Stat 153 Midterm 2

Samba Njie Jr., Veronika Yang

4/7/2017

Report

Appendix : Code

Establishing working directory:

```
setwd("/Users/sambamamba/Documents/Cal Spring 2017/STAT_153/MT_2/GoogleTimeSeries")

wd <- getwd(); items <- dir()
```

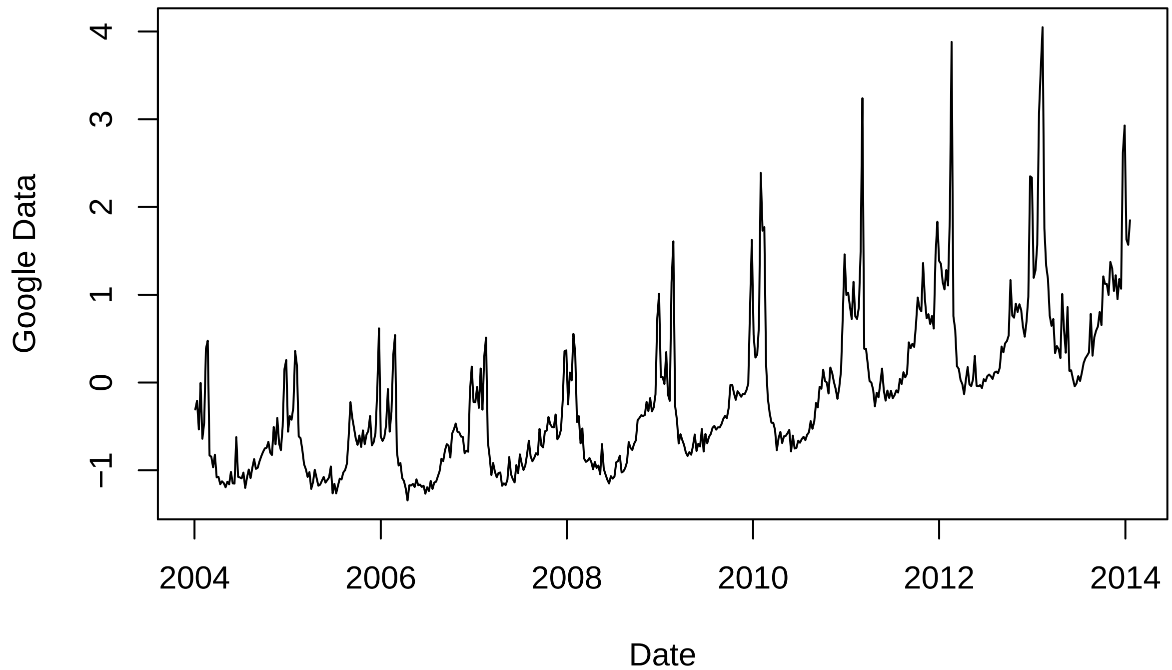
Read in data sets:

```
readData <- function() { # creates a list of the 5 Google data sets
  dtasets <-  items[grepl(".csv", items) == TRUE]
  dataList <- lapply(dtasets, function(dta) read_csv(file.path(wd, dta)))
  names(dataList) <-lapply(1:5, function(x) as.vector(paste0("Q",x,"Train")))
  return(dataList)
}

data <- readData() # where question i can be found by data[[i]] or data$QiTrain
```

Question 1

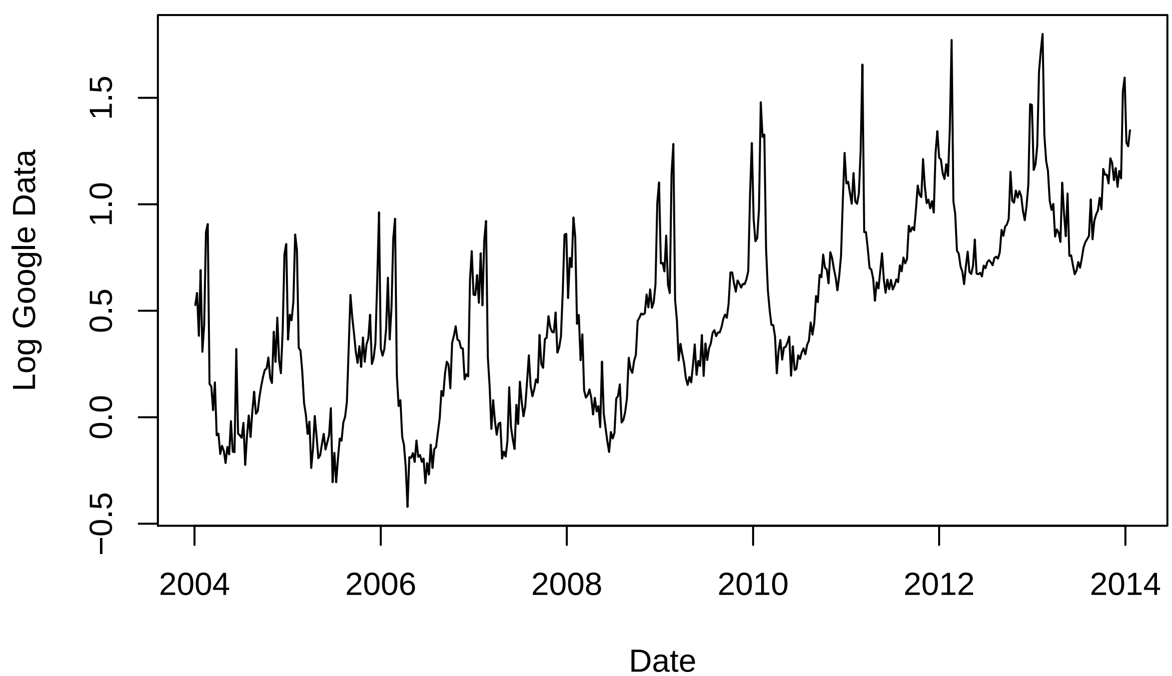
Exploratory Data Analysis

```
Q1Train <- data$Q1Train

plot(Q1Train, type = 'l', xlab = "Date", ylab = "Google Data")
```



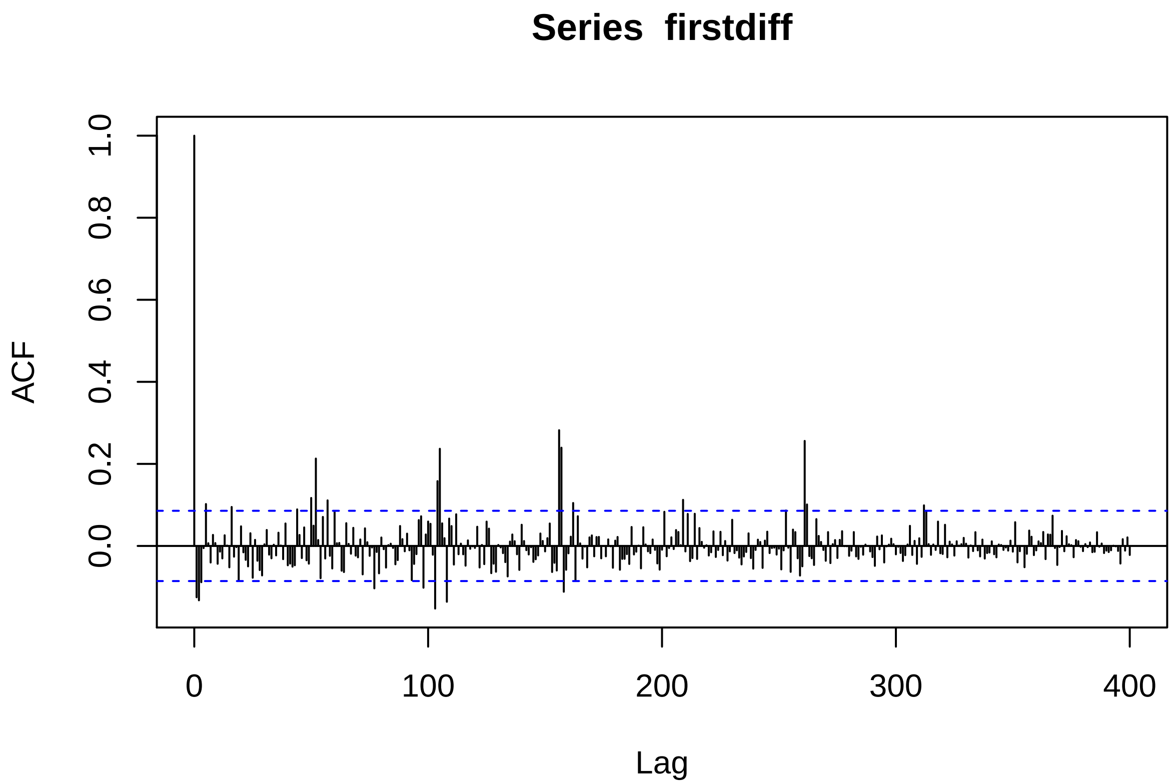
There seems to be an increasing linear trend and a clear seasonality in the data set, with a period of around a year. Homoscedasticity in the data set exists. Meaning, as time increases, there seems to be increasing variance in every period. For more convenient analysis and making variance more consistent, we will implement a log transformation of the data. However while log transformation reduces homoskedasticity, logarithms return `NaN` values with negative data. since the minimum data point in this question is -1.3435, we will shift the data by 2, then perform a log transform, as can be seen in the plot:

```
Q1Train.Log <- data.frame(Date = Q1Train$Date, Activity = log(Q1Train$activity + 2))
plot(Q1Train.Log, type = 'l', xlab = "Date", ylab = "Log Google Data")
```



With the shifted log data at hand, we have reduced homoskedasticity extensively. Now, we must remove the trend by using differencing, aspiring to achieve a stationary data set.

```r
q1train.log <- Q1Train.Log$Activity
#difference <- function(dta, lag.input = 1, order = 1) {
  # Performs differencing for any degree of regular differencing
  #time <- dta[,1]; ts <- dta[,2];
  #ts.out <- diff(ts, lag = lag.input, differences = order)
  #return(data.frame(Date = time[(order + 1):length(time)], Activity = ts.out))
#}

acfIndex <- function(vec, n.max = 1, mod = NA) {
  # input : vector of acf values; output : data frame of the top n.max values and their indices
  stopifnot(n.max <= length(vec));
  val <- rep(NA, n.max); idx <- rep(NA, n.max)
  for (i in 1:n.max) {
    val[i] <- max(vec); idx[i] <- which.max(vec)
    vec <- vec[-idx[i]]
  }
  if (is.na(mod) == FALSE) {
    mod.vec <- idx %% mod
    return(data.frame(index = idx, value = val, remainder = mod.vec))
  }
  return(data.frame(index = idx, value = val))
}


# Observe first and second differenced log data
firstdiff <- diff(q1train.log)
seconddiff <- diff(q1train.log, differences = 2)
thirddiff <- diff(q1train.log, differences = 3)

checkSeas <- acfIndex(acf(firstdiff, lag.max = 400)$acf, n.max = 20, mod = 52)
```

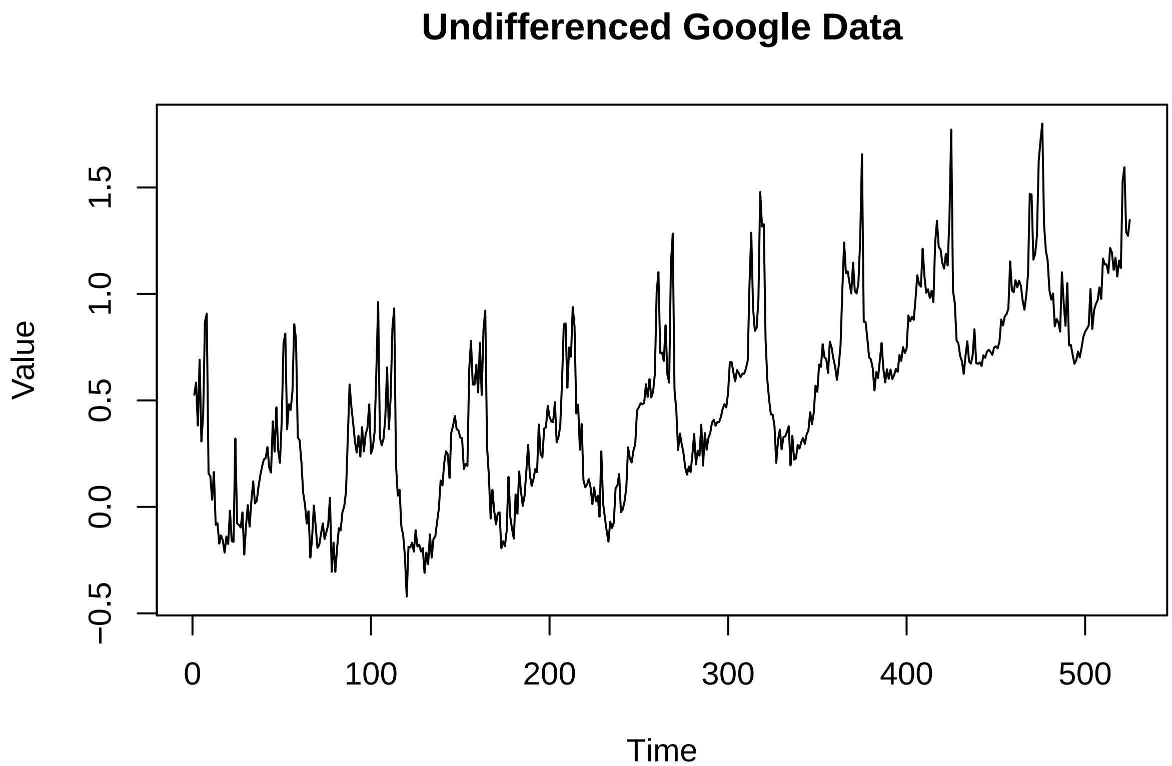



```r
# check if yearly
# the acfIndex indicates indices and values of the n highest acf values, and found a lot of them are ye
# seasonal data differencing
print(checkSeas)
```

