PSTAT 174 Final Project
Average Sales Price of Houses Sold for the West Coast Region

Marissa Santiago

March 12, 2022

Abstract

The cost of owning a home has only gone up since the 1980's with the housing market only looking to increase over time. In this project, we look at average quarterly sales prices of houses sold in the west coast region from 1975 to 2017. Using this data set, our goal is to forcast the average sales price of houses sold over the last 3 years of the dataset and see how how our estimates compare to our true price for houses sold in the same time period.

We do this by applying key time series techniques to real-world aplications. First I created a training set and did not include the last 12 data points, classifying those as the testing data. Since the original data set was not stationary, I applied a Box-Cox transformation. The Box-Cox transformed data displayed a trend, a seasonal component, and a changing variance so I used differencing to remove these components. After differencing twice, I examined the ACF and PACF and ran a for-loop to identify the best candidate model: SARIMA $(1, 1, 0)$ x $(4, 1, 2)_4$. For this model, I examined plots of the residuals, preformed diagnostic checks, and calculated causality and invertibility. After all test and checks, I concluded that the model was suitable to use in forecasting. Plotting forecasted future values, it is clear that the true values are within the 95% confidence interval of the forecasted values. Thus, SARIMA $(1, 1, 0)$ x $(4, 1, 2)_4$ is an accurate model to forecast our data set.

Introduction

Housing prices play a significant factor in people's life. Therefore, finding an adequate time series model to forecast future home sales prices would provide insite on when is the optimal time to buy or sell a house. Our data set is quarterly distributed and provides the average sales prices of houses sold in the west coast region from 1975 to 2017. I establish data from 1975 to 2014 as my training data and set years 2015 to 2017 as my testing data (last 12 values) in hopes of forecasting estimated values that are close to the true values we see in our testing data.

In order to effectively forecast, we must have a stationary model. I first plotted the time series of my data without the last 12 values of the data set (training data). I checked for trend, seasonality, or changes in variance. I then applied a Box-Cox transformation to make the data stationary and to stabilize the variance. Next I applied differencing at lags 4 and 1 since there was a trend and seasonal component and since the data is distributed quarterly. Then I plotted the ACF and PACF to identify potential models for the data. I identified SARIMA $(1, 1, 0)$ x $(4, 1, 2)_4$ as a possible model choice. After checking causality and invertibility via unit circle, plotting the residuals, and performing diagnostic checks, I concluded that SARIMA $(1, 1, 0)$ x $(4, 1, 2)_4$ was a suitable model for forecasting. it passed all of the tests for independence. Despite not having a p-value > 0.05 and failing the Shapiro-wilk test, we were able to justify with the Yule-walker test that the final model followed an AR(0) model, i.e WN.

Data Source

The Data used is from U.S. Census Bureau and U.S. Department of Housing and Urban Development, sourced from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/ASPW

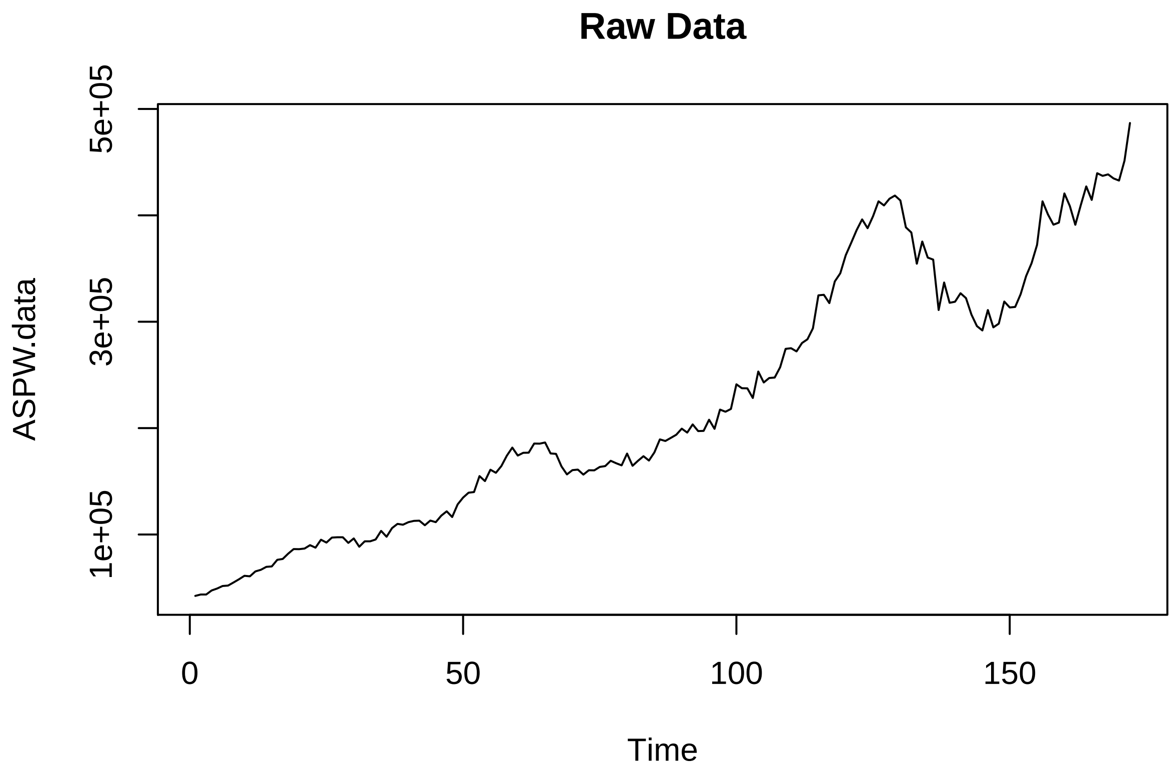
- 'DATE': The time interval that reflects the average sales price of houses. There are four quarters recorded for each year, for example, "1975-01-01" stands for the first quarter of year 1975.
- 'ASPW': Average Sales Price of Houses Sold for the West Census Region, in dollars.

All statistical analysis on the project was performed using RStudio.

Analysis

First we start by plotting the original data to confirm non-stationarity. We can see from our graph that the data follows a positive linear trend; there are also visible spikes and dips within our upward trend that suggests a seasonal component to the data.

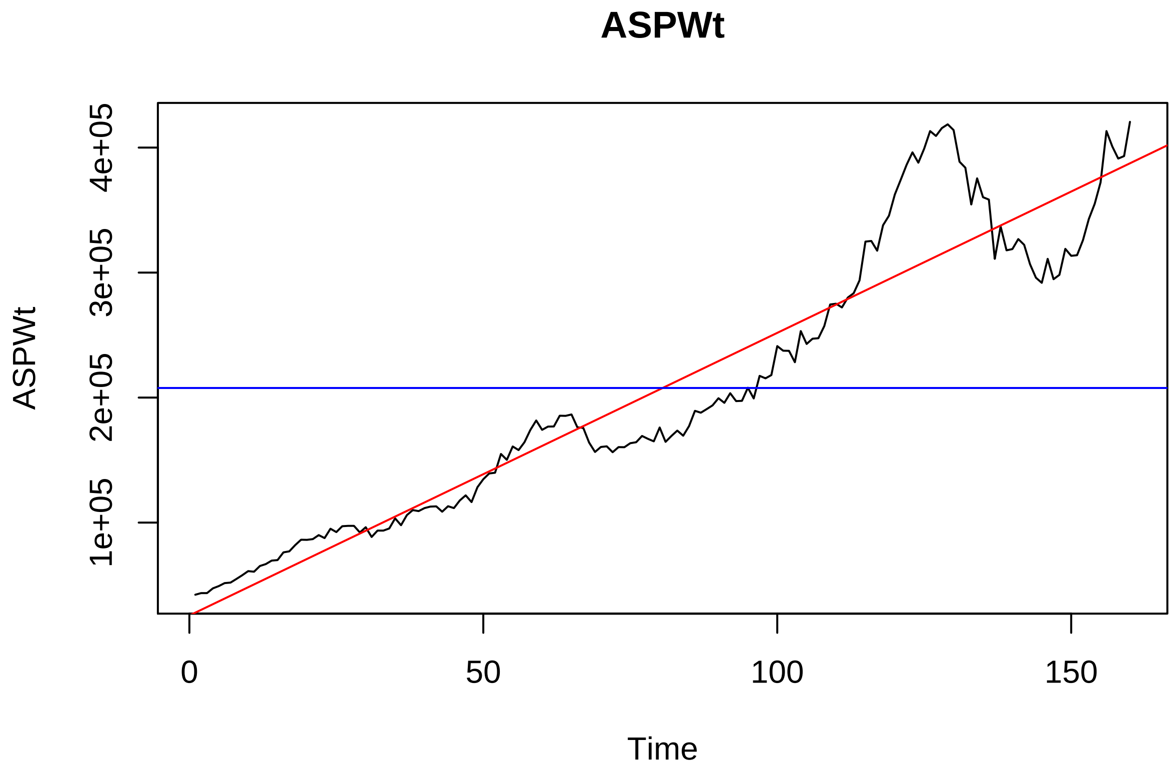
```
ts.plot(ASPW.data, main="Raw Data")
```

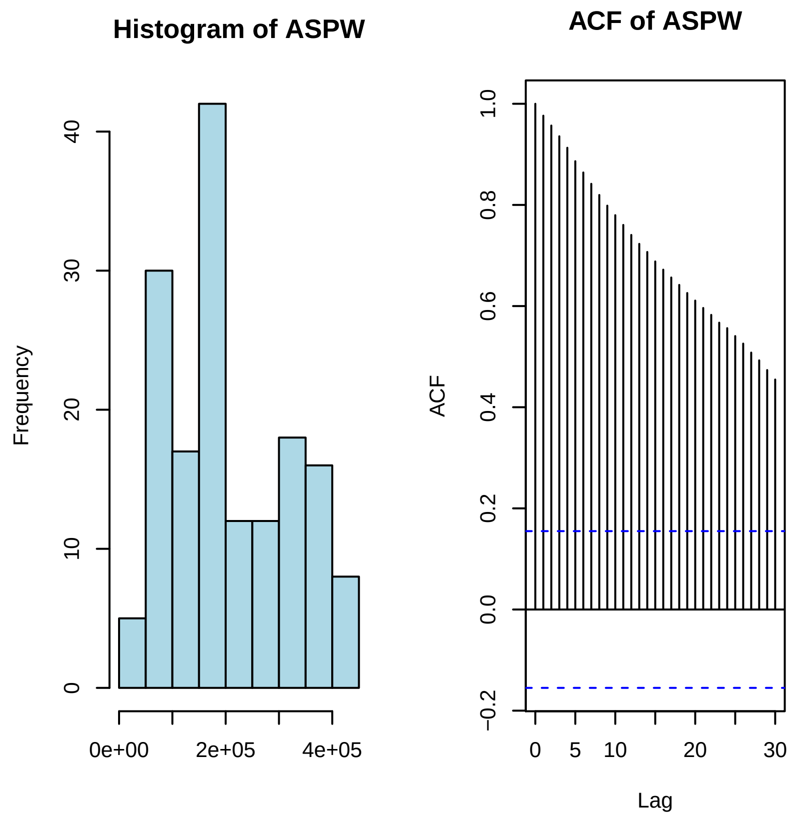


Data Exploration

I chose the data from the beginning of 1975 to the end of 2014 as the training dataset and the data from the beginning of 2015 to the end of 2017 as the test dataset.

After plotting the time series and histogram we see the histogram is skewed to the right and the ACF for the time series model depicts a slow decay indicating non-stationarity and seasonality throughout. Thus we conclude that our data does not appear Gaussian and we must perform the Box-Cox transformation.



