ISYE 6402 Homework 6 Template

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 Tesla stock price data from January 2015 through end of December 2023. 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

```
## 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)
library(quantmod,warn.conflicts=FALSE)
```

```
#importing the data
dailydata <- read.csv("TSLA_Daily.csv", head = TRUE)
weeklydata <- read.csv("TSLA_Weekly.csv", head = TRUE)

#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
TWeekly <- ts(weeklydata$AdjClose,start=c(2015),freq=52)
TDaily <- ts(dailydata$AdjClose,start=c(2015),freq=252)
```

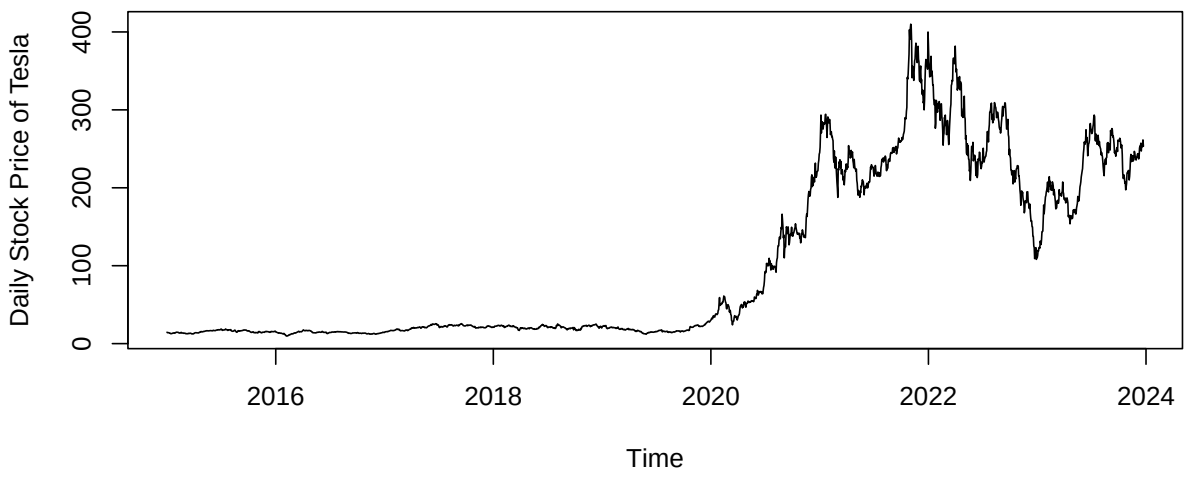
Question 1: Exploratory Data Analysis

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

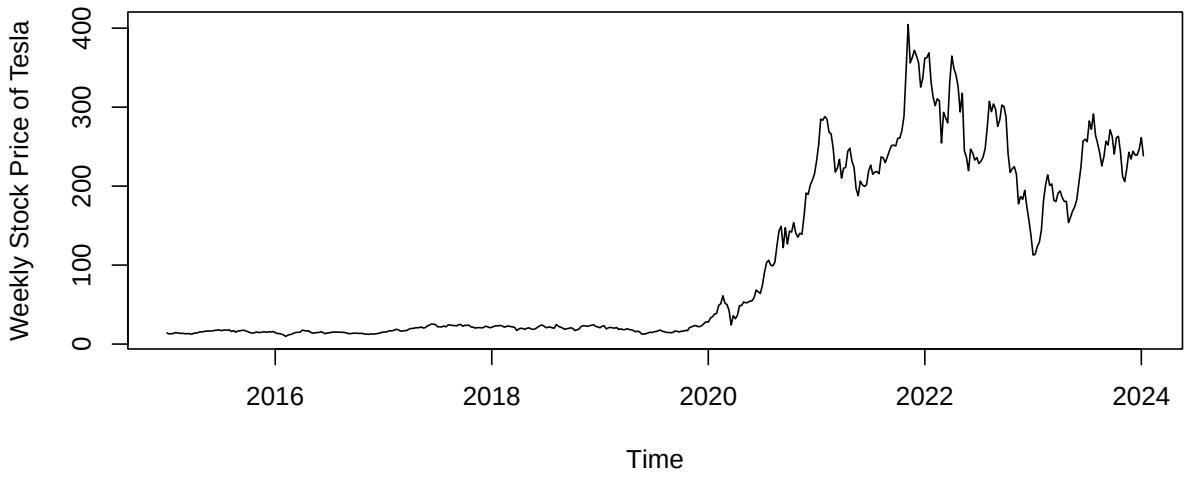
The decision to use daily stock price data or weekly stock price data depends on the trading horizon. If you are a higher frequency trader, you would trade stocks on a daily basis, whereas if you were a long term investor, the weekly stock price would be more suitable for your needs.

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

```
ts.plot(TDaily,ylab="Daily Stock Price of Tesla")
```




```
ts.plot(TWeekly,ylab="Weekly Stock Price of Tesla")
```



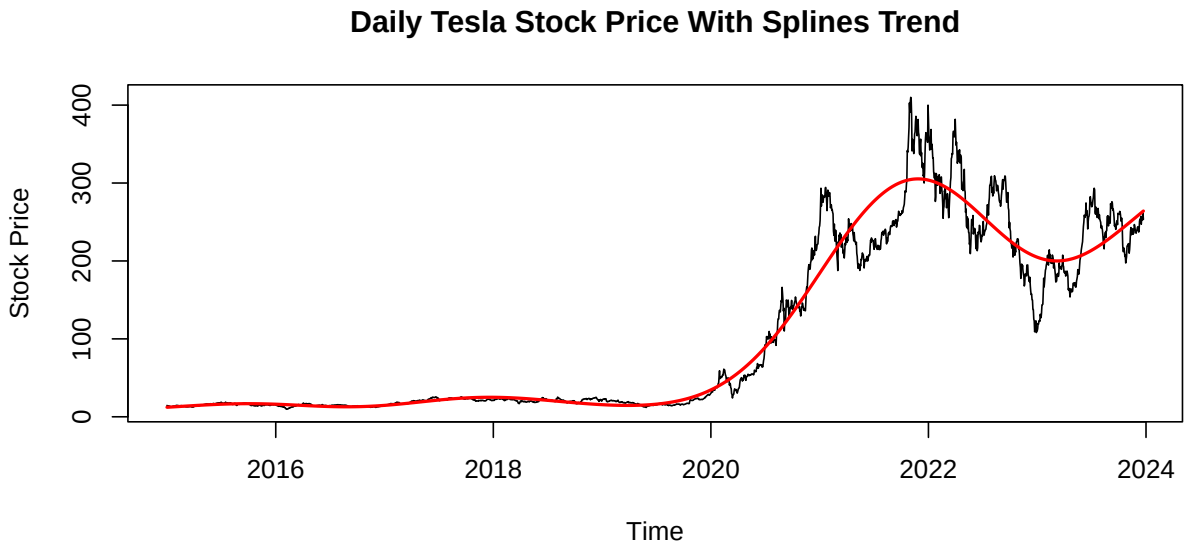
Response: Weekly vs Monthly Time Series data comparison

When plotting the entire time series, they look very similar, as the granularity is very small for both weekly and daily time series when observing a period of almost 10 years.

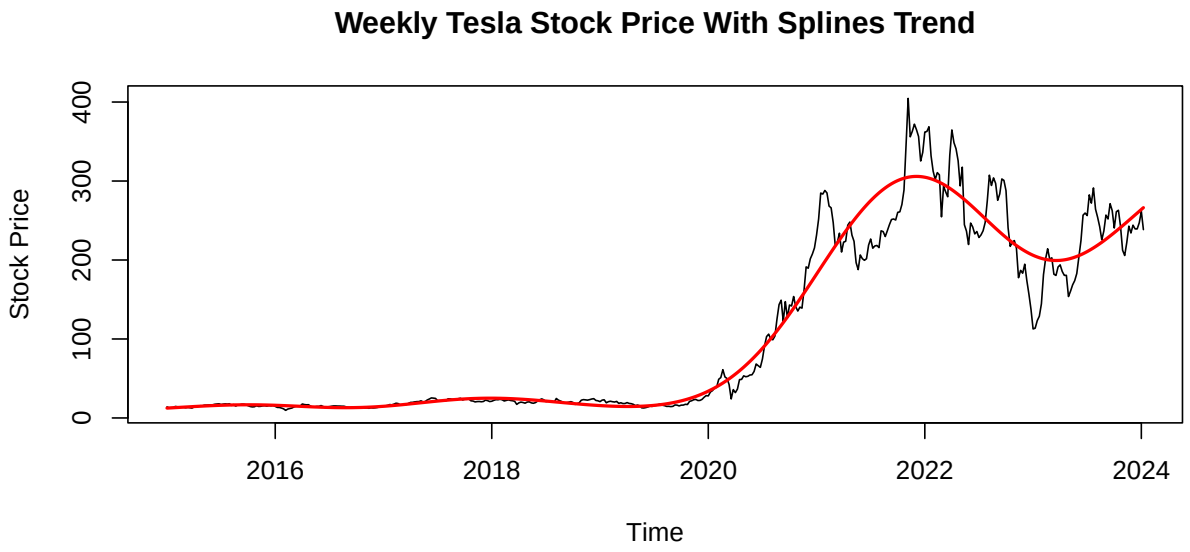
1c. Fit a non-parametric trend using splines regression to both the daily and weekly time series data. Overlay the fitted trends. How do the trends compare?

Analyzing weekly and daily data with trend fitting

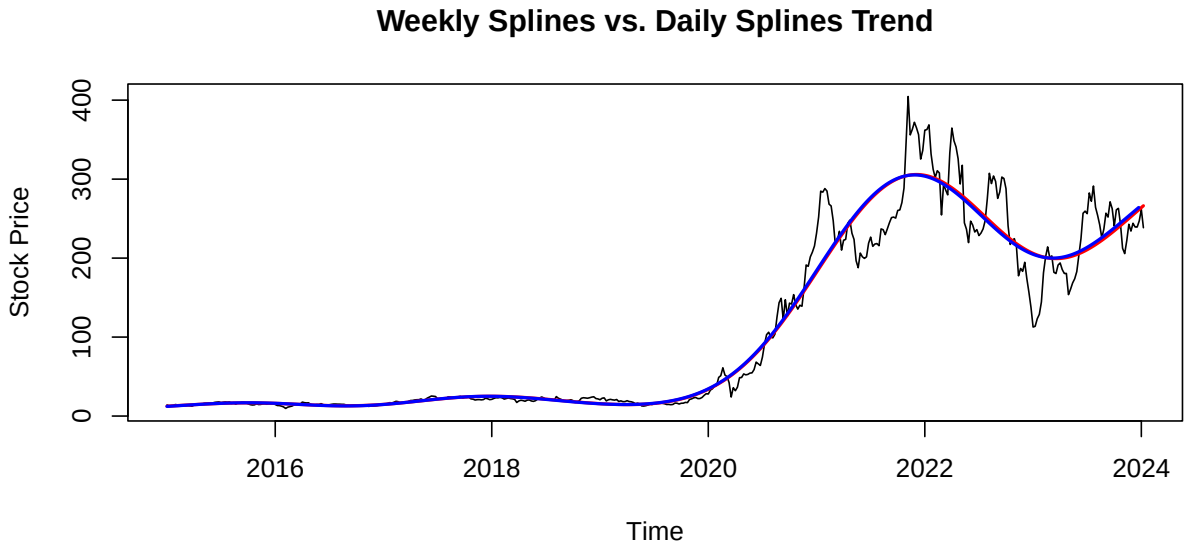
```r
time.pts = c(1:length(TDaily))
time.pts = c(time.pts - min(time.pts))/max(time.pts)

## Splines Trend Estimation Daily
gam.fit = gam(TDaily~s(time.pts))
daily.fit.gam = ts(fitted(gam.fit),start=c(2015, 1),frequency=252)
ts.plot(TDaily,ylab="Stock Price", main = "Daily Tesla Stock Price With Splines Trend")
lines(daily.fit.gam,lwd=2,col="red")
```




```r
time.pts = c(1:length(TWeekly))
time.pts = c(time.pts - min(time.pts))/max(time.pts)

## Splines Trend Estimation Weekly
gam.fit = gam(TWeekly~s(time.pts))
weekly.fit.gam = ts(fitted(gam.fit),start=c(2015, 1),frequency=52)
ts.plot(TWeekly,ylab="Stock Price", main = "Weekly Tesla Stock Price With Splines Trend")
lines(weekly.fit.gam,lwd=2,col="red")
```




```
ts.plot(TWeekly,ylab="Stock Price", main = "Weekly Splines vs. Daily Splines Trend")
lines(weekly.fit.gam,lwd=2,col="red")
lines(daily.fit.gam,lwd=2,col="blue")
```

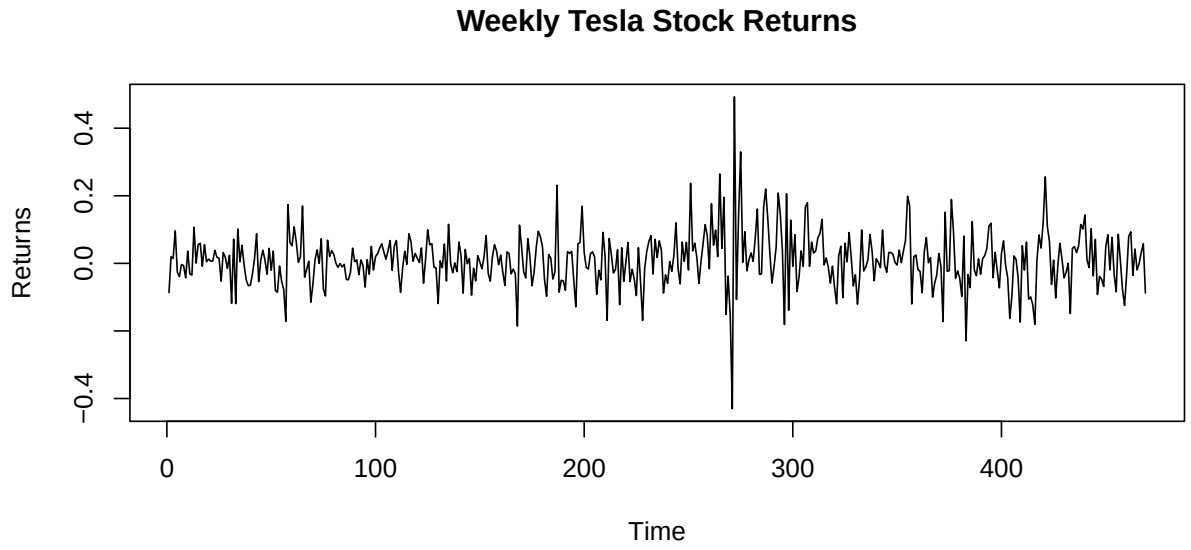


Response: Weekly vs Monthly Time Series data trend fit Both trends look almost the same when inspecting both daily and weekly trends.

