PSTAT 174 Final Project

Akul Bajaj

2022-03-10

Libraries

Here we are downloading the libraries to work with.

```
library(tsdl)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(zoo)
```

```
##
## Attaching package: 'zoo'
```

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

```
library(ggfortify)
```

```
## Loading required package: ggplot2
```

```
library(ggplot2)
library(MuMIn)
library(UnitCircle)
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
## Registered S3 methods overwritten by 'forecast':
##   method                 from
##   autoplot.Arima         ggfortify
##   autoplot.acf           ggfortify
##   autoplot.ar            ggfortify
##   autoplot.bats          ggfortify
##   autoplot.decomposed.ts ggfortify
##   autoplot.ets           ggfortify
##   autoplot.forecast      ggfortify
##   autoplot.stl           ggfortify
##   autoplot.ts            ggfortify
##   fitted.ar              ggfortify
##   fortify.ts             ggfortify
##   residuals.ar           ggfortify
```

Executive Summary

This was a successful time series analysis project. Our goal was to forecast gasoline demand, and we were successfully able to do that. What we did here was we plotted and analyzed the time series. We examined main features of the graphs, checking for trend and seasonality particularly. We then used necessary transformations to achieve a stationary series. We transformed our data using a box cox transformation then we differenced it at lag 12 and lag 1. Afterwards we plotted and analyzed the ACF and PACF to identify models. Following we fit this models, estimating coefficients and performing diagnostic checking. We were able to write our models in algebraic form. From the analysis of the residuals we found model A satisfactory and model B non-satisfactory. Finally we finished by doing forecasting, including confidence intervals, and returning to the original data, from the box cox transformed data.

Introduction

In this project we look at Monthly gasoline demand in millions of gallons. This data is taken from Ontario and is available in the tsdl library. The data is taken from January 1960 to December 1975. We have 15 years of data. The problem is that we want to forecast data for the following year. We use box cox transformations, SARIMA models, and several other time series techniques to solve our problem. We are successfully able to build a model to predict the 1976 gasoline demand in millions of gallons.

Data Cleaning

We are using data from the tsdl library. We assigned it the variable name "ap". There are 192 observation in this data set. We then plotted the data.

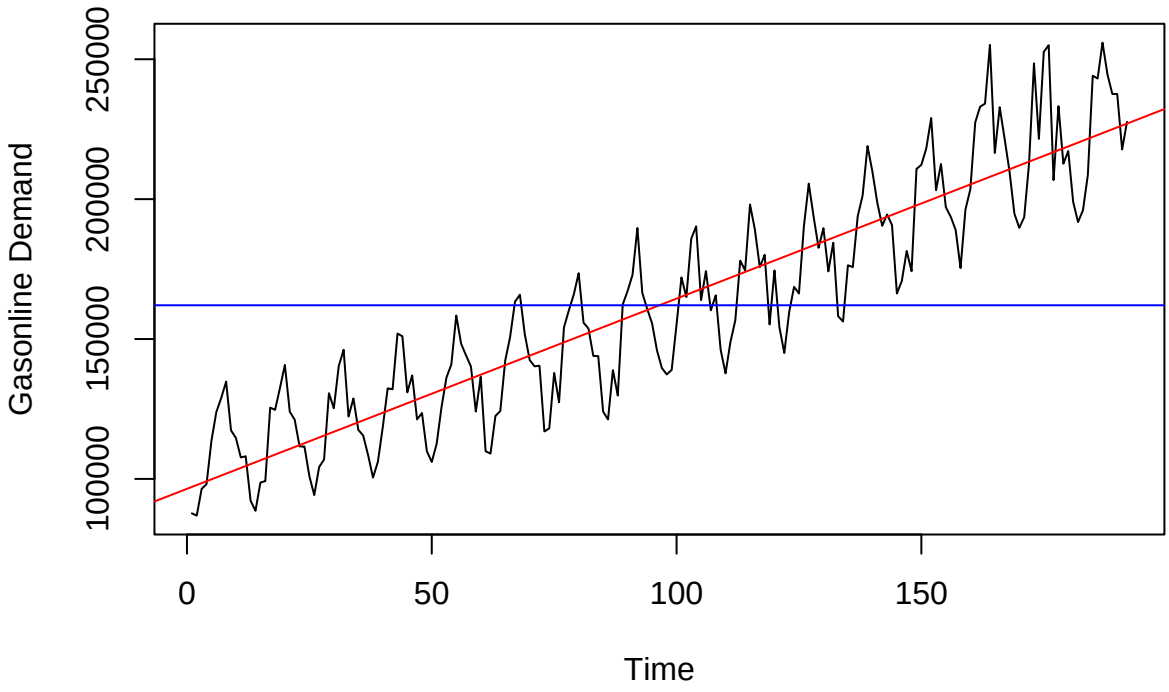
```
ap<-as.data.frame(tsdl[[4]])



plot.ts(ap, ylab='Gasonline Demand')

dim(ap)
```

```
## [1] 192   1
```

```
fit <- lm(tsdl[[4]] ~ as.numeric(1:192))%>%
  abline(fit, col="red")
mean(tsdl[[4]])[1]
```

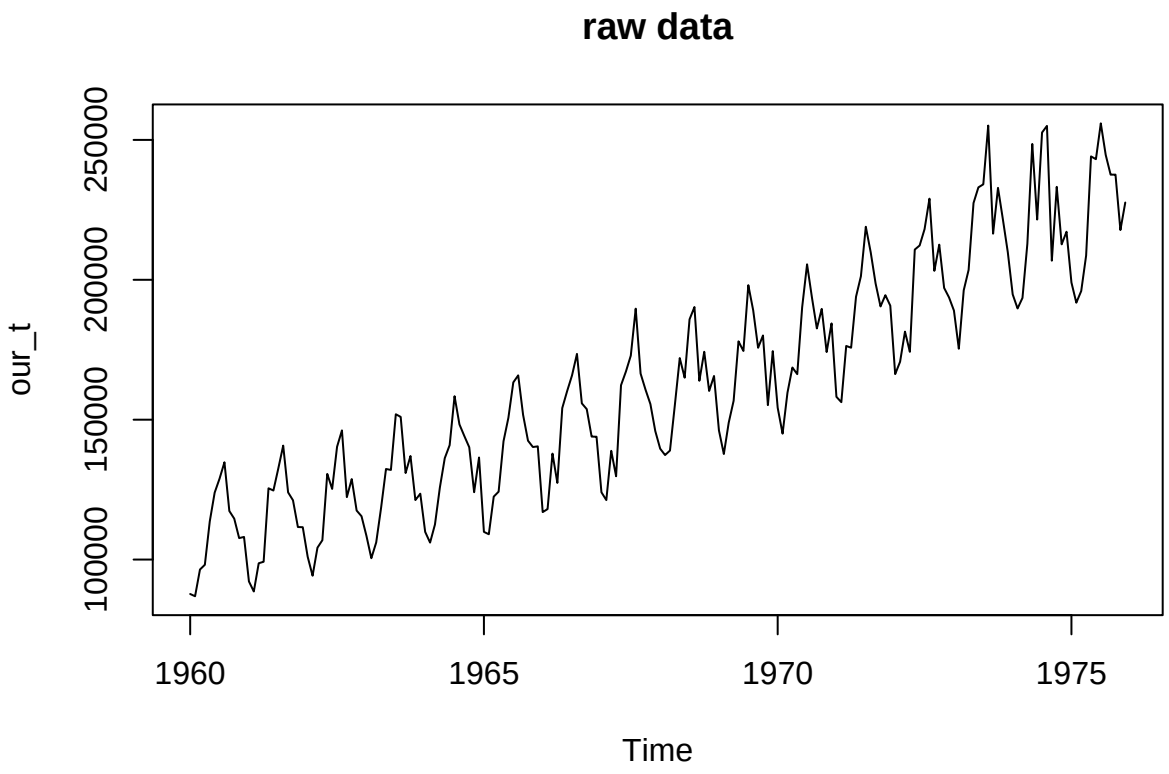
```
## [1] 162063.7
```

```
abline(h=mean(tsdl[[4]]), col='blue')
```



Now we convert the data frame into a time series friendly data frame and we split the data into training and testing data sets. We use the last 12 values for testing.

```
#monthly demand for gasoline in Ontario
our_t<-ts(tsdl[[4]],start = c(1960,1),end = c(1975,12), frequency = 12)

ts.plot(our_t, main="raw data")
```

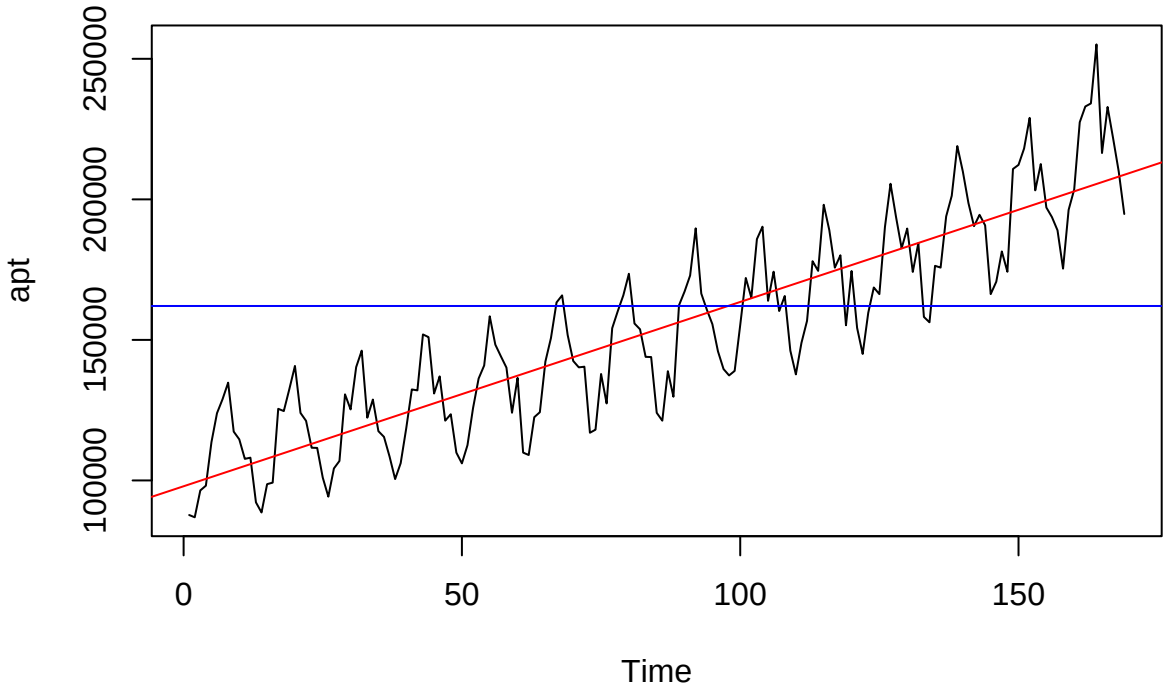


```r
apt = tsdl[[4]][c(1:169)] #the training set

ap.testing = tsdl[[4]][c(170:181)] #testing set
```

Data Visualization

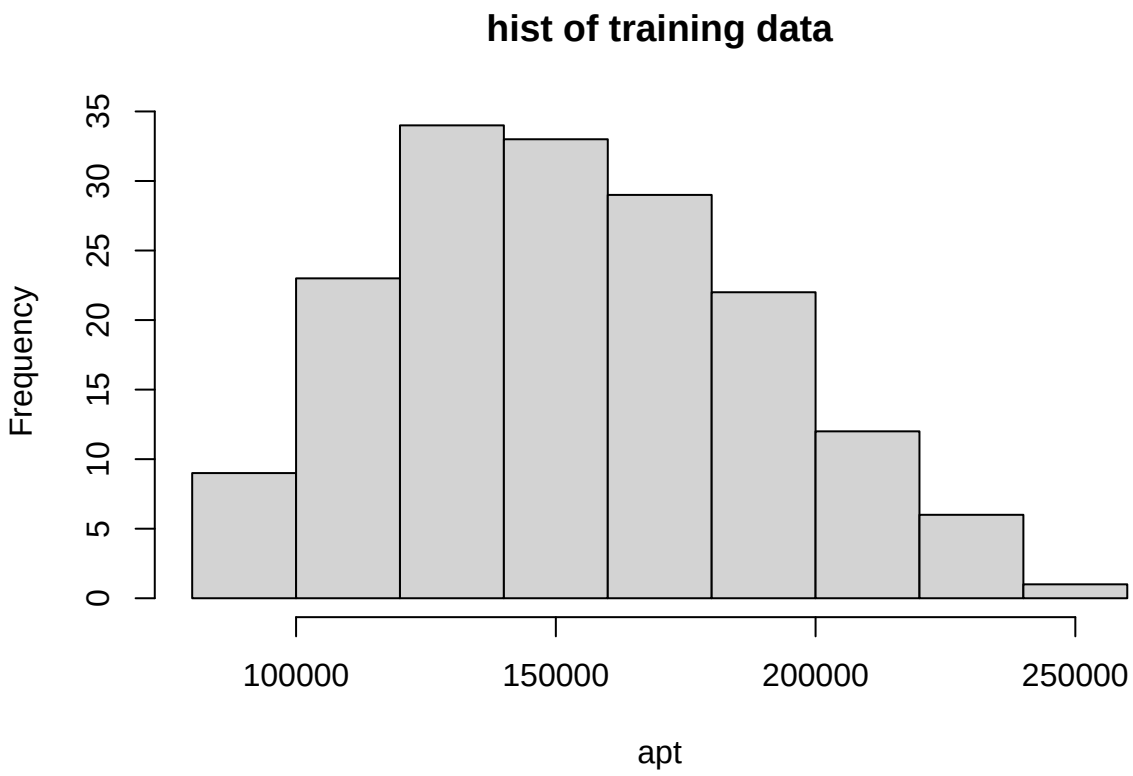
We plot our training data here.

```r
plot.ts(apt)
fit <- lm(apt ~ as.numeric(1:length(apt))); abline(fit, col="red")
abline(h=mean(tsdl[[4]]), col="blue")
```

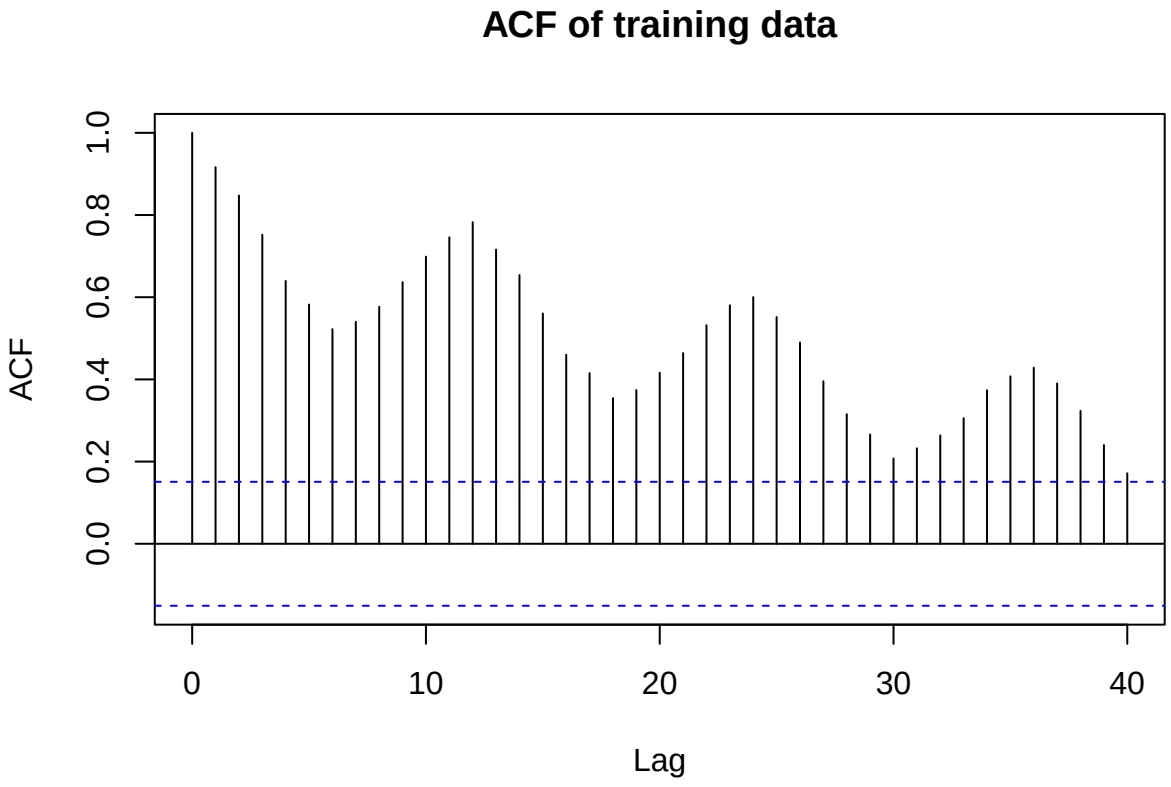


Here is a histogram and acf graph to visualize the training data. We can see the histogram is not normally distributed; it is slightly skewed right. We can also see seasonality in the acf graph. This data is not yet stationary.

```r
hist(apt, main="hist of training data")
```




```
acf(apt, lag.max = 40, main="ACF of training data")
```



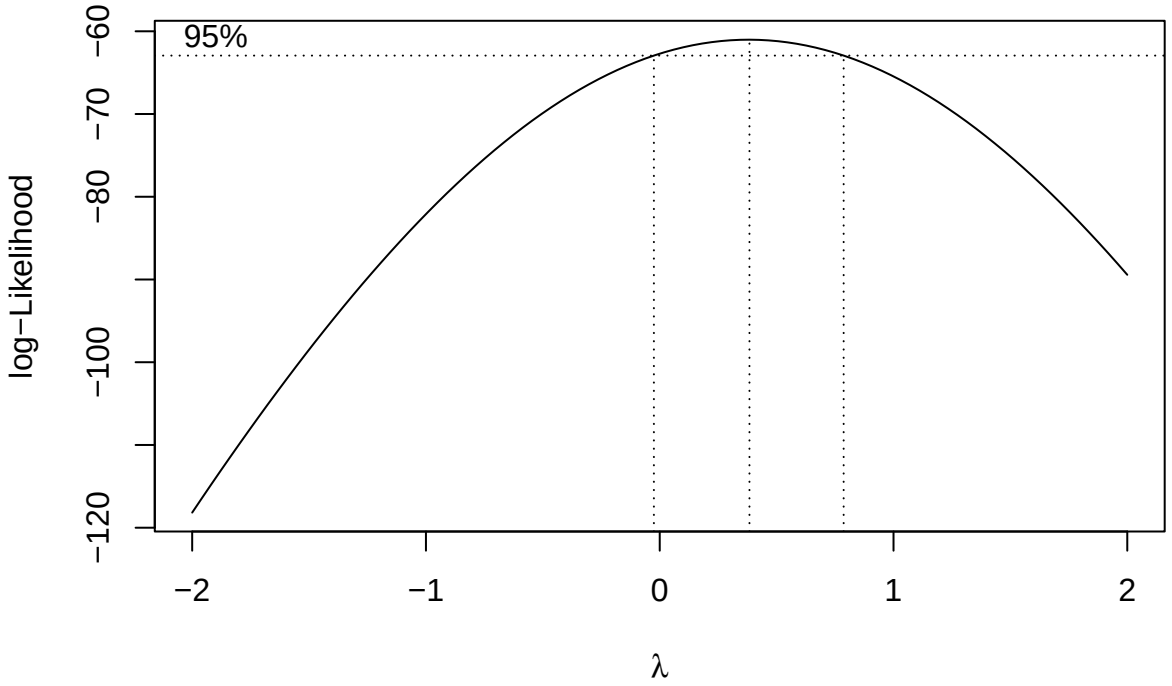
Transformations

We transform the data using a box cox transformation. We are first looking for the ideal lambda value here.

```
length(apt)
```

```
## [1] 169
```

```
t = 1:169
fit = lm(apt ~ t)

bcTransform = boxcox(apt ~ t, plotit = TRUE)
```



The confidence interval barely includes $\lambda = 0$, so the box cox transformation is given by $Y_t = \frac{1}{\lambda}(X_t^\lambda - 1)$. The ideal lambda value is (0.3838384).