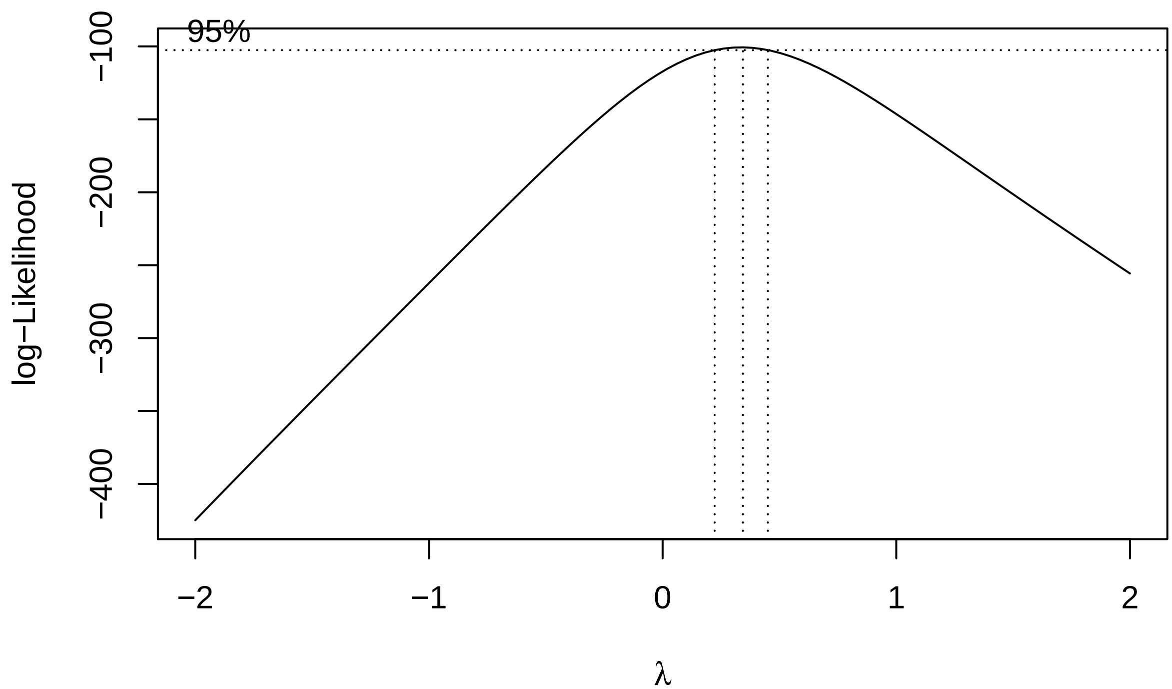


To chose the right transformation for our data, I used a Box-Cox transformation to get the value of λ. The dashed vertical lines in the plot are the 95% confidence interval for the true λ. Since the interval does not include $\lambda = 0$, then the Box-Cox transformation to stabilize the variance is

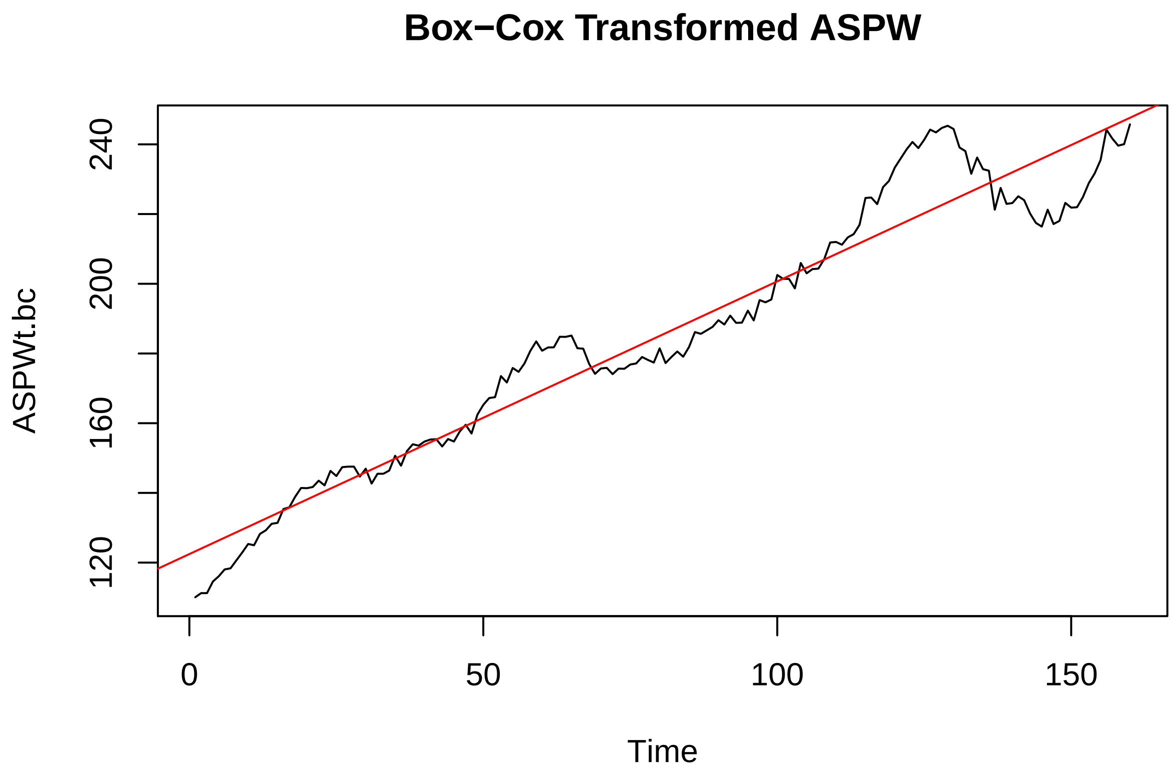
$$Y_t = \frac{1}{\lambda}(X_t^{\lambda} - 1);$$

.

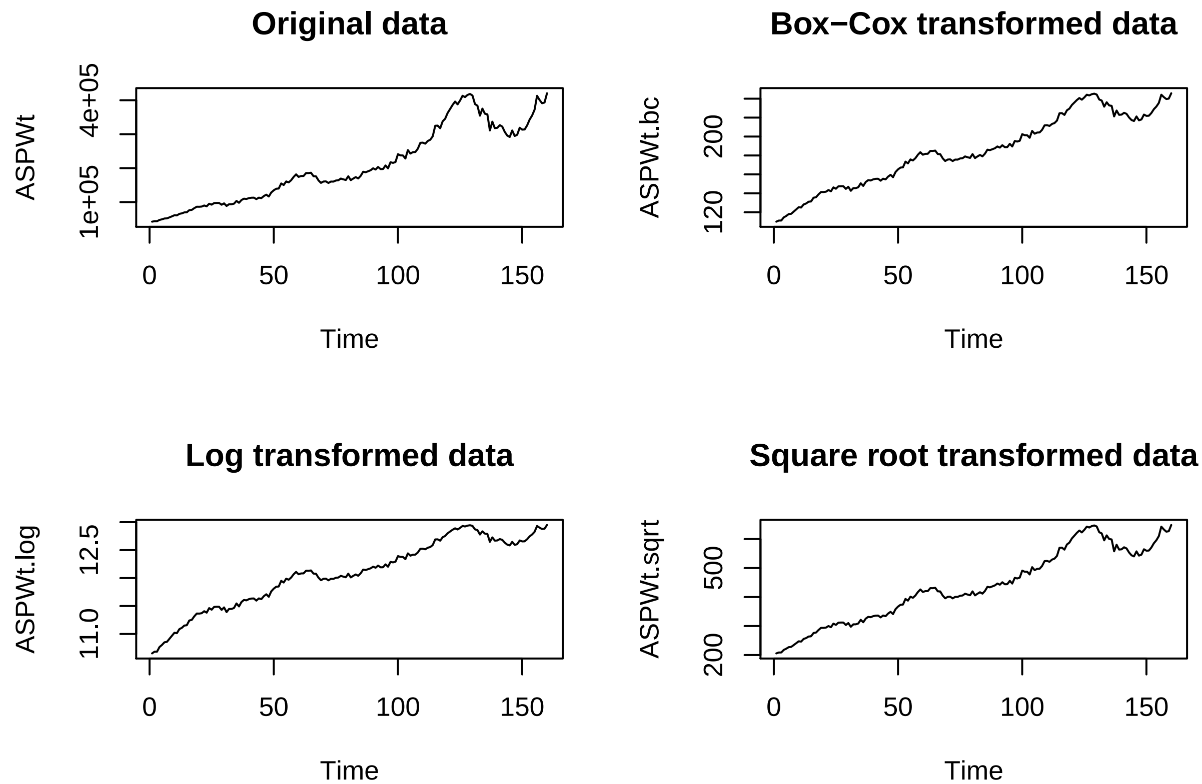
We can also consider both the log and square root transformation to make a better comparison.



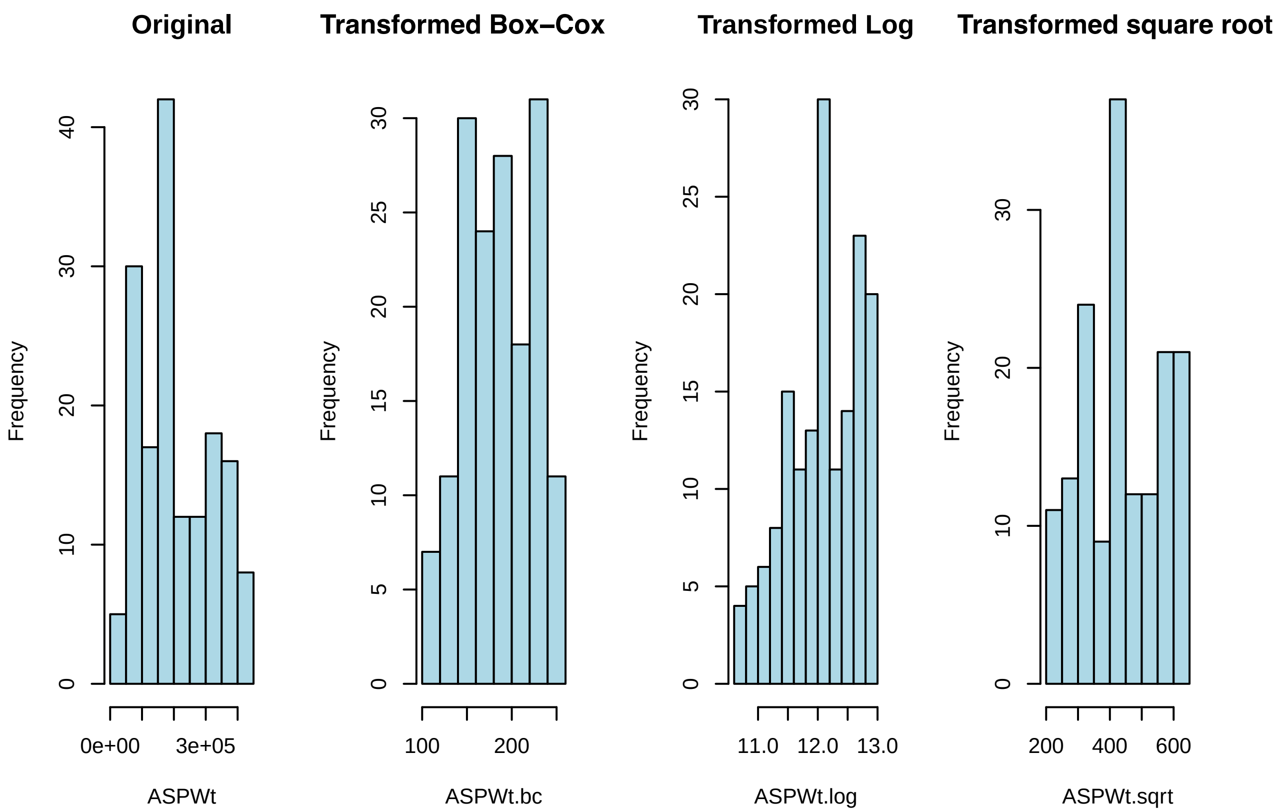

```
## [1] "lambda = "         "0.343434343434343"
```



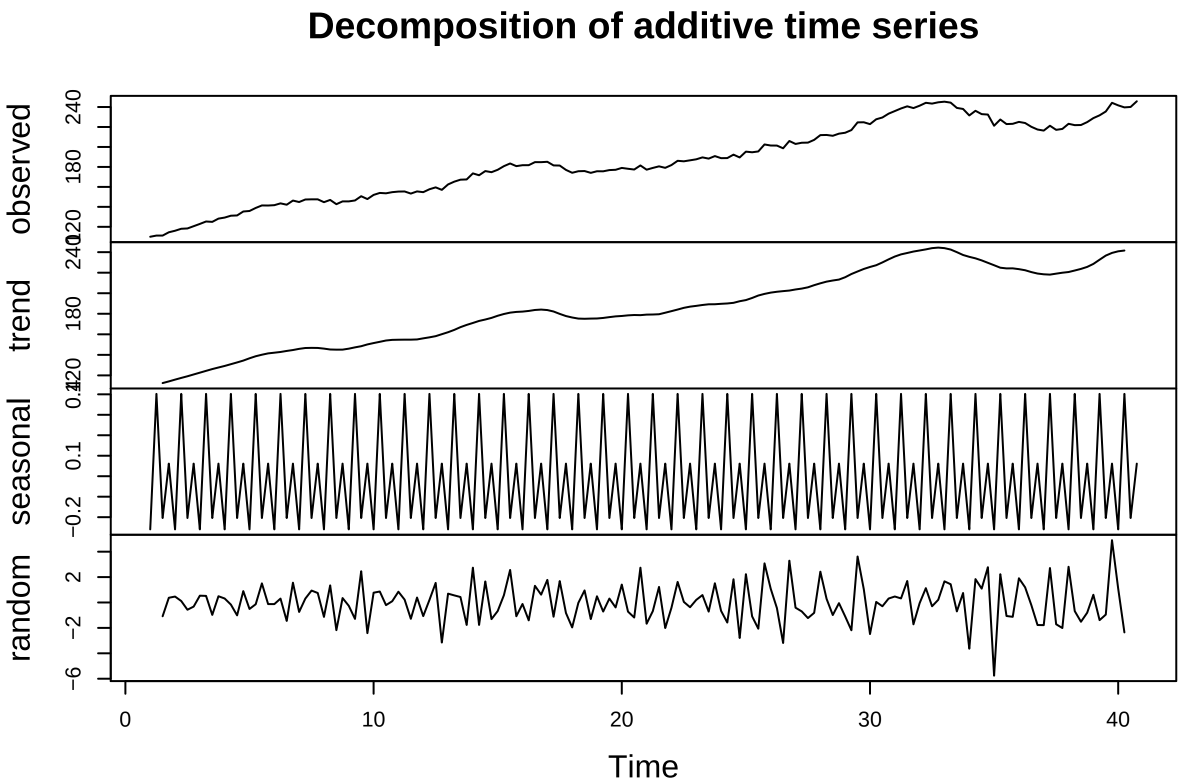

```
## Mean after BoxCox transformation: 185.4353

## Variance after BoxCox transformation: 1402.667

## Mean after Log transformation: 12.08011

## Variance after Log transformation: 0.3667761

## Mean after Square root transformation: 438.5272

## Variance after Square root transformation:  15471.87
```

