorfolio", "CAL"), cex=.8, col=c("green1", "red","blue"),
       lty =c(0,0,1),pch=c(1,16,16))
```



Estimate CAPM for your portfolio and graph the estimated β of the CAPM and the average return of your portfolio as a point relative to SML.

The expected risk premium of the portfolio based on the CAPM model is given as:

$$\begin{aligned} E(R_a - R_f) &= \beta * (R_m - R_f) \\ &\text{or} \\ R_a &= R_f + \beta * (R_m - R_f) \\ R_a - R_f &= \alpha_{Jensen} + \beta * (R_m - R_f) \\ &\text{or} \\ Y &= \alpha_{Jensen} + \beta * X + epsilon \\ &\text{with} \\ Y &= R_a - R_f \\ X &= R_m - R_f \\ \beta &= \text{Market risk or systematic risk} \\ \epsilon &= \text{stochastic error term} \end{aligned} \tag{10}$$

Here, the risk premium of the S&P 500 is the independent variable and the expected risk premium of the portfolio is the dependent variable.

Hypothesis for regression:

$$
\begin{aligned}
H_0 &: \alpha = 0 \\
H_a &: \alpha \neq 0 \\
&\text{and} \\
H_0 &: \beta = 0 \\
H_a &: \beta \neq 0
\end{aligned}
\tag{11}
$$

```
# Calculate and normalized the CAPM holdings
ra <- diff(log(to.monthly(holdings)[,1]))

Y <- na.omit(ra - rTNX1)
names(Y)[1] <- "Portfolio Risk Premium"  # Rename column
Y_bar <- mean(Y)
Y_bar
```

```
## [1] 0.02665436
```

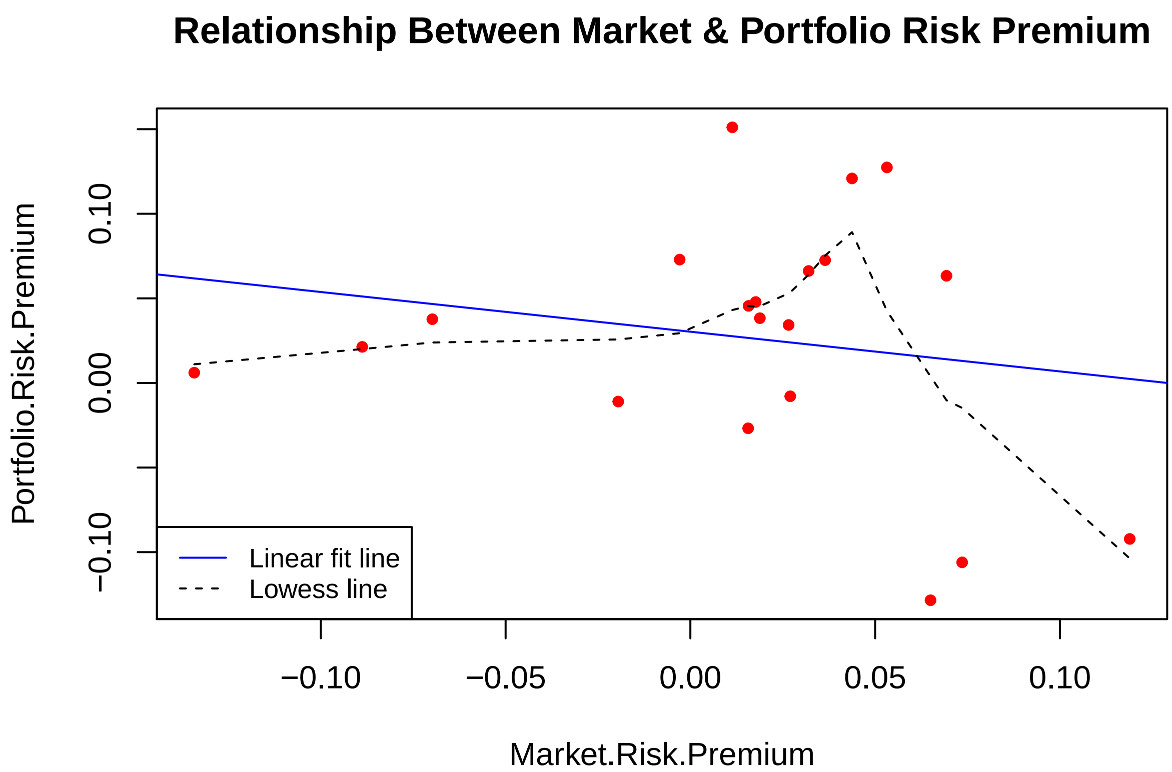
```
X <- na.omit(rGSPC1 - rTNX1)
mean_X <- mean(X)

names(X)[1] <- "Market Risk Premium"  # Rename column
data1 <- data.frame(X, Y)

plot(data1, col='red', main="Relationship Between Market & Portfolio Risk Premium", pch=20, cex=1)

# Add fit lines
abline(lm(Y~X), col="blue")  # Regression line Y ~ X
lines(lowess(X,Y), col="black", lty=2)  # Lowess line (X,Y)

legend("bottomleft",c("Linear fit line", "Lowess line"), cex=.8, col=c("blue", "black"), lty=1:2)
```



Through inspection, we observe the cluster observation scattering in a big range from left to right. This implies a weak linear relationship between the Market Portfolio Risk Premium (the independent X variable on the x-axis) and the CAPM Portfolio Risk Premium (the dependent Y variable on the y-axis).

Next, we attempts to fit an equation of a line: $Y = \alpha_{Jensen} + \beta * X + \epsilon$

```
fit1 <- lm(Y~X, data=data1)
summary(fit1)
```

```
##
## Call:
## lm(formula = Y ~ X, data = data1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.14347 -0.04775  0.01132  0.04489  0.12346
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.03029    0.01735   1.746   0.0979 .
## X           -0.23470    0.29421  -0.798   0.4354
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07488 on 18 degrees of freedom
## Multiple R-squared:  0.03415,    Adjusted R-squared:  -0.01951
## F-statistic: 0.6364 on 1 and 18 DF,  p-value: 0.4354
```

The estimated equation is $Y = .04050 - .34504 * X$, where the p_{value} for the intercept $.0308 < .05$. Therefore, we reject the null hypothesis at 95% confidence level that the intercept α_{Jensen} statistically is

no different from zero. Thus, we reject the null hypothesis $H_0 : \alpha = 0$ and accept the null hypothesis $H_a : \alpha \neq 0$.
The coefficient $\beta = 1.07468$ represents the increase in portfolio risk premium relative to increase in the market portfolio risk premium. The p_{value} for β is $.2544 > .05$, implying that the coefficient β statistically is insignificant at 95% or more, and we accept the null hypothesis $H_0 : \beta = 0$ and reject the alternative hypothesis $H_a : \beta \neq 0$.

Goodness of Fit:
Through inspection, we observe the $R^2 = .07151$ value to not be close to 1 at all. $R^2 = .07151$ implies that 7.15 of the variations in the portfolio risk premium is explained by the market risk premium.

Standard Error of Regression:
We can see that the Standard Error of Regression is $S.E. = .07458$.
From this, we can calculate the forecasting efficiency statistic to be:

$$\begin{aligned} \frac{S.E.}{\overline{Y}} &= \frac{.05618}{0.0266544} \\ &= 279.8\% > 10\% \end{aligned} \tag{12}$$

This statistic implies that this is not a good forecasting model.

Thus, upon exploring the goodness of fit and standard error of regression, we confirm our initial observation that the portfolio risk premium and the market portfolio risk premium has a weak linear relationship.

The Security Market Line:

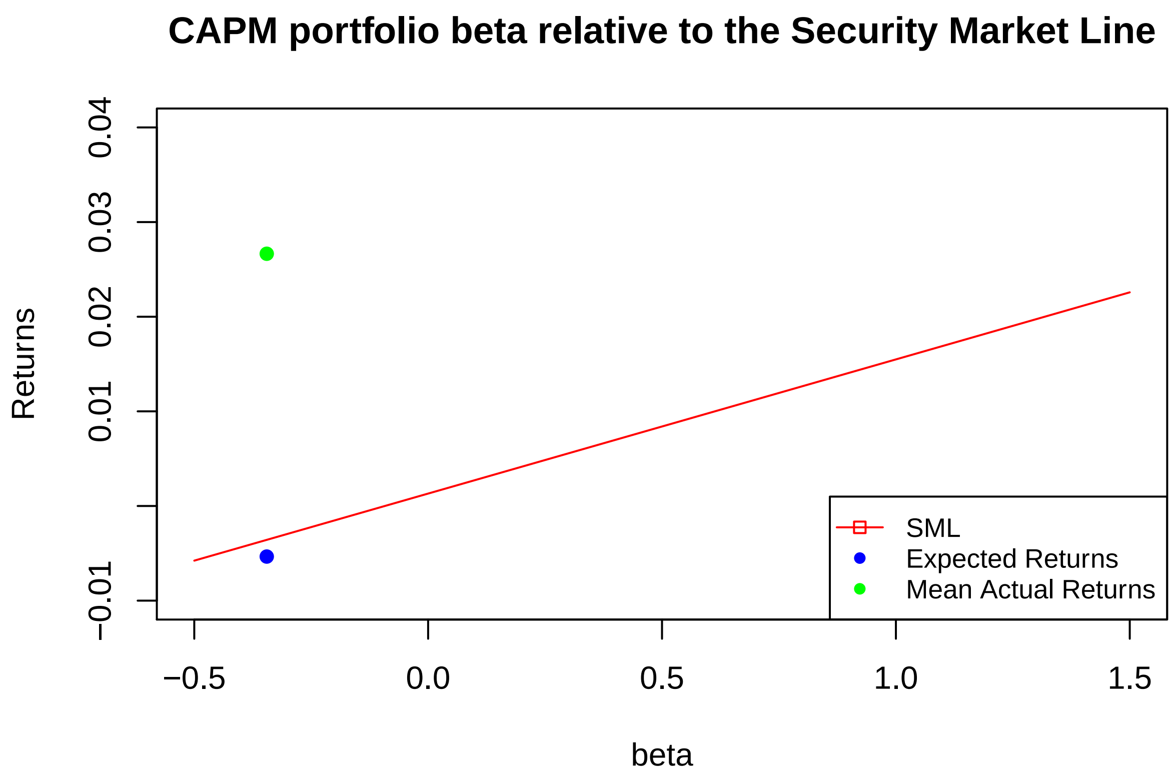
```
# Generate the SML equation
slope_SML <- (mean_X - mean_rTNX1) / (1-0)
#slope_SML
SML <- function(beta) mean_rTNX1 + slope_SML * beta

# Plot the SML
beta <- seq(-.5, 1.5)
plot(beta, SML(beta), col="red", type="l", main="CAPM portfolio beta relative to the Security Market Lin

# Plot the expected returns
points(-.34504, -.34504*mean_X, col="blue", pch=16)

# Plot the average returns
points(-.34504, Y_bar, col="green", pch=16)

legend("bottomright",c("SML", "Expected Returns", "Mean Actual Returns"), cex=.8,
       col=c("red", "blue", "green"), lty=c(1,0,0), pch=c(0,16,16))
```



The Security Market Line pass through the point $(0, \overline{R_f})$ and $(1, \overline{X})$, which are $(0, 0.0013135)$ and $(1, 0.0154876)$.
Relative to its market risk of $beta = -.34504$, the expected return is -0.0053439 and the average return is 0.0266544. We can observe that at this estimated β, the expected return is below the security market line and the actual average return is above the security market line.

Forecasting of Portfolio

3) Do Naive, MA(5), MA(15), ES, Holt, and Holt-Winters forecasting of your portfolio returns and do a three-period-ahead forecasting of the portfolio returns for each forecast. Estimate the accuracy statistics.

Get the portfolio monthly returns over the period based on its daily closing price:

