Theoretical Problems

Monday, May 7, 2018 2:04 PM

Nicholas Colonna

"I pledge my honor that I have abided by the Stevens Honor System." N. Colonna

1. P(reservation cancelled) = P(C) = 0.20

P(reservation used) = P(U) = 0.80

• Accepted 52 reservations, only 50 tables

a) $P(\text{not enough tables}) = \underbrace{P(0 \text{ cancelled}) + P(1 \text{ cancelled})}_{\text{if 2 or more cancel, there's room}}$

$$= \binom{52}{0}(0.2)^0(0.8)^{52} + \binom{52}{1}(0.2)^1(0.8)^{51}$$

$$= (0.8)^{52} + 52(0.2)(0.8)^{51}$$

$$= 0.00000913 + 0.00011874$$

$$\boxed{P(\text{not enough tables}) = 0.000127877}$$

b) $P(\text{more than 3 empty}) = 1 - P(3 \text{ or less empty}) = 1 - P(5 \text{ or less cancel})$

$$= 1 - [P(0 \text{ empty}) + P(1 \text{ empty}) + P(2 \text{ empty}) + P(3 \text{ empty})]$$

$$= 1 - \left[\binom{52}{0}(0.2)^0(0.8)^{52} + \binom{52}{1}(0.2)^1(0.8)^{51} + \binom{52}{2}(0.2)^2(0.8)^{50} + \binom{52}{3}(0.2)^3(0.8)^{49} + \binom{52}{4}(0.2)^4(0.8)^{48} + \binom{52}{5}(0.2)^5(0.8)^{47}\right]$$

$$= 1 - [1(.2)^0(.8)^{52} + 52(.2)^1(.8)^{51} + 1326(.2)^2(.8)^{50} + 22100(.2)^3(.8)^{49} + 270725(.2)^4(.8)^{48} + 2598960(.2)^5(.8)^{47}]$$

$$= 1 - [0.00000913 + 0.00011874 + 0.00075701 + 0.00315421 + 0.00965979 + 0.02318349]$$

$$= 1 - 0.03688237$$

$$\boxed{P(\text{more than 3 empty}) = 0.96311763}$$

2. $f: \mathbb{R} \to \mathbb{R}$ such that $f(x) = \dfrac{2}{\pi(1+x^2)} \mathbb{1}_{[0,\infty]}(x)$ $\qquad \mathbb{1}_{[0,\infty]}(x) = \begin{cases} 0, & x<0 \\ 1, & x \geq 0 \end{cases}$

Show that f is a PDF.

* We know that the integral of a pdf $= 1$, so lets integrate.

$$\int_{-\infty}^{\infty} f(x)dx = \int_{-\infty}^{\infty} \frac{2}{\pi(1+x^2)} \mathbb{1}_{[0,\infty]}(x)\,dx = \int_0^{\infty} \frac{2}{\pi(1+x^2)}\,dx$$

$$= \frac{2}{\pi}\int_0^{\infty} \frac{1}{1+x^2}\,dx$$

$$= \frac{2}{\pi} \tan^{-1}(x)\Big]_0^{\infty}$$

$$= \frac{2}{\pi}\tan^{-1}(\infty) - \frac{2}{\pi}(0)$$

$$= \frac{2}{\pi}\left[\lim_{x\to\infty}(\tan^{-1}(\infty))\right]$$

$$= \frac{2}{\pi} \times \left(\frac{\pi}{2}\right)$$

$$= 1$$

$\int f(x) = 1 \quad \therefore f$ is a probability density function

---

3. $X \sim \exp(\lambda)$, $\lambda > 0$

$$f(x) = \begin{cases} \lambda e^{-\lambda x}, & \text{if } x>0 \\ 0, & \text{otherwise} \end{cases}$$

Prove that for every $s>0$ and $t>0$,

$$P(X \geq s+t \mid X \geq s) = P(X \geq t)$$

Using the CDF of the exponential distribution:

$$P(X < s+t \mid X \geq s) = \frac{P(X < s+t \cap X \geq s)}{P(X \geq s)}$$

$$= \frac{P(s \leq X < s+t)}{P(X \geq s)}$$

$$= \frac{e^{-\lambda s} - e^{-\lambda(s+t)}}{e^{-\lambda s}}$$

$$= \frac{e^{-\lambda s}}{e^{-\lambda s}} - \frac{e^{-\lambda s - \lambda t}}{e^{-\lambda s}}$$

$$= 1 - \frac{e^{-\lambda s} e^{-\lambda t}}{e^{-\lambda s}}$$

$$P(X < s+t \mid X \geq s) = 1 - e^{-\lambda t}$$

Therefore,

$$P(X \geq s+t \mid X \geq s) = P(X \geq t)$$
$$1 - P(X < s+t \mid X \geq s) = P(X \geq t)$$
$$1 - [1 - e^{-\lambda t}] = e^{-\lambda t}$$
$$e^{-\lambda t} = e^{-\lambda t}$$