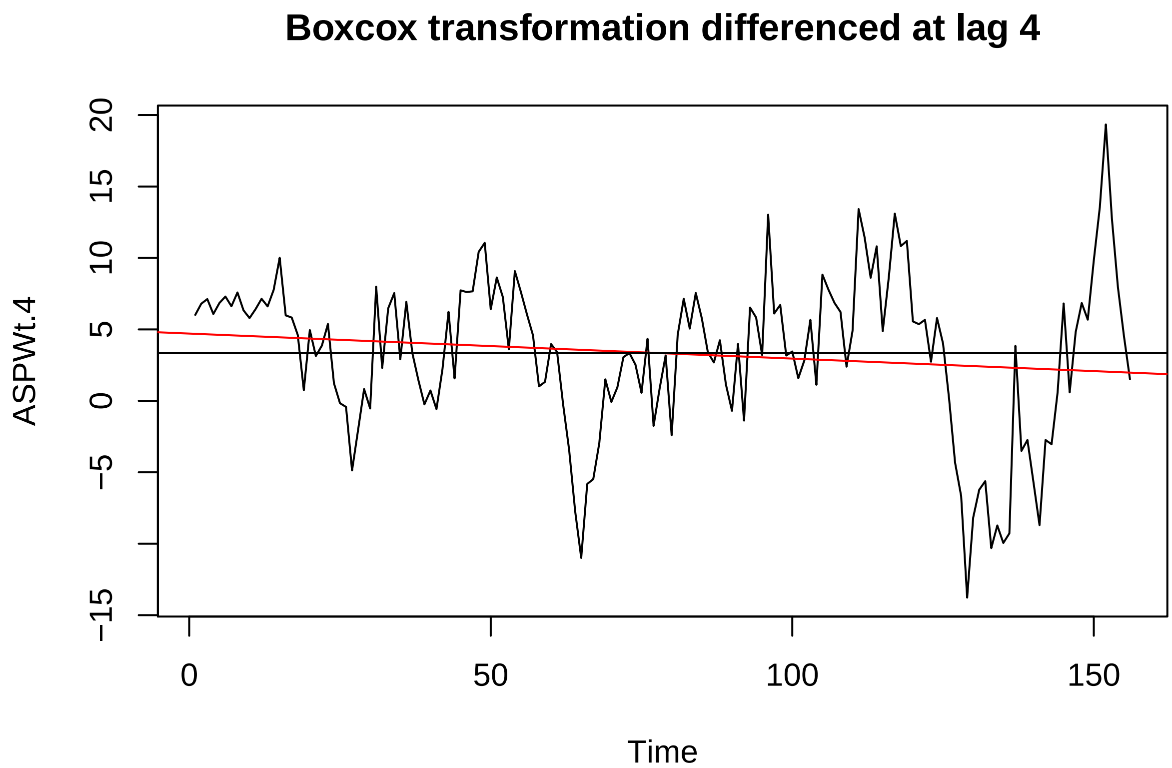


The time series plot for both the boxcox transformation and square root transformation look the same. After plotting the historgram and getting their variance we are shown that the log transformation has the smallest variance but does not make our model symmetrical. Since the interval does not include $\lambda = 0$, our best option would be Box cox transformation: $Y_t = \frac{1}{\lambda}(X_t^\lambda - 1)$;; it has the next smallest variance and has lessened the volatility by making our model more symmetrical when looking at the boxcox histogram.


Original
Transformed Box−Cox
Transformed Log
Transformed square root
Frequency
ASPWt
ASPWt.bc
ASPWt.log
ASPWt.sqrt
0e+00
3e+05
100
200
11.0
12.0
13.0
400
600

Decomposition of the boxcox transformation shows seasonality and almost linear trend, so our model needs to be differenced.
In order to remove seasonality we can difference at lag 4 since the data in the model is distributed quarterly.

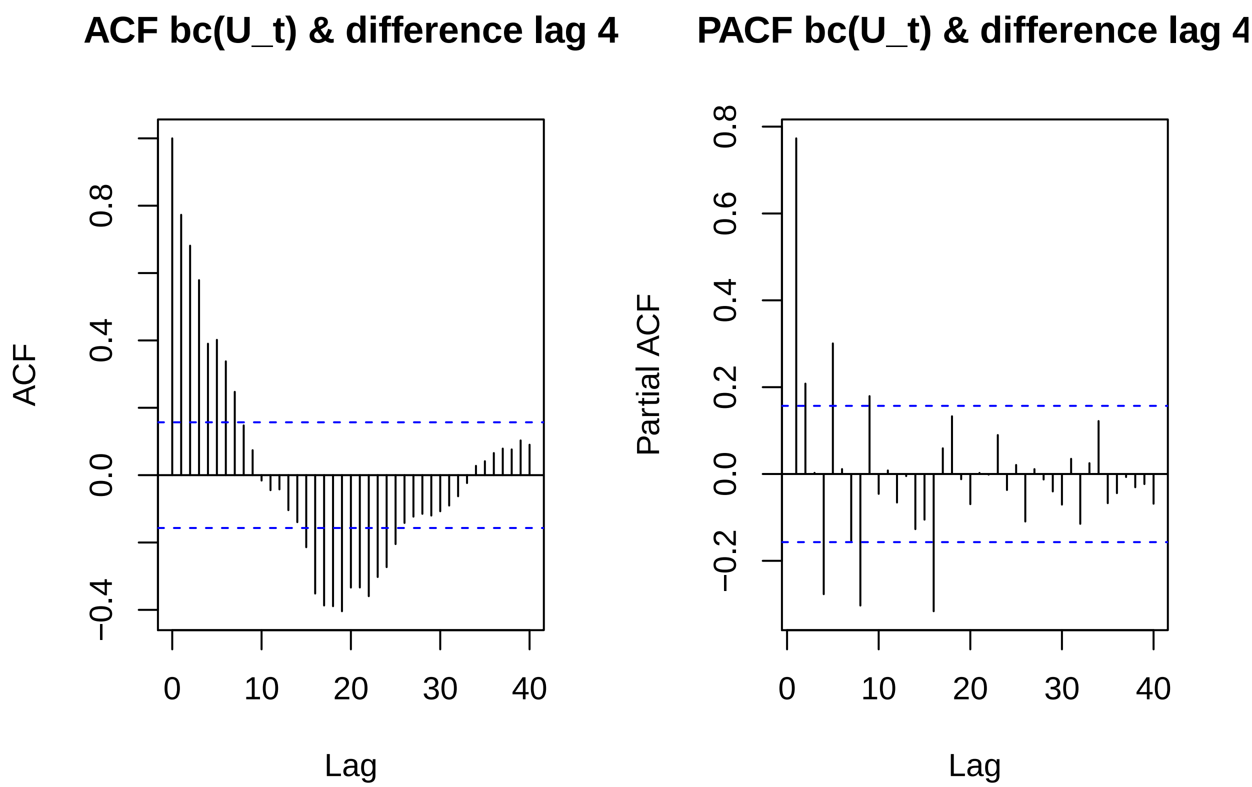




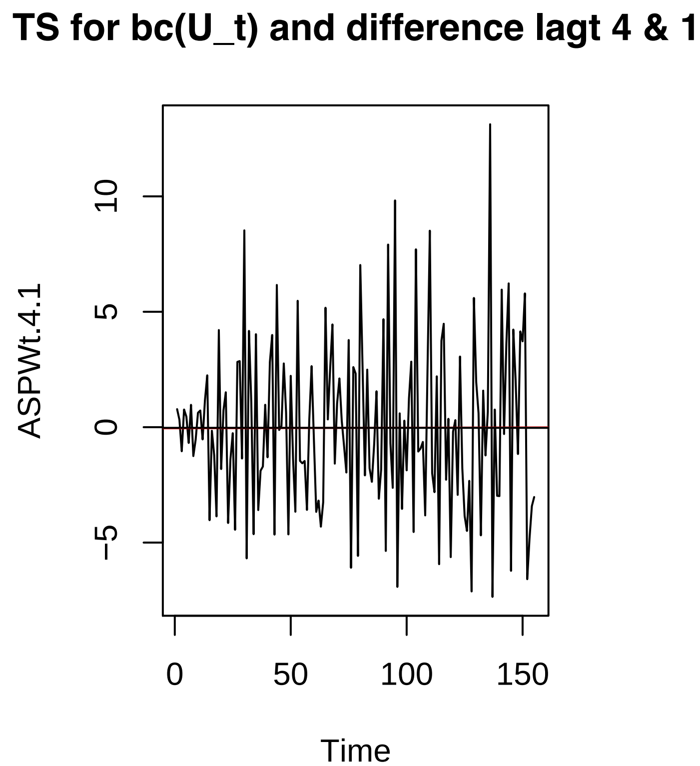

```
##   variance after boxcox transformation: 1402.667
```

```
##   variance after boxcox and difference at lag 4: 30.21134
```

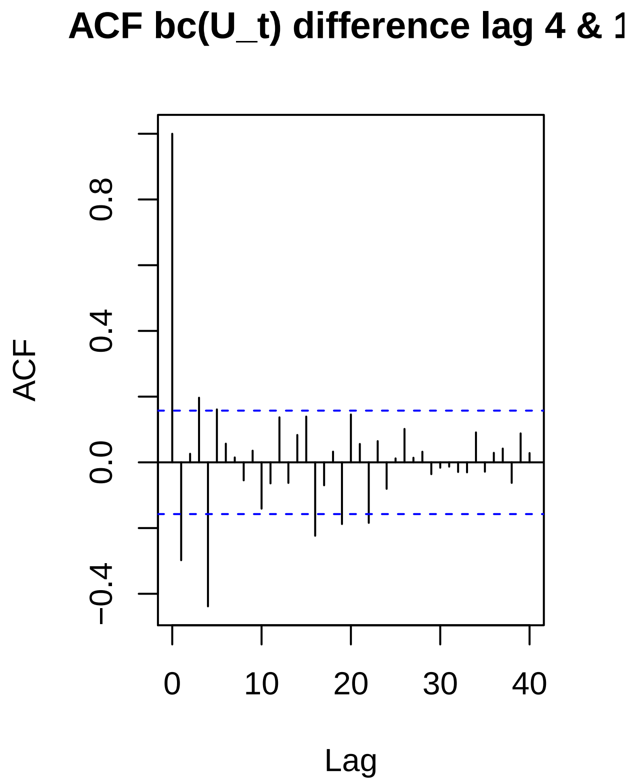
Differencing was successful as variance decreased for the model. After plotting the acf and pacf we can see the model still depicts non-stationarity and needs to be differentiated again. But, there is no longer a visible seasonal component. To remove the trend component we differentiate at lag 1 and the variance decreased again.

