ISYE 6402 Homework 6 Solutions

Background

Individuals stock prices tend to exhibit high amounts of non-constant variance, and thus ARIMA models build upon that data would likely exhibit non-constant variance in residuals. In this problem we are going to analyze the Starbucks stock price data from 2012 through end of 2021. We will use the ARIMA-GARCH to model daily and weekly stock price (adjusted close price at the end of a day for daily data or at the end of the week for weekly data), with a focus on the behavior of its volatility as well as forecasting both the price and the volatility.

##Data import and cleaning

```r
## Libraries used within this homework are uploaded here
library(zoo,warn.conflicts=FALSE)
library(lubridate,warn.conflicts=FALSE)
library(mgcv,warn.conflicts=FALSE)
library(rugarch,warn.conflicts=FALSE)
```

```r
#importing the data
dailydata <- read.csv("/Users/rawil/OneDrive/Documents/GT Analytics/Teaching Assistant/ISYE6402_Spring2
weeklydata <- read.csv("/Users/rawil/OneDrive/Documents/GT Analytics/Teaching Assistant/ISYE6402_Spring

#cleaning the data

#dates to date format
weeklydata$Date<-as.Date(weeklydata$Date,format='%m/%d/%Y')
dailydata$Date<-as.Date(dailydata$Date,format='%m/%d/%Y')

#prices to timeseries format
SBUXWeekly <- ts(weeklydata$Adj.Close,start=c(2011,1,1),freq=52)
SBUXDaily <- ts(dailydata$Adj.Close,start=c(2011,1,1),freq=252)
```

#Question 1: Exploratory Data Analysis (20 points)

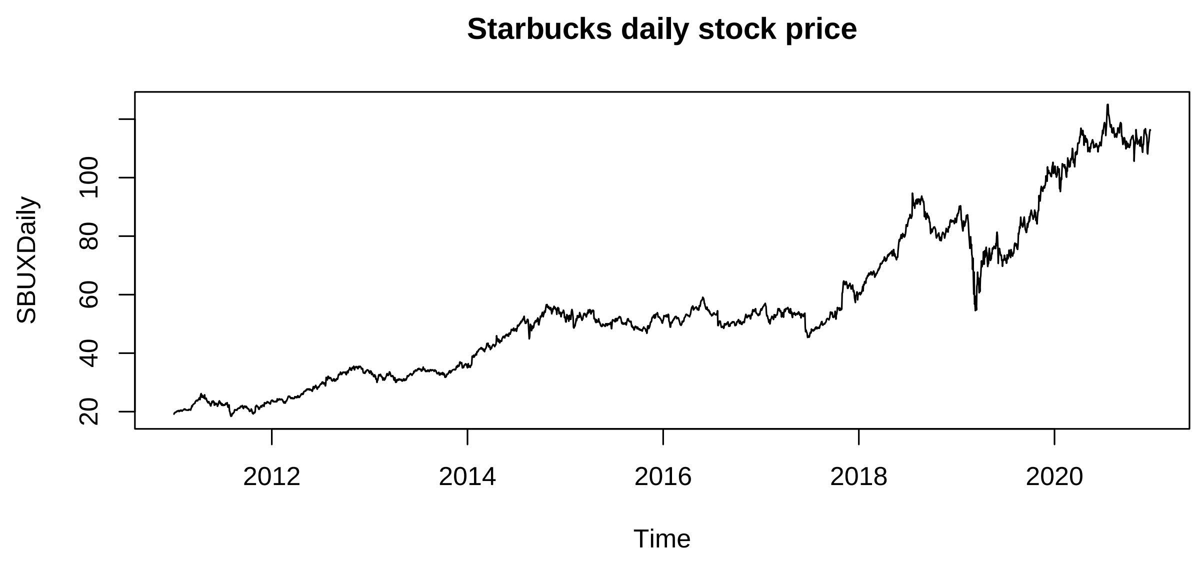
1a. Based on your intuition, when would you use daily vs weekly stock price data?

Response: Intuition

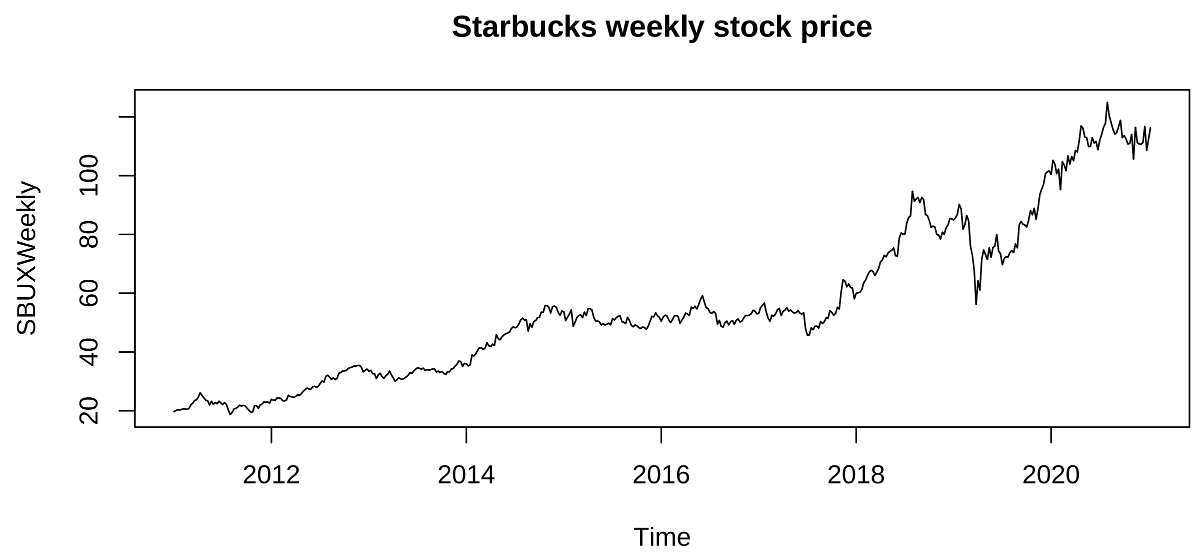
It would be a better idea to use daily data as opposed to the weekly data when the price movements seem to be displaying volatility at shorter time intervals. However, in the case of Starbucks, the overall change of stock price over longer intervals of time would seem large enough so as to not require analysis over shorter intervals of time.

1b. Plot the time series plots comparing daily vs weekly data. How do the daily vs weekly time series data compare?

```r
plot(SBUXDaily,main="Starbucks daily stock price",type="l")
```

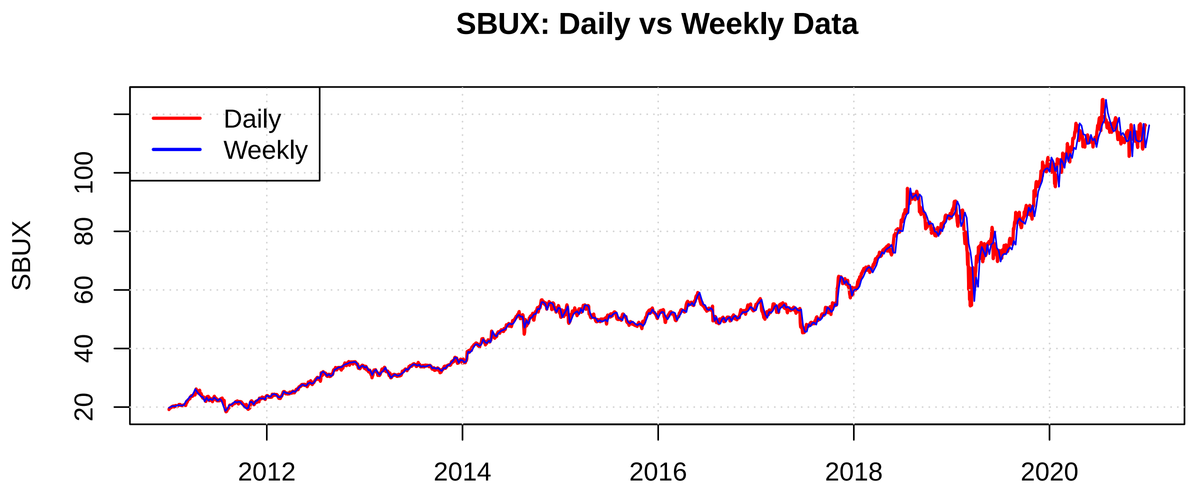


```
plot(SBUXWeekly,main="Starbucks weekly stock price",type="l")
```



```
## overlap graphs
```

```
### THIS IS AN ALTERNATIVE GRAPH
plot(SBUXDaily, xlab = "", ylab = "SBUX", col = "red", lwd = 2,type="l",
     main = "SBUX: Daily vs Weekly Data")
grid()
lines(SBUXWeekly, col = "blue", lwd = 1, type='l')
legend("topleft", c("Daily", "Weekly"), col = c("red", "blue"), lwd = 2)
```



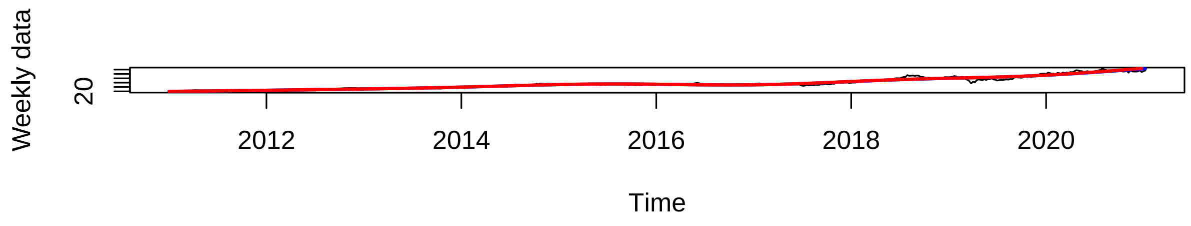
Response: Weekly vs Montly Time Series data comparison

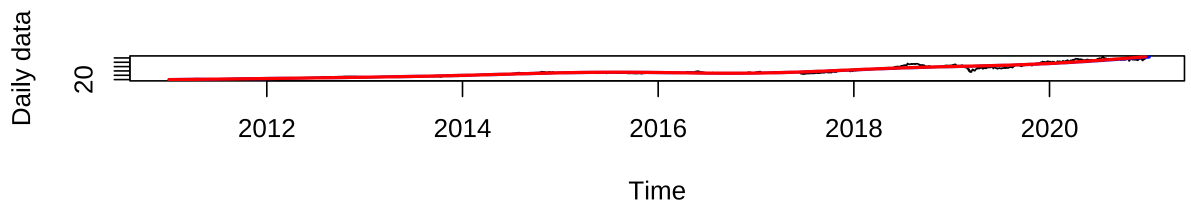
As expected, the Starbucks stock seems to be more volatile in recent years and this volatility in price is significant enough for us to consider the weekly data as sufficient for analyzing the movements, trends, stationarity and residual analysis. The movements in mean and variance seem to be quite significant over the whole time period; the weekly data, which is only slightly smoothed over from the daily data, should suffice for our analysis.

1c. Fit a non-parametric trend using splines regression to both the daily and weekly time series data. Overlay the fitted trends. How do the trends compare?

