QF202 Final

Om Mehta

2022-05-11

```r
library(knitr)
opts_chunk$set(tidy.opts=list(width.cutoff=60),tidy=TRUE)
```

```r
# Problem 1
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
```

```r
library(forecast)
library(fBasics)
```

```
## Loading required package: timeDate

## Loading required package: timeSeries

##
## Attaching package: 'timeSeries'

## The following object is masked from 'package:zoo':
##
##     time<-

##
## Attaching package: 'fBasics'
```

```
## The following object is masked from 'package:TTR':
##
##     volatility
```

```r
library("TSA")
```

```
## Registered S3 methods overwritten by 'TSA':
##   method       from
##   fitted.Arima forecast
##   plot.Arima   forecast

##
## Attaching package: 'TSA'

## The following objects are masked from 'package:timeDate':
##
##     kurtosis, skewness

## The following objects are masked from 'package:stats':
##
##     acf, arima

## The following object is masked from 'package:utils':
##
##     tar
```

```r
library(orcutt)
```

```
## Loading required package: lmtest
```

```r
library(tseries)
library(timeSeries)
library(readr)
```

```
##
## Attaching package: 'readr'

## The following object is masked from 'package:TSA':
##
##     spec
```

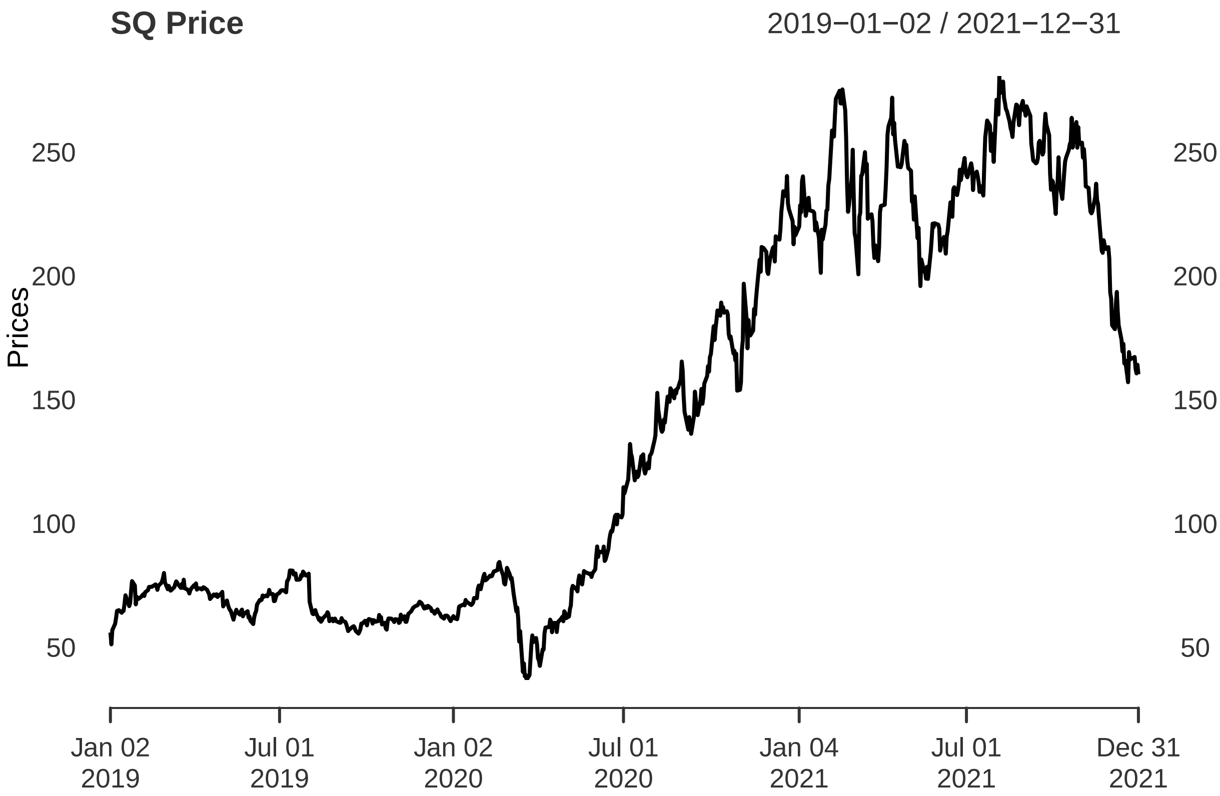
```r
library(nlme)
```

```
##
## Attaching package: 'nlme'

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

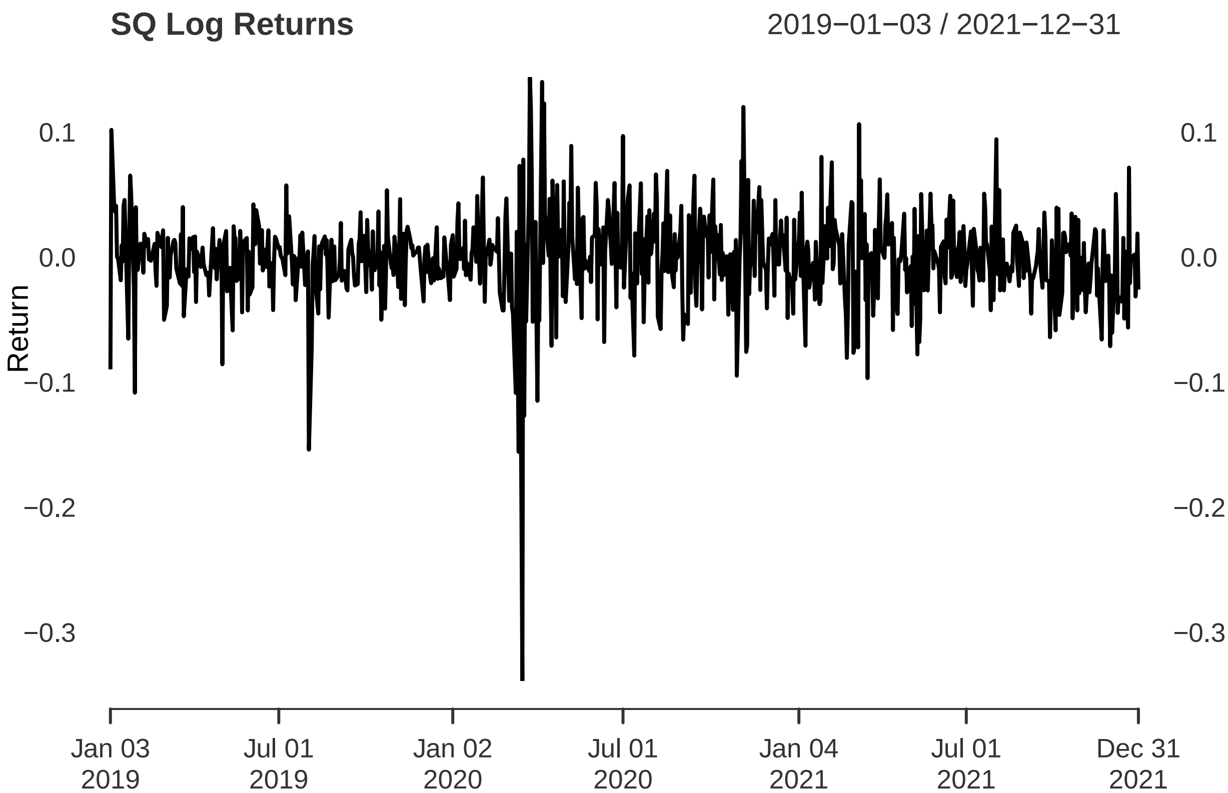
```r
library(pracma)
```