$\therefore$ I have proven that $P(X \geq s+t \mid X \geq s) = P(X \geq t)$

---

4. AR(1) model: $r_t = 0.1 + 0.3 r_{t-1} + a_t$, $\sigma_a = 0.025$

Assume $r_{1000} = 0.001$

- Calculate the 1-step & 2-step ahead forecast at $t = 1000$
- Calculate the standard deviation of corresponding forecast errors
- Compute the numerical values for the 1, 2, and 3 lag autocorrelation

$X_t = r_t$

1 step ahead forecast: at time $t = 1000$

$$\hat{X}_{1000}(1) = E[r_{1001} \mid F_{1000}] = E[0.1 + 0.3 r_{1000} + a_{1001} \mid F_{1000}]$$
$$= 0.1 + 0.3 E[r_{1000} \mid F_{1000}] + E[\underbrace{a_{1001} \mid F_{1000}}_{\text{independent}}]$$

$$= 0.1 + 0.3(0.001) + \underbrace{E[a_{1001}]}_{=0}$$

$$\boxed{\hat{X}_{1000}(1) = 0.1003}$$

1 step forecast Standard Deviation

1 step forecast error $\rightarrow e_{1000}(1) = X_{1001} - \hat{X}_{1000}(1)$

$$e_{1000}(1) = X_{1001} - 0.1003$$

$$V(e_{1000}(1)) = V((X_{1001} - 0.1003)) = V(a_{n+1}) = \sigma_a^2 = 0.025^2 = 0.000625$$

$$\sigma_{e_{1000}(1)} = \sqrt{V(e_{1000}(1))} = \sqrt{\sigma_a^2}$$

$$\boxed{\sigma_{e_{1000}(1)} = \sigma_a = 0.025}$$

2 steps ahead forecast: at time $t=1000$

$$\hat{X}_{1000}(2) = E[X_{1002} | F_{1000}] = E[0.1 + 0.3X_{1001} + a_{1002} | F_{1000}]$$

$$= 0.1 + 0.3E[X_{1001}|F_{1000}] + E[a_{1002}|F_{1000}]$$

independent

$$= 0.1 + 0.3[\hat{X}_{1000}(1)] + \underbrace{E[a_{1002}]}_{=0}$$

$$= 0.1 + 0.3(0.1003)$$

$$\boxed{\hat{X}_{1000}(2) = 0.13009}$$

2 steps forecast Standard Deviation

2 steps forecast error $\rightarrow e_{1000}(2) = X_{1002} - \hat{X}_{1000}(2)$

$$e_{1000}(2) = X_{1002} - 0.13009$$

$$V(e_{1000}(2)) = V(X_{1002} - 0.13009)$$

$$= V(0.1 + 0.3X_{1001} - 0.13009 + a_{1002})$$

$$= V(0.1 + 0.3(X_{1001} - \hat{X}_{1000}(1)) + a_{1002})$$

$$= V(0.1) + 0.09V(X_{1001} - \hat{X}_{1000}(1)) + V(a_{1002})$$

$$= 0.09\, V(e_{1000}(1)) + \sigma_a^2$$

$$= 0.09\sigma_a^2 + \sigma_a^2$$
$$= 1.09\sigma_a^2$$
$$V(e_{1000}(2)) = 1.09(.025)^2 = 0.00068125$$

$$\sigma_{e_{1000}(2)} = \sqrt{V(e_{1000}(2))} = \sqrt{0.00068125}$$
$$\boxed{\sigma_{e_{1000}(2)} = 0.0261}$$

Lag Autocorrelations

$$\rho_\ell = \frac{\gamma_\ell}{\gamma_0}$$

$$\gamma_\ell = cov(r_t, r_{t-\ell}) = E[(r_t - \mu)(r_{t-\ell} - \mu)]$$
$$\gamma_\ell = \varphi\gamma_{\ell-1} = 0.3\gamma_\ell$$
$$\gamma_0 = V(r_t) = \frac{\sigma_a^2}{1-\varphi^2} = \frac{.025^2}{1-.3^2} = 0.00068681$$
$$\gamma_1 = 0.3 \times \gamma_0 = 0.3(.00068681) = 0.00020604$$
$$\gamma_2 = 0.3 \times \gamma_1 = 0.3(.00020604) = 0.00006181$$
$$\gamma_3 = 0.3 \times \gamma_2 = 0.3(.00006181) = 0.00001854$$

Lag 1: $\rho_1 = \frac{\gamma_1}{\gamma_0} = \frac{0.00020604}{0.00068681} \Rightarrow \boxed{\rho_1 = 0.3}$

Lag 2: $\rho_2 = \frac{\gamma_2}{\gamma_0} = \frac{0.00006181}{0.00068681} \Rightarrow \boxed{\rho_2 = 0.09}$

Lag 3: $\rho_3 = \frac{\gamma_3}{\gamma_0} = \frac{0.00001854}{0.00068681} \Rightarrow \boxed{\rho_3 = 0.027}$