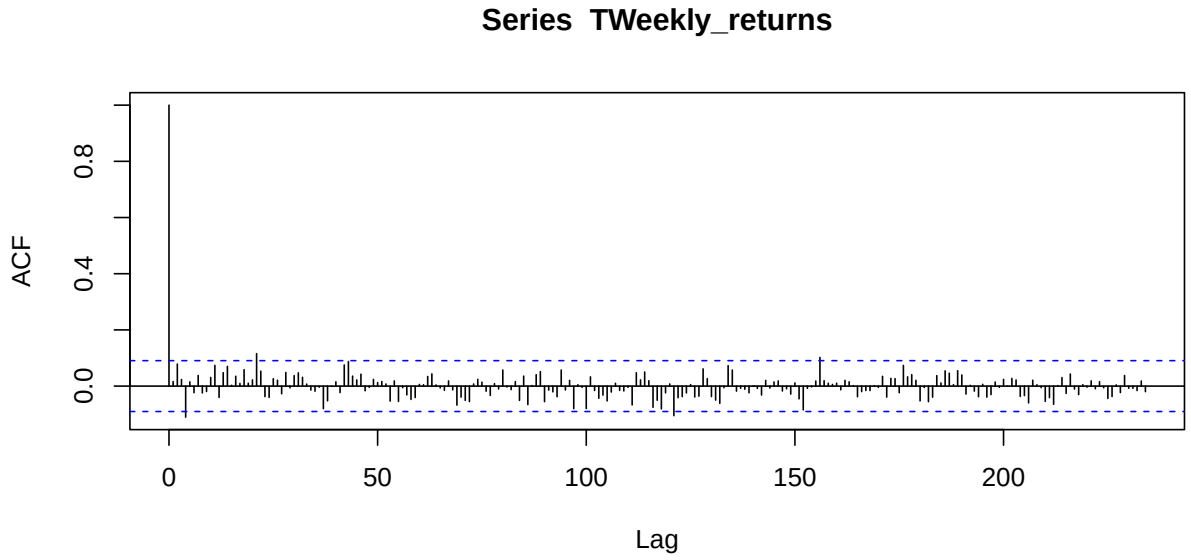
1d. Consider the return stock price computed as provided in the canvas homework assignment. Apply the formula described in Canvas to compute the return price based on the daily and weekly time series data. Plot the return time series and their corresponding ACF plots. How do the return time series compare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

Analyzing weekly and daily return data and comparing with original data

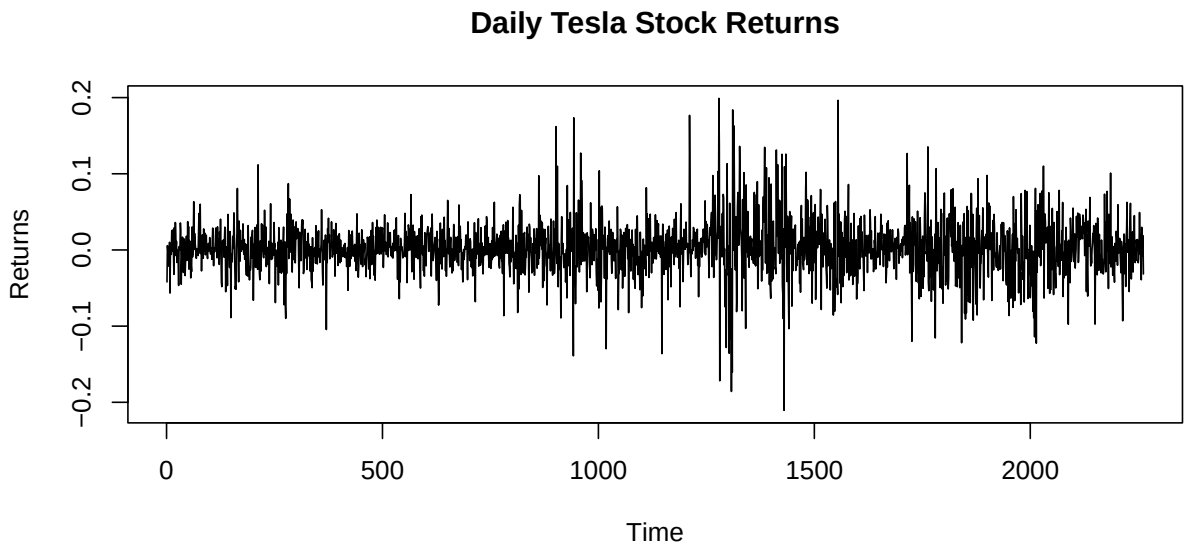
```
TWeekly_returns <- Delt(weeklydata$AdjClose, type='arithmetic', k=1)
TWeekly_returns <- ts(TWeekly_returns,start=c(2015),freq=52)
TWeekly_returns <- TWeekly_returns[!is.na(TWeekly_returns)]
ts.plot(TWeekly_returns,ylab="Returns", main = "Weekly Tesla Stock Returns")
```



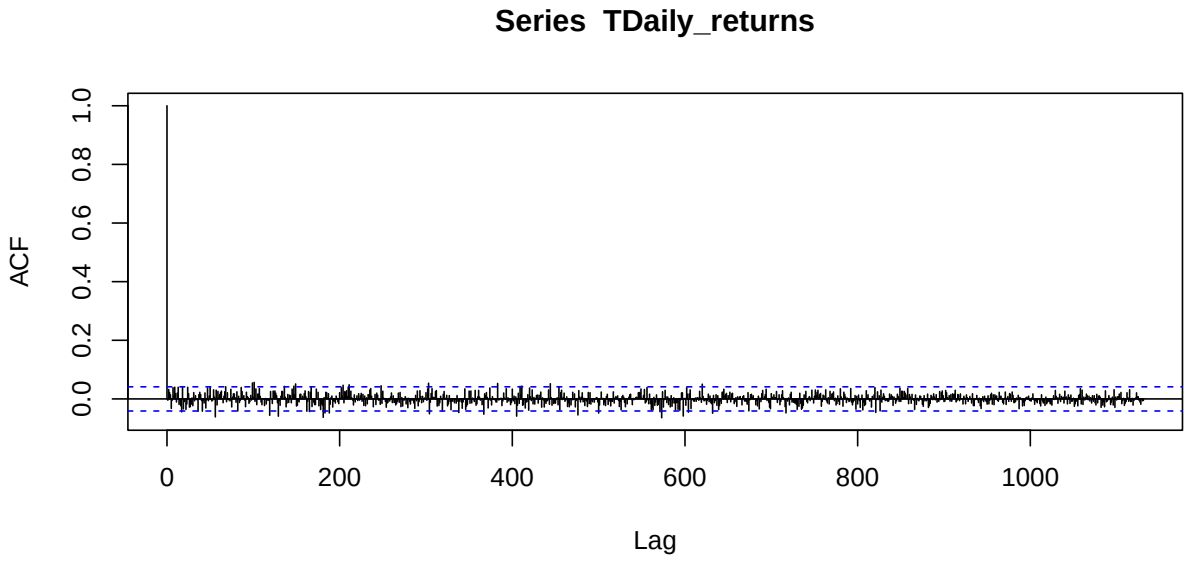

```
acf(TWeekly_returns, lag.max = length(TWeekly_returns)*0.5)
```




```
TDaily_returns <- Delt(dailydata$AdjClose, type='arithmetic', k=1)
TDaily_returns <- ts(TDaily_returns,start=c(2015),freq=252)
TDaily_returns <- TDaily_returns[!is.na(TDaily_returns)]
ts.plot(TDaily_returns,ylab="Returns", main = "Daily Tesla Stock Returns")
```




```
acf(TDaily_returns, lag.max = length(TDaily_returns)*0.5)
```



Response: Return series vs price series analysis

In terms of serial dependence, it appears that both time series do not have autocorrelation. In terms of stationarity, the mean is constant for both time series. For the variance there seems to be a small spike at around time=270, but otherwise it seems to be constant. This was likely during the period when Tesla stock skyrocketted in value. Due to this, stationarity may not be met, but we continue with caution.

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

2a. Divide the data into training and testing data set, where the training data exclude the last two weeks of data: December 21th-December 28th for weekly data, and December 18th-December 28th for daily data, with the testing data including the last 2 weeks of data. Apply the iterative model to fit an ARIMA(p,d,q) model with max AR and MA orders of 10 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. (If the AIC difference between d=1 and d=2 models is smaller than 2.5, use the simpler model).

```
n<-length(TDaily)
TDaily.train<-TDaily[1:(n-8)]
TDaily.test<-TDaily[(n-7):n]

n2<-length(TWeekly)
TWeekly.train<-TWeekly[1:(n2-2)]
TWeekly.test<-TWeekly[(n2-1):n2]


allorders <- function(ts_data, max_p, max_d, max_q){
  orders = data.frame()
  for (p in 0:max_p){
    for (d in 1:max_d){
      for (q in 0:max_q) {
        possibleError <- tryCatch({
          mod = arima(ts_data, order=c(p,d,q), method="ML")
          current.aic = AIC(mod)
          orders<-rbind(orders,c(p,d,q,current.aic))
        },
          error=function(e) e
        )
        if(inherits(possibleError, "error")) next

      }
    }
  }
  names(orders) <- c("p","d","q","AIC")
  return(orders[order(orders$AIC),])
}

head(allorders(TWeekly.train,10,2,10), 5)
```

```
##      p d  q      AIC
## 143  6 1 10 3623.779
## 152  6 2  8 3624.995
## 174  7 2  8 3624.996
## 109  4 2  9 3625.231
## 238 10 2  7 3626.059
```

The chosen p-d-q combination for the weekly data is (6, 1, 10). The model summary is below.

```
weekly_arima <- arima(TWeekly.train, order=c(6,1,10), method="ML")
weekly_arima
```

```
##
## Call:
## arima(x = TWeekly.train, order = c(6, 1, 10), method = "ML")
##
## Coefficients:
##           ar1      ar2      ar3      ar4     ar5      ar6     ma1     ma2
```

```
##       -0.5073  -0.1046  -1.0407  -0.1855  0.2833  -0.3673  0.6607  0.1901
## s.e.   0.2572   0.2832   0.2255   0.2716  0.2256   0.1881  0.2532  0.3139
##          ma3      ma4      ma5      ma6      ma7      ma8      ma9     ma10
##       1.1419  0.2877  -0.2522  0.3885  -0.1556  -0.0736  -0.0656  -0.2260
## s.e.  0.2605  0.3190   0.2767  0.2387   0.0816   0.0611   0.0612   0.0483
##
## sigma^2 estimated as 125.6:  log likelihood = -1794.89,  aic = 3623.78
```

Let's fit the daily data.

```
head(allorders(TDaily.train,10,2,10), 5)
```

```
##      p d  q      AIC
## 185  8 1  8 14144.37
## 230 10 1  9 14165.49
## 197  8 2  9 14166.95
## 242 10 2 10 14167.25
## 231 10 1 10 14171.32
```

The chosen p-d-q combination for the daily data is (8, 1, 8). The model summary is below.

```
daily_arima <- arima(TDaily.train, order=c(8,1,8), method="ML")
daily_arima
```

