Modeling and Forecasting on Industrial Production for Electric and Gas Utilities

By: Erica Chen

Fall 2023

Summary

In this analysis, the focus is on the Industrial Production of Electric and Gas Utilities in the United States. The chosen dataset aims to explore trends in industrial production from 1983 to 2007. From the graph, industrial production for electric and gas utilities has demonstrated a consistent upward trend in recent years, mirroring a growing demand for these resources within the US economy. Seasonal fluctuations are evident, with summer and winter experiencing peak production due to increased air conditioning and heating demand, respectively. Conversely, production dips during spring and fall as milder weather conditions lead to lower energy consumption. Through visualizing time series data, we aim to discern notable trends and variations that can provide valuable insights for strategic decision-making, guiding investments, and infrastructure development efforts in the future.

Introduction

The dataset, comprising 300 observations from 1983 to 2007, focuses on Industrial Production data related to Electric and Gas Utilities in the United States, sourced from the Federal Reserve Economic Data (FRED) at https://www.federalreserve.gov/. I decided to restrict analysis to this period due to the impact of the 2008 financial crisis on trends.

In the initial stages, I divided the dataset into a training set and a test set, with 290 observations allocated for training and the remaining 10 for model testing and validation. Then, I employed Box-Cox transformation to normalize the dataset and obtain my optimal lambda. To address concerns related to stationarity, I applied differencing at lag 12 to eliminate seasonality and at lag 1 to remove the trend from the data. Subsequently, through analysis of the Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots, I identified potential p's and q's for my candidate models. After that, coefficients and AICc values from the candidate models play a pivotal role in determining the ultimate choice of the time series model, which undergoes checks for stationarity and invertibility.

The selected model, meeting most of the diagnostic testing, is then utilized for forecasting, and predicting future values with precision. From the final model, the true values are within the confidence interval of the forecasting, and from the zoomed-in forecasting plot, the forecasting results are very close to the true values. Therefore, the final chosen model performs well in forecasting future industrial production of electric and gas utilities.

Sections

1.1 Plot and analyze the time series

From the time series plot of “Electric and Gas Utilities from 1985 - 2007”, we can see a steady upward trend in production throughout the years. The regular fluctuations every 12 points indicate a strong seasonal pattern, with production typically peaking in winter. Overall, the plot suggests that the natural gas market is relatively stable and predictable, with a clear upward trend in the long term.

1.2 Transformation

After creating the training and testing sets, I performed a Box-Cox transformation on the “ train_data” set to stabilize the variance and enhance its normality for statistical modeling. Our optimal lambda given by the plot was 0.3838. Box-Cox transformation for stabilizing the variance is given by this equation:

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

This lambda value is then applied on the “train_data” set to obtain the “eg. bc” dataset. The effectiveness of the transformation is visually assessed through side-by-side histograms and time series plots, comparing the distribution of the original data with the transformed data. From the time series plot, the transformed data has a more stable variance across time, and the histogram of the transformed data showed a more Gaussian distribution. Therefore, I concluded that the box Cox transformed data with a lambda of 0.38383 was more appropriate than the original data.

Although the box-cox transformed data is more stabilized and normal than the original data, I could see from the decomposition plot and ACF/PACF plots of the transformed data that it wasn’t completely stationary and there were clear seasonal peaks and trends. To remove seasonality, I performed differencing at lag 12. By checking the variance, it was clear that differencing at lag 12 improved the stationarity of the data because the variance went from 0.951 to 0.0360. However, from the differenced time series plot, we could still see a trend in the regression line, implying that further differencing is needed to remove the trend. Thus, I differenced the data again at lag 1.

To prevent overfitting the model, we checked the variance to see if it increased or decreased; after differencing at lag 1, the variance of the data went down to 0.0314. From this new differenced time series plot we see no more trend and from the ACF plot, we see that the ACF's decayed quickly after lag 0. This indicates that the time series is stationary now, thus we can conclude that the de-seasonalized (at lag 12) and de-trended (at lag 1) data is the best candidate and is enough for further analyses.

1.3 Identify Model(s)

We proceed with the Box-Cox transformed data, performing differencing at lags 12 and 1 to eliminate seasonality and trend, respectively. Analyzing the ACF and PACF plots, we identify tentative SARIMA model candidates. Knowing our models are SARIMA due to seasonality, we set D=1 and s=12 for seasonal differencing. Further differencing at lag 1 for trend removal implies d=1. The ACF plot reveals potential MA parameters: q=1 or 2 based on the first two lags falling outside the confidence interval, and Q=1 or 2 due to the lag 12 and 24 exceeding the confidence bounds. Similarly, the PACF plot suggests possible AR parameters: p=1 or p=2 based on lags 1 and 2 exceeding the confidence interval, with lag 2 showing the most significant deviation. Additionally, P=1 due to the lag 12 PACF exceeding the confidence bounds. Therefore, with all the combinations of the numbers, we have eight candidate SARIMA models:

All 1's
SARIMA (1,1,1) x (1,1,1)s=12

One 2
SARIMA (2,1,1) x (1,1,1)s=12
SARIMA (1,1,2) x (1,1,1)s=12
SARIMA (1,1,1) x (1,1,2)s=12

Two 2's
SARIMA (1,1,2) x (1,1,2)s=12
SARIMA (2,1,2) x (1,1,1)s=12
SARIMA (2,1,1) x (1,1,2)s=12

All three 2's
SARIMA (2,1,2) x (1,1,2)s=12

These models will be further evaluated and compared for the lowest AICc value in the next steps.

1.4 Fit Model(s)

We fit all 8 SARIMA models and compare their coefficients, standard errors, and AICc values. After fitting all 8 models, we could see that model fit 4 - SARIMA(1,1,1)x(1,1,2) has the lowest AICc value at -326.1422. From the above coefficients table of fit 4, sma1 is not significant because the confidence interval of the estimated coefficient contains 0. Therefore, we set these coefficients to 0, and evidently, the model became better because it resulted in a smaller AICc value at -327.8048. So we now have the final two models as **fit4 -** SARIMA (1,1,1) x (1,1,2) and **fit4a** - SARIMA (1, 1, 1) x (1, 1, 2)12 with coefficient of sma2 equal to 0.

Fit4 model can be written as:

$$(1-0.5474B)(1+0.5517B^{12})(1-B)(1-B^{12})Y_t = (1-0.9584B)(1-0.1525B^{12})(1-0.5084B^{24})Z_t$$

Fit4a the model can be written as:

$$(1-0.5452B)(1+0.7148B^{12})(1-B)(1-B^{12})Y_t = (1-0.9585B)(1-0.6133B^{24})Z_t$$

Next, we need to check the models' stationarity and invertibility. To do so, I plotted all the coefficients of models Fit4 and Fit4a. For each plot, the blue dot lay within the circle and the red dot lay outside the circle. Since all the roots lie outside the unit circle, we can conclude that both models are stationary and invertible.

1.5 Diagnostic Tests

To check if the residuals of model fit4 and fit4a follow the White Noise distribution, we perform several diagnostic tools in this section. We first check normality assumptions and get the visualizations including histograms, residuals plots, and Q-Q plots. We can observe from the plots that both models follow a normal distribution from the histogram and Q-Q plot. Also, there is no trend or obvious seasonality from the time series plot of the residuals. Next, we also check for several independence assumptions by Portmanteau Statistics, which includes the Shapiro-Wilk, Box-Pierce, Ljung-Box, and the McLeod-Li test. To determine the lag and fitdf for these tests, we need to look at number of observations in the training set and our p's and q's. The lag is determined by squaring the number of observations in the training set, in which I got 17 (squaring 290). Then I set my fitdf to 2 because fitdf is p + q, and my p and q are both 1's. All the tests except for Shapiro-Wilk came back with p-values lower than 0.05, meaning that the residuals are uncorrelated and resemble a Gaussian white noise. Since the models did not pass the Shapiro-Wilk test, we need to take into account that the forecasted values might be closer than they really are. Although the models didn't pass the Shapiro-Wilk test, all other tests passed so we can continue with the other diagnostic tests. Next, the ar() function recommended order 0 for both models, meaning that the residuals are indeed white noise.