Analyzing weekly and daily data with trend fitting

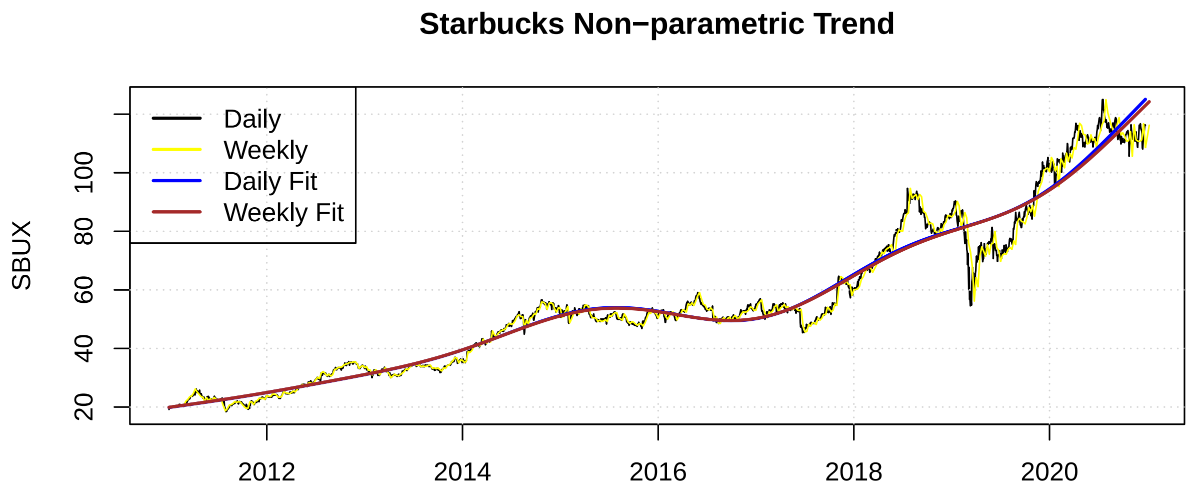
```
time.ptsw = c(1:length(SBUXWeekly))
time.ptsw = c(time.ptsw - min(time.ptsw))/max(time.ptsw)

time.ptsd = c(1:length(SBUXDaily))
time.ptsd = c(time.ptsd - min(time.ptsd))/max(time.ptsd)

egam.fit.weekly = gam(SBUXWeekly~s(time.ptsw))
eu.fit.gam.weekly = ts(fitted(egam.fit.weekly),c(2011,1,1),frequency=52)
egam.fit.daily = gam(SBUXDaily~s(time.ptsd))
eu.fit.gam.daily = ts(fitted(egam.fit.daily),start=c(2011,1,1),frequency=252)

## Is there a trend? Which data does it fit better?
par(mfrow=c(2,1))
ts.plot(SBUXWeekly,ylab="Weekly data")
lines(eu.fit.gam.weekly,lwd=2,col="blue")
lines(eu.fit.gam.daily,lwd=2,col="red")

ts.plot(SBUXDaily,ylab="Daily data")
lines(eu.fit.gam.weekly,lwd=2,col="blue")
lines(eu.fit.gam.daily,lwd=2,col="red")
```





```
# overlap graphs
```

```
par(mfrow=c(1,1))
### THIS IS AN ALTERNATIVE GRAPH ###
ts.plot(SBUXDaily, xlab = "", ylab = "SBUX",
        main = "Starbucks Non-parametric Trend")
grid()
lines(SBUXWeekly, col = "yellow")
lines(eu.fit.gam.daily, lwd = 2, col = "blue")
lines(eu.fit.gam.weekly, lwd = 2, col = "brown")
legend("topleft", legend = c("Daily", "Weekly", "Daily Fit", "Weekly Fit"),
       col = c("black", "yellow", "blue", "brown"), lwd = 2)
```



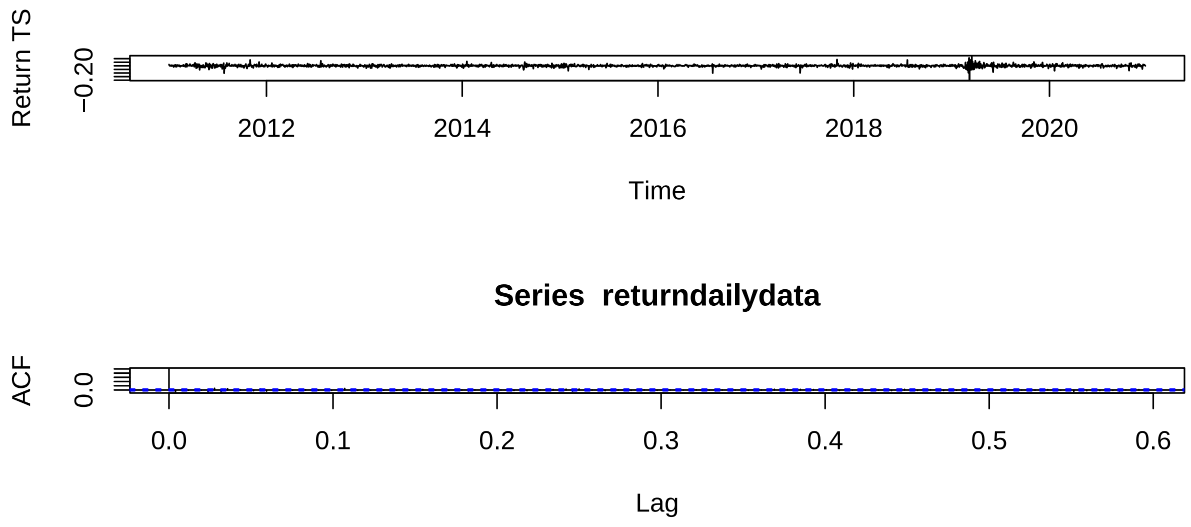
Response: Weekly vs Montly Time Series data trend fit

The weekly data seems to encapsulate the trend fit well, which gives us reason to not go by daily trend fitting.

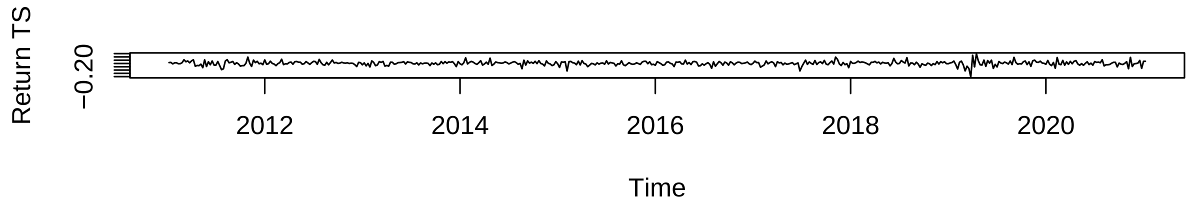
1d. Consider the return stock price computed as provided in the canvas homework assignment. Apply this formula to compute the return price based on the daily and weekly time series data. Plot the return time series and their corresponding ACF plots. How do the return time series compare in terms of stationarity and serial dependence?

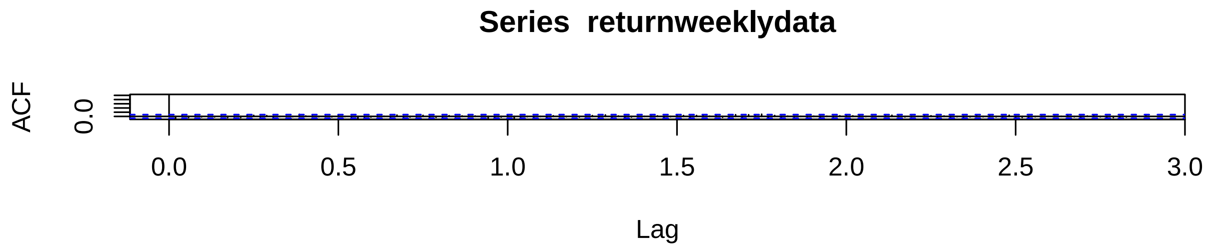
Analyzing weekly and daily return data and comparing with original data

```
par(mfrow=c(2,1))
datadailydiff=diff(SBUXDaily)
returndailydata=datadailydiff/SBUXDaily[-1]
plot(returndailydata,ylab= "Return TS")
acf(returndailydata,lag.max=6*25)
```



```
dataweeklydiff=diff(SBUXWeekly)
returnweeklydata=dataweeklydiff/SBUXWeekly[-1]
plot(returnweeklydata, ylab= "Return TS")
acf(returnweeklydata,lag.max=6*25)
```





Response: Return series vs price series analysis

The residual acf plots for the return time series indicate no stationarity; in fact, the acf plots look just like those of white noise, hence potentially indicating no serial dependence.

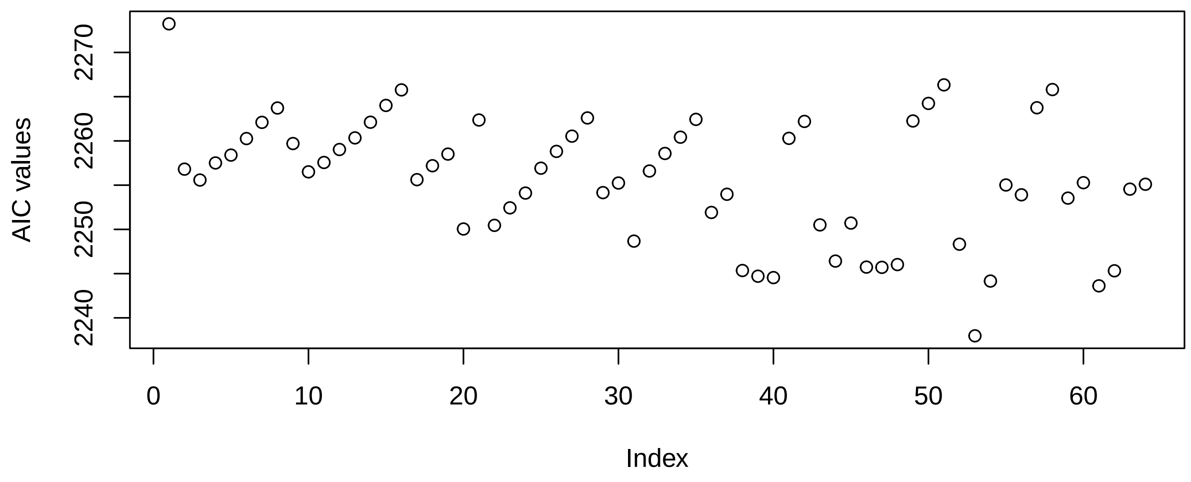
#Question 2: ARIMA(p,d,q) for Stock Price (20 Points)

2a. Divide the data into training and testing data set, where the training data exclude the last week of data (December 27th - December 30th) with the testing data including the last week of data. Apply the iterative model to fit an ARIMA(p,d,q) model with max AR and MA orders of 8 and difference orders 1 and 2 separately to the training datasets of the daily and weekly data. Display the summary of the final model fit.

Analyzing weekly and daily data with ARIMA model fitting

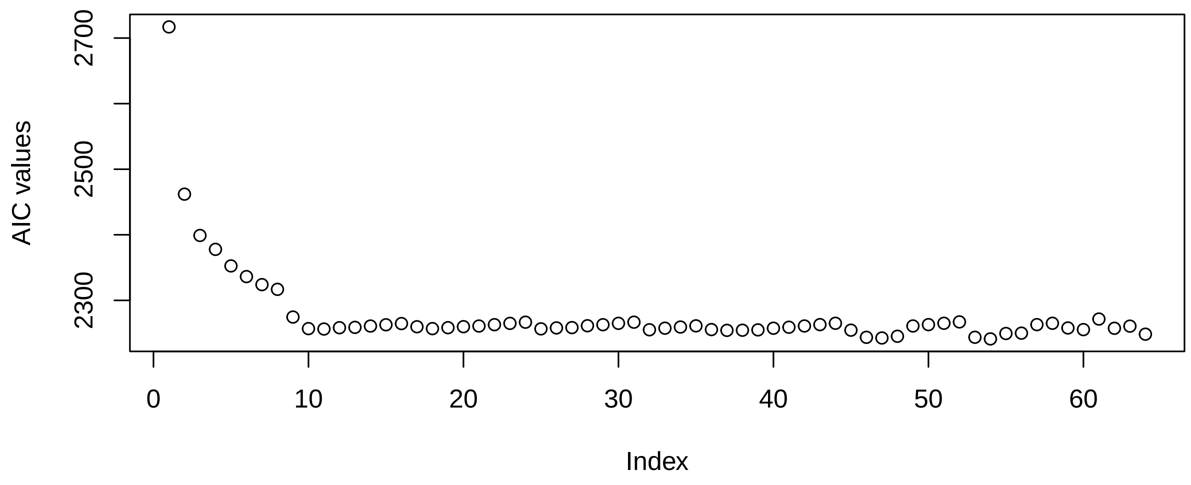
```
#dividing the data into training and testing

SBUXWeekly.train=SBUXWeekly[1:(length(SBUXWeekly)-1)]
SBUXWeekly.test=SBUXWeekly[(length(SBUXWeekly)):length(SBUXWeekly)]

SBUXDaily.train=SBUXDaily[1:(length(SBUXDaily)-4)]
SBUXDaily.test=SBUXDaily[(length(SBUXDaily)-3):length(SBUXDaily)]


############# Starbucks Weekly Data ##########################
#differencing order = 1
n = length(SBUXWeekly) ####### w_n, deleted w_
w_n_fit=length(SBUXWeekly.train)
w_n_forward=n-w_n_fit
w_n=length(SBUXWeekly.train)
norder = 8
p = c(1:norder)-1; q = c(1:norder)-1
aic = matrix(0,norder,norder)
for(i in 1:norder){
  for(j in 1:norder){
    modij = stats::arima(SBUXWeekly.train,order = c(p[i],1,q[j]), method='ML')
    aic[i,j] = modij$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)
```

```r
  }
}

# Extract the "best" one according to AIC
aicv <- as.vector(aic)
plot(aicv,ylab="AIC values")
```



```r
indexp <- rep(c(1:norder),norder)
indexq <- rep(c(1:norder),each=norder)
indexaic <- which(aicv == min(aicv))
porder_1 <- indexp[indexaic]-1
qorder_1 <- indexq[indexaic]-1


# Fit the "best" arima model
w_final_model <- arima(SBUXWeekly.train, order = c(porder_1,1,qorder_1), method = "ML")



#differencing order as 2
n = length(SBUXWeekly.train)
norder = 8
p = c(1:norder)-1; q = c(1:norder)-1
aic = matrix(0,norder,norder)
for(i in 1:norder){
  for(j in 1:norder){
    modij = stats::arima(SBUXWeekly.train,order = c(p[i],2,q[j]), method='ML')
    aic[i,j] = modij$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)

  }
}
# Extract the "best" one according to AIC
aicv <- as.vector(aic)
```

```
plot(aicv,ylab="AIC values")
```



```
indexp <- rep(c(1:norder),norder)
indexq <- rep(c(1:norder),each=norder)
indexaic <- which(aicv == min(aicv))
porder_2 <- indexp[indexaic]-1
qorder_2 <- indexq[indexaic]-1
# Fit the "best" arima model
w_final_model2 <- arima(SBUXWeekly.train, order = c(porder_2,1,qorder_2), method = "ML")

print("Orders and AIC Values: Weekly values")
```

```
## [1] "Orders and AIC Values: Weekly values"
```

```
#Diff=1
c(porder_1,qorder_1)
```

```
## [1] 4 6
```

```
#diff=2
c(porder_2,qorder_2)
```