```
##
## Attaching package: 'pracma'

## The following objects are masked from 'package:fBasics':
##
##     akimaInterp, inv, kron, pascal
```

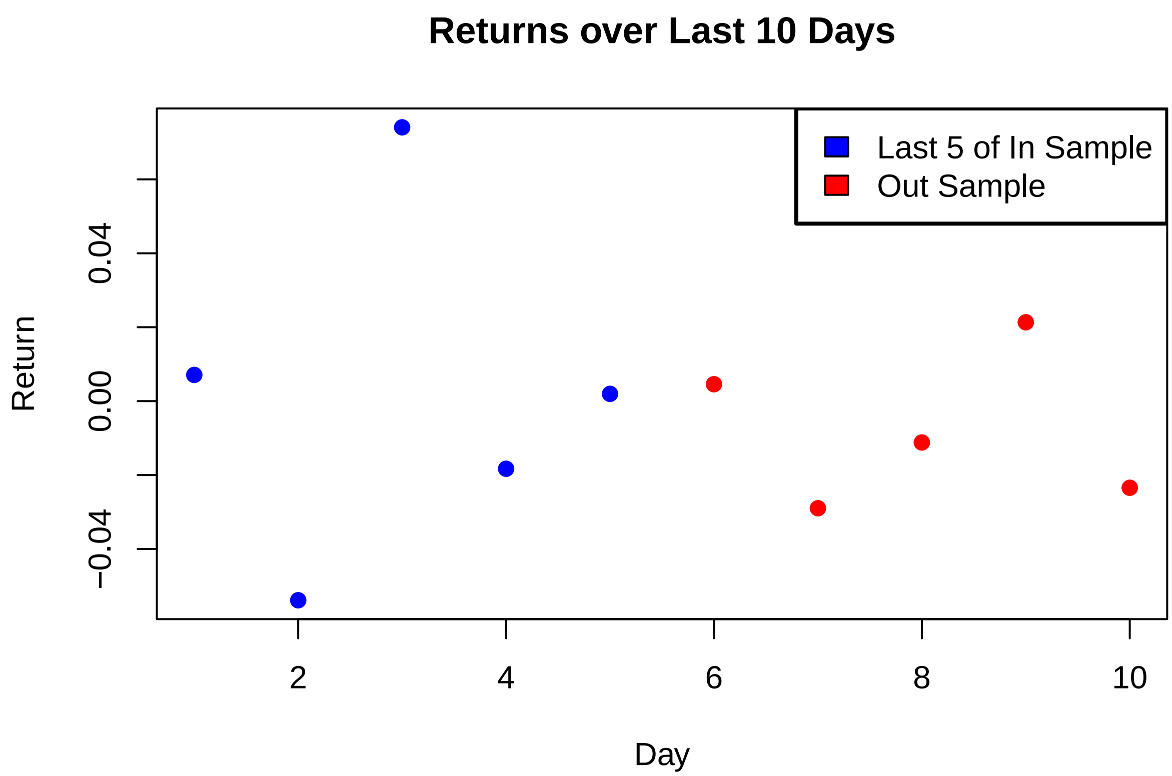
```r
square <- getSymbols("SQ", from = "2019-01-01", to = "2022-01-01")
logret <- dailyReturn(SQ$SQ.Adjusted, type = "log")
SQret <- logret[2:length(logret)]
plot(SQ$SQ.Adjusted, main = "SQ Price", xlab = "Date", ylab = "Prices",
    grid.col = NA)
```



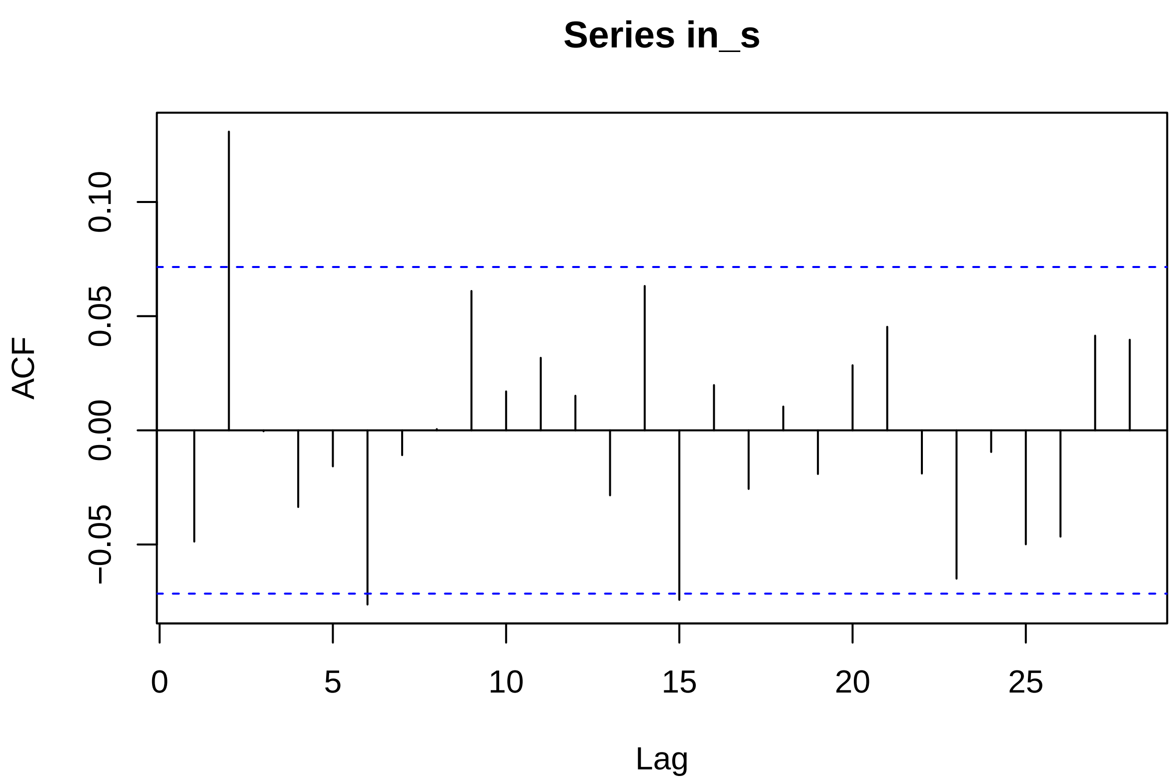