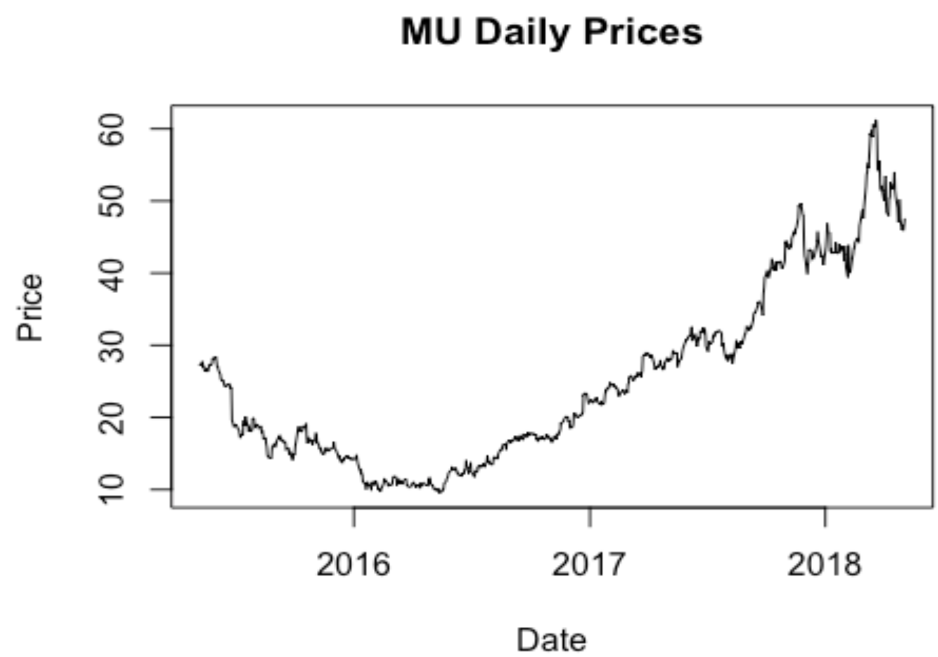
Final Exam

Nicholas Colonna

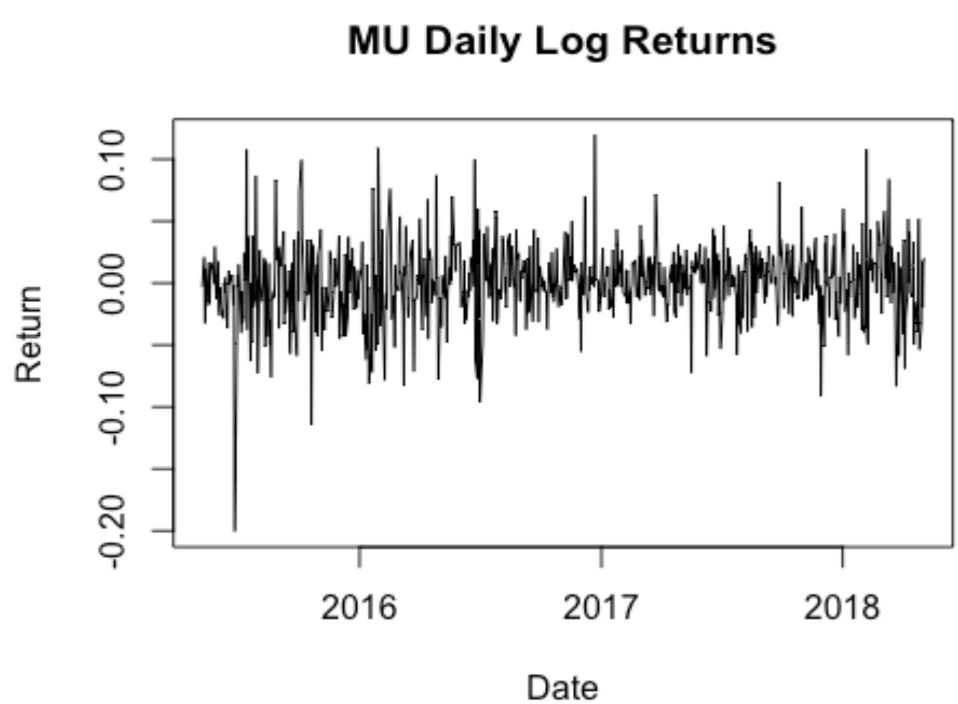
5/7/2018

Practical Problems

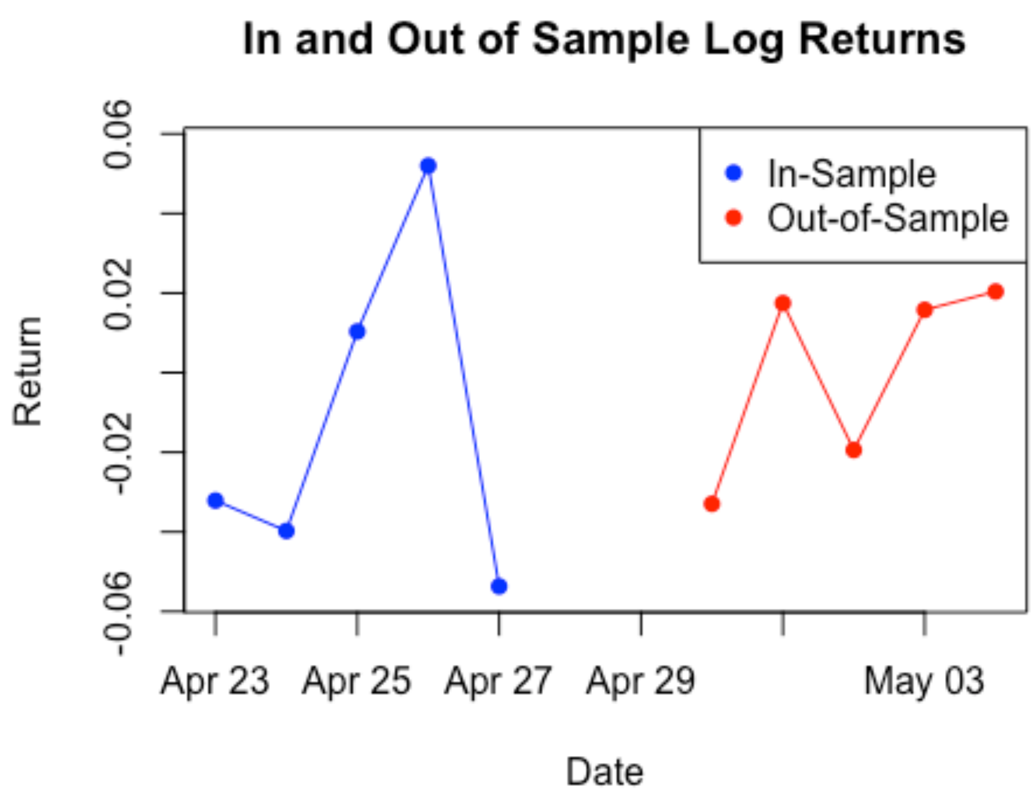
```r
library("forecast")

## Loading required package: zoo

##
## Attaching package: 'zoo'

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

## Loading required package: timeDate

## This is forecast 7.3

library("fBasics")

## Loading required package: timeSeries

##
## Attaching package: 'timeSeries'

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