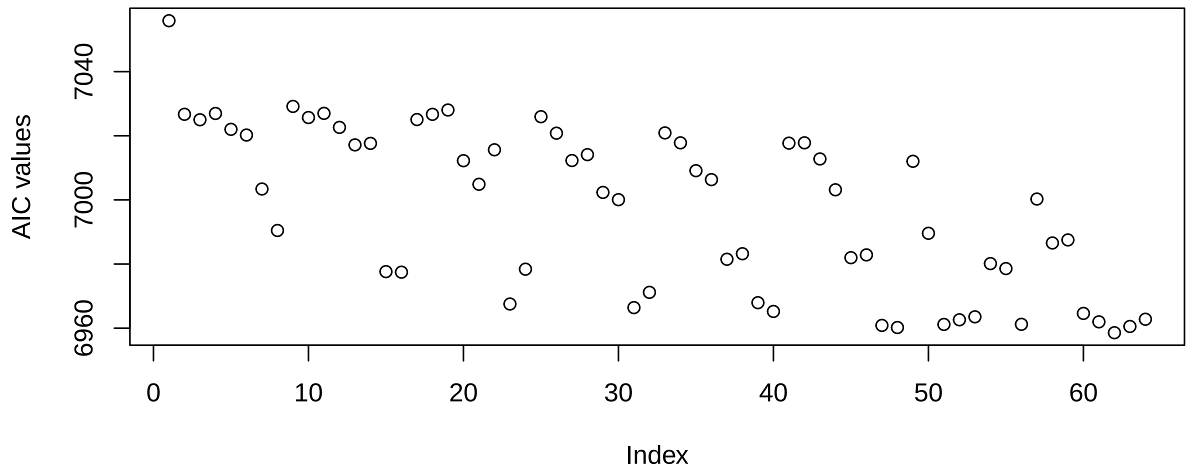
```
## [1] 5 6
```

```
c(w_final_model$aic, w_final_model2$aic)
```

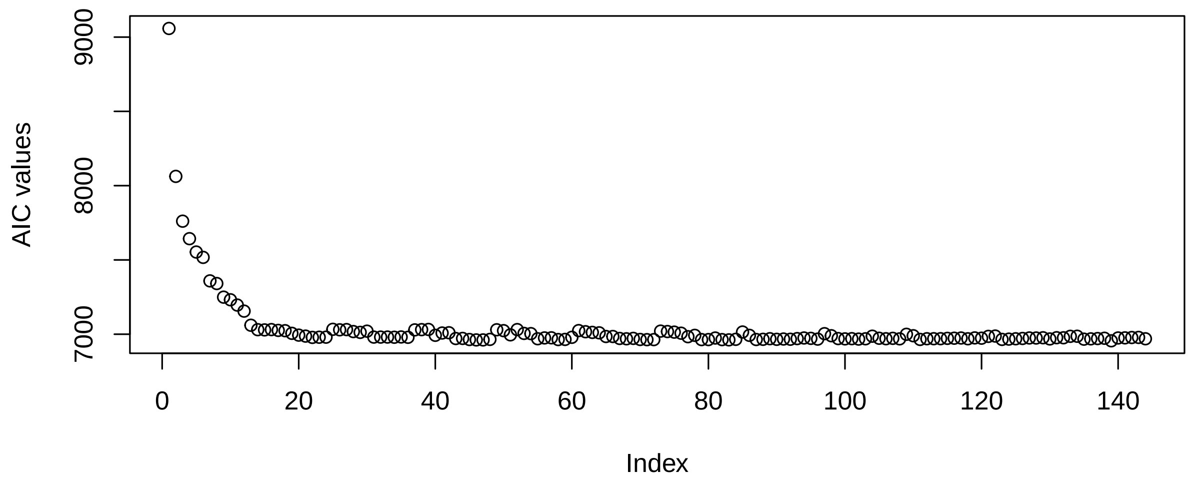
```
## [1] 2237.453 2243.545
```

```
############# SBUX Daily Data ##########################
#differencing order = 1
n = length(SBUXDaily)
n_fit=length(SBUXDaily.train)
n_forward=n-n_fit
n=length(SBUXDaily.train)
norder = 8
p = c(1:norder)-1; q = c(1:norder)-1
aic = matrix(0,norder,norder)
```

```r
for(i in 1:norder){
  for(j in 1:norder){
    modij = stats::arima(SBUXDaily.train,order = c(p[i],1,q[j]), method='ML')
    aic[i,j] = modij$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)

  }
}

# Extract the "best" one according to AIC
aicv <- as.vector(aic)
plot(aicv,ylab="AIC values")
```



```r
indexp <- rep(c(1:norder),norder)
indexq <- rep(c(1:norder),each=norder)
indexaic <- which(aicv == min(aicv))
porder_3 <- indexp[indexaic]-1
qorder_3 <- indexq[indexaic]-1


# Fit the "best" arima model
d_final_model <- arima(SBUXDaily.train, order = c(porder_3,1,qorder_3), method = "ML")


#differencing order as 2
n = length(SBUXDaily.train)
norder = 12
p = c(1:norder)-1; q = c(1:norder)-1
aic = matrix(0,norder,norder)
for(i in 1:norder){
  for(j in 1:norder){
    modij = stats::arima(SBUXDaily.train,order = c(p[i],2,q[j]), method='ML')
    aic[i,j] = modij$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)
  }
}
# Extract the "best" one according to AIC
aicv <- as.vector(aic)
plot(aicv,ylab="AIC values")
```



```
indexp <- rep(c(1:norder),norder)
indexq <- rep(c(1:norder),each=norder)
indexaic <- which(aicv == min(aicv))
porder_4 <- indexp[indexaic]-1
qorder_4 <- indexq[indexaic]-1
# Fit the "best" arima model
d_final_model2 <- arima(SBUXDaily.train, order = c(porder_4,2,qorder_4), method = "ML")

print("Orders and AIC Values: Daily values")
```

```
## [1] "Orders and AIC Values: Daily values"
```

```
#Diff=1
c(porder_3,qorder_3)
```

```
## [1] 5 7
```

```
#diff=2
c(porder_4,qorder_4)
```

```
## [1]  6 11
```

```
c(d_final_model$aic, d_final_model2$aic)
```

```
## [1] 6958.434 6955.427
```

Response: Analysis of the ARIMA Fit for the Weekly and Monthly Data

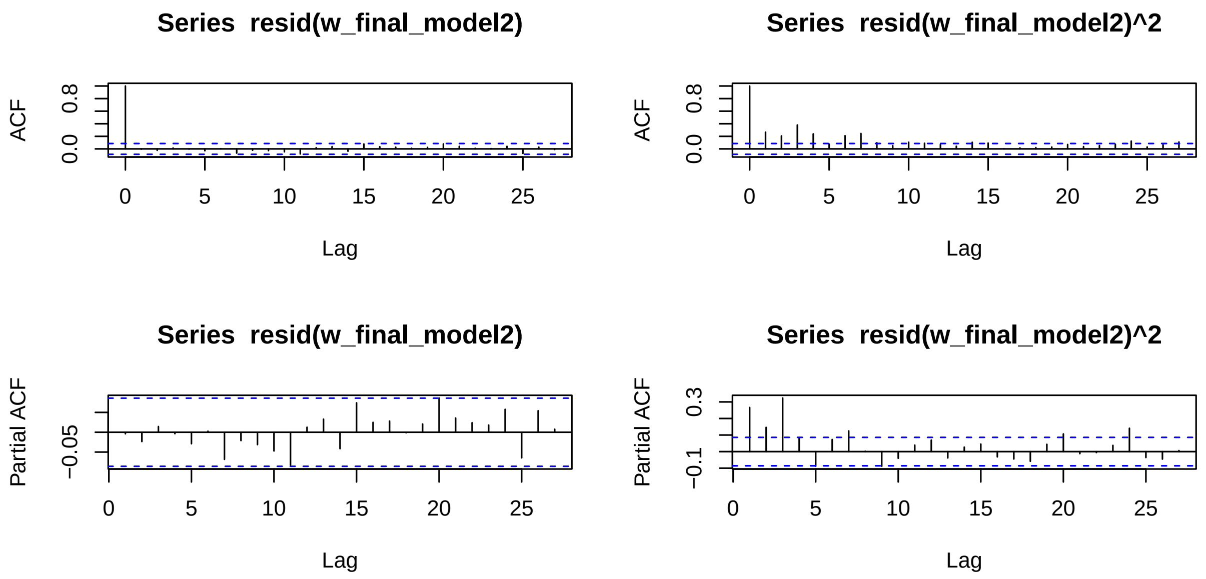
The selected models for the weekly data are as follows:

- Daily Model Selection: ARIMA(5,1,7) with AICc = 6958.43 vs ARIMA(6,2,11) with AICc = 6955.43, where the more complex model ARIMA(6,2,11) model has a smaller AICc value.

- Weekly Model Selection: ARIMA(4,1,6) with AICc = 2237.45 vs ARIMA(5,2,6) with AICc = 2243.55, with the ARIMA(4,1,6) having a lower AICc value hence selected.