Since both models passed the same tests and failed the same test, the only difference between the two models is the AICc value. Since model fit4a has a lower AICc value, we will proceed to forecast with model fit4a.

1.6 Forecasting

In the forecasting section, we predict the gas and electric utilities from March 2007-December 2007 based on our model fit4a SARIMA(1, 1, 1) x (1, 1, 2)s=12 and then compare with the true values. We calculate upper and lower prediction intervals using the predicted values (pred.tr$pred) and their corresponding standard errors (pred.tr$se). The resulting visualization, created with a ts.plot, displays the original time series (eg.bc), the confidence intervals represented by blue dashed lines, and the predicted values for the next 10 time points indicated by red points. The predicted values look like they all lie within the confidence interval, so now we need to compare the predicted values to the true values. Before that, we need to convert the forecasting values back to the scale before the box-cox transformation. Using inverse box-cox to convert the data back to the scale before box-cox transformation, we compare the true values with predicted values.

Both the forecasted values and true values lie within the confidence interval. Upon zooming in on the lower part, it becomes evident that the forecasting results closely match the actual values. This implies that our model is adept at accurately predicting future industrial production of gas and electricity utilities for the U.S.

Conclusion

This report aims to create a forecasting model that accurately predicts the industrial production of gas and electricity in the United States. Time series techniques are employed to achieve this goal and our model - SARIMA(1, 1, 1) x (1, 1, 2)s=12 has successfully done so. We first examine this dataset to detect any potential seasonal patterns and trends from the time series plot and ACF/PACF plots. Next, Box-Cox transformation is used to appropriate data formatting, striving for a normal distribution. Subsequently, differencing is applied to achieve stationarity in the time series, followed by the use of various techniques to identify (p's and q's), select (AICc value), and validate (Portmanteau Statistics) candidate models. In the end, our final model is: SARIMA(1, 1, 1) x (1, 1, 2)s=12 (coefficient SMA = 0), and the algebraic form can be written as:

$$\boxed{(1-0.5452B)(1+0.7148B^{12})(1-B)(1-B^{12})Y_t = (1-0.9585B)(1-0.6133B^{24})Z_t}$$

The final model was able to accurately predict the future 10 months of industrial production of gas and electricity in the United States since all forecasted values fell within the confidence interval and were close to the actual values. The forecasted values match the previous trends where gas and electricity usage are usually lower during the fall and spring months, and peaks again during the winter and summer months.

Acknowledgment to Professor Raisa Feldman and Cosmin Borsa for helping with this report.

References

https://fred.stlouisfed.org/series/IPG2211A2N

Appendix Below

Erica chen 174 Final Project

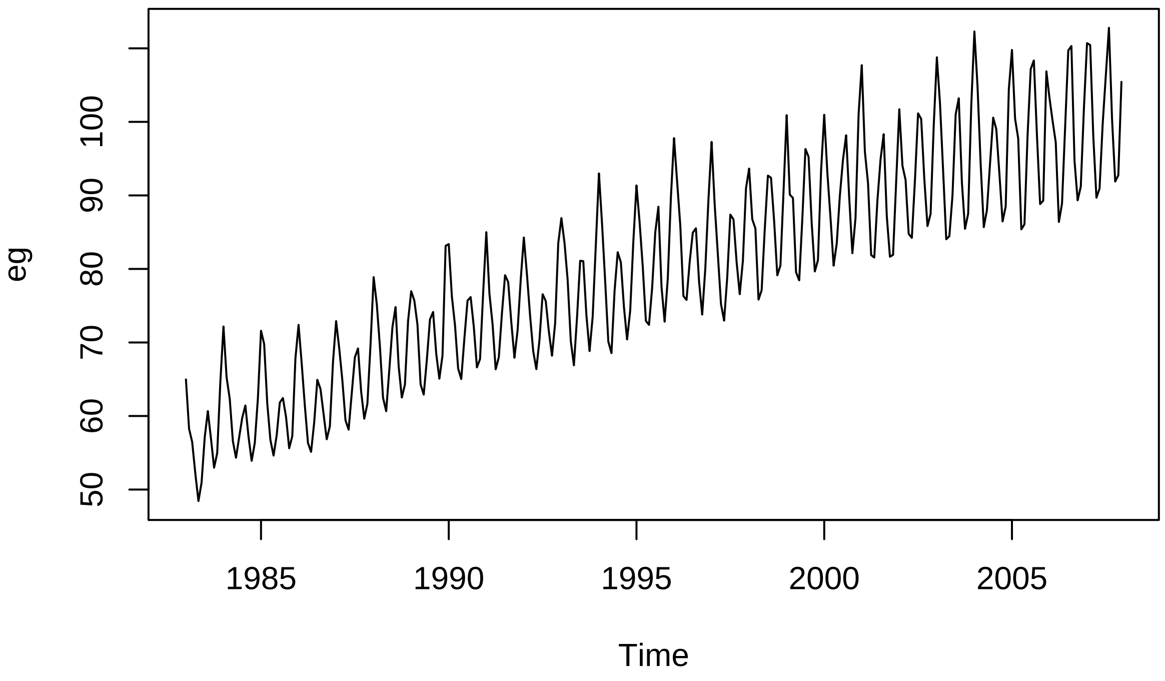
Exploration of Time Series

Erica Chen

Organizing the Dataset - Training/Testing Set

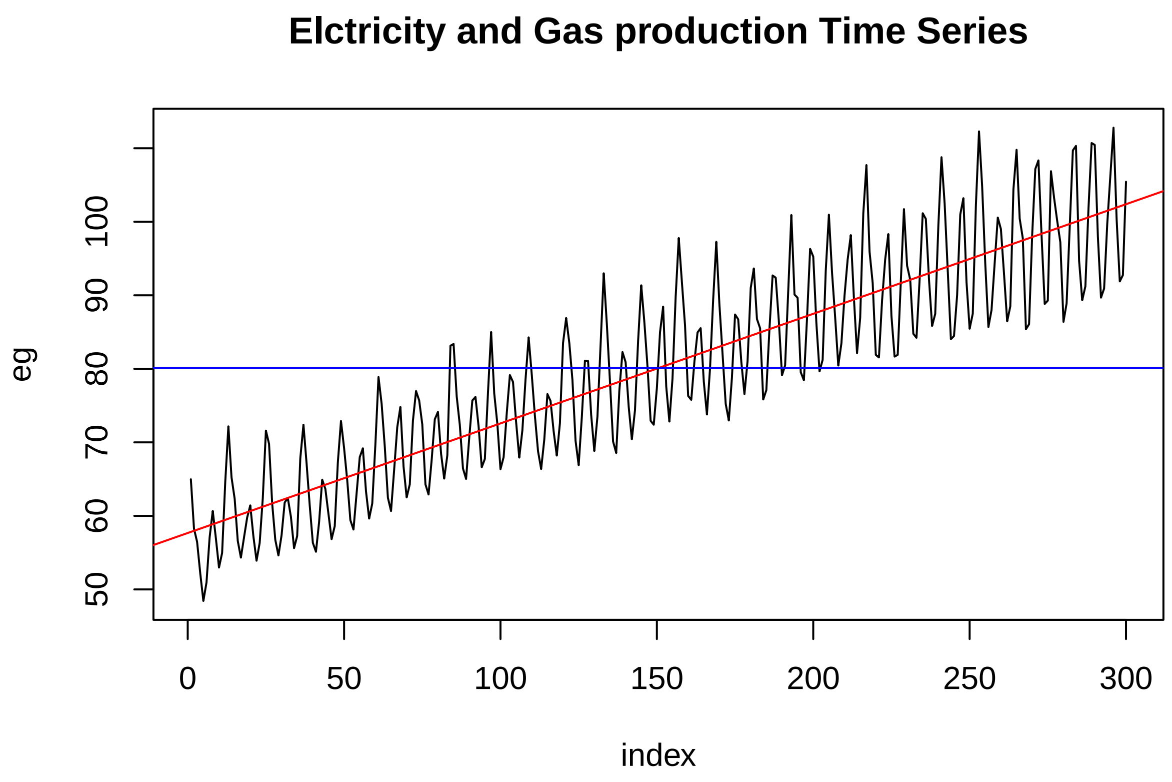
```r
#https://fred.stlouisfed.org/series/IPG2211A2N
data <- read.csv("~/Downloads/eg.csv")
```

```r
eg = ts(data[,2], start = c(1983,1), frequency = 12)
ts.plot(eg)
```