library("fUnitRoots")




#Part A
MU <- read.csv("MU.csv", header=T)
MU$Date <- as.Date(MU$Date, format="%Y-%m-%d")


prices <- data.frame(MU$Date, MU$Adj.Close)
colnames(prices) <- c("Date", "Price")
plot(prices, main="MU Daily Prices", type="l")
```



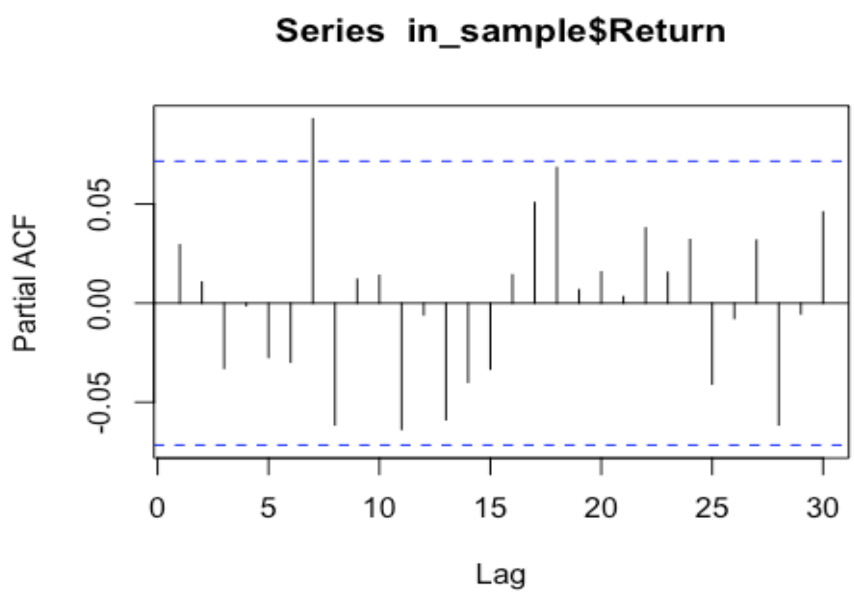

```r
returns <- data.frame(MU$Date[2:755], diff(log(prices$Price), lag=1))
colnames(returns) <- c("Date", "Return")
plot(returns, main="MU Daily Log Returns", type="l")
```