2b. Evaluate the model residuals and squared residuals using the ACF and PACF plots as well as hypothesis testing for serial correlation. What would you conclude based on this analysis?

```
#plotting the acfs and pacfs of residuals and squared residuals

par(mfrow=c(2,2))
acf(resid(w_final_model2))
acf(resid(w_final_model2)^2)

pacf(resid(w_final_model2))
pacf(resid(w_final_model2)^2)
```



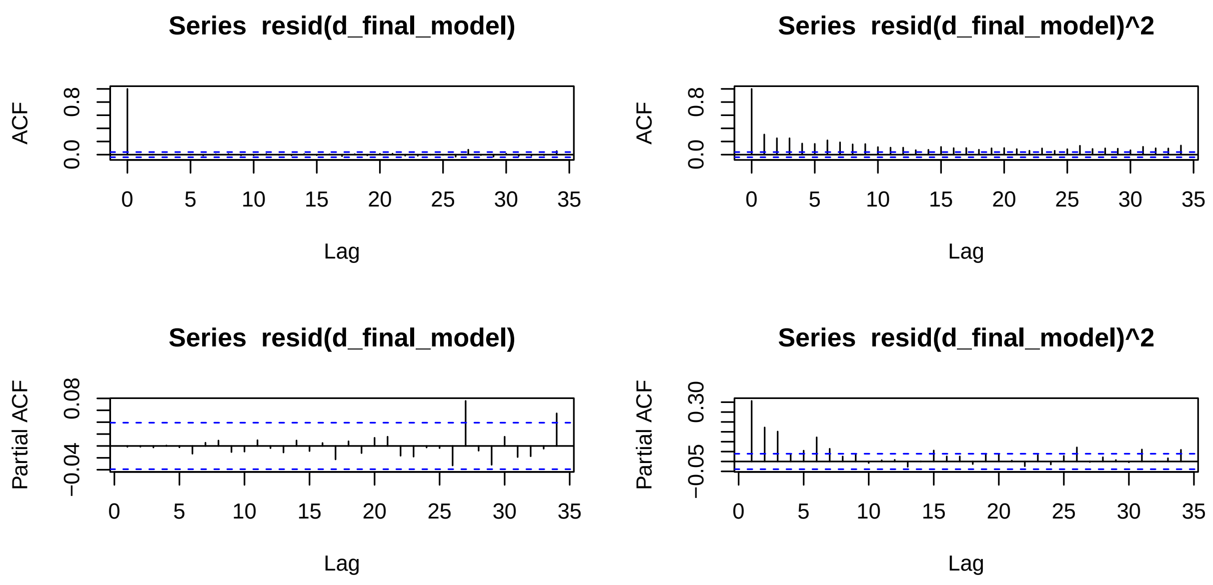
```
Box.test(w_final_model2$resid, lag = (porder_2+qorder_2+1), type = "Ljung-Box", fitdf = (porder_2+qorde
```

```
##
##  Box-Ljung test
##
## data:  w_final_model2$resid
## X-squared = 8.7108, df = 1, p-value = 0.003163
```

```
Box.test(w_final_model2$resid^2, lag = (porder_2+qorder_2+1), type = "Ljung-Box", fitdf = (porder_2+qor
```

```
##
##  Box-Ljung test
##
## data:  w_final_model2$resid^2
## X-squared = 243.59, df = 1, p-value < 2.2e-16
```

```
par(mfrow=c(2,2))
acf(resid(d_final_model))
acf(resid(d_final_model)^2)

pacf(resid(d_final_model))
pacf(resid(d_final_model)^2)
```



```
#test for serial correlation
Box.test(d_final_model$resid, lag = (porder_4+qorder_4+1), type = "Ljung-Box", fitdf = (porder_4+qorder_
```

```
##
##  Box-Ljung test
##
## data:  d_final_model$resid
## X-squared = 3.8235, df = 1, p-value = 0.05054
```

```
Box.test(d_final_model$resid^2, lag = (porder_4+qorder_4+1), type = "Ljung-Box", fitdf = (porder_4+qord
```

```
##
##  Box-Ljung test
##
## data:  d_final_model$resid^2
## X-squared = 1243.6, df = 1, p-value < 2.2e-16
```

Response:ARIMA residual analysis for the Weekly and Monthly Data

The fit for the weekly residuals data seems to be well fitting based on the residual analysis. The null hypothesis is that the residual process consists of uncorrelated variables, which is not rejected since the the p-values are large. There appears to be minor arch effect present based on a p-value that is nearly 0.05 (0.051).

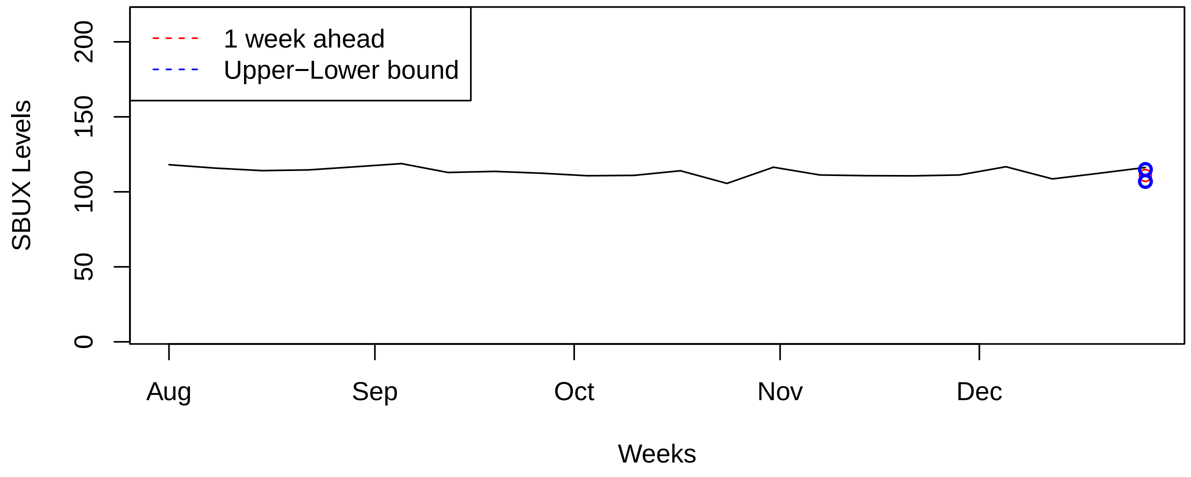
For the squared residuals appear to be better fitting. The p-value is minor and the ACF/PACF follow the trend we would expect to see. Hence it is plausible that the residuals to follow the distribution assumptions hence the models fit the data.

2c. Apply the model identified in (2a) and forecast the last week of data. Plot the predicted data to compare the predicted values to the actual observed ones. Include 95% confidence intervals for the forecasts in the corresponding plots.

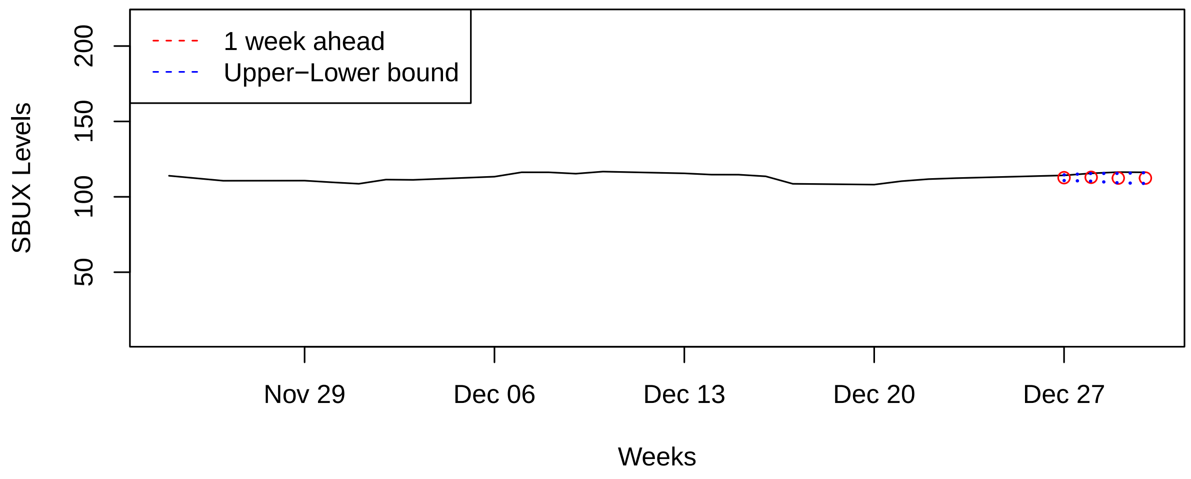
Forecasting and plotting weekly and daily data using the ARIMA model

```
#predicting last week data using weekly data model
outpred = predict(w_final_model2,n.ahead=w_n_forward)
# 95% confidence interval
```

```
ubound = outpred$pred+1.96*outpred$se
lbound = outpred$pred-1.96*outpred$se
ymin = min(lbound)-100
ymax = max(ubound)+100
dates.diff = weeklydata$Date
#nfit = n-n_forward
par(mfrow=c(1,1))
n = length(SBUXWeekly)
plot((dates.diff)[(n-w_n_forward-20):n],SBUXWeekly[(n-w_n_forward-20):n],type="l", ylim=c(ymin,ymax), xl
points((dates.diff)[(w_n_fit+1):n],outpred$pred,col="red")
points((dates.diff)[(w_n_fit+1):n],ubound,lty=3,lwd= 2, col="blue")
points((dates.diff)[(w_n_fit+1):n],lbound,lty=3,lwd= 2, col="blue")
legend('topleft', legend=c("1 week ahead ","Upper-Lower bound"),lty = 2, col=c("red","blue"))
```



```
#predicting last week data using daily data model
outpred = predict(d_final_model,n.ahead=n_forward)
# 95% confidence interval
ubound = outpred$pred+1.96*outpred$se
lbound = outpred$pred-1.96*outpred$se
ymin = min(lbound)-100
ymax = max(ubound)+100
dates.diff = dailydata$Date
#nfit = n-n_forward
par(mfrow=c(1,1))
n = length(SBUXDaily)
plot((dates.diff)[(n-n_forward-20):n],SBUXDaily[(n-n_forward-20):n],type="l", ylim=c(ymin,ymax), xlab="W
points((dates.diff)[(n_fit+1):n],outpred$pred,col="red")
lines((dates.diff)[(n_fit+1):n],ubound,lty=3,lwd= 2, col="blue")
lines((dates.diff)[(n_fit+1):n],lbound,lty=3,lwd= 2, col="blue")
legend('topleft', legend=c("1 week ahead ","Upper-Lower bound"),lty = 2, col=c("red","blue"))
```



Response: Predictions

The model for the daily data seems to fit the data better hence resulting in generally better predictions. The variability of the daily data helps the model predict values with a lower variance than predicting a single weekly value with a large probability of being away from the actual value. This is also due to the volatile nature of the stock price data.

2d. Calculate Mean Absolute Percentage Error (MAPE) and Precision Measure (PM). How many observations are within the prediction bands? Compare the accuracy of the predictions for the daily and weekly time series using these two measures.

Calculating accuracy measures of the ARIMA Fit for the Weekly and Monthly Data

```
#Computing Accuracy measures

#Weekly Data
outpred = predict(w_final_model2,n.ahead=w_n_forward)
ubound = outpred$pred+1.96*outpred$se  #confidence interval
lbound = outpred$pred-1.96*outpred$se
n = length(SBUXWeekly)
consump_true = as.vector(SBUXWeekly[(w_n_fit+1):n])
consump_pred = outpred$pred
#MAPE
mean(abs(consump_pred-consump_true)/consump_true)
```

```
## [1] 0.04586448
```

```
#PM
sum((consump_pred-consump_true)^2)/sum((consump_true-mean(consump_true))^2)
```

```
## [1] Inf
```

```
#Daily Data
outpred = predict(d_final_model,n.ahead=n_forward)
ubound = outpred$pred+1.96*outpred$se  #confidence interval
lbound = outpred$pred-1.96*outpred$se
n = length(SBUXDaily)
consump_true = as.vector(SBUXDaily[(n_fit+1):n])
```

```
consump_pred = outpred$pred
#MAPE
mean(abs(consump_pred-consump_true)/consump_true)
```

```
## [1] 0.02577881
```

```
#PM
sum((consump_pred-consump_true)^2)/sum((consump_true-mean(consump_true))^2)
```

```
## [1] 13.53883
```

Response: Prediction Comparison

For the weekly data, we cannot compute the PM as there is only one value. But we see that by the MAPE, the daily data fitting seems to be a better idea.

#Question 3: ARMA(p,q)-GARCH(m,n) for Return Stock Price (20 Points)

3a. Divide the data into training and testing data set, where the training data exclude the last week of data (December 27th - December 30th) with the testing data including the last week of data. Apply the iterative model to fit an ARMA(p,q)-GARCH(m,n) model by selecting the orders for p & q up to 3 and orders for m & n up to 2. Display the summary of the final model fit. Write up the equation of the estimated model.