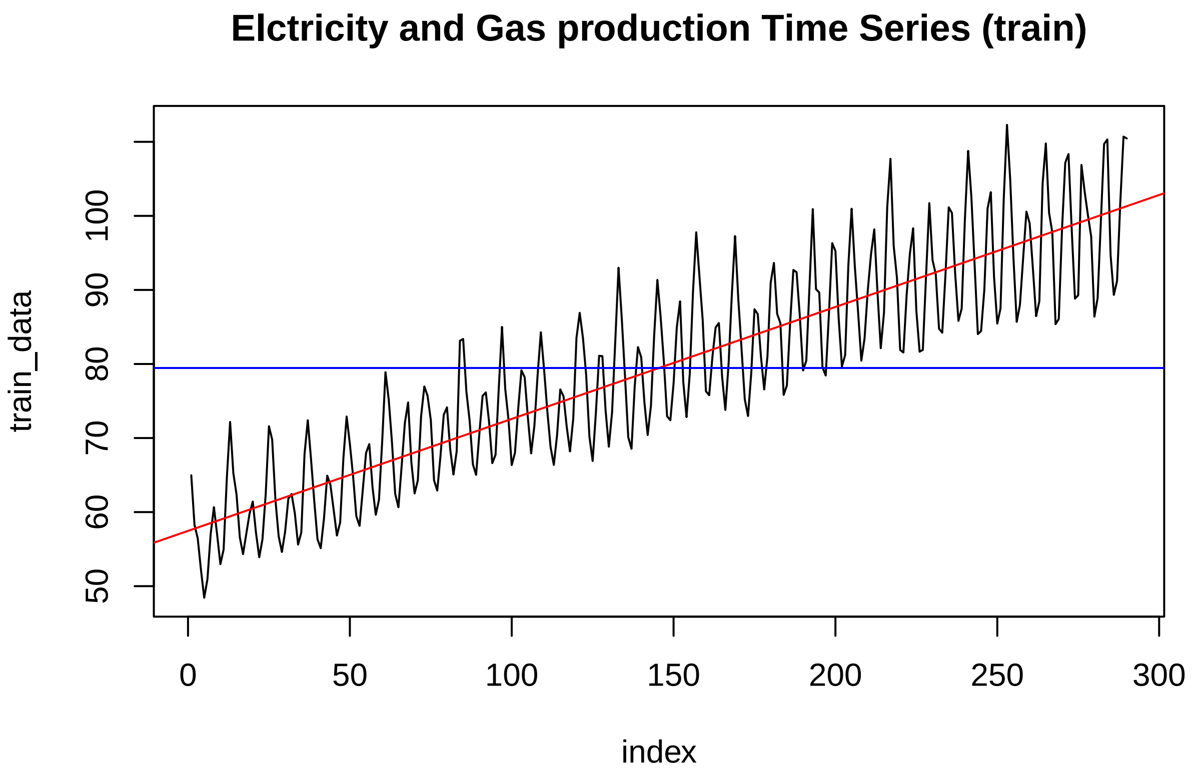


```r
# Calculate the index for the time series
index <- 1:length(eg)

# Plot the time series with the correct index
plot(index, eg, main = "Elctricity and Gas production Time Series", type = 'l', xlab = "index")

# Fit a linear trend
trend <- lm(eg ~ index)
```

```r
# Add a regression line
abline(trend, col = 'red')

# Add a mean line
abline(h = mean(eg), col = 'blue')
```



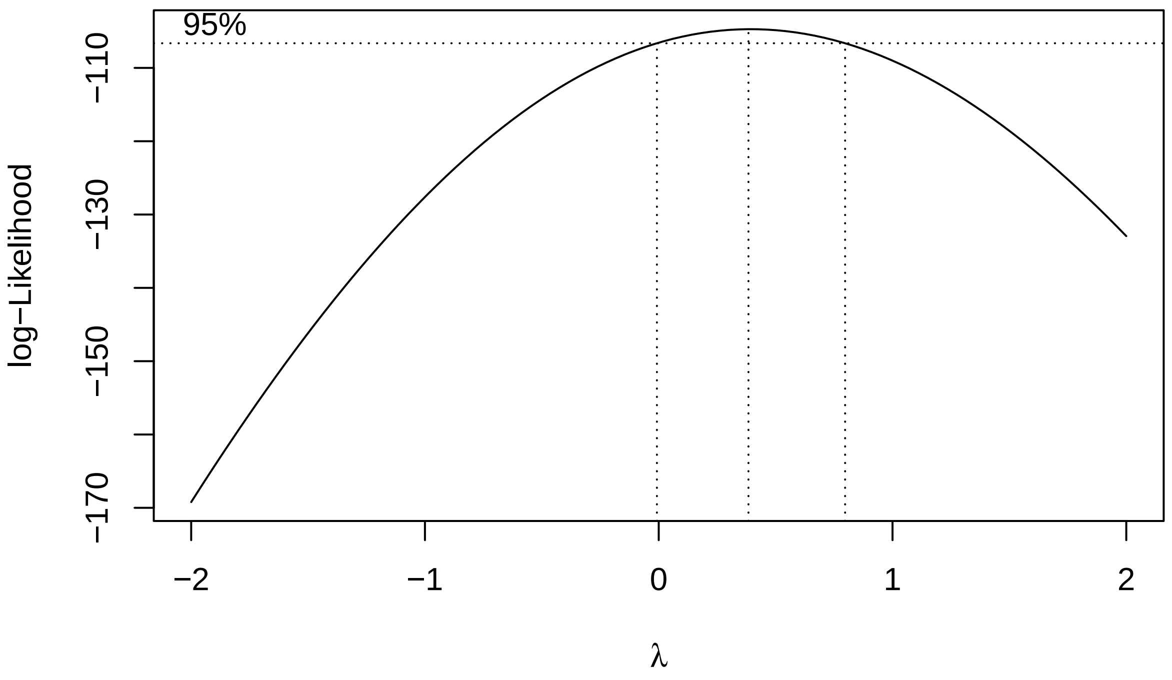
```r
train_data = eg[c(1: 290)]
test_data = eg[c(291:300)]
index_train <- 1:length(train_data)

# Plot the time series with the correct index
plot(index_train, train_data, main = "Elctricity and Gas production Time Series (train)", type = 'l', xl

# Fit a linear trend
trend <- lm(train_data ~ index_train)

# Add a regression line
abline(trend, col = 'red')

# Add a mean line
abline(h = mean(train_data), col = 'blue')
```



Box-Cox Transformation

We can see that the variance of the data also increases as time increases from the plot above, so in this section, we first transform the data and stabilize the variance.

```
library(MASS)
t = 1:length(train_data)
fit = lm(train_data ~ t)
bcTransform = boxcox(train_data ~ t,plotit = TRUE)
```