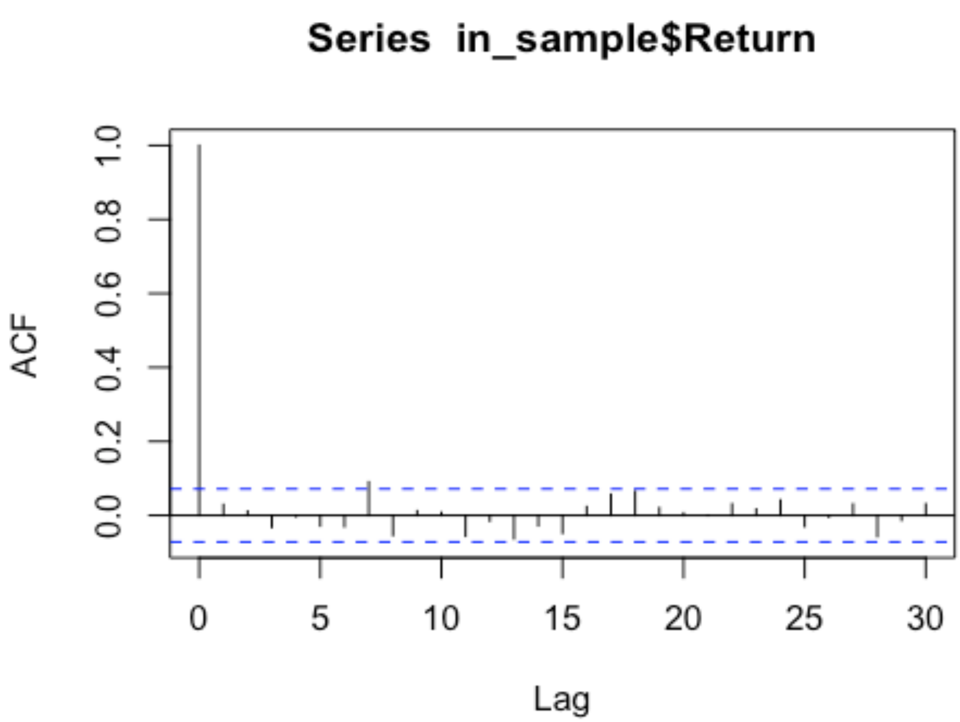
```
#Part B
in_sample <- data.frame(returns$Date[1:749], returns$Return[1:749])
colnames(in_sample) <- c("Date", "Return")

out_sample <- data.frame(returns$Date[750:754], returns$Return[750:754])
colnames(out_sample) <- c("Date", "Return")

in_sample_last5 <- data.frame(in_sample$Date[745:749], in_sample$Return[745:749])
colnames(in_sample_last5) <- c("Date", "Return")

plot(in_sample_last5, main="In and Out of Sample Log Returns", col="blue", xlim=as.Date(c("2018-04-23","2018-05-04")), ylim=c(min(out_sample$Return, in_sample_last5$Return)-.002, max(out_sample$Return, in_sample_last5$Return)+.005), type="o", pch=16)
lines(out_sample, col="red", type="o", pch=16)
legend("topright", c("In-Sample", "Out-of-Sample"), col=c("blue", "red"), pch=16)
```



```
#Part C
pacf(in_sample$Return, lag.max=30) #The PACF is used to give a recommended orde
r for the AR model. In this case it is order 7.
```



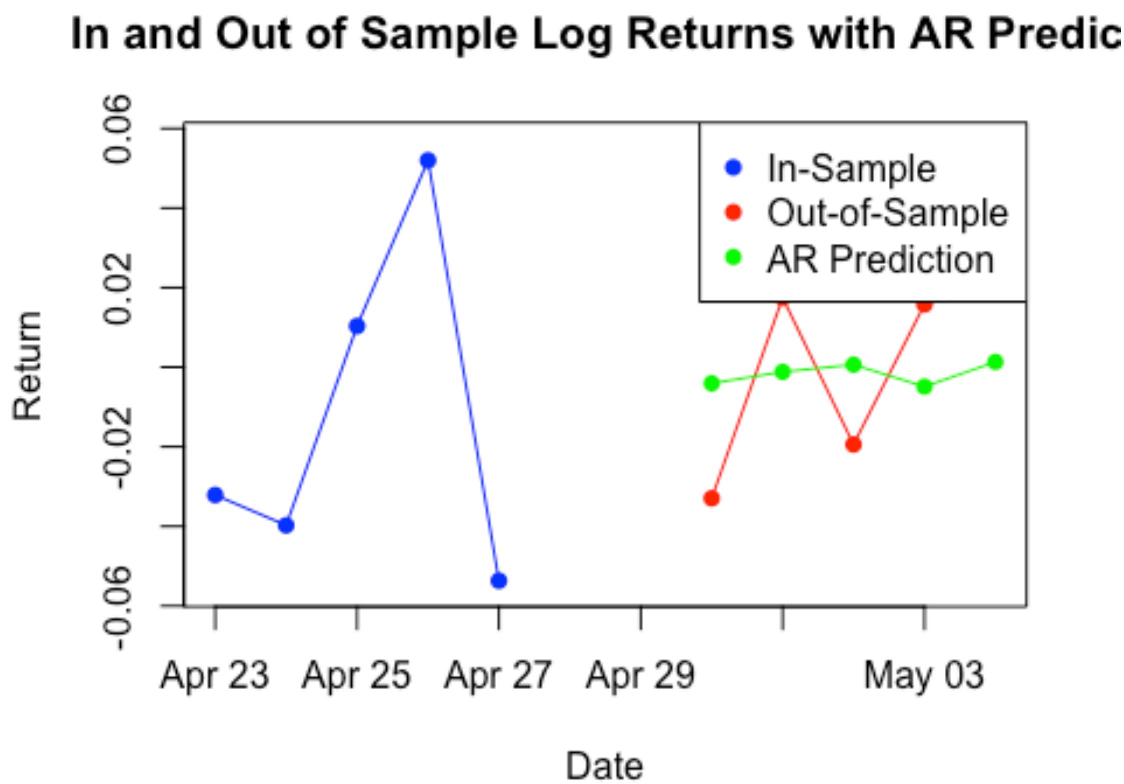

```
acf(in_sample$Return,lag.max=30) #The ACF is used to give a recommended order f
or the MA model. In this case it is order 7.
```



```
adfTest(in_sample$Return, lags=7)   #one test we learned in class was the Augme
nted Dickey-Fuller Test (ADF). If the p-value is small, it is likely that there
are no unit roots. Since the p-value is significant here, we can conclude no un
it-root.

## Warning in adfTest(in_sample$Return, lags = 7): p-value smaller than
## printed p-value

##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 7
##   STATISTIC:
##     Dickey-Fuller: -9.7529
##   P VALUE:
##     0.01
##
## Description:
##  Fri May 11 17:12:05 2018 by user:



Box.test(in_sample$Return, type="Ljung") #When no lag, this gives us a pretty h
igh p-value, so we accept H0 that all correlations are 0.

##
##  Box-Ljung test
##
## data:  in_sample$Return
## X-squared = 0.64955, df = 1, p-value = 0.4203



Box.test(in_sample$Return, lag=7, type="Ljung") #When we run the test again at
lag 7, we still accept H0, since the p-value is greater than 0.05. However, we
saw improvement from no lag. This could potentially mean this lag isn’t the bes
t fit for our model.

##
##  Box-Ljung test
##
## data:  in_sample$Return
## X-squared = 9.033, df = 7, p-value = 0.2503
```