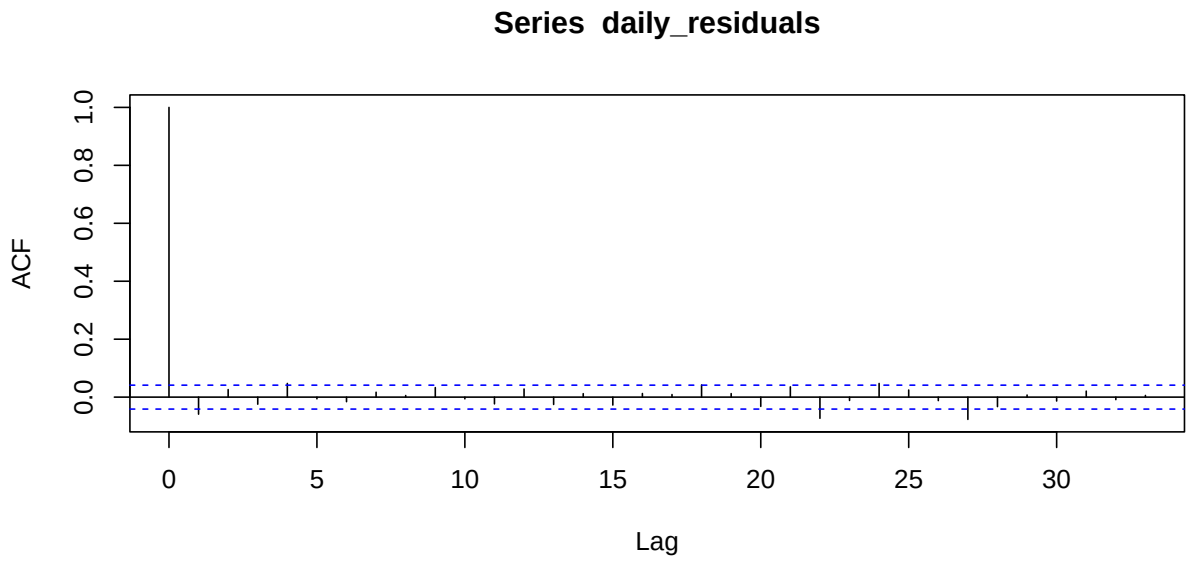
```
##
## Call:
## arima(x = TDaily.train, order = c(8, 1, 8), method = "ML")
##
## Coefficients:
##           ar1     ar2      ar3      ar4     ar5     ar6      ar7      ar8
##       -0.1631  0.6842  -0.0307  -0.3227  0.1487  0.5461  -0.2123  -0.8771
## s.e.      NaN     NaN   0.0263      NaN     NaN  0.0350      NaN      NaN
##          ma1      ma2     ma3     ma4     ma5      ma6     ma7     ma8
##       0.2024  -0.7012  0.0174  0.3678  -0.171  -0.5849  0.3024  0.9036
## s.e.     NaN      NaN     NaN     NaN     NaN      NaN     NaN     NaN
##
## sigma^2 estimated as 30.47:  log likelihood = -7055.19,  aic = 14144.37
```

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

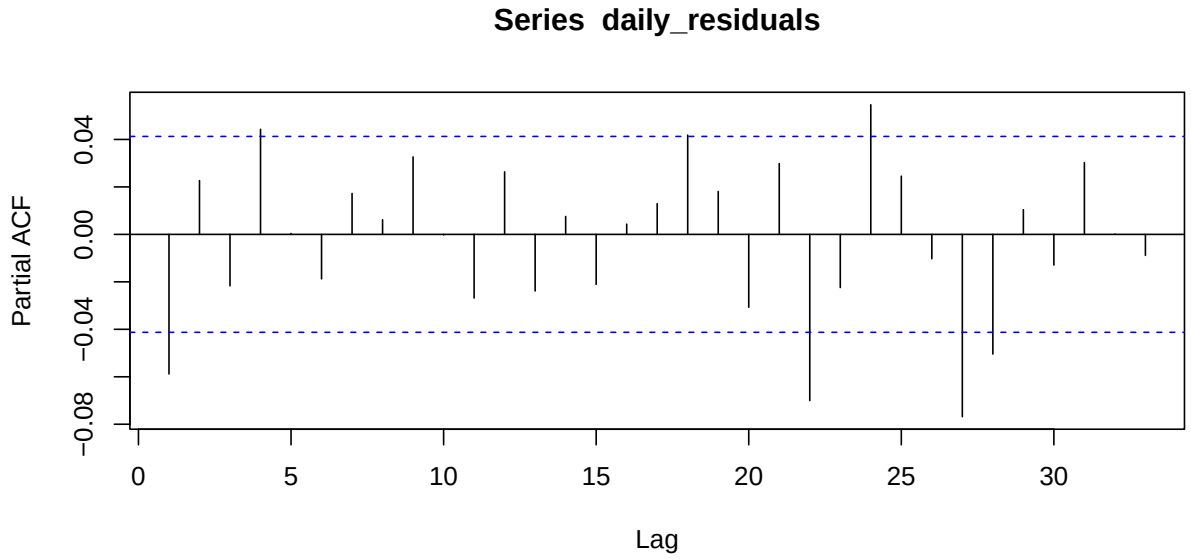
Based on the iterative approach, the best order values for the weekly data is 6, 1, 10, and for the daily data it is 8, 1, 8. Both are have a d-order of 1, so there's no need to check whether the next best combination is within an AIC value of 2.5.

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

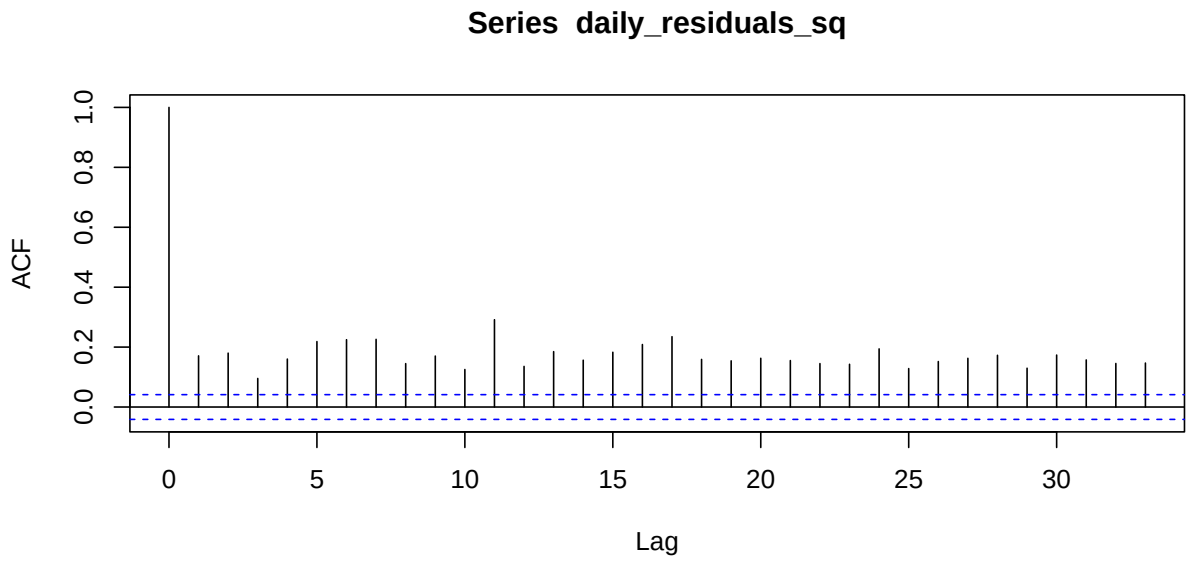
```
daily_residuals <- daily_arima$residuals
daily_residuals_sq <- daily_residuals ^ 2

weekly_residuals <- weekly_arima$residuals
weekly_residuals_sq <- weekly_residuals ^ 2

acf(daily_residuals)
```