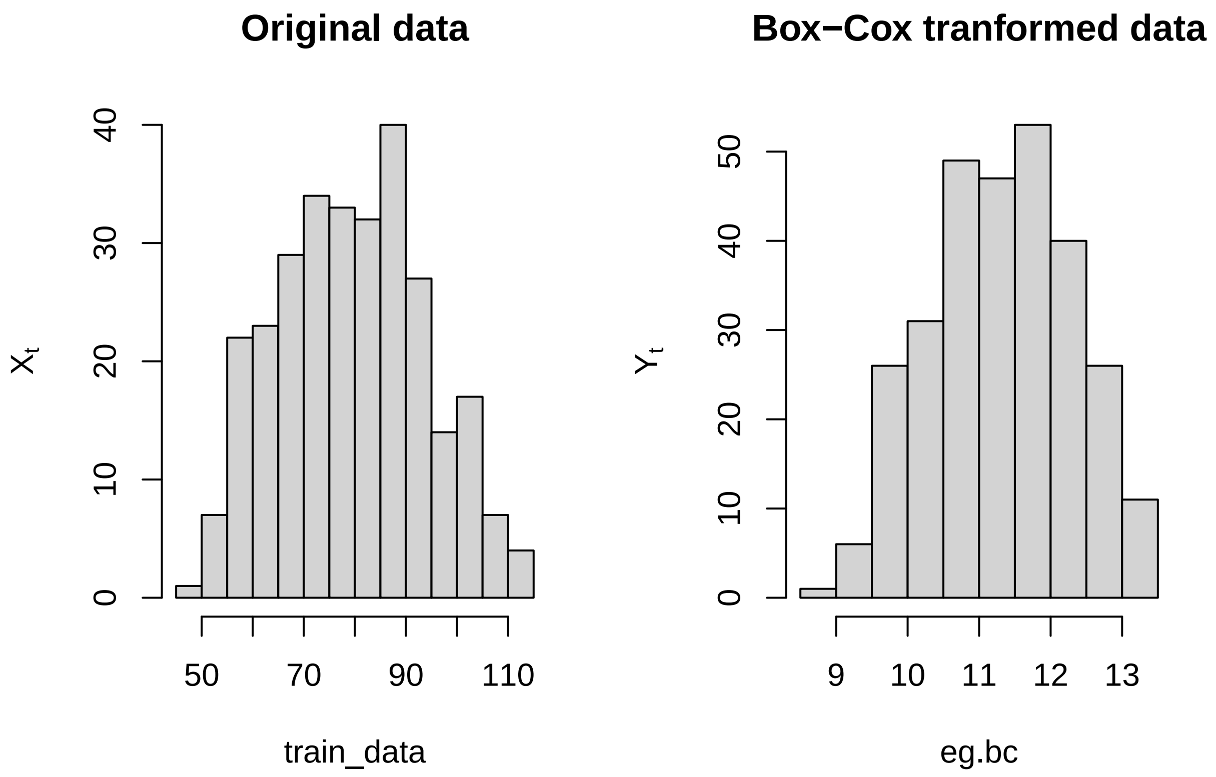
From the above plot, the best lambda to get the maximum log-likelihood is 0.383834, so we transformed the training data by the following code below:

```
lambda = bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
eg.bc = (1/lambda)*(train_data ^lambda-1)
lambda
```

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

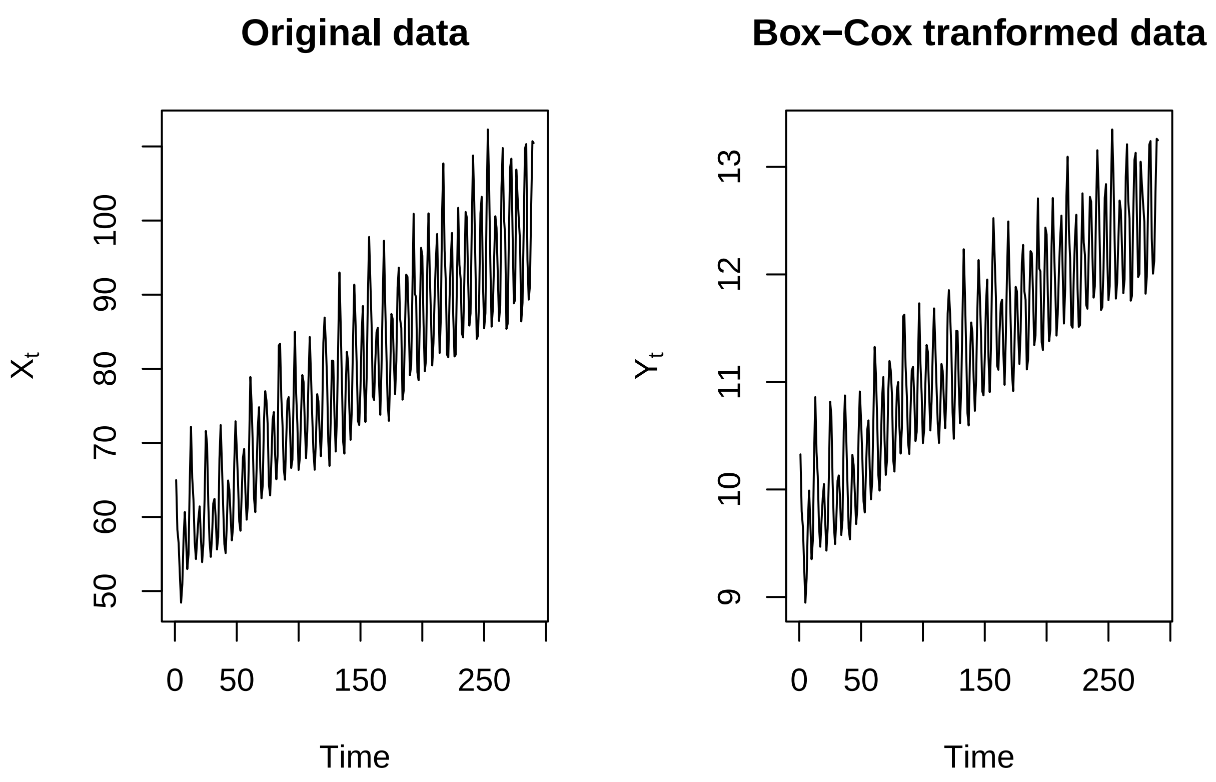
We now plot the original data vs Box-Cox transformed data:

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



Compared with the histogram of original data, the transformed data is more Gaussian. Therefore, the transformed data is more appropriate than the original.

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



From the two plots above, the transformed data has a more stable variance across time.

```
# Calculate the sample variance and plot the acf/pacf
var(train_data)
```