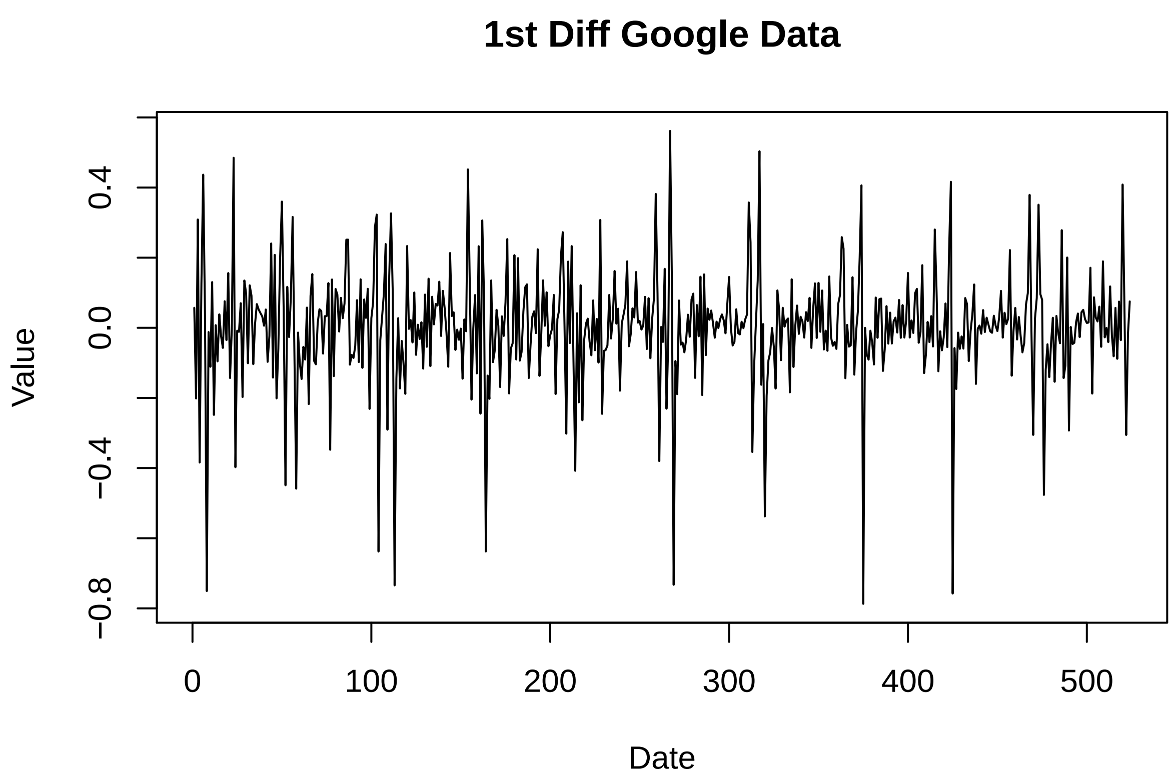
```
##    index      value remainder
## 1      1 1.00000000         1
## 2    156 0.28215879         0
## 3    260 0.25600402         0
## 4    156 0.23953679         0
## 5    105 0.23686474         1
## 6     52 0.21301361         0
## 7    103 0.15798856        51
## 8     50 0.11698184        50
## 9    203 0.11235940        47
## 10    55 0.11112628         3
## 11   155 0.10472789        51
## 12     5 0.10215912         5
## 13   251 0.10111736        43
## 14   300 0.09900733        40
## 15    15 0.09504322        15
## 16    42 0.08916602        42
## 17   241 0.08642917        33
## 18   297 0.08526497        37
## 19   190 0.08322715        34
## 20    54 0.08272026         2
```

```r
firstdiff.52 <- diff(firstdiff, lag = 50)
```

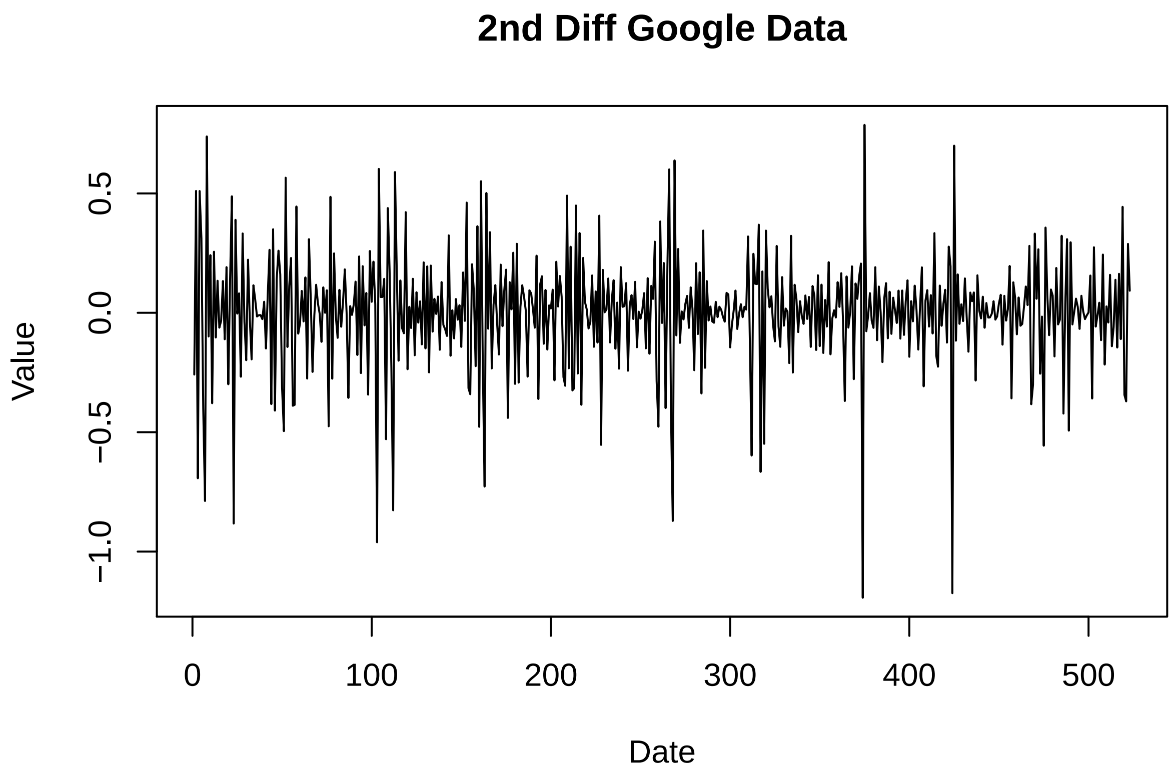
```r
# Observe differenced data of orders 1,2
#par(mfrow = c(2,1))
plot(q1train.log, type = 'l', xlab = "Time", ylab = "Value", main = "Undifferenced Google Data")
```



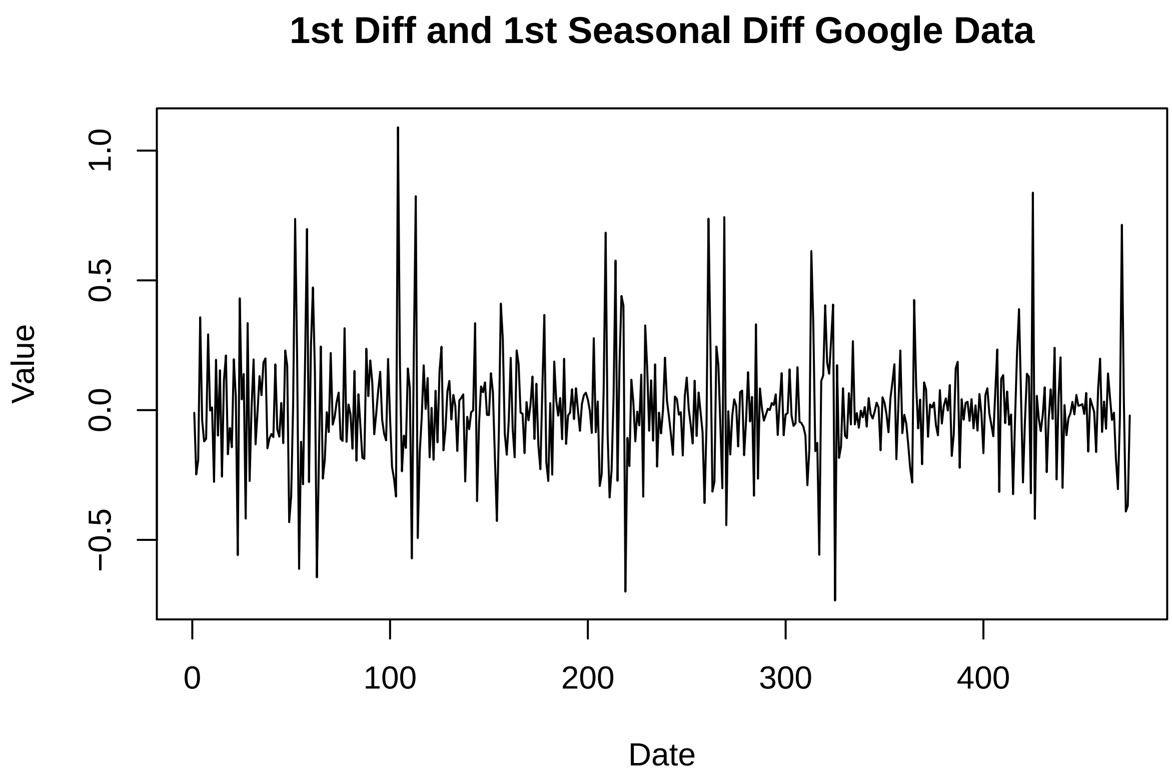

```r
plot(firstdiff, type = 'l', xlab = "Date", ylab = "Value", main =  "1st Diff Google Data");
```



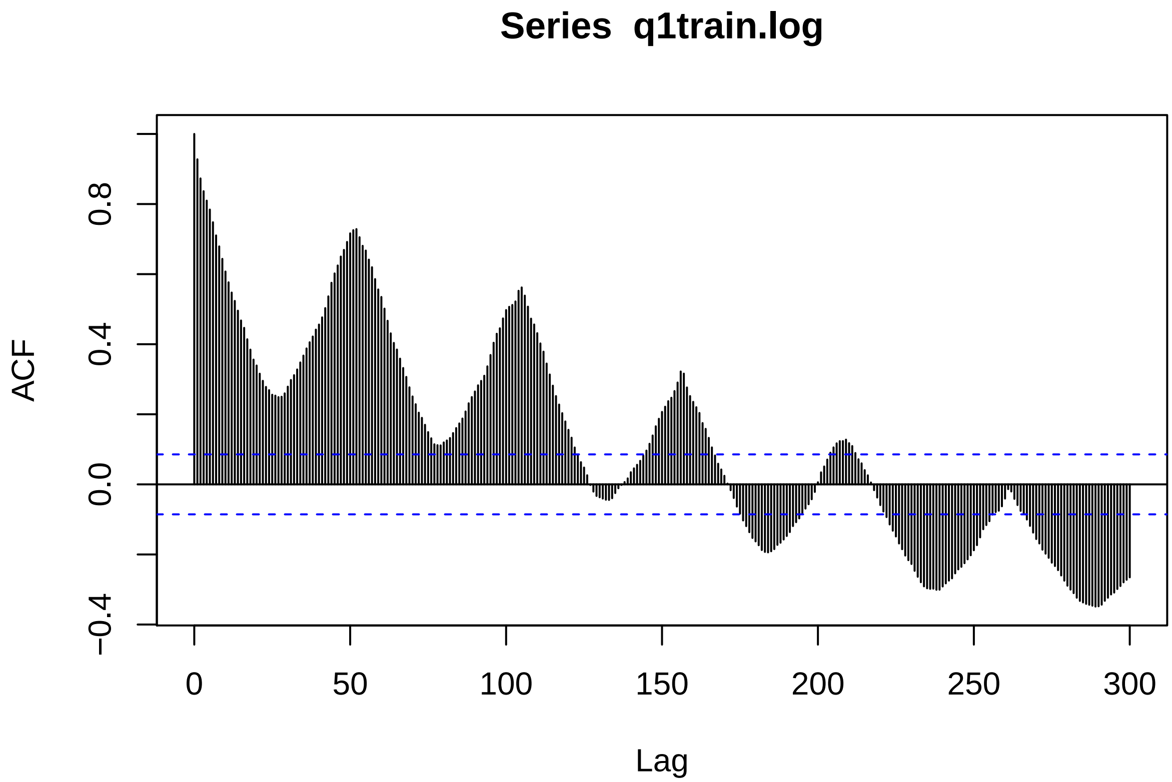
```r
plot(seconddiff, type = 'l', xlab = "Date", ylab = "Value", main =  "2nd Diff Google Data")
```