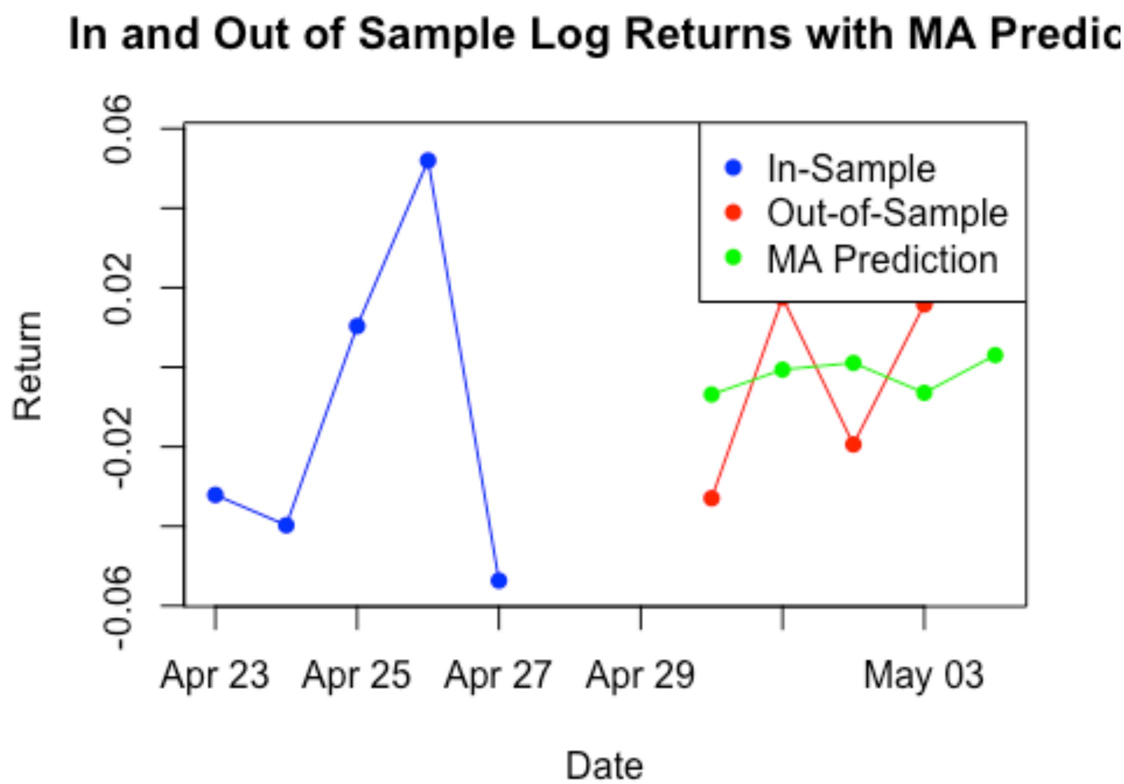
```
max.p=7
max.q=7
model.aic <- matrix(NA,nrow=max.p+1,ncol=max.q+1)
for (p in 0:max.p){
  for (q in 0:max.q){
    model.aic[p+1,q+1] = arima(in_sample$Return, order=c(p,0,q), method="ML")$a
ic
  }
}

min(model.aic)

## [1] -3111.207

which(model.aic == min(model.aic), arr.ind = TRUE)

##      row col
## [1,]   3   3

#This means that our recommended p=2 and recommonded q=2 for our ARMA model.


#Now we build our models with this information
AR <- arima(in_sample$Return, order=c(7,0,0)) #We build our AR(p) model with p=
7, since we found that lag to be significant with our PACF.
MA <- arima(in_sample$Return, order=c(0,0,7)) #We build our MA(q) model with q=
7, since we found that lag to be significant with our ACF.
ARMA <- arima(in_sample$Return, order=c(2,0,2)) #Above, we tested various value
s for p and q for our ARMA(p,q) model to find the one with the lowest AIC value
. After checking multiple values, I found that the best model to be p=2 and q=2
.




#Part D
predictAR <- data.frame(out_sample$Date,predict(AR, 5))
predictMA <- data.frame(out_sample$Date,predict(MA, 5))
predictARMA <- data.frame(out_sample$Date,predict(ARMA, 5))
colnames(predictAR) <- c("Date", "Return")
colnames(predictMA) <- c("Date", "Return")
colnames(predictARMA) <- c("Date", "Return")
```

```
#plotting AR prediction over actual
plot(in_sample_last5, main="In and Out of Sample Log Returns with AR Predict",
col="blue", xlim=as.Date(c("2018-04-23","2018-05-04")), ylim=c(min(out_sample$R
eturn, in_sample_last5$Return, predictAR$Return)-.002, max(out_sample$Return, i
n_sample_last5$Return, predictAR$Return)+.005), type="o", pch=16)
lines(out_sample, col="red", type="o", pch=16)
lines(predictAR, col="green", type="o", pch=16)
legend("topright", c("In-Sample", "Out-of-Sample","AR Prediction"), col=c("blue
", "red", "green"), pch=16)
```