```r
plot(SQret, main = "SQ Log Returns", xlab = "Date", ylab = "Return",
    grid.col = NA)
```



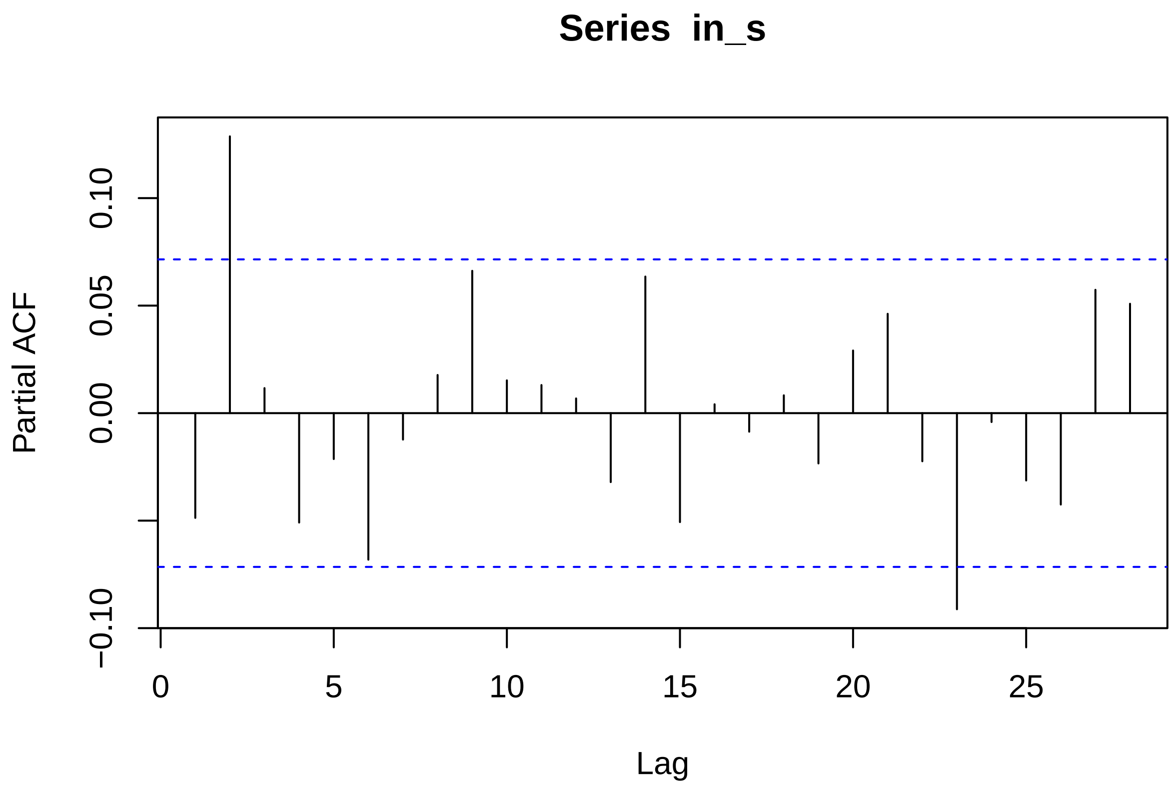

```r
# Problem 2
out_s <- SQret[(length(SQret) - 4):length(SQret)]
in_s <- SQret[1:(length(SQret) - 5)]
last_5 <- in_s[(length(in_s) - 4):length(in_s)]
days = as.vector(c(last_5, out_s))

plot(days, main = "Returns over Last 10 Days", xlab = "Day",
    ylab = "Return", col = ifelse(1:10 < 6, "blue", "red"), pch = 19)
legend(x = "topright", box.col = "black", bg = "white", box.lwd = 2,
    legend = c("Last 5 of In Sample", "Out Sample"), fill = c("blue",
        "red"))
```




```r
# Above two lines allow me to see the full output

# Problem 3
acf(in_s)  #ACF is used to determine the best order for an MA model
```



```r
pacf(in_s)  #PACF is used to determine the best order for an AR model
```




```r
# Creating AR model
for (i in 1:23) {
    m <- arima(in_s, order = c(1, 0, 0), seasonal = list(order = c(1,
        0, 0), period = i))
    print(i)
    print(m$aic)
}
```

```
## [1] 1
## [1] -2843.936
## [1] 2
## [1] -2856.959
## [1] 3
## [1] -2844.062
## [1] 4
## [1] -2844.955
## [1] 5
## [1] -2844.392
## [1] 6
## [1] -2848.751
## [1] 7
## [1] -2844.212
## [1] 8
## [1] -2844.052
## [1] 9
## [1] -2847.036
```

```
## [1] 10
## [1] -2844.412
## [1] 11
## [1] -2844.924
## [1] 12
## [1] -2844.233
## [1] 13
## [1] -2844.535
## [1] 14
## [1] -2846.739
## [1] 15
## [1] -2847.941
## [1] 16
## [1] -2844.229
## [1] 17
## [1] -2844.516
## [1] 18
## [1] -2844.094
## [1] 19
## [1] -2844.28
## [1] 20
## [1] -2844.758
## [1] 21
## [1] -2845.731
## [1] 22
## [1] -2844.359
## [1] 23
## [1] -2847.604
```

```r
# I am using a for loop here to determine the lowest AIC
# value of all AR models between orders 1 and 23, which
# indicated the range of statistically significant values
# that could be a possible order in the PACF plot. The
# recommended order of this AR model is 2, as it has the
# lowest AIC value.