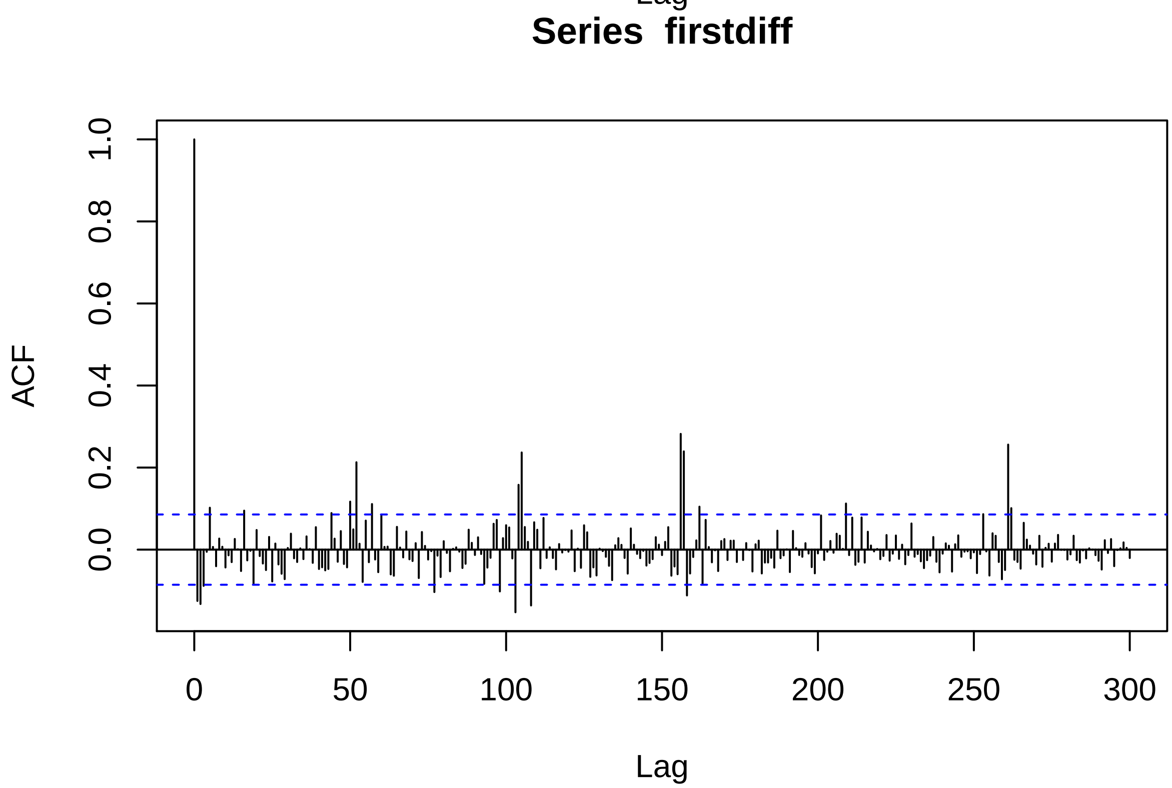


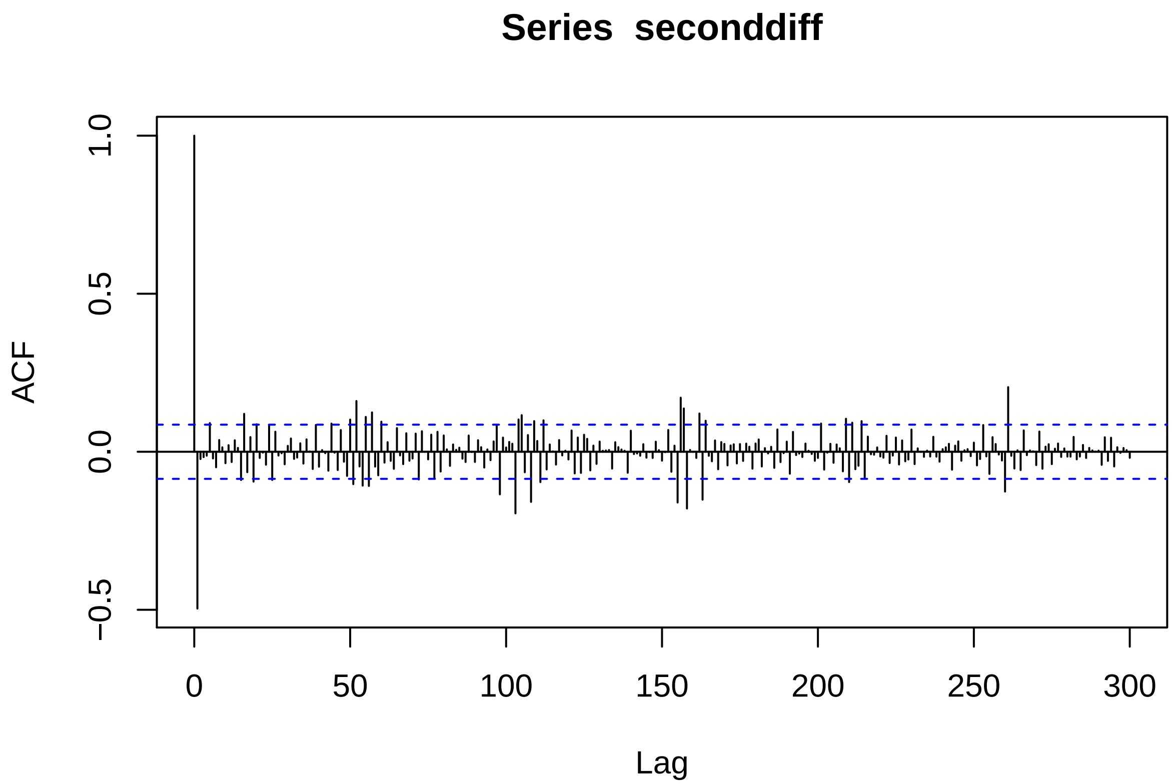

```r
plot(firstdiff.52, type = 'l', xlab = "Date", ylab = "Value", main =  "1st Diff and 1st Seasonal Diff Go
```

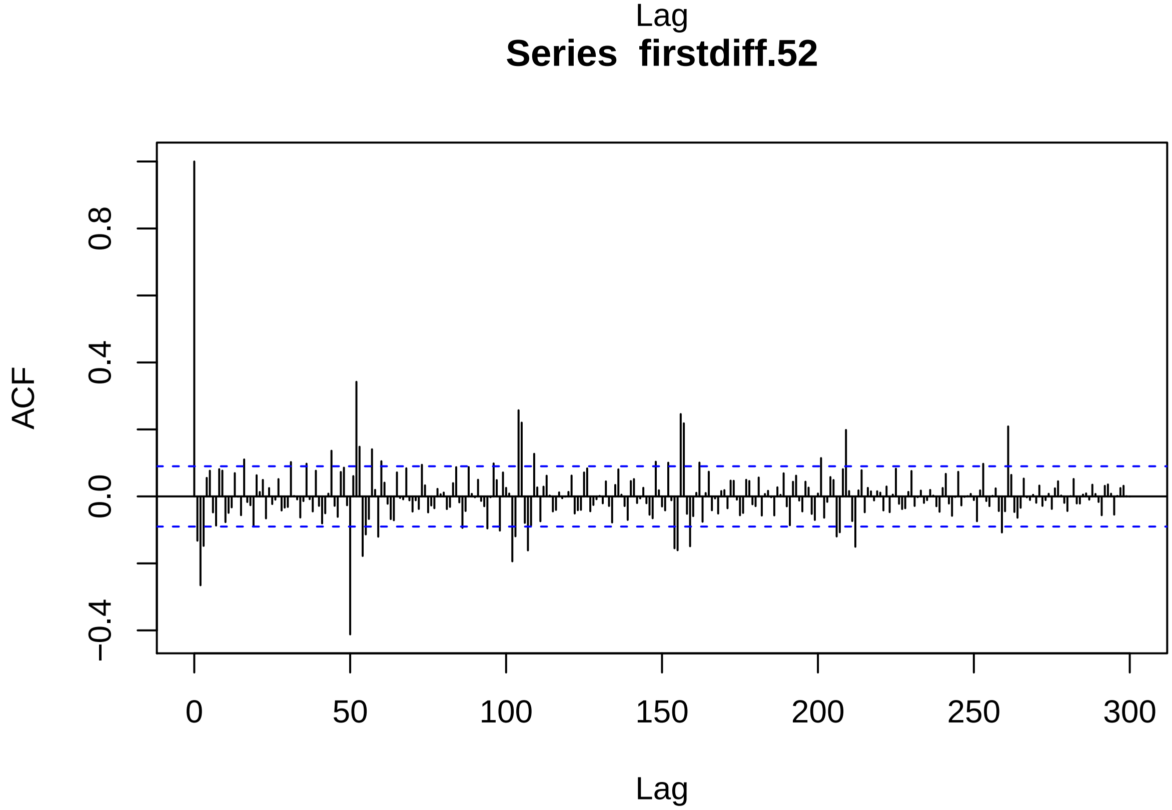


```
# Observe acf of data of orders 1, 2
acf(q1train.log, lag.max = 300); acf(firstdiff, lag.max = 300); acf(seconddiff, lag.max = 300); acf(fir
```