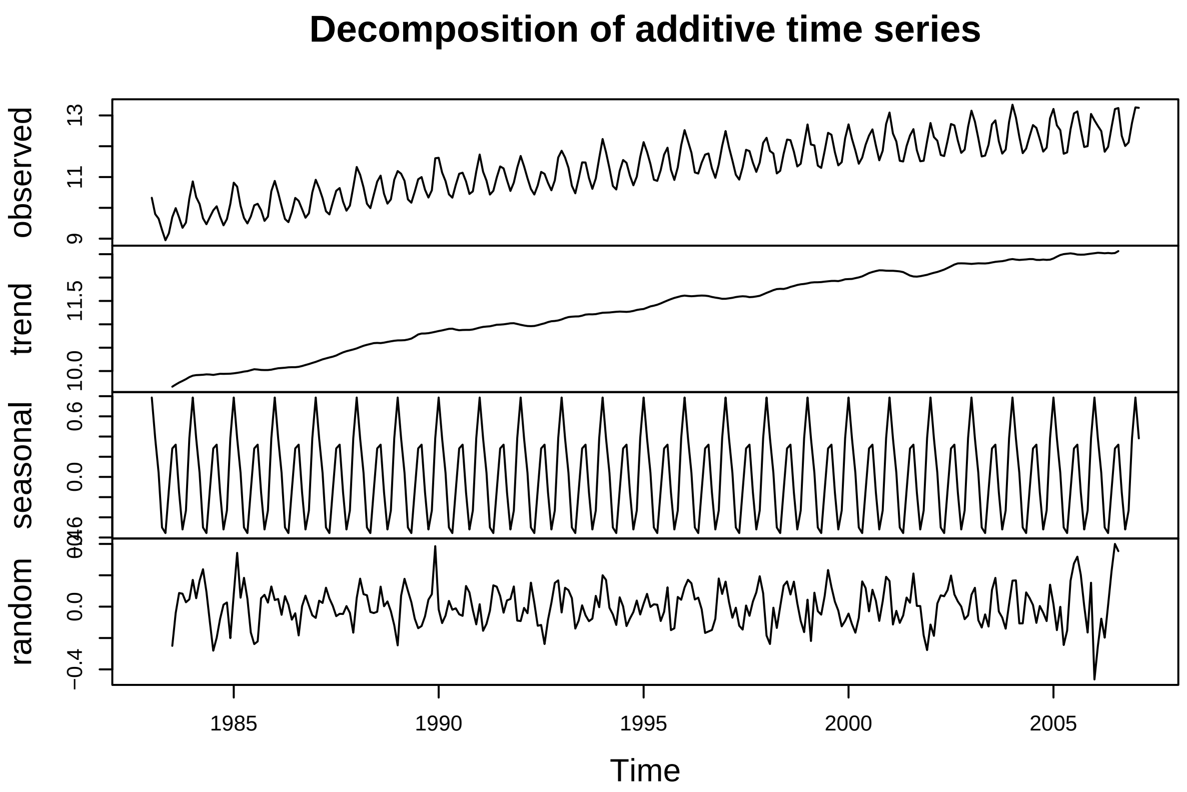
```
## [1] 205.5916
```

```
var(eg.bc)
```

```
## [1] 0.9514796
```

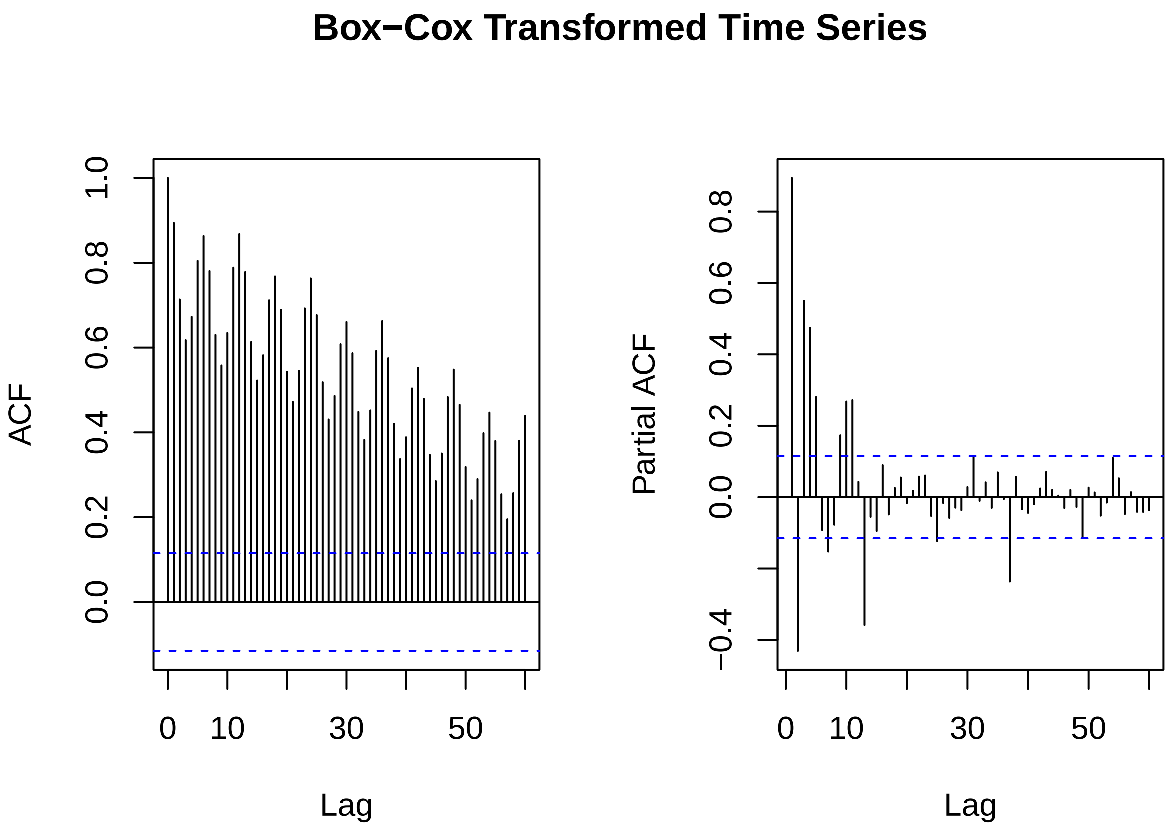
The variance of eg.bc is a lot lower than the original train_data's variance. So it was correct to transform the data with Box-Cox technique.

```
# Create a time series object
eg.bc_ts <- ts(eg.bc, start = c(1983,1), frequency = 12)

# Perform decomposition
eg.timeseriescomponents <- decompose(eg.bc_ts)

# Plot the decomposed components
plot(eg.timeseriescomponents)
```



From the plot above, we can see an upward trend between Jan 1982 to Dec 2007. Also, a clear seasonal pattern is presented in the plot, as we can see the observations regularly go up and down. Therefore, this time series is still not completely stationary.

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

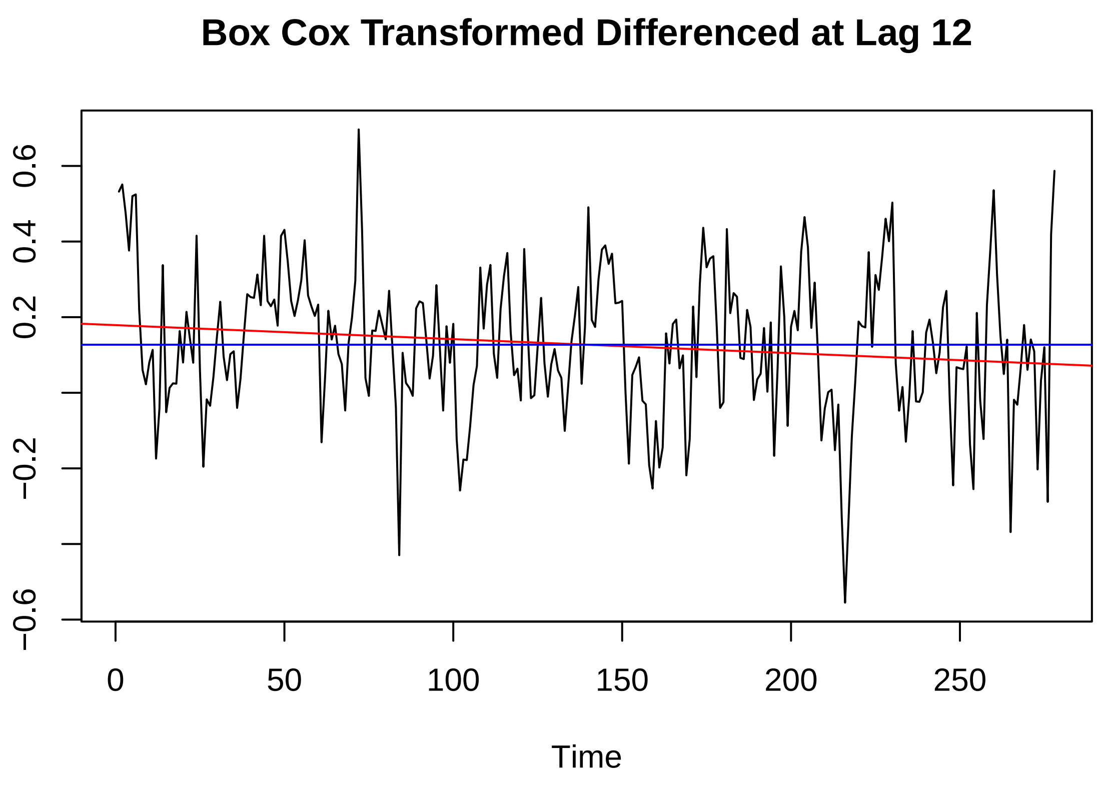


The ACF plots proves the same point too, there are definitely seasonal pattern which we need to fix.