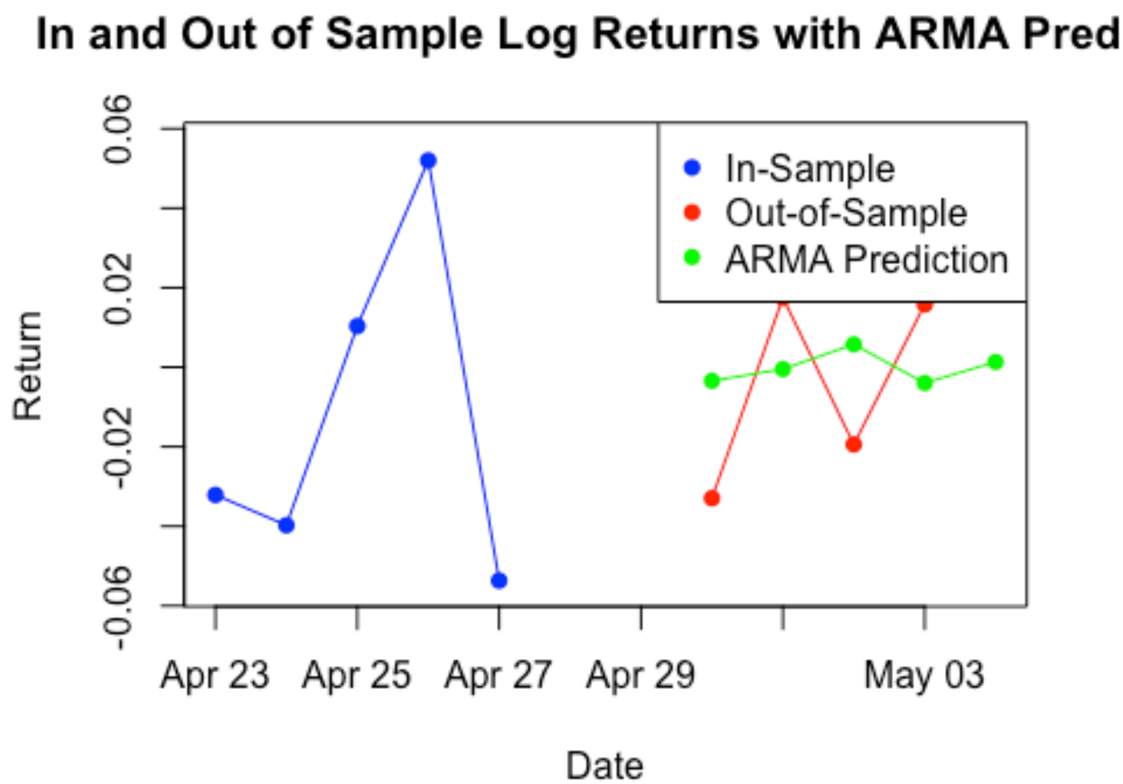


For some reason the legend appears a lot larger after I converted the Rmarkdown to PDF. I couldn't figure out how to prevent this, so I apologize for the slight overlap with my data points.

```
#plotting MA prediction over actual
plot(in_sample_last5, main="In and Out of Sample Log Returns with MA Predict", col="blue", xlim=as.Date(c("2018-04-23","2018-05-04")), ylim=c(min(out_sample$Return, in_sample_last5$Return, predictMA$Return)-.002, max(out_sample$Return, in_sample_last5$Return, predictMA$Return)+.005), type="o", pch=16)
lines(out_sample, col="red", type="o", pch=16)
lines(predictMA, col="green", type="o", pch=16)
legend("topright", c("In-Sample", "Out-of-Sample","MA Prediction"), col=c("blue", "red", "green"), pch=16)
```



```r
#plotting ARMA prediction over actual
plot(in_sample_last5, main="In and Out of Sample Log Returns with ARMA Predict"
, col="blue", xlim=as.Date(c("2018-04-23","2018-05-04")), ylim=c(min(out_sample
$Return, in_sample_last5$Return, predictARMA$Return)-.002, max(out_sample$Retur
n, in_sample_last5$Return, predictARMA$Return)+.005), type="o", pch=16)
lines(out_sample, col="red", type="o", pch=16)
lines(predictARMA, col="green", type="o", pch=16)
legend("topright", c("In-Sample", "Out-of-Sample","ARMA Prediction"), col=c("bl
ue", "red","green"), pch=16)
```



```r
#Part E
sseAR <- sum((out_sample$Return-predictAR$Return)^2)
sseAR

## [1] 0.002374043

sseMA <- sum((out_sample$Return-predictMA$Return)^2)
sseMA

## [1] 0.002222692

sseARMA <- sum((out_sample$Return-predictARMA$Return)^2)
sseARMA

## [1] 0.002580017

#Here we computed the sum of squared errors between each predicted model and th
e out of sample data. AR gave a value of 0.002374043, MA gave a value of 0.0022
22692, and ARMA gave a value of 0.002580017. You will notice that all of these
models gave low values, however, the best choice of a model would be the one wi
th the lowest sum of squared errors, which is the MA model in this case.
```

Interview Problem

```
x <- read.csv("qf-202-final-data-xxx.csv", header=T)
y <- read.csv("qf-202-final-data-yyy.csv", header=T)
z <- read.csv("qf-202-final-data-zzz.csv", header=T)

estimateParameters <- function(data, fileName){
  model <- arima(data, order=c(3,2,6))
  mu <- model$coef[9]