Analyzing weekly and daily data with ARMA GARCH model fitting

```
#dividing the data into training and testing

SBUXWeekly.train=returnweeklydata[1:(length(returnweeklydata)-1)]
SBUXWeekly.test=returnweeklydata[(length(returnweeklydata)):length(returnweeklydata)]

SBUXDaily.train=returndailydata[1:(length(returndailydata)-4)]
SBUXDaily.test=returndailydata[(length(returndailydata)-3):length(returndailydata)]



#applying the ARIMA-GARCH model to the weekly data

#Fit ARIMA on differenced series
    w.final.aic=Inf
    w.final.order=c(0,0,0)
    for (p in 0:3) for (d in 0:1) for (q in 0:3)
    {
        w.current.aic=AIC(arima(returnweeklydata,order=c(p, d, q)))
        if(w.current.aic<w.final.aic)
        {
            w.final.aic=w.current.aic
            w.final.order=c(p,d,q)
            w.final.arima=arima(returnweeklydata, order=w.final.order)
        }
    }

    # What is the selected order?
    w.final.order
```

```
## [1] 1 0 1
```

```
#Select model with smallest BIC
            final.bic = Inf
            final.order = c(0,0)
            for (p in 0:2) for (q in 0:2)
            {
                spec = ugarchspec(variance.model=list(garchOrder=c(p,q)),
                mean.model=list(armaOrder=c(w.final.order[1], w.final.order[3]), include.mean=T),
                distribution.model="std")
                fit = ugarchfit(spec, SBUXWeekly.train, solver = 'nloptr')
                current.bic = infocriteria(fit)[2]
                if (current.bic < final.bic)
                {
                    final.bic = current.bic
                    final.order = c(p, q)
                }
            }
        final.order
```

```
## [1] 2 0
```

```
    #Refine the ARMA order
        final.bic = Inf
        final.order.arma = c(0,0)
        for (p in 0:3) for (q in 0:3) ####### CHANGE 1 TO 0 #####
        {
            spec = ugarchspec(variance.model=list(garchOrder=c(final.order[1],final.order[2])),
            mean.model=list(armaOrder=c(p, q), include.mean=T),
            distribution.model="std")
            fit = ugarchfit(spec, SBUXWeekly.train, solver = 'hybrid')
            current.bic = infocriteria(fit)[2]
            if (current.bic < final.bic)
            {
                final.bic = current.bic
                final.order.arma = c(p, q)
            }
        }
        final.order.arma
```

```
## [1] 0 0
```

```
        #[1] 0 0

    #Refine the GARCH order
        final.bic = Inf
        final.order.garch = c(0,0)
        for (p in 0:2) for (q in 0:2)
        {
            spec = ugarchspec(variance.model=list(garchOrder=c(p,q)),
            mean.model=list(armaOrder=c(final.order.arma[1], final.order.arma[2]),
                include.mean=T), distribution.model="std")
                fit = ugarchfit(spec, SBUXWeekly.train, solver = 'hybrid')
                current.bic = infocriteria(fit)[2]
                if (current.bic < final.bic)
            {
                final.bic = current.bic
                final.order.garch = c(p, q)
                }
              }
        final.order.garch
```

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

```
    final.order.garch.weekly=final.order.garch


    #Goodness of Fit
        w.spec.1 = ugarchspec(variance.model=list(garchOrder=c(1,1)),
            mean.model=list(armaOrder=c(2,4),
            include.mean=T), distribution.model="std")
        w.final.model.1 = ugarchfit(w.spec.1, SBUXWeekly.train, solver = 'hybrid')
w.final.model.1
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(2,0,4)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.004242    0.001063  3.98870 0.000066
## ar1     0.172420    0.470640  0.36635 0.714102
## ar2    -0.508797    0.731865 -0.69521 0.486926
## ma1    -0.266630    0.470816 -0.56632 0.571179
## ma2     0.521718    0.748237  0.69726 0.485638
## ma3    -0.092427    0.062402 -1.48115 0.138566
## ma4     0.049253    0.051853  0.94986 0.342183
## omega   0.000148    0.000086  1.72348 0.084801
## alpha1  0.117440    0.050432  2.32867 0.019877
## beta1   0.756865    0.103666  7.30099 0.000000
## shape   4.239876    0.846011  5.01161 0.000001
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.004242    0.001173  3.61700 0.000298
## ar1     0.172420    0.626239  0.27533 0.783065
## ar2    -0.508797    1.386408 -0.36699 0.713627
## ma1    -0.266630    0.633310 -0.42101 0.673747
## ma2     0.521718    1.426543  0.36572 0.714573
## ma3    -0.092427    0.093130 -0.99246 0.320974
## ma4     0.049253    0.057385  0.85829 0.390730
## omega   0.000148    0.000068  2.16226 0.030598
## alpha1  0.117440    0.059498  1.97386 0.048398
## beta1   0.756865    0.081481  9.28888 0.000000
## shape   4.239876    0.913269  4.64253 0.000003
##
## LogLikelihood : 1093.098
```

```
##
## Information Criteria
## ------------------------------------
##
## Akaike       -4.1619
## Bayes        -4.0719
## Shibata      -4.1628
## Hannan-Quinn -4.1267
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                          statistic p-value
## Lag[1]                      0.2922  0.5888
## Lag[2*(p+q)+(p+q)-1][17]    3.5581  1.0000
## Lag[4*(p+q)+(p+q)-1][29]    9.2066  0.9886
## d.o.f=6
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                    0.06061  0.8055
## Lag[2*(p+q)+(p+q)-1][5]   1.50754  0.7380
## Lag[4*(p+q)+(p+q)-1][9]   4.39269  0.5236
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]     1.295 0.500 2.000  0.2551
## ARCH Lag[5]     2.473 1.440 1.667  0.3758
## ARCH Lag[7]     4.854 2.315 1.543  0.2397
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  1.7128
## Individual Statistics:
## mu     0.06231
## ar1    0.10403
## ar2    0.44736
## ma1    0.08503
## ma2    0.40604
## ma3    0.03544
## ma4    0.54369
## omega  0.22008
## alpha1 0.28542
## beta1  0.21667
## shape  0.19053
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          2.49 2.75 3.27
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                     t-value   prob sig
## Sign Bias             0.156 0.8761
## Negative Sign Bias    1.068 0.2859
## Positive Sign Bias    1.199 0.2309
## Joint Effect          6.187 0.1029
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     16.85       0.6003
## 2    30     33.62       0.2536
## 3    40     36.31       0.5933
## 4    50     52.69       0.3332
##
##
## Elapsed time : 0.8025389
```

```
        w.spec.2 = ugarchspec(variance.model=list(garchOrder=c(1,1)),
            mean.model=list(armaOrder=c(0,0),
            include.mean=T), distribution.model="std")
        w.final.model.2 = ugarchfit(w.spec.2, SBUXWeekly.train, solver = 'hybrid')
w.final.model.2
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model   : sGARCH(1,1)
## Mean Model    : ARFIMA(0,0,0)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.004169    0.001161   3.5921 0.000328
## omega   0.000154    0.000084   1.8375 0.066136
## alpha1  0.129643    0.051378   2.5233 0.011625
## beta1   0.738807    0.103736   7.1220 0.000000
## shape   4.417401    0.873622   5.0564 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.004169    0.001108   3.7626 0.000168
## omega   0.000154    0.000063   2.4692 0.013542
## alpha1  0.129643    0.056644   2.2887 0.022096
## beta1   0.738807    0.074784   9.8792 0.000000
## shape   4.417401    0.786685   5.6152 0.000000
##
## LogLikelihood : 1089.63
##
```

```
## Information Criteria
## ------------------------------------
##
## Akaike       -4.1717
## Bayes        -4.1308
## Shibata      -4.1718
## Hannan-Quinn -4.1556
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      1.841  0.1749
## Lag[2*(p+q)+(p+q)-1][2]     1.883  0.2829
## Lag[4*(p+q)+(p+q)-1][5]     3.145  0.3812
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                  0.0001245  0.9911
## Lag[2*(p+q)+(p+q)-1][5] 1.3502231  0.7766
## Lag[4*(p+q)+(p+q)-1][9] 3.5995692  0.6573
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.8793 0.500 2.000  0.3484
## ARCH Lag[5]    2.0081 1.440 1.667  0.4695
## ARCH Lag[7]    3.8286 2.315 1.543  0.3719
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  0.5528
## Individual Statistics:
## mu     0.05703
## omega  0.19066
## alpha1 0.26089
## beta1  0.19450
## shape  0.15555
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.28 1.47 1.88
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias           0.4454 0.6562
## Negative Sign Bias  0.6305 0.5286
## Positive Sign Bias  1.1165 0.2647
## Joint Effect        5.3699 0.1466
##
```

```
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     17.77       0.5379
## 2    30     36.04       0.1724
## 3    40     31.69       0.7907
## 4    50     40.58       0.7986
##
##
## Elapsed time : 0.254853
```

```
        #Compare Information Criteria
        infocriteria(w.final.model.1)
```

```
##
## Akaike       -4.161916
## Bayes        -4.071932
## Shibata      -4.162787
## Hannan-Quinn -4.126666
```

```
        infocriteria(w.final.model.2)
```

```
##
## Akaike       -4.171652
## Bayes        -4.130750
## Shibata      -4.171835
## Hannan-Quinn -4.155629
```

```
#applying the ARIMA-GARCH model to the daily data


#Fit ARIMA on differenced series
    d.final.aic=Inf
    d.final.order=c(0,0,0)
    for (p in 0:3) for (d in 0:1) for (q in 0:3)
    {
        d.current.aic=AIC(arima(returndailydata,order=c(p, d, q)))
        if(d.current.aic<d.final.aic)
        {
            d.final.aic=d.current.aic
            d.final.order=c(p,d,q)
            d.final.arima=arima(returndailydata, order=d.final.order)
        }
    }

    # What is the selected order?
    d.final.order
```

```
## [1] 3 0 3
```

```
#Select model with smallest BIC
            final.bic = Inf
            final.order = c(0,0)
            for (p in 0:2) for (q in 0:2)
            {
                spec = ugarchspec(variance.model=list(garchOrder=c(p,q)),
                mean.model=list(armaOrder=c(d.final.order[1], d.final.order[3]), include.mean=T),
                distribution.model="std")
                fit = ugarchfit(spec, SBUXDaily.train, solver = 'nloptr')
                current.bic = infocriteria(fit)[2]
                if (current.bic < final.bic)
                {
                    final.bic = current.bic
                    final.order = c(p, q)
                }
            }
        final.order
```

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

```
    #Refine the ARMA order
        final.bic = Inf
        final.order.arma = c(0,0)
        for (p in 0:3) for (q in 0:3)
        {
            spec = ugarchspec(variance.model=list(garchOrder=c(final.order[1],final.order[2])),
            mean.model=list(armaOrder=c(p, q), include.mean=T),
            distribution.model="std")
            fit = ugarchfit(spec, SBUXDaily.train, solver = 'hybrid')
            current.bic = infocriteria(fit)[2]
            if (current.bic < final.bic)
            {
                final.bic = current.bic
                final.order.arma = c(p, q)
            }
        }
        final.order.arma
```

```
## [1] 0 0
```

```
    #Refine the GARCH order
        final.bic = Inf
        final.order.garch = c(0,0)
        for (p in 0:2) for (q in 0:2)
        {
            spec = ugarchspec(variance.model=list(garchOrder=c(p,q)),
            mean.model=list(armaOrder=c(final.order.arma[1], final.order.arma[2]),
                include.mean=T), distribution.model="std")
                fit = ugarchfit(spec, SBUXDaily.train, solver = 'hybrid')
                current.bic = infocriteria(fit)[2]
                if (current.bic < final.bic)
            {
                final.bic = current.bic
                final.order.garch = c(p, q)
                }
              }
        final.order.garch
```