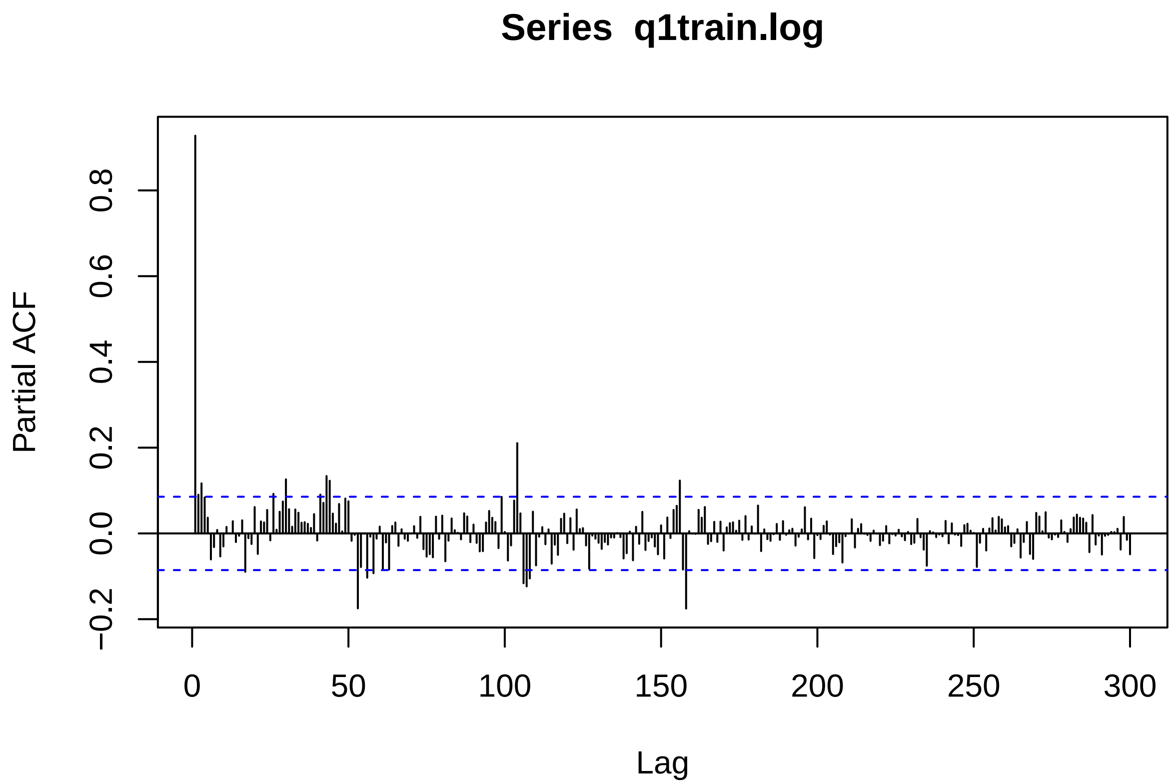


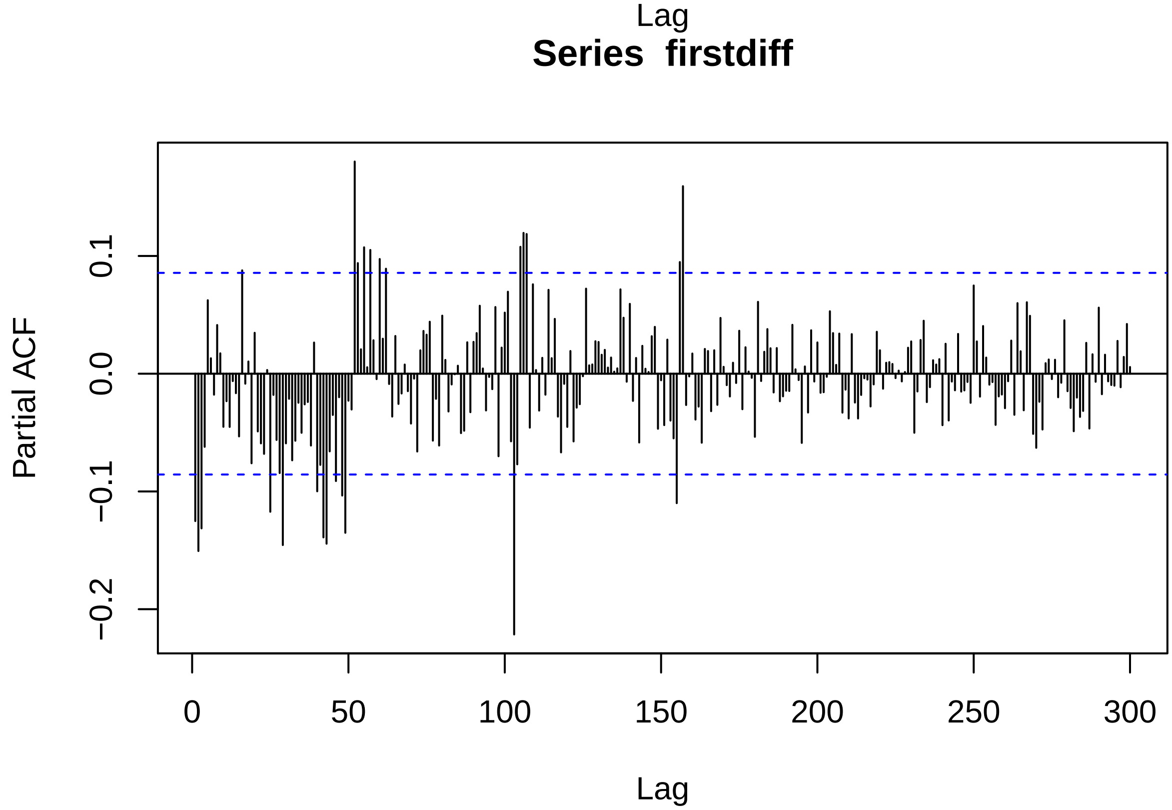


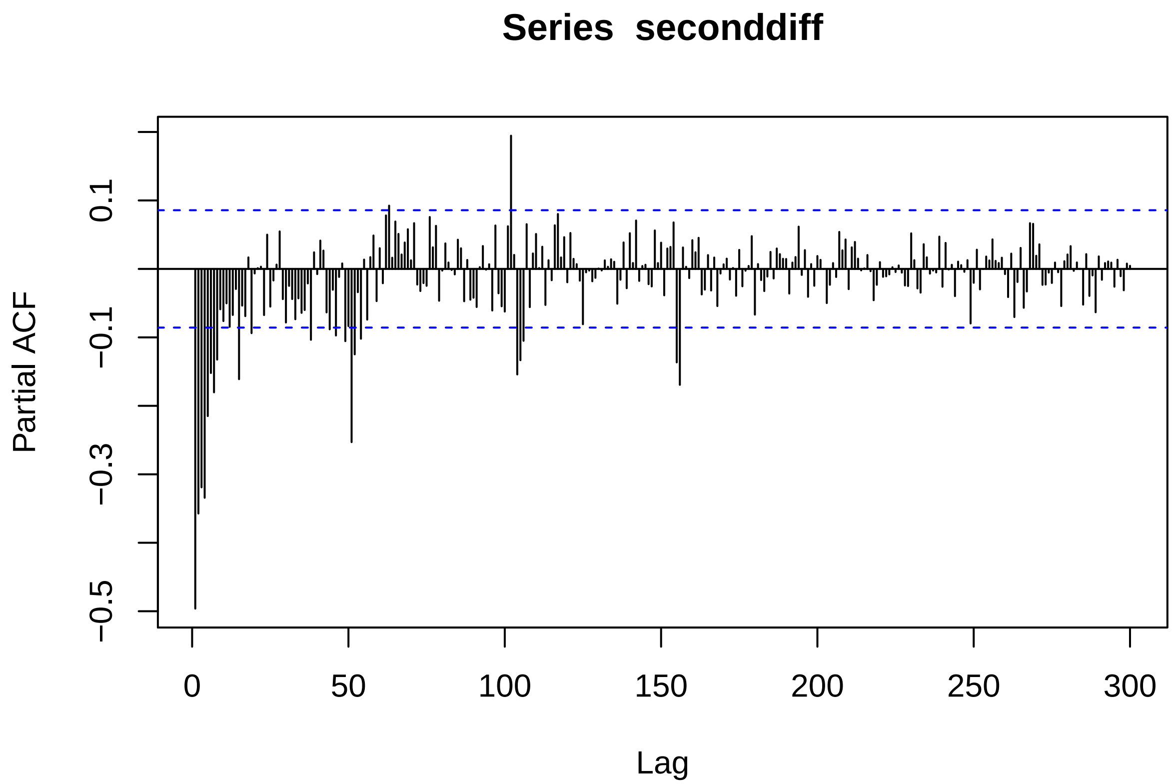


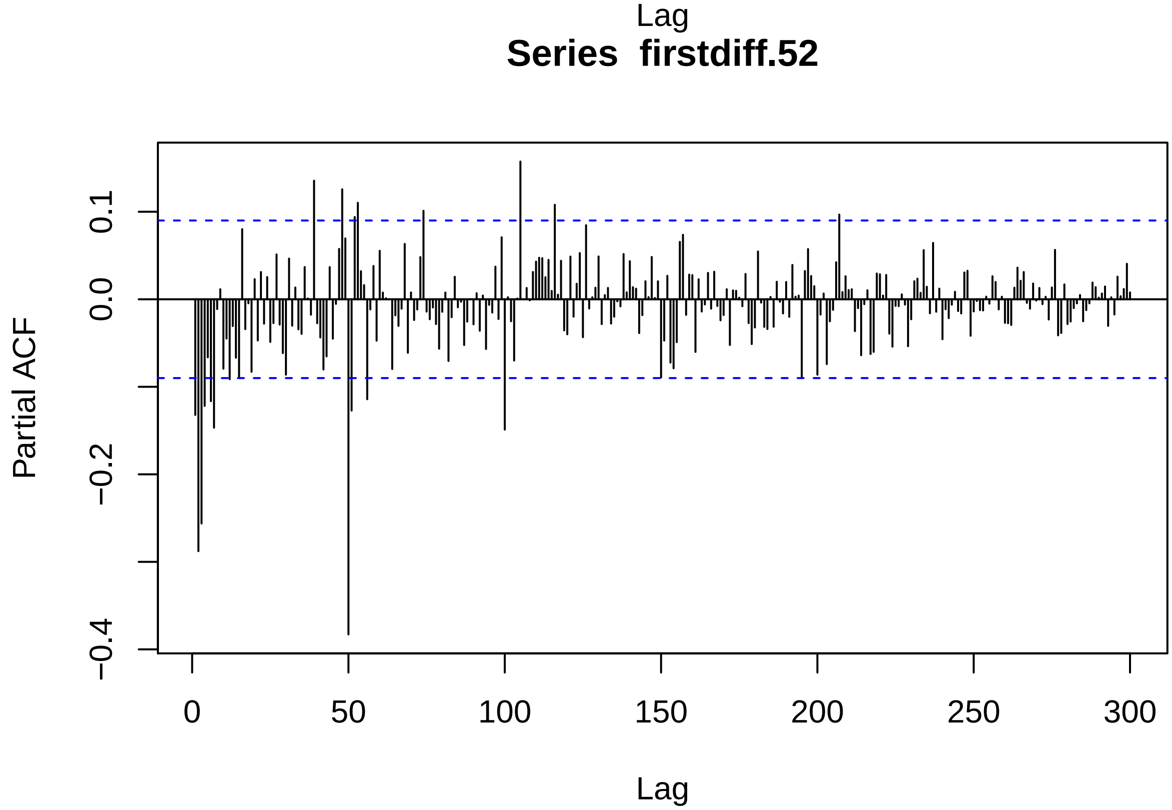



```
pacf(q1train.log, lag.max = 300); pacf(firstdiff, lag.max = 300); pacf(seconddiff, lag.max = 300); pacf
```


Series q1train.log
Partial ACF
−0.2
0.0
0.2
0.4
0.6
0.8
0
50
100
150
200
250
300
Lag


Series firstdiff
Partial ACF
−0.2
−0.1
0.0
0.1
0
50
100
150
200
250
300
Lag






```
# Lowest sd usually is the best model
which.min(sapply(list(q1train.log, firstdiff, seconddiff, firstdiff.52), function(x) print(sd(x)))) # l
```

```
## [1] 0.4390374
## [1] 0.1630348
```

```
## [1] 0.2447713
## [1] 0.2131558
## [1] 2
```

From the plots above, observe that with the second-order differenced data, there is significantly more higher magnitude of lags than the first-order differenced data. As such, we assume that the second-order differencing over-differences the data due to an increase in the breadth of lags with higher magnitudes, and also the existence of lags with higher than -0.5 magnitudes, not seen in the first-order plot. Also, notice that since the lag-1 acf signal is negative (slightly less than -0.5), we have a non-zero degree MA component to our model. The PACF would have significant evidence of showing an AR signature if its lag-1 component were positive, however this is not the case, and the sporadic nature of the lags tells us that the PACF does not provide enough information. There is also a non-negligible periodicity of high positive magnitude lags in the first-order difference plot, in which we assume that there is a seasonal component to the model. Since the ACF of the first-order differenced data is the one with closest stationarity, a strong lag-1 magnitude in the ACF we guess an $ARIMA(0,1,2)$, $ARIMA(0,1,1)$, $ARIMA(0,0,1)$, $ARIMA(0,1,1)x(0,1,1)_{52}$, $ARIMA(0,1,2)x(0,1,2)_{52}$.

Model Fitting

Now fit model into data set

- Always fit model into transformed data set.

```
m1 <- arima(q1train.log, order = c(0, 1, 1))
m2 <- arima(q1train.log, order = c(0,1,2))
m3 <- arima(q1train.log, order = c(0,0,1))
m4 <- arima(q1train.log, order = c(0,1,1), seasonal = list(order = c(0, 1, 1), period = 52))
m5 <- arima(q1train.log, order = c(0,1,2), seasonal = list(order = c(0, 1, 1), period = 52))
m6 <- arima(q1train.log, order = c(0,1,1), seasonal = list(order = c(0, 1, 2), period = 52))
m7 <- arima(q1train.log, order = c(0,1,2), seasonal = list(order = c(0, 1, 2), period = 52))
```

1. **standardized residuals**:

- want to see: rectangular scatter with no trends
- extreme points around 260, which indicate outliers in the data
- want to see plot that is relatively homoskedastic and don't want to see trends in any of the residuals

2. **acf of residuals**:

- want to see: no significant autocorrelations
- want all of acf to be close to 0 because don't want residuals to be correlated/ want them to be random errors

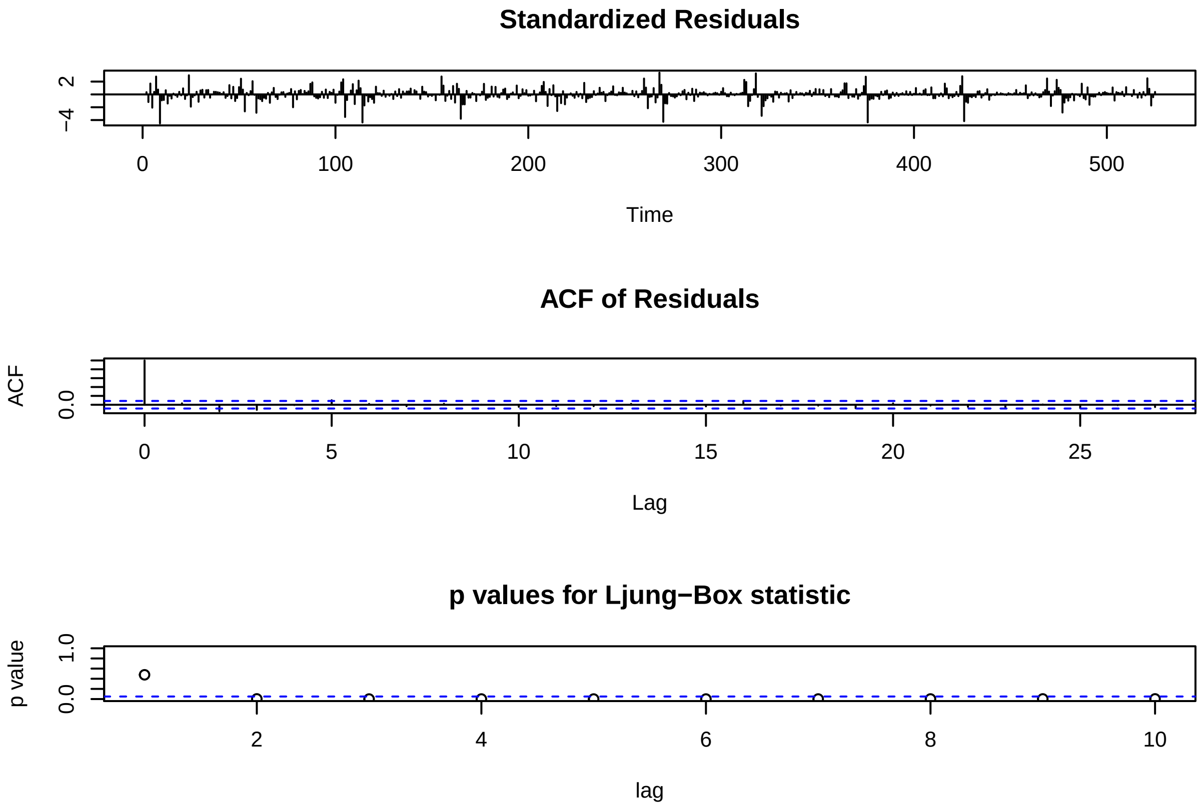
3. **Ljung-Box p-values**:

- test that checks if a group of autocorrelations is significantl different from 0
- tests as a group, are acf values significantly different from zero, as opposed to others to check if each lag is sig diff from 0
  - H0: data is iid
  - Ha: data exhibits serial correlation