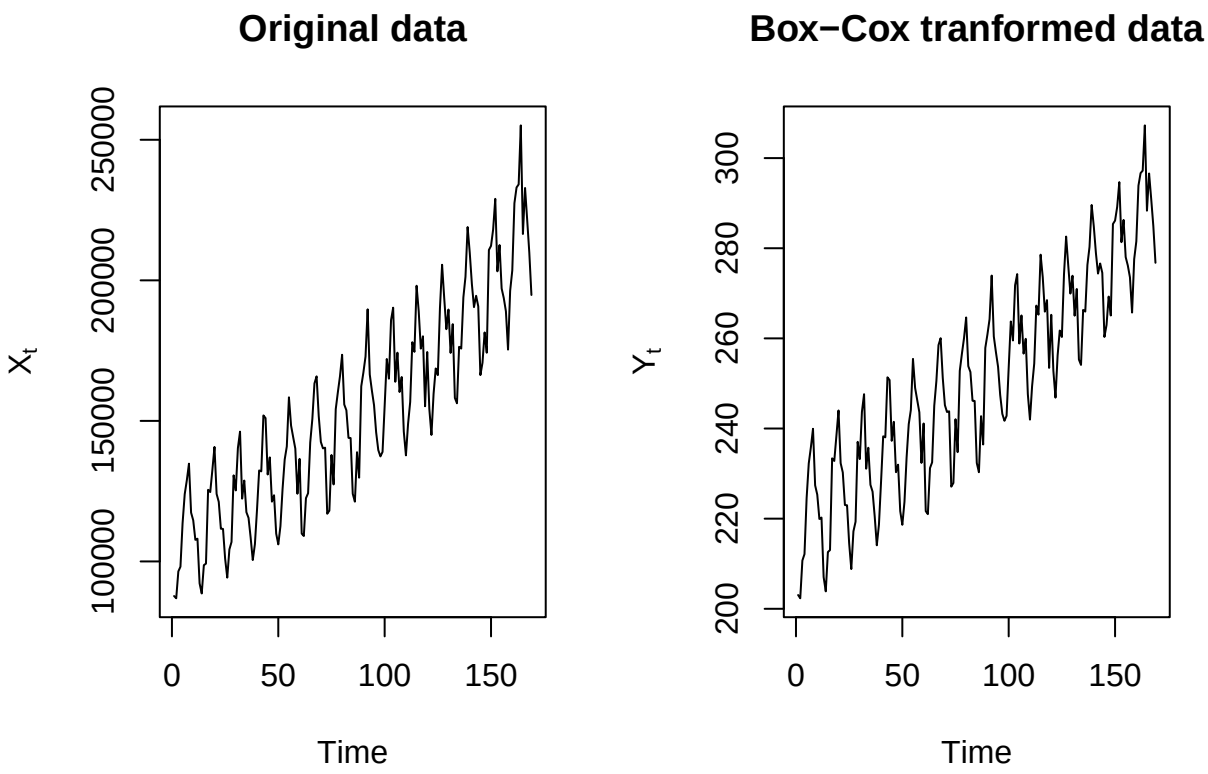
```
lambda = bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
lambda
```

```
## [1] 0.3838384
```

```
apt.bc = (1/lambda)*(apt^lambda-1)
```

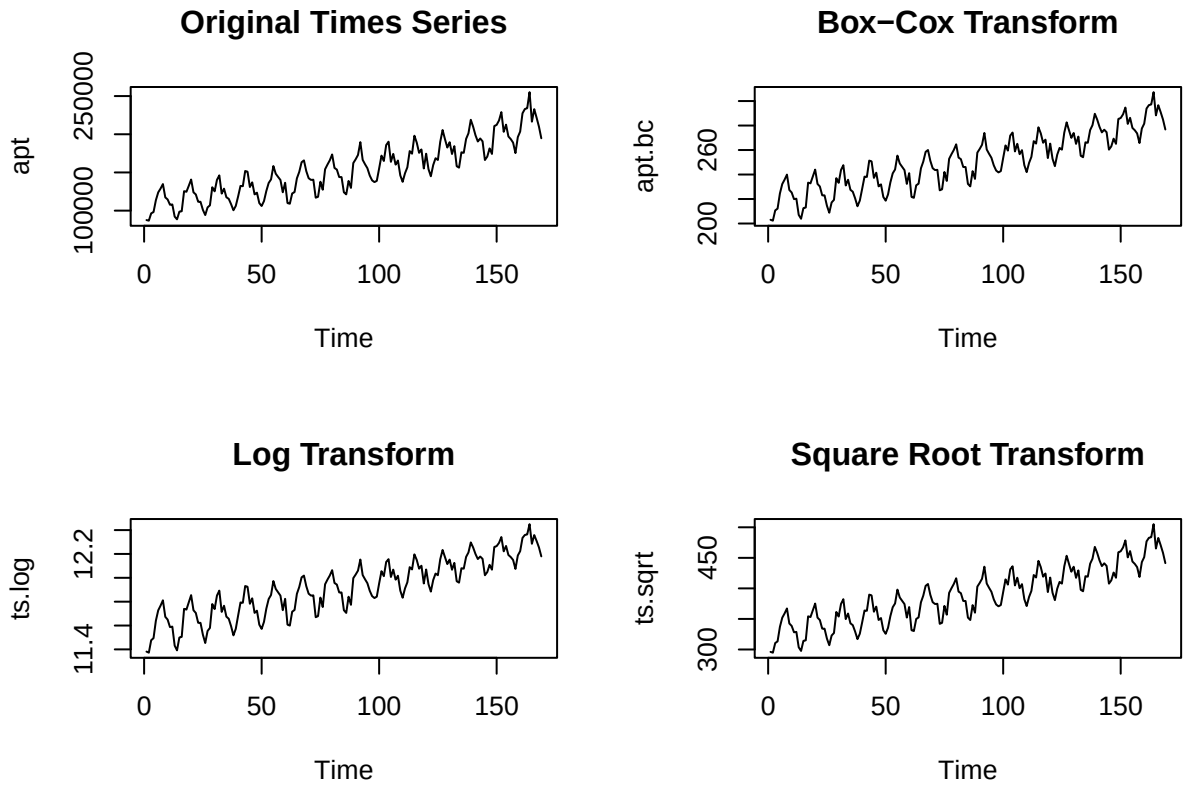
We now plot the box cox transformed data next to the original time series data.

```
op <- par(mfrow = c(1,2))
ts.plot(apt,main = "Original data",ylab = expression(X[t]))
ts.plot(apt.bc,main = "Box-Cox tranformed data", ylab = expression(Y[t]))
```



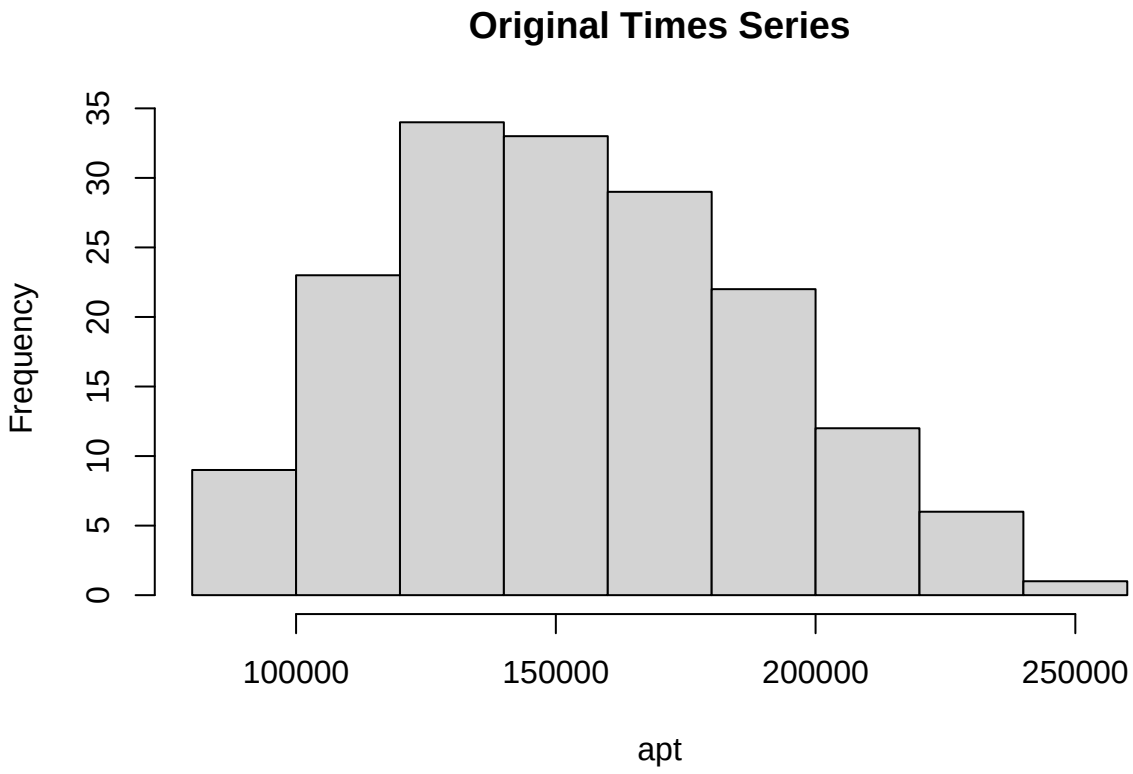
We now use log transformations on the data and square-root transformation and then we plot all the transformed data next to each other so we can see which transformation is the most normal.

```
#log transform
ts.log = log(apt)
# square root transform
ts.sqrt = sqrt(apt)

#Compare transforms
op= par(mfrow=c(2,2))
ts.plot(apt, main = "Original Times Series")
ts.plot(apt.bc, main = "Box-Cox Transform")
ts.plot(ts.log, main = "Log Transform")
ts.plot(ts.sqrt, main = "Square Root Transform")
```



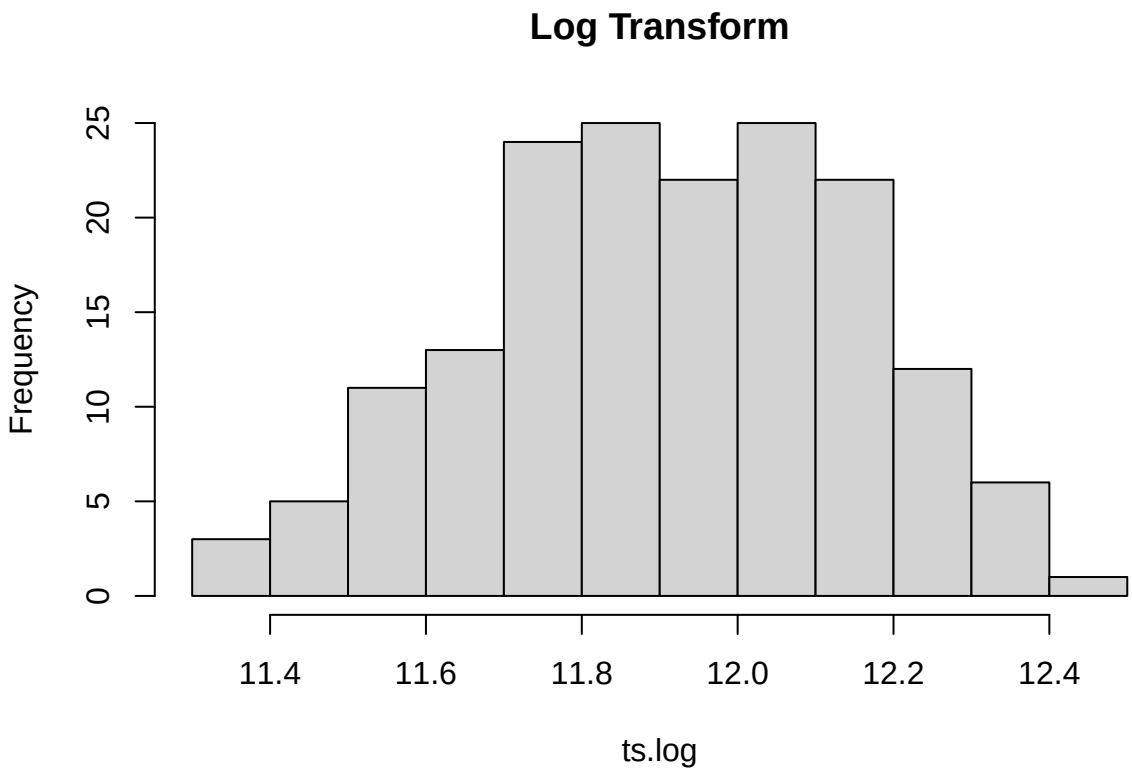
We can see from the 4 histograms that the most normally distributed transformation is the box-cox transformation.

```
hist(apt, main = "Original Times Series")
```

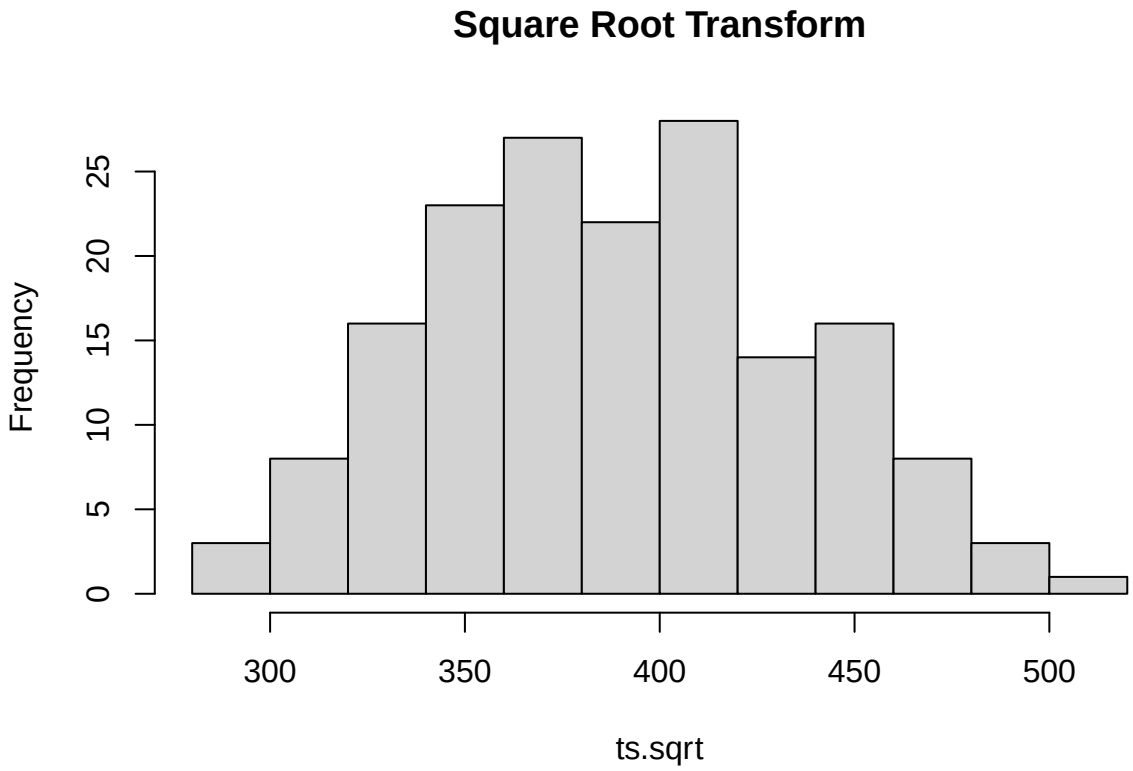


```
hist(apt.bc, main = "Box-Cox Transform")
```


```
hist(ts.log, main = "Log Transform")
```



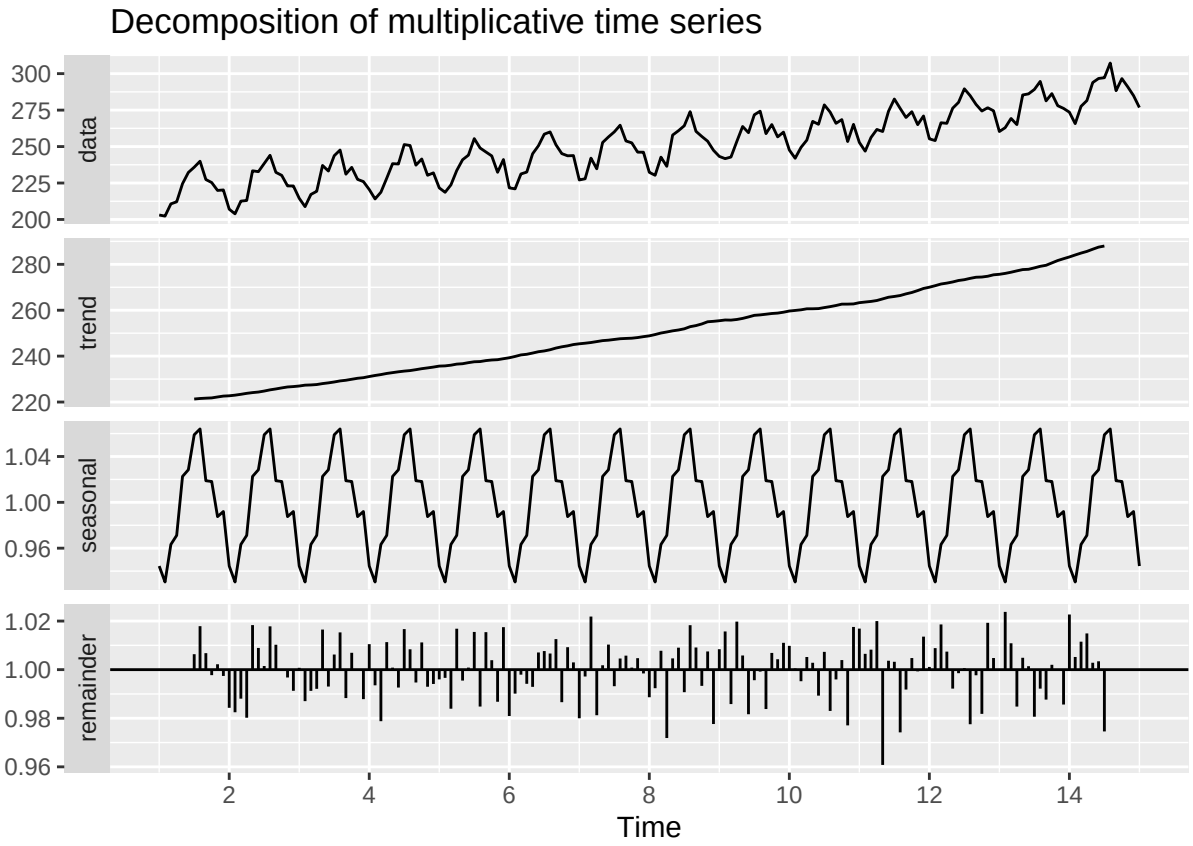
```
hist(ts.sqrt, main = "Square Root Transform")
```



Focusing On Box Cox Transformation

Here we show the decomposition of the time series and we can see that the box cox transformed time series almost has a linear trend and there is definitely seasonality.

```
z<-ts(as.ts(apt.bc),frequency=12)
decomposed_ts<-decompose(z, type='multiplicative')

autoplot(decomposed_ts)
```



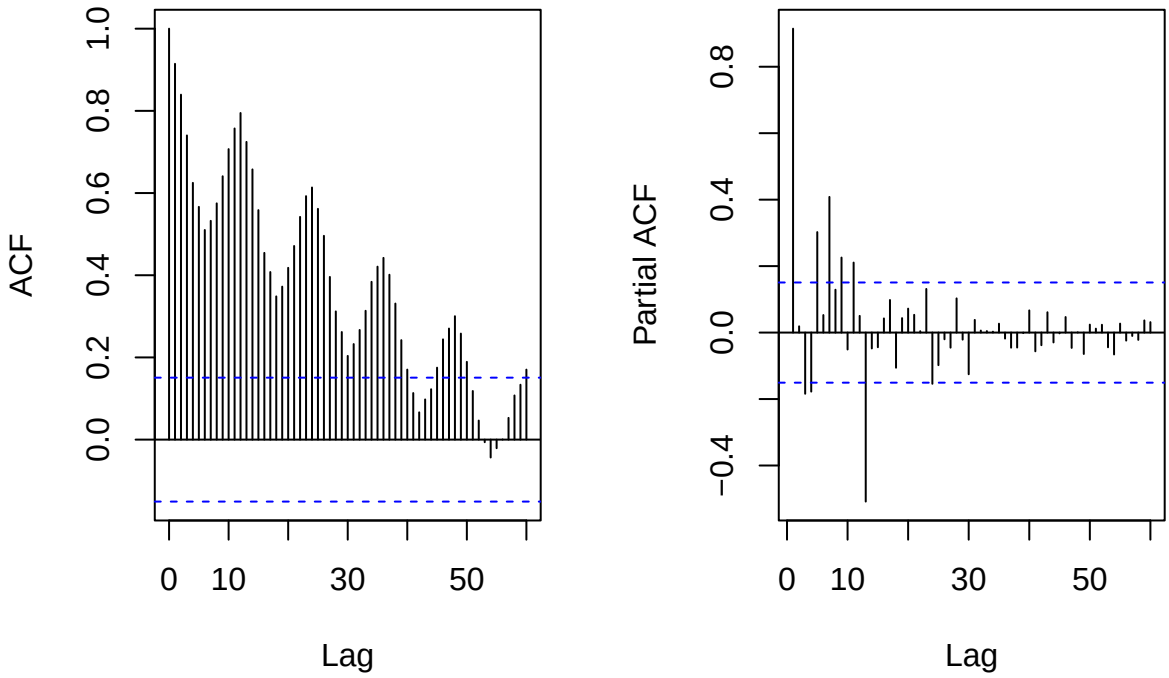
We can take the variance of the box cox transformed data and compare it the data after differencing it by 1 lag and 12 lags.

```
var(apt.bc)
```

```
## [1] 534.4009
```

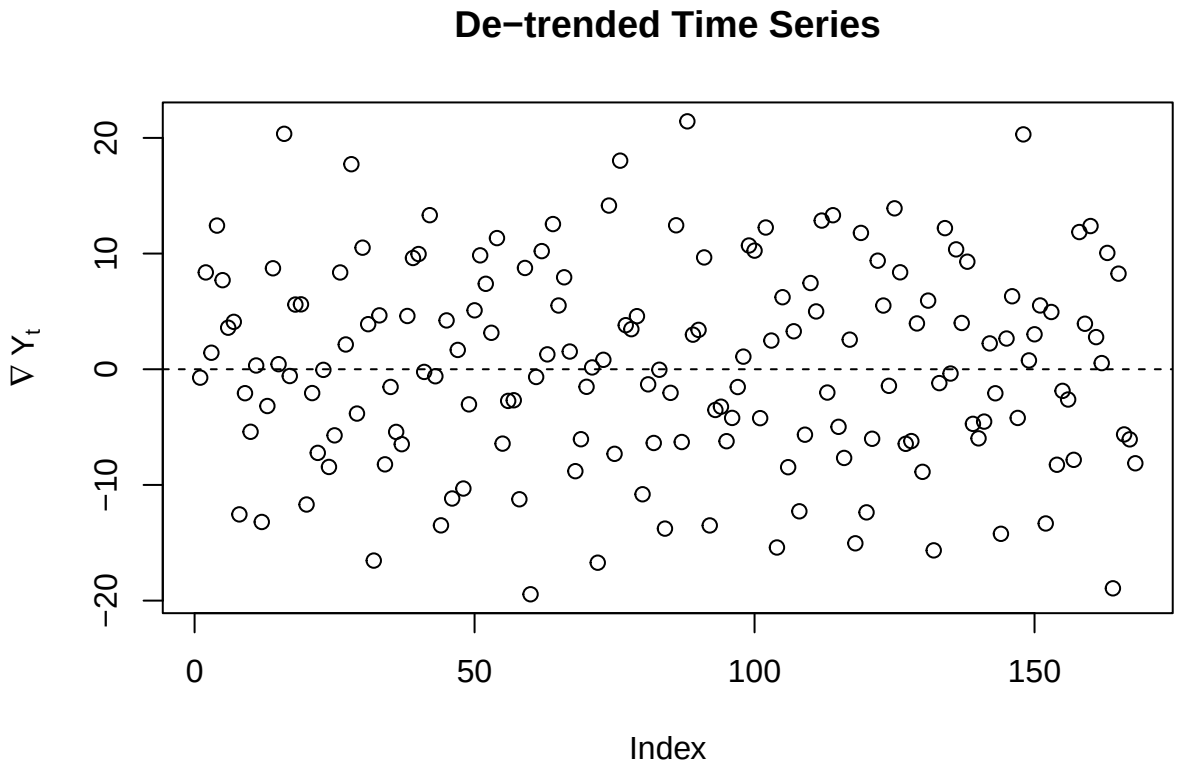
Here we plotted the ACF and PACF. We can see non-stationary because the ACF graph demonstrates seasonality.

```
op = par(mfrow = c(1,2))
acf(apt.bc, lag.max = 60, main = "")
pacf(apt.bc, lag.max = 60, main = "")
title("Box-Cox Transformed Time Series", line = -1, outer=TRUE)
```



First we difference at lag = 1 to remove trend.

```
# Diference at lag = 1 to remove trend component
y1 = diff(apt.bc, 1)
plot(y1,main = "De-trended Time Series",ylab = expression(nabla~Y[t]))
abline(h = 0,lty = 2)
```