# Creating MA model
for (i in 2:15) {
    m <- arima(in_s, order = c(0, 0, 1), seasonal = list(order = c(0,
        0, 1), period = i))
    print(i)
    print(m$aic)
}
```

```
## [1] 2
## [1] -2857.885
## [1] 3
## [1] -2843.683
## [1] 4
## [1] -2844.564
## [1] 5
## [1] -2843.971
## [1] 6
## [1] -2848.214
```

```
## [1] 7
## [1] -2843.803
## [1] 8
## [1] -2843.676
## [1] 9
## [1] -2846.612
## [1] 10
## [1] -2843.987
## [1] 11
## [1] -2844.566
## [1] 12
## [1] -2843.866
## [1] 13
## [1] -2844.246
## [1] 14
## [1] -2846.243
## [1] 15
## [1] -2847.622
```

```r
# The recommended order of this MA model is also 2, as it
# shows the lowest AIC value, meaning there is the best
# trade off for parameter usage and log-likelihood; where
# AIC = 2k-2l


# Creating ARMA model - brute force method
aic.matrix <- function(data, ar_order, ma_order) {
    AIC_matrix <- matrix(NA, nrow = ar_order + 1, ncol = ma_order +
        1)
    for (i in 0:ar_order) {
        for (j in 0:ma_order) {
            tem <- tryCatch(arima(data, order = c(i, 0, j))$aic,
                error = function(cond) {
                  message(cond)
                  message(". AR: ", i, "; MA: ", j)
                  return(NA)
                })

            AIC_matrix[i + 1, j + 1] <- tem
        }
    }
    AIC_matrix
}

# find the best model with the lowest AIC
matrix <- aic.matrix(in_s, 10, 10)
```

```
## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
```

```
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1
```

```
which(matrix == min(na.omit(matrix)), arr.ind = TRUE) - 1
```

```
##      row col
## [1,]   8   9
```

```r
# The recommended order of the ARMA model is (8, 0, 4).


# Here, the models are outputted.
ar_model <- arima(in_s, order = c(2, 0, 0), include.mean = FALSE)
ar_model
```

```
##
## Call:
## arima(x = in_s, order = c(2, 0, 0), include.mean = FALSE)
##
## Coefficients:
##           ar1     ar2
##       -0.0422  0.1327
## s.e.   0.0363  0.0364
##
## sigma^2 estimated as 0.001296:  log likelihood = 1430.95,  aic = -2857.89
```

```r
ma_model <- arima(in_s, order = c(0, 0, 2), include.mean = FALSE)
ma_model
```

```
##
## Call:
## arima(x = in_s, order = c(0, 0, 2), include.mean = FALSE)
##
## Coefficients:
##           ma1     ma2
##       -0.0443  0.1427
## s.e.   0.0363  0.0365
##
## sigma^2 estimated as 0.001294:  log likelihood = 1431.47,  aic = -2858.93
```

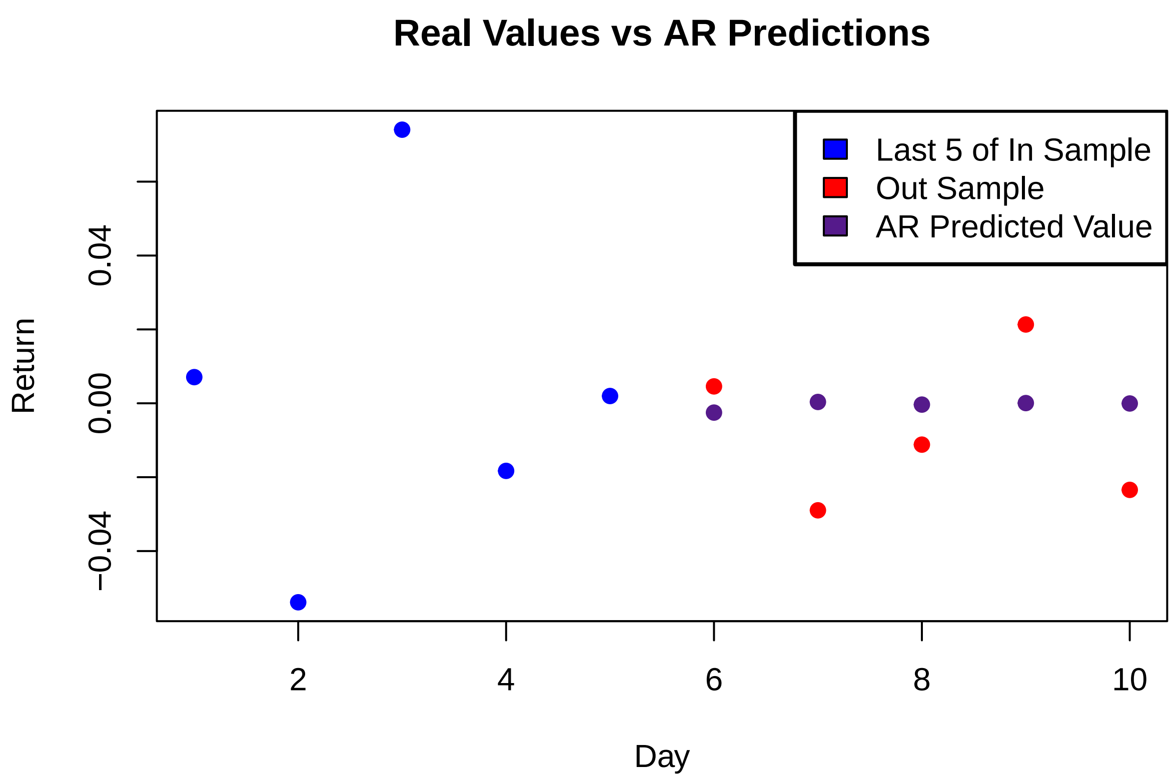
```r
arma_model <- arima(in_s, order = c(8, 0, 4))
arma_model
```