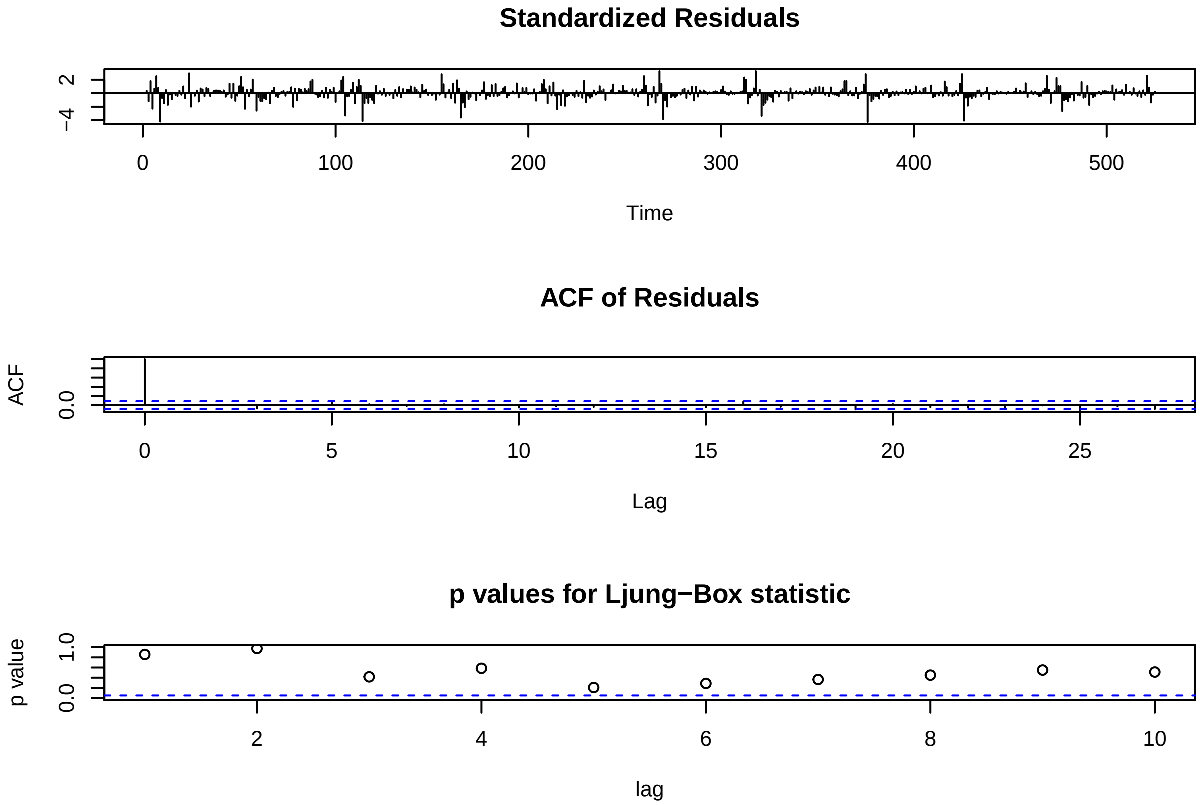
  want differences to be close to 0 –> want H0 to be true –> want p-vales > 0.05

since Ljung-Box is only one where data set are beyond the confidence bands, so there may be a better model to be fit

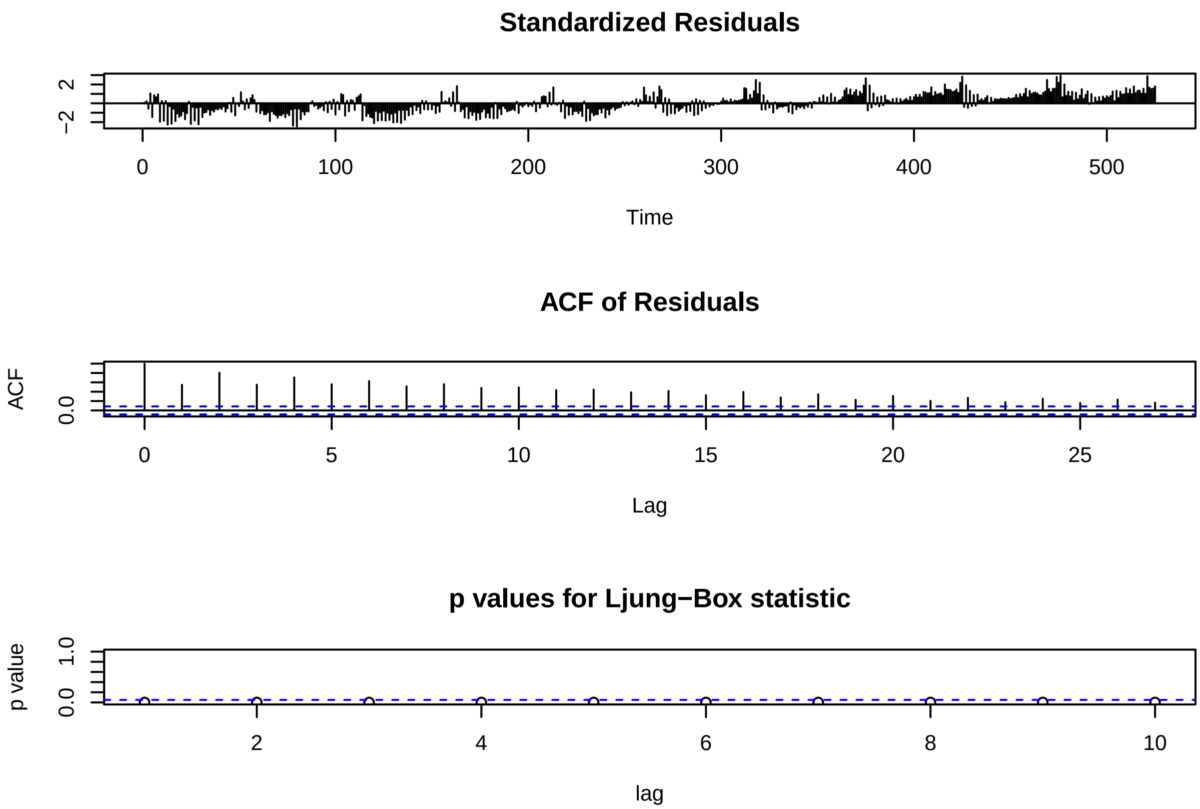
```r
# run diagnostics:

tsdiag(m1) # only one significant value in Ljung Box Statistic
```



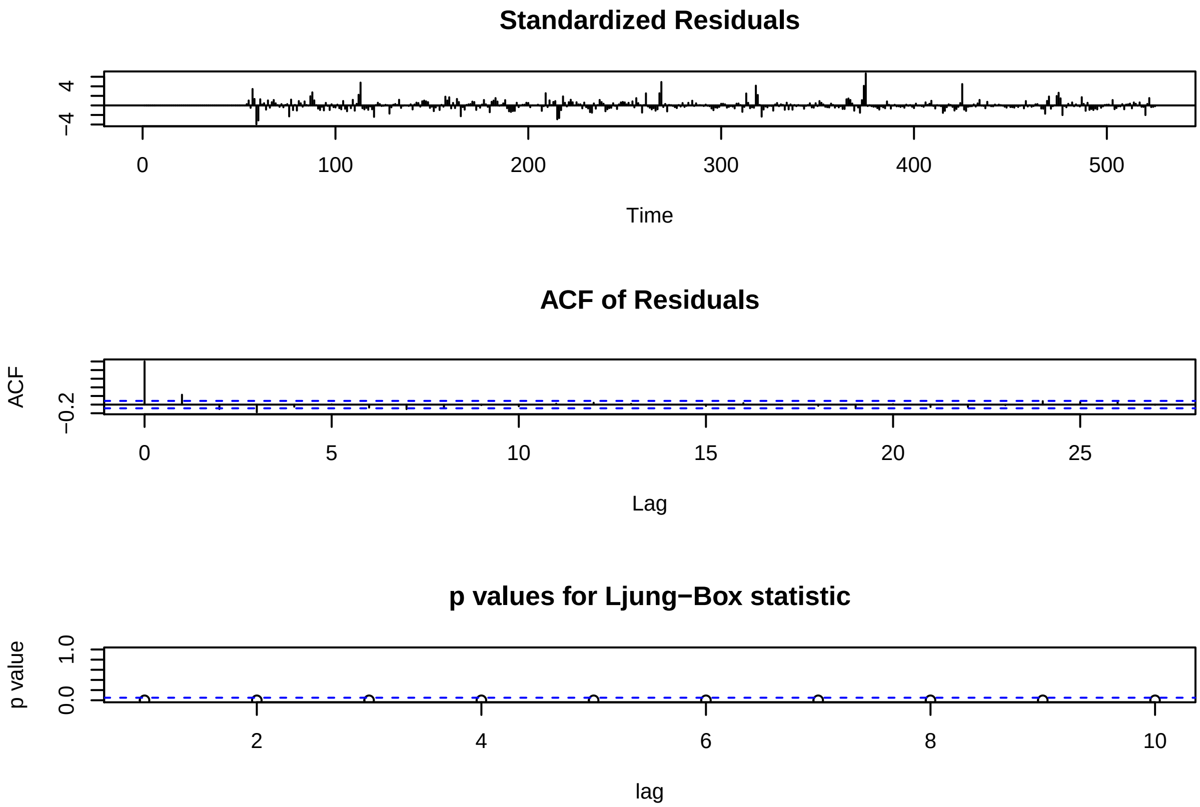

```r
tsdiag(m2) # nearly all significant from 0 for Ljung
```



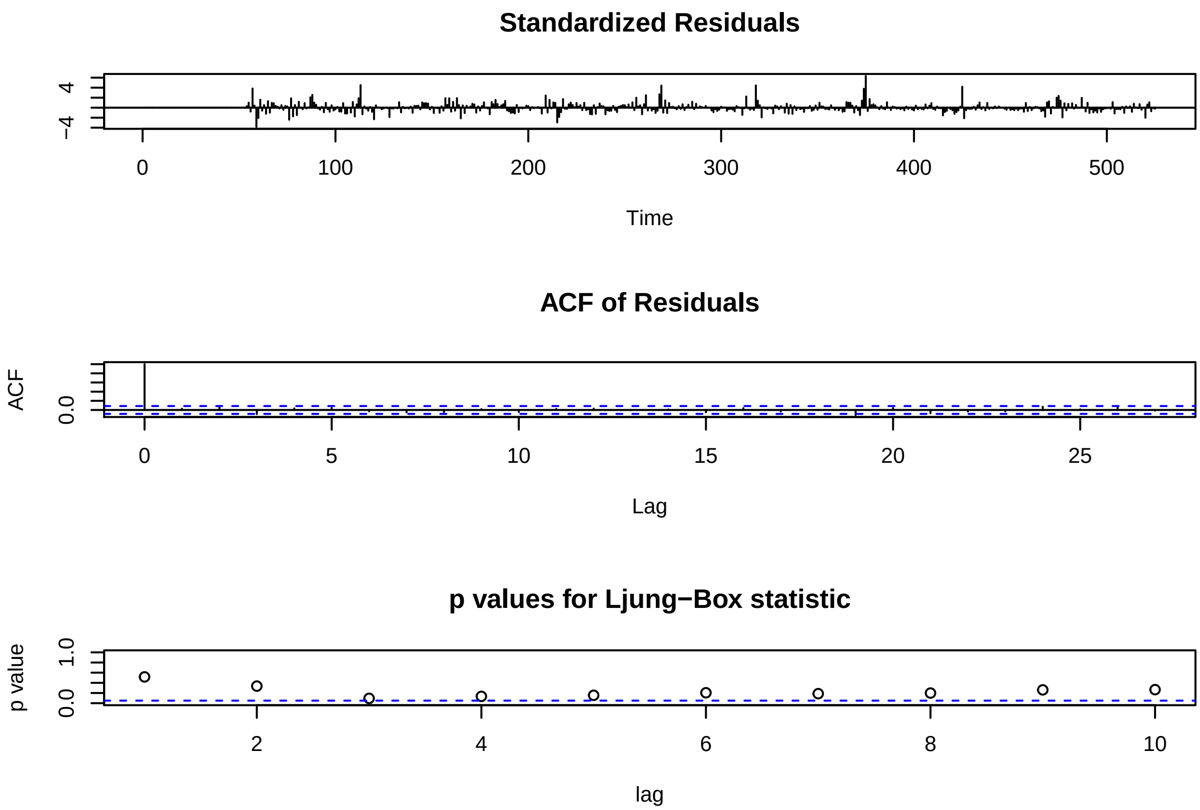

```
tsdiag(m3) # way off, ACFs all significant, trend in standardized residuals
```



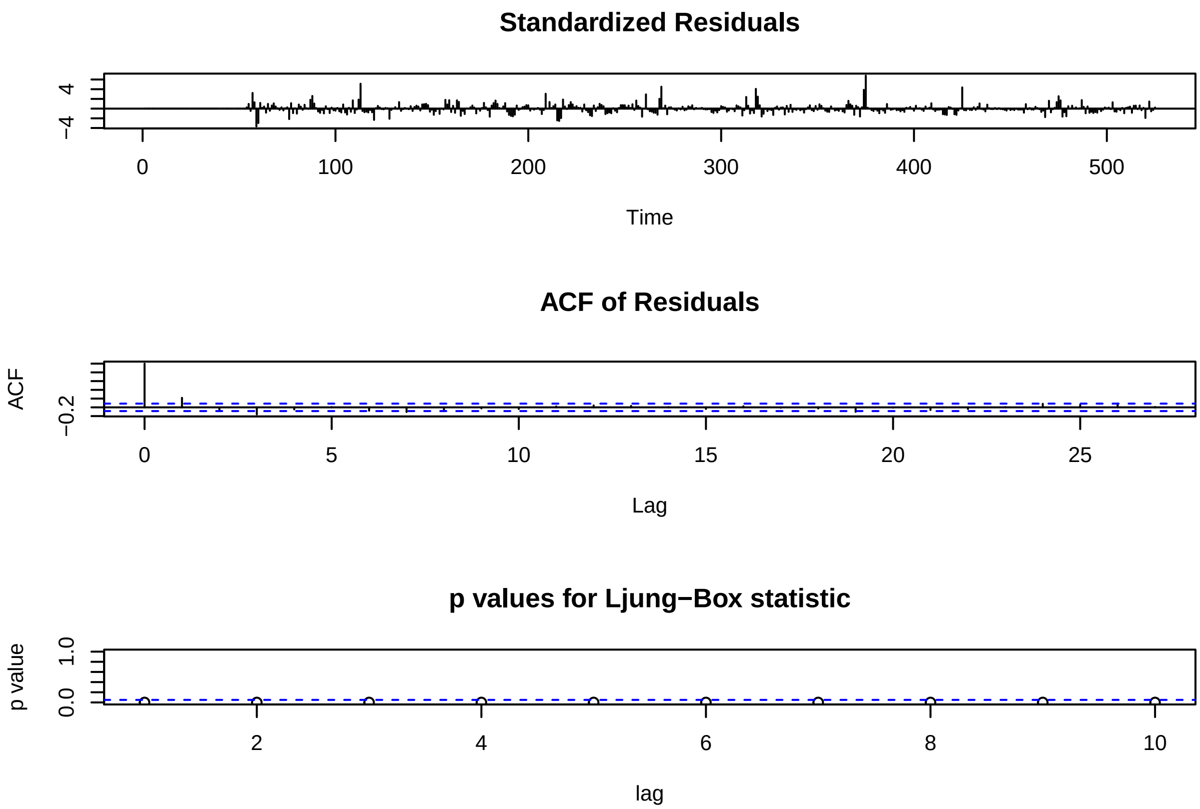
```r
tsdiag(m4) # Ljung has all values near 0
```