```
monthIndex <- c("Jan 2019", "Feb 2019", "Mar 2019",
                "Apr 2019", "May 2019", "Jun 2019",
                "Jul 2019", "Aug 2019", "Sep 2019",
                "Oct 2019", "Nov 2019", "Dec 2019",
                "Jan 2020", "Feb 2020", "Mar 2020",
                "Apr 2020", "May 2020", "Jun 2020",
                "Jul 2020", "Aug 2020")
rHoldings <- ts(ra)
```

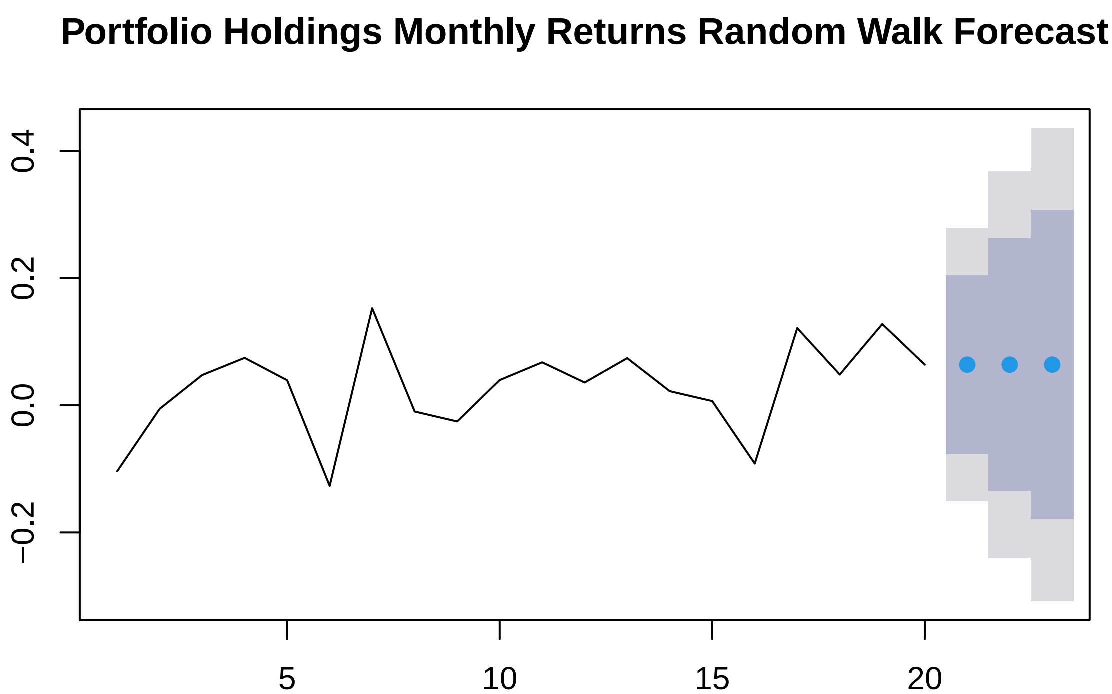
Naive Foreacsting

We look at the 3-period ahead forecasted value.

```r
rwf <- rwf(rHoldings, 3)
rwf
```

```
##    Point Forecast       Lo 80     Hi 80      Lo 95     Hi 95
## 21     0.06395246 -0.07647209 0.2043770 -0.1508084 0.2787133
## 22     0.06395246 -0.13463784 0.2625428 -0.2397652 0.3676701
## 23     0.06395246 -0.17927000 0.3071749 -0.3080242 0.4359291
```

```r
plot(rwf, main="Portfolio Holdings Monthly Returns Random Walk Forecast")
```



We examine the accuracy statistics:

```r
accuracy(rwf)
```

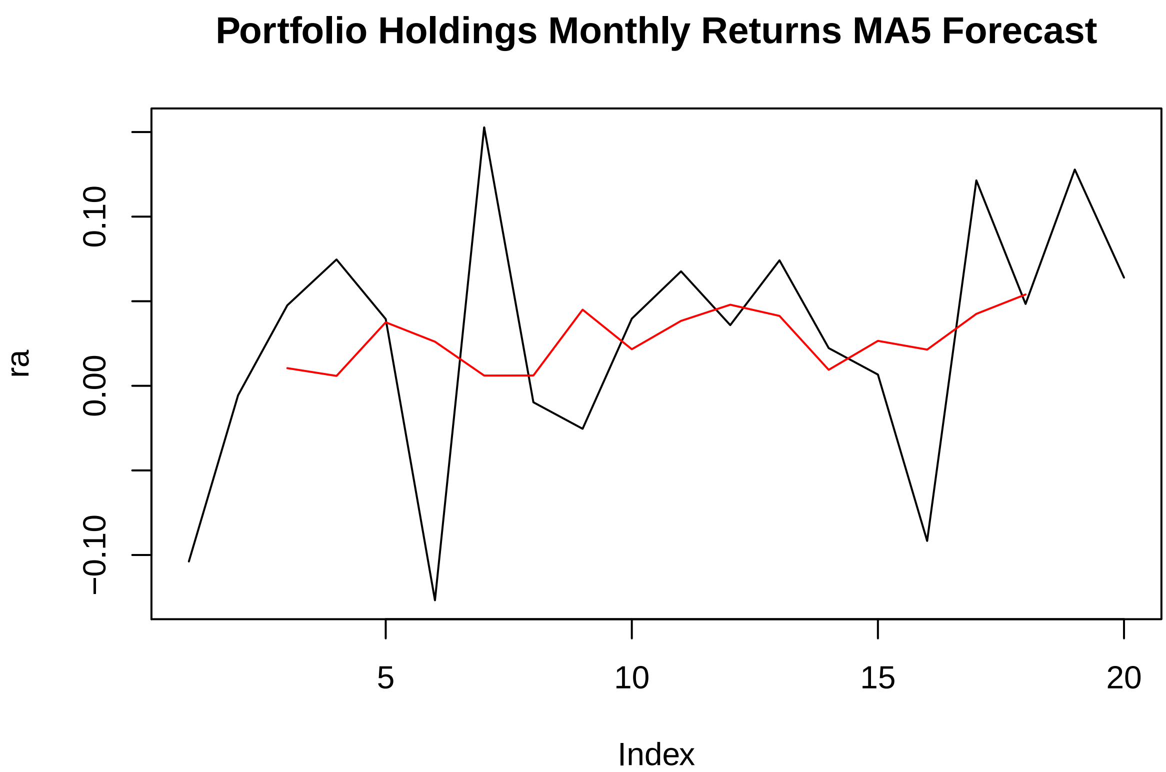
```
##                       ME      RMSE        MAE      MPE     MAPE MASE       ACF1
## Training set 0.008831144 0.1095739 0.08400768 7.773319 286.3907    1 -0.5620567
```

MA(5) Forecast

We look at the 3-period ahead forecasted value.

```r
ma5 <- ma(rHoldings, order=5)

plot(ra, main="Portfolio Holdings Monthly Returns MA5 Forecast", type = "l")
lines(ma5, col="red")
```

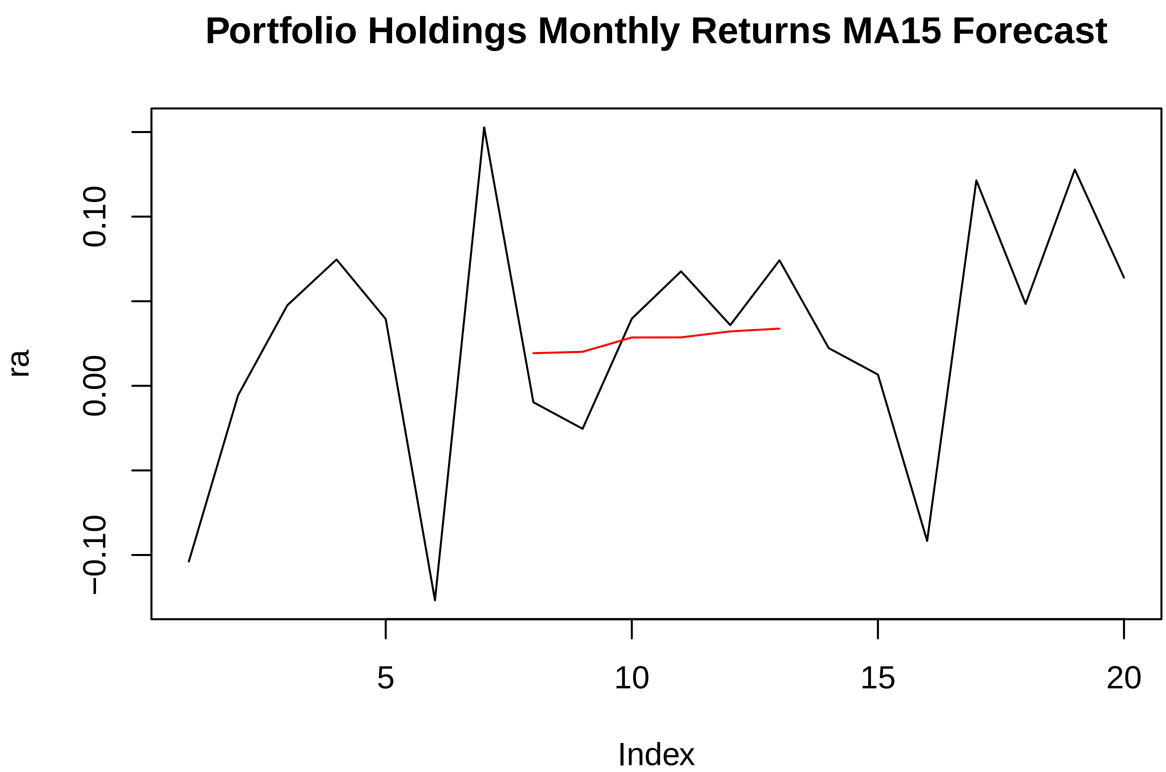


MA(15) Forecast

We look at the 3-period ahead forecasted value.

```
ma15 <- ma(rHoldings, order=15)

plot(ra, main="Portfolio Holdings Monthly Returns MA15 Forecast", type = "l")
lines(ma15, col="red")
```

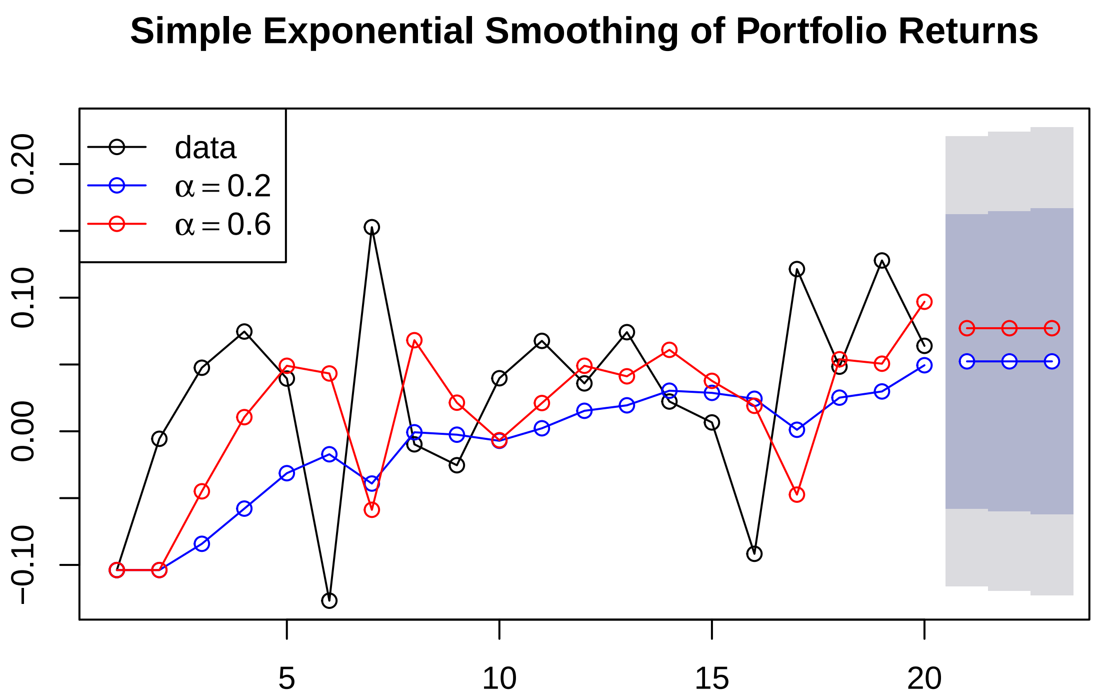


Exponential Smoothing Forecast

We look at the 3-period ahead forecasted value.