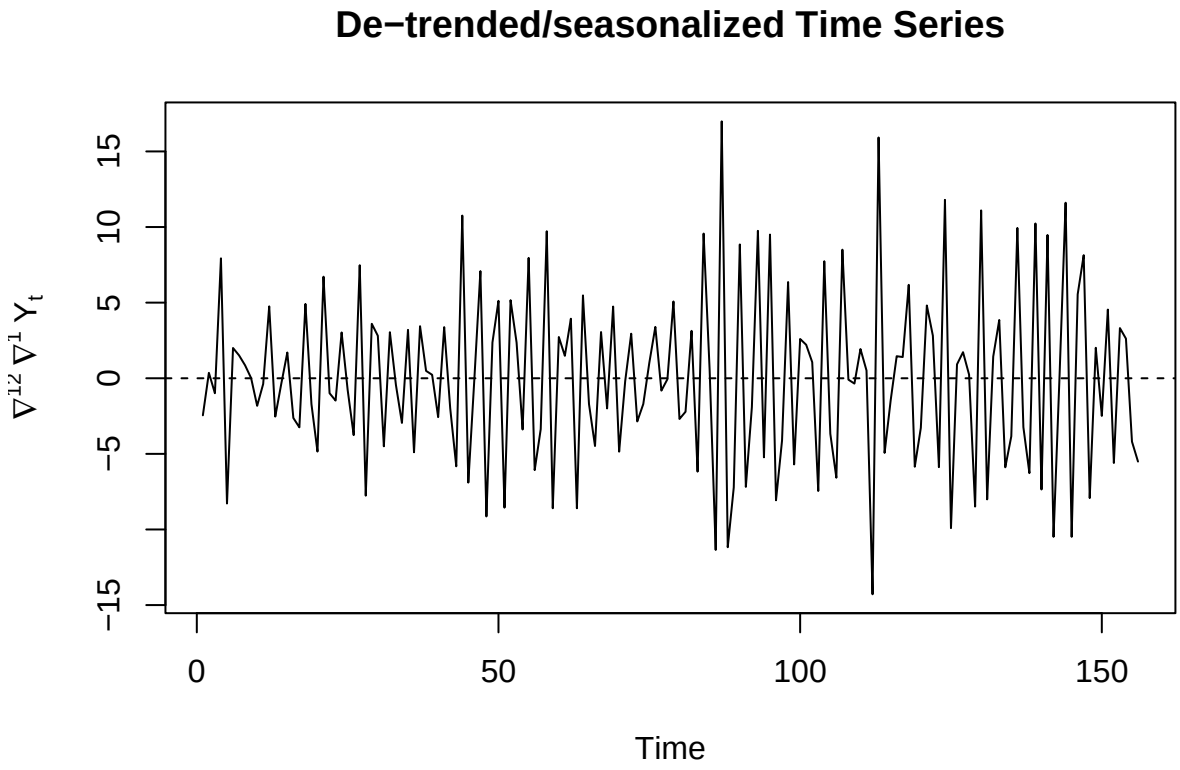
The variance after differencing by lag 1 is 74.053. This is better than the variance of the regular box cox transformed data, which was 543.4009

```
var(y1)
```

```
## [1] 74.0533
```

Next we difference at lag = 12 to remove seasonality.

```
# Diference at lag = 12 (cycle determined by the ACF) to remove seasonal component
y12 = diff(y1, 12)
ts.plot(y12,main = "De-trended/seasonalized Time Series",ylab = expression(nabla^{12}~nabla^{1}~Y[t]))
abline(h = 0,lty = 2)
```



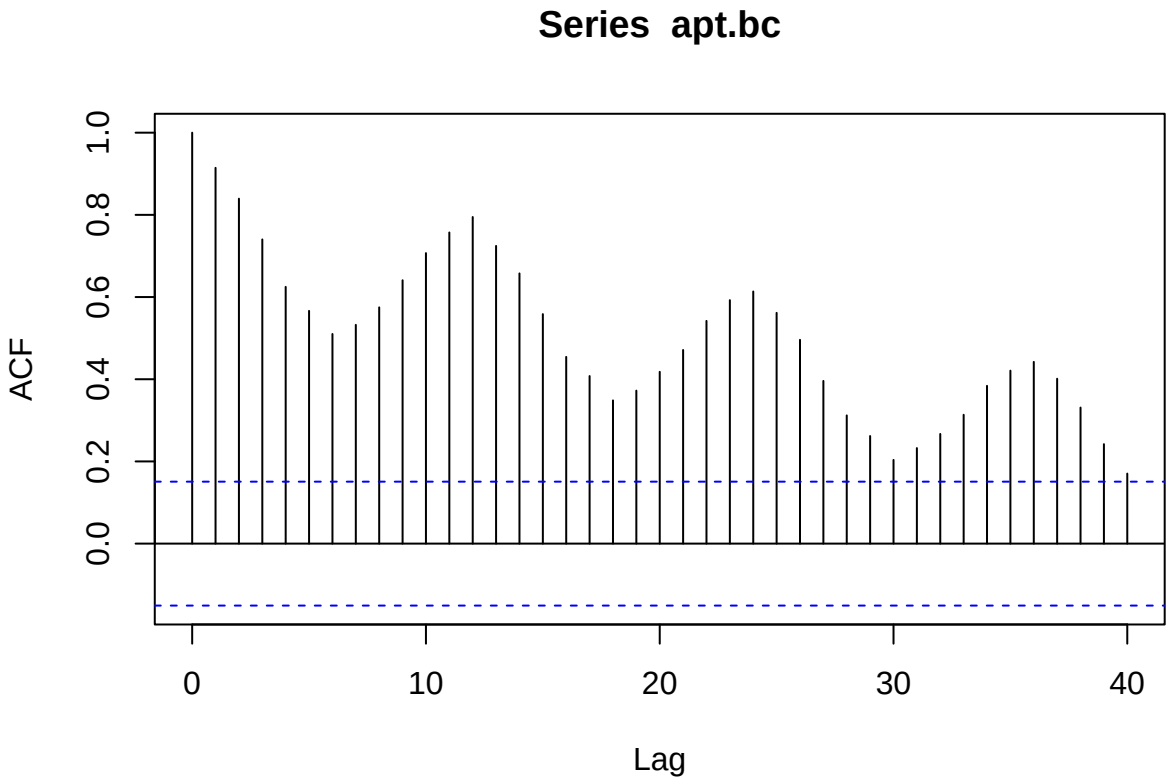
The variance after both lag differences is 34.51169, which is the lowest variance we have achieved.

```
var(y12)
```

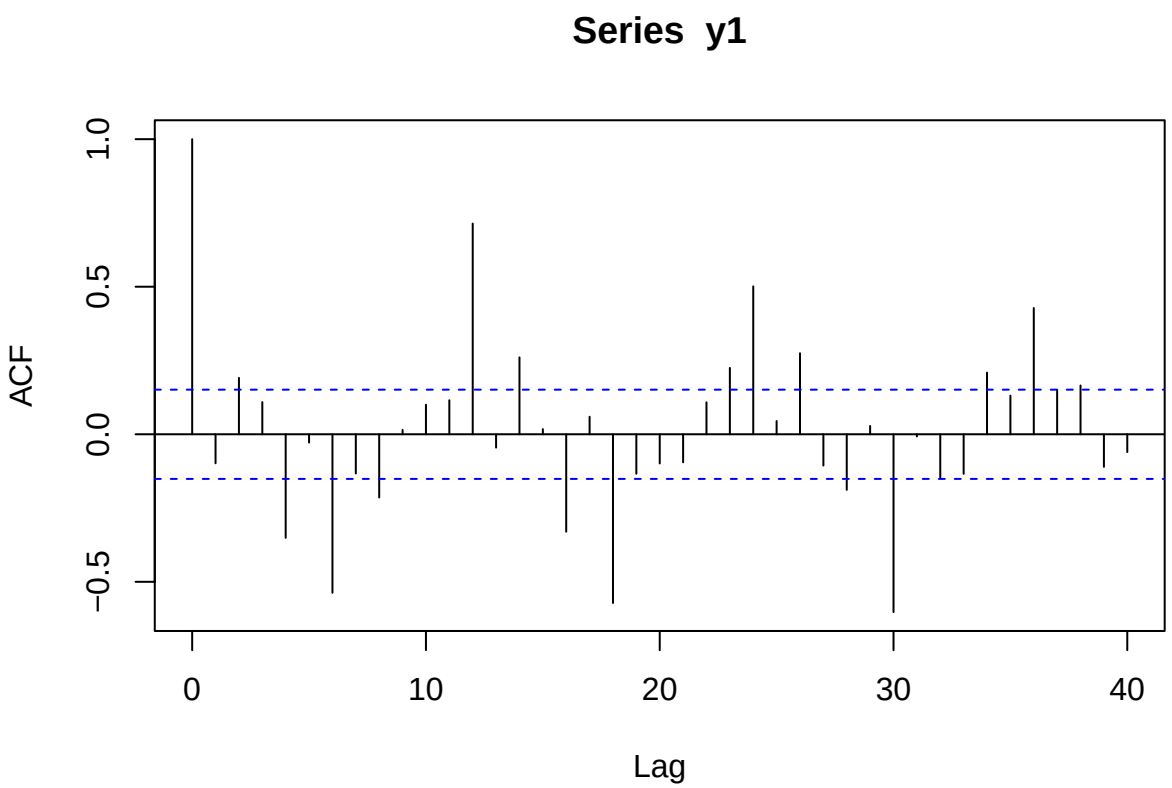
```
## [1] 34.51169
```

Data looks stationary, but lets check ACFs. We can see that the data is most stationary after lag 1, lag 12.

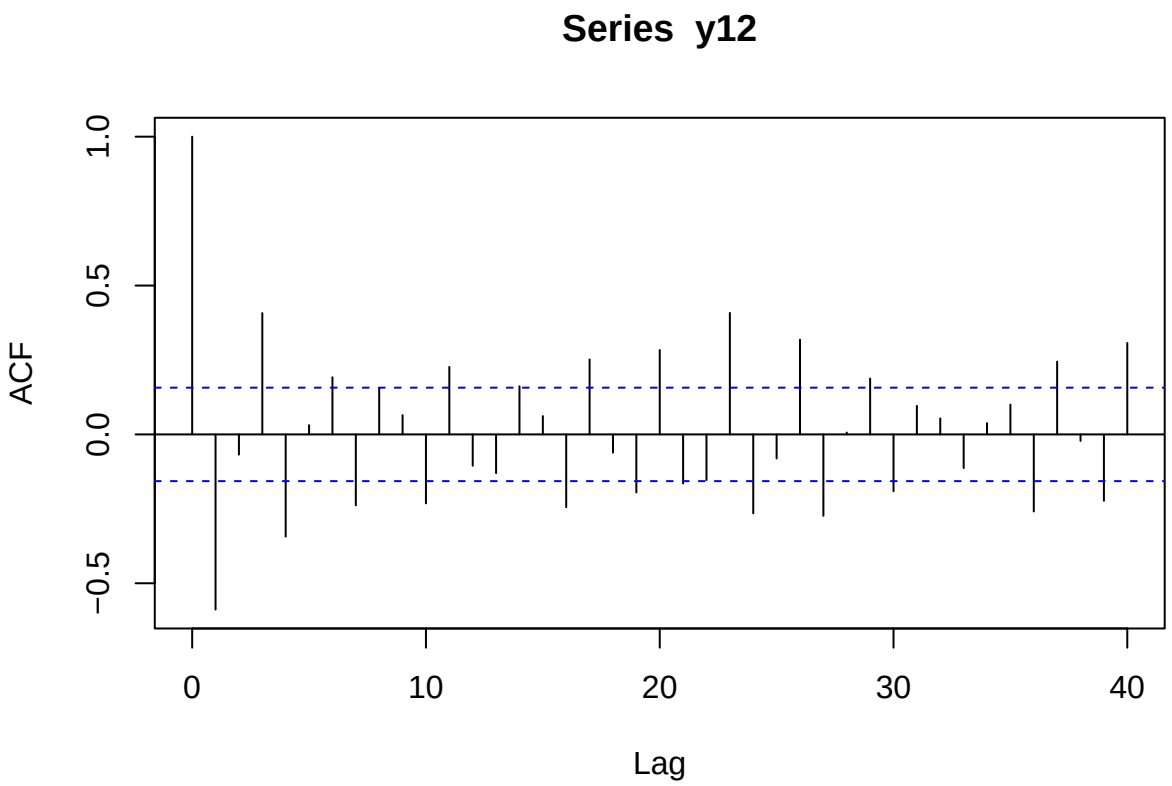
```
acf(apt.bc, lag.max = 40)
```




```
acf(y1, lag.max = 40)
```




```
acf(y12, lag.max = 40)
```



Now see here that even if we difference at lag 12 first then difference at lag 1, then we still get the same results for the final product!

```
#differencing our_data.bc at lag 12 first this time
var(apt.bc)
```

```
## [1] 534.4009
```

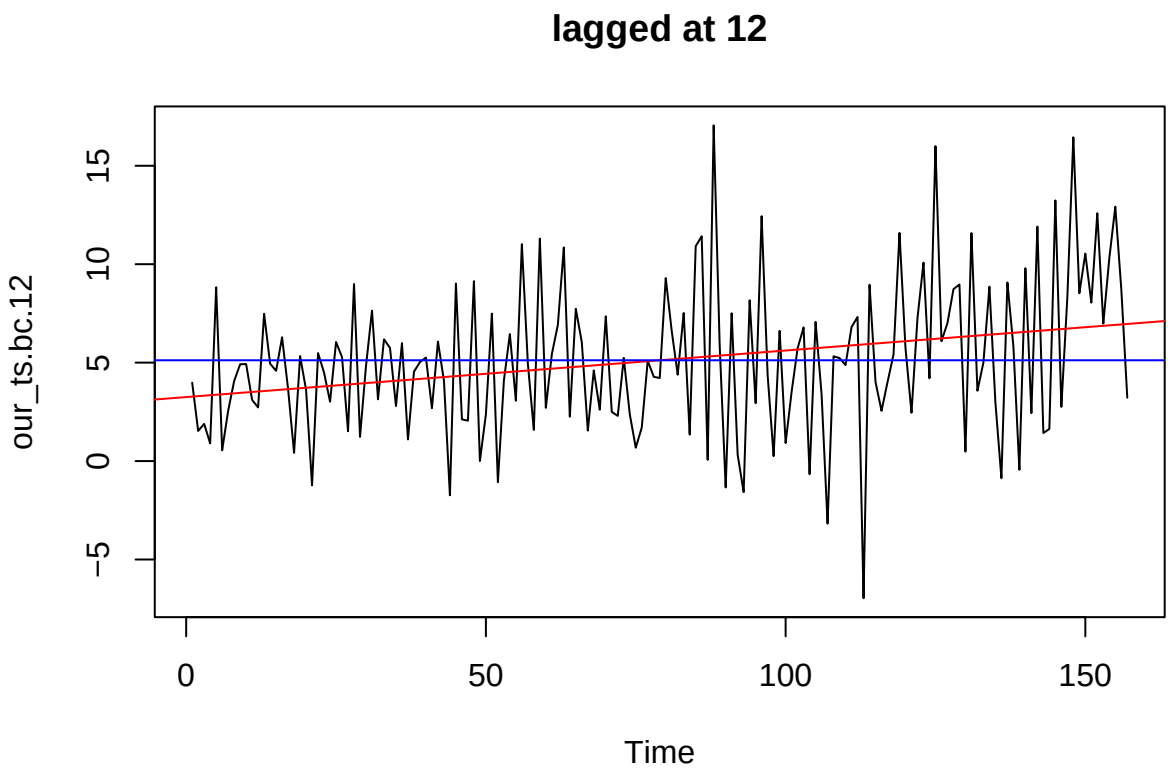
```
our_ts.bc.12<-diff(apt.bc, lag=12)
plot.ts(our_ts.bc.12, main='lagged at 12')
var(our_ts.bc.12)
```

```
## [1] 15.30921
```

```
fit <- lm(our_ts.bc.12 ~ as.numeric(1:length(our_ts.bc.12)))
abline(fit, col="red")
mean(our_ts.bc.12)
```

```
## [1] 5.119186
```

```
abline(h=mean(our_ts.bc.12), col="blue")
```



The red and blue lines are overlapping after differencing by lag 1 and lag 12. This proves the data is stationary. Trend and seasonality have been removed.

```
#now lag 1
var(our_ts.bc.12)
```

```
## [1] 15.30921
```

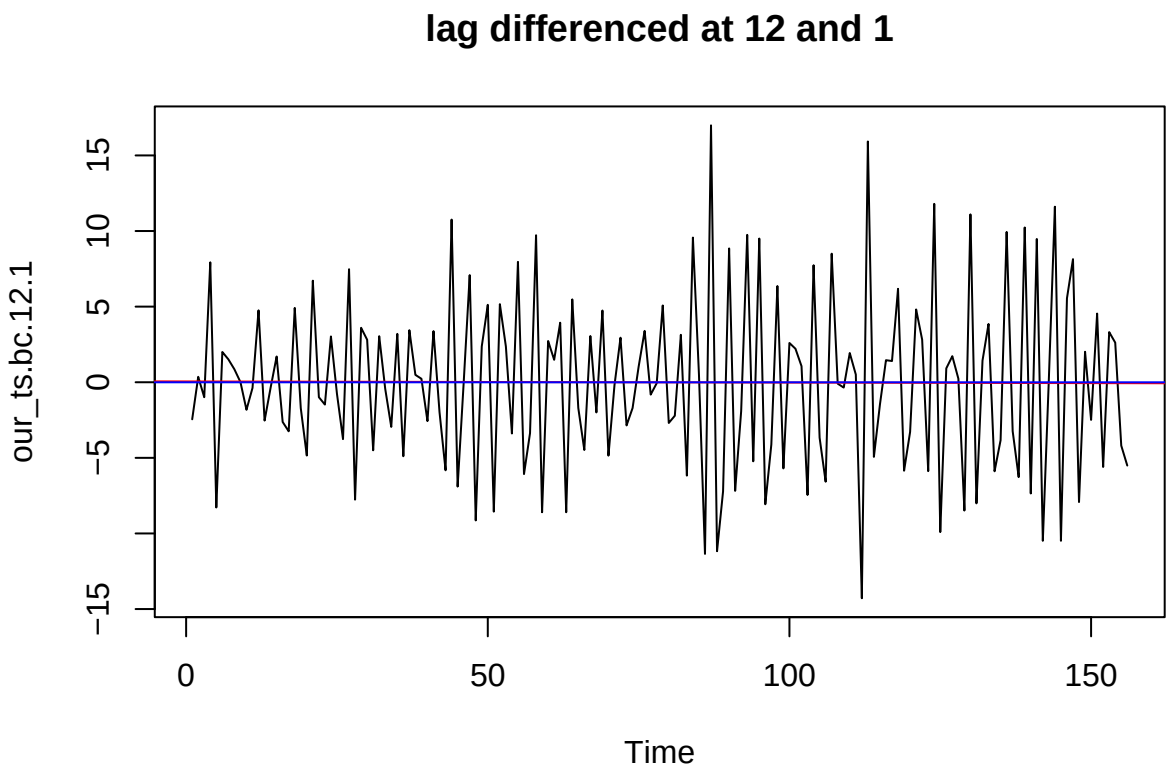
```
our_ts.bc.12.1<-diff(our_ts.bc.12, lag=1)
plot.ts(our_ts.bc.12.1, main='lag differenced at 12 and 1')
var(our_ts.bc.12.1)
```

```
## [1] 34.51169
```

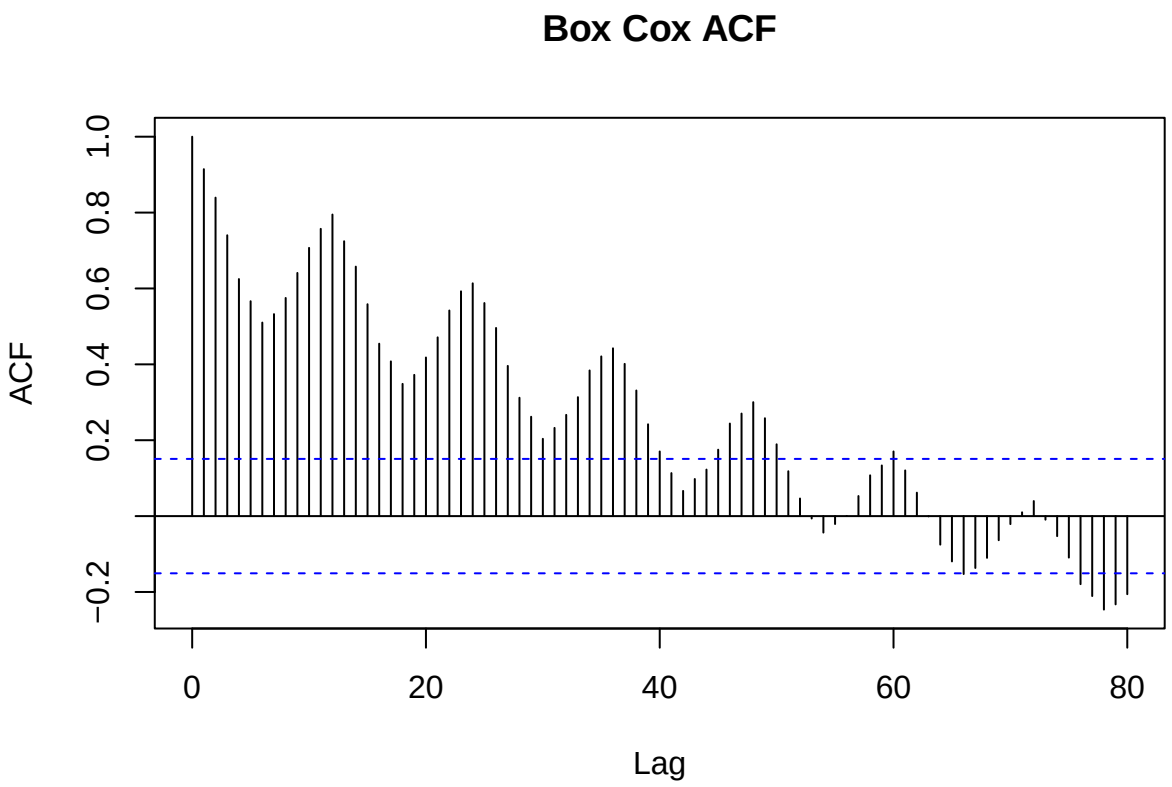
```
fit <- lm(our_ts.bc.12.1 ~ as.numeric(1:length(our_ts.bc.12.1))); abline(fit, col="red")
mean(our_ts.bc.12.1)
```

```
## [1] -0.004897527
```