Differencing - Removing Seasonality and Trend

ACF plot shows seasonality where there is a fluctuation every 12 lags. So difference at lag 12

```r
#removing seasonal
eg.lag12 <- diff(eg.bc, lag=12)
plot.ts(eg.lag12, main="Box Cox Transformed Differenced at Lag 12",ylab="")
fit_11 <- lm(eg.lag12 ~ as.numeric(1:length(eg.lag12)))
abline(fit_11, col="red")
abline(h=mean(eg.lag12), col="blue")
```



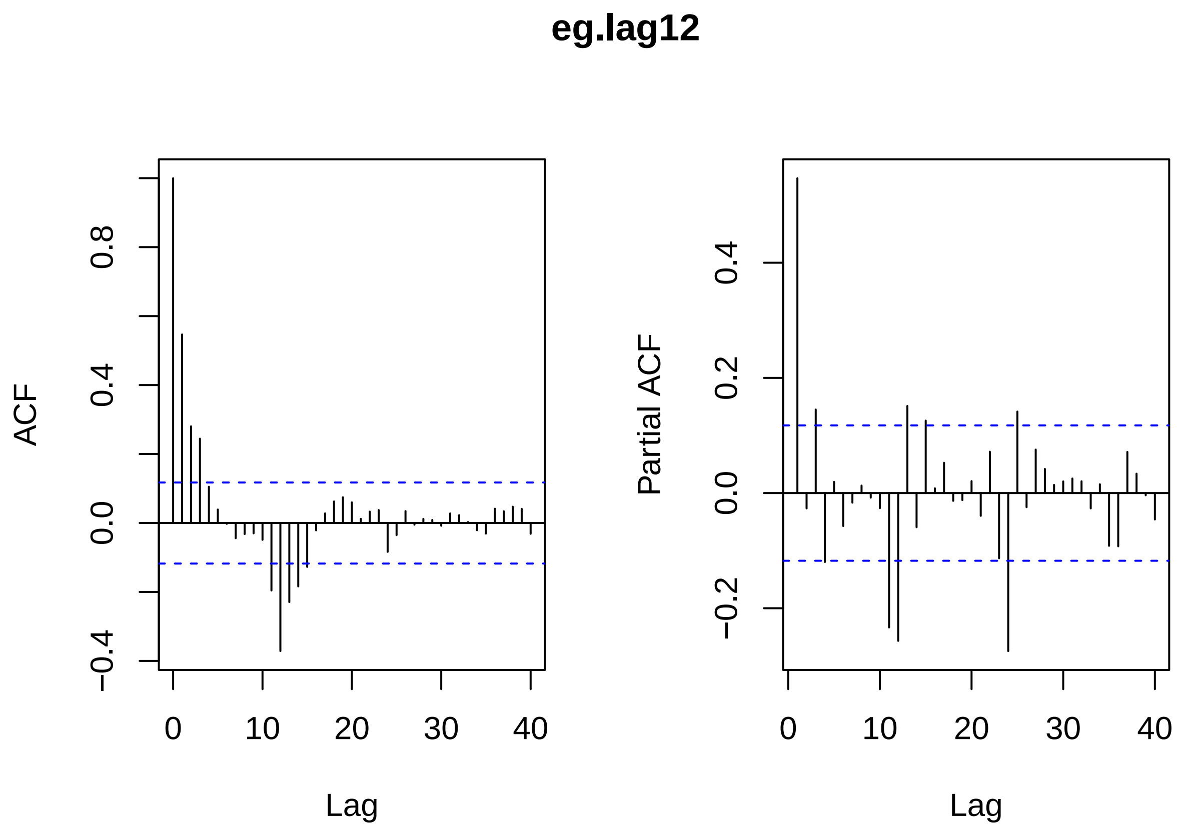
```r
#variance
var(eg.lag12)
```

```
## [1] 0.03608923
```

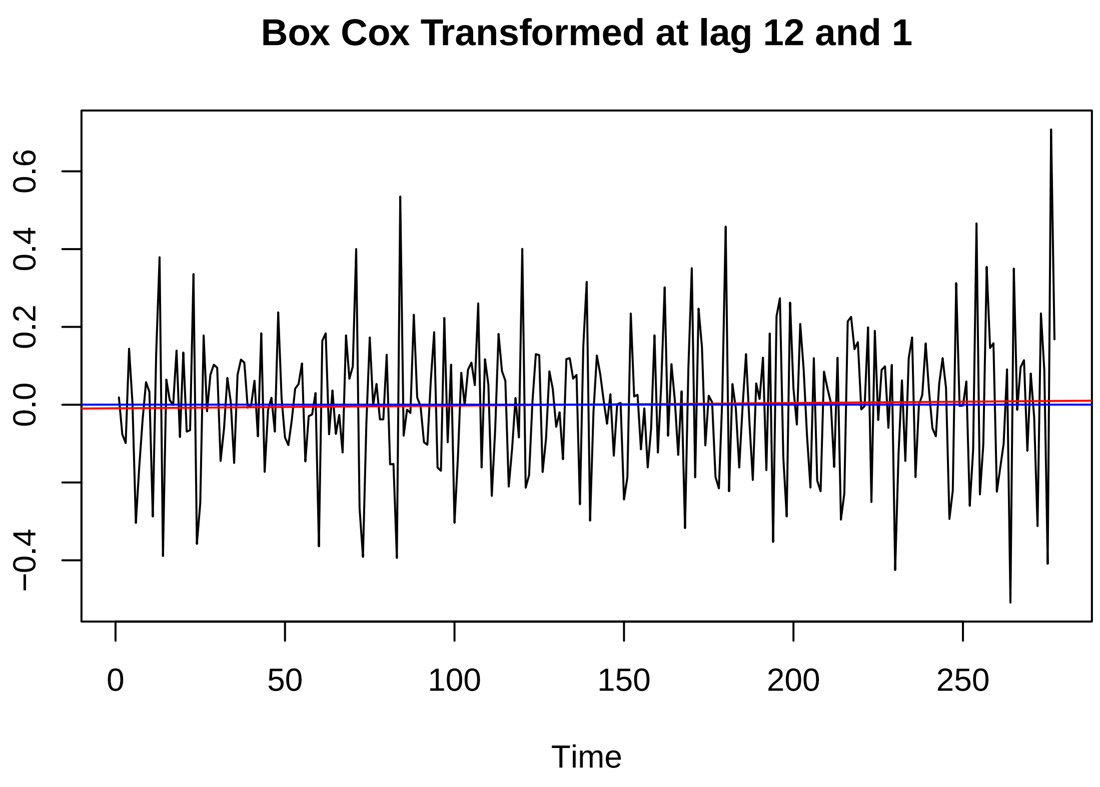
```r
var(eg.bc)
```

```
## [1] 0.9514796
```

```r
#PACF and ACF of eg.lag12
op = par(mfrow = c(1,2))
acf(eg.lag12,lag.max = 40,main = "")
pacf(eg.lag12,lag.max = 40,main = "")
title("eg.lag12",line = -1, outer=TRUE)
```