```
##
## Call:
## arima(x = in_s, order = c(8, 0, 4))
##
## Coefficients:
```

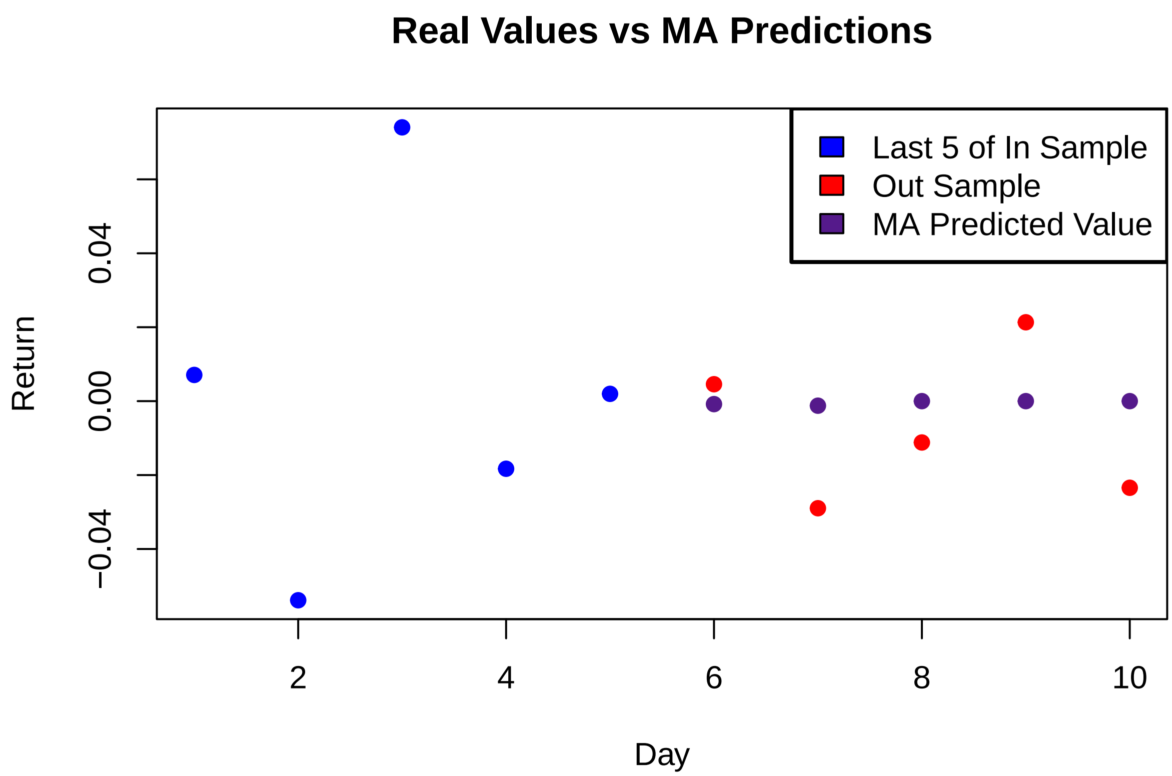
```
## Warning in sqrt(diag(x$var.coef)): NaNs produced
```