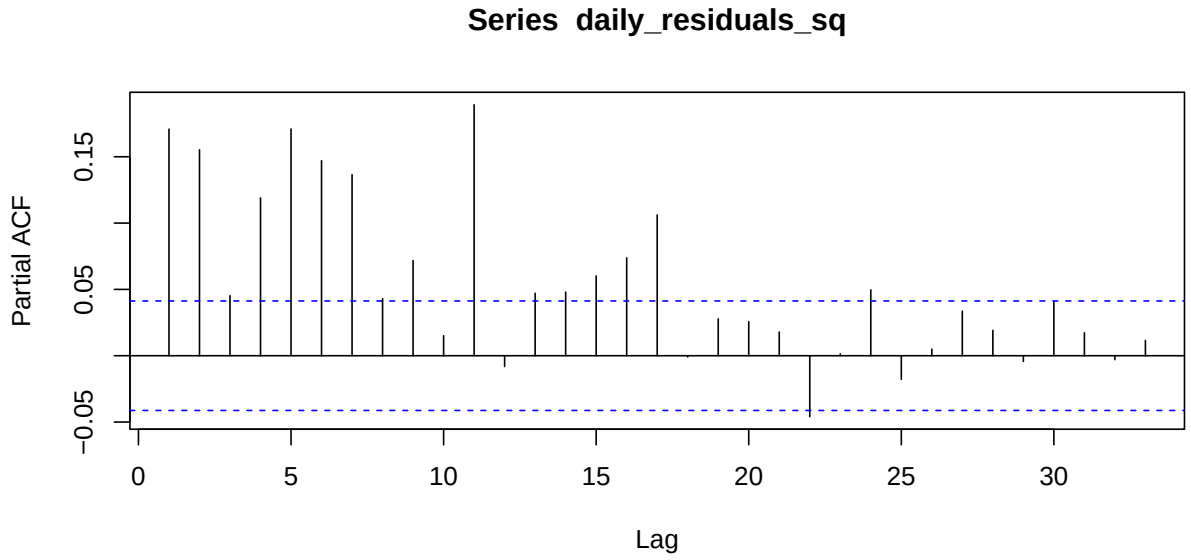

```
pacf(daily_residuals)
```



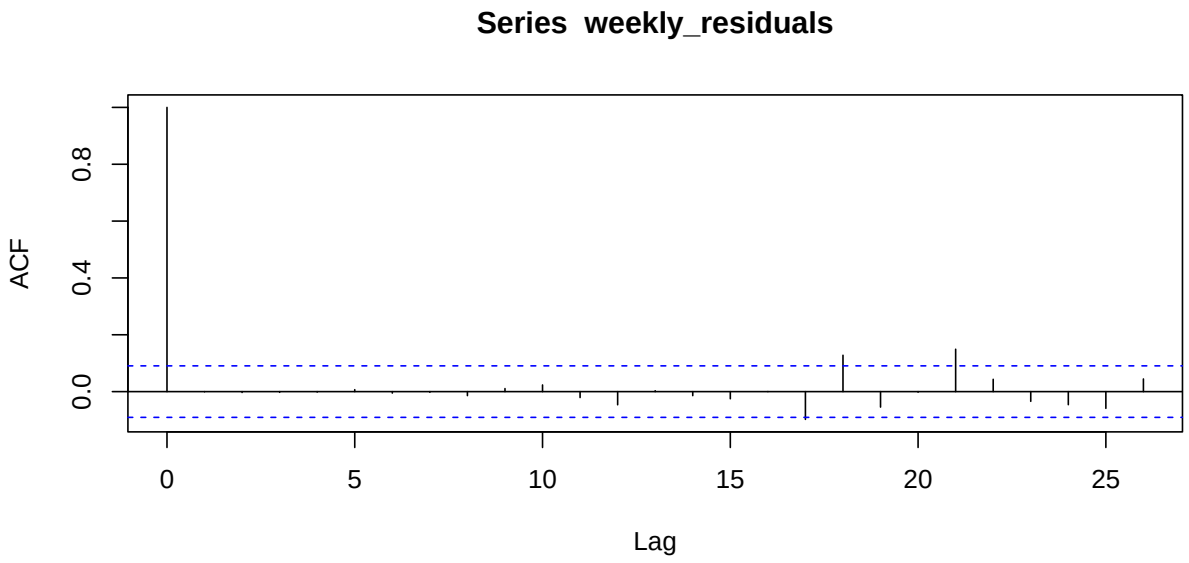

```
acf(daily_residuals_sq)
```



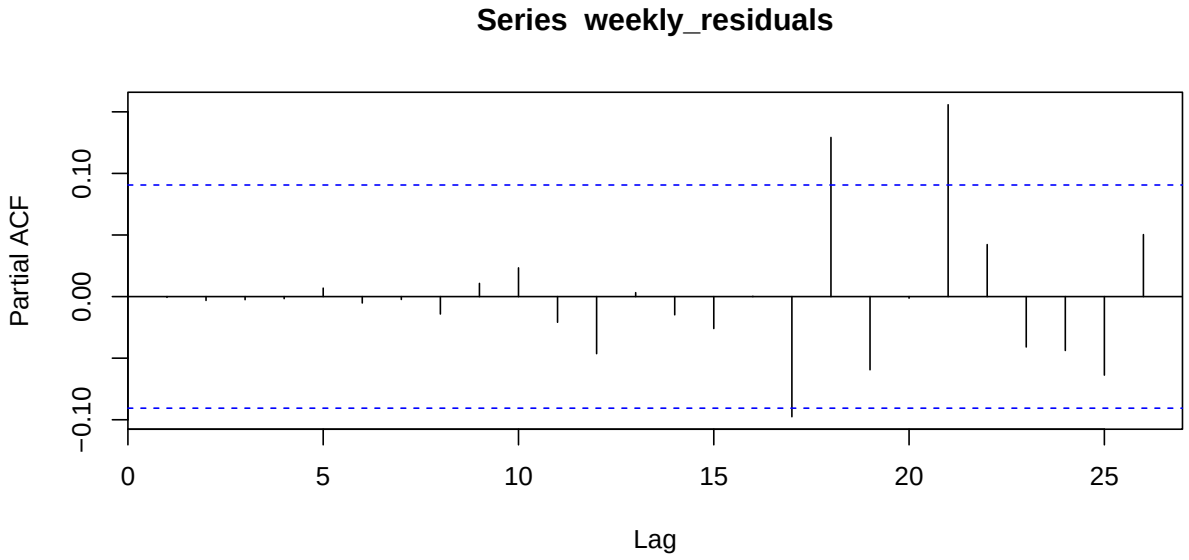

```
pacf(daily_residuals_sq)
```



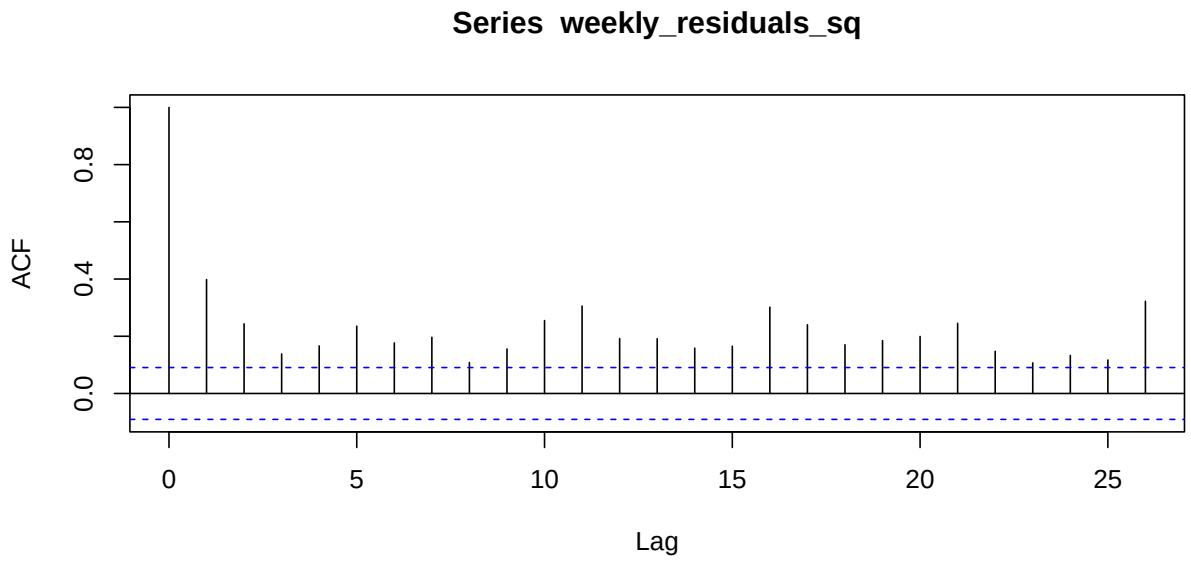

```
acf(weekly_residuals)
```



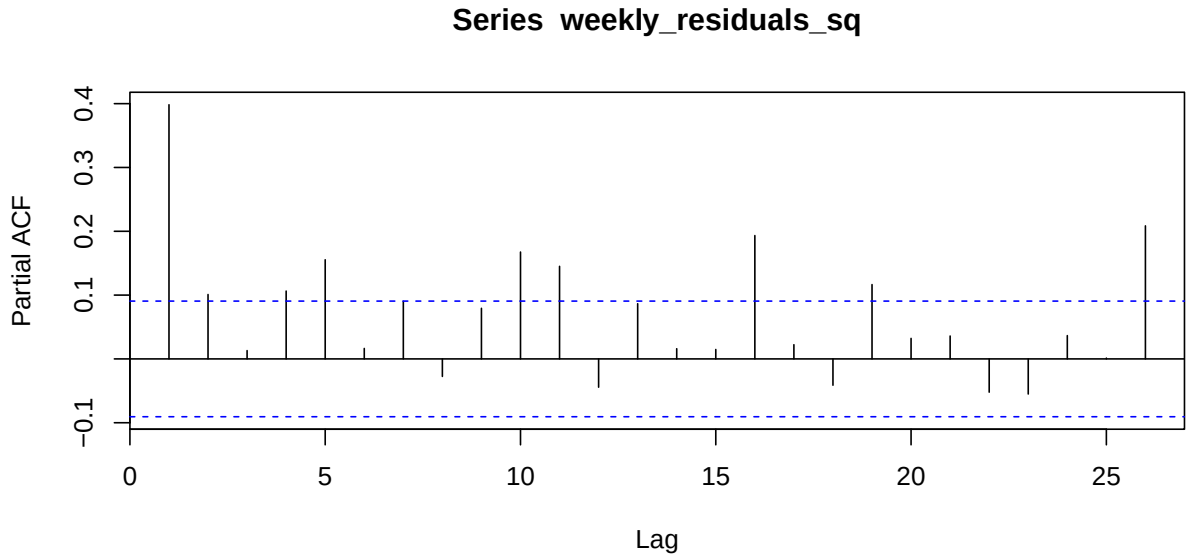

```
pacf(weekly_residuals)
```




```
acf(weekly_residuals_sq)
```




```
pacf(weekly_residuals_sq)
```




```
Box.test(daily_residuals,lag=18,type='Ljung',fitdf=17)
```

```
##
##  Box-Ljung test
##
## data:  daily_residuals
## X-squared = 30.745, df = 1, p-value = 2.942e-08
```

```r
Box.test(daily_residuals_sq,lag=18,type='Ljung',fitdf=17)
```

```
##
##  Box-Ljung test
##
## data:  daily_residuals_sq
## X-squared = 1429.3, df = 1, p-value < 2.2e-16
```

```r
Box.test(weekly_residuals,lag=18,type='Ljung',fitdf=17)
```

```
##
##  Box-Ljung test
##
## data:  weekly_residuals
## X-squared = 14.792, df = 1, p-value = 0.0001201
```

```r
Box.test(weekly_residuals_sq,lag=18,type='Ljung',fitdf=17)
```

```
##
##  Box-Ljung test
##
## data:  weekly_residuals_sq
## X-squared = 425.53, df = 1, p-value < 2.2e-16
```

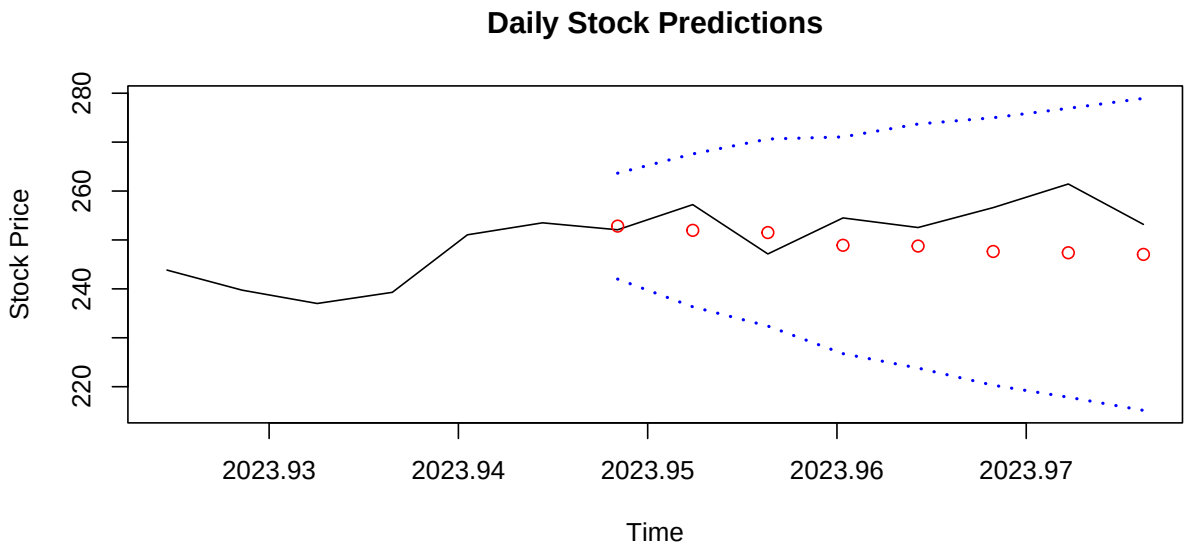
Response:ARIMA residual analysis for the Weekly and Monthly Data

For the daily data, the acf plots of the residuals resembles that of white noise, but the acf of the squared residuals does not. This means that the time series may plausibly be uncorrelated, but not independent. The same is true for the residuals and squared residuals of the weekly data set.

For the hypothesis tests, the p-values in all cases are very small (including the squared residuals), which indicates that the residuals are correlated.

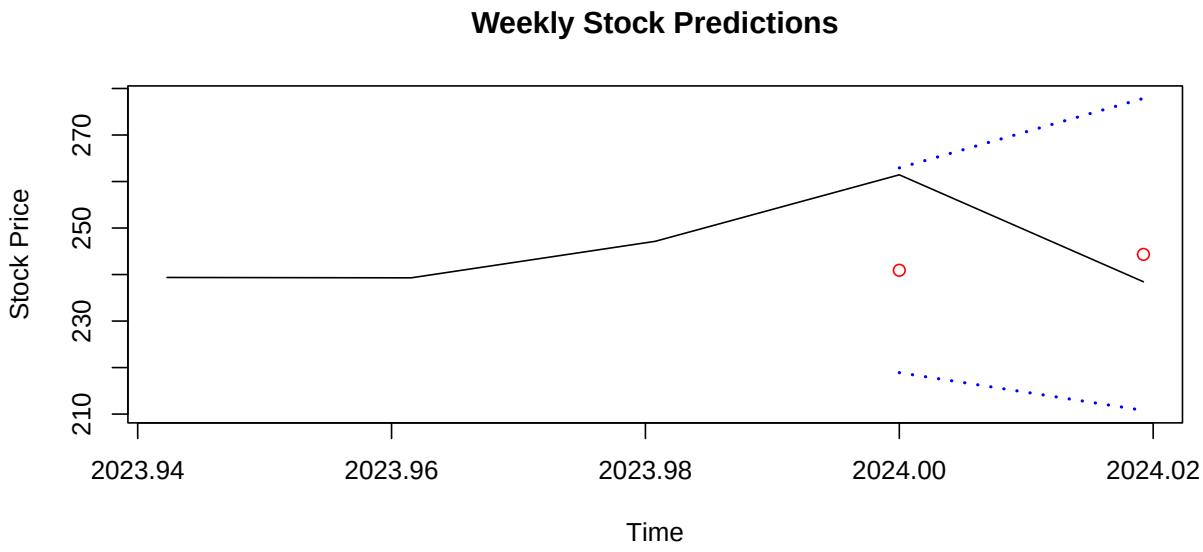
2c. Apply the models identified in (2a) and forecast the last two weeks of data using both daily and weekly data. Plot the predicted data to compare the predicted values to the actual observed ones. Include 95% confidence intervals for the forecasts in the corresponding plots.

```r
n = length(TDaily); nfit = n-8
TDaily_pred = as.vector(predict(daily_arima, n.ahead=8))

time_vals = time(TDaily)
ubound = TDaily_pred$pred+1.96*TDaily_pred$se
lbound = TDaily_pred$pred-1.96*TDaily_pred$se
ymin=min(lbound)
ymax=max(ubound)
plot(time_vals[2250:n], TDaily[2250:n], type="l", ylim=c(ymin,ymax), xlab="Time", ylab="Stock Price", ma
points(time_vals[(nfit+1):n], TDaily_pred$pred, col="red")
lines(time_vals[(nfit+1):n], ubound, lty=3, lwd=2, col="blue")
lines(time_vals[(nfit+1):n], lbound, lty=3, lwd=2, col="blue")
```



Here are the predictions for the Weekly Dataset.

```
n = length(TWeekly); nfit = n-2
TWeekly_pred = as.vector(predict(weekly_arima, n.ahead=2))

time_vals = time(TWeekly)
ubound = TWeekly_pred$pred+1.96*TWeekly_pred$se
lbound = TWeekly_pred$pred-1.96*TWeekly_pred$se
ymin=min(lbound)
ymax=max(ubound)
plot(time_vals[466:n], TWeekly[466:n], type="l", ylim=c(ymin,ymax), xlab="Time", ylab="Stock Price", ma
points(time_vals[(nfit+1):n], TWeekly_pred$pred, col="red")
lines(time_vals[(nfit+1):n], ubound, lty=3, lwd=2, col="blue")
lines(time_vals[(nfit+1):n], lbound, lty=3, lwd=2, col="blue")
```



Response: Predictions Both models fit the data well, and both are within the 95% confidence band. For the weekly model, the first test value is very close to the band, but still within.