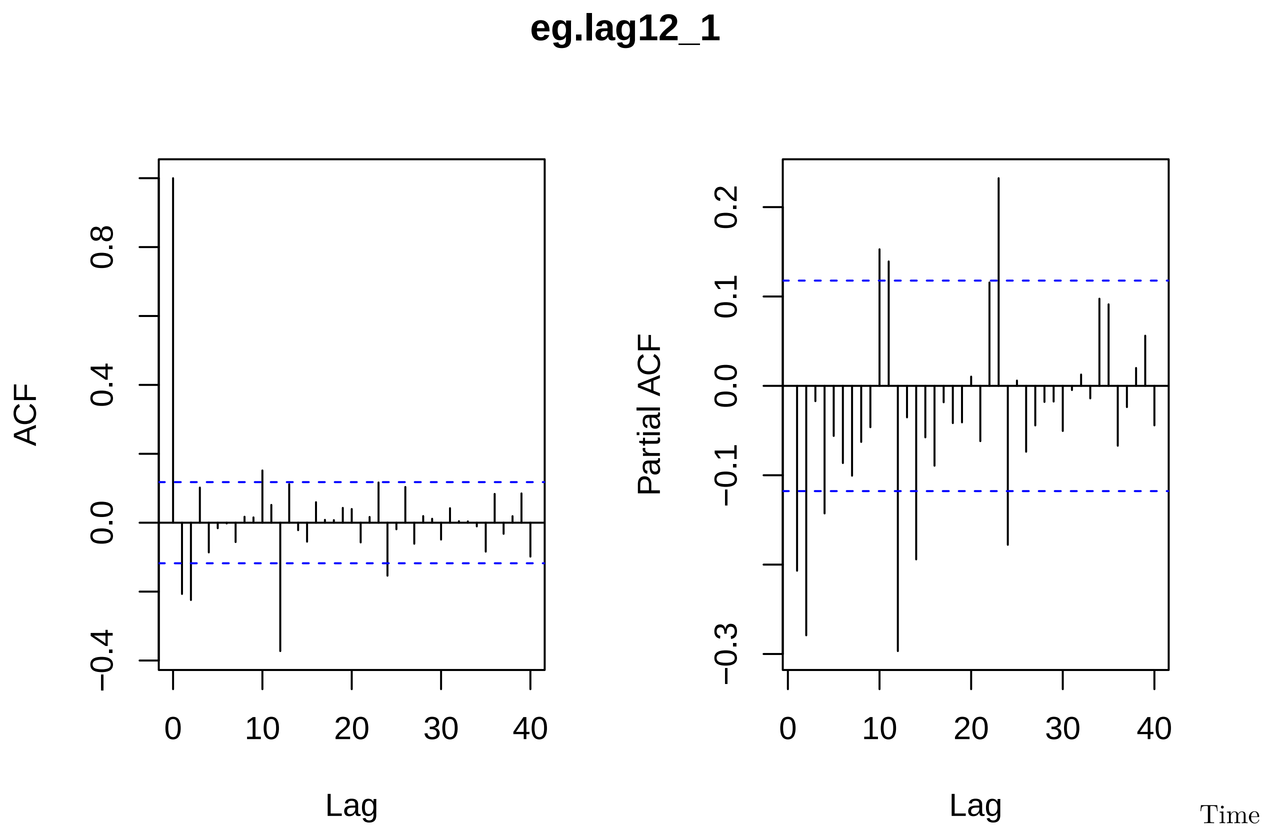
```
#removing seasonal and trend
eg.lag12_1 <- diff(eg.lag12, lag=1)
plot.ts(eg.lag12_1, main="Box Cox Transformed at lag 12 and 1",ylab="")
fit_12 <- lm(eg.lag12_1 ~ as.numeric(1:length(eg.lag12_1)))
abline(fit_12, col="red")
abline(h=mean(eg.lag12_1), col="blue")
```



```
#variance
var(eg.lag12_1)
```

```
## [1] 0.03146527
```

```
#PACF and ACF of eg.lag12_1
op = par(mfrow = c(1,2))
acf(eg.lag12_1,lag.max = 40,main = "")
pacf(eg.lag12_1,lag.max = 40,main = "")
title("eg.lag12_1",line = -1, outer=TRUE)
```



Series plot of Box-Cox transformed differenced at 12 & lag 1 has no trend and no seasonality so it is stationary. ACF plot shows acf quickly decaying after lag 0 so it also shows that it is stationary. Variances decreased so there's no signs of overdifferencing.

p = 1 or 2 CF shows a significant spike at lag 12, suggesting an AR(1) term for the seasonal component (P = 1) d = 1 (because we differenced at lag 1) q = 1 or 2 (acf lags at 1/2)

P = 1 (pacfs outside around lag 12) D = 1 (because we differenced at lag 12) Q = 1 or 2 (because of lags 12/24)

SARIMA (1,1,1) (1,1,1)s = 12 and all other combinations

Find suitable ARMA models using maximum likelihood estimation

```
library(qpcR)
```

```
## Loading required package: minpack.lm
```

```
## Loading required package: rgl
```

```
## Loading required package: robustbase
```

```
## Loading required package: Matrix
```

```
# Calculate AICc for ARMA models with p and q running from 0 to 5
aiccs <- matrix(NA, nr = 6, nc = 6)
dimnames(aiccs) = list(p=0:5, q=0:5)
for(p in 0:5) {
for(q in 0:5) {
aiccs[p+1,q+1] = AICc(arima(eg.bc, order = c(p,1,q), method="ML")) }
}
```

```
## Warning in log(s2): NaNs produced

## Warning in log(s2): NaNs produced

## Warning in log(s2): NaNs produced

## Warning in arima(eg.bc, order = c(p, 1, q), method = "ML"): possible
## convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced

## Warning in log(s2): NaNs produced

## Warning in log(s2): NaNs produced

## Warning in log(s2): NaNs produced

## Warning in log(s2): NaNs produced

## Warning in arima(eg.bc, order = c(p, 1, q), method = "ML"): possible
## convergence problem: optim gave code = 1
```

```
aiccs
```

```
##    q
## p            0           1           2          3         4          5
##   0 334.316764 239.0598723  189.262020  138.18896  117.5933   90.02938
##   1 292.163025 240.7986641  160.881281  136.98659  119.2256   58.47480
##   2 130.589923  10.4962661    9.437049  -99.02082 -111.5421 -113.63594
##   3  38.873770   7.3700235    8.692063   13.56704 -118.6485 -111.98613
##   4  -4.962172  -3.0331269 -100.149436  -95.50576 -121.5467 -117.12374
##   5  -3.070242  -0.9119751   -9.227147 -117.78526 -118.0324 -131.87993
```

```
(aiccs==min(aiccs))
```

```
##    q
## p       0     1     2     3     4     5
##   0 FALSE FALSE FALSE FALSE FALSE FALSE
##   1 FALSE FALSE FALSE FALSE FALSE FALSE
##   2 FALSE FALSE FALSE FALSE FALSE FALSE
##   3 FALSE FALSE FALSE FALSE FALSE FALSE
##   4 FALSE FALSE FALSE FALSE FALSE FALSE
##   5 FALSE FALSE FALSE FALSE FALSE  TRUE
```

From the loop to find the best p and q, we could get the optimal ARMA model to be ARMA(5,5).