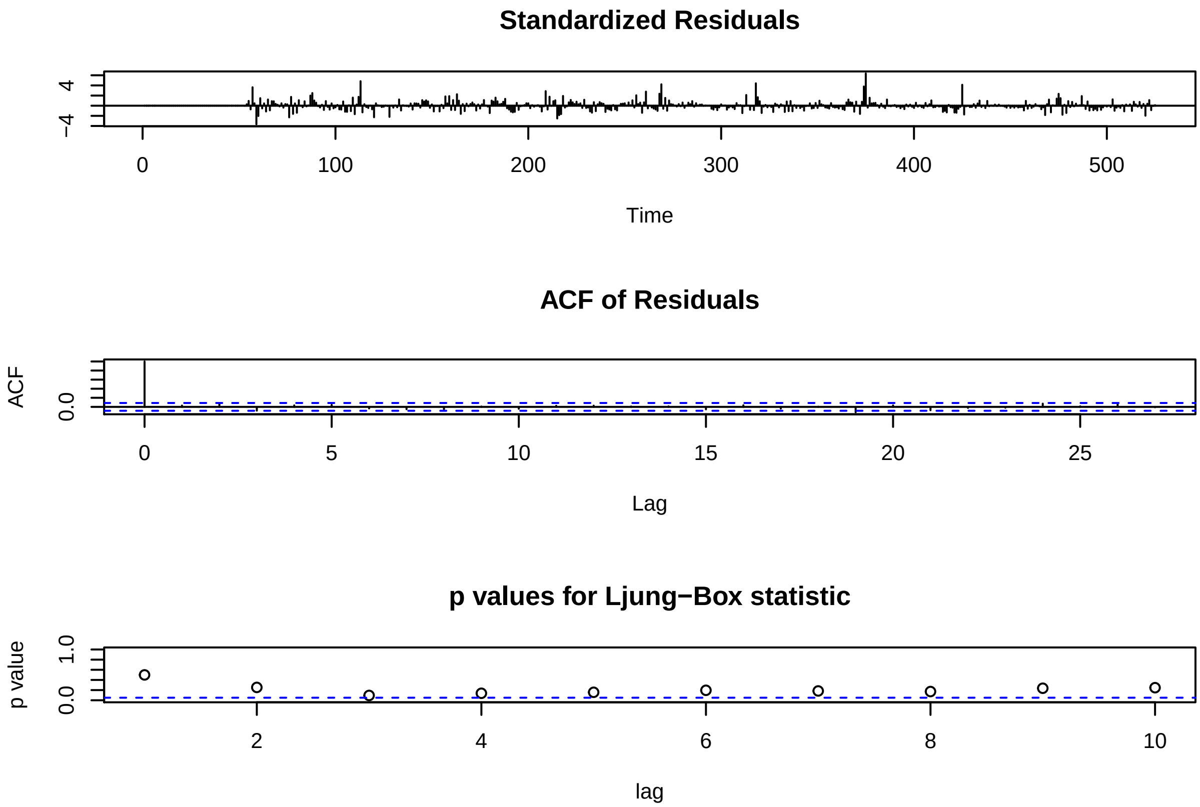

```r
tsdiag(m5) # all but one Ljung insignificant
```




```
tsdiag(m6) # below zero, near 0 Ljung statistic values
```



```r
tsdiag(m7) # teetering above 0 line a bit for Ljung
```



m2 seems to have the best model of all of them. we try AIC, BIC now

```r
sapply(list(m1,m2,m3,m4,m5,m6,m7), AIC); which.min(sapply(list(m1,m2,m3,m4,m5,m6,m7), AIC))
```

```
## [1] -422.9226 -434.6739  161.4187 -426.9510 -473.2317 -437.6907 -479.0921
```

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

```r
sapply(list(m1,m2,m3,m4,m5,m6,m7), BIC); which.min(sapply(list(m1,m2,m3,m4,m5,m6,m7), BIC))
```

```
## [1] -414.3996 -421.8894  174.2089 -414.4801 -456.6038 -421.0628 -458.3072
```

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

Both AIC and BIC point to m7 to be the best model. This is also the most complex model, so maybe we overfit the data set. We can look at cross valudation to check.

```r
computeCVmse <- function(ts, order.totry = c(0L, 0L, 0L),
                         seasorder.totry = c(0L, 0L, 0L), d = 52, num.seas = 5){
  # computes MSE
  MSE = numeric()
  len = length(ts)
  for (k in num.seas:1) {
    train.dt = ts[1:(len - d*k)]
    test.dt = ts[(len - d*k + 1):(len - d*(k - 1))] # split into training and test data
    mod = arima(train.dt, order = order.totry,
                seasonal = list(order = seasorder.totry, period = d)) # fit model into training data
    fcst = predict(mod, n.ahead = d) # because we are predicting seasons, so 52 points
    MSE[k] = mean((exp(fcst$pred) - exp(test.dt))^2) # reverses our previous log transform
```

```r
  }
  return(MSE)
}
```

The above function runs cross validation and predicts up to `d` points ahead, where `d` is the period of a seasonal ARIMA.

```r
#m1 <- arima(q1train.log, order = c(0, 1, 1))
#m2 <- arima(q1train.log, order = c(0,1,2))
#m3 <- arima(q1train.log, order = c(0,0,1))
#m4 <- arima(q1train.log, order = c(0,1,1), seasonal = list(order = c(0, 1, 1), period = 52))
#m5 <- arima(q1train.log, order = c(0,1,2), seasonal = list(order = c(0, 1, 1), period = 52))
#m6 <- arima(q1train.log, order = c(0,1,1), seasonal = list(order = c(0, 1, 2), period = 52))
#m7 <- arima(q1train.log, order = c(0,1,2), seasonal = list(order = c(0, 1, 2), period = 52))

MSE1 <- computeCVmse(q1train.log, c(0,1,1), num.seas = 9)
MSE2 <- computeCVmse(q1train.log, c(0,1,2), num.seas = 9)
MSE3 <- computeCVmse(q1train.log, c(0,0,1), num.seas = 9)
MSE4 <- computeCVmse(q1train.log, c(0,1,1), c(0,1,1), 52, num.seas = 9)
MSE5 <- computeCVmse(q1train.log, c(0,1,2), c(0,1,1), 52, num.seas = 9)
MSE6 <- computeCVmse(q1train.log, c(0,1,1), c(0,1,2), 52, num.seas = 9)
MSE7 <- computeCVmse(q1train.log, c(0,1,2), c(0,1,2), 52, num.seas = 9)

apply(ldply(list(MSE1, MSE2, MSE3,MSE4,MSE5,MSE6,MSE7), print), 2, which.min)
```

```
## [1] 1.1057777 0.8114794 0.4986750 0.9599229 0.5837007 1.2289713 0.6797553
## [8] 0.4799915 0.7502497
## [1] 1.1005396 0.8648398 0.5201557 0.9456109 0.5800050 1.1116585 0.6459627
## [8] 0.4353872 0.6956977
## [1] 2.7329251 1.8059945 1.7493398 0.8505631 0.5825329 0.2524226 0.2321354
## [8] 0.2045409 0.1351749
## [1] 0.31101379 0.21804876 0.52517779 0.16064631 0.15071189 0.07010756
## [7] 0.04622394 0.08304913 0.31555695
## [1] 0.28799284 0.20974837 0.55069226 0.16111668 0.17601515 0.06375086
## [7] 0.06353072 0.07904211 0.84738605
## [1] 0.24682166 0.19665552 0.49738196 0.16621257 0.14964012 0.07010615
## [7] 0.05829928 0.08288729 0.31555778
## [1] 0.24071332 0.20292747 0.52083604 0.16495310 0.17605789 0.06393120
## [7] 0.07591506 0.07904465 0.84739525
```

```
## V1 V2 V3 V4 V5 V6 V7 V8 V9
##  7  6  6  4  6  5  4  5  3
```

Surprisingly enough, cross-validation by seasons generates that model 6 generates the lowest mean-squared error for the second, third, and fifth

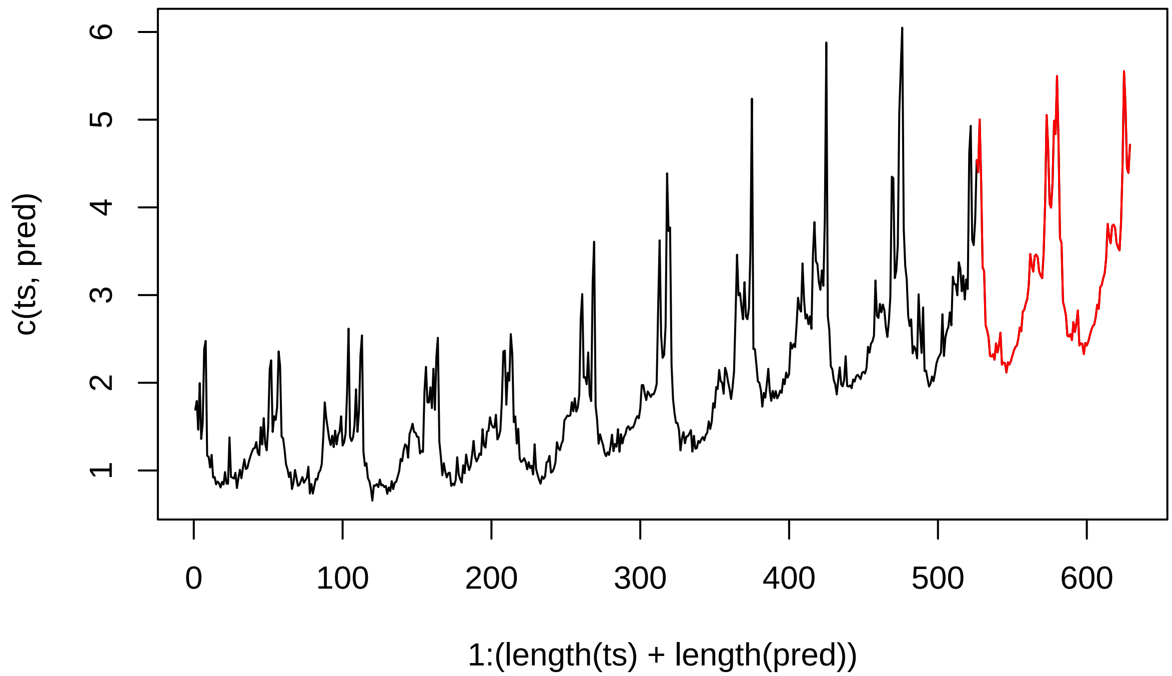
Forecasting

```r
predictions <- predict(m6, n.ahead = 104)$pred
```

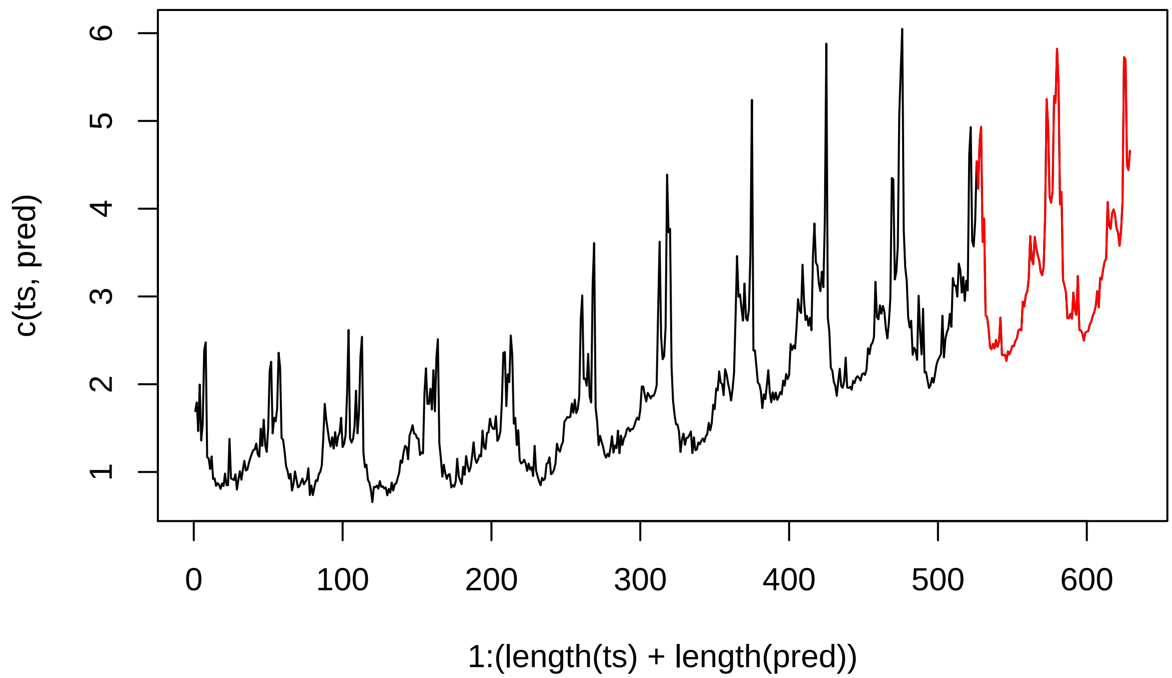
need to reverse transformation (take exponential of predicted values)

```r
predictions4 <- exp(predict(m4, n.ahead = 104)$pred)
predictions6 <- exp(predict(m6, n.ahead = 104)$pred)
predictions7 <- exp(predict(m7, n.ahead = 104)$pred)
```