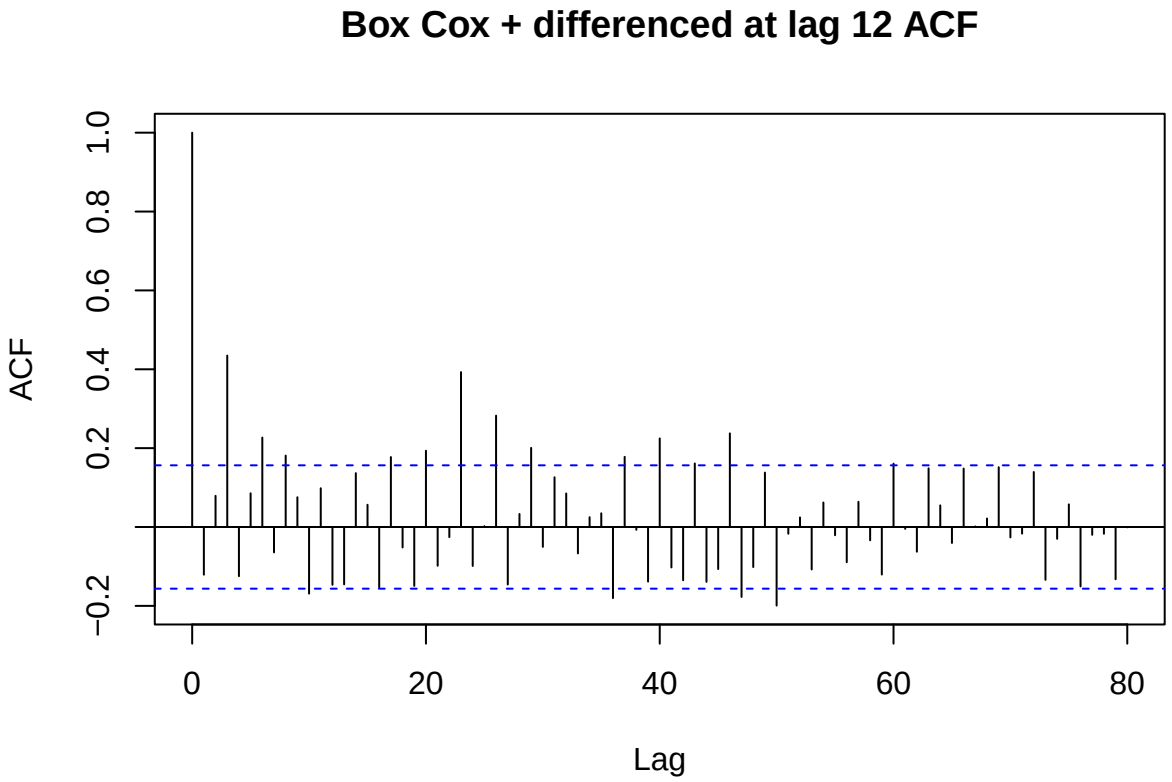
```
abline(h=mean(our_ts.bc.12.1), col="blue")
```



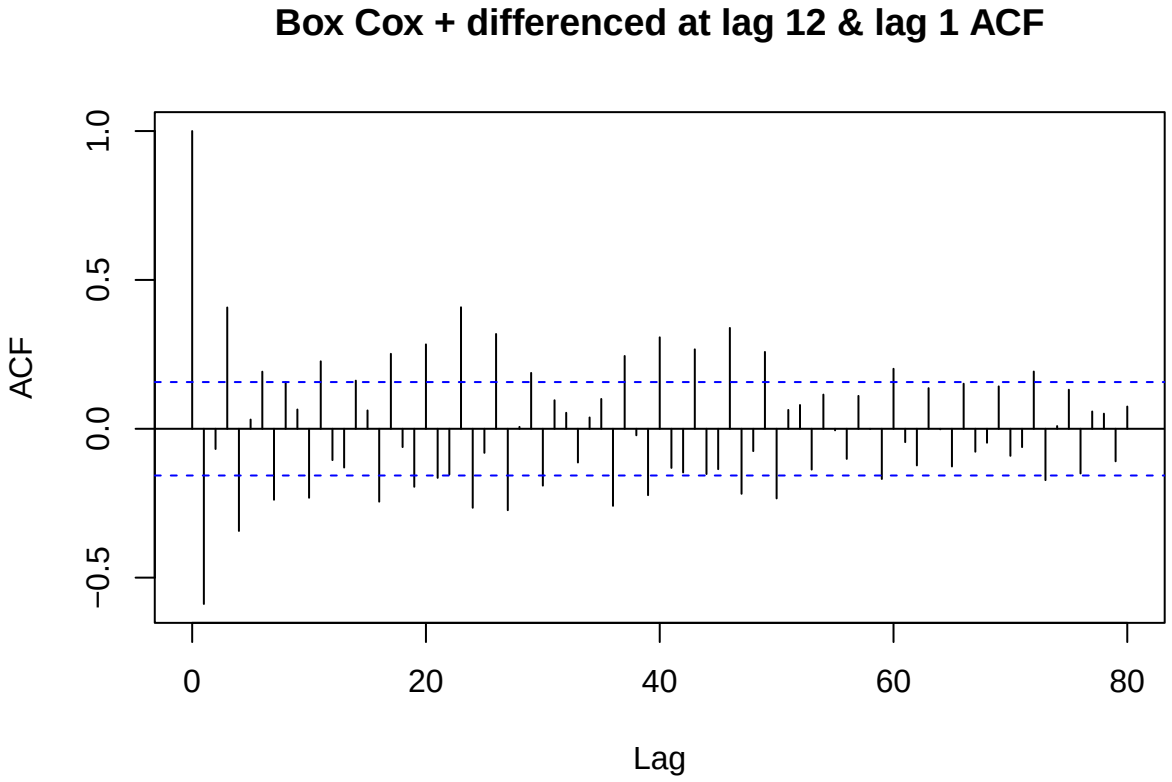

```
acf(apt.bc, lag.max = 80,
    main="Box Cox ACF")
```




```
acf(our_ts.bc.12, lag.max = 80,
    main="Box Cox + differenced at lag 12 ACF")
```



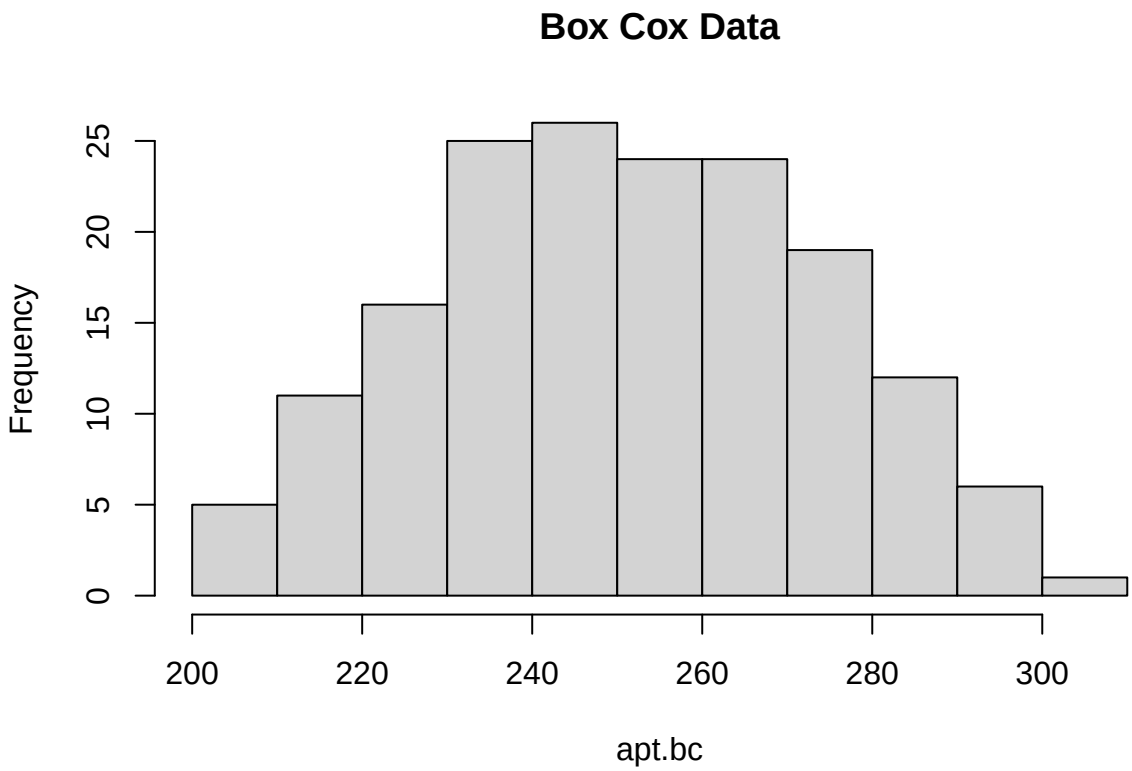

```
acf(our_ts.bc.12.1, lag.max = 80,
    main="Box Cox + differenced at lag 12 & lag 1 ACF")
```



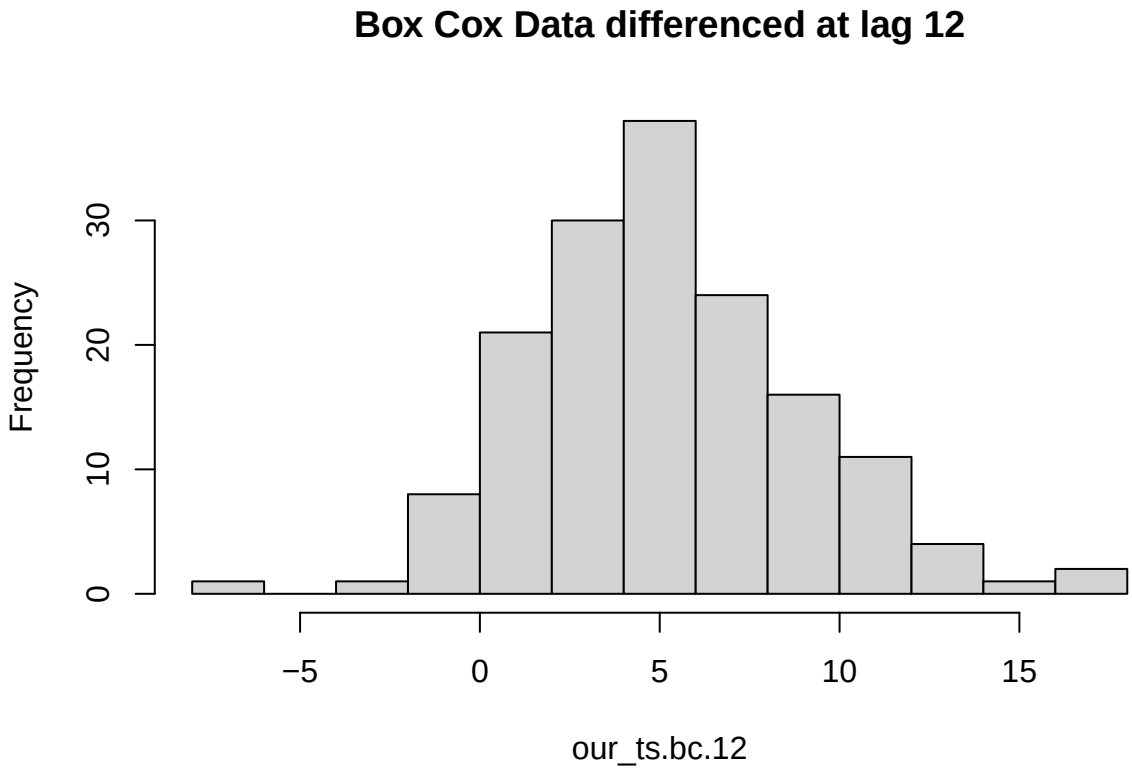
Data Visualization

Now we make histograms on the Box Cox transformed data.

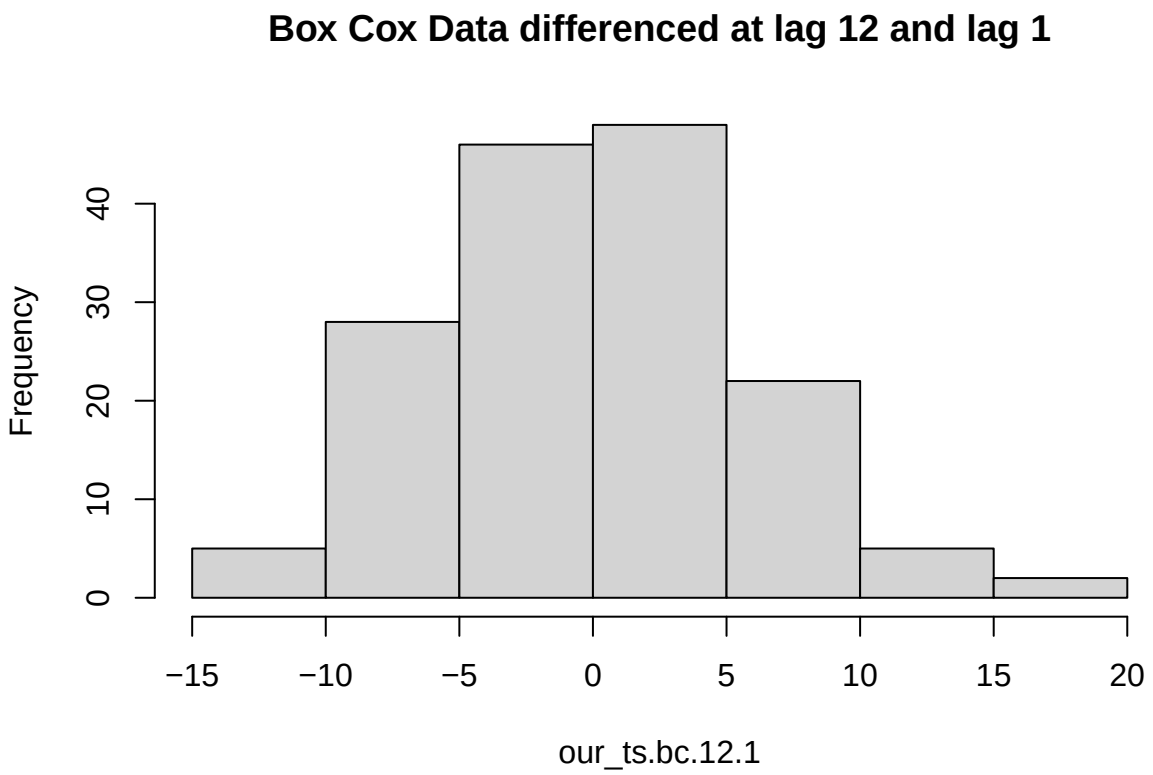
```
hist(apt.bc, main = "Box Cox Data")
```




```
hist(our_ts.bc.12, main = "Box Cox Data differenced at lag 12")
```

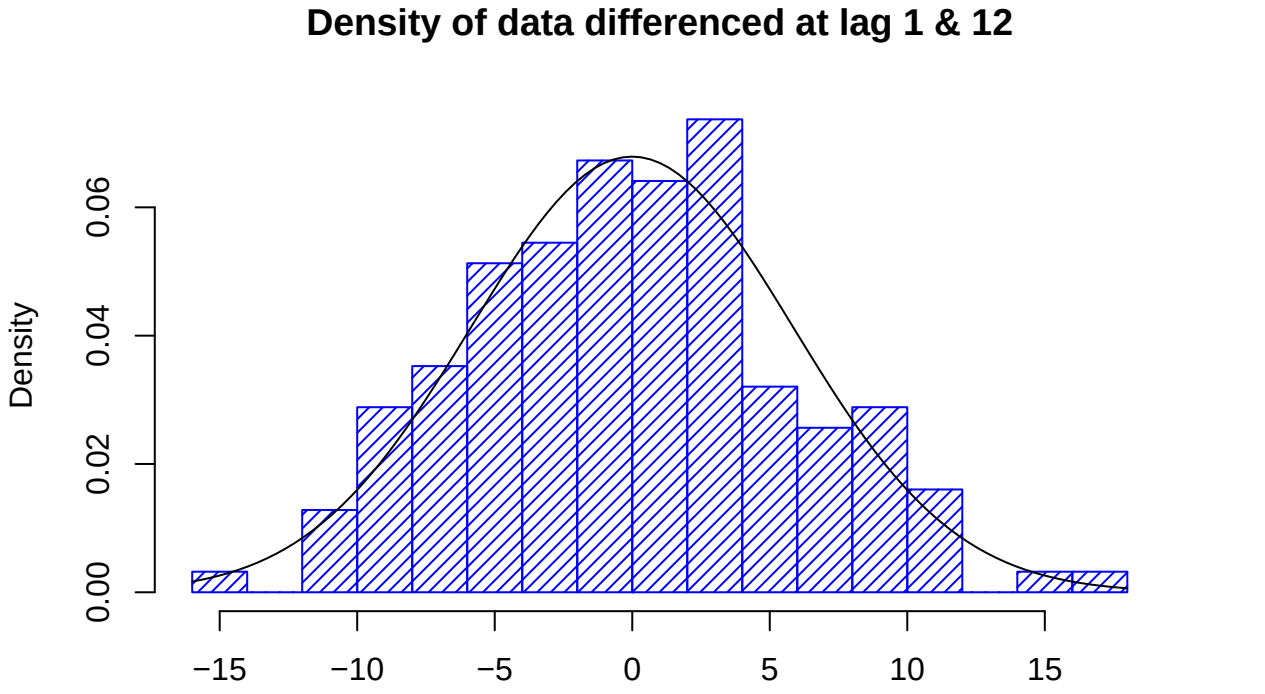


```
hist(our_ts.bc.12.1, main = "Box Cox Data differenced at lag 12 and lag 1")
```

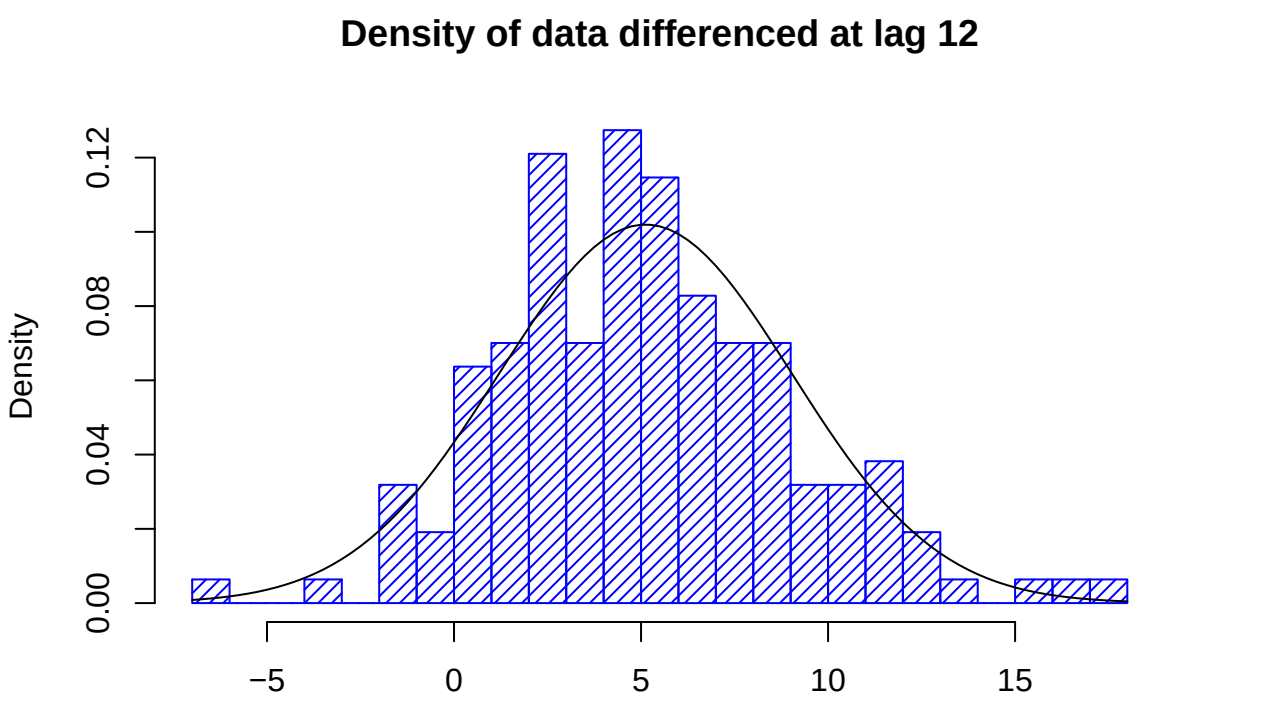



```
wts<-our_ts.bc.12
```

```
hist(our_ts.bc.12.1, density=20,breaks=20,
     main = "Density of data differenced at lag 1 & 12", col="blue", xlab="", prob=TRUE)
m<-mean(our_ts.bc.12.1)
std<- sqrt(var(our_ts.bc.12.1))
curve( dnorm(x,m,std), add=TRUE )
```

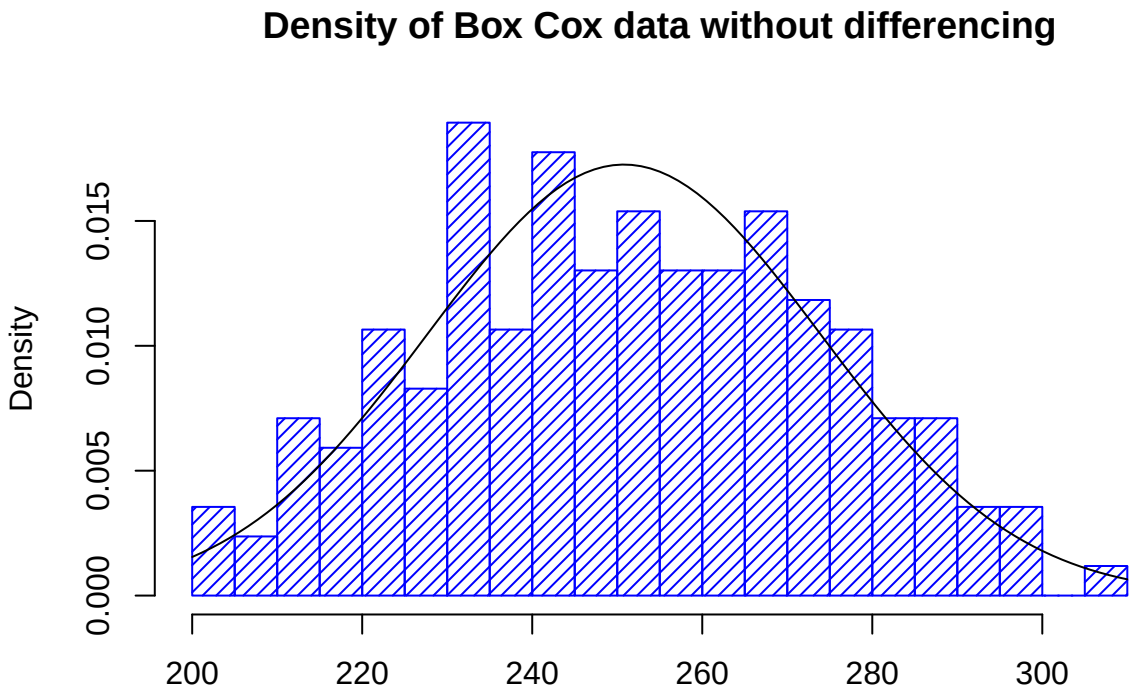



```
hist(wts, density=20,breaks=20,main =
       "Density of data differenced at lag 12", col="blue", xlab="", prob=TRUE)
m<-mean(wts)
std<- sqrt(var(wts))
curve( dnorm(x,m,std), add=TRUE )
```