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

```
        final.order.garch.daily=final.order.garch
        #Goodness of Fit
        d.spec.1 = ugarchspec(variance.model=list(garchOrder=c(1,1)),
            mean.model=list(armaOrder=c(5,6),
            include.mean=T), distribution.model="std")
        d.final.model.1 = ugarchfit(d.spec.1, SBUXDaily.train, solver = 'hybrid')
d.final.model.1
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(5,0,6)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error    t value Pr(>|t|)
## mu      0.001010    0.000203     4.9760 0.000001
## ar1    -0.541460    0.014404   -37.5917 0.000000
## ar2     1.397867    0.004534   308.2852 0.000000
## ar3     1.304124    0.019932    65.4283 0.000000
## ar4    -0.603742    0.004975  -121.3647 0.000000
## ar5    -0.927424    0.010177   -91.1290 0.000000
## ma1     0.485790    0.000230  2110.1990 0.000000
## ma2    -1.443656    0.000907 -1592.0763 0.000000
## ma3    -1.235745    0.000230 -5371.7117 0.000000
## ma4     0.699399    0.000111  6304.0675 0.000000
## ma5     0.902705    0.000126  7183.7101 0.000000
## ma6    -0.066762    0.000972   -68.6824 0.000000
## omega   0.000019    0.000005     3.4720 0.000517
## alpha1  0.137141    0.029915     4.5843 0.000005
## beta1   0.780061    0.047249    16.5097 0.000000
## shape   4.270910    0.359670    11.8745 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error    t value Pr(>|t|)
## mu      0.001010    0.000197     5.1199 0.000000
## ar1    -0.541460    0.031447   -17.2184 0.000000
## ar2     1.397867    0.005934   235.5673 0.000000
## ar3     1.304124    0.042598    30.6147 0.000000
## ar4    -0.603742    0.006470   -93.3125 0.000000
## ar5    -0.927424    0.020165   -45.9922 0.000000
## ma1     0.485790    0.000435  1115.9271 0.000000
## ma2    -1.443656    0.002104  -686.1660 0.000000
## ma3    -1.235745    0.000589 -2099.6943 0.000000
## ma4     0.699399    0.000334  2092.7489 0.000000
## ma5     0.902705    0.000256  3526.8900 0.000000
## ma6    -0.066762    0.002000   -33.3765 0.000000
## omega   0.000019    0.000009     2.1722 0.029843
## alpha1  0.137141    0.049356     2.7786 0.005460
## beta1   0.780061    0.079090     9.8629 0.000000
## shape   4.270910    0.388859    10.9832 0.000000
##
## LogLikelihood : 7383.065
##
## Information Criteria
## ------------------------------------
##
## Akaike       -5.8678
## Bayes        -5.8307
## Shibata      -5.8679
## Hannan-Quinn -5.8544
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                           statistic p-value
## Lag[1]                       0.9507  0.3295
## Lag[2*(p+q)+(p+q)-1][32]   11.1556  1.0000
## Lag[4*(p+q)+(p+q)-1][54]   18.2393  0.9963
## d.o.f=11
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                          statistic p-value
## Lag[1]                      0.0243  0.8761
## Lag[2*(p+q)+(p+q)-1][5]     1.0733  0.8429
## Lag[4*(p+q)+(p+q)-1][9]     1.9294  0.9133
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.3850 0.500 2.000  0.5349
## ARCH Lag[5]    0.6506 1.440 1.667  0.8385
## ARCH Lag[7]    1.2738 2.315 1.543  0.8654
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  3.8791
## Individual Statistics:
## mu     0.07344
## ar1    0.10751
## ar2    0.07068
## ar3    0.02440
## ar4    0.02378
## ar5    0.22935
## ma1    0.10595
## ma2    0.08845
## ma3    0.03335
## ma4    0.02809
## ma5    0.20334
## ma6    0.03553
## omega  0.33582
## alpha1 0.40595
## beta1  0.41455
## shape  0.36669
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          3.46 3.75 4.3
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias           0.6060 0.5445
## Negative Sign Bias  0.6347 0.5257
## Positive Sign Bias  0.5812 0.5612
## Joint Effect        1.4086 0.7035
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     15.62       0.6825
## 2    30     27.77       0.5303
## 3    40     38.15       0.5084
## 4    50     53.58       0.3031
##
##
## Elapsed time : 3.710871
```

```
        d.spec.2 = ugarchspec(variance.model=list(garchOrder=c(1,1)),
            mean.model=list(armaOrder=c(1,0),
            include.mean=T), distribution.model="std")
        d.final.model.2 = ugarchfit(d.spec.2, SBUXDaily.train, solver = 'hybrid')
d.final.model.2
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(1,0,0)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000999    0.000211   4.7436 0.000002
## ar1    -0.046471    0.020305  -2.2887 0.022098
## omega   0.000019    0.000005   3.5228 0.000427
## alpha1  0.131396    0.028608   4.5930 0.000004
## beta1   0.787900    0.044994  17.5112 0.000000
## shape   4.256556    0.356544  11.9384 0.000000
```

```
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000999    0.000199   5.0210 0.000001
## ar1    -0.046471    0.021094  -2.2031 0.027590
## omega   0.000019    0.000008   2.3968 0.016538
## alpha1  0.131396    0.045595   2.8818 0.003954
## beta1   0.787900    0.069834  11.2825 0.000000
## shape   4.256556    0.379937  11.2033 0.000000
##
## LogLikelihood : 7375.033
##
## Information Criteria
## ------------------------------------
##
## Akaike       -5.8694
## Bayes        -5.8555
## Shibata      -5.8694
## Hannan-Quinn -5.8643
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.3823  0.5364
## Lag[2*(p+q)+(p+q)-1][2]    0.4366  0.9775
## Lag[4*(p+q)+(p+q)-1][5]    2.3282  0.6159
## d.o.f=1
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                    0.05002  0.8230
## Lag[2*(p+q)+(p+q)-1][5]   1.18566  0.8164
## Lag[4*(p+q)+(p+q)-1][9]   2.04234  0.8999
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.4925 0.500 2.000  0.4828
## ARCH Lag[5]    0.7064 1.440 1.667  0.8215
## ARCH Lag[7]    1.3105 2.315 1.543  0.8585
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  1.0171
## Individual Statistics:
## mu     0.07179
## ar1    0.09968
## omega  0.31792
## alpha1 0.39571
## beta1  0.39194
## shape  0.34047
```

```
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.49 1.68 2.12
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias           0.1064 0.9153
## Negative Sign Bias  0.4392 0.6606
## Positive Sign Bias  0.9485 0.3429
## Joint Effect        1.3472 0.7179
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     17.77       0.5379
## 2    30     29.70       0.4288
## 3    40     47.45       0.1660
## 4    50     54.17       0.2836
##
##
## Elapsed time : 1.053396
```

```
        #Compare Information Criteria
        infocriteria(d.final.model.1)
```

```
##
## Akaike       -5.867833
## Bayes        -5.830695
## Shibata      -5.867914
## Hannan-Quinn -5.854353
```

```
        infocriteria(d.final.model.2)
```

```
##
## Akaike       -5.869401
## Bayes        -5.855474
## Shibata      -5.869413
## Hannan-Quinn -5.864346
```

```
w.final.model.1
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(2,0,4)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.004242    0.001063  3.98870 0.000066
## ar1     0.172420    0.470640  0.36635 0.714102
## ar2    -0.508797    0.731865 -0.69521 0.486926
## ma1    -0.266630    0.470816 -0.56632 0.571179
## ma2     0.521718    0.748237  0.69726 0.485638
## ma3    -0.092427    0.062402 -1.48115 0.138566
## ma4     0.049253    0.051853  0.94986 0.342183
## omega   0.000148    0.000086  1.72348 0.084801
## alpha1  0.117440    0.050432  2.32867 0.019877
## beta1   0.756865    0.103666  7.30099 0.000000
## shape   4.239876    0.846011  5.01161 0.000001
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.004242    0.001173  3.61700 0.000298
## ar1     0.172420    0.626239  0.27533 0.783065
## ar2    -0.508797    1.386408 -0.36699 0.713627
## ma1    -0.266630    0.633310 -0.42101 0.673747
## ma2     0.521718    1.426543  0.36572 0.714573
## ma3    -0.092427    0.093130 -0.99246 0.320974
## ma4     0.049253    0.057385  0.85829 0.390730
## omega   0.000148    0.000068  2.16226 0.030598
## alpha1  0.117440    0.059498  1.97386 0.048398
## beta1   0.756865    0.081481  9.28888 0.000000
## shape   4.239876    0.913269  4.64253 0.000003
##
## LogLikelihood : 1093.098
##
## Information Criteria
## ------------------------------------
##
## Akaike       -4.1619
## Bayes        -4.0719
## Shibata      -4.1628
## Hannan-Quinn -4.1267
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                          statistic p-value
## Lag[1]                      0.2922  0.5888
## Lag[2*(p+q)+(p+q)-1][17]    3.5581  1.0000
## Lag[4*(p+q)+(p+q)-1][29]    9.2066  0.9886
## d.o.f=6
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                    0.06061  0.8055
## Lag[2*(p+q)+(p+q)-1][5]   1.50754  0.7380
## Lag[4*(p+q)+(p+q)-1][9]   4.39269  0.5236
## d.o.f=2
##
```

```
## Weighted ARCH LM Tests
## ------------------------------------
##              Statistic Shape Scale P-Value
## ARCH Lag[3]     1.295 0.500 2.000  0.2551
## ARCH Lag[5]     2.473 1.440 1.667  0.3758
## ARCH Lag[7]     4.854 2.315 1.543  0.2397
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  1.7128
## Individual Statistics:
## mu     0.06231
## ar1    0.10403
## ar2    0.44736
## ma1    0.08503
## ma2    0.40604
## ma3    0.03544
## ma4    0.54369
## omega  0.22008
## alpha1 0.28542
## beta1  0.21667
## shape  0.19053
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          2.49 2.75 3.27
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                     t-value   prob sig
## Sign Bias             0.156 0.8761
## Negative Sign Bias    1.068 0.2859
## Positive Sign Bias    1.199 0.2309
## Joint Effect          6.187 0.1029
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     16.85       0.6003
## 2    30     33.62       0.2536
## 3    40     36.31       0.5933
## 4    50     52.69       0.3332
##
##
## Elapsed time : 0.8025389
```

```
w.final.model.2 ##### IS THIS BETTER? ######
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(0,0,0)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.004169    0.001161   3.5921 0.000328
## omega   0.000154    0.000084   1.8375 0.066136
## alpha1  0.129643    0.051378   2.5233 0.011625
## beta1   0.738807    0.103736   7.1220 0.000000
## shape   4.417401    0.873622   5.0564 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.004169    0.001108   3.7626 0.000168
## omega   0.000154    0.000063   2.4692 0.013542
## alpha1  0.129643    0.056644   2.2887 0.022096
## beta1   0.738807    0.074784   9.8792 0.000000
## shape   4.417401    0.786685   5.6152 0.000000
##
## LogLikelihood : 1089.63
##
## Information Criteria
## ------------------------------------
##
## Akaike       -4.1717
## Bayes        -4.1308
## Shibata      -4.1718
## Hannan-Quinn -4.1556
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      1.841  0.1749
## Lag[2*(p+q)+(p+q)-1][2]     1.883  0.2829
## Lag[4*(p+q)+(p+q)-1][5]     3.145  0.3812
## d.o.f=0
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                  0.0001245  0.9911
## Lag[2*(p+q)+(p+q)-1][5] 1.3502231  0.7766
## Lag[4*(p+q)+(p+q)-1][9] 3.5995692  0.6573
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.8793 0.500 2.000  0.3484
## ARCH Lag[5]    2.0081 1.440 1.667  0.4695
## ARCH Lag[7]    3.8286 2.315 1.543  0.3719
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  0.5528
## Individual Statistics:
## mu     0.05703
## omega  0.19066
## alpha1 0.26089
## beta1  0.19450
## shape  0.15555
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.28 1.47 1.88
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias           0.4454 0.6562
## Negative Sign Bias  0.6305 0.5286
## Positive Sign Bias  1.1165 0.2647
## Joint Effect        5.3699 0.1466
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     17.77       0.5379
## 2    30     36.04       0.1724
## 3    40     31.69       0.7907
## 4    50     40.58       0.7986
##
##
## Elapsed time : 0.254853
```

```
d.final.model.1
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(5,0,6)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error    t value Pr(>|t|)
## mu      0.001010    0.000203     4.9760 0.000001
## ar1    -0.541460    0.014404   -37.5917 0.000000
## ar2     1.397867    0.004534   308.2852 0.000000
## ar3     1.304124    0.019932    65.4283 0.000000
## ar4    -0.603742    0.004975  -121.3647 0.000000
## ar5    -0.927424    0.010177   -91.1290 0.000000
## ma1     0.485790    0.000230  2110.1990 0.000000
## ma2    -1.443656    0.000907 -1592.0763 0.000000
## ma3    -1.235745    0.000230 -5371.7117 0.000000
## ma4     0.699399    0.000111  6304.0675 0.000000
## ma5     0.902705    0.000126  7183.7101 0.000000
## ma6    -0.066762    0.000972   -68.6824 0.000000
## omega   0.000019    0.000005     3.4720 0.000517
## alpha1  0.137141    0.029915     4.5843 0.000005
## beta1   0.780061    0.047249    16.5097 0.000000
## shape   4.270910    0.359670    11.8745 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error    t value Pr(>|t|)
## mu      0.001010    0.000197     5.1199 0.000000
## ar1    -0.541460    0.031447   -17.2184 0.000000
## ar2     1.397867    0.005934   235.5673 0.000000
## ar3     1.304124    0.042598    30.6147 0.000000
## ar4    -0.603742    0.006470   -93.3125 0.000000
## ar5    -0.927424    0.020165   -45.9922 0.000000
## ma1     0.485790    0.000435  1115.9271 0.000000
## ma2    -1.443656    0.002104  -686.1660 0.000000
## ma3    -1.235745    0.000589 -2099.6943 0.000000
## ma4     0.699399    0.000334  2092.7489 0.000000
## ma5     0.902705    0.000256  3526.8900 0.000000
## ma6    -0.066762    0.002000   -33.3765 0.000000
## omega   0.000019    0.000009     2.1722 0.029843
## alpha1  0.137141    0.049356     2.7786 0.005460
## beta1   0.780061    0.079090     9.8629 0.000000
## shape   4.270910    0.388859    10.9832 0.000000
##
## LogLikelihood : 7383.065
##
## Information Criteria
## ------------------------------------
##
## Akaike       -5.8678
## Bayes        -5.8307
## Shibata      -5.8679
## Hannan-Quinn -5.8544
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                          statistic p-value
## Lag[1]                      0.9507  0.3295
## Lag[2*(p+q)+(p+q)-1][32]   11.1556  1.0000
## Lag[4*(p+q)+(p+q)-1][54]   18.2393  0.9963
## d.o.f=11
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                          statistic p-value
## Lag[1]                      0.0243  0.8761
## Lag[2*(p+q)+(p+q)-1][5]     1.0733  0.8429
## Lag[4*(p+q)+(p+q)-1][9]     1.9294  0.9133
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.3850 0.500 2.000  0.5349
## ARCH Lag[5]    0.6506 1.440 1.667  0.8385
## ARCH Lag[7]    1.2738 2.315 1.543  0.8654
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  3.8791
## Individual Statistics:
## mu     0.07344
## ar1    0.10751
## ar2    0.07068
## ar3    0.02440
## ar4    0.02378
## ar5    0.22935
## ma1    0.10595
## ma2    0.08845
## ma3    0.03335
## ma4    0.02809
## ma5    0.20334
## ma6    0.03553
## omega  0.33582
## alpha1 0.40595
## beta1  0.41455
## shape  0.36669
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          3.46 3.75 4.3
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias           0.6060 0.5445
## Negative Sign Bias  0.6347 0.5257
## Positive Sign Bias  0.5812 0.5612
## Joint Effect        1.4086 0.7035
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     15.62       0.6825
## 2    30     27.77       0.5303
## 3    40     38.15       0.5084
## 4    50     53.58       0.3031
```

```
##
##
## Elapsed time : 3.710871
```

```
d.final.model.2
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(1,0,0)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000999    0.000211   4.7436 0.000002
## ar1    -0.046471    0.020305  -2.2887 0.022098
## omega   0.000019    0.000005   3.5228 0.000427
## alpha1  0.131396    0.028608   4.5930 0.000004
## beta1   0.787900    0.044994  17.5112 0.000000
## shape   4.256556    0.356544  11.9384 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000999    0.000199   5.0210 0.000001
## ar1    -0.046471    0.021094  -2.2031 0.027590
## omega   0.000019    0.000008   2.3968 0.016538
## alpha1  0.131396    0.045595   2.8818 0.003954
## beta1   0.787900    0.069834  11.2825 0.000000
## shape   4.256556    0.379937  11.2033 0.000000
##
## LogLikelihood : 7375.033
##
## Information Criteria
## ------------------------------------
##
## Akaike       -5.8694
## Bayes        -5.8555
## Shibata      -5.8694
## Hannan-Quinn -5.8643
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.3823  0.5364
## Lag[2*(p+q)+(p+q)-1][2]    0.4366  0.9775
## Lag[4*(p+q)+(p+q)-1][5]    2.3282  0.6159
## d.o.f=1
## H0 : No serial correlation
##
```