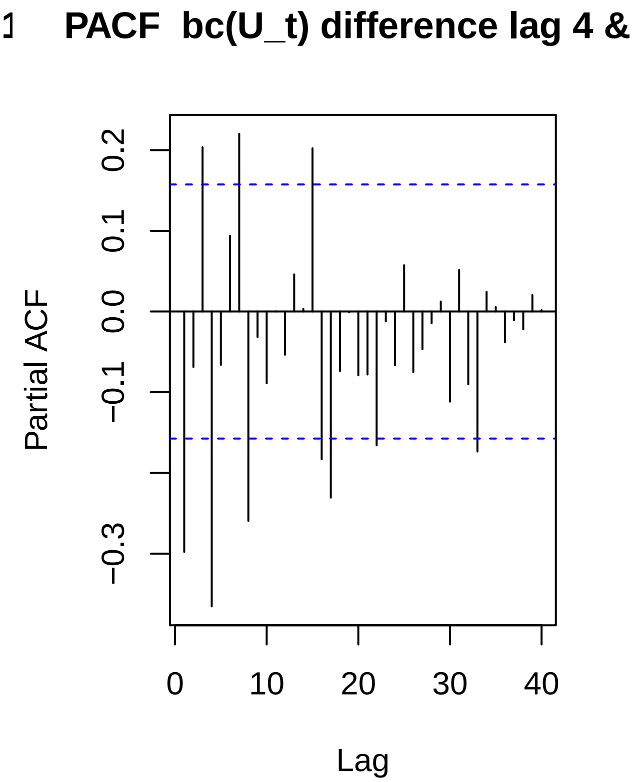


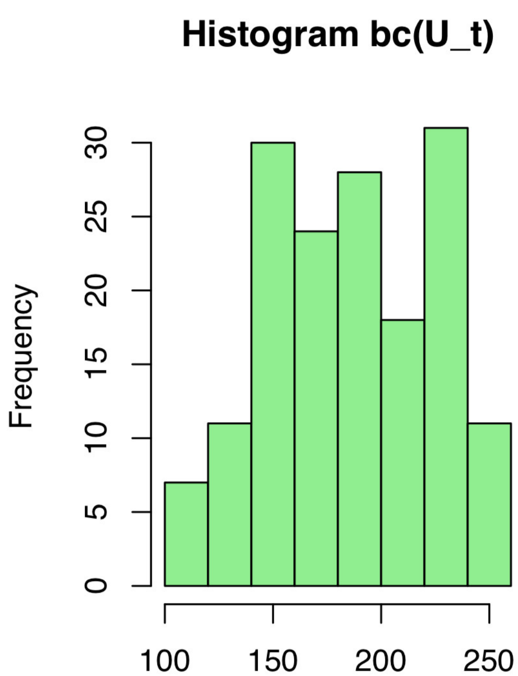

```
##  variance after boxcox and difference at lag 4 & 1: 13.73318
```

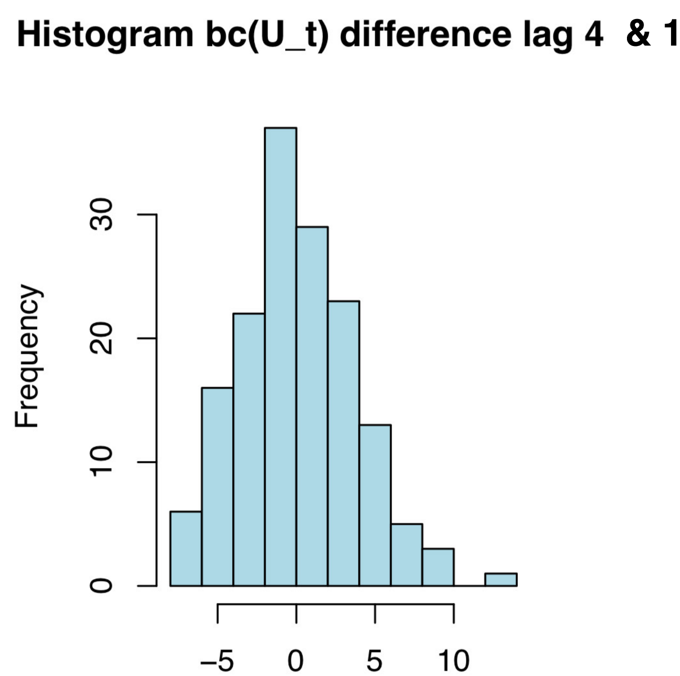


Looking at our model transformed and differenced at lag 4 & 1, we see the ACF and PACF look stationary; Our histrogram also appears more gaussian and symmetric compared to the histogram of original data. The model now appears stationary.

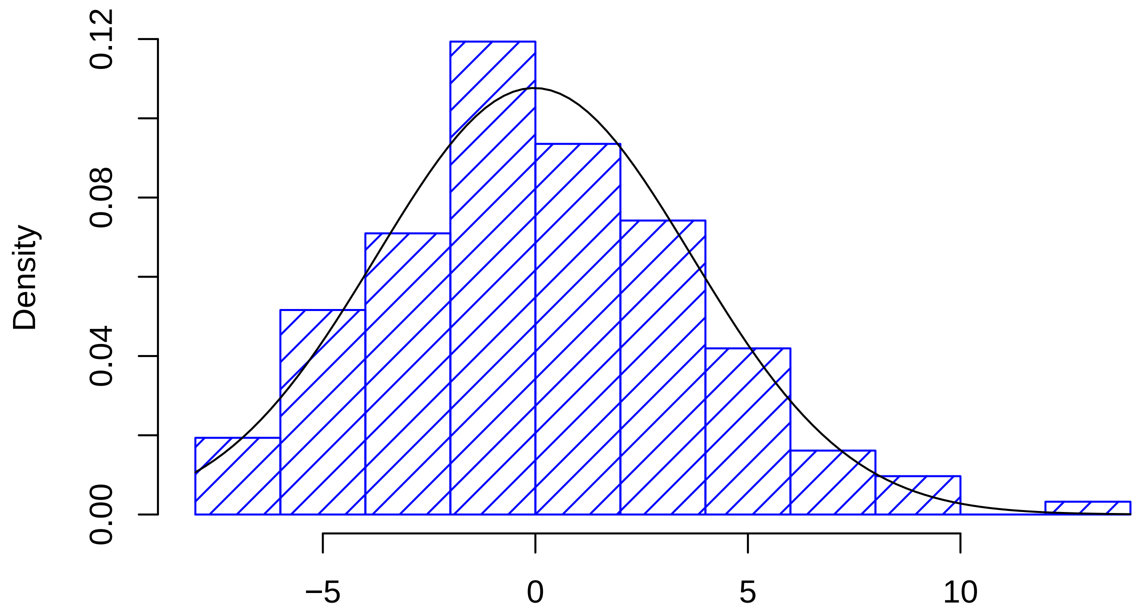

ACF bc(U_t) difference lag 4 & 1
ACF
−0.4
0.0
0.4
0.8
0
10
20
30
40
Lag


PACF bc(U_t) difference lag 4 &
Partial ACF
−0.3
−0.1
0.0
0.1
0.2
0
10
20
30
40
Lag

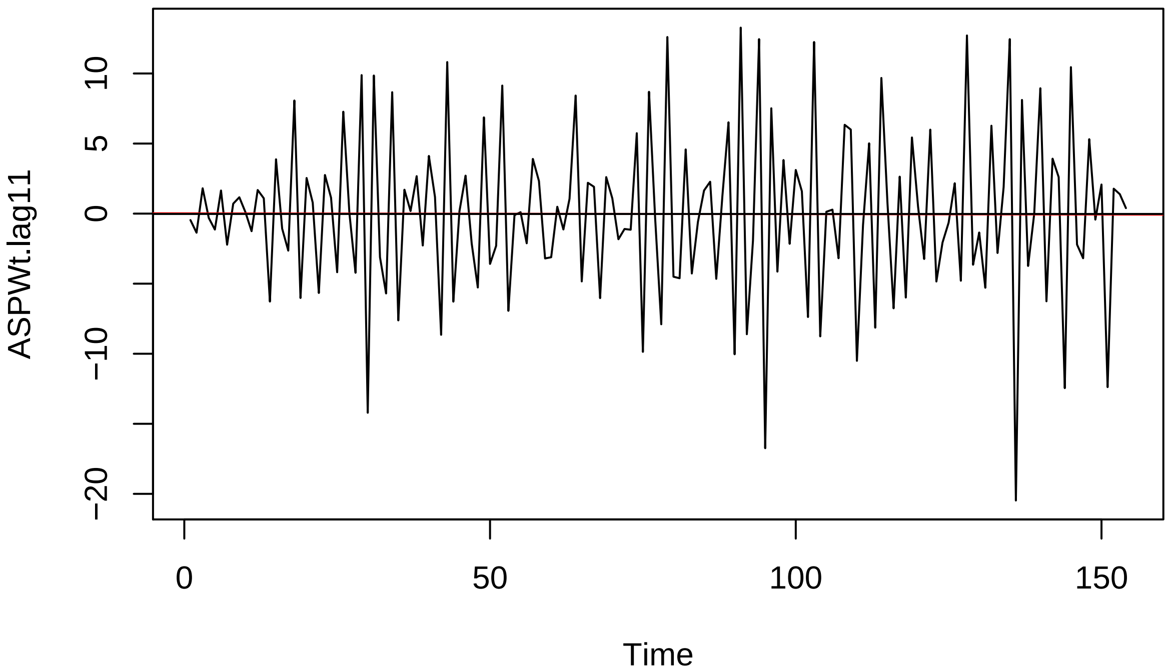

Histogram bc(U_t)
Frequency
0
5
10
15
20
25
30
100
150
200
250


Histogram bc(U_t) difference lag 4 & 1
Frequency
0
10
20
30
−5
0
5
10

To see the normality,I plotted the histogram of transformed and differenced data with normal curve. The plot shows nearly normal feature and acceptable difference so we may proceed to model identification.



Alternately, we can check to see if our model needs to be differenciated a third time. After differencing at lag 1 another time, we see our mean and variance increased which means this model is not a better fit.



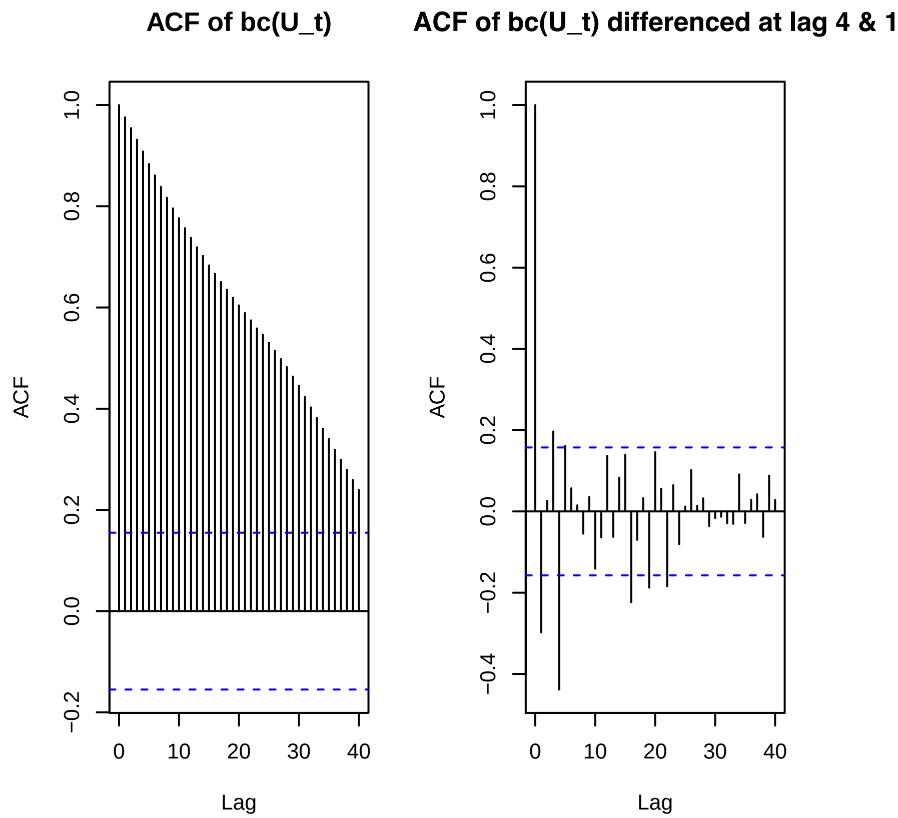

```
## [1] -0.0247247
```

```
## [1] 35.81816
```

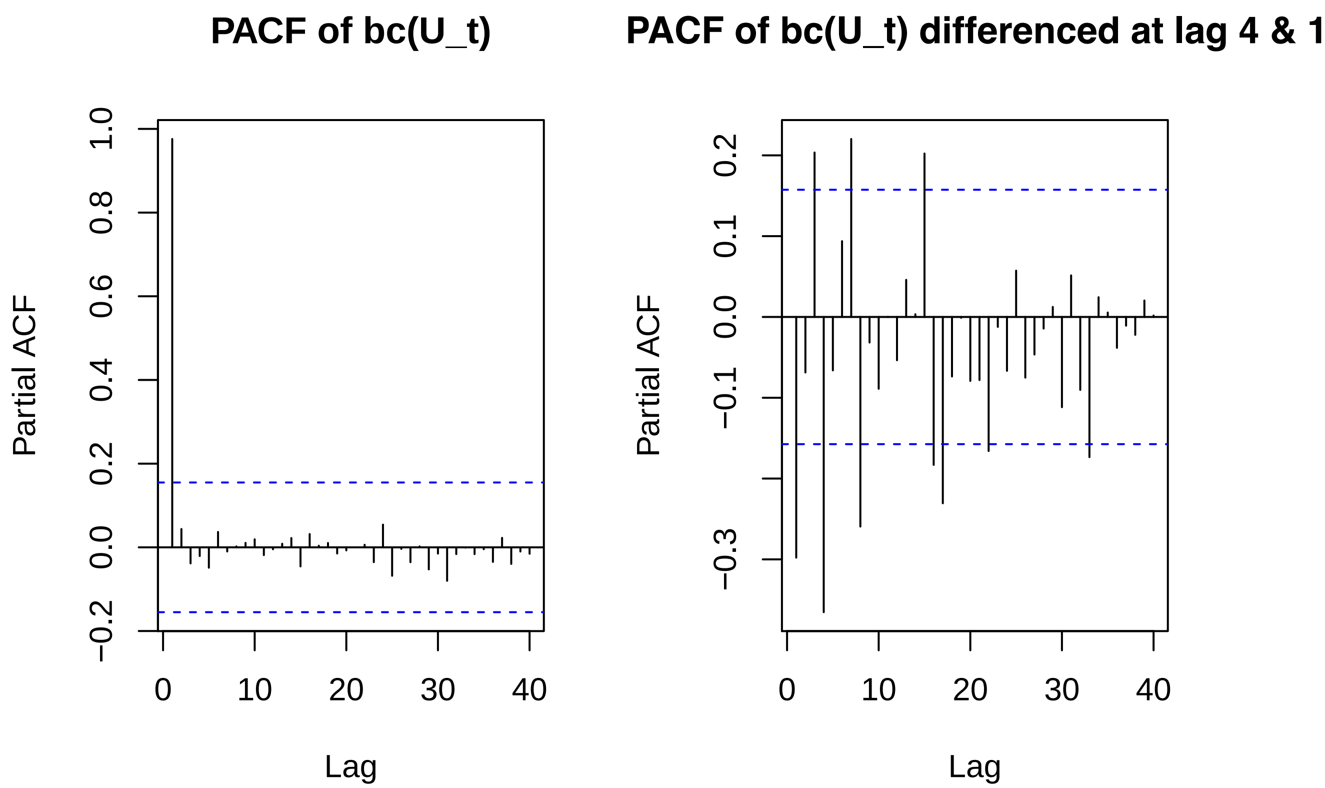
We proceed with the previous model, not this one.