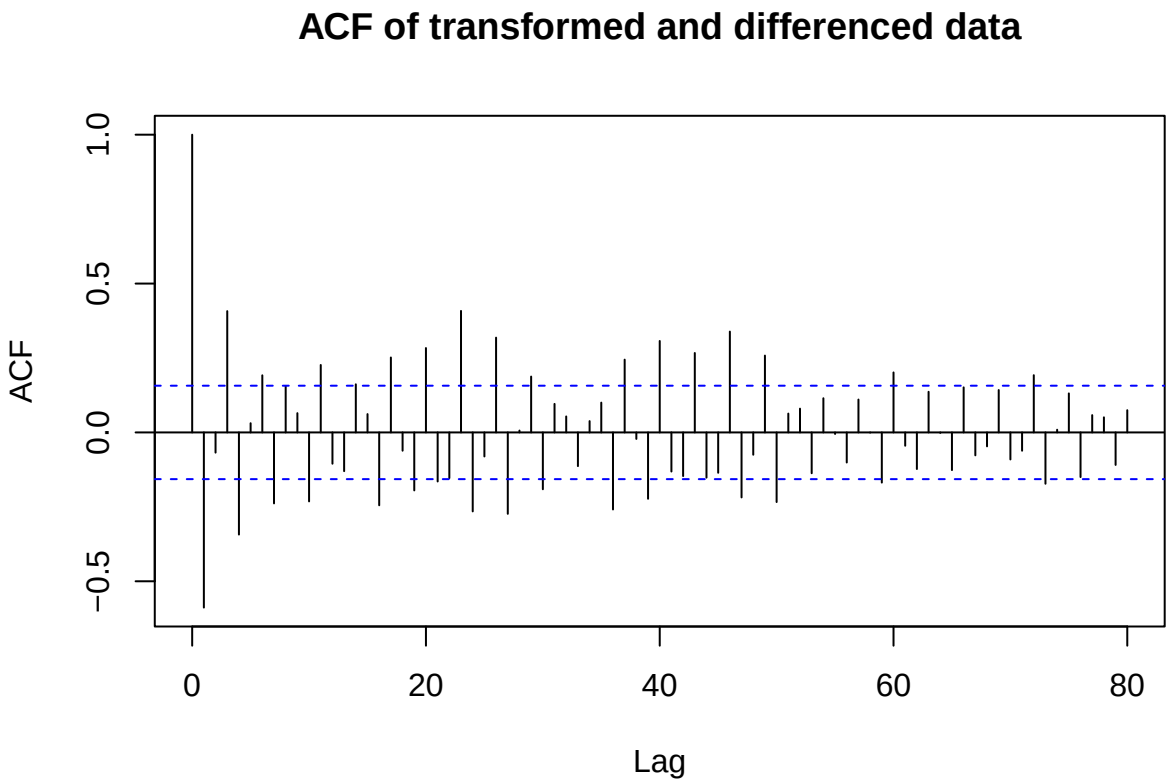



```
hist(apt.bc, density=20,breaks=20,
     main="Density of Box Cox data without differencing",
     col="blue", xlab="", prob=TRUE)
m<-mean(apt.bc)
std<- sqrt(var(apt.bc))
```

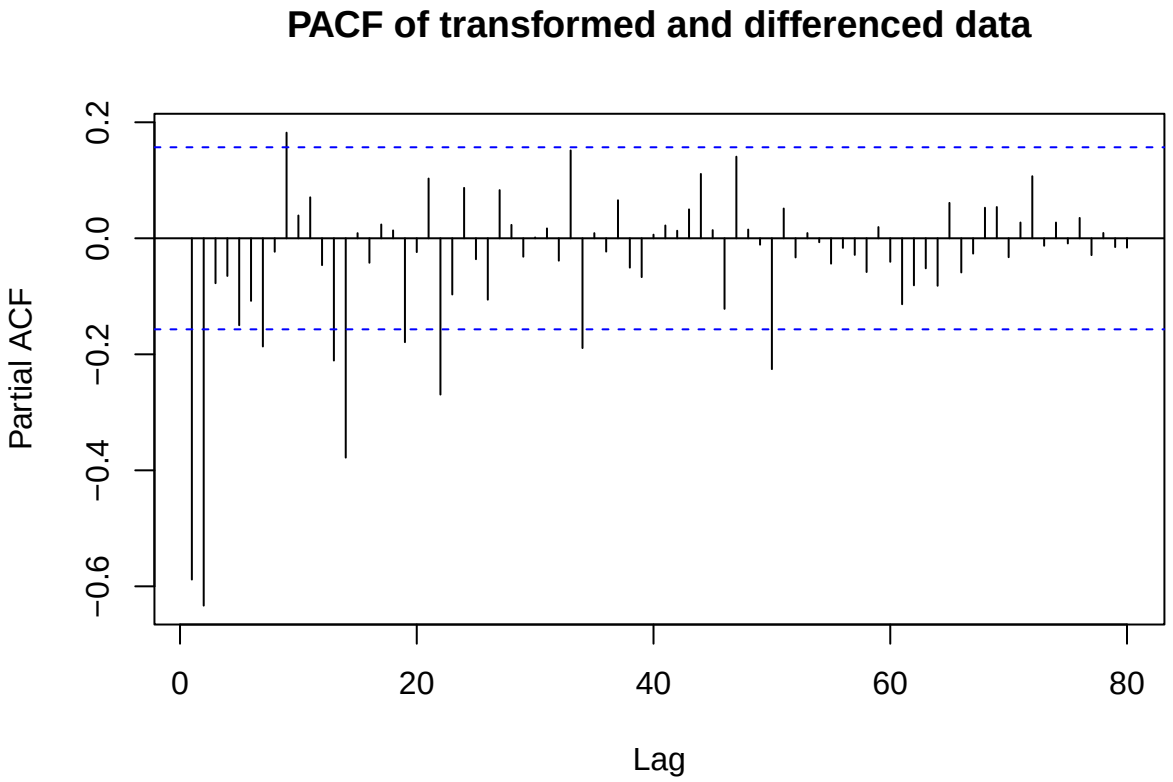
```
curve( dnorm(x,m,std), add=TRUE )
```




```
acf(our_ts.bc.12.1, lag.max = 80,
    main="ACF of transformed and differenced data" )
```




```
pacf(our_ts.bc.12.1, lag.max = 80,
     main="PACF of transformed and differenced data")
```



We will have sarima model because I differenced at lag 12 and at lag 1. Lowercase s is equal to 12 because its the period. Capital D equals 1 because we differenced one time at lag 12. We differenced one time at lag 1 so lowercase d equals 1.

At lag 12, 24, 36 we have a large ACF outside confidence interval. So we can look into uppercase Q equals 1 and 2, 3. Also we have ACFs outside the confidence interval at lags 0,1,3,4 so little q could be 0, 1, or 3, 4.

Check at multiples of S. So we should check 12, 24, 36, etc. The multiples all seem to be in the confidence interval so P is 0.

Now when we look at the PACF.Check at multiples of S. So we should check 12, 24, 36, etc. The multiples all seem to be in the confidence interval so P is 0. For lowercase p we look at lags outside the interval between lag 1 and lag 12. Lowercase p can be 1, 2, 7 or 9, but 7 and 9 are very complicated.

I am going to be using 'our_ts.bc.12.1' because it is more stationary.

Trying Models

Time to try models. We use "our_ts.bc" because we are supposed to use the transformed data, but not the lag differenced data. We are looking for a model with the lowest AICc.

```
arima(apt.bc, order=c(1,1,1),seasonal = list(order =c(1,1,1), period=12), method="ML")
```

```
##
## Call:
## arima(x = apt.bc, order = c(1, 1, 1), seasonal = list(order = c(1, 1, 1), period = 12),
##     method = "ML")
##
## Coefficients:
##           ar1      ma1    sar1     sma1
##       -0.3099  -0.8020  0.3268  -0.9986
## s.e.   0.0865   0.0585  0.0878   0.3530
##
```

```
## sigma^2 estimated as 9.302:  log likelihood = -408.2,  aic = 826.4
```

```
AICc(arima(wts, order=c(1,1,1), seasonal = list(order = c(1,1,1), period = 12), method="ML"))
```

```
## [1] 833.3954
```

```
arima(apt.bc, order=c(1,1,1),seasonal = list(order =c(1,1,2), period=12), method="ML")
```

```
##
## Call:
## arima(x = apt.bc, order = c(1, 1, 1), seasonal = list(order = c(1, 1, 2), period = 12),
##     method = "ML")
##
## Coefficients:
```

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

```
##           ar1      ma1    sar1     sma1     sma2
##       -0.2941  -0.7955  0.1551  -0.8136  -0.1849
## s.e.   0.0897   0.0615  0.2110      NaN      NaN
##
## sigma^2 estimated as 9.181:  log likelihood = -407.74,  aic = 827.49
```

```
AICc(arima(apt.bc, order=c(1,1,1), seasonal = list(order = c(1,1,2), period = 12), method="ML"))
```

```
## [1] 828.0515
```

```
arima(apt.bc, order=c(1,1,3),seasonal = list(order =c(1,1,1), period=12), method="ML")
```

```
##
## Call:
## arima(x = apt.bc, order = c(1, 1, 3), seasonal = list(order = c(1, 1, 1), period = 12),
##     method = "ML")
##
## Coefficients:
##           ar1      ma1      ma2     ma3    sar1     sma1
##       -0.8403  -0.1530  -0.8094  0.3435  0.3219  -0.9071
## s.e.   0.0495   0.0846   0.0737  0.0773  0.1145   0.1439
##
## sigma^2 estimated as 9.107:  log likelihood = -403.31,  aic = 820.61
```

```
AICc(arima(apt.bc, order=c(1,1,3), seasonal = list(order = c(1,1,1), period = 12), method="ML"))
```

```
## [1] 821.3704
```

```
# Second best
b.sarima<-arima(apt.bc, order=c(3,1,1),seasonal = list(order =c(1,1,1), period=12), method="ML")
b.sarima
```

```
##
## Call:
## arima(x = apt.bc, order = c(3, 1, 1), seasonal = list(order = c(1, 1, 1), period = 12),
##     method = "ML")
##
## Coefficients:
##           ar1     ar2    ar3      ma1    sar1     sma1
##       -0.2319  0.0450  0.456  -0.8978  0.1694  -0.8207
## s.e.   0.1005  0.1054  0.091   0.0575  0.1228   0.1034
##
## sigma^2 estimated as 8.31:  log likelihood = -392.81,  aic = 799.61
```

```
AICc(arima(apt.bc, order=c(3,1,1), seasonal = list(order = c(1,1,1), period = 12), method="ML"))
```

```
## [1] 800.3682
```

```
arima(apt.bc, order=c(0,1,1),seasonal = list(order =c(1,1,1), period=12), method="ML")
```

```
##
## Call:
## arima(x = apt.bc, order = c(0, 1, 1), seasonal = list(order = c(1, 1, 1), period = 12),
##     method = "ML")
##
## Coefficients:
##           ma1    sar1     sma1
##       -0.8862  0.3276  -1.0000
## s.e.   0.0345  0.0885   0.1169
##
## sigma^2 estimated as 10.01:  log likelihood = -414.15,  aic = 836.3
```

```
AICc(arima(apt.bc, order=c(0,1,1), seasonal = list(order = c(1,1,1), period = 12), method="ML"))
```

```
## [1] 836.5601
```

```
arima(apt.bc, order=c(0,1,1),seasonal = list(order =c(0,1,1), period=12), method="ML")
```

```
##
## Call:
## arima(x = apt.bc, order = c(0, 1, 1), seasonal = list(order = c(0, 1, 1), period = 12),
##     method = "ML")
##
## Coefficients:
##           ma1     sma1
##       -0.8856  -0.8372
## s.e.   0.0336   0.0991
##
## sigma^2 estimated as 11.6:  log likelihood = -420.69,  aic = 847.38
```

```
AICc(arima(apt.bc, order=c(0,1,1), seasonal = list(order = c(0,1,1), period = 12), method="ML"))
```

```
## [1] 847.5348
```

```
#third best
b2.sarima<-arima(apt.bc, order=c(2,1,2),
                 seasonal = list(order =c(0,1,2), period=12), method="ML")
b2.sarima
```

```
##
## Call:
## arima(x = apt.bc, order = c(2, 1, 2), seasonal = list(order = c(0, 1, 2), period = 12),
##     method = "ML")
##
## Coefficients:
##           ar1      ar2      ma1      ma2     sma1     sma2
##       -0.9544  -0.6101  -0.1365  -0.1289  -0.6615  -0.1088
## s.e.   0.1028   0.0884   0.1319   0.1700   0.0910   0.0865
##
## sigma^2 estimated as 8.746:  log likelihood = -396.32,  aic = 806.64
```

```
AICc(arima(apt.bc, order=c(2,1,2),
           seasonal = list(order = c(0,1,2), period = 12), method="ML"))
```

```
## [1] 807.3926
```

Here we can hold ar2 constant and that actually ends up being our best model.

```
# Best
abs<-arima(apt.bc, order=c(3,1,1),seasonal =
             list(order =c(1,1,1), period=12),
           fixed = c(NA, 0, NA, NA, NA, NA), method="ML")
```

```
## Warning in arima(apt.bc, order = c(3, 1, 1), seasonal = list(order = c(1, : some
## AR parameters were fixed: setting transform.pars = FALSE
```

```
abs
```

```
##
## Call:
## arima(x = apt.bc, order = c(3, 1, 1), seasonal = list(order = c(1, 1, 1), period = 12),
##     fixed = c(NA, 0, NA, NA, NA, NA), method = "ML")
##
## Coefficients:
##           ar1  ar2     ar3      ma1    sar1     sma1
##       -0.2601    0  0.4339  -0.8796  0.1622  -0.8133
## s.e.   0.0783    0  0.0758   0.0477  0.1223   0.1011
##
## sigma^2 estimated as 8.338:  log likelihood = -392.89,  aic = 797.79
```

```
AICc(arima(apt.bc, order=c(3,1,1), seasonal =
             list(order = c(1,1,1), period = 12),
           fixed = c(NA, 0, NA, NA, NA, NA), method="ML"))
```

```
## Warning in arima(apt.bc, order = c(3, 1, 1), seasonal = list(order = c(1, : some
## AR parameters were fixed: setting transform.pars = FALSE
```

```
## [1] 798.3512
```

Models

Model A is the best and Model B is the second best. They are both mixed models. In model A ar2 is held constant.

let Y(X)=$\frac{1}{\lambda}(X_t^{\lambda}-1)$

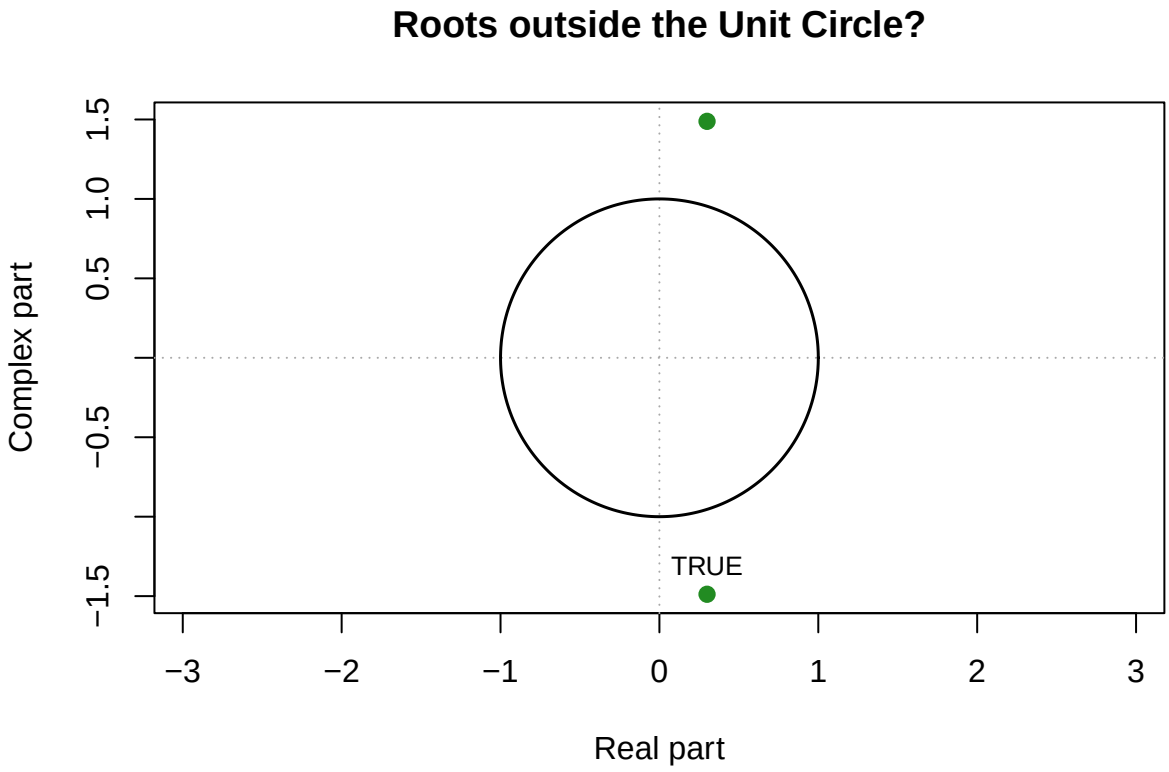
(A) $(1-.2601_{.0783}B+.4339_{.0758}B^3)(1+.1622_{.1223}B^{12})\nabla_1\nabla_{12}Y(U_t)=(1-.8796_{0.0477}B)(1-.8133_{.1011}B^{12})Z_t$ $\hat{\theta}_z^2=8.338$

(B) $(1-.2319_{.1005}B+.0450_{.1054}B^2+.456_{.091}B^3)(1+.1694_{.1228}B^{12}B)\nabla_1\nabla_{12}Y(U_t)=(1-.8978_{0.0575}B)(1-.8207_{.1034}B^{12})Z_t$ $\hat{\theta}_z^2=8.31$

Check if Models are Stationary and Invertible

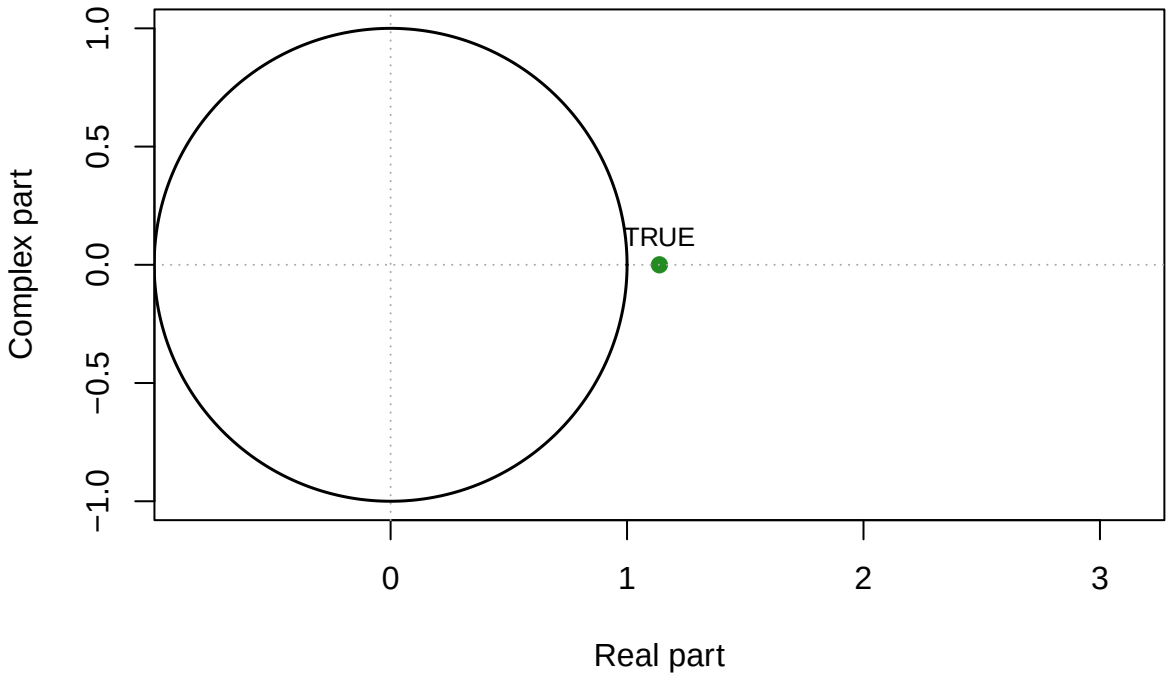
```
#Checking Stationarity for A
uc.check(pol_=c(1, -.2601, .4339, 0),plot_output = TRUE)
```

```
##       real   complex outside
## 1 0.299723  1.488235    TRUE
## 2 0.299723 -1.488235    TRUE
## *Results are rounded to 6 digits.
```



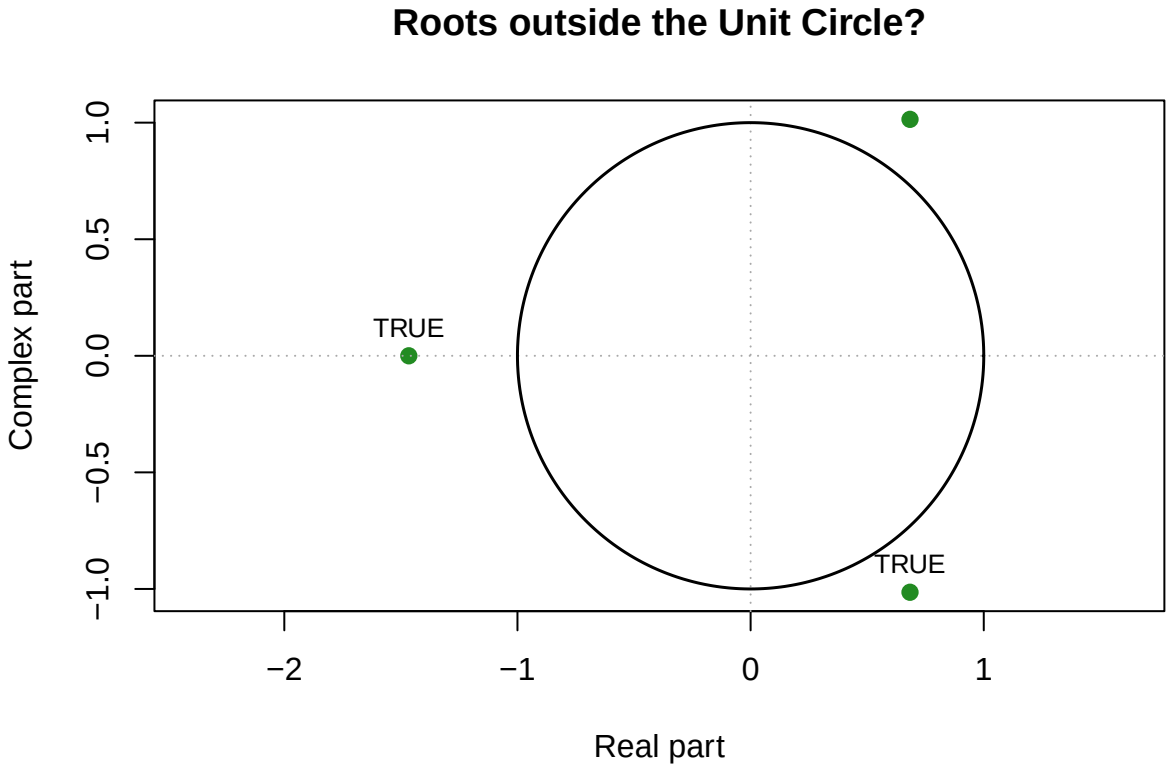

```
# Checking invertibility for A
uc.check(pol_=c(1, -0.8796, 0),plot_output = TRUE)
```

```
##      real complex outside
## 1 1.13688       0    TRUE
## *Results are rounded to 6 digits.
```