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.

```
cat("MAPE Daily:",mean(abs((TDaily.test-as.numeric(TDaily_pred$pred))/TDaily.test)))
```

```
## MAPE Daily: 0.02387378
```

```
cat("\nPM Daily:",sum((as.numeric(TDaily_pred$pred) - TDaily.test)^2)/sum((TDaily.test-mean(TDaily.test
```

```
##
## PM Daily: 3.257798
```

```
cat("\nMAPE Weekly:",mean(abs((TWeekly.test-as.numeric(TWeekly_pred$pred))/TWeekly.test)))
```

```
##
## MAPE Weekly: 0.05158308
```

```
cat("\nPM Weekly:",sum((as.numeric(TWeekly_pred$pred) - TWeekly.test)^2)/sum((TWeekly.test-mean(TWeekly
```

```
##
## PM Weekly: 1.724614
```

Response: Prediction Comparison

The MAPE values show that the daily arima model has a better performance than the weekly arima model. The precision measure values, however, show that the weekly model has a better performance, as 3.25 is much farther from a value of 1, indicating that the variance of the model predictions is not the same as the variance of the actual values.

As explained in the previous question, the daily model has all predictions very close to the test points, with all predictions within the confidence band. The same is true for the weekly predictions, however the first prediction is very close to the confidence band (but both data points are still within the confidence band).

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

3a. Divide the data into training and testing data set, where the training data exclude the last two weeks of data : December 21th-December 28th for weekly data, and December 18th-December 28th for daily data. Apply the iterative model to fit an ARMA(p,q)-GARCH(m,n) model by selecting the orders for p & q up to 5 and orders for m & n up to 2. Display the summary of the final model fit. Write up the equation of the estimated model.

```
TDaily_returns <- ts(TDaily_returns,start=c(2015),freq=252)
TWeekly_returns <- ts(TWeekly_returns,start=c(2015),freq=52)

n<-length(TDaily_returns)
TDaily_returns.train<-TDaily_returns[1:(n-8)]
TDaily_returns.test<-TDaily_returns[(n-7):n]

n2<-length(TWeekly_returns)
TWeekly_returns.train<-TWeekly_returns[1:(n2-2)]
TWeekly_returns.test<-TWeekly_returns[(n2-1):n2]

# Step 1: Find p0, q0


allorders2 <- function(ts_data, max_p, max_d, max_q){
  orders = data.frame()
  for (p in 0:max_p){
    for (d in 0:max_d){
      for (q in 0:max_q) {
        possibleError <- tryCatch({
          mod = arima(ts_data, order=c(p,d,q), method="ML")
          current.aic = AIC(mod)
          orders<-rbind(orders,c(p,d,q,current.aic))
        },
          error=function(e) e
        )
        if(inherits(possibleError, "error")) next

      }
    }
  }
  names(orders) <- c("p","d","q","AIC")
  return(orders[order(orders$AIC),])
}

head(allorders2(TDaily_returns.train,5,0,5), 5)
```

```
##    p d q       AIC
## 36 5 0 5 -8647.705
## 21 3 0 2 -8645.278
## 16 2 0 3 -8645.258
## 22 3 0 3 -8642.803
## 17 2 0 4 -8642.451
```

```
head(allorders2(TWeekly_returns.train,5,0,5), 5)
```

```
##    p d q       AIC
## 15 2 0 2 -1027.342
## 21 3 0 2 -1025.370
## 16 2 0 3 -1025.368
## 24 3 0 5 -1023.979
## 34 5 0 3 -1023.965
```

For the first step, p0=q0=5 for the daily data, and p0=q0=2 for the weekly data. Now let's move onto step 2.

```
test_modelAGG <- function(m,n){
    spec = ugarchspec(variance.model=list(garchOrder=c(m,n)),
                      mean.model=list(armaOrder=c(5,5),
                                      include.mean=T),
                      distribution.model="std")
    fit = ugarchfit(spec, TDaily_returns.train, solver = 'hybrid')
    current.bic = infocriteria(fit)[2]
    df = data.frame(m,n,current.bic)
    names(df) <- c("m","n","BIC")
    print(paste(m,n,current.bic,sep=" "))
    return(df)
}

ordersAGG = data.frame(Inf,Inf,Inf)
names(ordersAGG) <- c("m","n","BIC")

for (m in 0:2){
    for (n in 0:2){
        possibleError <- tryCatch(
            ordersAGG<-rbind(ordersAGG,test_modelAGG(m,n)),
            error=function(e) e
        )
        if(inherits(possibleError, "error")) next
    }
}
```

```
## [1] "0 1 -4.00419337967139"
## [1] "0 2 -3.99867281041785"
## [1] "1 0 -3.9785234486711"
## [1] "1 1 -4.05784098447682"
## [1] "1 2 -4.0544157550423"
## [1] "2 0 -4.00035784090125"
## [1] "2 1 -4.05443272081007"
## [1] "2 2 -4.05127517907206"
```

```
ordersAGG <- ordersAGG[order(-ordersAGG$BIC),]
tail(ordersAGG)
```

```
##   m n       BIC
## 7 2 0 -4.000358
```

```
## 2 0 1 -4.004193
## 9 2 2 -4.051275
## 6 1 2 -4.054416
## 8 2 1 -4.054433
## 5 1 1 -4.057841
```

For the Daily Data, m0=n0=1. Let's try for the Weekly Data.

```
test_modelAGG2 <- function(m,n){
    spec = ugarchspec(variance.model=list(garchOrder=c(m,n)),
                      mean.model=list(armaOrder=c(2,2),
                                      include.mean=T),
                      distribution.model="std")
    fit = ugarchfit(spec, TWeekly_returns.train, solver = 'hybrid')
    current.bic = infocriteria(fit)[2]
    df = data.frame(m,n,current.bic)
    names(df) <- c("m","n","BIC")
    print(paste(m,n,current.bic,sep=" "))
    return(df)
}

ordersAGG2 = data.frame(Inf,Inf,Inf)
names(ordersAGG2) <- c("m","n","BIC")

for (m in 0:2){
    for (n in 0:2){
        possibleError <- tryCatch(
            ordersAGG2<-rbind(ordersAGG2,test_modelAGG2(m,n)),
            error=function(e) e
        )
        if(inherits(possibleError, "error")) next
    }
}
```

```
## [1] "0 0 -2.28391467900842"
## [1] "0 1 -2.28438053786444"
## [1] "0 2 -2.28212354941323"
## [1] "1 0 -2.25553173905563"
## [1] "1 1 -2.32130701419349"
## [1] "1 2 -2.32678515081623"
## [1] "2 0 -2.29645040908971"
## [1] "2 1 -2.32780082553996"
## [1] "2 2 -2.31362384770804"
```

```
ordersAGG2 <- ordersAGG2[order(-ordersAGG2$BIC),]
tail(ordersAGG2)
```

```
##    m n       BIC
## 3  0 1 -2.284381
## 8  2 0 -2.296450
## 10 2 2 -2.313624
## 6  1 1 -2.321307
## 7  1 2 -2.326785
## 9  2 1 -2.327801
```

For the weekly data, it is m0=2, and n0=1. Let's move on to step 3 with the daily data.

```
# Step 3:

test_modelAGA <- function(p,q){
    spec = ugarchspec(variance.model=list(garchOrder=c(1,1)),
                      mean.model=list(armaOrder=c(p,q),
                                      include.mean=T),
                      distribution.model="std")
    fit = ugarchfit(spec, TDaily_returns.train, solver = 'hybrid')
    current.bic = infocriteria(fit)[2]
    df = data.frame(p,q,current.bic)
    names(df) <- c("p","q","BIC")
    print(paste(p,q,current.bic,sep=" "))
    return(df)
}

ordersAGA = data.frame(Inf,Inf,Inf)
names(ordersAGA) <- c("p","q","BIC")
for (p in 0:5){
    for (q in 0:5){
        possibleError <- tryCatch(
            ordersAGA<-rbind(ordersAGA,test_modelAGA(p,q)),
            error=function(e) e
        )
        if(inherits(possibleError, "error")) next
    }
}
```

```
## [1] "0 0 -4.08257210356743"
## [1] "0 1 -4.07916283413759"
## [1] "0 2 -4.07603583192548"
## [1] "0 3 -4.07282971996947"
## [1] "0 4 -4.06959740871936"
## [1] "0 5 -4.06667848433339"
## [1] "1 0 -4.07916334036512"
## [1] "1 1 -4.07635282210768"
## [1] "1 2 -4.07321931667984"
## [1] "1 3 -4.0698415957357"
## [1] "1 4 -4.06644789734701"
## [1] "1 5 -4.06327775200391"
## [1] "2 0 -4.07604514100433"
## [1] "2 1 -4.07321087304172"
## [1] "2 2 -4.07006550573873"
## [1] "2 3 -4.06981067909478"
## [1] "2 4 -4.06543180290022"
## [1] "2 5 -4.0657208243899"
## [1] "3 0 -4.07286426608618"
## [1] "3 1 -4.06981739418591"
## [1] "3 2 -4.07193611510899"
## [1] "3 3 -4.06873618263146"
## [1] "3 4 -4.06543375817342"
## [1] "3 5 -4.06266847559809"
## [1] "4 0 -4.06960517951526"
```

```
## [1] "4 1 -4.0664053526716"
## [1] "4 2 -4.0651253466954"
## [1] "4 3 -4.0654383162024"
## [1] "4 4 -4.06452917577275"
## [1] "4 5 -4.06127955594074"
## [1] "5 0 -4.06658991594317"
## [1] "5 1 -4.0631858239262"
## [1] "5 2 -4.06572334520097"
## [1] "5 3 -4.06276812714617"
## [1] "5 4 -4.05914801626537"
## [1] "5 5 -4.05784098447682"
```

```
ordersAGA <- ordersAGA[order(-ordersAGA$BIC),]
tail(ordersAGA)
```

```
##    p q       BIC
## 4  0 2 -4.076036
## 14 2 0 -4.076045
## 9  1 1 -4.076353
## 3  0 1 -4.079163
## 8  1 0 -4.079163
## 2  0 0 -4.082572
```

For the daily data, p1=1 and q1 = 0 (0,0 is the trivial solution). Let's try the Weekly data.

```
test_modelAGA <- function(p,q){
    spec = ugarchspec(variance.model=list(garchOrder=c(2,1)),
                      mean.model=list(armaOrder=c(p,q),
                                      include.mean=T),
                      distribution.model="std")
    fit = ugarchfit(spec, TWeekly_returns.train, solver = 'hybrid')
    current.bic = infocriteria(fit)[2]
    df = data.frame(p,q,current.bic)
    names(df) <- c("p","q","BIC")
    print(paste(p,q,current.bic,sep=" "))
    return(df)
}

ordersAGA = data.frame(Inf,Inf,Inf)
names(ordersAGA) <- c("p","q","BIC")
for (p in 0:5){
    for (q in 0:5){
        possibleError <- tryCatch(
            ordersAGA<-rbind(ordersAGA,test_modelAGA(p,q)),
            error=function(e) e
        )
        if(inherits(possibleError, "error")) next
    }
}
```

```
## [1] "0 0 -2.32001141551381"
## [1] "0 1 -2.30938957967975"
## [1] "0 2 -2.29701411625398"
```

```
## [1] "0 3 -2.28334211571415"
## [1] "0 4 -2.27604597380907"
## [1] "0 5 -2.26311362779761"
## [1] "1 0 -2.30951600883453"
## [1] "1 1 -2.2965499038335"
## [1] "1 2 -2.28388182665485"
## [1] "1 3 -2.27119588405673"
## [1] "1 4 -2.26290857747377"
## [1] "1 5 -2.2500079045418"
## [1] "2 0 -2.29692261149294"
## [1] "2 1 -2.28383294227495"
## [1] "2 2 -2.32780082553996"
## [1] "2 3 -2.30270431460244"
## [1] "2 4 -2.2967418852044"
## [1] "2 5 -2.23775479304791"
## [1] "3 0 -2.28363104153202"
## [1] "3 1 -2.27071932687223"
## [1] "3 2 -2.29000141961953"
## [1] "3 3 -2.28289243303475"
## [1] "3 4 -2.30346120989499"
## [1] "3 5 -2.2355392810857"
## [1] "4 0 -2.27507144649972"
## [1] "4 1 -2.26203000189354"
## [1] "4 2 -2.27884897542991"
## [1] "4 3 -2.30956288122166"
## [1] "4 4 -2.23471790064918"
## [1] "4 5 -2.22037490612458"
## [1] "5 0 -2.26305325250559"
## [1] "5 1 -2.25052164165049"
## [1] "5 2 -2.23749112717001"
## [1] "5 3 -2.28132664521813"
## [1] "5 4 -2.22184632462952"
## [1] "5 5 -2.20973868934507"
```

```
ordersAGA <- ordersAGA[order(-ordersAGA$BIC),]
tail(ordersAGA)
```

```
##    p q       BIC
## 24 3 4 -2.303461
## 3  0 1 -2.309390
## 8  1 0 -2.309516
## 29 4 3 -2.309563
## 2  0 0 -2.320011
## 16 2 2 -2.327801
```