Model Identification

To see the effect of differencing and also get to know more about the MA part of our model, lets plot ACF. Looking at the ACF plot of bc(U_t) and differenced at lag 4 & 1 we see the ACF is outside the confidence interval at lags 1,3,4,16,19,22.



Lets also look at the pacf of the transformed and differenced data. From the PACF, lags 1,3,4,7,8,15,16,17,22 are outside of the confidence intervals.



Lets look at some candidate models for our SARIMA(p,d,q) x (P,D,Q).

We know from the boxcox transformation that we differenced once to remove seasonality and differenced again to remove trend. The data is also distributed quarterly:
s=4, D=1, d=1;

Looking at the acf we see spikes at lags 4 and 16. These are all multiples of 4 which would give us:
Q=1,2;

Since in our PACF graph, lags 1,4,and 16, which are multiple of 4, are all outside of the confidence interval, then for the seasonal AR part, 1,2,3,4 are all possible for the model:
P=1,2,3,4;

Among the lags outside of confidence interval in ACF, choose the order of the first spike that is smaller than 10 to test its AICc:
q=1;

Among the lags outside of confidence interval in PACF, choose the order of the first spike that is smaller than 10 to test its AICc:
p=1;

Since lag 22 is the largest lag outside of confidence interval for both ACF and PACF, possible models include: AR(22) and MA(22).

After checking the AICc of AR(22) and MA(22), I got AICc of MA(22) = 790.2941 and AICc of of AR(22) = 795.7016, in which MA(22) could be potential model since its AICc is smaller.

```r
#AICc of MA(22) # 790.2941
AICc(arima(ASPWt.bc, order = c(0,1,22), seasonal=list(order=c(0,1,0),period=4),method = "ML"))
```

```
## [1] 790.2941
```

```r
#AICc of AR(22) # 795.7016
AICc(arima(ASPWt.bc, order = c(22,1,0), seasonal=list(order=c(0,1,0),period=4),method = "ML"))
```

```
## [1] 795.7016
```

To check the AICc of candidate models, I set up a for-loop to show AICc and compare them to each other to get the most adequate model. Remember that d=1,D=1.

```
##    p q P Q     AICc
## 58 1 0 4 2 773.6474
```

The model with the lowest AICc is: $SARIMA(1,1,0)x(4,1,2)_4$

```r
# Model
model <- arima(ASPWt.bc, order = c(1,1,0), seasonal=list(order=c(4,1,2),period=4),method = "ML")
print(model)
```

```
##
## Call:
## arima(x = ASPWt.bc, order = c(1, 1, 0), seasonal = list(order = c(4, 1, 2),
##     period = 4), method = "ML")
##
## Coefficients:
##           ar1    sar1    sar2    sar3     sar4     sma1     sma2
##       -0.2555  -0.424  0.1972  0.0978  -0.1971  -0.3014  -0.6986
## s.e.   0.0807   0.168  0.0976  0.0938   0.0946   0.1682   0.1639
##
## sigma^2 estimated as 7.033:  log likelihood = -378.46,  aic = 772.91
```

```
## [1] "AICc for Model 1 =" "773.647419357496"
```

Since "sar3" is insignificant we can let this coefficient be 0.

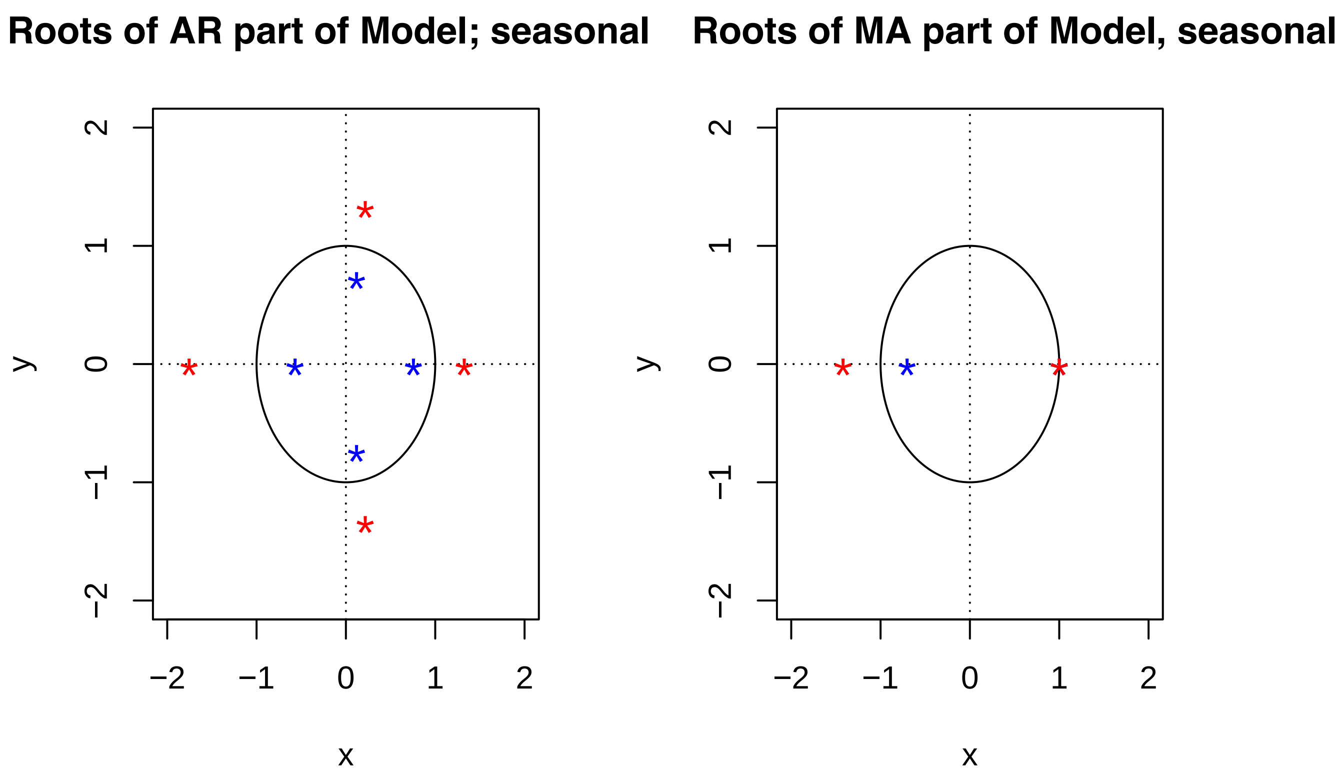
```r
# Revised Model
model <- arima(ASPWt.bc, order = c(1,1,0), seasonal=list(order=c(4,1,2),period=4),
      fixed = c(NA,NA,NA,0,NA,NA,NA),transform.pars = FALSE, method = "ML")
print(model)
```

```
##
## Call:
## arima(x = ASPWt.bc, order = c(1, 1, 0), seasonal = list(order = c(4, 1, 2),
##     period = 4), transform.pars = FALSE, fixed = c(NA, NA, NA, 0, NA, NA, NA),
##     method = "ML")
##
## Coefficients:
```

```
##          ar1     sar1    sar2  sar3     sar4     sma1     sma2
##       -0.264  -0.4227  0.1626     0  -0.2362  -0.2962  -0.7038
## s.e.   0.080   0.1679  0.0897     0   0.0835   0.1739   0.1676
##
## sigma^2 estimated as 7.046:  log likelihood = -379,  aic = 772.01
```

To check whether the model is invertible and causal, I plotted the roots of different parts of the model to see if they are outside the unit circle.

```r
par(mfrow=c(1,2))
source("plot.roots.R")
plot.roots(NULL,polyroot(c(1,-0.4227,0.1626,0,-0.2362)), main="roots of ar part of Model; seasonal")
plot.roots(NULL,polyroot(c(1,-0.2962,-0.7038)), main="oots of ma part of Model, seasonal")
```



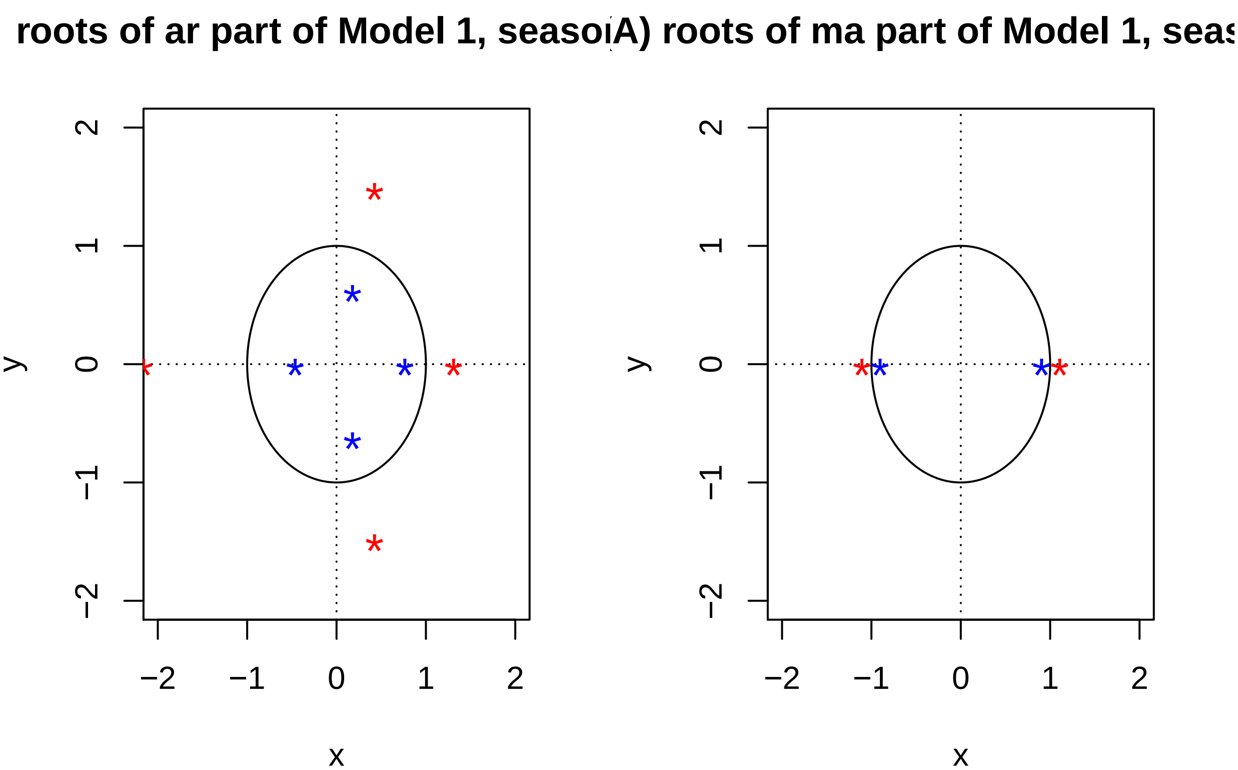
This is not a sufficent model as one of the roots for the seasonal MA is within the unit circle (on the line). We must go back to the revised model and change “sma1” to 0 as its coefficent as it may be insignificant.

```r
model1 <- AICc(arima(ASPWt.bc, order = c(1,1,0),seasonal=list(order=c(4,1,2),period=4),fixed = c(NA,NA,
print(model1)
```

```
## [1] 776.8292
```