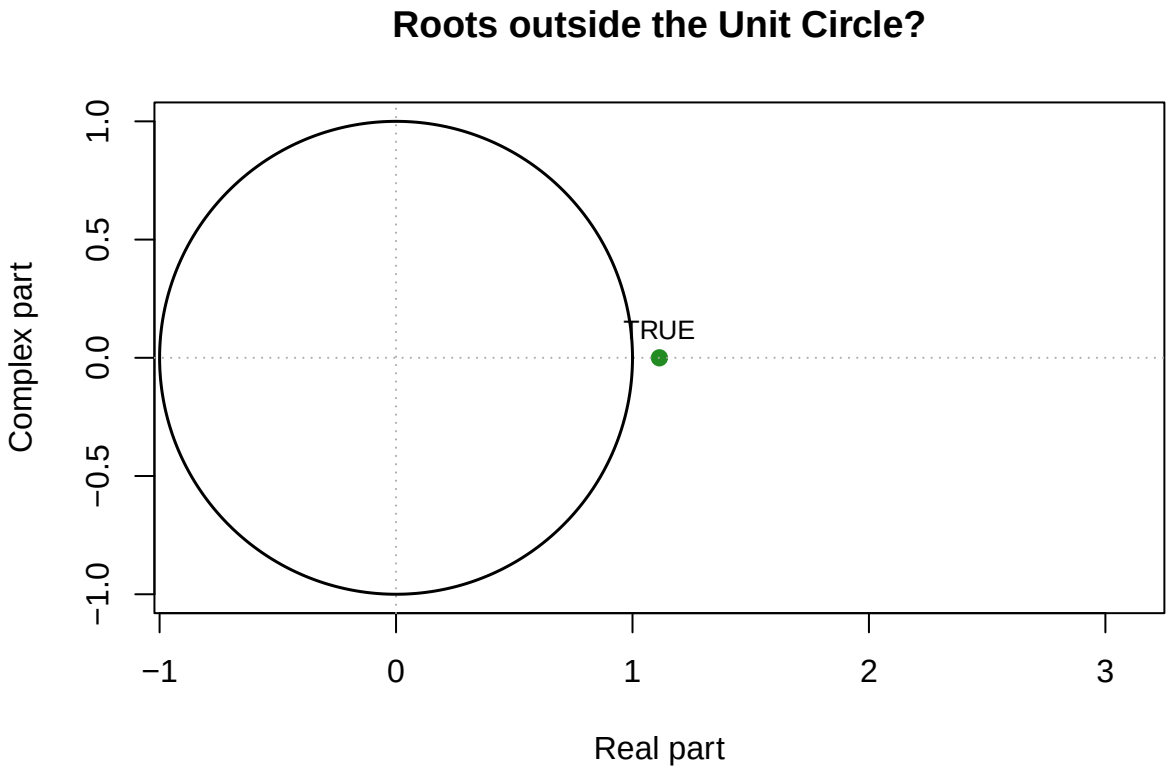

```
#Checking Stationarity for B
uc.check(pol_=c(1, -.2319, .0450, .456, 0),plot_output = TRUE)
```

```
##        real   complex outside
## 1  0.683653  1.014161    TRUE
## 2 -1.465990  0.000000    TRUE
## 3  0.683653 -1.014161    TRUE
## *Results are rounded to 6 digits.
```




```
# Checking invertibility for B
uc.check(pol_=c(1, -0.8978, 0),plot_output = TRUE)
```

```
##       real complex outside
## 1 1.113834       0    TRUE
## *Results are rounded to 6 digits.
```



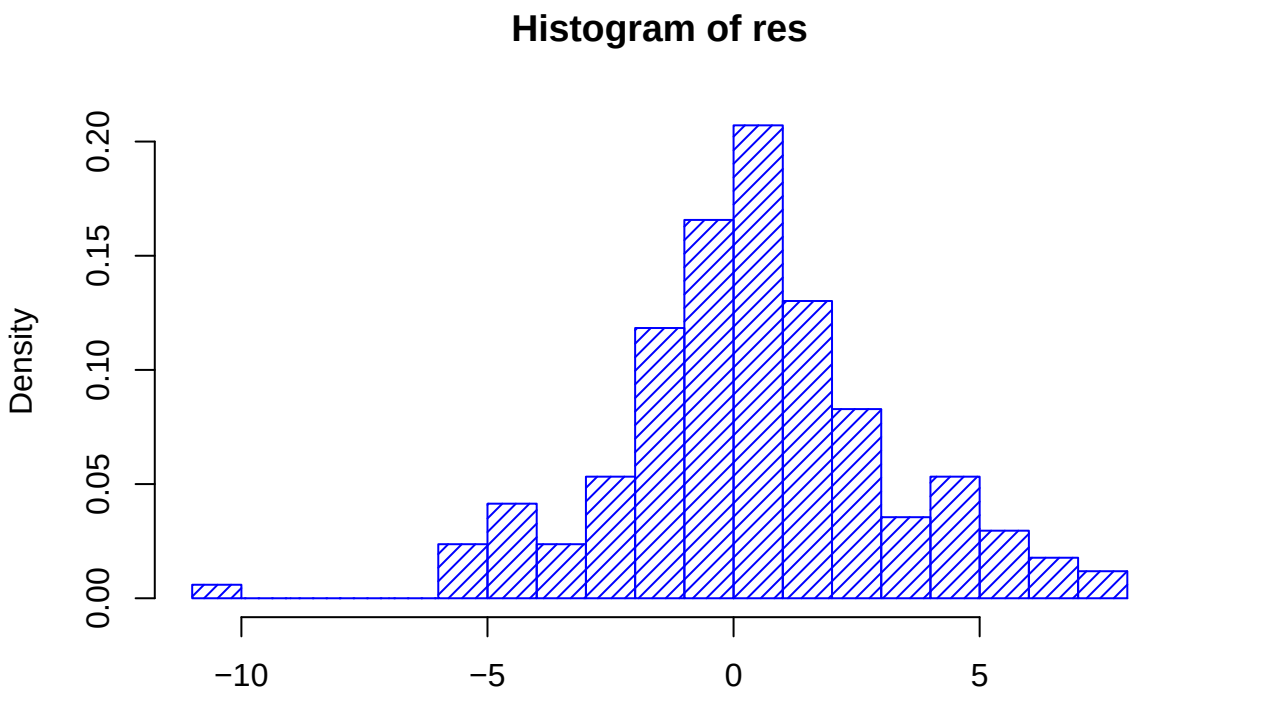
Both model A and model B pass for invertibility and stationarity because all the roots are outside the unit circle.

Diagnostic Checking for model A

```
fit<-arima(apt.bc, order=c(3,1,1),seasonal =
             list(order =c(1,1,1), period=12),
           fixed = c(NA, 0, NA, NA, NA, NA), method="ML")
```

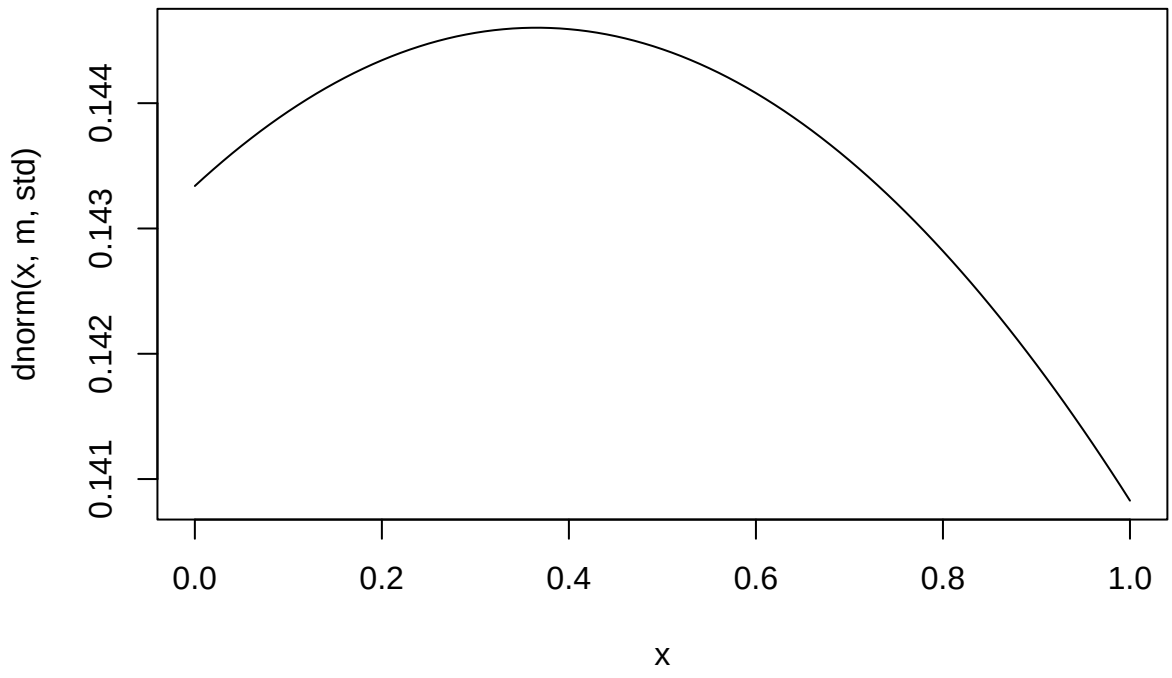
```
## Warning in arima(apt.bc, order = c(3, 1, 1), seasonal = list(order = c(1, : some
## AR parameters were fixed: setting transform.pars = FALSE
```

```
res <- residuals(fit)
#histogram of residuals
hist(res,density=20,breaks=20, col="blue", xlab="", prob=TRUE)
```



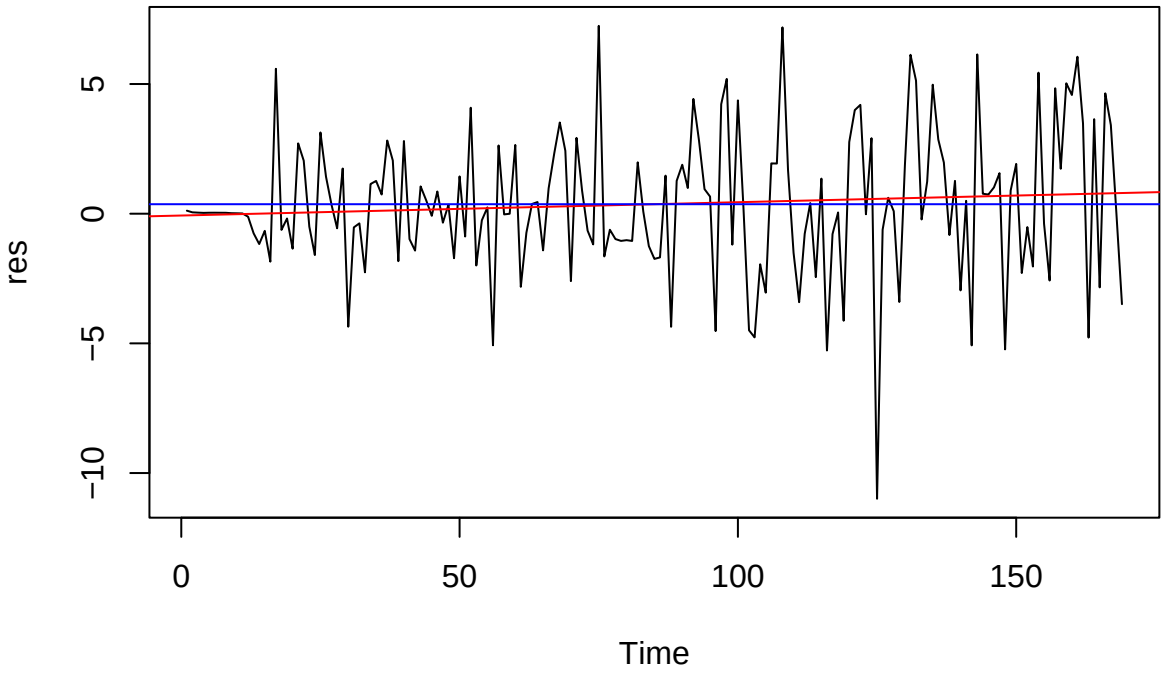

```
res <- residuals(fit)
# mean of residuals
m <- mean(res)
# standard deviation
std <- sqrt(var(res))
```

```
curve( dnorm(x,m,std))
```

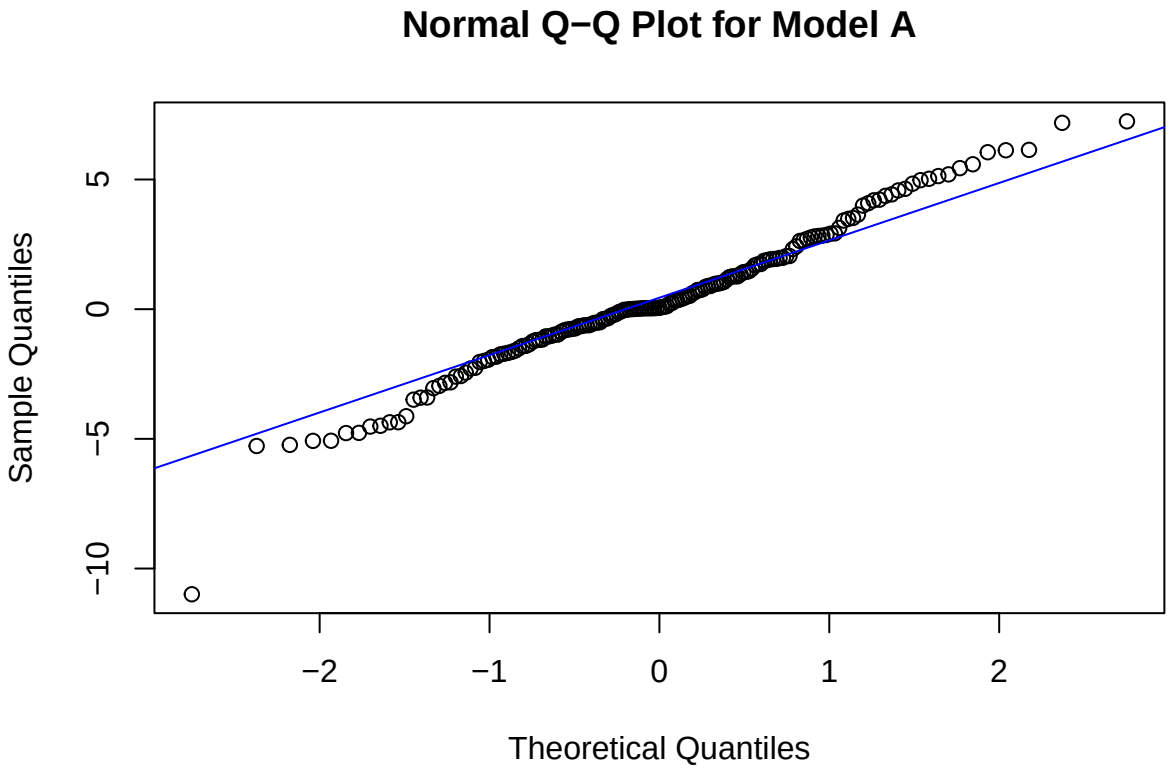



```
plot.ts(res)
```

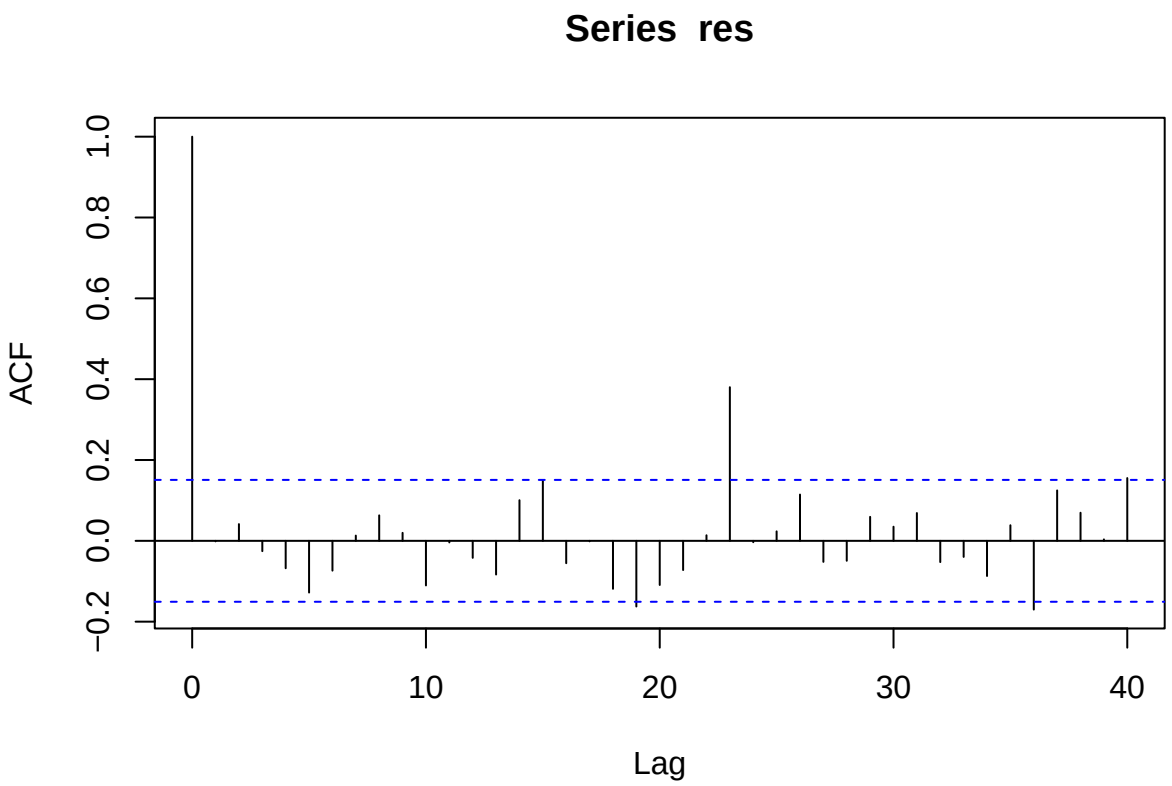
```
fitt <- lm(res ~ as.numeric(1:length(res))); abline(fitt, col="red")
abline(h=mean(res), col="blue")
```



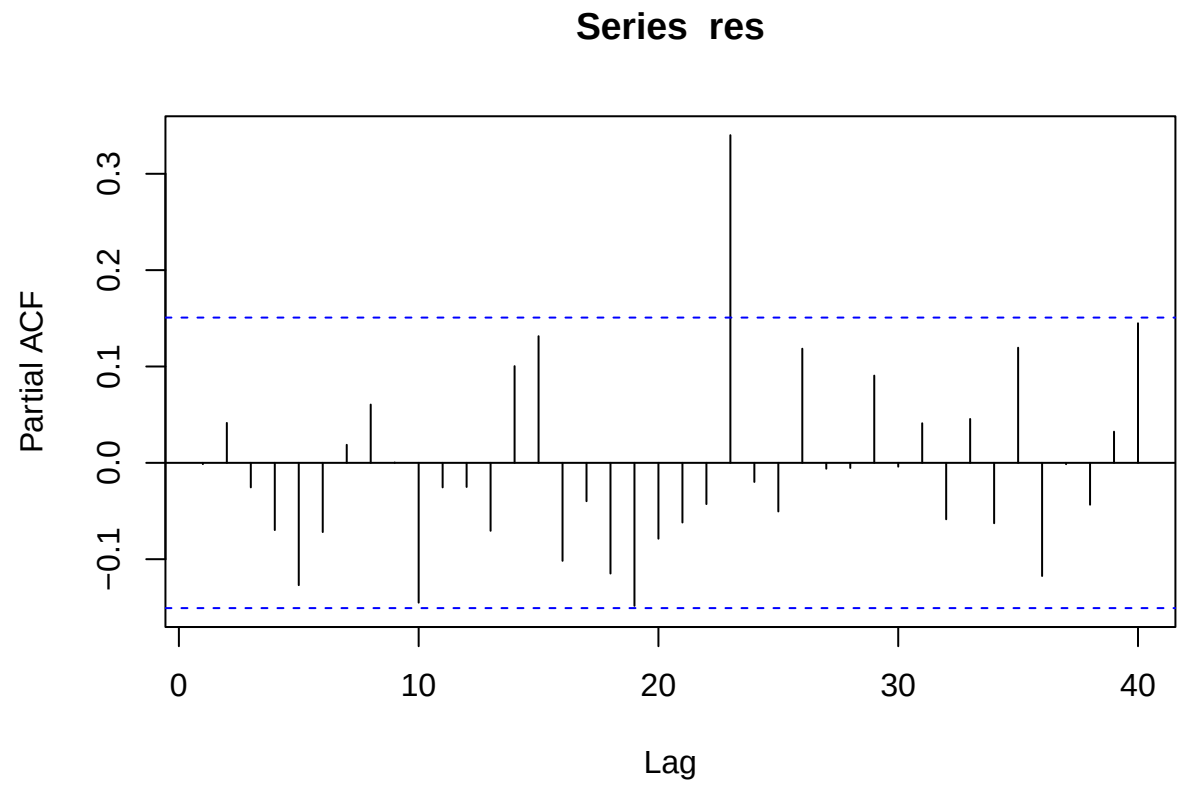

```
#after plotting the residuals we can see there is pracically no trend



qqnorm(res,main= "Normal Q-Q Plot for Model A")
qqline(res,col="blue")
```




```
#qq plot also looks good, there are some deviations,
#for a normal distribution 95% of the values
#should be between -2 and +2.

acf(res, lag.max=40)
```




```
pacf(res, lag.max=40)
```




```
#acf and pacf of residuals
#acf and pacf graphs looks good for the most part except
#there is a value outside the interval at approximately lag 24.
```

```r
shapiro.test(res)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res
## W = 0.97884, p-value = 0.01103
```

```r
#lag must be an integer number and we can estimate it by taking the
#square root of our sample size.
#Our training set has 169 observations so I chose lag=13.

#fit dif is how many coefficients we estimated (all coefficients)
Box.test(res, lag = 13, type = c("Box-Pierce"), fitdf = 5)
```

```
##
##  Box-Pierce test
##
## data:  res
## X-squared = 9.1697, df = 8, p-value = 0.3282
```