  mu <- c(mu, NA,NA,NA,NA,NA)
  phi <- model$model$phi
  phi <- c(phi,NA,NA,NA)
  theta <- model$model$theta
  sigma <- sqrt(model$sigma2)
  sigma <- c(sigma, NA,NA,NA,NA,NA)
  result <- data.frame(mu, phi, theta, sigma)
  write.csv(result, file=fileName)
}

estimateParameters(x$x, "xxx-result.csv")
estimateParameters(y$x, "yyy-result.csv")
estimateParameters(z$x, "zzz-result.csv")



#The approach I took to this problem is as follows. First, I read each csv file
, saving it with its corresponding letter. I made sure to indicate header=T, si
nce there are headers to our data. The next step I took was to create a functio
n that would estimate the parameters for the data, no matter the data length. T
he function, estimateParameters, takes in 2 parameters, the first being the dat
a and the second being the name of the file you want the result to output to. T
he function creates an ARIMA(3,2,6) model using the data it was called with. Fr
om there, I was able to access each component of the result. First, I found the
mean (mu), which is the last coefficient for the model. Next, I found all the p
hi and theta parameters, referencing them from our created model. Lastly, I too
k the variance of the model and took the square root, so that I can get the sta
ndard deviation. Since there were a different number of parameters for each of
the values we were seeking (mu, phi, theta, sigma), I padded my results with 'N
A' to fill in empty spots, so that I could return a data frame that was nicely
formatted.
```

Below are screenshots from the outputted csv files from the Interview Type Problem

xxx-result.csv

	A	B	C	D	E
1		mu	phi	theta	sigma
2	1	0.43290739	0.76351488	0.24787526	0.42651013
3	2	NA	-0.2389413	0.66746293	NA
4	3	NA	0.34461485	0.33150744	NA
5	4	NA	NA	0.78818715	NA
6	5	NA	NA	0.19731868	NA
7	6	NA	NA	0.43290739	NA

yyy-result.csv

	A	B	C	D	E
1		mu	phi	theta	sigma
2	1	-0.510384	0.24270746	-0.4176316	0.44600512
3	2	NA	0.56179668	0.24321747	NA
4	3	NA	-0.4292908	-0.2056018	NA
5	4	NA	NA	0.11126603	NA
6	5	NA	NA	0.26901919	NA
7	6	NA	NA	-0.510384	NA

zzz-result.csv

	A	B	C	D	E
1		mu	phi	theta	sigma
2	1	0.78405266	-0.4310662	0.20879335	0.99126864
3	2	NA	0.64169784	-0.5550526	NA
4	3	NA	0.22216389	0.43734038	NA
5	4	NA	NA	-0.3296473	NA
6	5	NA	NA	0.11334138	NA
7	6	NA	NA	0.78405266	NA