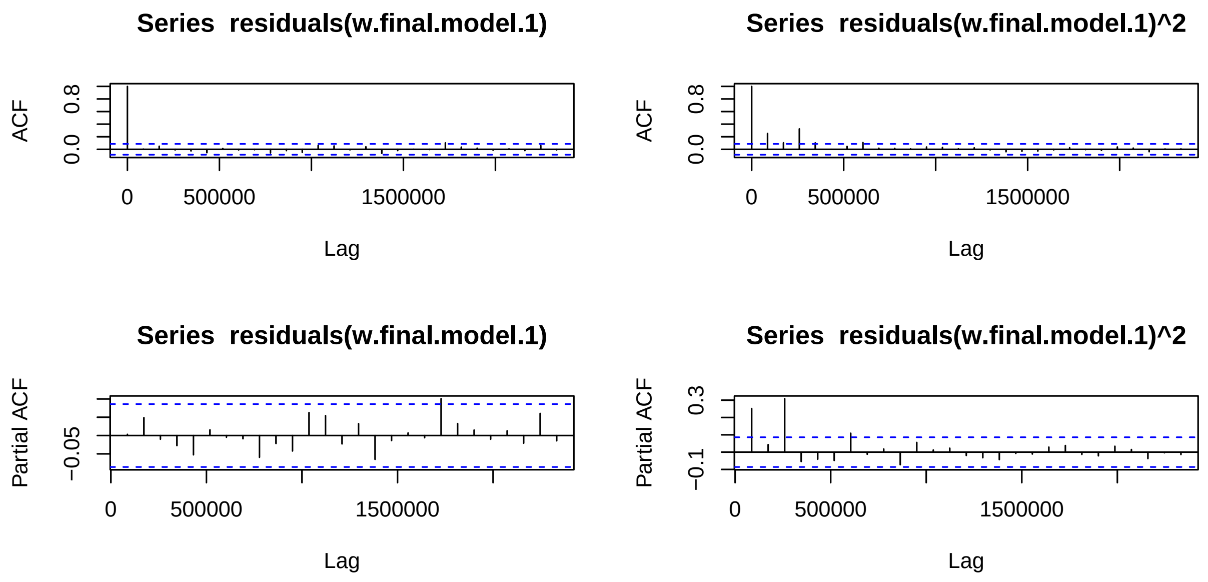
```
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                    0.05002  0.8230
## Lag[2*(p+q)+(p+q)-1][5]   1.18566  0.8164
## Lag[4*(p+q)+(p+q)-1][9]   2.04234  0.8999
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.4925 0.500 2.000  0.4828
## ARCH Lag[5]    0.7064 1.440 1.667  0.8215
## ARCH Lag[7]    1.3105 2.315 1.543  0.8585
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  1.0171
## Individual Statistics:
## mu     0.07179
## ar1    0.09968
## omega  0.31792
## alpha1 0.39571
## beta1  0.39194
## shape  0.34047
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.49 1.68 2.12
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias           0.1064 0.9153
## Negative Sign Bias  0.4392 0.6606
## Positive Sign Bias  0.9485 0.3429
## Joint Effect        1.3472 0.7179
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     17.77       0.5379
## 2    30     29.70       0.4288
## 3    40     47.45       0.1660
## 4    50     54.17       0.2836
##
##
## Elapsed time : 1.053396
```

Response: Analysis of the ARMA GARCH Fit for the Weekly and Monthly Data

The estimated equation for the weekly differenced data is: $Y_{t} = 0.004 + 0.172Y_{t-1} - 0.51Y_{t-2} - 0.27Z_{t-1} + 0.52Z_{t-2} - 0.09Z_{t-3} +0.05Z_{t-4}

3b. Evaluate the model residuals and squared residuals using the ACF and PACF plots as well as hypothesis testing for serial correlation. What would you conclude based on this analysis?

```
#plotting the acfs and pacfs of residuals and squared residuals

par(mfrow=c(2,2))
acf(residuals(w.final.model.1))
acf(residuals(w.final.model.1)^2)

pacf(residuals(w.final.model.1))
pacf(residuals(w.final.model.1)^2)
```



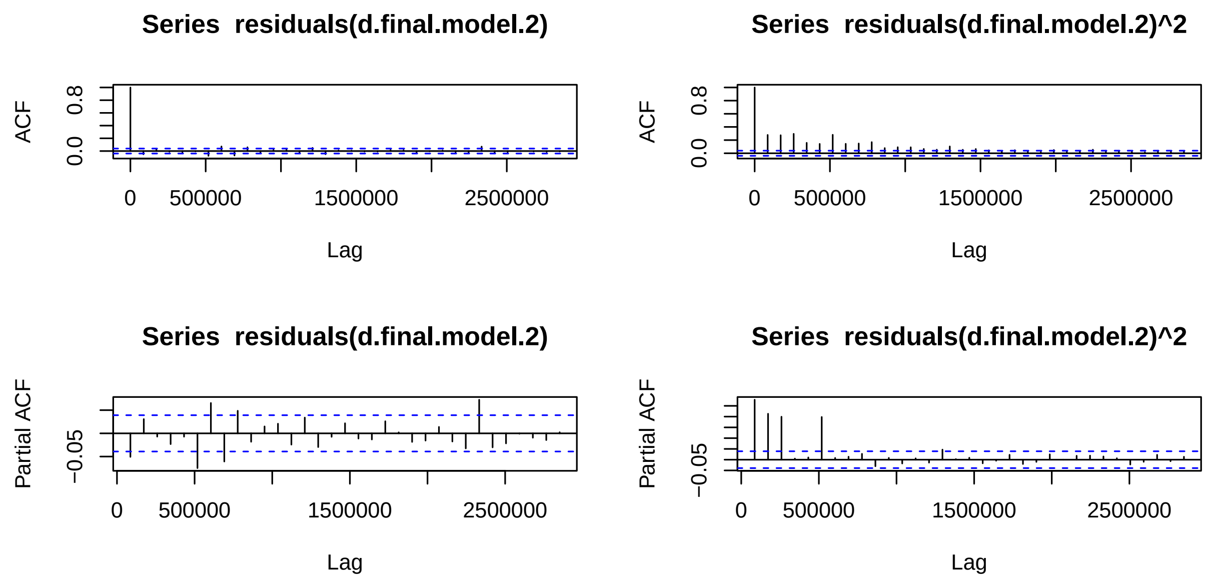
```
#test for serial correlation
Box.test(residuals(w.final.model.1),lag=,type='Ljung',fitdf=)
```

```
##
##  Box-Ljung test
##
## data:  residuals(w.final.model.1)
## X-squared = 0.005514, df = 1, p-value = 0.9408
```

```
Box.test(residuals(w.final.model.1)^2,lag=,type='Ljung',fitdf=)
```

```
##
##  Box-Ljung test
##
## data:  residuals(w.final.model.1)^2
## X-squared = 33.14, df = 1, p-value = 8.576e-09
```

```
par(mfrow=c(2,2))
acf(residuals(d.final.model.2))
acf(residuals(d.final.model.2)^2)

pacf(residuals(d.final.model.2))
pacf(residuals(d.final.model.2)^2)
```



```
#test for serial correlation
Box.test(residuals(d.final.model.2),lag=,type='Ljung',fitdf=)
```

```
##
##  Box-Ljung test
##
## data:  residuals(d.final.model.2)
## X-squared = 6.4722, df = 1, p-value = 0.01096
```

```
Box.test(residuals(d.final.model.2)^2,lag=,type='Ljung',fitdf=)
```

```
##
##  Box-Ljung test
##
## data:  residuals(d.final.model.2)^2
## X-squared = 194.78, df = 1, p-value < 2.2e-16
```

Response:ARIMA residual analysis for the Weekly and daily Data

From the residuals' plots for the model w.final.model.1, we can see that there is no serial correlation, also confirmed by the p_value of the Box test, as it is large enough to fail to reject the null hypothesis. Regarding the plots of the squared residuals, we can see that the Arch effect was somewhat controlled by the model, still showing a cyclical pattern. Based on the Box test we reject the null hypothesis that there is no serial correlation.