Since the SAR and SMA polynomial roots for Model 1 are outside of the unit circle, Model 1 is both causal and invertible.

```r
# Check their invertibility
# For same model: SARIMA(1,1,0)*(4,1,2)^4
par(mfrow=c(1,2))
source("plot.roots.R")
plot.roots(NULL,polyroot(c(1,-0.6566,0.1723,0,-0.1484)), main="roots of ar part of Model 1, seasonal")
plot.roots(NULL,polyroot(c(1,0,-0.8164)), main="(A) roots of ma part of Model 1, seasonal")
```



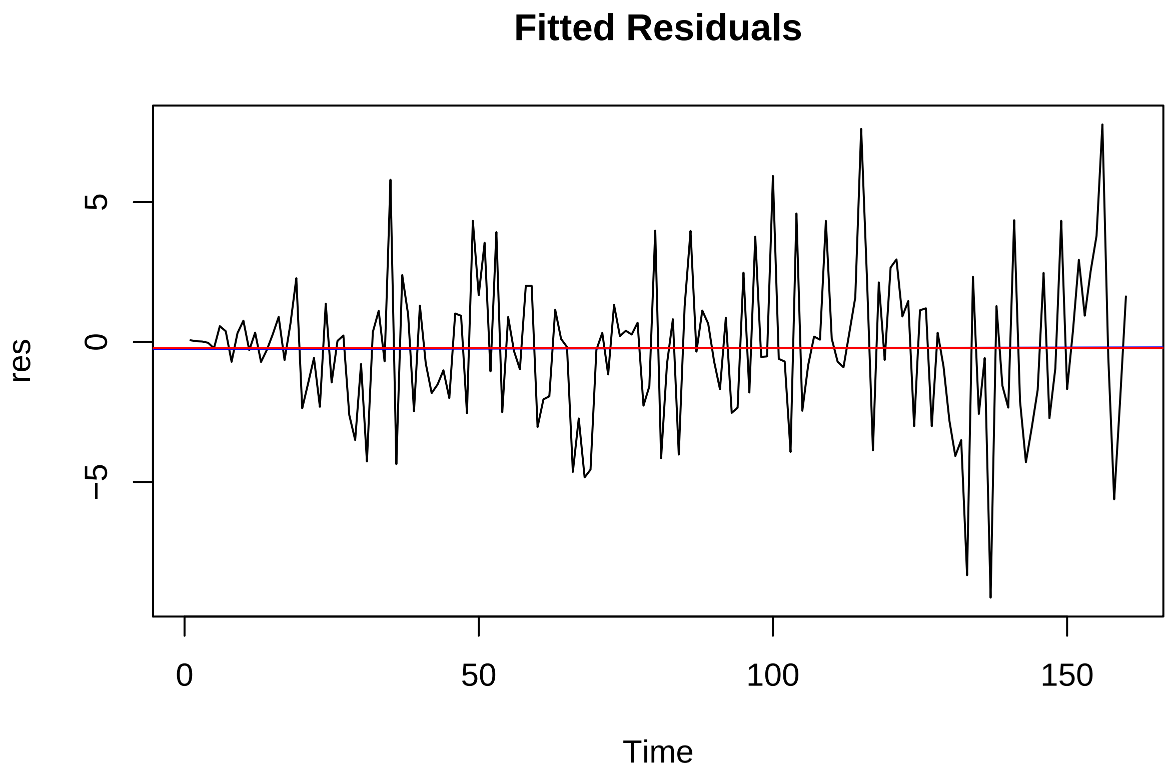
Final model: $(X_t - 0.2382X_{0.0809}B^4)(1 - B)(1 - B^4)(X_t - 0.6566X_{0.0947}B^4 + 0.1723X_{0.1064}B^8 - 0.1484X_{0.1205}B^16)X_t = (Z_t - 0.8164X_{0.1205}B^4)$

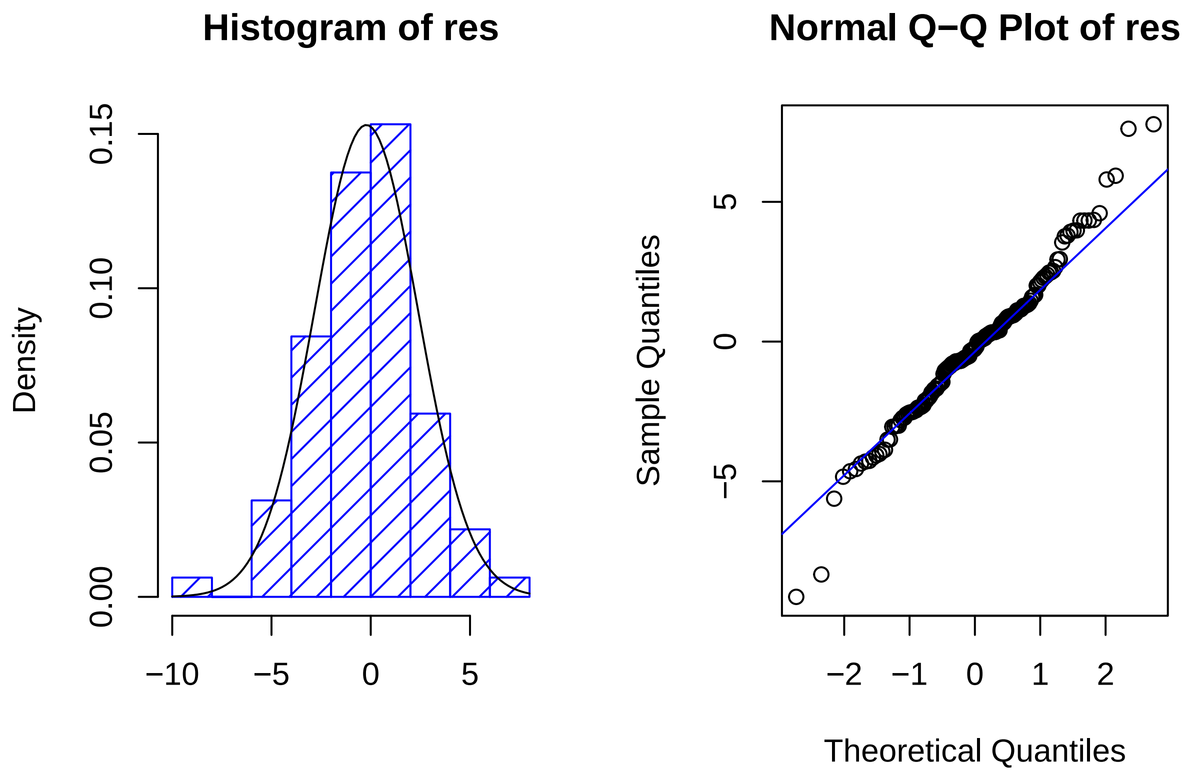
Next we must perform diagnostics on revised model 1 to see if the residuals appear to be white noise and if they appear normally distributed.

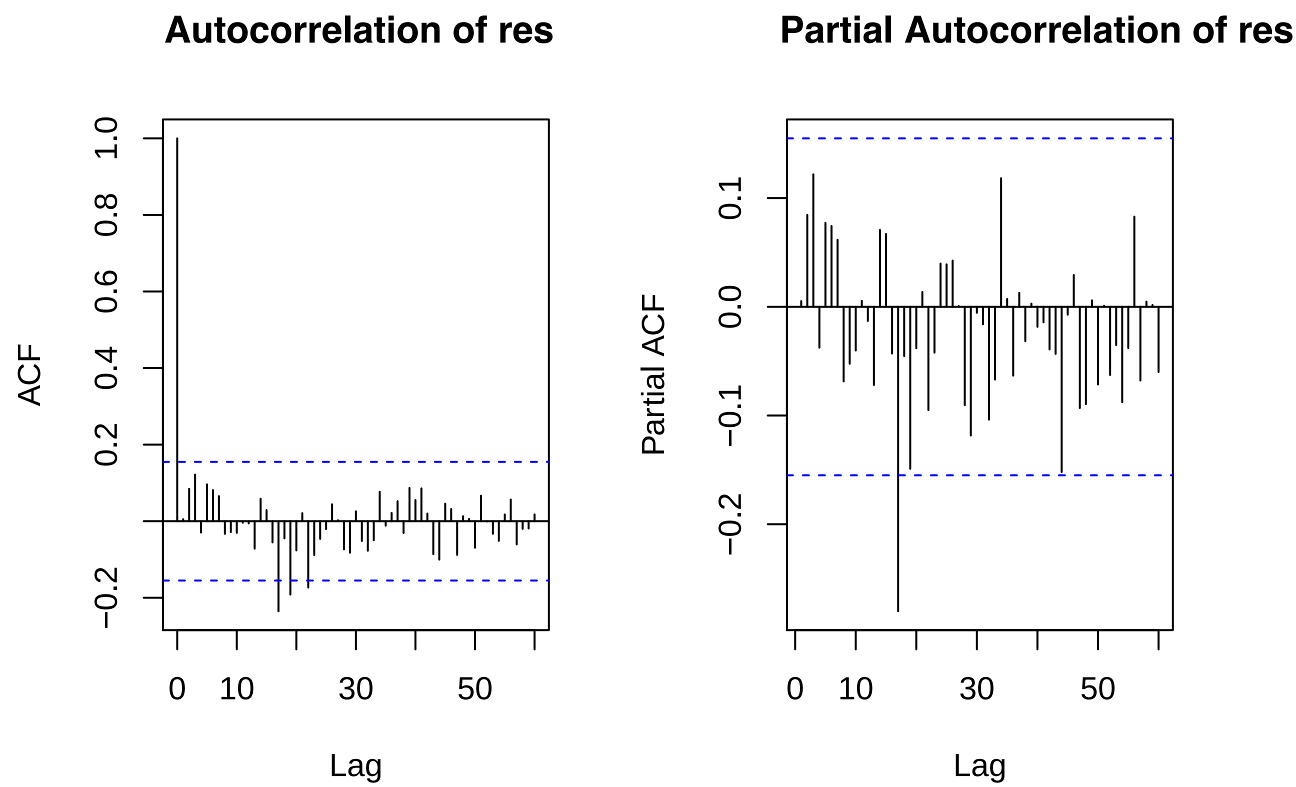
First, we check the white noise assumption with some plots.

```
## Mean of residuals: -0.2223735
```

```
## Variance of residuals: 6.806938
```


Fitted Residuals
res
5
0
−5
0
50
100
150
Time


Histogram of res
Density
0.00
0.05
0.10
0.15
−10
−5
0
5
Normal Q−Q Plot of res
Sample Quantiles
−5
0
5
−2
−1
0
1
2
Theoretical Quantiles




```r
#Test for independence of residuals
Box.test(res, lag = 14, type = c("Box-Pierce"), fitdf = 7) #lag = sqrt(n)
```

```
##
##  Box-Pierce test
##
## data:  res
## X-squared = 8.7254, df = 7, p-value = 0.273
```

```r
Box.test(res, lag = 14, type = c("Ljung-Box"), fitdf = 7) #fitdf = # of parameters
```

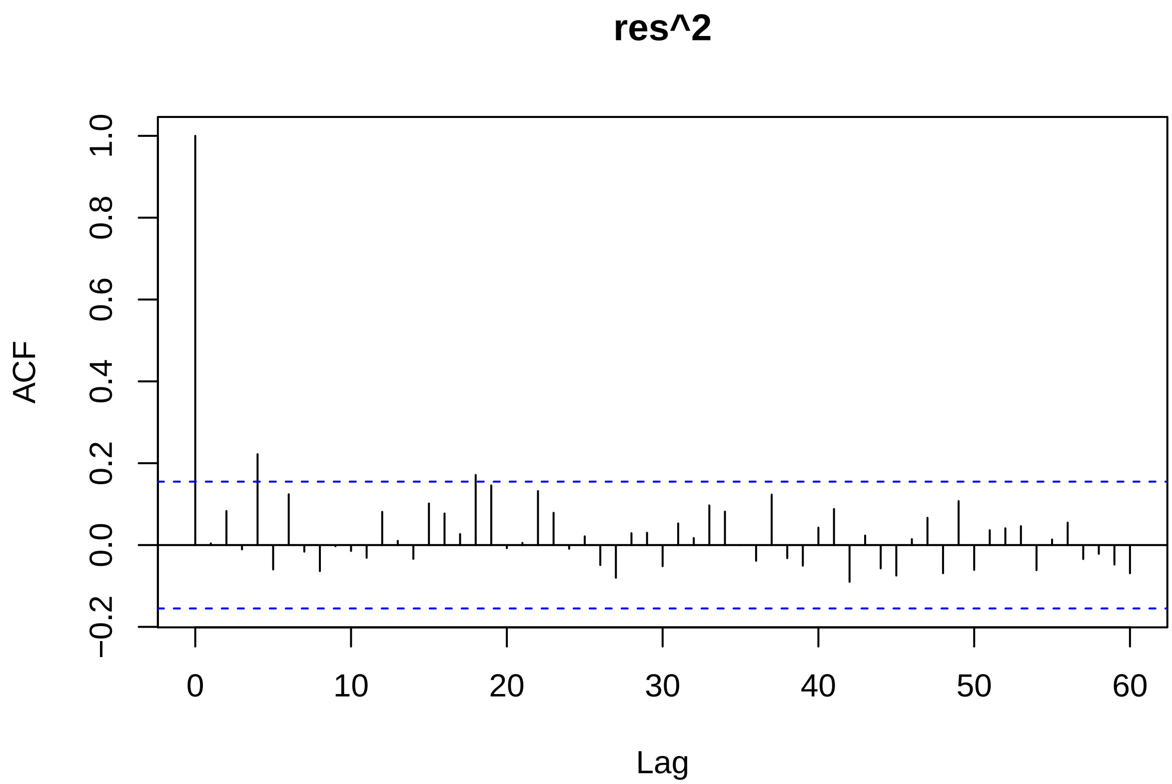
```
##
##  Box-Ljung test
##
## data:  res
## X-squared = 9.1758, df = 7, p-value = 0.2403
```

```r
Box.test(res^2, lag = 14, type = c("Ljung-Box"), fitdf = 0)
```

```
##
##  Box-Ljung test
##
## data:  res^2
## X-squared = 14.862, df = 14, p-value = 0.3877
```

```r
# Test for normality of residuals
shapiro.test(res)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res
## W = 0.98106, p-value = 0.02723
```




```
##
## Call:
## ar(x = res, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  6.807
```

The ACF extends slightly beyond the confidence interval around lag 17,19 but after running the Yule-walker test on our fitted residuals we found our model followed an AR(0), and thus resembles a white noise.