For the weekly data, p1=q1=2. Let's move on to step 4 (last step) with the daily data.

```
test_modelAGG <- function(m,n){
    spec = ugarchspec(variance.model=list(garchOrder=c(m,n)),
                      mean.model=list(armaOrder=c(1,0),
                                      include.mean=T), distribution.model="std")
    fit = ugarchfit(spec, TDaily_returns.train, solver = 'hybrid')
    current.bic = infocriteria(fit)[2]
```

```
    df = data.frame(m,n,current.bic)
    names(df) <- c("m","n","BIC")
    print(paste(m,n,current.bic,sep=" "))
    return(df)
}

ordersAGG = data.frame(Inf,Inf,Inf)
names(ordersAGG) <- c("m","n","BIC")

for (m in 0:2){
    for (n in 0:2){
        possibleError <- tryCatch(
            ordersAGG<-rbind(ordersAGG,test_modelAGG(m,n)),
            error=function(e) e
        )
        if(inherits(possibleError, "error")) next
    }
}
```

```
## [1] "0 0 -3.99899367137056"
## [1] "0 1 -4.01946902773143"
## [1] "0 2 -4.01920393389086"
## [1] "1 0 -4.00939890857329"
## [1] "1 1 -4.07916334036512"
## [1] "1 2 -4.07580427838264"
## [1] "2 0 -4.02313837420785"
## [1] "2 1 -4.07580640135874"
## [1] "2 2 -4.07240821549934"
```

```
ordersAGG <- ordersAGG[order(-ordersAGG$BIC),]
tail(ordersAGG)
```

```
##    m n       BIC
## 3  0 1 -4.019469
## 8  2 0 -4.023138
## 10 2 2 -4.072408
## 7  1 2 -4.075804
## 9  2 1 -4.075806
## 6  1 1 -4.079163
```

The final daily model is p=1, q=0, m=1, n=1. Let's find the final weekly model.

```
test_modelAGG <- function(m,n){
    spec = ugarchspec(variance.model=list(garchOrder=c(m,n)),
                      mean.model=list(armaOrder=c(2,2),
                                      include.mean=T), distribution.model="std")
    fit = ugarchfit(spec, TWeekly_returns.train, solver = 'hybrid')
    current.bic = infocriteria(fit)[2]
    df = data.frame(m,n,current.bic)
    names(df) <- c("m","n","BIC")
    print(paste(m,n,current.bic,sep=" "))
    return(df)
```

```
}

ordersAGG = data.frame(Inf,Inf,Inf)
names(ordersAGG) <- c("m","n","BIC")

for (m in 0:2){
    for (n in 0:2){
        possibleError <- tryCatch(
            ordersAGG<-rbind(ordersAGG,test_modelAGG(m,n)),
            error=function(e) e
        )
        if(inherits(possibleError, "error")) next
    }
}
```

```
## [1] "0 0 -2.28391467900842"
## [1] "0 1 -2.28438053786444"
## [1] "0 2 -2.28212354941323"
## [1] "1 0 -2.25553173905563"
## [1] "1 1 -2.32130701419349"
## [1] "1 2 -2.32678515081623"
## [1] "2 0 -2.29645040908971"
## [1] "2 1 -2.32780082553996"
## [1] "2 2 -2.31362384770804"
```

```
ordersAGG <- ordersAGG[order(-ordersAGG$BIC),]
tail(ordersAGG)
```

```
##    m n       BIC
## 3  0 1 -2.284381
## 8  2 0 -2.296450
## 10 2 2 -2.313624
## 6  1 1 -2.321307
## 7  1 2 -2.326785
## 9  2 1 -2.327801
```

The final weekly model is p=2, q=2, m=2, n=1. Let's find the summary of the final models.

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

```
##
## *---------------------------------*
## *          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      0.001632    0.000554   2.94356 0.003245
## ar1    -0.003732    0.019455  -0.19181 0.847888
## omega   0.000009    0.000005   1.77231 0.076343
## alpha1  0.043394    0.007848   5.52968 0.000000
## beta1   0.952568    0.005631 169.15775 0.000000
## shape   4.151674    0.400868  10.35671 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error   t value Pr(>|t|)
## mu      0.001632    0.000587   2.78171 0.005407
## ar1    -0.003732    0.017367  -0.21487 0.829868
## omega   0.000009    0.000009   1.00181 0.316436
## alpha1  0.043394    0.011830   3.66809 0.000244
## beta1   0.952568    0.003945 241.49172 0.000000
## shape   4.151674    0.420198   9.88028 0.000000
##
## LogLikelihood : 4620.378
##
## Information Criteria
## ------------------------------------
##
## Akaike       -4.0944
## Bayes        -4.0792
## Shibata      -4.0944
## Hannan-Quinn -4.0888
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      1.059  0.3035
## Lag[2*(p+q)+(p+q)-1][2]     1.940  0.2344
## Lag[4*(p+q)+(p+q)-1][5]     3.016  0.4274
## d.o.f=1
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.8958  0.3439
## Lag[2*(p+q)+(p+q)-1][5]    2.3057  0.5482
## Lag[4*(p+q)+(p+q)-1][9]    3.2819  0.7117
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.3079 0.500 2.000  0.5790
## ARCH Lag[5]    0.3320 1.440 1.667  0.9316
```

```
## ARCH Lag[7]    1.2208 2.315 1.543  0.8752
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  2.9451
## Individual Statistics:
## mu     0.1740
## ar1    0.0721
## omega  0.1199
## alpha1 0.4460
## beta1  0.3445
## shape  0.4554
##
## 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           0.6597 0.5095
## Negative Sign Bias  0.8272 0.4082
## Positive Sign Bias  0.8405 0.4007
## Joint Effect        1.4402 0.6961
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     27.60      0.09149
## 2    30     30.98      0.36647
## 3    40     48.08      0.15115
## 4    50     52.26      0.34868
##
##
## Elapsed time : 0.4079142
```

Here is the model summary for the weekly model.

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

weekly_fit
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## ---------------------------------
```

```
## GARCH Model  : sGARCH(2,1)
## Mean Model   : ARFIMA(2,0,2)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error   t value Pr(>|t|)
## mu      0.005862    0.005895   0.99438 0.320038
## ar1     0.056316    0.027541   2.04484 0.040871
## ar2    -0.925782    0.089839 -10.30495 0.000000
## ma1    -0.024085    0.001681 -14.32464 0.000000
## ma2     1.019217    0.003058 333.34647 0.000000
## omega   0.000243    0.000225   1.07915 0.280519
## alpha1  0.092466    0.056188   1.64565 0.099836
## alpha2  0.000000    0.140081   0.00000 1.000000
## beta1   0.869805    0.068657  12.66883 0.000000
## shape   5.791321    1.550249   3.73574 0.000187
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.005862    0.042562  0.13772 0.890462
## ar1     0.056316    0.231374  0.24340 0.807696
## ar2    -0.925782    0.756829 -1.22324 0.221240
## ma1    -0.024085    0.011592 -2.07775 0.037733
## ma2     1.019217    0.025874 39.39228 0.000000
## omega   0.000243    0.001430  0.17002 0.864997
## alpha1  0.092466    0.158609  0.58298 0.559906
## alpha2  0.000000    1.054181  0.00000 1.000000
## beta1   0.869805    0.370546  2.34736 0.018907
## shape   5.791321    1.491057  3.88404 0.000103
##
## LogLikelihood : 574.2731
##
## Information Criteria
## ------------------------------------
##
## Akaike       -2.4166
## Bayes        -2.3278
## Shibata      -2.4175
## Hannan-Quinn -2.3816
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.3534  0.5522
## Lag[2*(p+q)+(p+q)-1][11]   2.5740  1.0000
## Lag[4*(p+q)+(p+q)-1][19]   4.6728  0.9970
## d.o.f=4
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      3.123  0.0772
```

```
## Lag[2*(p+q)+(p+q)-1][8]      5.149  0.3329
## Lag[4*(p+q)+(p+q)-1][14]     8.304  0.3505
## d.o.f=3
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[4]    0.2526 0.500 2.000  0.6152
## ARCH Lag[6]    0.4897 1.461 1.711  0.8949
## ARCH Lag[8]    2.6402 2.368 1.583  0.6127
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  2.0219
## Individual Statistics:
## mu     0.02721
## ar1    0.07129
## ar2    0.06300
## ma1    0.04150
## ma2    0.08151
## omega  0.26211
## alpha1 0.16628
## alpha2 0.26253
## beta1  0.27760
## shape  0.12868
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          2.29 2.54 3.05
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value    prob sig
## Sign Bias          0.17855 0.85837
## Negative Sign Bias 2.28154 0.02297  **
## Positive Sign Bias 0.06413 0.94890
## Joint Effect       7.41574 0.05976   *
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     13.56       0.8089
## 2    30     20.00       0.8929
## 3    40     26.62       0.9342
## 4    50     38.67       0.8550
##
##
## Elapsed time : 0.2740042
```

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

The outputs of the models are given above, as well as their coefficient values. Here is the format of the equations.

Daily Dataset:

$$Y_t = \mu + \phi Y_{t-1} + Z_t$$

$$\sigma_t^2 = \psi_0 + \psi_1 Z_{t-1}^2 + \beta_1 \sigma_{t-1}^2$$

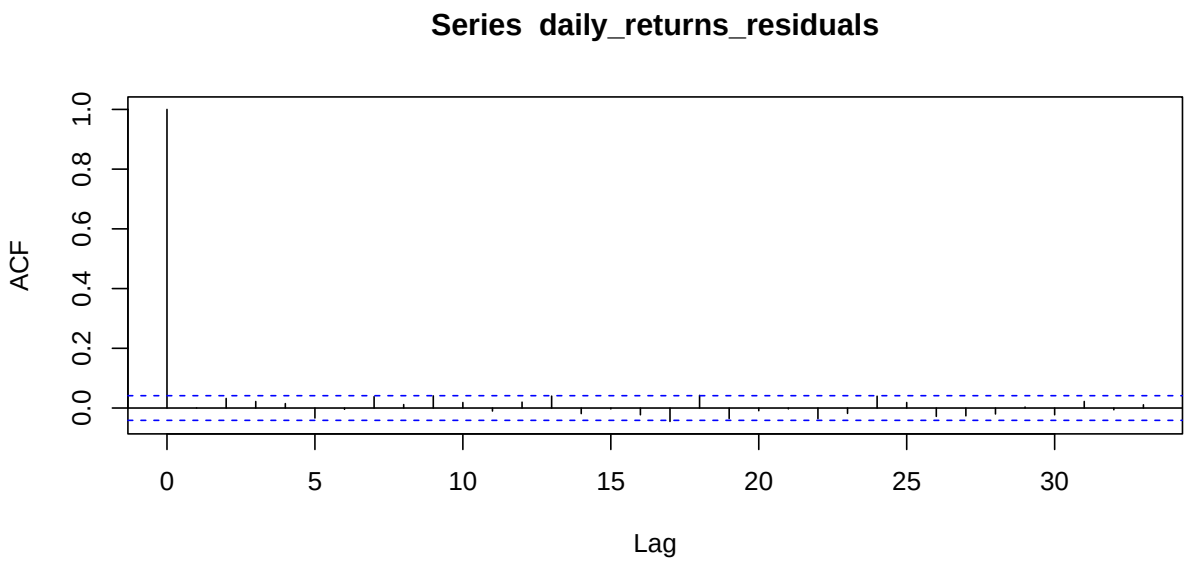
Weekly Dataset:

$$Y_t = \mu + \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + Z_t + \theta_1 Z_{t-1} + \theta_2 Z_{t-2}$$

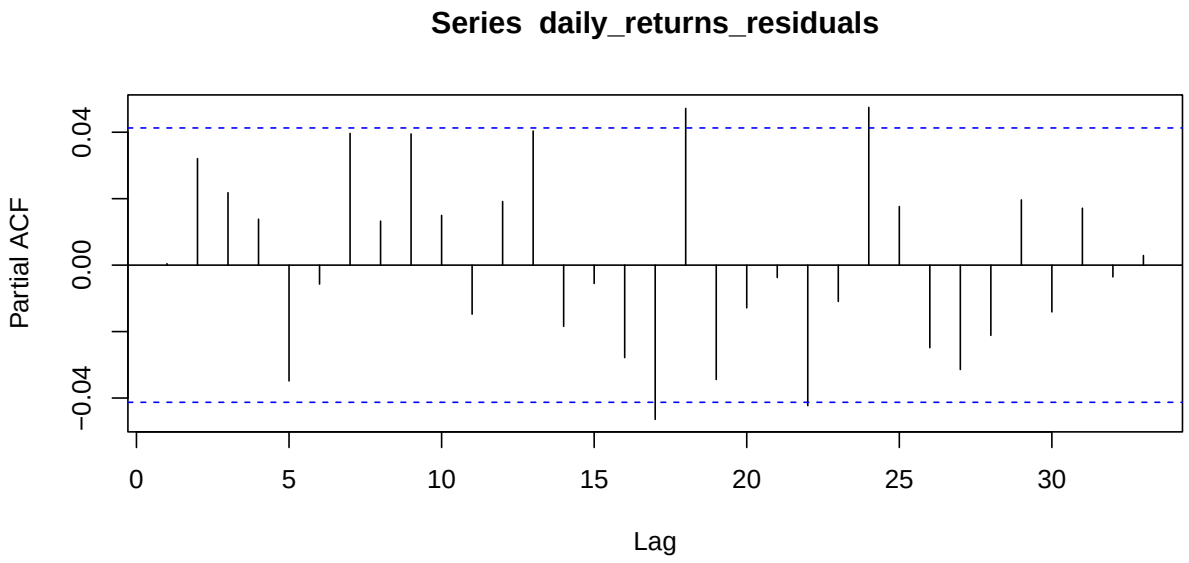
$$\sigma_t^2 = \psi_0 + \psi_1 Z_{t-1}^2 + \psi_2 Z_{t-2}^2 + \beta_1 \sigma_{t-1}^2$$

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?

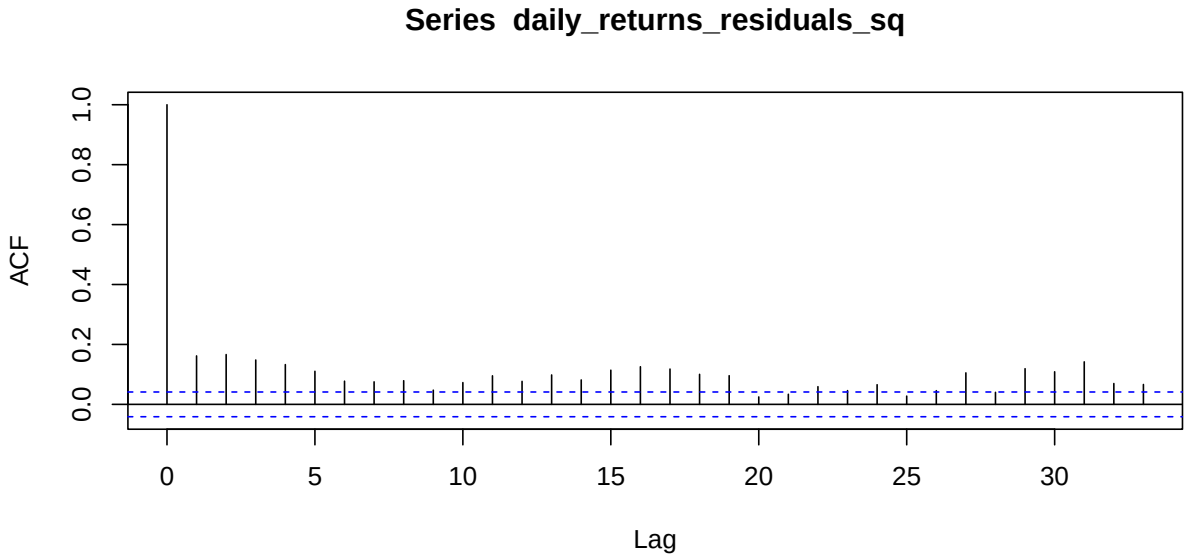
```
daily_returns_residuals <- daily_fit@fit$residuals
daily_returns_residuals_sq <- daily_returns_residuals ^ 2