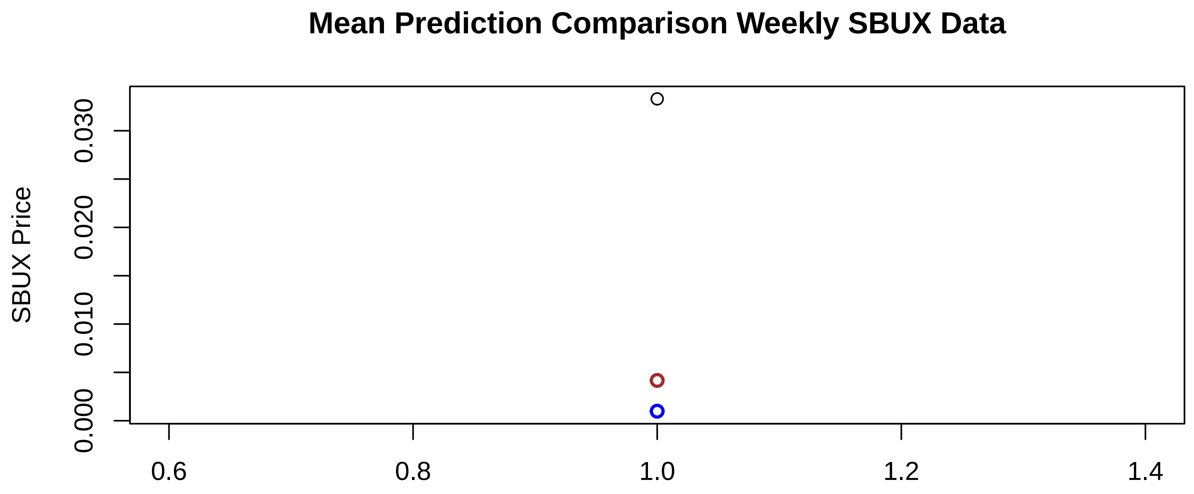
From the residuals' plots for the d.final.model.2 model, we can see that there is some serial correlation, also confirmed by the p_value of the Box test. Regarding the plots of the squared residuals, we can see that the Arch effect was somewhat controlled by the model, still showing a cyclical pattern. Based on the Box test we reject the null hypothesis that there is no serial correlation.

3c. Apply the model identified in (3a) and forecast the mean and the variance of the last week of data. Plot the predicted data to compare the predicted values to the actual observed ones. Include 95% confidence intervals for the forecasts in the corresponding plots. Interpret the results, particularly comparing forecast using daily versus weekly data.

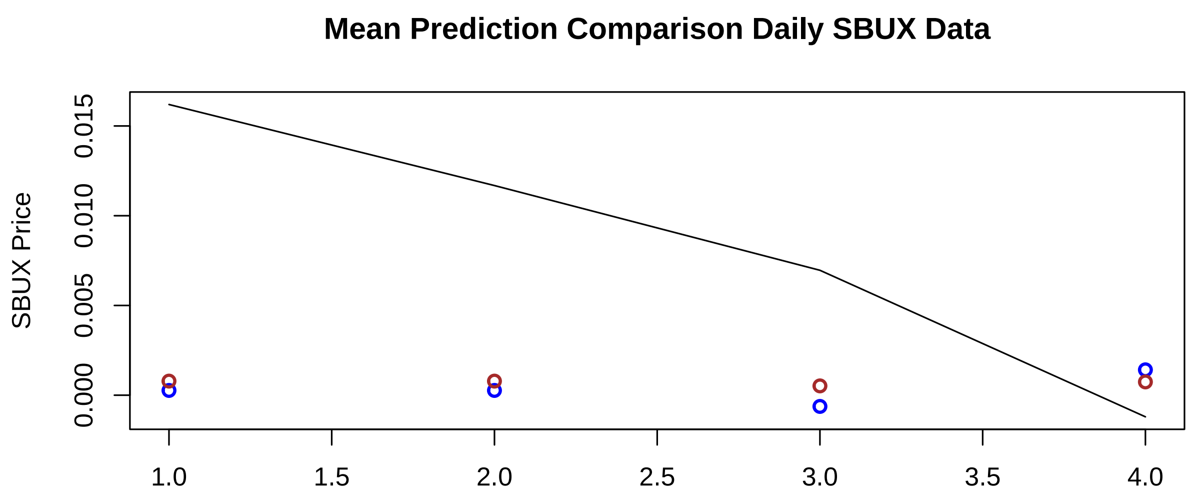
Forecasting and plotting weekly and daily data using the ARIMA model

```r
#predicting last week data using weekly data model
#Prediction of the return time series and the volatility sigma
        wnfore = length(SBUXWeekly.test)
        w.fore.series.1 = NULL
        w.fore.sigma.1 = NULL
        w.fore.series.2 = NULL
        w.fore.sigma.2 = NULL
        for(f in 1: wnfore)
        {
            #Fit models
            wdata = SBUXWeekly.train
            if(f>2)
            wdata = c(SBUXWeekly.train,SBUXWeekly.test[1:(f-1)])
            w.final.model.1 = ugarchfit(w.spec.1, wdata, solver = 'hybrid')
            w.final.model.2 = ugarchfit(w.spec.2, wdata, solver = 'hybrid')
            #Forecast
            wfore = ugarchforecast(w.final.model.1, n.ahead=1)
            w.fore.series.1 = c(w.fore.series.1, wfore@forecast$seriesFor)
            w.fore.sigma.1 = c(w.fore.sigma.1, wfore@forecast$sigmaFor)
            wfore = ugarchforecast(w.final.model.2, n.ahead=1)
            w.fore.series.2 = c(w.fore.series.2, wfore@forecast$seriesFor)
            w.fore.sigma.2 = c(w.fore.sigma.2, wfore@forecast$sigmaFor)

        }
        w.fore.series.1[is.nan(w.fore.series.1)]=0

#predicting last week data using daily data model
#Prediction of the return time series and the volatility sigma
        dnfore = length(SBUXDaily.test)
        d.fore.series.1 = NULL
        d.fore.sigma.1 = NULL
        d.fore.series.2 = NULL
        d.fore.sigma.2 = NULL
        for(f in 1: dnfore)
        {
            #Fit models
            ddata = SBUXDaily.train
            if(f>2)
            ddata = c(SBUXDaily.train,SBUXDaily.test[1:(f-1)])
            d.final.model.1 = ugarchfit(d.spec.1, ddata, solver = 'hybrid')
            d.final.model.2 = ugarchfit(d.spec.2, ddata, solver = 'hybrid')
            #Forecast
            dfore = ugarchforecast(d.final.model.1, n.ahead=1)
            d.fore.series.1 = c(d.fore.series.1, dfore@forecast$seriesFor)
            d.fore.sigma.1 = c(d.fore.sigma.1, dfore@forecast$sigmaFor)
            dfore = ugarchforecast(d.final.model.2, n.ahead=1)
            d.fore.series.2 = c(d.fore.series.2, dfore@forecast$seriesFor)
            d.fore.sigma.2 = c(d.fore.sigma.2, dfore@forecast$sigmaFor)

        }
        d.fore.series.1[is.nan(d.fore.series.1)]=0
```

```r
#plotting forecasts and comparing with train data
        #Weekly
        #Mean Prediction Comparison Plot
        n=length(SBUXWeekly)
        ymin = min(c(as.vector(SBUXWeekly.test),w.fore.series.1,w.fore.series.2))
        ymax = max(c(as.vector(SBUXWeekly.test),w.fore.series.1,w.fore.series.2))
        data.plot = SBUXWeekly.test
        names(data.plot)="Fore"
        plot(SBUXWeekly.test,type="o", ylim=c(ymin,ymax), xlab=" ",
        ylab="SBUX Price",main="Mean Prediction Comparison Weekly SBUX Data")
        data.plot$Fore=d.fore.series.1
        points(w.fore.series.1,lwd= 2, col="blue")
        #data.plot$Fore=d.fore.series.2
        points(w.fore.series.2,lwd= 2, col="brown")
```



```r
        #Daily
        #Mean Prediction Comparison Plot
        n=length(SBUXDaily)
        ymin = min(c(as.vector(SBUXDaily.test),d.fore.series.1,d.fore.series.2))
        ymax = max(c(as.vector(SBUXDaily.test),d.fore.series.1,d.fore.series.2))
        data.plot = SBUXDaily.test
        names(data.plot)="Fore"
        plot(SBUXDaily.test,type="l", ylim=c(ymin,ymax), xlab=" ",
        ylab="SBUX Price",main="Mean Prediction Comparison Daily SBUX Data")
        #data.plot$Fore=d.fore.series.1
        points(d.fore.series.1,lwd= 2, col="blue")
        #data.plot$Fore=d.fore.series.2
        points(d.fore.series.2,lwd= 2, col="brown")
```



3d. Calculate Mean Absolute Percentage Error (MAPE) and Precision Measure (PM) for the mean forecasts. Compare the accuracy of the predictions for the daily and weekly time series using these two measures. Compare the accuracy of the forecasts with those obtained in (2d). Interpret the results.

Calculating accuracy measures of the ARIMA Fit for the Weekly and Monthly Data

```
#Compute Accuracy Measures for weekly data
        #Mean Absolute Percentage Error (MAPE)
            mean(abs(w.fore.series.1 - SBUXWeekly.test)/abs(SBUXWeekly.test+0.000001))
```

```
## [1] 0.9703872
```

```
            mean(abs(w.fore.series.2 - SBUXWeekly.test)/abs(SBUXWeekly.test+0.000001))
```

```
## [1] 0.8747545
```

```
        #Precision Measure (PM)
            sum((w.fore.series.1 - SBUXWeekly.test)^2)/sum((SBUXWeekly.test-mean(SBUXWeekly.test))^2)
```

```
## [1] Inf
```

```
            sum((w.fore.series.2 - SBUXWeekly.test)^2)/sum((SBUXWeekly.test-mean(SBUXWeekly.test))^2)
```

```
## [1] Inf
```

```
            #inf due to single value

#Compute Accuracy Measures for daily data
        #Mean Absolute Percentage Error (MAPE)
            mean(abs(d.fore.series.1 - SBUXDaily.test)/abs(SBUXDaily.test+0.000001))
```

```
## [1] 1.306349
```

```
            mean(abs(d.fore.series.2 - SBUXDaily.test)/abs(SBUXDaily.test+0.000001))
```

```
## [1] 1.106237
```

```
        #Precision Measure (PM)
            sum((d.fore.series.1 - SBUXDaily.test)^2)/sum((SBUXDaily.test-mean(SBUXDaily.test))^2)
```

```
## [1] 2.703897
```

```
sum((d.fore.series.2 - SBUXDaily.test)^2)/sum((SBUXDaily.test-mean(SBUXDaily.test))^2)
```

```
## [1] 2.420641
```

Response: Model comparison

While the accuracy measures clearly tell us which weekly and daily ARMA GARCH dimensions are better, the comparison with ARIMA tells us that the ARIMA implementation is a better fit. This is not comparable due to the different quantities we are evaluating the precision for.(return vs absolute level).

#Question 4: Reflection on the Modeling and Forecasting (10 points)

Based on the analysis above, discuss the application of ARIMA on the stock price versus the application of ARMA-GARCH on the stock return. How do the model fit the data? How well do the model predict? How do the models perform when using daily versus weekly data? Would you use one approach over another for different settings? What are some specific points of caution one would need to consider when applying those models?

For a stock like SBUX, where the volatility is quite high and the variance can have sudden jumps in the local means, it would be difficult to forecast values with a high precision.

The models we have implemented also give us a similar experience. However, it seems better to use daily data in this case for the simple reason of capturing more fluctuations rather than less. This adds a numerical predictability to the forecast data and also reduces some variance we see in a single forecast value (in case of using weekly data).

It would be advisable to prefer ARMA-GARCH as a tool over ARIMA modeling in cases where the data fluctuations are a rare event, or can be captured in less accurate forecasting. It is also a generally better idea to use return data for better precision and reducing trend and stationarity issues.

First and foremost, we need to be careful with the data cleaning and allocation with training and testing data that is polished. We need to ensure uniformity in order to compare models. Precision measure calculations are not wary of the inputs given to them, hence, it can become quite easy to miscompare numbers. Finally, the theoretical attributions that underly numerical analysis might not be the case. Causality is an important consideration while applying models that can give us results contrary to the actual scenarios.