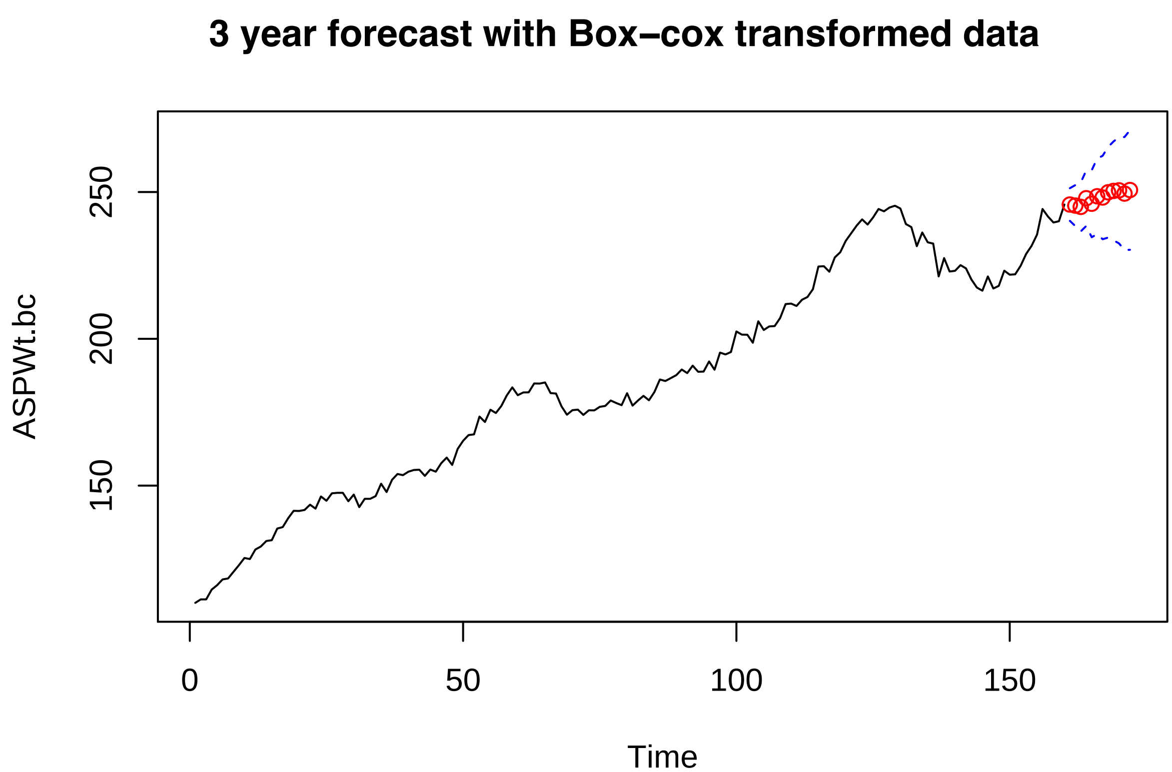
Residuals do not seem to display a trend, change in variance, or seasonal component. Both our histogram and QQ plot appear to be normally distributed. Performing diagnostic checks, our Model does not pass the Shapiro Wilk, but passes all other tests;Box-Pierce, Ljung-Box and Mcleod-Li test with p-values greater than .05.

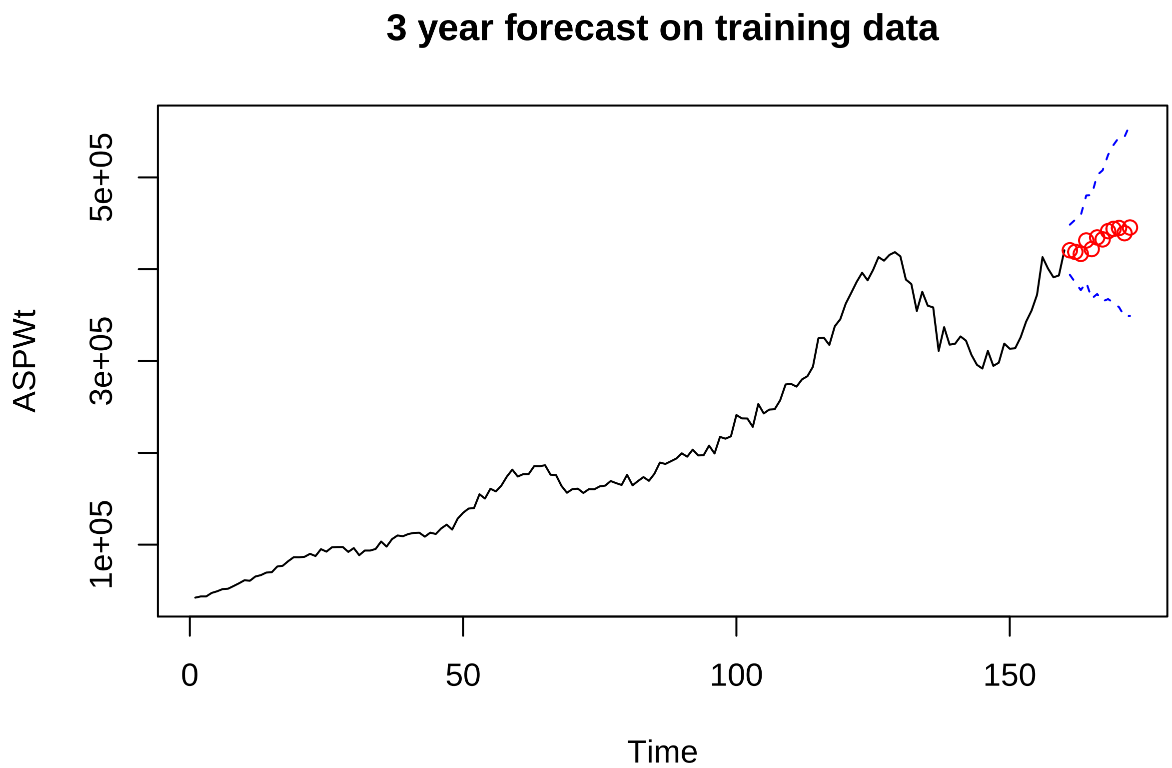
In addition, this model has the lowest AIC and is both causal and invertible. Thus it is sufficient to proceed with the model for data forecasting.

Data Forecasting

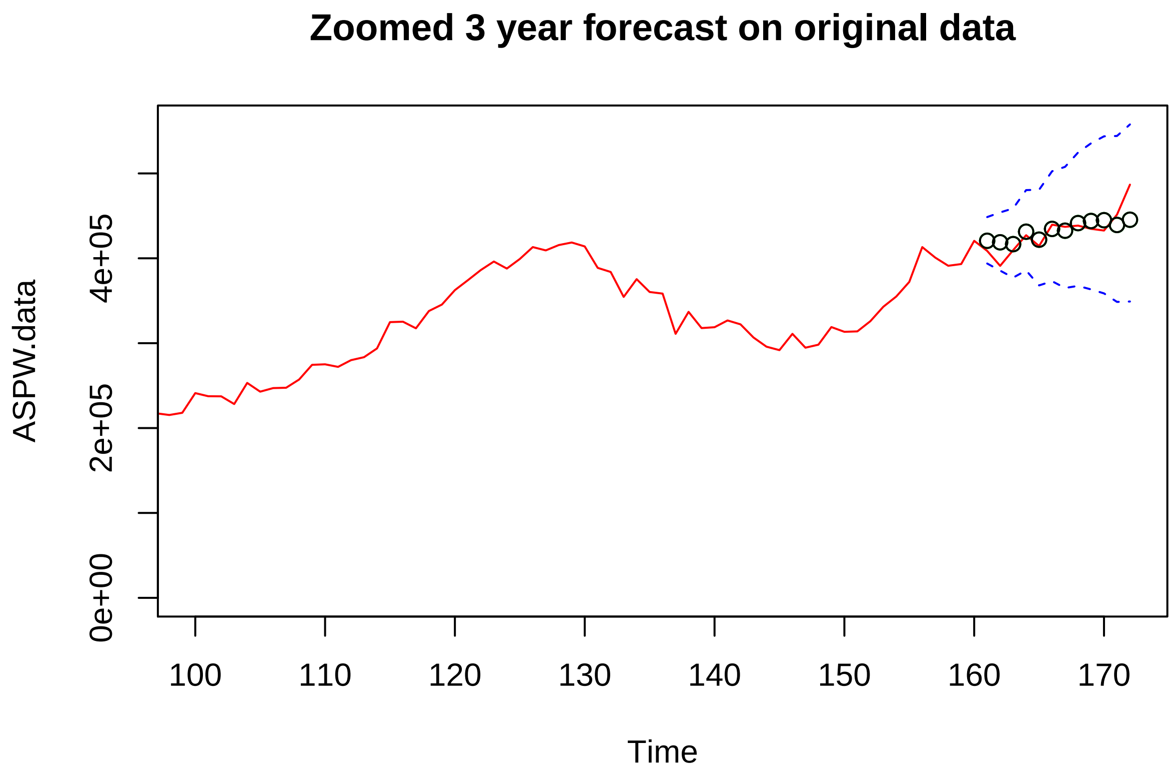
Lets look at a table with our forecasted values and its prediction bounds. Our forcasted values can be seen projected on the boxcox and original data.

```
##     Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 161       245.7423 242.1855 249.2991 240.3026 251.1820
## 162       245.3788 240.9075 249.8501 238.5406 252.2170
## 163       244.9393 239.6037 250.2749 236.7793 253.0994
## 164       247.8936 241.8383 253.9489 238.6328 257.1544
## 165       246.0082 238.6974 253.3189 234.8274 257.1889
## 166       248.5510 240.3105 256.7915 235.9482 261.1538
## 167       248.1325 239.0275 257.2376 234.2075 262.0575
## 168       249.9056 240.0178 259.7935 234.7834 265.0278
```





I forecasted 12 points with the raw data along with confidence intervals. The original points are dots and the forecasted points are a red line. Since the true values are within the confidence intervals of the forecasted values, it is clear that the model accurately forecasts the data.



Zoomed 3 year forecast on original data
ASPW.data
0e+00
2e+05
4e+05
100
110
120
130
140
150
160
170
Time

Conclusion

Final model: SARIMA $(1,1,0)$ x $(4,1,2)_4$

- $(X_t - 0.2382X_{t-1})(X_t - 0.6566X_{t-4} + 0.1723X_{t-8} - 0.1484X_{t-16})\nabla_4\nabla X_t = (Z_t - 0.8164X_{t-4})$

Our final model was used to forecast future sales prices of houses in the west region since it had the lowest AICc, the residuals resembled white noise, and it passed most of the diagnostic checks. We see that it has successfully forecasted values close to the true values of the data set as they fall within the 95% confidence interval. We conclude our final model,SARIMA $(1,1,0)$ x $(4,1,2)_4$, met Box-Jenkins assumptions for forecasting time series and our goal of forceasting the last 3 years of our dataset was achieved.

Acknowledgements

Thank you to TA's Sunpeng and Youhong for being readily available for general class and project questions. I would like to thank my professor Raya Feldman for showing kindness and supporting me through remote instruction. You have provided great informationand have helped equipped me with the necessary tools to complete this time series project.