weekly_returns_residuals <- weekly_fit@fit$residuals
weekly_returns_residuals_sq <- weekly_returns_residuals ^ 2

acf(daily_returns_residuals)
```



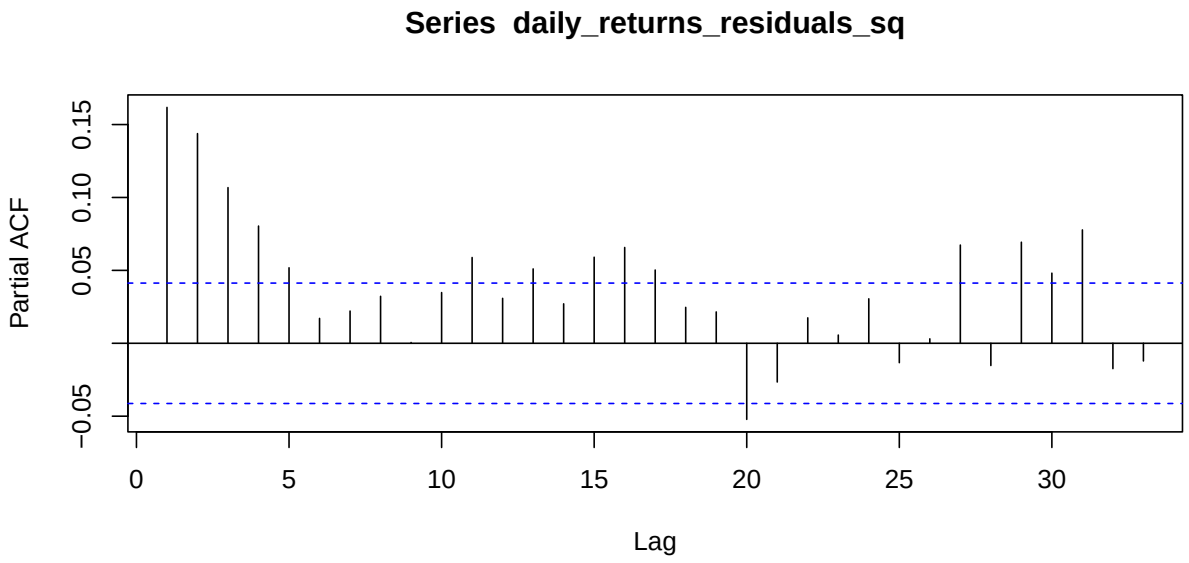

```
pacf(daily_returns_residuals)
```



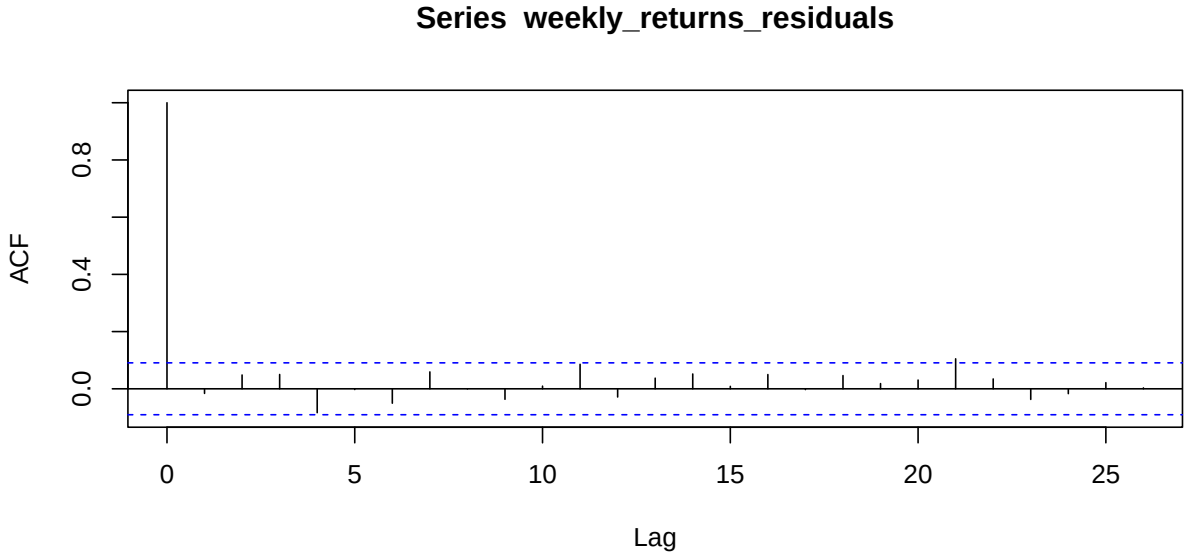

```
acf(daily_returns_residuals_sq)
```



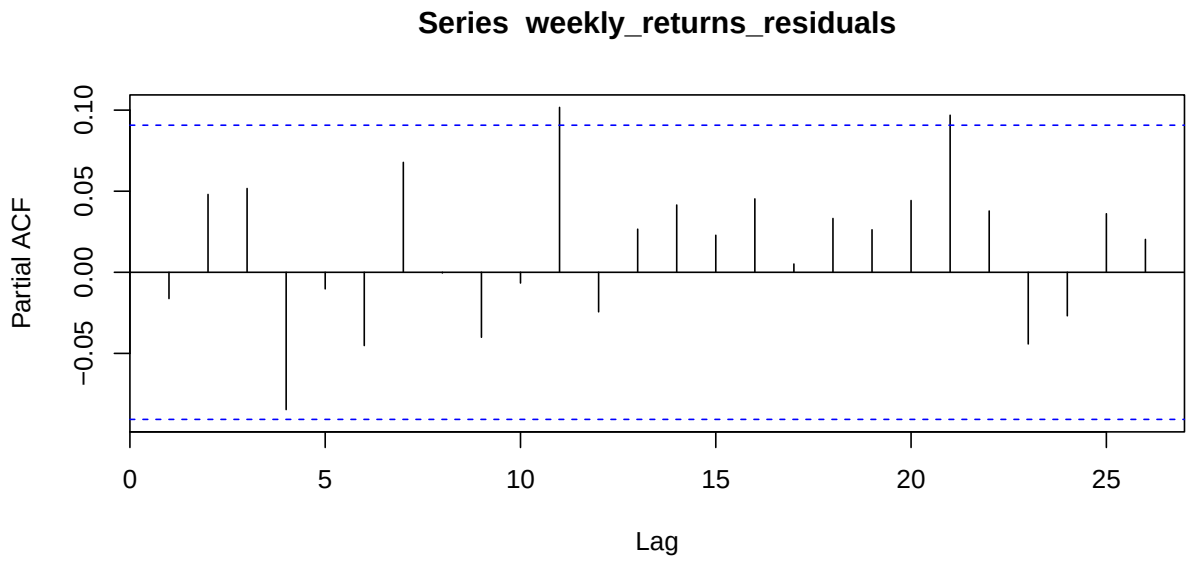

```
pacf(daily_returns_residuals_sq)
```



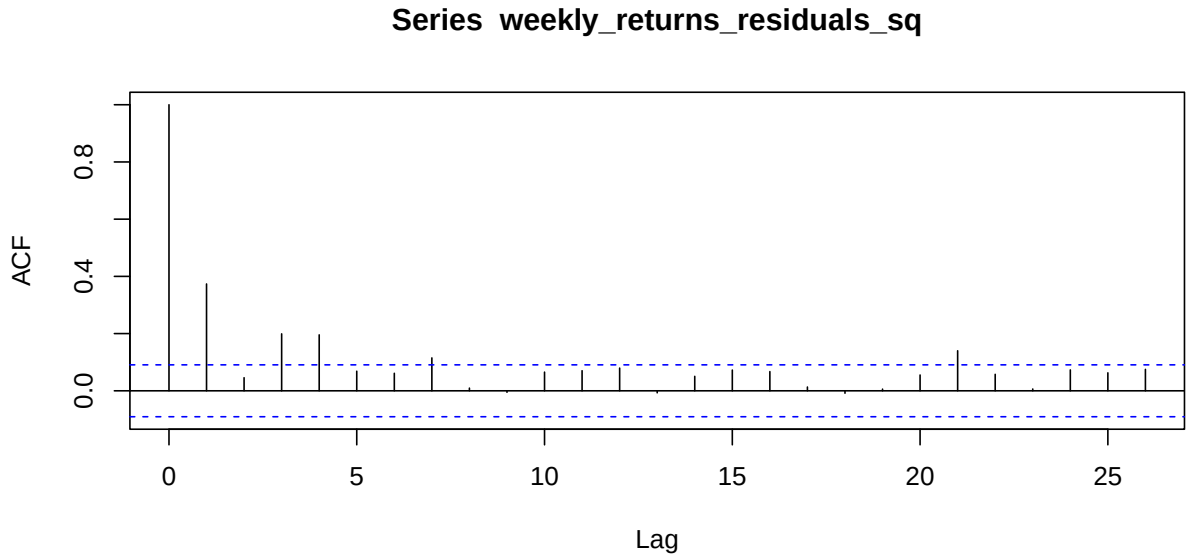