```
##          ar1      ar2      ar3      ar4     ar5      ar6     ar7      ar8
##       0.7769  -0.8633  -0.1088  -0.0130  0.0179  -0.0862  0.0274  -0.0597
## s.e.  0.2588      NaN      NaN   0.2506  0.0523   0.0645  0.0459   0.0403
##           ma1     ma2      ma3     ma4  intercept
##       -0.8332  1.0556  -0.0378  0.1033     0.0015
## s.e.   0.2573     NaN      NaN  0.2382     0.0013
##
## sigma^2 estimated as 0.001262:  log likelihood = 1438.54,  aic = -2851.07
```

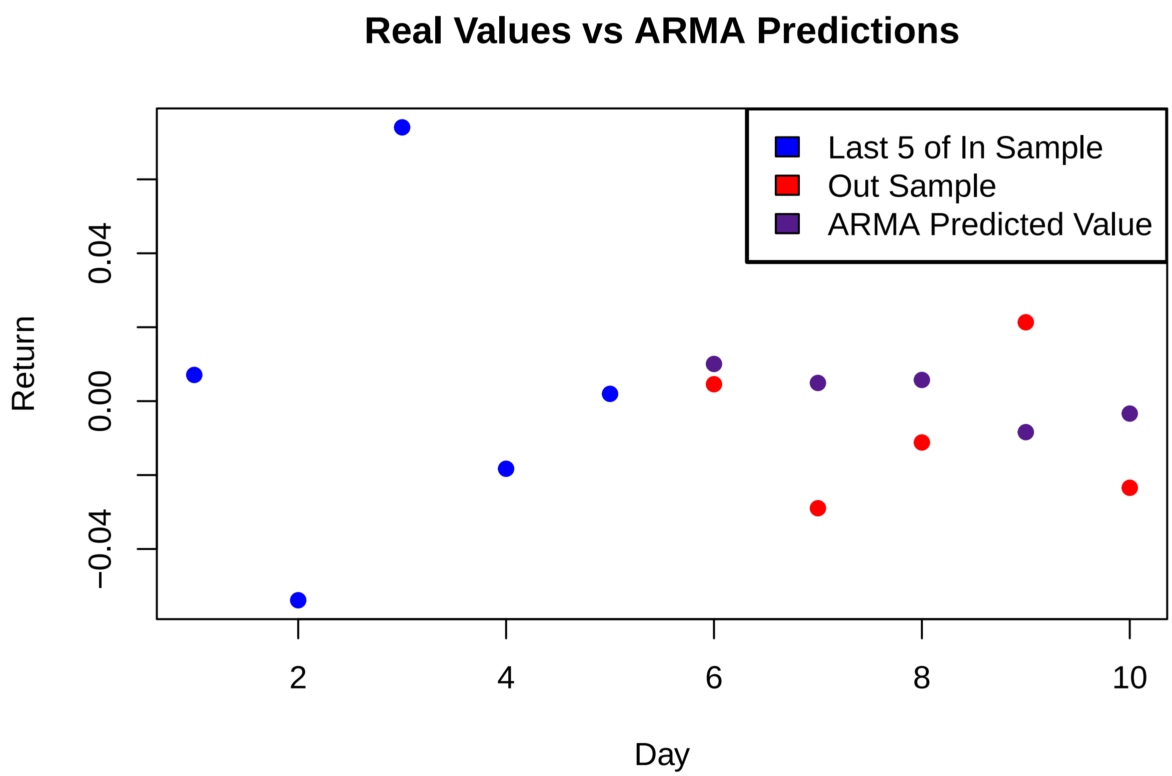
```r
# Problem 4

# Plotting real values against the predicted values from
# the AR model
pred_1 <- predict(ar_model, n.ahead = 5)
pred_1 <- as.vector(pred_1$pred)
plot(days, main = "Real Values vs AR Predictions", ylab = "Return",
    xlab = "Day", col = ifelse(1:10 < 6, "blue", "red"), pch = 19)
points(x = 6:10, pred_1, col = "purple4", pch = 19)
legend(x = "topright", box.col = "black", bg = "white", box.lwd = 2,
    legend = c("Last 5 of In Sample", "Out Sample", "AR Predicted Value"),
    fill = c("blue", "red", "purple4"))
```




```r
# Plotting real values against the predicted values from
# the MA model
pred_2 <- predict(ma_model, n.ahead = 5)
pred_2 <- as.vector(pred_2$pred)
plot(days, main = "Real Values vs MA Predictions", ylab = "Return",
    xlab = "Day", col = ifelse(1:10 < 6, "blue", "red"), pch = 19)
points(x = 6:10, pred_2, col = "purple4", pch = 19)
legend(x = "topright", box.col = "black", bg = "white", box.lwd = 2,
    legend = c("Last 5 of In Sample", "Out Sample", "MA Predicted Value"),
    fill = c("blue", "red", "purple4"))
```




```r
# Plotting real values against the predicted values from
# the ARMA model
pred_3 <- predict(arma_model, n.ahead = 5)
pred_3 <- as.vector(pred_3$pred)
plot(days, main = "Real Values vs ARMA Predictions", ylab = "Return",
    xlab = "Day", col = ifelse(1:10 < 6, "blue", "red"), pch = 19)
points(x = 6:10, pred_3, col = "purple4", pch = 19)
legend(x = "topright", box.col = "black", bg = "white", box.lwd = 2,
    legend = c("Last 5 of In Sample", "Out Sample", "ARMA Predicted Value"),
    fill = c("blue", "red", "purple4"))
```