References

- Data Set Link
- Lab 4-7
- Lecture 15 - Let's Do a Time Series Project!

Appendix

```r
knitr::opts_chunk$set(echo = TRUE,fig.pos = 'H')
library(ggplot2)
library(ggfortify)
#install.packages("qpcR")
#install.packages("rgl")
library(qpcR)
```

```
## Loading required package: MASS

## Loading required package: minpack.lm

## Loading required package: rgl

## Loading required package: robustbase

## Loading required package: Matrix
```

```r
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
```

```r
#read data
ASPW.csv <- read.table("ASPW.csv",sep=",", header=FALSE, skip=1, nrows=180)
#print(ASPW.csv)
ASPW <- ts(ASPW.csv[, 2], start = c(1975,01), frequency = 4)
ASPW.data <- ASPW[1:172]
ts.plot(ASPW.data, main="Raw Data")
```

```r
# training dataset
ASPWt = ASPW[c(1:160)]
# test dataset
ASPW.test = ASPW[c(161:172)]
# plot of training dataset
plot.ts(ASPWt)
fitt <- lm(ASPWt~as.numeric(1:length(ASPWt))); abline(fitt,col="red")
abline(h=mean(ASPWt),col="blue")
```

```r
par(mfrow=c(1,3))
hist(ASPWt, col="light blue",xlab = "",main = "Histogram of ASPW")
acf(ASPWt,lag.max = 30,main="ACF of ASPW")
#pacf(ASPWt,lag.max = 30,main="PACF of ASPW")

#find lambda value
require(MASS)
bc <- boxcox(ASPWt ~ as.numeric(1:length(ASPWt)))
```

```r
lambda=bc$x[which(bc$y==max(bc$y))]
print(c("lambda = ",lambda))

#boxcox transformation
ASPWt.bc <- (1/lambda)*(ASPWt^lambda-1)
plot.ts(ASPWt.bc, main = "BoxCox Transformed ASPW")
n.bc <- length(ASPWt.bc)

fit.bc <- lm(ASPWt.bc ~ as.numeric((1:n.bc)))
abline(fit.bc,col="red")

cat(" Mean after BoxCox transformation:", mean(ASPWt.bc))
abline(h=mean(ASPWt.bc),col="blue")
cat(" Variance after BoxCox transformation:", var(ASPWt.bc))

#log transformation
ASPWt.log <- log(ASPWt)

cat(" Mean after Log transformation:", mean(ASPWt.log))
cat(" Variance after Log transformation:", var(ASPWt.log))

#square root transformation
ASPWt.sqrt <- sqrt(ASPWt)

cat(" Mean after Square root transformation:", mean(ASPWt.sqrt))
cat(" Variance after Square root transformation: ", var(ASPWt.sqrt))

#compare transformations
par(mfrow = c(2,2))
```

```r
ts.plot(ASPWt, main = "Original data")
ts.plot(ASPWt.bc, main = "Box-Cox transformed data")
ts.plot(ASPWt.log, main = "Log transformed data")
ts.plot(ASPWt.sqrt, main = "Square root transformed data")
```

```r
#compare histograms
par(mfrow = c(1,4))
hist(ASPWt, col="light blue", main="Original")
hist(ASPWt.bc, col="light blue", main="Transformed with Box-Cox")
hist(ASPWt.log, col="light blue", main="Transformed with Log")
hist(ASPWt.sqrt, col="light blue", main="Transformed with square root ")
```

```r
#decompose to find trend
y1 <- ts(as.ts(ASPWt.bc), frequency = 4)
decomp1 <- decompose(y1)
plot(decomp1)
```

```r
#Differencing data at lag 4
ASPWt.4 <- diff(ASPWt.bc, lag = 4)
plot.ts(ASPWt.4, main = "Boxcox transformation differenced at lag 4")
fit4 <- lm(ASPWt.4 ~ as.numeric(1:length(ASPWt.4)))
#mean(ASPWt.4) #3.332874
abline(fit4, col = "red")
abline(h=mean(ASPWt.4))
cat(" variance after boxcox transformation:",var(ASPWt.bc))
cat(" variance after boxcox and difference at lag 4:",var(ASPWt.4))

par(mfrow=c(1,2))
```

```r
acf(ASPWt.4, lag.max = 40)
pacf(ASPWt.4, lag.max = 40)
```

```r
#Differencing data at lag 1
ASPWt.4.1 <-diff(ASPWt.4, lag = 1)
plot.ts(ASPWt.4.1, main = "TS for bc(U_t) and difference lagt 4 & 1")
fit4.1 <- lm(ASPWt.4.1 ~ as.numeric(1:length(ASPWt.4.1)))
#mean(ASPWt.4.1) #-0.02907558
abline(fit4.1, col = "red")
abline(h=mean(ASPWt.4.1))
cat(" variance after boxcox and difference at lag 4 & 1:",var(ASPWt.4.1)) #13.73318

par(mfrow=c(1,2))
```

```r
acf(ASPWt.4.1, lag.max = 40,  main = "ACF after difference at lag 4 and lag 1")
pacf(ASPWt.4.1, lag.max = 40,  main = "PACF after difference at lag 4 and lag 1")
```

```r
par(mfrow=c(1,2))
hist(ASPWt.bc,col = "light green",xlab = "",main = "Histogram boxcox transformation")
hist(ASPWt.4.1, col="light blue", xlab="", main="Histogram difference at lag 4 & 1")
```

```r
# Histogram of transformed and differenced data with normal
# curve
hist(ASPWt.4.1, density = 10, breaks = 10,
     col = "blue",xlab = "",prob=TRUE)
m <- mean(ASPWt.4.1)
std <- sqrt(var(ASPWt.4.1))
curve(dnorm(x,m,std),add=TRUE)

#ACF
par(mfrow=c(1,2))
```

```r
acf(ASPWt.bc, lag.max = 40, main="ACF of bc(U_t)")
acf(ASPWt.4.1, lag.max = 40, main= "ACF of bc(U_t) differenced at lag 4 &1")
```

```r
#PACF
par(mfrow=c(1,2))
pacf(ASPWt.bc, lag.max = 40, main="PACF of bc(U_t)")
pacf(ASPWt.4.1, lag.max = 40, main= "PACF of bc(U_t) differenced at lag 4 & 1")
```

```r
#AICc of MA(22) # 790.2941
AICc(arima(ASPWt.bc, order = c(0,1,22), seasonal=list(order=c(0,1,0),period=4),method = "ML"))

#AICc of AR(22) # 795.7016
AICc(arima(ASPWt.bc, order = c(22,1,0), seasonal=list(order=c(0,1,0),period=4),method = "ML"))


# Candidate models:
df <- expand.grid(p=0:1, q=0:1, P=0:4, Q=0:2)
df <- cbind(df, AICc=NA)
# Compute AICc:
for (i in 1:nrow(df)) {
  sarima.obj <- NULL
  try(arima.obj <- arima(ASPWt.bc, order=c(df$p[i], 1, df$q[i]),
                         seasonal=list(order=c(df$P[i], 1, df$Q[i]), period=4),
                         method="ML"))
  if (!is.null(arima.obj)) { df$AICc[i] <- AICc(arima.obj) }
  # print(df[i, ])
}
df[which.min(df$AICc), ] #773.6474


# Model 1
model1 <- arima(ASPWt.bc, order = c(1,1,0), seasonal=list(order=c(4,1,2),period=4),method = "ML")
print(model1)
print(c("AICc for Model 1 =",AICc(arima(ASPWt.bc, order = c(1,1,0),
                                        seasonal=list(order=c(4,1,2),period=4),method = "ML"))))
#773.647419357496


# Revised Model 1
model <- arima(ASPWt.bc, order = c(1,1,0), seasonal=list(order=c(4,1,2),period=4),
               fixed = c(NA,NA,NA,0,NA,NA,NA),transform.pars = FALSE, method = "ML")
print(model1)
model <- AICc(arima(ASPWt.bc, order = c(1,1,0),
                    seasonal=list(order=c(4,1,2),period=4),fixed = c(NA,NA,NA,0,NA,NA,NA),
                    transform.pars = FALSE, method = "ML"))
print(model) #772.7436


par(mfrow=c(1,2))
source("plot.roots.R")
plot.roots(NULL,polyroot(c(1,-0.4227,0.1626,0,-0.2362)), main="(A) roots of ar part of Model, seasonal")
plot.roots(NULL,polyroot(c(1,-0.2962,-0.7038)), main="(A) roots of ma part of Model, seasonal")
```

```r
model1 <- AICc(arima(ASPWt.bc, order = c(1,1,0),
                     seasonal=list(order=c(4,1,2),period=4),fixed = c(NA,NA,NA,0,NA,0,NA),
                     transform.pars = FALSE, method = "ML"))
```

```r
model1 <- arima(ASPWt.bc, order = c(1,1,0),seasonal=list(order=c(4,1,2),period=4),fixed = c(NA,NA,NA,0,
                transform.pars = FALSE, method = "ML")
print(model1) #AICc = 776.8292


# Check their invertibility

# For Model  SARIMA(1,1,0)*(4,1,2)^4
par(mfrow=c(1,2))
source("plot.roots.R")
plot.roots(NULL,polyroot(c(1,-0.6566,0.1723,0,-0.1484)), main="(A) roots of ar part of Model 1, seasona
plot.roots(NULL,polyroot(c(1,0,-0.8164)), main="(A) roots of ma part of Model 1, seasonal")
```

```r
#residuals plots
model1 <- arima(ASPWt.bc, order = c(1,1,0), seasonal=list(order=c(4,1,2),period=4),method = "ML")
res <- residuals(model1)
cat("Mean of residuals:",mean(res))
cat("Variance of residuals:",var(res))


#Time series plot for fitted residuals
par(mfrow=c(1, 1))
ts.plot(res, main="Fitted Residuals")
fitt <- lm(res ~ as.numeric(1:length(res))); abline(fitt, col="blue")
abline(h=mean(res),col=("red"))
```

```r
# Histogram and QQ-plot:
par(mfrow=c(1,2))
hist(res,main = "Histogram of res")
qqnorm(res, main = "Normal Q-Q Plot of res")
qqline(res,col ="blue")
```

```r
# ACF and PACF:
par(mfrow=c(1, 2))
acf(res, lag.max = 60, main="Autocorrelation of res")
pacf(res, lag.max = 60, main="Partial Autocorrelation of res")
```

```r
#Diagnostic checking for Model
#Test for independence of residuals
Box.test(res, lag = 14, type = c("Box-Pierce"), fitdf = 7) #lag = sqrt(n)

Box.test(res, lag = 14, type = c("Ljung-Box"), fitdf = 7)

Box.test(res^2, lag = 14, type = c("Ljung-Box"), fitdf = 0)

# Test for normality of residuals
shapiro.test(res)

#R model Yule-walker
acf(res^2, lag.max = 60, main = "res^2")
ar(x=res, aic = TRUE,order.max = NULL,method = c("yule-walker"))
```

```r
# R forecast transformed
# To produce graph with 20 forecasts on transformed data
fit <- arima(ASPWt.bc, order = c(1,1,0),seasonal=list(order=c(4,1,2),period=4),
             fixed = c(NA,NA,NA,0,NA,0,NA),transform.pars = FALSE, method = "ML")
forecast(fit)

pred.tr <- predict(fit, n.ahead = 12)
U.tr = pred.tr$pred + 2*pred.tr$se
L.tr = pred.tr$pred - 2*pred.tr$se

ts.plot(ASPWt.bc,xlim=c(1,length(ASPWt.bc)+12),ylim=c(min(ASPWt.bc),max(U.tr)),main = "3 year forecast w
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(ASPWt.bc)+1):(length(ASPWt.bc)+12),pred.tr$pred,col="red")
```

```r
# r forecast original
# To produce graph with forecasts on original data
pred.orig <- (lambda*pred.tr$pred+1)^(1/lambda) #ASPWt.bc = (1/lambda)*(ASPWt^lambda-1)
U = (lambda*U.tr+1)^(1/lambda)
L = (lambda*L.tr+1)^(1/lambda)
ts.plot(ASPWt,xlim=c(1,length(ASPWt)+12),ylim=c(min(ASPWt),max(U)),main = "3 year forecast on training d
lines(U,col="blue",lty="dashed")
lines(L,col="blue",lty="dashed")
points((length(ASPWt)+1):(length(ASPWt)+12),pred.orig,col="red")

# r forecast zoomed and true
# To plot zoomed forecasts and true values:
ts.plot(ASPW.data,xlim=c(100,length(ASPWt)+12),ylim=c(250,max(U)),col="red", main = "Zoomed 3 year fore
lines(U,col="blue",lty="dashed")
lines(L,col="blue",lty="dashed")
points((length(ASPWt)+1):(length(ASPWt)+12),pred.orig,col="green")
points((length(ASPWt)+1):(length(ASPWt)+12),pred.orig,col="black")
```