```
acf(weekly_returns_residuals)
```



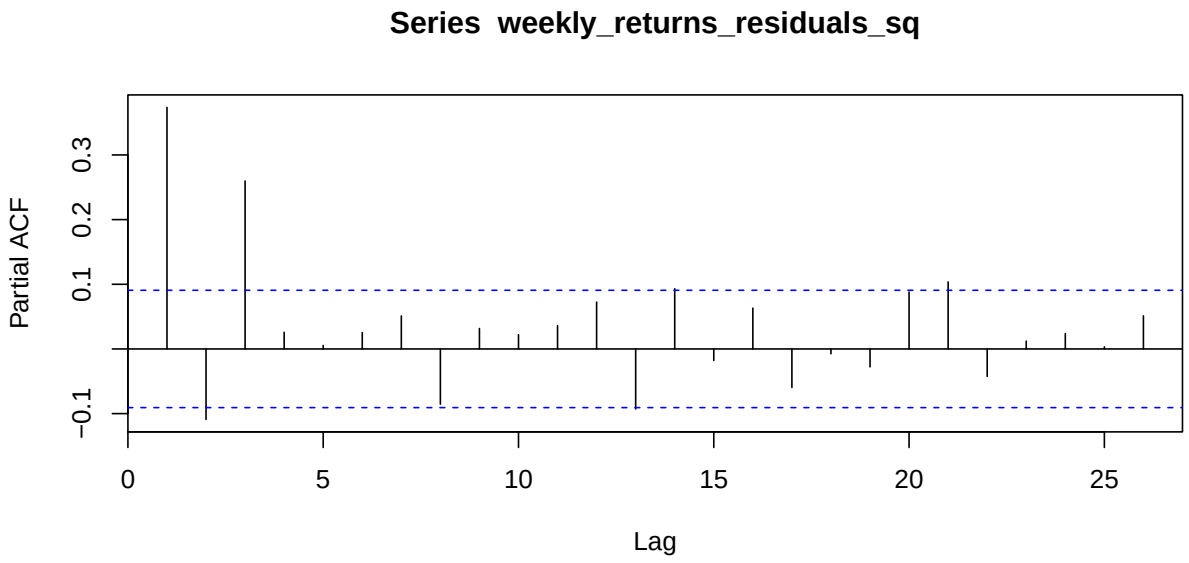

```
pacf(weekly_returns_residuals)
```




```
acf(weekly_returns_residuals_sq)
```




```
pacf(weekly_returns_residuals_sq)
```




```
Box.test(daily_returns_residuals,lag=4,type='Ljung',fitdf=3)
```

```
##
##  Box-Ljung test
##
## data:  daily_returns_residuals
## X-squared = 3.8931, df = 1, p-value = 0.04849
```

```
Box.test(daily_returns_residuals_sq,lag=4,type='Ljung',fitdf=3)
```

```
##
##  Box-Ljung test
##
## data:  daily_returns_residuals_sq
## X-squared = 210.83, df = 1, p-value < 2.2e-16
```

```
Box.test(weekly_returns_residuals,lag=8,type='Ljung',fitdf=7)
```

```
##
##  Box-Ljung test
##
## data:  weekly_returns_residuals
## X-squared = 8.5796, df = 1, p-value = 0.003399
```

```
Box.test(weekly_returns_residuals_sq,lag=8,type='Ljung',fitdf=7)
```

```
##
##  Box-Ljung test
##
## data:  weekly_returns_residuals_sq
## X-squared = 113.7, df = 1, p-value < 2.2e-16
```

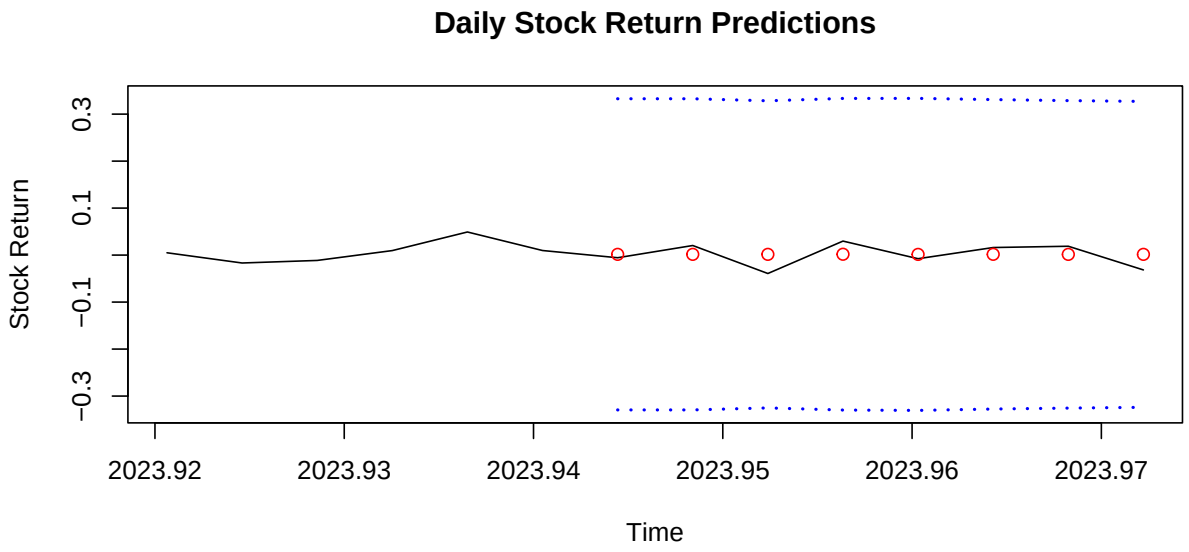
Response

For the daily data, the acf plots of the residuals resembles that of white noise, but the acf of the squared residuals does not. This means that the time series may plausibly be uncorrelated, but not independent. The same is true for the residuals and squared residuals of the weekly data set. Because the squared residuals acfs plots do not resemble that of white noise, it suggests that a higher order GARCH model be be necessary.

For the hypothesis tests, the p-values in all cases are very small (including the squared residuals), which indicates that the residuals are correlated.

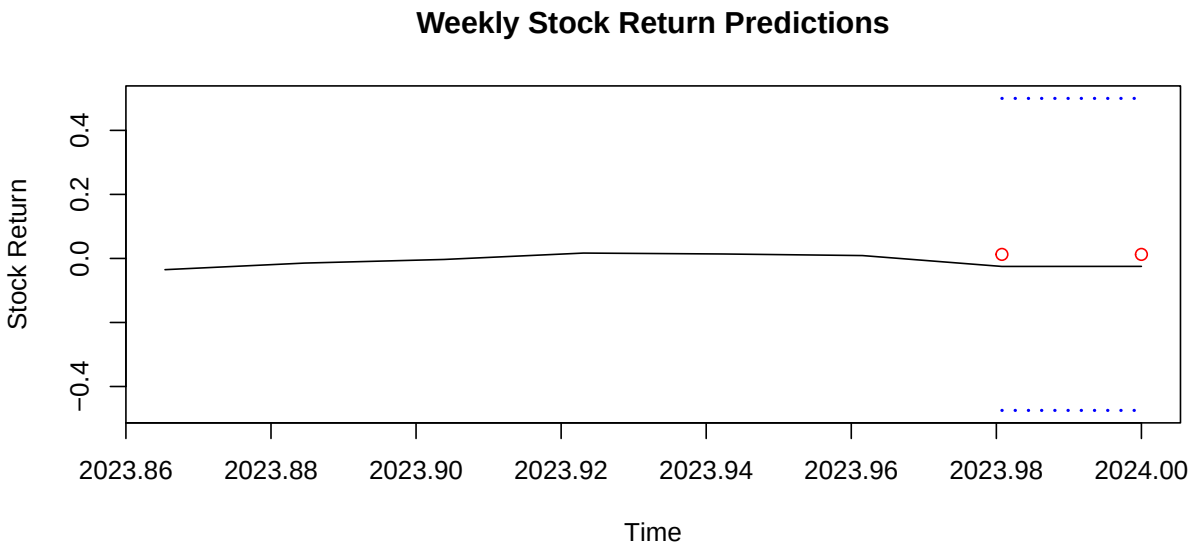
3c. Apply the models identified in (3a) and forecast the mean and the variance of the last two weeks of data. Plot the predicted data to compare the predicted values to the actual observed ones. Include 95% confidence intervals for the forecasts for the mean only in the corresponding plots. Interpret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

```
nfore = length(TDaily_returns.test)
fore.series.1 = NULL
fore.sigma.1 = NULL

for(f in 1: nfore){
    ## Fit models
    data = TDaily_returns.train
    if(f>2)
       data = c(TDaily_returns.train,TDaily_returns.test[1:(f-1)])
    final.model.1 = ugarchfit(spec.1, data, solver = 'hybrid')

    ## Forecast
    fore = ugarchforecast(final.model.1, n.ahead=1)
    fore.series.1 = c(fore.series.1, fore@forecast$seriesFor)
    fore.sigma.1 = c(fore.sigma.1, fore@forecast$sigmaFor)

}

n = length(TDaily_returns); nfit = n-8
#TDaily_pred = as.vector(predict(daily_arima, n.ahead=8))

time_vals = time(TDaily_returns)
ubound = fore.series.1+1.96*sqrt(fore.sigma.1)
lbound = fore.series.1-1.96*sqrt(fore.sigma.1)
ymin=min(lbound)
ymax=max(ubound)
plot(time_vals[2249:n], TDaily_returns[2249:n], type="l", ylim=c(ymin,ymax), xlab="Time", ylab="Stock R
points(time_vals[(nfit+1):n], fore.series.1, col="red")
lines(time_vals[(nfit+1):n], ubound, lty=3, lwd=2, col="blue")
lines(time_vals[(nfit+1):n], lbound, lty=3, lwd=2, col="blue")
```



Here are the predictions for the weekly returns.

```
nfore = length(TWeekly_returns.test)
fore.series.2 = NULL
fore.sigma.2 = NULL

for(f in 1: nfore){
    ## Fit models
    data = TWeekly_returns.train
    if(f>2)
       data = c(TWeekly_returns.train,TWeekly_returns.test[1:(f-1)])
    final.model.2 = ugarchfit(spec.2, data, solver = 'hybrid')

    ## Forecast
    fore = ugarchforecast(final.model.2, n.ahead=1)
    fore.series.2 = c(fore.series.2, fore@forecast$seriesFor)
    fore.sigma.2 = c(fore.sigma.2, fore@forecast$sigmaFor)

}

n = length(TWeekly_returns); nfit = n-2
#TDaily_pred = as.vector(predict(daily_arima, n.ahead=8))

time_vals = time(TWeekly_returns)
ubound = fore.series.2+1.96*sqrt(fore.sigma.2)
lbound = fore.series.2-1.96*sqrt(fore.sigma.2)
ymin=min(lbound)
ymax=max(ubound)
plot(time_vals[462:n], TDaily_returns[462:n], type="l", ylim=c(ymin,ymax), xlab="Time", ylab="Stock Retu
points(time_vals[(nfit+1):n], fore.series.2, col="red")
lines(time_vals[(nfit+1):n], ubound, lty=3, lwd=2, col="blue")
lines(time_vals[(nfit+1):n], lbound, lty=3, lwd=2, col="blue")
```



Response: Interpretation of the results

The predictions for both the daily data and the weekly data are very close to the test data points, and all points are within the 95% confidence interval. The accuracy of both prediction models are similar, and there does not appear to be a significant difference between the two.

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.

```
cat("MAPE Daily:",mean(abs((TDaily_returns.test-as.numeric(fore.series.1))/TDaily_returns.test)))
```

```
## MAPE Daily: 1.030331
```

```
cat("\nPM Daily:",sum((as.numeric(fore.series.1) - TDaily_returns.test)^2)/sum((TDaily_returns.test-mea
```

```
##
## PM Daily: 0.9995441
```

```
cat("\nMAPE Weekly:",mean(abs((TWeekly_returns.test-as.numeric(fore.series.2))/TWeekly_returns.test)))
```

```
##
## MAPE Weekly: 0.962692
```

```
cat("\nPM Weekly:",sum((as.numeric(fore.series.2) - TWeekly_returns.test)^2)/sum((TWeekly_returns.test-m
```

```
##
## PM Weekly: 1.143978
```

Response: Model comparison

The MAPE values show that the weekly model has a better performance than the daily model. The precision measure values show the daily model having a better balance between the variance of the actual vs prediction

variance, as the value of closer to 1. However, there are only two predictions from the weekly model, so it may not be definite. The precision measures are both close to 1, which means that the variability in the predictions is similar to the variability in the observed data over the prediction period.

This is a different result compared to the results from 2d, which showed a precision measure for the daily arima model much higher than 1. In addition, the MAPE value is slightly better for the weekly model, which was not the case for question 2d.

Question 4: Reflection on the Modeling and Forecasting

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 models fit the data? How well do the models 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?

Response: Final considerations

While both models performed well in the accuracy measures, the ARMA-GARCH model performed better according to the confidence interval, as one weekly data point was very close to the band. In addition, the ARIMA predictions did not have much variability. This was evident in the ARIMA daily stock price predictions, as the predictions were slowly decreasing, and exhibited no variation. On the other hand, the ARMA-GARCH model captured the variation of the data (as shown by the precision measure), as well as overall accuracy.

According to the MAPE, the ARIMA model performed better with the daily data, whereas the ARMA-GARCH model performed better with the weekly data. This is likely due to the fact that returns are on average very close to zero, and the variability in the predictions would significantly impact the values for MAPE. This is something to keep in mind when working with MAPE with different data, as it cannot be compared directly.

The ARIMA model would work best for data where variation is not expected to change, or is constant. Then the ARIMA model would be the preferred model due to its simplicity. This is not necessarily true for the stock market, as the variation changes constantly, and needs to be captured.