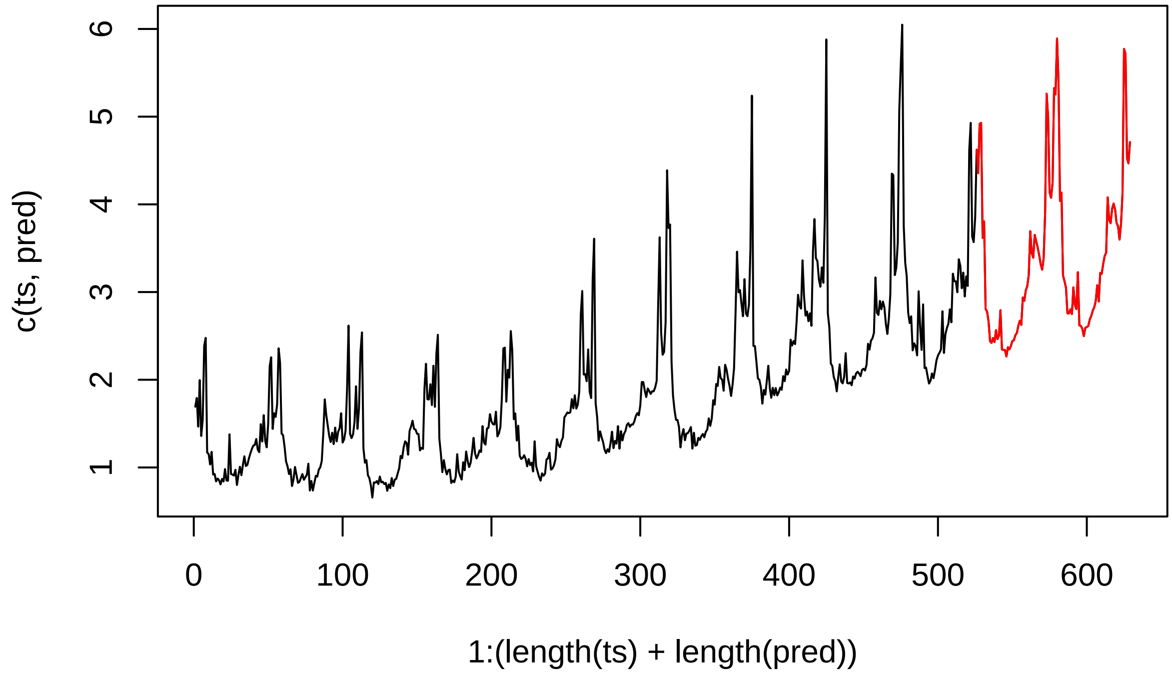
```r
plotPred <- function(ts, pred) {
  plot(1:(length(ts) + length(pred)), c(ts, pred), type = 'l', col = 1); points((length(ts) + 1) : (len
}

plotPred(exp(q1train.log), predictions4)
```




```r
plotPred(exp(q1train.log), predictions6)
```



```r
plotPred(exp(q1train.log), predictions7)
```

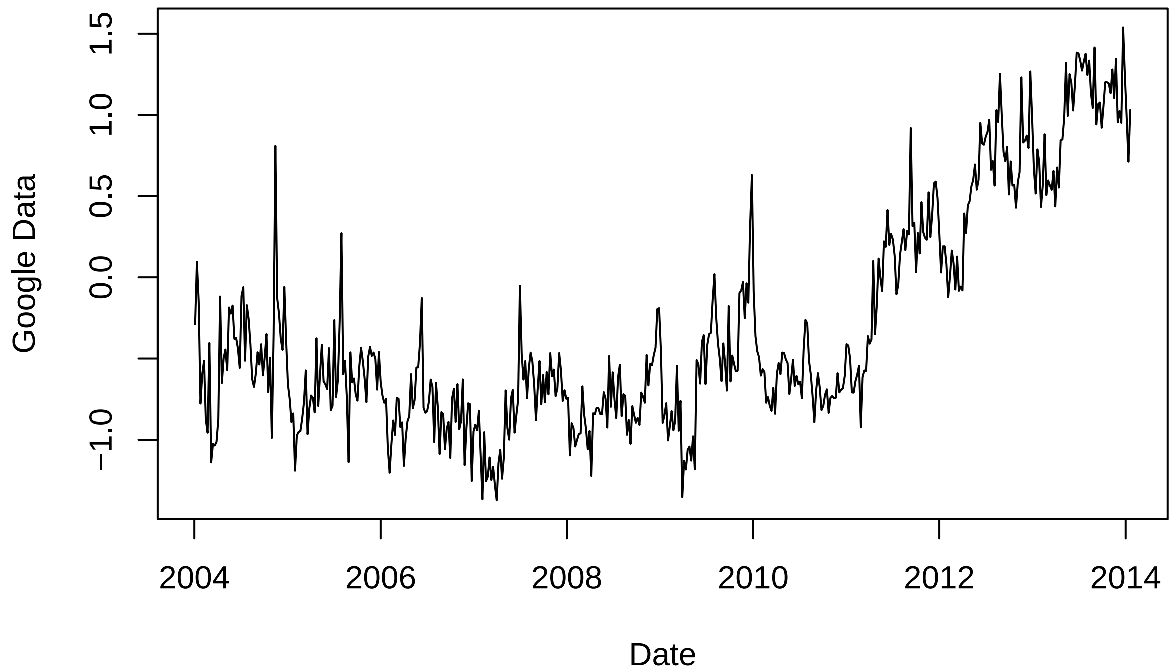


residuals of plot

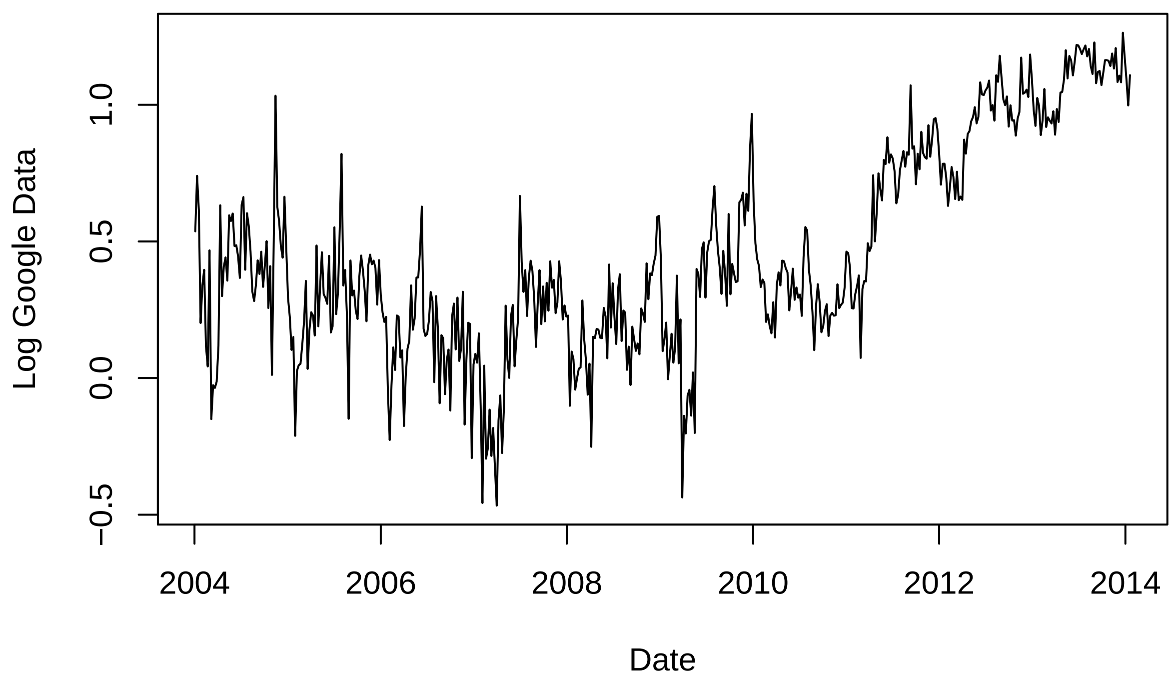
- doing adjusted R^2, AIC, BIC, differencing
- polynomial, exponential fit

Question 4

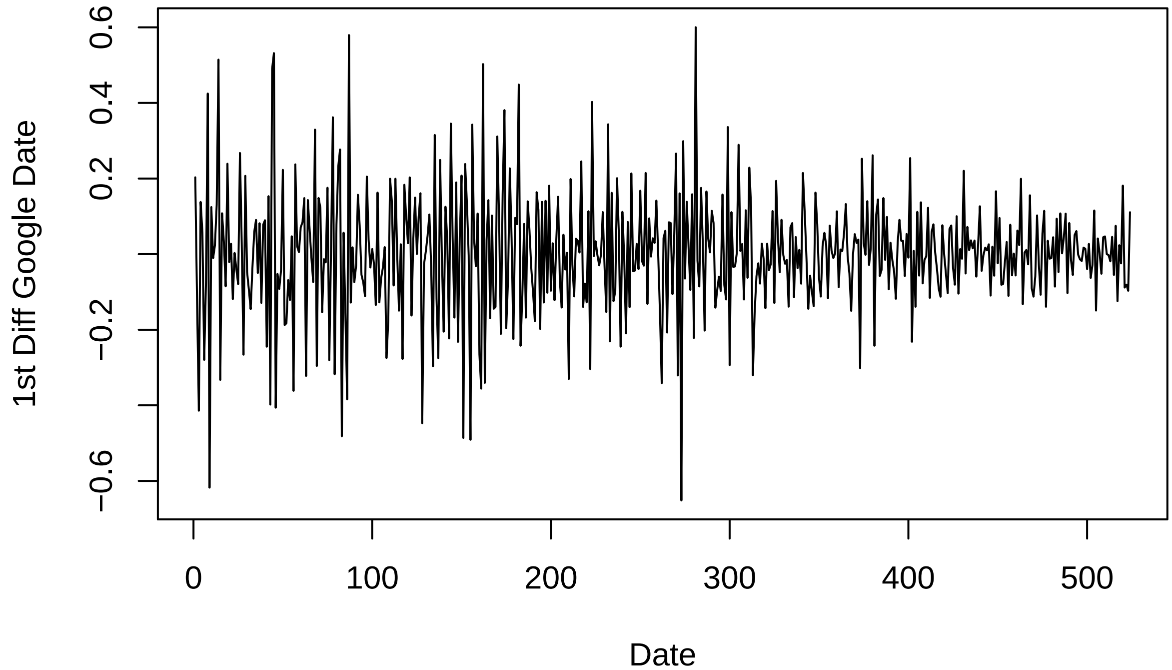
```r
Q4Train <- data$Q4Train

plot(Q4Train, type = 'l', xlab = "Date", ylab = "Google Data")
```



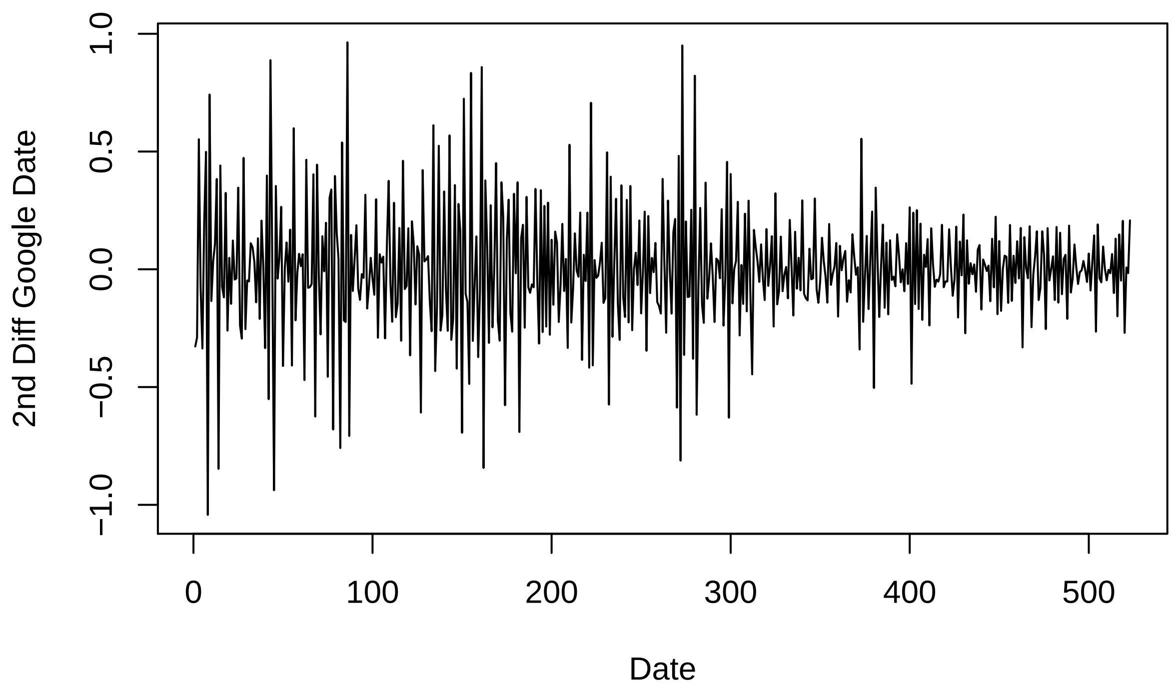

```
Q4Train.Log <- data.frame(Date = Q4Train$Date, Activity = log(Q4Train$activity + 2))
plot(Q4Train.Log, type = 'l', xlab = "Date", ylab = "Log Google Data")
```