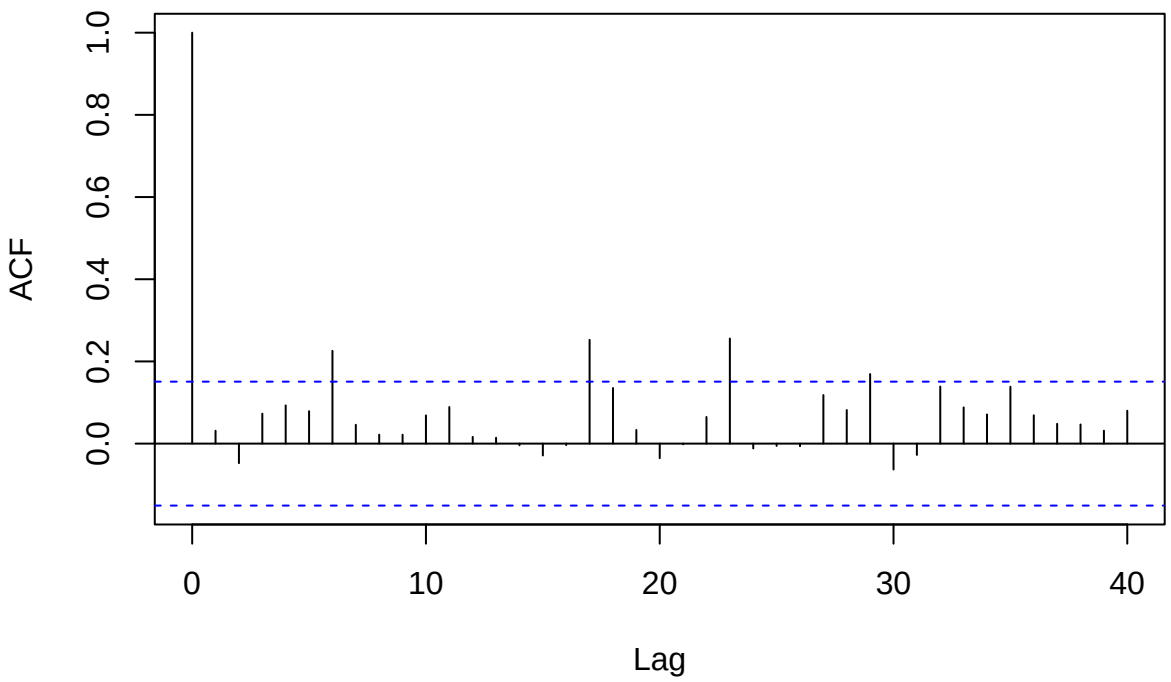
```r
Box.test(res, lag = 13, type = c("Ljung-Box"), fitdf = 5)
```

```
##
##  Box-Ljung test
##
## data:  res
## X-squared = 9.7147, df = 8, p-value = 0.2856
```

```r
#For Mchleo Lee test we set fitdf = 0
Box.test(res^2, lag = 13, type = c("Ljung-Box"), fitdf = 0)
```

```
##
##  Box-Ljung test
##
## data:  res^2
## X-squared = 16.071, df = 13, p-value = 0.2453
```

```r
#plug my residuals from model A to Yule-Walker method
acf(res^2, lag.max=40)
```



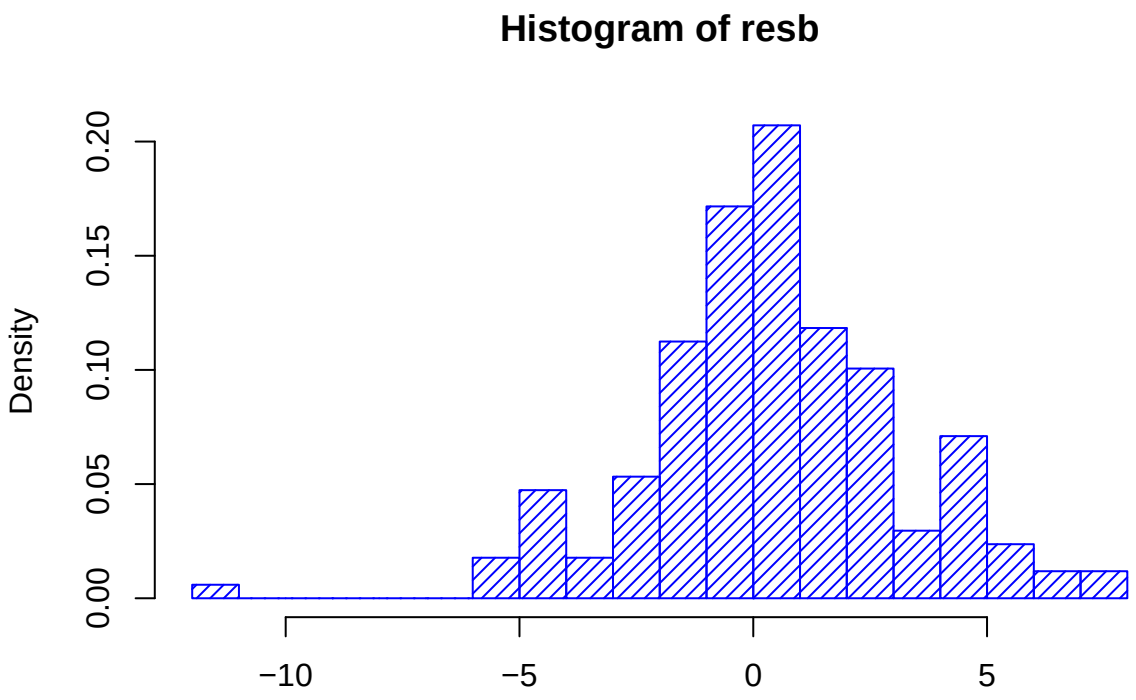

```
ar(res, aic = TRUE, order.max = NULL, method = c("yule-walker"))
```

```
##
## Call:
## ar(x = res, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  7.611
```

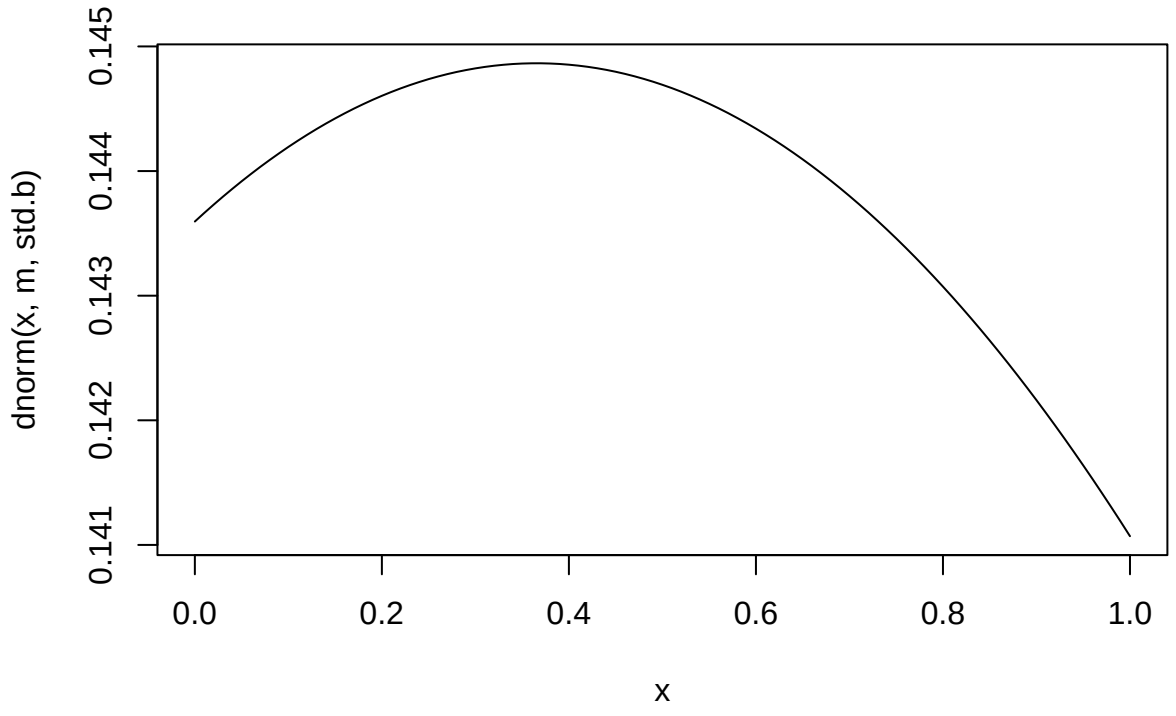
None of the P-Values are below .05 so model A is satisfactory.

Diagnostic Checking for model B

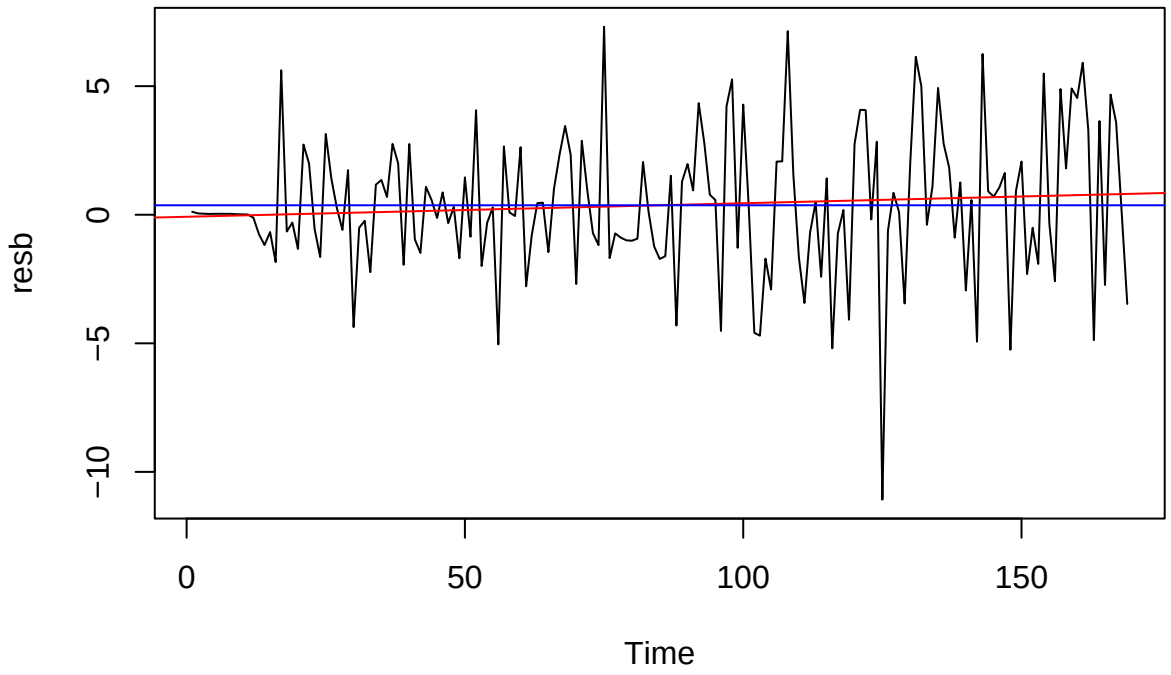
```
fitb<-arima(apt.bc, order=c(3,1,1),seasonal =
              list(order =c(1,1,1), period=12), method="ML")

resb <- residuals(fitb)
#histogram of residuals
hist(resb,density=20,breaks=20, col="blue", xlab="", prob=TRUE)
```



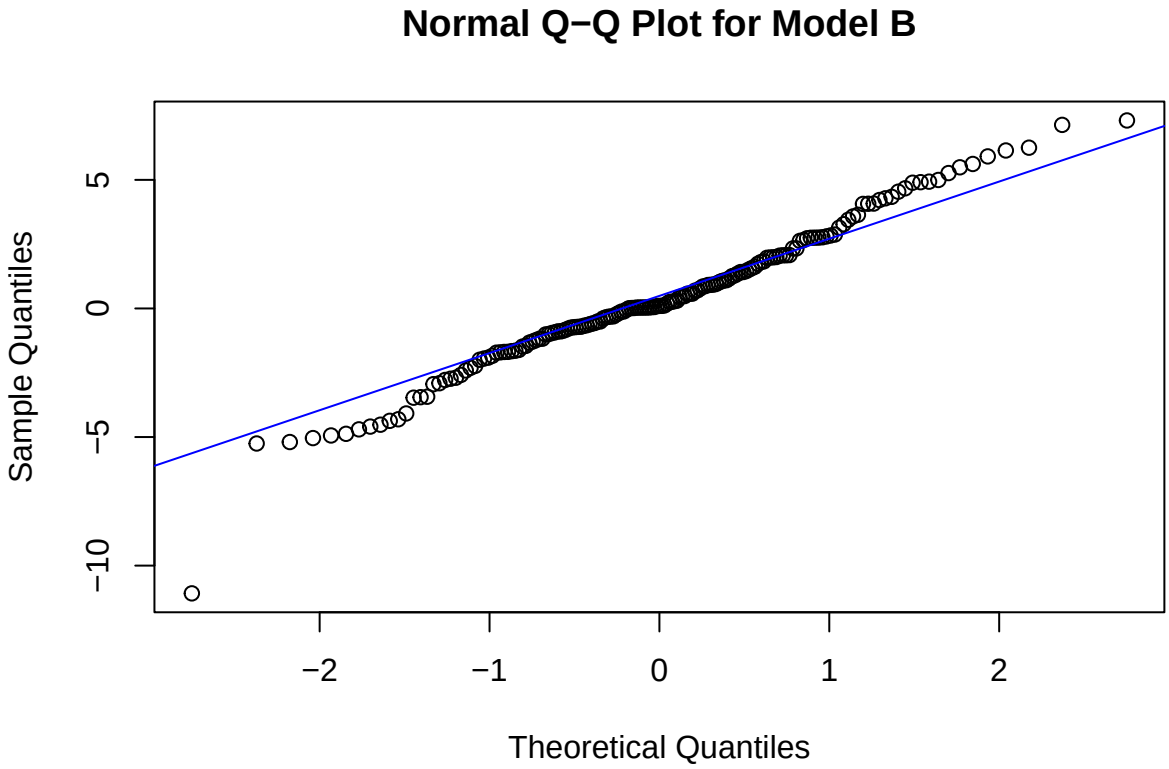

```
#mean of residuals
m.b <- mean(resb)
#standard deviation
std.b <- sqrt(var(resb))



curve( dnorm(x,m,std.b) )
```



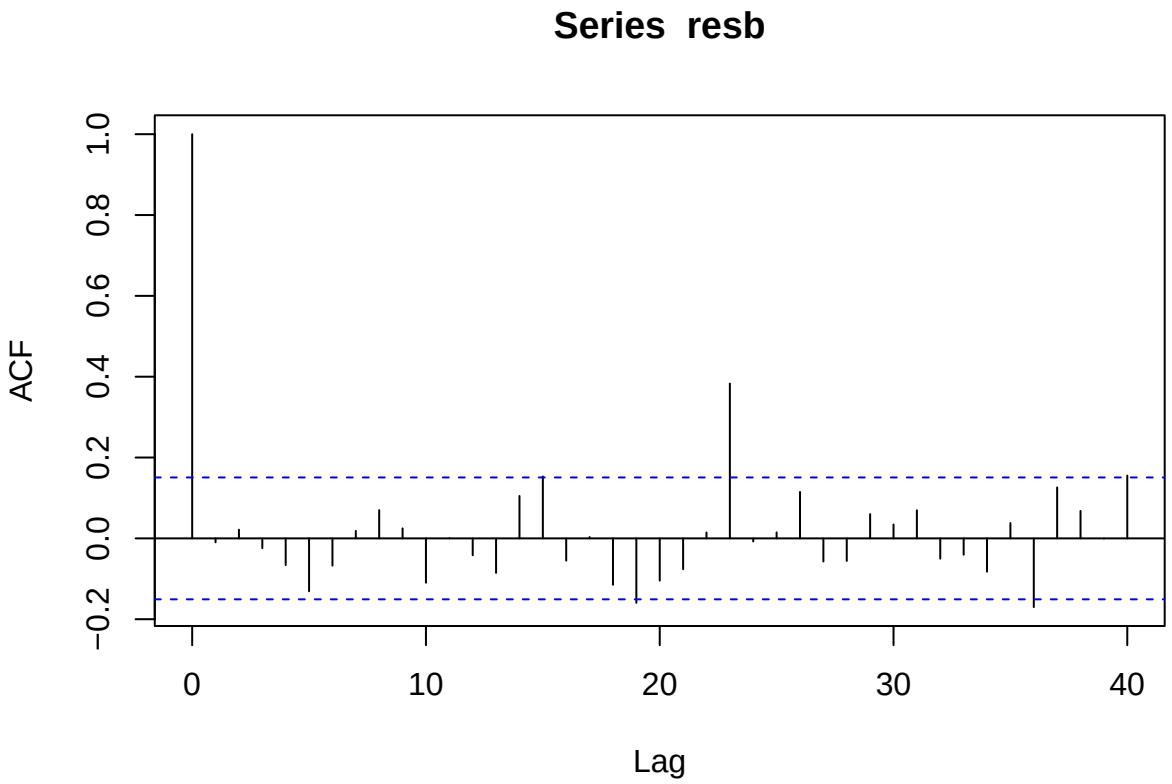
```
plot.ts(resb)
fitt.b <- lm(resb ~ as.numeric(1:length(resb))); abline(fitt.b, col="red")
abline(h=mean(resb), col="blue")
```



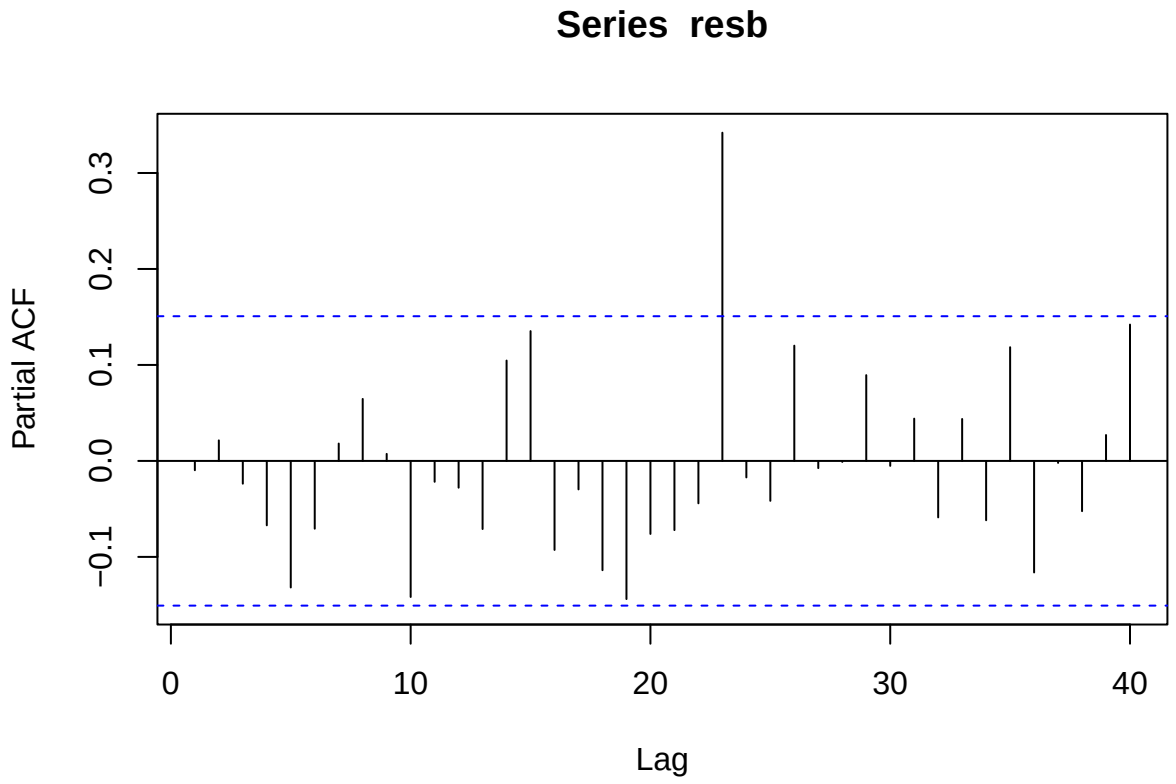

```
qqnorm(resb,main= "Normal Q-Q Plot for Model B")
qqline(resb,col="blue")
```



```
acf(resb, lag.max=40)
```




```
pacf(resb, lag.max=40)
```



```r
shapiro.test(resb)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resb
## W = 0.97891, p-value = 0.01124
```

```r
Box.test(resb, lag = 13, type = c("Box-Pierce"), fitdf = 6)
```

```
##
##  Box-Pierce test
##
## data:  resb
## X-squared = 9.1439, df = 7, p-value = 0.2425
```

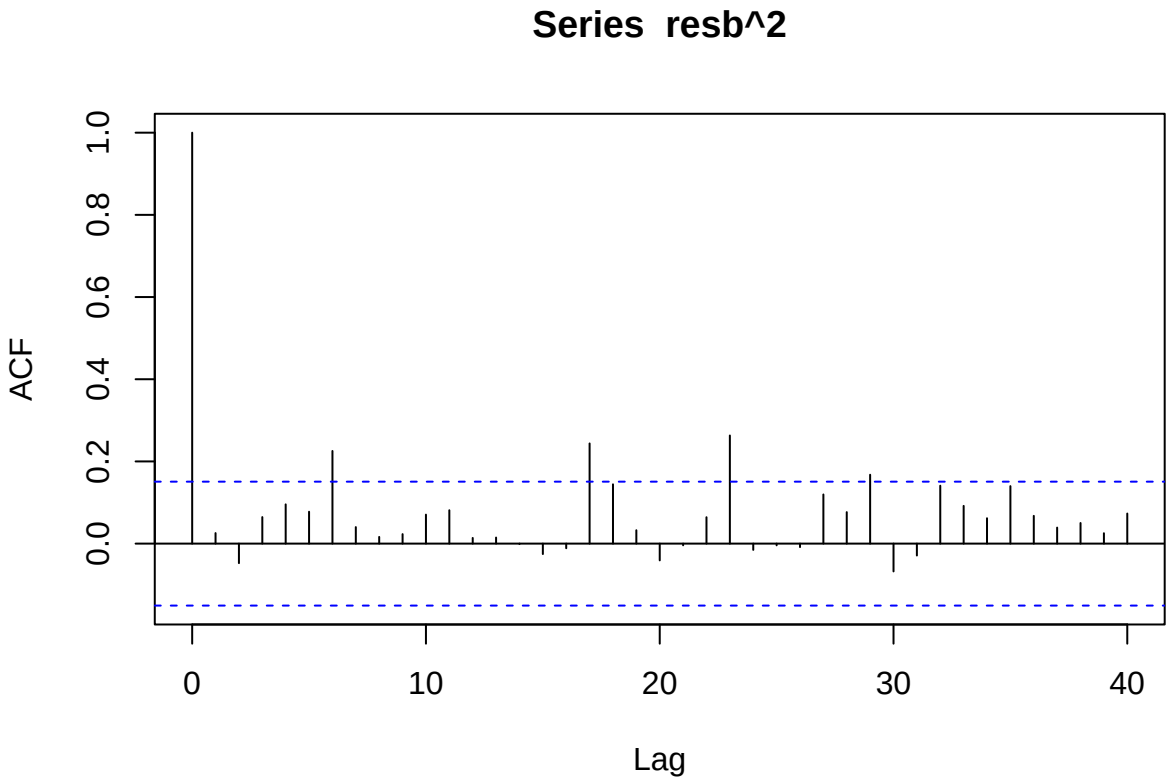
```r
Box.test(resb, lag = 13, type = c("Ljung-Box"), fitdf = 6)
```

```
##
##  Box-Ljung test
##
## data:  resb
## X-squared = 9.6971, df = 7, p-value = 0.2064
```

```r
#For Mchleo Lee test we set fitdf = 0
Box.test(resb^2, lag = 13, type = c("Ljung-Box"), fitdf = 0)
```

```
##
##  Box-Ljung test
##
## data:  resb^2
## X-squared = 15.525, df = 13, p-value = 0.2757
```

```r
acf(resb^2, lag.max=40)
```




```
ar(resb, aic = TRUE, order.max = NULL, method = c("yule-walker"))
```

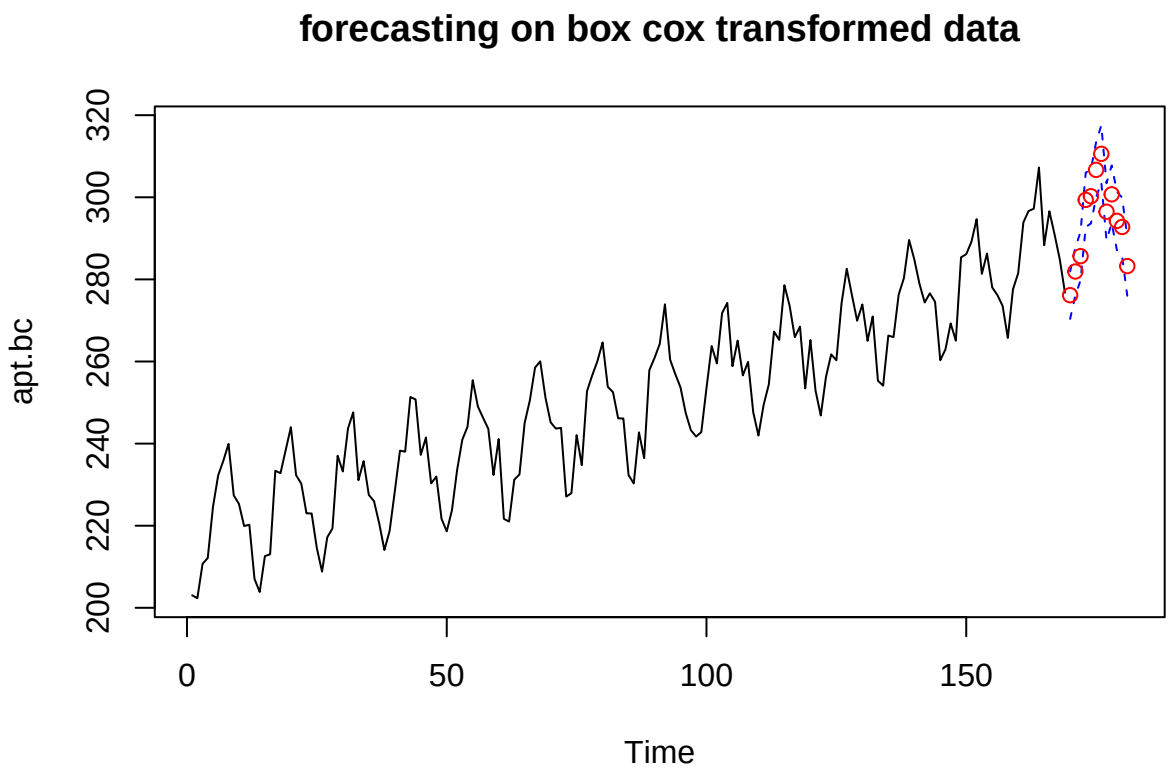
```
##
## Call:
## ar(x = resb, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  7.584
```

Model B fails the Shapiro-Wilk normality test. The p-value is .01124 which is less than .05.