```
fit1 <- ses(rHoldings, alpha=.2, initial="simple", h=3)
fit2 <- ses(rHoldings, alpha=.6, initial="simple", h=3)
#fit3 <- ses(rHoldings, h=3)
plot(fit1, main="Simple Exponential Smoothing of Portfolio Returns", fcol="white", type="o")
lines(fitted(fit1), col="blue", type="o")
lines(fitted(fit2), col="red", type="o")
#lines(fitted(fit3), col="green", type="o")
lines(fit1$mean, col="blue", type="o")
lines(fit2$mean, col="red", type="o")

#lines(fit3$mean, col="green", type="o")
legend("topleft",lty=1, col=c(1,"blue","red"),
       c("data", expression(alpha == 0.2), expression(alpha == 0.6)),pch=1)
```



With $\alpha = .2$:

```
fit1
```

```
##    Point Forecast       Lo 80     Hi 80      Lo 95     Hi 95
## 21     0.05233808 -0.05780524 0.1624814 -0.1161116 0.2207878
## 22     0.05233808 -0.05998651 0.1646627 -0.1194476 0.2241237
## 23     0.05233808 -0.06212622 0.1668024 -0.1227200 0.2273961
```

```
accuracy(fit1)
```

```
##                      ME       RMSE        MAE       MPE     MAPE     MASE
## Training set 0.03904434 0.08594529 0.06781889 -18.61967 193.3184 0.807294
##                    ACF1
## Training set -0.193465
```

With $\alpha = .6$:

```
fit2
```

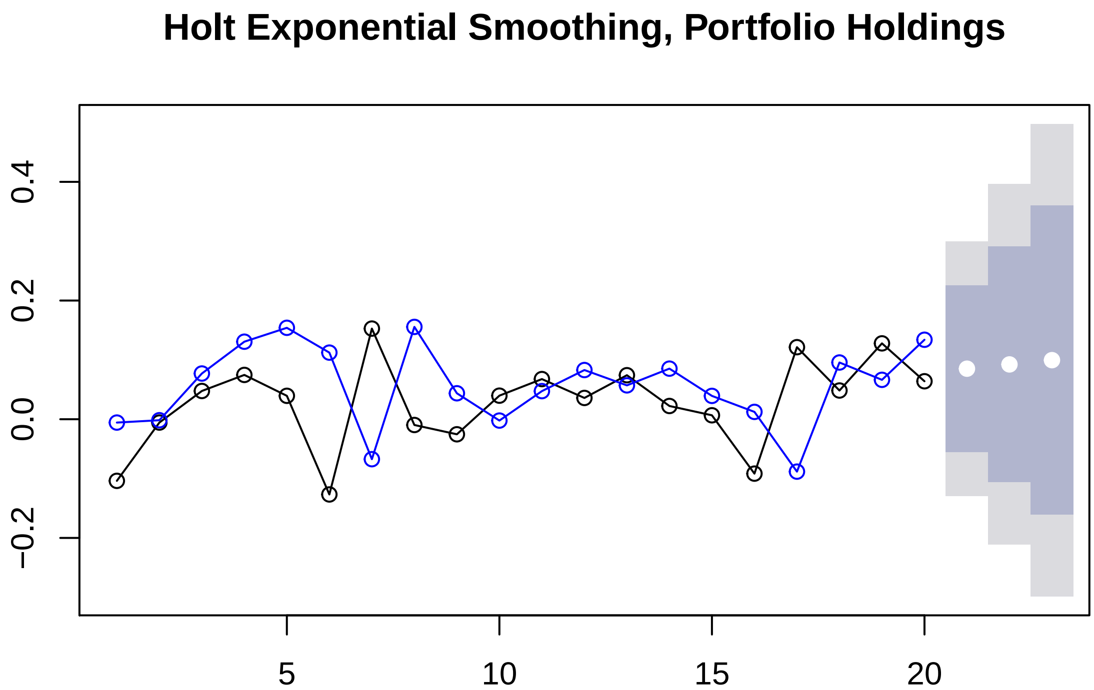
```
##    Point Forecast       Lo 80     Hi 80       Lo 95     Hi 95
## 21     0.07714779 -0.03755176 0.1918473 -0.09827003 0.2525656
## 22     0.07714779 -0.05661372 0.2109093 -0.12742279 0.2817184
## 23     0.07714779 -0.07327926 0.2275748 -0.15291053 0.3072061
```

```
accuracy(fit2)
```

```
##                      ME       RMSE        MAE       MPE     MAPE      MASE
## Training set 0.01508226 0.08950053 0.06874322 -21.41871 230.1674 0.8182968
##                     ACF1
## Training set -0.3748286
```

Holt Trend Model Forecast

```r
#Holt Trend Model
holt1 <- holt(ra, alpha=0.8, beta=0.2, initial="simple", h=3)
plot(holt1, main="Holt Exponential Smoothing, Portfolio Holdings", fcol="white", type="o")
lines(fitted(holt1), col="blue", type="o")
```



Retrieving forecast data from Holt ES:

```r
holt1$model$state
```

```
## Time Series:
## Start = 0
## End = 20
## Frequency = 1
##              l           b
##  0 -0.103839285  0.098225794
##  1 -0.084194126  0.082509667
##  2 -0.004827684  0.081881022
##  3  0.053470471  0.077164449
##  4  0.085854361  0.068208337
##  5  0.062386732  0.049873144
##  6 -0.078959875  0.011629194
##  7  0.108745781  0.046844486
##  8  0.023328100  0.020392053
##  9 -0.011577793  0.009332464
## 10  0.031312518  0.016044033
## 11  0.063601793  0.019293082
## 12  0.045271952  0.011768497
## 13  0.070751169  0.014510641
```

```
## 14  0.034876279  0.004433535
## 15  0.013175440 -0.000793340
## 16 -0.070872875 -0.017444335
## 17  0.079475437  0.016114194
## 18  0.057827287  0.008561725
## 19  0.115549073  0.018393738
## 20  0.077950530  0.007195282
```

```r
holt1$mean
```

```
## Time Series:
## Start = 21
## End = 23
## Frequency = 1
## [1] 0.08514581 0.09234109 0.09953637
```

```r
accuracy(holt1)
```

```
##                        ME      RMSE        MAE      MPE     MAPE     MASE
## Training set -0.02844704 0.1095262 0.08551491 71.01616 225.0103 1.017942
##                    ACF1
## Training set -0.4120045
```

Holt-Winter Seasonal Method:

There might not be seasonality in our data to run the Holt-Winter model. The numerical method is shown below.

```r
#hws2 <- hw(rHoldings)
#plot(hws2, plot.conf=FALSE, type="o", fcol="white")
#lines(fitted(hws1), col="red", lty=2)
```

4) Start with the regression analysis and forecasting of your portfolio returns. Use the CAPM model to estimate the coefficients of the models and use them for forecasting. Do a 10-days ex-post forecasting of the portfolio risk premiums and compare the forecasted value to actual ones. Do a three-period-ahead (ex-ante) forecasting of the portfolio risk premiums and write confidence intervals.

Regression of Market Risk Premium and Portfolio Risk Premium

First we run the linear regression and examine its result.

```r
# Recall Y is the portfolio risk premium from January 2019 to August 2020
# as calculated above.
# Likewise, rTNX1 is the risk free rate for the same period.
reg <- lm(Portfolio.Risk.Premium ~ Market.Risk.Premium, data=data1)
summary(reg)
```

```
##
## Call:
## lm(formula = Portfolio.Risk.Premium ~ Market.Risk.Premium, data = data1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.14347 -0.04775  0.01132  0.04489  0.12346
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          0.03029    0.01735   1.746   0.0979 .
## Market.Risk.Premium -0.23470    0.29421  -0.798   0.4354
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07488 on 18 degrees of freedom
## Multiple R-squared:  0.03415,    Adjusted R-squared:  -0.01951
## F-statistic: 0.6364 on 1 and 18 DF,  p-value: 0.4354
```

```
confint(reg)
```

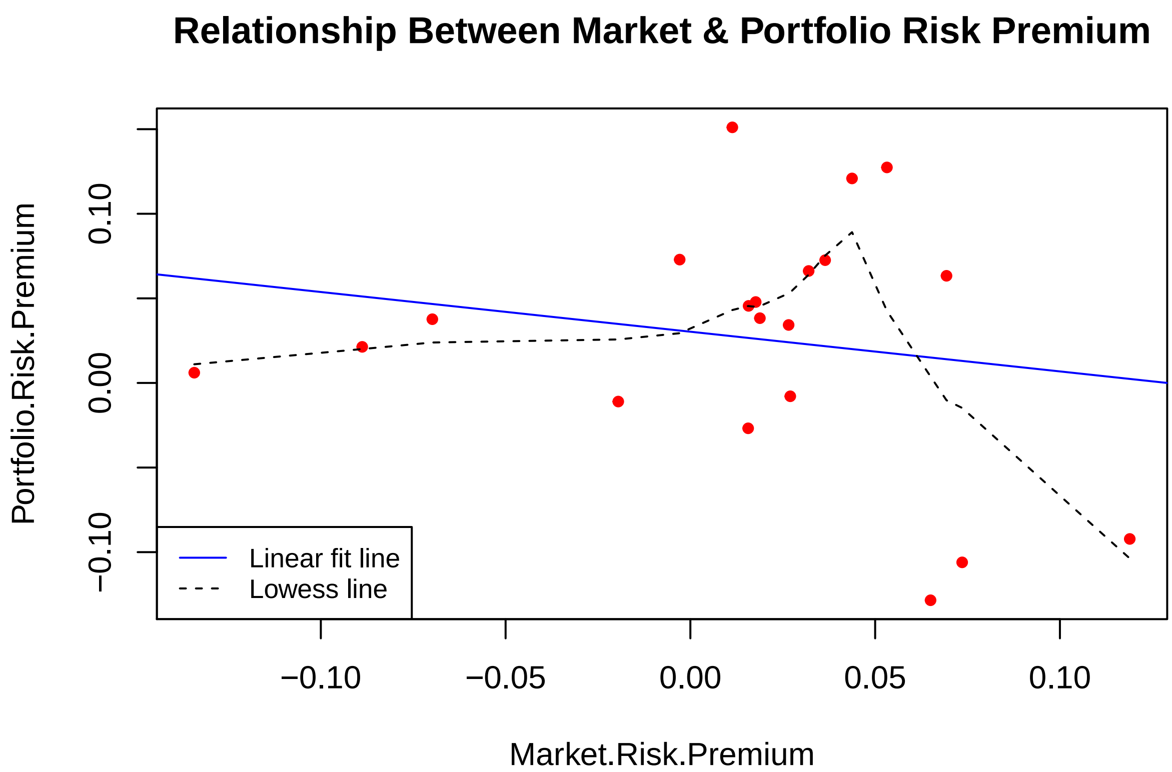
```
##                            2.5 %     97.5 %
## (Intercept)         -0.006166887 0.06674557
## Market.Risk.Premium -0.852820747 0.38341706
```

We can visual the result as follow:

```
plot(data1, col='red', main="Relationship Between Market & Portfolio Risk Premium", pch=20, cex=1)

# Add fit lines
abline(reg, col="blue")  # Regression line Y ~ X
lines(lowess(X,Y), col="black", lty=2)  # Lowess line (X,Y)

legend("bottomleft",c("Linear fit line", "Lowess line"), cex=.8, col=c("blue", "black"), lty=1:2)
```



Ex-post Forecasting

For ex-post forecasting, since our data is monthly portfolio returns with 20 observations, it makes more sense to do split the data on a 5 months ex-post forecasting instead of 10 days as recommended by the guide. Splitting the data:

```
data1_1 <- data1[1:15,]
data1_2 <- data1[16:20,]
```

Running regression on the $data1_1$:

```
regExPost <- lm(Portfolio.Risk.Premium ~ Market.Risk.Premium, data=data1_1)
summary(regExPost)
```

```
##
## Call:
## lm(formula = Portfolio.Risk.Premium ~ Market.Risk.Premium, data = data1_1)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.127170 -0.037323 -0.000716  0.043271  0.136564
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          0.01786    0.01817   0.983    0.344
## Market.Risk.Premium -0.29418    0.32971  -0.892    0.388
##
## Residual standard error: 0.07038 on 13 degrees of freedom
```

```
## Multiple R-squared:  0.0577, Adjusted R-squared:  -0.01478
## F-statistic: 0.7961 on 1 and 13 DF,  p-value: 0.3885
```

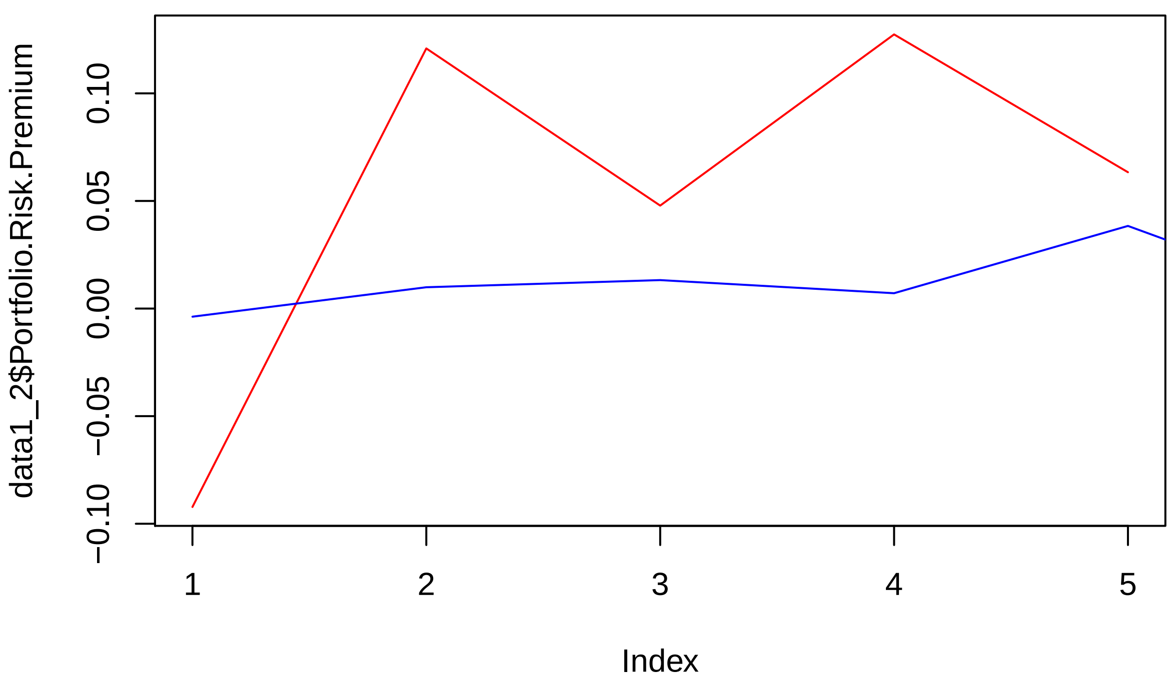
Doing the Ex-Post Forecasting:

```
predExPost <- predict(regExPost, newdata=data1_2, se.fit=TRUE)
predExPost
```

```
## $fit
##     Apr 2020     May 2020     Jun 2020     Jul 2020     Aug 2020
## -0.017120756  0.004988495  0.012657523  0.002210393 -0.002531414
##
## $se.fit
##   Apr 2020   May 2020   Jun 2020   Jul 2020   Aug 2020
## 0.04307276 0.02310715 0.01903903 0.02514972 0.02907819
##
## $df
## [1] 13
##
## $residual.scale
## [1] 0.07038463
```

Plotting the results:

```
plot(data1_2$Portfolio.Risk.Premium, type="l", col="red")
lines(regExPost$fitted.values, col="blue")
```



We have the 95% confidence intervals for our forecast as follow:

$$
\begin{aligned}
\hat{Y}_t &\pm 1.96\hat{\sigma}_\epsilon \\
-0.017120773 &\pm 1.96 * 0.07038 \\
0.004988486 &\pm 1.96 * 0.07038 \\
0.012657516 &\pm 1.96 * 0.07038 \\
0.002210383 &\pm 1.96 * 0.07038 \\
-0.002531426 &\pm 1.96 * 0.07038
\end{aligned} \tag{13}
$$

Ex-ante Forecasting

Running the regression:

```
regExAnte <- lm(Portfolio.Risk.Premium ~ Market.Risk.Premium, data=data1)
summary(regExAnte)
```

```
##
## Call:
## lm(formula = Portfolio.Risk.Premium ~ Market.Risk.Premium, data = data1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.14347 -0.04775  0.01132  0.04489  0.12346
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          0.03029    0.01735   1.746   0.0979 .
## Market.Risk.Premium -0.23470    0.29421  -0.798   0.4354
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07488 on 18 degrees of freedom
## Multiple R-squared:  0.03415,    Adjusted R-squared:  -0.01951
## F-statistic: 0.6364 on 1 and 18 DF,  p-value: 0.4354
```

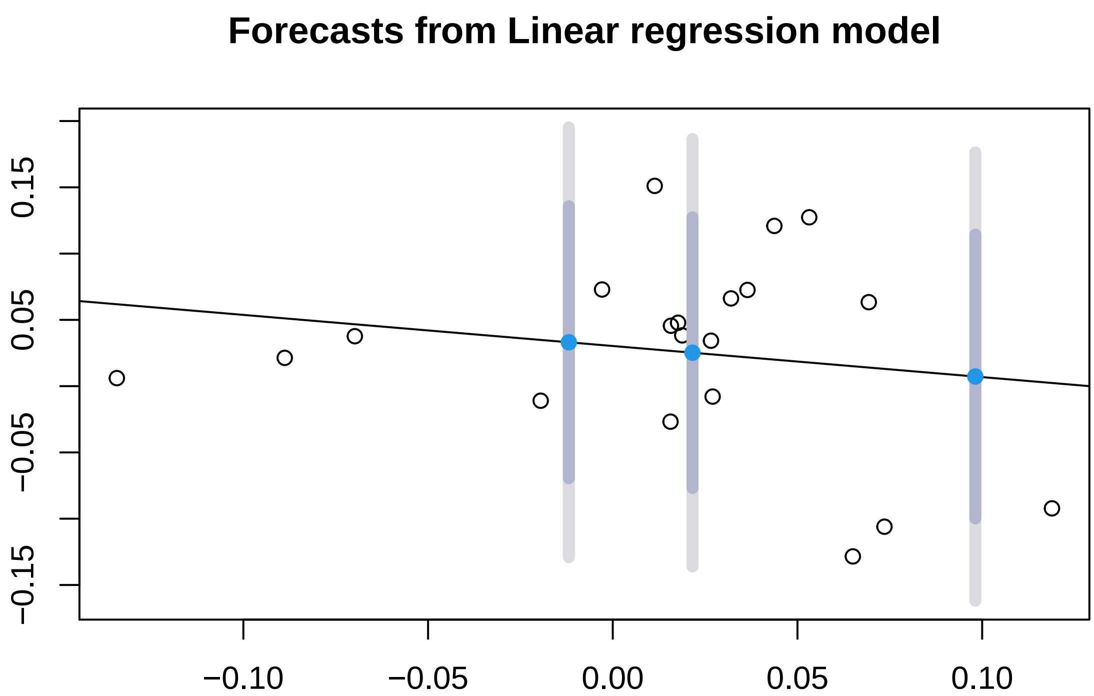
Perform a three-period-ahead ex-ante forecast of portfolio holdings returns:

```
predExAnte <- forecast::forecast(regExAnte, newdata=data.frame(Market.Risk.Premium=c(0.021584390, -0.01
predExAnte
```

```
##   Point Forecast       Lo 80     Hi 80      Lo 95     Hi 95
## 1    0.025223444 -0.07688332 0.1273302 -0.1360211 0.1864680
## 2    0.033080190 -0.06955966 0.1357200 -0.1290063 0.1951666
## 3    0.007253294 -0.09983072 0.1143373 -0.1618513 0.1763578
```

Graphing the regression and its forecast:

```
plot(predExAnte, type="l")
```



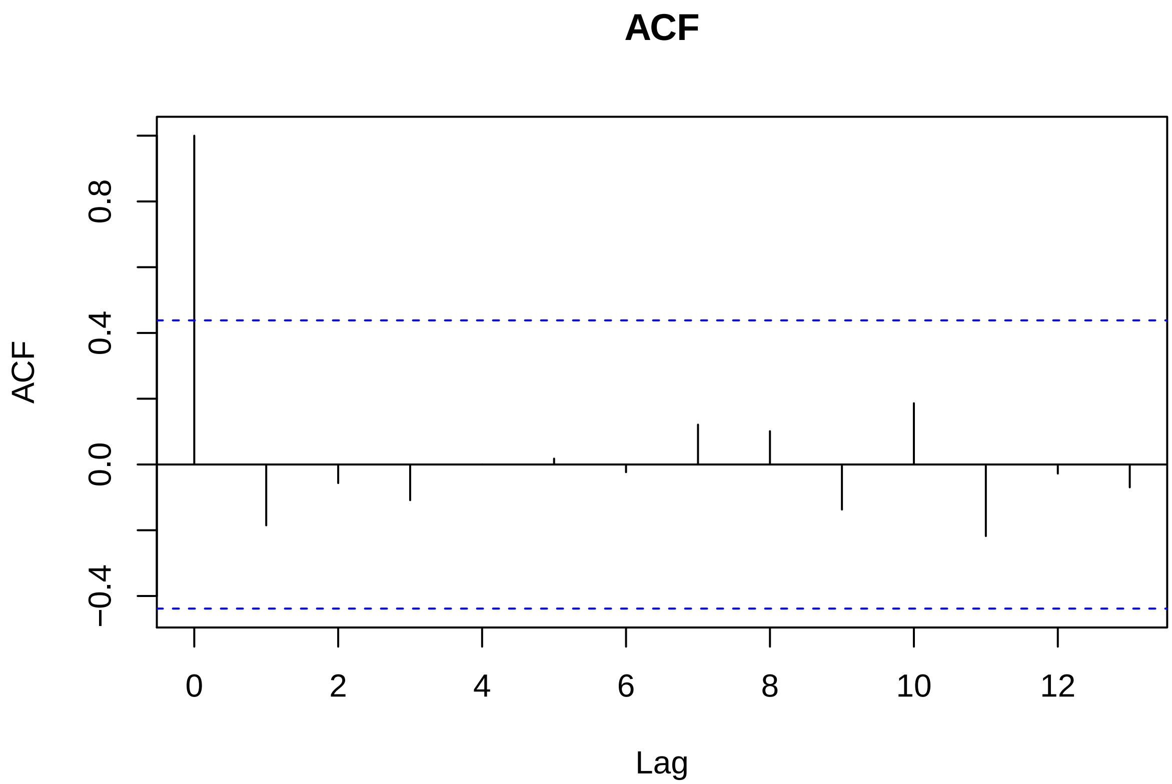
We have the 95% confidence interval for our forecast as follow:

$$\begin{gathered}\hat{Y}_t \pm 1.96\hat{\sigma}_\epsilon \\ 0.025223443 \pm 1.96 * 0.07488 \\ 0.033080188 \pm 1.96 * 0.07488 \\ 0.007253296 \pm 1.96 * 0.07488\end{gathered} \tag{14}$$

5) Do an ARIMA model of your portfolio returns and use it for three-period ahead forecasting of the returns to portfolio. Write confidence interval. Estimate the accuracy statistics.

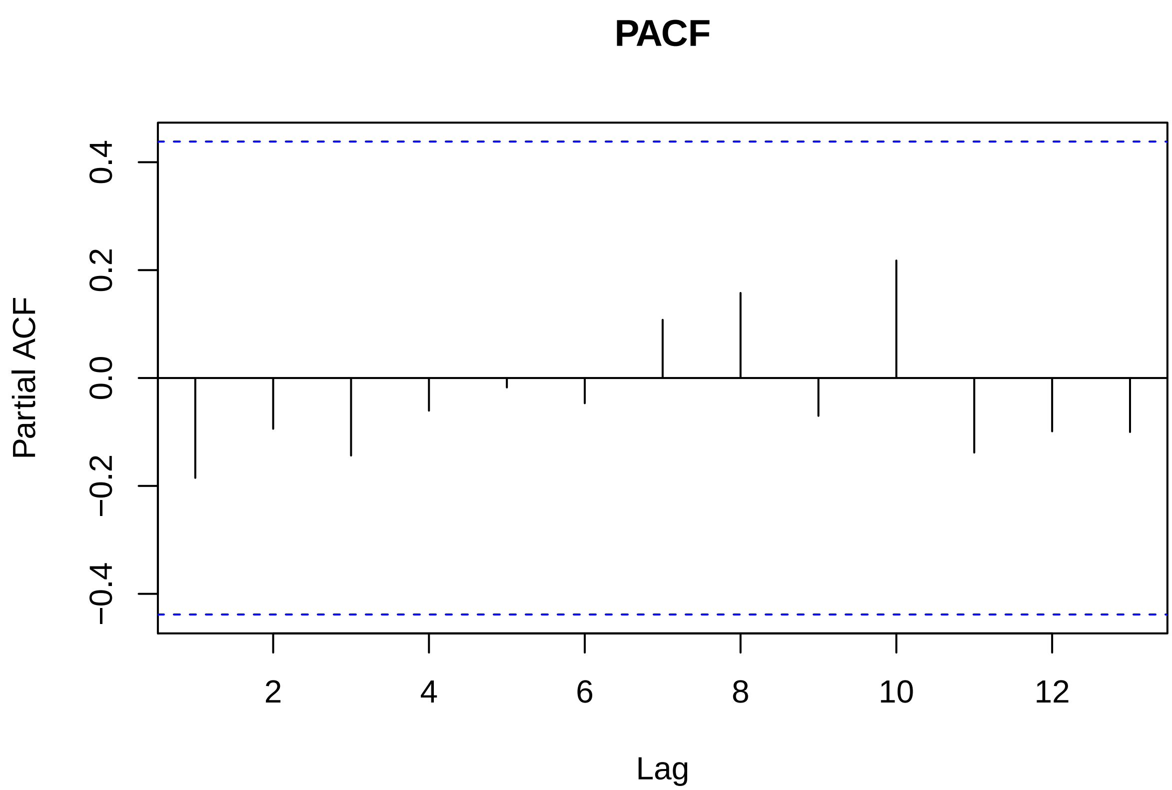
Review the ACF plot for portfolio holding monthly returns:

```
acf(rHoldings, main="ACF")
```



And the PACF:

```
pacf(rHoldings, main="PACF")
```



Perform an (A)DF Test:

```
adf.test(rHoldings)
```

```
## Augmented Dickey-Fuller Test
## alternative: stationary
##
## Type 1: no drift no trend
##      lag   ADF p.value
## [1,]   0 -4.44  0.0100
## [2,]   1 -2.41  0.0191
## [3,]   2 -1.51  0.1314
## Type 2: with drift no trend
##      lag   ADF p.value
## [1,]   0 -5.46  0.0100
## [2,]   1 -3.72  0.0111
## [3,]   2 -2.94  0.0574
## Type 3: with drift and trend
##      lag   ADF p.value
## [1,]   0 -5.64  0.0100
## [2,]   1 -3.91  0.0278
## [3,]   2 -3.27  0.0956
## ----
## Note: in fact, p.value = 0.01 means p.value <= 0.01
```

Fitting an auto-ARIMA model:

```
fitAutoARIMA <- auto.arima(rHoldings)
summary(fitAutoARIMA)
```

```
## Series: rHoldings
## ARIMA(0,0,0) with non-zero mean
##
## Coefficients:
##         mean
##       0.0280
## s.e.  0.0161
##
## sigma^2 estimated as 0.00548:  log likelihood=24.2
## AIC=-44.4   AICc=-43.7   BIC=-42.41
##
## Training set error measures:
##                           ME       RMSE        MAE      MPE     MAPE     MASE
## Training set -1.212274e-18 0.07215255 0.05578648 93.45544 128.1169 0.664064
##                    ACF1
## Training set -0.1851261
```

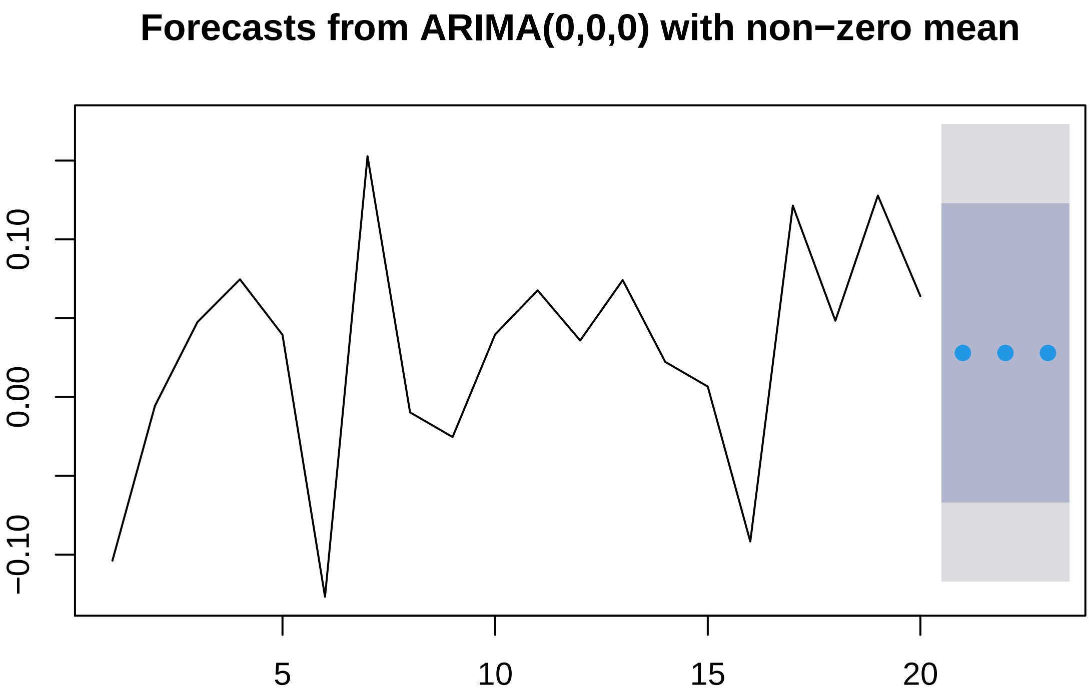
Performing a three-period ahead forecasting:

```
pred_autoARIMA <- forecast::forecast(fitAutoARIMA, h=3)
pred_autoARIMA
```

```
##    Point Forecast       Lo 80     Hi 80      Lo 95    Hi 95
```

```
## 21      0.02796782 -0.06690153 0.1228372 -0.1171223 0.173058
## 22      0.02796782 -0.06690153 0.1228372 -0.1171223 0.173058
## 23      0.02796782 -0.06690153 0.1228372 -0.1171223 0.173058
```

```
plot(pred_autoARIMA)
```



Review the confidence interval:

$$
\begin{aligned}
\hat{Y}_t \pm 1.96\hat{\sigma}_\epsilon \\
0.02796782 \pm 1.96 * \sqrt{0.00548} \\
0.02796782 \pm 1.96 * \sqrt{0.00548} \\
0.02796782 \pm 1.96 * \sqrt{0.00548}
\end{aligned}
\tag{15}
$$

Review the accuracy statistics:

```
accuracy(pred_autoARIMA)
```

```
##                         ME       RMSE        MAE      MPE     MAPE     MASE
## Training set -1.212274e-18 0.07215255 0.05578648 93.45544 128.1169 0.664064
##                    ACF1
## Training set -0.1851261
```

6) Test your ARIMA model for the stability of the ARIMA coefficients.

To test for stability of the ARIMA coefficients, we split the data in half, applies 2 separate ARIMA models and run an F-test.

```r
# Splitting the data
rHoldings1 <- rHoldings[1:10]
rHoldings2 <- rHoldings[11:20]

# Fitting ARIMA models
fitAutoARIMA1 <- auto.arima(rHoldings1)
fitAutoARIMA2 <- auto.arima(rHoldings2)

# Review the ARIMA models
summary(fitAutoARIMA1)
```

```
## Series: rHoldings1
## ARIMA(0,0,0) with zero mean
##
## sigma^2 estimated as 0.006193:  log likelihood=11.23
## AIC=-20.46   AICc=-19.96   BIC=-20.16
##
## Training set error measures:
##                       ME       RMSE        MAE MPE MAPE      MASE       ACF1
## Training set 0.008281128 0.07869685 0.06255259 100  100 0.6236386 -0.3443179
```

```r
summary(fitAutoARIMA2)
```

```
## Series: rHoldings2
## ARIMA(0,0,0) with non-zero mean
##
## Coefficients:
##         mean
##       0.0477
## s.e.  0.0187
##
## sigma^2 estimated as 0.003902:  log likelihood=14.07
## AIC=-24.14   AICc=-22.42   BIC=-23.53
##
## Training set error measures:
##                        ME       RMSE        MAE       MPE     MAPE      MASE
## Training set 6.243988e-18 0.05926408 0.04350334 -39.64687 113.2024 0.5883591
##                     ACF1
## Training set -0.09468788
```

We have the F-stat:

$$
\begin{aligned}
F =& \frac{\sigma_1^2}{\sigma_2^2} \\
=& \frac{0.006193}{0.003902} \\
=& 1.5871348 \\
F_{crit} =& 3.1789 \\
dof =& N - 1 = 10 - 1 = 9 \\
\alpha =& .05
\end{aligned} \tag{16}
$$

We notice a relative small F-stat, i.e. the differences in variances of two models are fairly insignificant, i.e. the coefficients of the ARIMA model are stable.

7) Test your ARIMA model for the existence of ARCH and GARCH and do proper corrections, if needed.