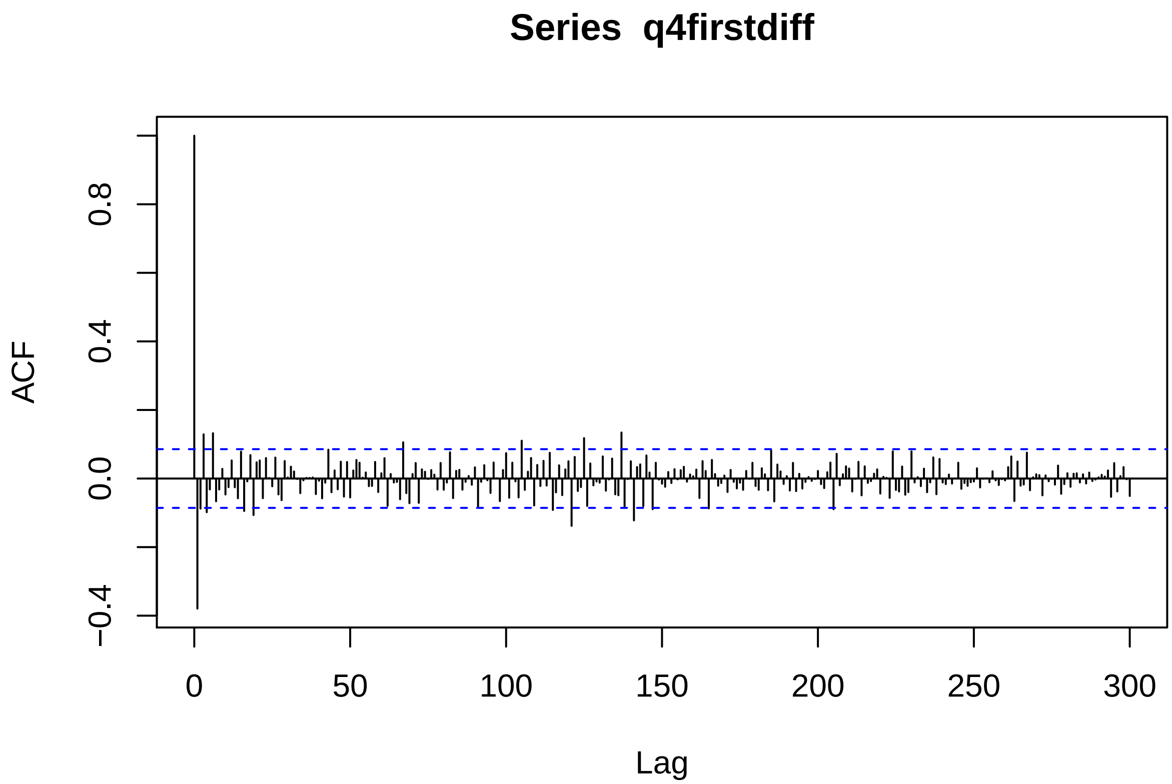

```
q4train.log <- Q4Train.Log$Activity

# Observe first and second differenced log data
q4firstdiff <- diff(q4train.log)
q4seconddiff <- diff(diff(q4train.log))

# Observe differenced data of orders 1,2
plot(q4firstdiff, type = 'l', xlab = "Date", ylab = "1st Diff Google Date");
```



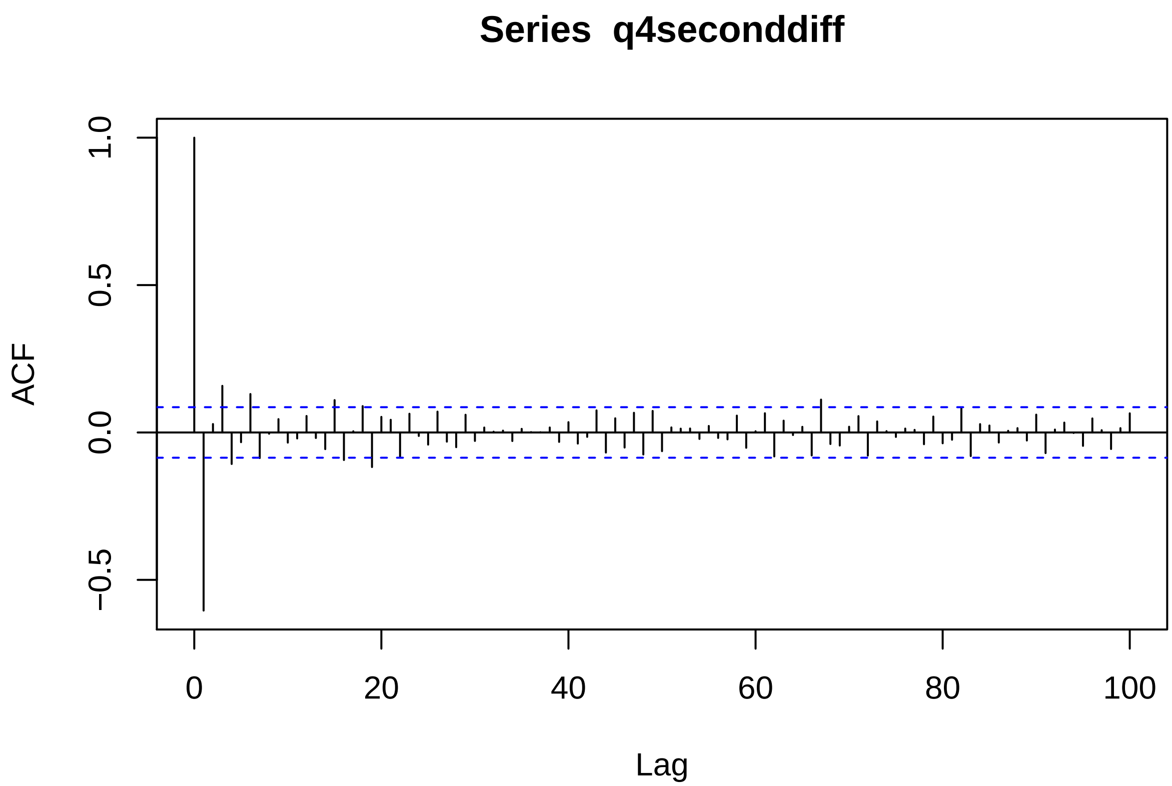

```
plot(q4seconddiff, type = 'l', xlab = "Date", ylab = "2nd Diff Google Date")
```



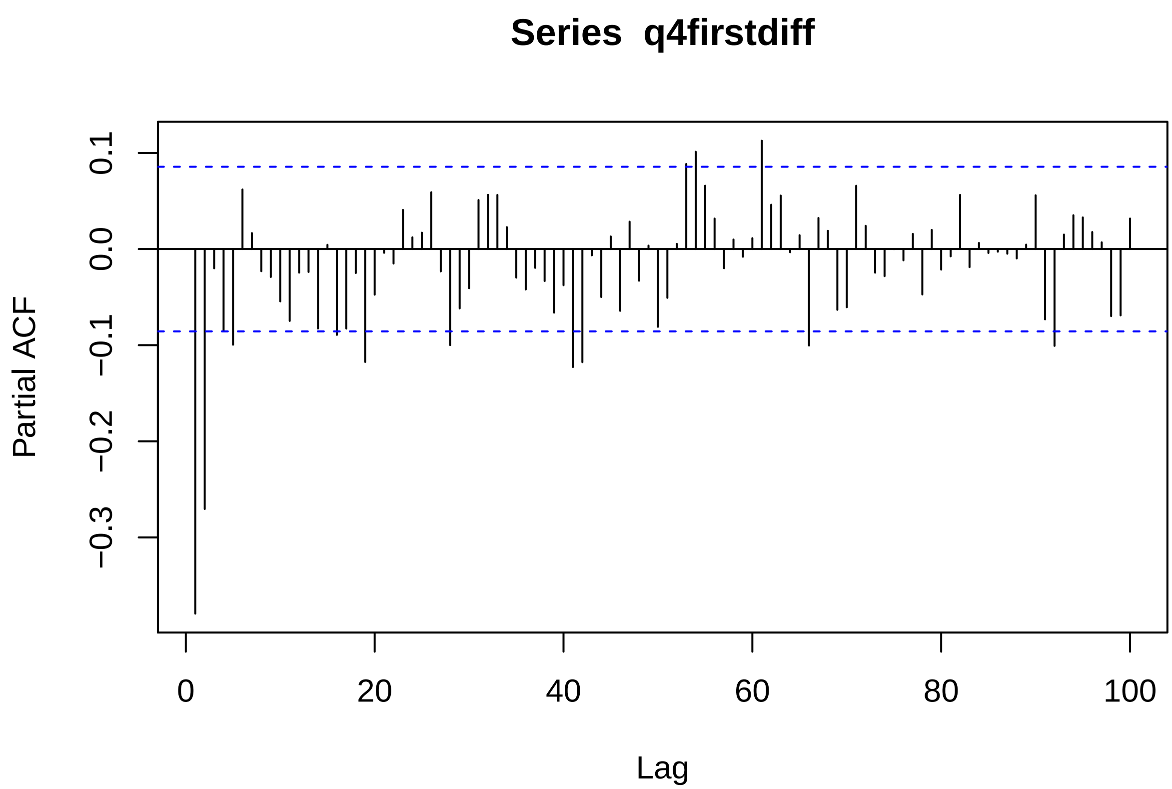

```
# Observe acf of data of orders 1, 2
#par(mfrow = c(2,1))
acf(q4firstdiff, lag.max = 300)
```



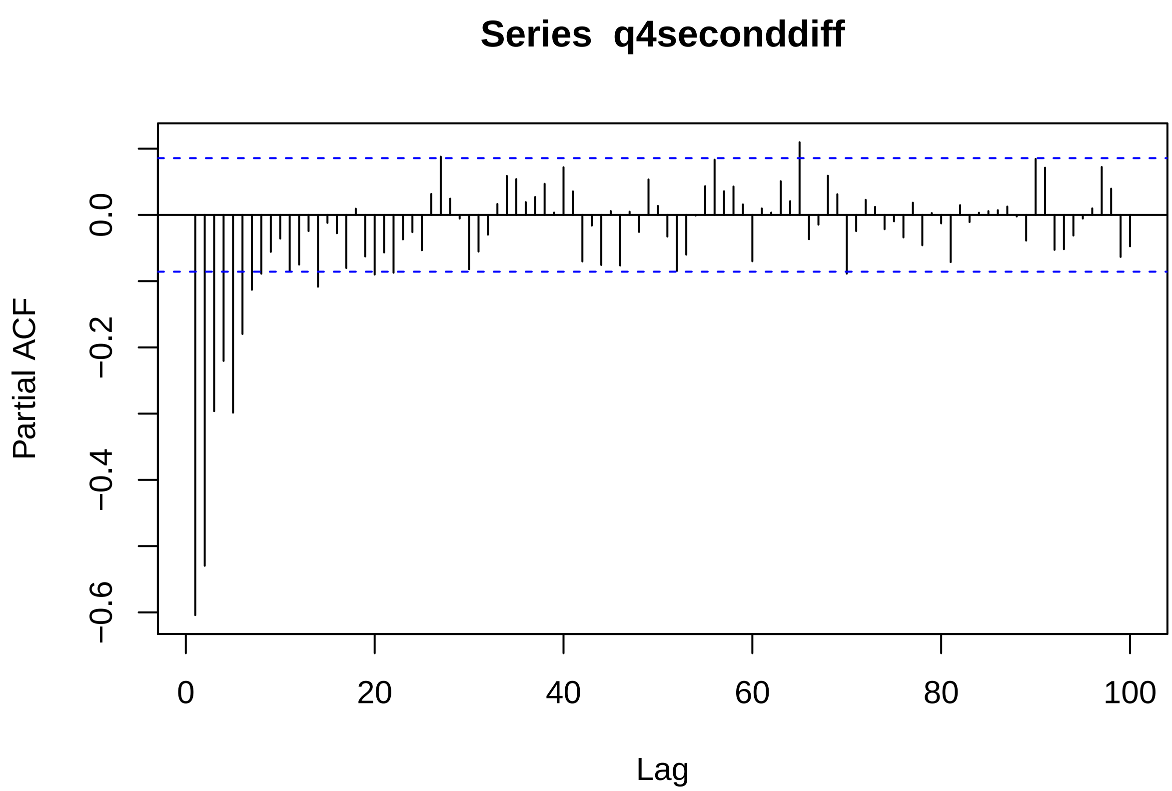

```r
acf(q4seconddiff, lag.max = 100)
```




```r
#par(mfrow = c(2,1))
pacf(q4firstdiff, lag.max = 100)
```




```r
pacf(q4seconddiff, lag.max = 100)
```



Creating the Submission File

```r
writeData <- function(dataset, q.num, firstname, lastname, SID) {
  output <- write.table(dataset,
            sep = ",",
            col.names = FALSE,
            row.names = FALSE,
            file = paste0("Q",q.num,"_",
                          firstname,"_",
                          lastname,"_",SID))
  return(output)
}
```