```r
# Problem 5
sse_ar <- sum((pred_1 - out_s)^2)
sse_ar
```

```
## [1] 0.002027463
```

```r
sse_ma <- sum((pred_2 - out_s)^2)
sse_ma
```

```
## [1] 0.001927625
```

```r
sse_arma <- sum((pred_3 - out_s)^2)
sse_arma
```

```
## [1] 0.002750773
```

```r
# The MA model has the lowest sum of squared errors, and
# therefore is the best model of the three.
```

```r
# Interview Problem

data <- read.csv("Final_Data.csv")
y <- as.ts(data$Y)
x <- as.ts(data$X)

# Build regression
```

```r
linreg <- lm(data$Y ~ data$X)

# Check if residuals follow assumptions
Box.test(linreg$residuals, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  linreg$residuals
## X-squared = 1211, df = 1, p-value < 2.2e-16
```

```r
# The Ljung-Box test has a very low p value, indicating
# that the residuals have an autocorrelation present.
# Therefore, the residuals do not satisfy the regression
# assumptions.

# Since the residuals do not satisfy the regression
# assumptions, we use the Cochrane procedure to fit a
# regression with time series errors.
cochrane <- cochrane.orcutt(lm(y ~ x))
cochrane
```

```
## Cochrane-orcutt estimation for first order autocorrelation
##
## Call:
## lm(formula = y ~ x)
##
##  number of interaction: 3
##  rho 0.492193
##
## Durbin-Watson statistic
## (original):    1.01581 , p-value: 1.089e-265
## (transformed): 2.01295 , p-value: 6.785e-01
##
##  coefficients:
## (Intercept)           x
##    1.999989    1.506950
```

```r
# Fitting ARIMAX model
aic.matrix <- function(x1, y1, ar_order, ma_order) {
    AIC_matrix <- matrix(NA, nrow = ar_order + 1, ncol = ma_order +
        1)
    for (i in 0:ar_order) {
        for (j in 0:ma_order) {
            tem <- tryCatch(arimax(y, order = c(i, 0, j), xtransf = x,
                transfer = list(c(0, 0)))$aic, error = function(cond) {
                message(cond)
                message(". AR: ", i, "; MA: ", j)
                return(NA)
            })

            AIC_matrix[i + 1, j + 1] <- tem
```

```r
        }
    }
    AIC_matrix
}
matrix <- aic.matrix(x, y, 3, 3)
```

```
## . AR: 0; MA: 0
```

```r
which(matrix == min(na.omit(matrix)), arr.ind = TRUE) - 1
```

```
##      row col
## [1,]   1   0
```

```r
# The optimal model is an ARIMAX(1, 0, 0).

arimax_model = arimax(y, order = c(1, 0, 0), xtransf = x, transfer = list(c(0,
    0)))
arimax_model
```

```
##
## Call:
## arimax(x = y, order = c(1, 0, 0), xtransf = x, transfer = list(c(0, 0)))
##
## Coefficients:
##          ar1  intercept  T1-MA0
##       0.4922     2.0001   1.507
## s.e.  0.0123     0.0088   0.004
##
## sigma^2 estimated as 0.09938:  log likelihood = -1322.94,  aic = 2651.88
```

```r
# The T1-MA0 coefficient of 1.507 is the same as the
# coefficient from the Cochrane linear model, which
# suggests that the ARIMAX model has low variability and
# small errors.
```

```r
normalTest(cochrane$residuals, method = "sw")
```

```
##
## Title:
##  Shapiro - Wilk Normality Test
##
## Test Results:
##   STATISTIC:
##     W: 0.9998
##   P VALUE:
##     0.9449
##
## Description:
##  Sun May 15 19:21:14 2022 by user:
```

```r
normalTest(arimax_model$residuals, method = "sw")
```

```
##
## Title:
##  Shapiro - Wilk Normality Test
##
## Test Results:
##   STATISTIC:
##     W: 0.9998
##   P VALUE:
##     0.9611
##
## Description:
##  Sun May 15 19:21:14 2022 by user:
```

```r
# Using the Shapiro-Wilk criteria for the residuals of each
# model, both have high p values indicating that the
# residuals are normally distributed.


Box.test(cochrane$residuals, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  cochrane$residuals
## X-squared = 1212, df = 1, p-value < 2.2e-16
```

```r
Box.test(arimax_model$residuals, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  arimax_model$residuals
## X-squared = 0.21314, df = 1, p-value = 0.6443
```

```r
# Using the Ljung-Box test, the cochrane model has a low p
# value, indicating that an autocorrelation exists and
# therefore white noise is present. The arimax model
# however has a high p value of 0.6443, indicating that no
# autocorrelation exists and therefore white noise is not
# present. Given this information, I conclude that the
# arimax model fits the data better.
```