```
library(rugarch)
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'rugarch'
```

```
## The following object is masked from 'package:stats':
##
##     sigma
```

```
#Write Specification of Your GARCH Model using "sGrach" or standard GARCH Mode.
garch1 <- ugarchspec(variance.model=list(model="sGARCH", garchOrder=c(1, 1)), mean.model=list(armaOrder=
#Fit the Model to Data
rHoldings_garch1 <- ugarchfit(spec=garch1, data=rHoldings)
```

```
## Warning in .sgarchfit(spec = spec, data = data, out.sample = out.sample, :
## ugarchfit-->waring: using less than 100 data
##  points for estimation
```

```
## Warning in arima(data, order = c(modelinc[2], 0, modelinc[3]), include.mean =
## modelinc[1], : possible convergence problem: optim gave code = 1
```

```
## Warning in .sgarchfit(spec = spec, data = data, out.sample = out.sample, :
## ugarchfit-->warning: solver failer to converge.
```

```
rHoldings_garch1
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(1,0,1)
## Distribution : std
##
## Convergence Problem:
## Solver Message:
```

8) Find different time-series measures of volatility for your portfolio returns (see the volatility file posted on Blackboard) and do a three-period ahead forecasting of the portfolio volatility. Compare the different measures of volatility with GARCH.

9) Use the accuracy statistics of the different forecasting techniques to decide which technique fits the data best.

10) Test whether your portfolio index conforms to the efficient market hypothesis.

11) Find 1% and 3% daily and monthly VaR of your portfolio.

1% VaR

```
# 1% VaR Holding Portfolio
VaR_threshold <- 1

# Set scaling factor based on the period of evaluation
# In this example: {1: '1 year', 1/2: '6 months', 1/12: '1 month', 1/250: '1 day'}

# 1% 1 month VaR
scalingFactor <- 1/12
z_stat <- qnorm(VaR_threshold/100, 0, 1, lower.tail = TRUE)
VaR_1mo_p_1percent <- ROPT*scalingFactor + SDOPT*sqrt(scalingFactor) * z_stat

# 1% 1 day VaR
scalingFactor <- 1/250
VaR_1day_p_1percent <- ROPT*scalingFactor + SDOPT*sqrt(scalingFactor) * z_stat
```

The 1% monthly VaR is -0.0351323. The 1% daily VaR is -0.008037.

3% VaR

```
# 1% VaR Holding Portfolio
VaR_threshold <- 3

# Set scaling factor based on the period of evaluation
# In this example: {1: '1 year', 1/2: '6 months', 1/12: '1 month', 1/250: '1 day'}

# 1% 1 month VaR
scalingFactor <- 1/12
z_stat <- qnorm(VaR_threshold/100, 0, 1, lower.tail = TRUE)
VaR_1mo_p_3percent <- ROPT*scalingFactor + SDOPT*sqrt(scalingFactor) * z_stat

# 1% 1 day VaR
scalingFactor <- 1/250
VaR_1day_p_3percent <- ROPT*scalingFactor + SDOPT*sqrt(scalingFactor) * z_stat
```

The 1% monthly VaR is -0.0280231. The 1% daily VaR is -0.0064795.

12) Find 1% and 3% daily and monthly equity EVaR of your portfolio.

Recall that the Risk Adjusted Portfolio or RAP, is the sum weighted values of assets in the portfolio, weighted by the corresponding assets' β_i, i.e. $\sum \beta_i R_i$.

Recall the summary statistics of each stock:

Instruments	Mean Returns	Variance of Returns	Beta (5Y Monthly)
MSFT	0.0190403	0.0027112	.87
GWPH	0.0183674	0.0299313	1.96
DIS	0.0045494	0.0017214	1.08
CAT	0.0223445	0.0058996	.98
AMZN	0.0263838	0.0062955	1.3

```
betas <- c(.87, 1.96, 1.08, .98, 1.3)
RAP <- 100*sum(WOPT*betas)
```

We have the Risk Adjusted Portfolio value as 90.5391477'.

Recall that $EVaR = RAP * VaR$. Thus:

The 1% monthly EVaR is -3.1808514. The 1% daily VaR is -0.7276675.

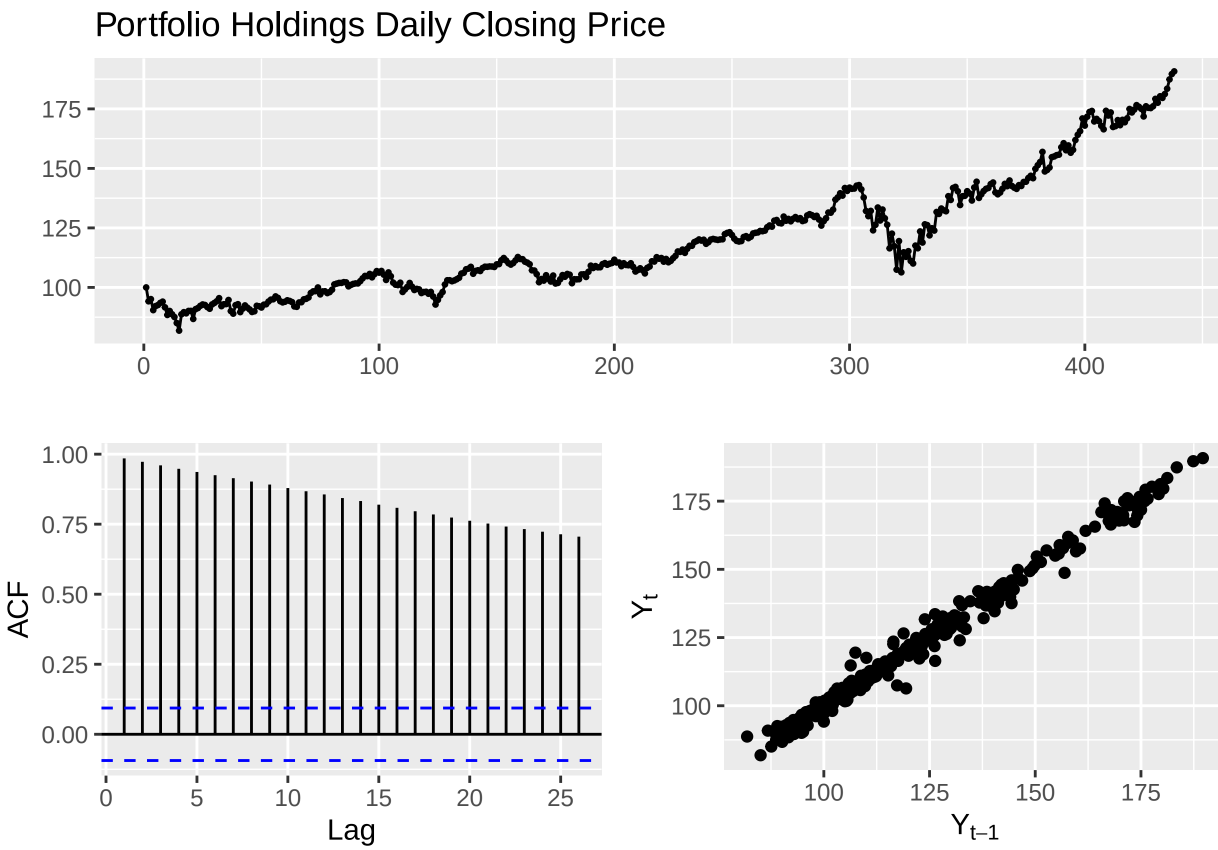
Likewise, the 3% monthly EVaR is -2.5371839. The 3% daily VaR is -0.586647.

13) Graph the security Market Line (SML) of your portfolio and test whether you would add a stock of your own choice to the portfolio or not.

14) Do an intervention function analysis of the March 15th closing of US economy due to COVID19. Did the event have any effect on return to your portfolio.

Observing the holdings data:

```
ggtsdisplay(holdings, main="Portfolio Holdings Daily Closing Price", plot.type="scatter")
```



Dividing the data into two periods, before and after the COVID-19 lockdown. The last trading day before lockdown was March 13, 2020.

```
holdings1 <- holdings[1:321,]
holdings2 <- holdings[322:438,]
```

Traditional method

We test whether the means and variance before and after the lockdown are the same:

```
var.test(holdings1, holdings2)
```

Variance Test:

```
##
##  F test to compare two variances
##
## data:  holdings1 and holdings2
## F = 0.46211, num df = 320, denom df = 116, p-value = 1.06e-07
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.3381402 0.6180695
## sample estimates:
```

```
## ratio of variances
##          0.4621054
```

From the result, we reject the null hypothesis that true ratio of variances is equal to 1, i.e. they are different before and after lockdown.

```r
t.test(holdings1, holdings2, var.equal = FALSE )
```

Means Test:

```
##
##  Welch Two Sample t-test
##
## data:  holdings1 and holdings2
## t = -21.553, df = 156.75, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -47.23185 -39.30145
## sample estimates:
## mean of x mean of y
##  108.5902  151.8568
```

Similarly, we reject the null hypothesis that true difference in means is equal to 0, i.e. the means prior to and post of lockdown are different.

Time Series Method

First, we find the order of ARIMA using the data before lockdown:

```r
fitlockdown1 <- auto.arima(holdings1)
fitlockdown1
```

```
## Series: holdings1
## ARIMA(1,1,3)
##
## Coefficients:
##           ar1    ma1      ma2     ma3
##       -0.5061  0.296  -0.0923  0.2453
## s.e.   0.1734  0.164   0.0672  0.0660
##
## sigma^2 estimated as 3.918:  log likelihood=-670.66
## AIC=1351.31   AICc=1351.51   BIC=1370.16
```

We can observe an ARIMA(1,1,3) model. Next, we define the dummy variable and fit intervention function using all data:

```r
holdings <- cbind(holdings, dummy=0)  # add a column of 0 to the df
holdings[321,2] = 1  ## replace the dummy element on March 13, 2020 to be 1

fitlockdownTS2 <- forecast::Arima(holdings[,1], xreg=holdings[,2], order=c(1, 1, 3))
summary(fitlockdownTS2)
```

```
## Series: holdings[, 1]
## Regression with ARIMA(1,1,3) errors
##
## Coefficients:
##           ar1     ma1      ma2     ma3    xreg
##       -0.8286  0.6308  -0.1162  0.0967  9.2410
## s.e.   0.1201  0.1259   0.0588  0.0484  1.9564
##
## sigma^2 estimated as 5.22:  log likelihood=-978.68
## AIC=1969.36   AICc=1969.56   BIC=1993.84
##
## Training set error measures:
##                     ME    RMSE      MAE       MPE     MAPE      MASE         ACF1
## Training set 0.23619 2.26907 1.580109 0.1481627 1.320953 0.9464687 -0.01227042
```

Next, we run VAR model to estimate the effect of lockdown on the time path of the adjustment:

```
library(vars)
```

```
## Loading required package: MASS

## Loading required package: strucchange

## Loading required package: sandwich

## Loading required package: urca

## Loading required package: lmtest

##
## Attaching package: 'vars'

## The following object is masked from 'package:aTSA':
##
##     arch.test
```

```
var <- VAR(holdings, p=5, type="const")
summary(var)
```

```
##
## VAR Estimation Results:
## =========================
## Endogenous variables: Port..Holdings.Val, dummy
## Deterministic variables: const
## Sample size: 433
## Log Likelihood: -227.053
## Roots of the characteristic polynomial:
## 1.005 0.716 0.716 0.6484 0.6484 0.6186 0.6186 0.6123 0.6123 0.5856
## Call:
## VAR(y = holdings, p = 5, type = "const")
##
```