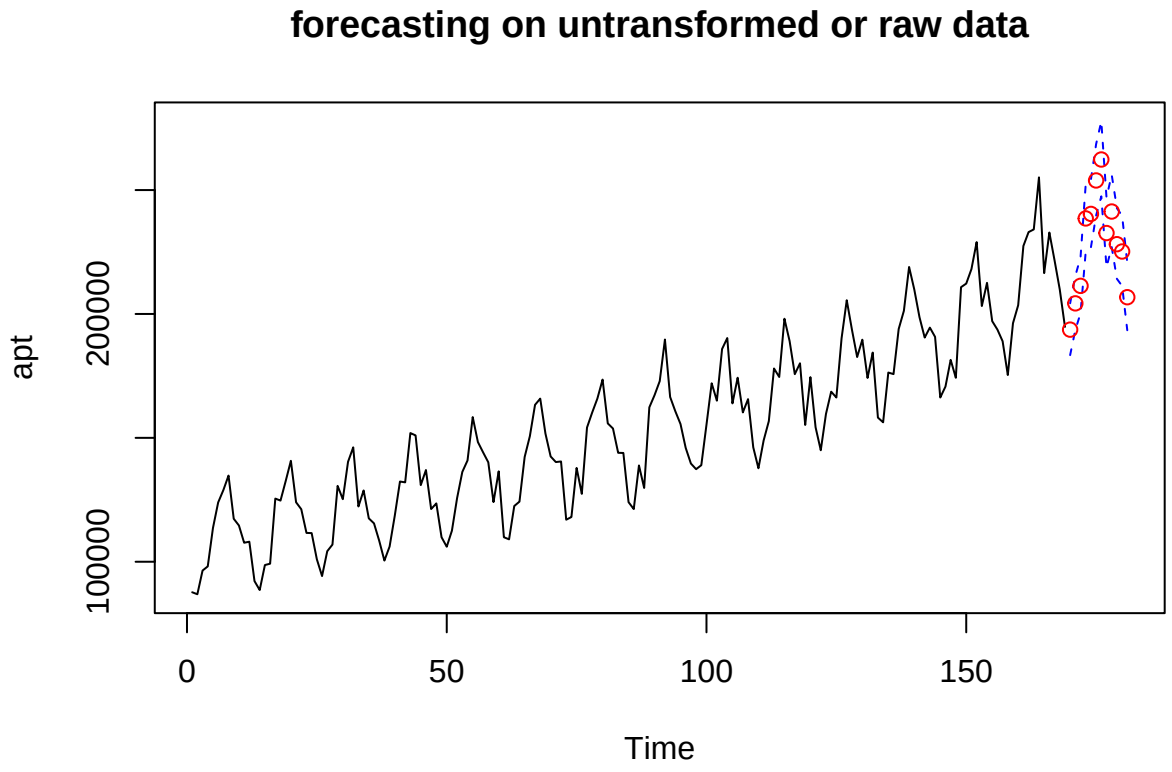
Forecasting

```
forecast(fit)
```

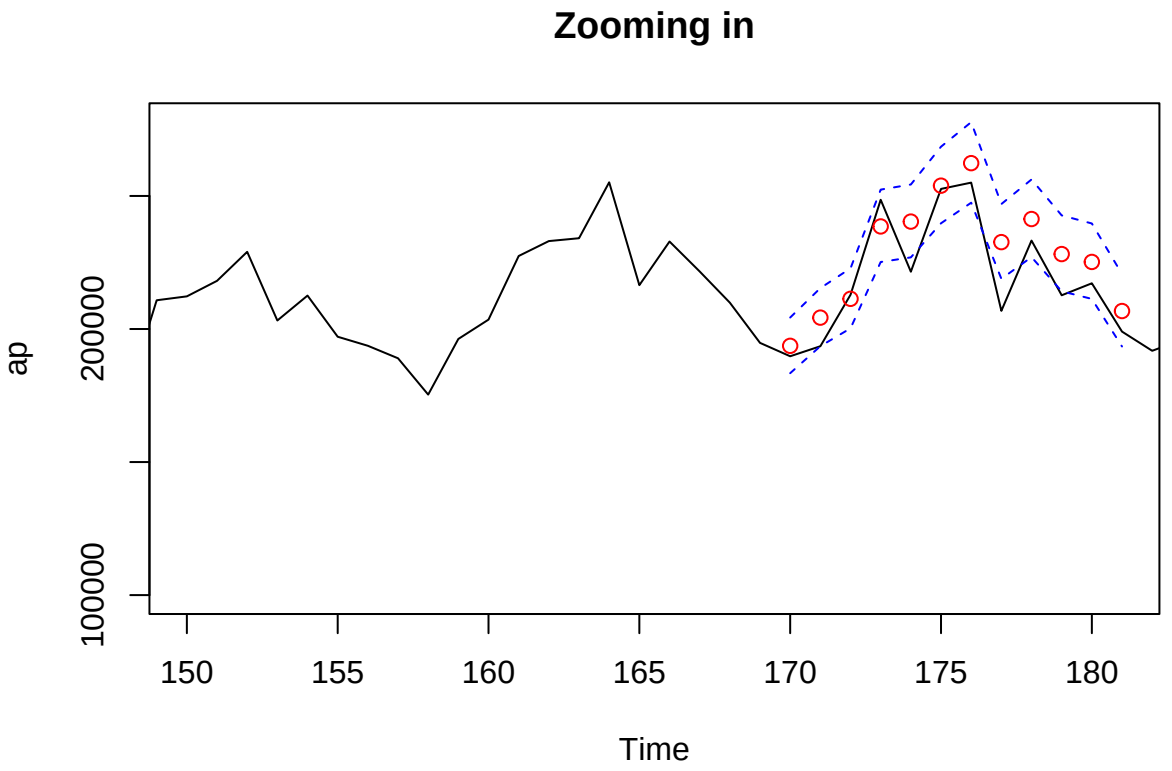
```
pred.tr <- predict(fit, n.ahead = 12)

#forecasting on box cox transformed data
U.tr= pred.tr$pred + 2*pred.tr$se # upper bound of prediction interval
L.tr= pred.tr$pred - 2*pred.tr$se # lower bound
ts.plot(apt.bc, xlim=c(1,length(apt.bc)+12), ylim = c(min(apt.bc),max(U.tr)),
        main='forecasting on box cox transformed data')
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(apt.bc)+1):(length(apt.bc)+12), pred.tr$pred, col="red")
```




```
#forecasting on untransformed or raw data
par(mfrow=c(1,1))
pred.orig <- (lambda*(pred.tr$pred)+1)^(1/lambda)
U= (lambda*(U.tr)+1)^(1/lambda)
L= (lambda*(L.tr)+1)^(1/lambda)
ts.plot(apt, xlim=c(1,length(apt)+12), ylim = c(min(apt),max(U)),
        main="forecasting on untransformed or raw data")
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(apt)+1):(length(apt)+12), pred.orig, col="red")
```




```
#zooming in
ts.plot(ap, xlim = c(150,length(apt)+12), ylim = c(100000,max(U)), main="Zooming in")
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(apt)+1):(length(apt)+12), pred.orig, col="red")
```



Conclusion Section

Here we can see that our testing data is in the interval we predicted it would be in, for the most part. At time equals 177 there is some deviation from the forecasted interval due to a dramatic drop in demand for gasoline. We can use the same model to predict gasoline demand in later years like 1980 and so forth. Overall, we used a mixed model to make this forecast, in addition to several other time series analysis techniques, which were referred to and explained throughout the code in either comments or ordinary text. Our final model equation is $(1-.2601_{.0783}B+.4339_{.0758}B^3)(1+.1622_{.1223}B^{12})\nabla_1\nabla_{12}Y(U_t)=(1-.8796_{0.0477}B)(1-.8133_{.1011}B^{12})Z_t$. I would like to thank Raya Feldman, and Youhong Lee for all the help on this exploratory analysis.

References

- Professor Raya Feldman, Teacher's Assistant Youhong Lee - Labs 1-7, winter 2022
- Professor Raya Feldman - Lecture 15 Slides, winter 2022
- TSDL time series library

Appendix

```
knitr::opts_chunk$set(echo = TRUE)
library(tsdl)
library(dplyr)
library(MASS)
library(zoo)
library(ggfortify)
library(ggplot2)
library(MuMIn)
library(UnitCircle)
library(forecast)
ap<-as.data.frame(tsdl[[4]])



plot.ts(ap, ylab='Gasonline Demand')

dim(ap)

fit <- lm(tsdl[[4]] ~ as.numeric(1:192))%>%
  abline(fit, col="red")
mean(tsdl[[4]])[1]
abline(h=mean(tsdl[[4]]), col='blue')

#monthly demand for gasoline in Ontario
our_t<-ts(tsdl[[4]],start = c(1960,1),end = c(1975,12), frequency = 12)

ts.plot(our_t, main="raw data")

apt = tsdl[[4]][c(1:169)] #the training set

ap.testing = tsdl[[4]][c(170:181)] #testing set



plot.ts(apt)
fit <- lm(apt ~ as.numeric(1:length(apt))); abline(fit, col="red")
abline(h=mean(tsdl[[4]]), col="blue")
```

```
hist(apt, main="hist of training data")
acf(apt, lag.max = 40, main="ACF of training data")
length(apt)

t = 1:169
fit = lm(apt ~ t)

bcTransform = boxcox(apt ~ t, plotit = TRUE)
lambda = bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
lambda
apt.bc = (1/lambda)*(apt^lambda-1)
op <- par(mfrow = c(1,2))
ts.plot(apt,main = "Original data",ylab = expression(X[t]))
ts.plot(apt.bc,main = "Box-Cox tranformed data", ylab = expression(Y[t]))
#log transform
ts.log = log(apt)
# square root transform
ts.sqrt = sqrt(apt)

#Compare transforms
op= par(mfrow=c(2,2))
ts.plot(apt, main = "Original Times Series")
ts.plot(apt.bc, main = "Box-Cox Transform")
ts.plot(ts.log, main = "Log Transform")
ts.plot(ts.sqrt, main = "Square Root Transform")
hist(apt, main = "Original Times Series")
hist(apt.bc, main = "Box-Cox Transform")
hist(ts.log, main = "Log Transform")
hist(ts.sqrt, main = "Square Root Transform")
z<-ts(as.ts(apt.bc),frequency=12)
decomposed_ts<-decompose(z, type='multiplicative')

autoplot(decomposed_ts)
var(apt.bc)
op = par(mfrow = c(1,2))
acf(apt.bc, lag.max = 60, main = "")
pacf(apt.bc, lag.max = 60, main = "")
title("Box-Cox Transformed Time Series", line = -1, outer=TRUE)
# Diference at lag = 1 to remove trend component
y1 = diff(apt.bc, 1)
plot(y1,main = "De-trended Time Series",ylab = expression(nabla~Y[t]))
abline(h = 0,lty = 2)
var(y1)
# Diference at lag = 12 (cycle determined by the ACF) to remove seasonal component
y12 = diff(y1, 12)
ts.plot(y12,main = "De-trended/seasonalized Time Series",ylab = expression(nabla^{12}~nabla^{1}~Y[t]))
abline(h = 0,lty = 2)
var(y12)
acf(apt.bc, lag.max = 40)
acf(y1, lag.max = 40)
acf(y12, lag.max = 40)
#differencing our_data.bc at lag 12 first this time
var(apt.bc)
```

```
our_ts.bc.12<-diff(apt.bc, lag=12)
plot.ts(our_ts.bc.12, main='lagged at 12')
var(our_ts.bc.12)
fit <- lm(our_ts.bc.12 ~ as.numeric(1:length(our_ts.bc.12)))
abline(fit, col="red")
mean(our_ts.bc.12)
abline(h=mean(our_ts.bc.12), col="blue")
#now lag 1
var(our_ts.bc.12)
our_ts.bc.12.1<-diff(our_ts.bc.12, lag=1)
plot.ts(our_ts.bc.12.1, main='lag differenced at 12 and 1')
var(our_ts.bc.12.1)
fit <- lm(our_ts.bc.12.1 ~ as.numeric(1:length(our_ts.bc.12.1))); abline(fit, col="red")
mean(our_ts.bc.12.1)
abline(h=mean(our_ts.bc.12.1), col="blue")
acf(apt.bc, lag.max = 80,
    main="Box Cox ACF")
acf(our_ts.bc.12, lag.max = 80,
    main="Box Cox + differenced at lag 12 ACF")
acf(our_ts.bc.12.1, lag.max = 80,
    main="Box Cox + differenced at lag 12 & lag 1 ACF")
hist(apt.bc, main = "Box Cox Data")
hist(our_ts.bc.12, main = "Box Cox Data differenced at lag 12")
hist(our_ts.bc.12.1, main = "Box Cox Data differenced at lag 12 and lag 1")

wts<-our_ts.bc.12
hist(our_ts.bc.12.1, density=20,breaks=20,
     main = "Density of data differenced at lag 1 & 12", col="blue", xlab="", prob=TRUE)
m<-mean(our_ts.bc.12.1)
std<- sqrt(var(our_ts.bc.12.1))
curve( dnorm(x,m,std), add=TRUE )

hist(wts, density=20,breaks=20,main =
       "Density of data differenced at lag 12", col="blue", xlab="", prob=TRUE)
m<-mean(wts)
std<- sqrt(var(wts))
curve( dnorm(x,m,std), add=TRUE )

hist(apt.bc, density=20,breaks=20,
     main="Density of Box Cox data without differencing",
     col="blue", xlab="", prob=TRUE)
m<-mean(apt.bc)
std<- sqrt(var(apt.bc))
curve( dnorm(x,m,std), add=TRUE )

acf(our_ts.bc.12.1, lag.max = 80,
    main="ACF of transformed and differenced data" )
pacf(our_ts.bc.12.1, lag.max = 80,
     main="PACF of transformed and differenced data")
arima(apt.bc, order=c(1,1,1),seasonal = list(order =c(1,1,1), period=12), method="ML")
AICc(arima(wts, order=c(1,1,1), seasonal = list(order = c(1,1,1), period = 12), method="ML"))
arima(apt.bc, order=c(1,1,1),seasonal = list(order =c(1,1,2), period=12), method="ML")
AICc(arima(apt.bc, order=c(1,1,1), seasonal = list(order = c(1,1,2), period = 12), method="ML"))
```

```
arima(apt.bc, order=c(1,1,3),seasonal = list(order =c(1,1,1), period=12), method="ML")
AICc(arima(apt.bc, order=c(1,1,3), seasonal = list(order = c(1,1,1), period = 12), method="ML"))
# Second best
b.sarima<-arima(apt.bc, order=c(3,1,1),seasonal = list(order =c(1,1,1), period=12), method="ML")
b.sarima
AICc(arima(apt.bc, order=c(3,1,1), seasonal = list(order = c(1,1,1), period = 12), method="ML"))
arima(apt.bc, order=c(0,1,1),seasonal = list(order =c(1,1,1), period=12), method="ML")
AICc(arima(apt.bc, order=c(0,1,1), seasonal = list(order = c(1,1,1), period = 12), method="ML"))
arima(apt.bc, order=c(0,1,1),seasonal = list(order =c(0,1,1), period=12), method="ML")
AICc(arima(apt.bc, order=c(0,1,1), seasonal = list(order = c(0,1,1), period = 12), method="ML"))
#third best
b2.sarima<-arima(apt.bc, order=c(2,1,2),
                 seasonal = list(order =c(0,1,2), period=12), method="ML")
b2.sarima
AICc(arima(apt.bc, order=c(2,1,2),
           seasonal = list(order = c(0,1,2), period = 12), method="ML"))
# Best
abs<-arima(apt.bc, order=c(3,1,1),seasonal =
             list(order =c(1,1,1), period=12),
           fixed = c(NA, 0, NA, NA, NA, NA), method="ML")
abs
AICc(arima(apt.bc, order=c(3,1,1), seasonal =
             list(order = c(1,1,1), period = 12),
           fixed = c(NA, 0, NA, NA, NA, NA), method="ML"))
#Checking Stationarity for A
uc.check(pol_=c(1, -.2601, .4339, 0),plot_output = TRUE)

# Checking invertibility for A
uc.check(pol_=c(1, -0.8796, 0),plot_output = TRUE)

#Checking Stationarity for B
uc.check(pol_=c(1, -.2319, .0450, .456, 0),plot_output = TRUE)
# Checking invertibility for B
uc.check(pol_=c(1, -0.8978, 0),plot_output = TRUE)

fit<-arima(apt.bc, order=c(3,1,1),seasonal =
             list(order =c(1,1,1), period=12),
           fixed = c(NA, 0, NA, NA, NA, NA), method="ML")

res <- residuals(fit)
#histogram of residuals
hist(res,density=20,breaks=20, col="blue", xlab="", prob=TRUE)

res <- residuals(fit)
# mean of residuals
m <- mean(res)
# standard deviation
std <- sqrt(var(res))
curve( dnorm(x,m,std))


plot.ts(res)
```

```
fitt <- lm(res ~ as.numeric(1:length(res))); abline(fitt, col="red")
abline(h=mean(res), col="blue")
#after plotting the residuals we can see there is pracically no trend



qqnorm(res,main= "Normal Q-Q Plot for Model A")
qqline(res,col="blue")
#qq plot also looks good, there are some deviations,
#for a normal distribution 95% of the values
#should be between -2 and +2.

acf(res, lag.max=40)
pacf(res, lag.max=40)
#acf and pacf of residuals
#acf and pacf graphs looks good for the most part except
#there is a value outside the interval at approximately lag 24.


shapiro.test(res)

#lag must be an integer number and we can estimate it by taking the
#square root of our sample size.
#Our training set has 169 observations so I chose lag=13.

#fit dif is how many coefficients we estimated (all coefficients)
Box.test(res, lag = 13, type = c("Box-Pierce"), fitdf = 5)

Box.test(res, lag = 13, type = c("Ljung-Box"), fitdf = 5)


#For Mchleo Lee test we set fitdf = 0
Box.test(res^2, lag = 13, type = c("Ljung-Box"), fitdf = 0)

#plug my residuals from model A to Yule-Walker method
acf(res^2, lag.max=40)
ar(res, aic = TRUE, order.max = NULL, method = c("yule-walker"))



fitb<-arima(apt.bc, order=c(3,1,1),seasonal =
              list(order =c(1,1,1), period=12), method="ML")

resb <- residuals(fitb)
#histogram of residuals
hist(resb,density=20,breaks=20, col="blue", xlab="", prob=TRUE)
#mean of residuals
m.b <- mean(resb)
#standard deviation
std.b <- sqrt(var(resb))
```

```
curve( dnorm(x,m,std.b) )

plot.ts(resb)
fitt.b <- lm(resb ~ as.numeric(1:length(resb))); abline(fitt.b, col="red")
abline(h=mean(resb), col="blue")

qqnorm(resb,main= "Normal Q-Q Plot for Model B")
qqline(resb,col="blue")

acf(resb, lag.max=40)
pacf(resb, lag.max=40)


shapiro.test(resb)


Box.test(resb, lag = 13, type = c("Box-Pierce"), fitdf = 6)

Box.test(resb, lag = 13, type = c("Ljung-Box"), fitdf = 6)

#For Mchleo Lee test we set fitdf = 0
Box.test(resb^2, lag = 13, type = c("Ljung-Box"), fitdf = 0)

acf(resb^2, lag.max=40)
ar(resb, aic = TRUE, order.max = NULL, method = c("yule-walker"))


forecast(fit)
pred.tr <- predict(fit, n.ahead = 12)

#forecasting on box cox transformed data
U.tr= pred.tr$pred + 2*pred.tr$se # upper bound of prediction interval
L.tr= pred.tr$pred - 2*pred.tr$se # lower bound
ts.plot(apt.bc, xlim=c(1,length(apt.bc)+12), ylim = c(min(apt.bc),max(U.tr)),
        main='forecasting on box cox transformed data')
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(apt.bc)+1):(length(apt.bc)+12), pred.tr$pred, col="red")


#forecasting on untransformed or raw data
par(mfrow=c(1,1))
pred.orig <- (lambda*(pred.tr$pred)+1)^(1/lambda)
U= (lambda*(U.tr)+1)^(1/lambda)
L= (lambda*(L.tr)+1)^(1/lambda)
ts.plot(apt, xlim=c(1,length(apt)+12), ylim = c(min(apt),max(U)),
        main="forecasting on untransformed or raw data")
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(apt)+1):(length(apt)+12), pred.orig, col="red")

 #zooming in
 ts.plot(ap, xlim = c(150,length(apt)+12), ylim = c(100000,max(U)), main="Zooming in")
```

```
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(apt)+1):(length(apt)+12), pred.orig, col="red")
```