```
##
## Estimation results for equation Port..Holdings.Val:
## ====================================================
## Port..Holdings.Val = Port..Holdings.Val.l1 + dummy.l1 + Port..Holdings.Val.l2 + dummy.l2 + Port..Hol
##
##                        Estimate Std. Error t value Pr(>|t|)
## Port..Holdings.Val.l1   0.74624    0.04936  15.119  < 2e-16 ***
## dummy.l1              -10.40456    2.41003  -4.317 1.97e-05 ***
## Port..Holdings.Val.l2   0.19929    0.06383   3.122  0.00192 **
## dummy.l2                5.97361    2.46175   2.427  0.01566 *
## Port..Holdings.Val.l3   0.12232    0.06459   1.894  0.05893 .
## dummy.l3               -1.78756    2.44485  -0.731  0.46509
## Port..Holdings.Val.l4  -0.08871    0.06398  -1.387  0.16632
## dummy.l4                3.25512    2.41385   1.349  0.17822
## Port..Holdings.Val.l5   0.02771    0.05062   0.547  0.58433
## dummy.l5               -4.76779    2.37481  -2.008  0.04532 *
## const                  -0.51285    0.55871  -0.918  0.35919
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 2.263 on 422 degrees of freedom
## Multiple R-Squared: 0.9918,  Adjusted R-squared: 0.9916
## F-statistic:  5104 on 10 and 422 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation dummy:
## ======================================
## dummy = Port..Holdings.Val.l1 + dummy.l1 + Port..Holdings.Val.l2 + dummy.l2 + Port..Holdings.Val.l3 +
##
##                         Estimate Std. Error t value Pr(>|t|)
## Port..Holdings.Val.l1 -0.0051213  0.0009958  -5.143 4.16e-07 ***
## dummy.l1               0.0577292  0.0486248   1.187  0.23580
## Port..Holdings.Val.l2  0.0020080  0.0012878   1.559  0.11970
## dummy.l2              -0.0447934  0.0496682  -0.902  0.36765
## Port..Holdings.Val.l3  0.0042940  0.0013031   3.295  0.00107 **
## dummy.l3              -0.0569365  0.0493273  -1.154  0.24905
## Port..Holdings.Val.l4 -0.0052865  0.0012908  -4.096 5.05e-05 ***
## dummy.l4               0.0773519  0.0487018   1.588  0.11297
## Port..Holdings.Val.l5  0.0041801  0.0010213   4.093 5.10e-05 ***
## dummy.l5              -0.0623321  0.0479142  -1.301  0.19400
## const                 -0.0041397  0.0112725  -0.367  0.71362
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.04566 on 422 degrees of freedom
## Multiple R-Squared: 0.118,   Adjusted R-squared: 0.09713
## F-statistic: 5.647 on 10 and 422 DF,  p-value: 6.019e-08
##
##
##
## Covariance matrix of residuals:
##                    Port..Holdings.Val    dummy
```

```
## Port..Holdings.Val             5.1223 0.019499
## dummy                          0.0195 0.002085
##
## Correlation matrix of residuals:
##                    Port..Holdings.Val  dummy
## Port..Holdings.Val             1.0000 0.1887
## dummy                          0.1887 1.0000
```

Run Impulse Response Function:

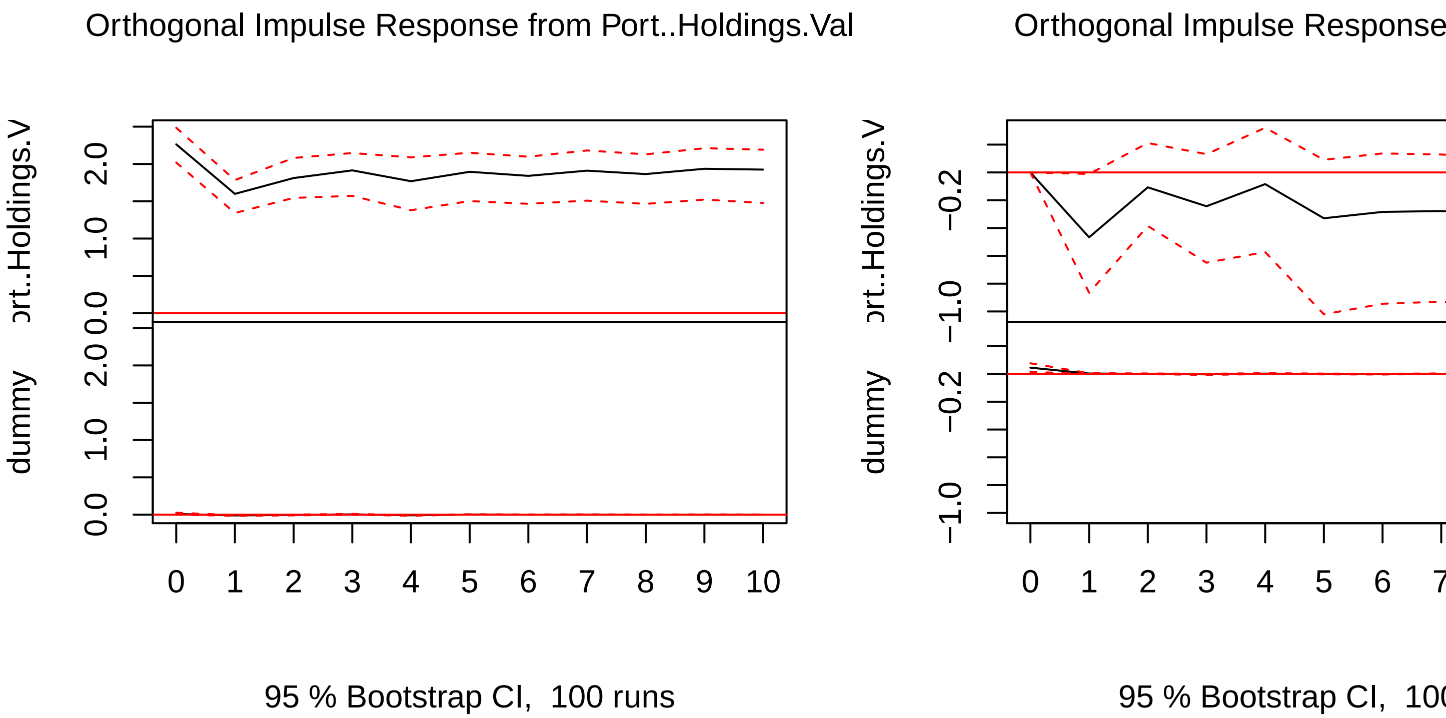
```
IRF <- irf(var, impulse.variable = 2, response.variable = 1,  t = NULL, nhor = 20, scenario = 2, draw.pl
IRF
```

```
##
## Impulse response coefficients
## $Port..Holdings.Val
##       Port..Holdings.Val         dummy
##  [1,]           2.263261  0.0086152309
##  [2,]           1.599289 -0.0110933773
##  [3,]           1.811384 -0.0046720616
##  [4,]           1.914224  0.0033899542
##  [5,]           1.769008 -0.0095603422
##  [6,]           1.894894  0.0017355418
##  [7,]           1.840826 -0.0001575245
##  [8,]           1.910511  0.0004369417
##  [9,]           1.864588 -0.0003186611
## [10,]           1.935712  0.0002702828
## [11,]           1.924721  0.0001085633
##
## $dummy
##       Port..Holdings.Val         dummy
##  [1,]         0.00000000  0.0448434181
##  [2,]        -0.46657596  0.0025887756
##  [3,]        -0.10723402  0.0005302127
##  [4,]        -0.24321963 -0.0030262899
##  [5,]        -0.08394262  0.0021496542
##  [6,]        -0.32961065 -0.0004178706
##  [7,]        -0.28388245 -0.0009768365
##  [8,]        -0.27775087  0.0008418493
##  [9,]        -0.29587562 -0.0006648184
## [10,]        -0.28187545  0.0009433472
## [11,]        -0.30371624 -0.0002335029
##
##
## Lower Band, CI= 0.95
## $Port..Holdings.Val
##       Port..Holdings.Val         dummy
##  [1,]           2.018913 -0.0004837962
##  [2,]           1.342764 -0.0155750464
##  [3,]           1.545256 -0.0105297438
##  [4,]           1.572498 -0.0024250048
##  [5,]           1.380226 -0.0148761175
##  [6,]           1.502519  0.0000220893
##  [7,]           1.466030 -0.0018238826
```

```
##  [8,]           1.508344 -0.0012821129
##  [9,]           1.464842 -0.0015149817
## [10,]           1.522649 -0.0006693348
## [11,]           1.477567 -0.0005442234
##
## $dummy
##       Port..Holdings.Val         dummy
##  [1,]          0.0000000  1.429605e-02
##  [2,]         -0.8654349 -3.879198e-04
##  [3,]         -0.3854690 -1.784272e-03
##  [4,]         -0.6491981 -7.126749e-03
##  [5,]         -0.5725577 -6.360306e-04
##  [6,]         -1.0204623 -3.123276e-03
##  [7,]         -0.9446568 -4.179674e-03
##  [8,]         -0.9295083 -2.344944e-04
##  [9,]         -0.9627560 -2.043734e-03
## [10,]         -0.9567836 -9.323969e-05
## [11,]         -0.9919105 -9.560432e-04
##
##
## Upper Band, CI= 0.95
## $Port..Holdings.Val
##       Port..Holdings.Val         dummy
##  [1,]           2.485167  0.0256352510
##  [2,]           1.784300 -0.0076908439
##  [3,]           2.080900  0.0005340283
##  [4,]           2.146507  0.0084664326
##  [5,]           2.088522 -0.0060767617
##  [6,]           2.150450  0.0038773030
##  [7,]           2.097564  0.0015415913
##  [8,]           2.181038  0.0026251061
##  [9,]           2.128816  0.0009170059
## [10,]           2.211012  0.0019648545
## [11,]           2.191994  0.0010200646
##
## $dummy
##       Port..Holdings.Val        dummy
##  [1,]         0.00000000 7.646888e-02
##  [2,]        -0.01099155 5.117875e-03
##  [3,]         0.21138092 2.999100e-03
##  [4,]         0.13138879 5.254211e-04
##  [5,]         0.32121471 5.206895e-03
##  [6,]         0.09124768 1.999899e-03
##  [7,]         0.13664320 5.574397e-04
##  [8,]         0.12894136 3.161981e-03
##  [9,]         0.10633377 2.711458e-04
## [10,]         0.12177484 2.706036e-03
## [11,]         0.09913925 9.145583e-05
```

And graph:

```
plot(IRF)
```



We can observe that the λ coefficient of the dummy is statistically significant, implying that the lockdown has changed the mean.

15) Do a 2-variable VAR between your portfolio index and S&P500 index. Graph the Impulse response function of the VAR and comment on the relationship.

We run VAR model to estimate the effect of lockdown on the time path of the adjustment:

```
holdingsGSPC <- cbind(holdings[1], adjGSPC)
var_holdingsGSPC <- VAR(holdingsGSPC, p=5, type="const")
summary(var_holdingsGSPC)
```

```
##
## VAR Estimation Results:
## =========================
## Endogenous variables: Port..Holdings.Val, GSPC.Adjusted
## Deterministic variables: const
## Sample size: 433
## Log Likelihood: -2927.249
## Roots of the characteristic polynomial:
## 1.005 0.9825 0.6617 0.6617   0.6   0.6 0.5369 0.4633 0.4633 0.3203
## Call:
## VAR(y = holdingsGSPC, p = 5, type = "const")
##
##
## Estimation results for equation Port..Holdings.Val:
## ===================================================
## Port..Holdings.Val = Port..Holdings.Val.l1 + GSPC.Adjusted.l1 + Port..Holdings.Val.l2 + GSPC.Adjuste
##
##                        Estimate Std. Error t value Pr(>|t|)
```

```
## Port..Holdings.Val.l1  0.853323   0.091886   9.287  < 2e-16 ***
## GSPC.Adjusted.l1      -0.010561   0.004820  -2.191   0.0290 *
## Port..Holdings.Val.l2 -0.124349   0.127743  -0.973   0.3309
## GSPC.Adjusted.l2       0.027894   0.006857   4.068 5.66e-05 ***
## Port..Holdings.Val.l3  0.296198   0.126971   2.333   0.0201 *
## GSPC.Adjusted.l3      -0.015601   0.006877  -2.269   0.0238 *
## Port..Holdings.Val.l4  0.131977   0.128976   1.023   0.3068
## GSPC.Adjusted.l4      -0.013502   0.006951  -1.942   0.0528 .
## Port..Holdings.Val.l5 -0.140938   0.093620  -1.505   0.1330
## GSPC.Adjusted.l5       0.010356   0.004884   2.120   0.0346 *
## const                  2.525557   1.541540   1.638   0.1021
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 2.272 on 422 degrees of freedom
## Multiple R-Squared: 0.9917,  Adjusted R-squared: 0.9915
## F-statistic:  5065 on 10 and 422 DF,  p-value: < 2.2e-16
##
##
## Estimation results for equation GSPC.Adjusted:
## ==============================================
## GSPC.Adjusted = Port..Holdings.Val.l1 + GSPC.Adjusted.l1 + Port..Holdings.Val.l2 + GSPC.Adjusted.l2 +
##
##                       Estimate Std. Error t value Pr(>|t|)
## Port..Holdings.Val.l1 -2.73384    1.74529  -1.566 0.118001
## GSPC.Adjusted.l1       0.84914    0.09155   9.275  < 2e-16 ***
## Port..Holdings.Val.l2 -3.34468    2.42636  -1.378 0.168787
## GSPC.Adjusted.l2       0.60916    0.13025   4.677 3.92e-06 ***
## Port..Holdings.Val.l3  8.95245    2.41170   3.712 0.000233 ***
## GSPC.Adjusted.l3      -0.53439    0.13062  -4.091 5.15e-05 ***
## Port..Holdings.Val.l4  0.01486    2.44979   0.006 0.995164
## GSPC.Adjusted.l4      -0.18706    0.13203  -1.417 0.157289
## Port..Holdings.Val.l5 -2.64953    1.77823  -1.490 0.136977
## GSPC.Adjusted.l5       0.23545    0.09278   2.538 0.011512 *
## const                 55.95718   29.28018   1.911 0.056670 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 43.15 on 422 degrees of freedom
## Multiple R-Squared: 0.9671,  Adjusted R-squared: 0.9664
## F-statistic:  1242 on 10 and 422 DF,  p-value: < 2.2e-16
##
##
##
## Covariance matrix of residuals:
##                    Port..Holdings.Val GSPC.Adjusted
## Port..Holdings.Val              5.161          83.2
## GSPC.Adjusted                  83.199        1862.0
##
## Correlation matrix of residuals:
##                    Port..Holdings.Val GSPC.Adjusted
## Port..Holdings.Val             1.0000        0.8487
```

```
## GSPC.Adjusted                  0.8487       1.0000
```

Run Impulse Response Function:

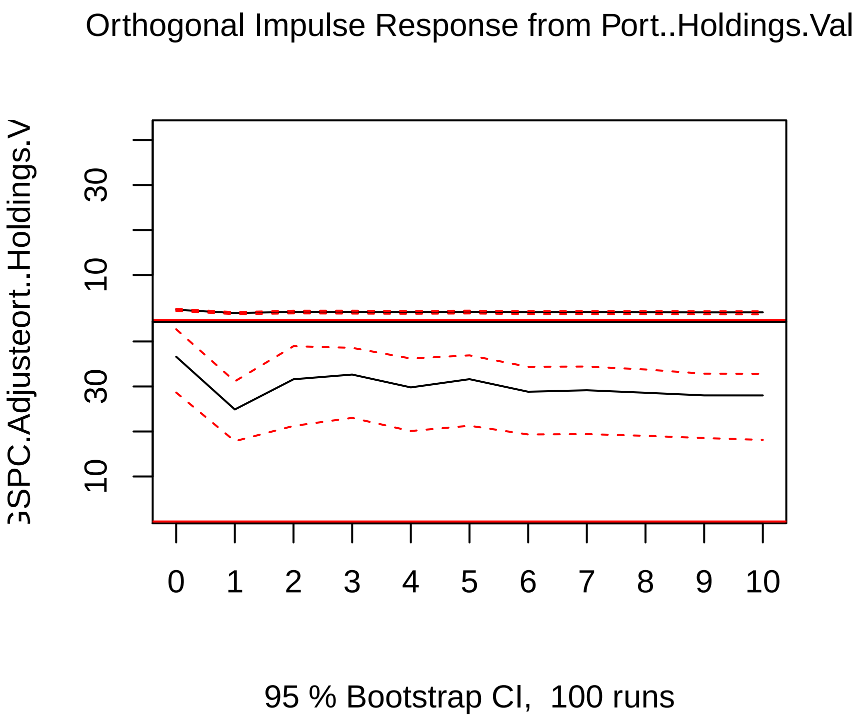
```r
IRF_holdingsGSPC <- irf(var_holdingsGSPC, impulse.variable = 2, response.variable = 1,  t = NULL, nhor =
IRF_holdingsGSPC
```

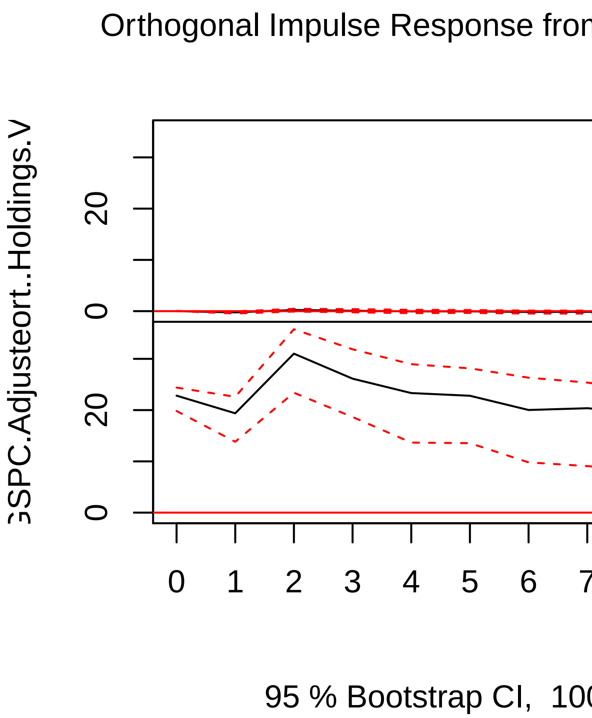
```
##
## Impulse response coefficients
## $Port..Holdings.Val
##       Port..Holdings.Val GSPC.Adjusted
##  [1,]           2.271805      36.62260
##  [2,]           1.551823      24.88685
##  [3,]           1.800432      31.60063
##  [4,]           1.805407      32.64863
##  [5,]           1.730139      29.79220
##  [6,]           1.816083      31.62035
##  [7,]           1.707027      28.82619
##  [8,]           1.727045      29.15661
##  [9,]           1.711979      28.60435
## [10,]           1.690659      28.01395
## [11,]           1.696096      28.01130
##
## $GSPC.Adjusted
##       Port..Holdings.Val GSPC.Adjusted
##  [1,]         0.00000000      22.82077
##  [2,]        -0.24100389      19.37796
##  [3,]         0.22625928      31.01500
##  [4,]         0.08000182      26.13278
##  [5,]        -0.05253380      23.32636
##  [6,]        -0.04613530      22.79885
##  [7,]        -0.16119512      20.02485
##  [8,]        -0.13979861      20.36737
##  [9,]        -0.18764883      19.39490
## [10,]        -0.20453684      19.07850
## [11,]        -0.21988492      18.80821
##
##
## Lower Band, CI= 0.95
## $Port..Holdings.Val
##       Port..Holdings.Val GSPC.Adjusted
##  [1,]           2.016695      28.65700
##  [2,]           1.305371      17.83893
##  [3,]           1.472684      21.22690
##  [4,]           1.433018      23.02006
##  [5,]           1.387982      20.09758
##  [6,]           1.459083      21.27221
##  [7,]           1.319101      19.33533
##  [8,]           1.313197      19.41420
##  [9,]           1.302355      19.03205
## [10,]           1.263942      18.53493
## [11,]           1.259864      18.12424
##
```

```
## $GSPC.Adjusted
##       Port..Holdings.Val GSPC.Adjusted
##  [1,]         0.00000000     19.825384
##  [2,]        -0.45589915     13.825713
##  [3,]        -0.07739182     23.393852
##  [4,]        -0.22633433     18.653823
##  [5,]        -0.38701749     13.661634
##  [6,]        -0.34428821     13.552101
##  [7,]        -0.46935501      9.799243
##  [8,]        -0.47293149      9.075486
##  [9,]        -0.54075226      8.042412
## [10,]        -0.58309011      7.144083
## [11,]        -0.61842910      6.628372
##
##
## Upper Band, CI= 0.95
## $Port..Holdings.Val
##       Port..Holdings.Val GSPC.Adjusted
##  [1,]           2.465618      42.71467
##  [2,]           1.746104      31.14152
##  [3,]           2.068185      38.94677
##  [4,]           2.069553      38.58842
##  [5,]           1.997216      36.20953
##  [6,]           2.077432      36.91153
##  [7,]           1.954584      34.37385
##  [8,]           1.956697      34.42840
##  [9,]           1.946236      33.77332
## [10,]           1.929647      32.84892
## [11,]           1.922229      32.82692
##
## $GSPC.Adjusted
##       Port..Holdings.Val GSPC.Adjusted
##  [1,]         0.00000000      24.38710
##  [2,]        -0.06211952      22.59738
##  [3,]         0.45761294      35.77586
##  [4,]         0.39111916      31.86790
##  [5,]         0.25707901      28.95416
##  [6,]         0.21274170      28.14651
##  [7,]         0.11163281      26.32342
##  [8,]         0.10263663      25.36816
##  [9,]         0.05465048      23.93218
## [10,]         0.05335256      23.39399
## [11,]         0.03656294      23.22802
```

And graph:

```
plot(IRF_holdingsGSPC)
```





We can observe that the λ coefficient of the S&P 500 is significant, meaning that the movement of S&P 500 has an effect on the movement of the portfolio holdings.
Citations

"Amazon.com, Inc. (AMZN) Stock Price, News, Quote & History." Yahoo! Finance, Yahoo!, 14 Nov. 2020, ca.finance.yahoo.com/quote/amzn/?p=amzn.

"Caterpillar, Inc. (CAT) Stock Price, News, Quote & History." Yahoo! Finance, Yahoo!, 13 Nov. 2020, ca.finance.yahoo.com/quote/CAT/?p=CAT.

JeffCoxCNBCcom. "Fed Hikes Rate, Lowers 2019 Projection to 2 Increases." CNBC, CNBC, 19 Dec. 2018, www.cnbc.com/2018/12/19/fed-hikes-rates-by-a-quarter-point-.html.

"GW Pharmaceuticals Plc (GWPH) Stock Price, News, Quote & History." Yahoo! Finance, Yahoo!, 13 Nov. 2020, ca.finance.yahoo.com/quote/GWPH/?p=GWPH.

Moyer, Liz. "We're Finding out Now Why the Stock Market Tanked in December." CNBC, CNBC, 9 Jan. 2019, www.cnbc.com/2019/01/09/markets-december-tumble-may-have-hinted-at-profit-revisions-to-come.html.

"Microsoft Corporation (MSFT) Stock Price, News, Quote & History." Yahoo! Finance, Yahoo!, 14 Nov. 2020, ca.finance.yahoo.com/quote/msft/?p=msft.

"Walt Disney Company (The) (DIS) Stock Price, News, Quote & History." Yahoo! Finance, Yahoo!, 14 Nov. 2020, ca.finance.yahoo.com/quote/dis/?p=dis.