##Model Selection

```
fit1 = arima(eg.bc, order=c(1,1,1), seasonal = list(order = c(1,1,1), period = 12)
, method="ML")
fit1
```

```
##
## Call:
## arima(x = eg.bc, order = c(1, 1, 1), seasonal = list(order = c(1, 1, 1), period = 12),
##     method = "ML")
##
## Coefficients:
##          ar1      ma1    sar1     sma1
##       0.5401  -0.9576  0.0369  -0.7841
## s.e.  0.0579   0.0207  0.0795   0.0534
##
## sigma^2 estimated as 0.01658:  log likelihood = 168.03,  aic = -326.06
```

```
fit2 = arima(eg.bc, order=c(2,1,1), seasonal = list(order = c(1,1,1), period = 12)
, method="ML")
fit2
```

```
##
## Call:
## arima(x = eg.bc, order = c(2, 1, 1), seasonal = list(order = c(1, 1, 1), period = 12),
##     method = "ML")
##
## Coefficients:
##          ar1      ar2      ma1    sar1     sma1
##       0.5587  -0.0448  -0.9534  0.0375  -0.7877
## s.e.  0.0643   0.0638   0.0236  0.0795   0.0536
##
## sigma^2 estimated as 0.01655:  log likelihood = 168.27,  aic = -324.55
```

```
fit3 = arima(eg.bc, order=c(1,1,2), seasonal = list(order = c(1,1,1), period = 12)
, method="ML")
fit3
```

```
##
## Call:
## arima(x = eg.bc, order = c(1, 1, 2), seasonal = list(order = c(1, 1, 1), period = 12),
##     method = "ML")
##
## Coefficients:
##          ar1      ma1      ma2    sar1     sma1
##       0.4250  -0.8044  -0.1391  0.0375  -0.7884
## s.e.  0.1506   0.1730   0.1533  0.0796   0.0536
##
## sigma^2 estimated as 0.01653:  log likelihood = 168.45,  aic = -324.9
```

```
fit4 = arima(eg.bc, order=c(1,1,1), seasonal = list(order = c(1,1,2), period = 12)
, method="ML")
fit4
```

```
##
## Call:
## arima(x = eg.bc, order = c(1, 1, 1), seasonal = list(order = c(1, 1, 2), period = 12),
##     method = "ML")
##
## Coefficients:
##          ar1      ma1      sar1     sma1     sma2
##       0.5474  -0.9584  -0.5517  -0.1526  -0.5084
## s.e.  0.0573   0.0200   0.2767   0.2611   0.1955
##
## sigma^2 estimated as 0.01643:  log likelihood = 169.18,  aic = -326.35
```

```
fit5 = arima(eg.bc, order=c(1,1,2), seasonal = list(order = c(1,1,2), period = 12)
, method="ML")
fit5
```

```
##
## Call:
## arima(x = eg.bc, order = c(1, 1, 2), seasonal = list(order = c(1, 1, 2), period = 12),
##     method = "ML")
##
## Coefficients:
##          ar1      ma1      ma2     sar1     sma1     sma2
##       0.4328  -0.8041  -0.1407  -0.5733  -0.1350  -0.5280
## s.e.  0.1433   0.1650   0.1470   0.2782   0.2615   0.1967
##
## sigma^2 estimated as 0.01637:  log likelihood = 169.64,  aic = -325.28
```

```
fit6 = arima(eg.bc, order=c(2,1,2), seasonal = list(order = c(1,1,1), period = 12)
, method="ML")
fit6
```

```
##
## Call:
## arima(x = eg.bc, order = c(2, 1, 2), seasonal = list(order = c(1, 1, 1), period = 12),
##     method = "ML")
##
## Coefficients:
##           ar1     ar2      ma1      ma2    sar1     sma1
##       -0.1068  0.2820  -0.2576  -0.6636  0.0432  -0.7827
## s.e.   0.1821  0.1251   0.1656   0.1606  0.0808   0.0553
##
## sigma^2 estimated as 0.01638:  log likelihood = 169.88,  aic = -325.76
```

```
fit7 = arima(eg.bc, order=c(2,1,1), seasonal = list(order = c(1,1,2), period = 12)
, method="ML")
fit7
```

```
##
## Call:
## arima(x = eg.bc, order = c(2, 1, 1), seasonal = list(order = c(1, 1, 2), period = 12),
##     method = "ML")
##
## Coefficients:
##          ar1      ar2      ma1      sar1     sma1     sma2
##       0.5679  -0.0477  -0.9543  -0.5676  -0.1400  -0.5233
## s.e.  0.0642   0.0637   0.0227   0.2772   0.2605   0.1961
##
## sigma^2 estimated as 0.01639:  log likelihood = 169.46,  aic = -324.91
```

```
fit8 = arima(eg.bc, order=c(2,1,2), seasonal = list(order = c(1,1,2), period = 12)
, method="ML")
fit8
```

```
##
## Call:
## arima(x = eg.bc, order = c(2, 1, 2), seasonal = list(order = c(1, 1, 2), period = 12),
##     method = "ML")
##
## Coefficients:
##           ar1     ar2      ma1      ma2     sar1     sma1     sma2
##       -0.0998  0.2881  -0.2587  -0.6647  -0.5446  -0.1561  -0.5018
## s.e.   0.1906  0.1304   0.1740   0.1686   0.2823   0.2669   0.1986
##
## sigma^2 estimated as 0.01623:  log likelihood = 171,  aic = -326
```

```
fit9 = arima(eg.bc, order = c(5,1,5), method="ML")
```

```
## Warning in arima(eg.bc, order = c(5, 1, 5), method = "ML"): possible
## convergence problem: optim gave code = 1
```

```
fit9
```

```
##
## Call:
## arima(x = eg.bc, order = c(5, 1, 5), method = "ML")
##
## Coefficients:
##           ar1     ar2      ar3      ar4     ar5      ma1      ma2     ma3
##       -0.1309  0.2490  -0.6659  -0.4640  0.5832  -0.0334  -0.5371  0.6571
## s.e.   0.0604  0.0506   0.0210   0.0497  0.0611   0.0439   0.0364  0.0231
##          ma4      ma5
##       0.2365  -0.8886
## s.e.  0.0440   0.0431
##
## sigma^2 estimated as 0.03221:  log likelihood = 77.33,  aic = -132.67
```

Lowest AICc value

```
AICc(fit1)
```

```
## [1] -325.917
```

```
AICc(fit2)
```

```
## [1] -324.3383
```

```
AICc(fit3)
```

```
## [1] -324.6903
```

```
AICc(fit4) # lowest AICc value, -326.1422
```

```
## [1] -326.1422
```

```
AICc(fit5)
```

```
## [1] -324.9877
```

```
AICc(fit6)
```

```
## [1] -325.4618
```

```
AICc(fit7)
```

```
## [1] -324.6168
```

```
AICc(fit8)
```

```
## [1] -325.6055
```

```
AICc(fit9)
```

```
## [1] -131.8799
```

Algebraic Form of Fit 4

```
fit4 = arima(eg.bc, order=c(1,1,1), seasonal = list(order = c(1,1,2), period = 12)
, method="ML")
fit4
```

```
##
## Call:
## arima(x = eg.bc, order = c(1, 1, 1), seasonal = list(order = c(1, 1, 2), period = 12),
##     method = "ML")
##
## Coefficients:
##          ar1      ma1      sar1     sma1     sma2
##       0.5474  -0.9584  -0.5517  -0.1526  -0.5084
## s.e.  0.0573   0.0200   0.2767   0.2611   0.1955
##
## sigma^2 estimated as 0.01643:  log likelihood = 169.18,  aic = -326.35
```

$$(1-0.5474B)(1+0.5517B^{12})(1-B)(1-B^{12})Y_t = (1-0.9584B)(1-0.1525B^{12})(1-0.5084B^{24})Z_t$$

SMA1 is not significant from the above output. Therefore, we fix SMA1 to 0 and refit the model.

```
fit4a = arima(eg.bc, order=c(1,1,1), seasonal = list(order = c(1,1,2)
, period = 12) ,fixed = c(NA, NA, NA, 0, NA), method="ML")
fit4a
```

```
##
## Call:
## arima(x = eg.bc, order = c(1, 1, 1), seasonal = list(order = c(1, 1, 2), period = 12),
##     fixed = c(NA, NA, NA, 0, NA), method = "ML")
##
## Coefficients:
##          ar1      ma1      sar1  sma1     sma2
##       0.5452  -0.9585  -0.7148     0  -0.6133
## s.e.  0.0574   0.0202   0.0589     0   0.0672
##
## sigma^2 estimated as 0.01646:  log likelihood = 169.01,  aic = -328.02
```

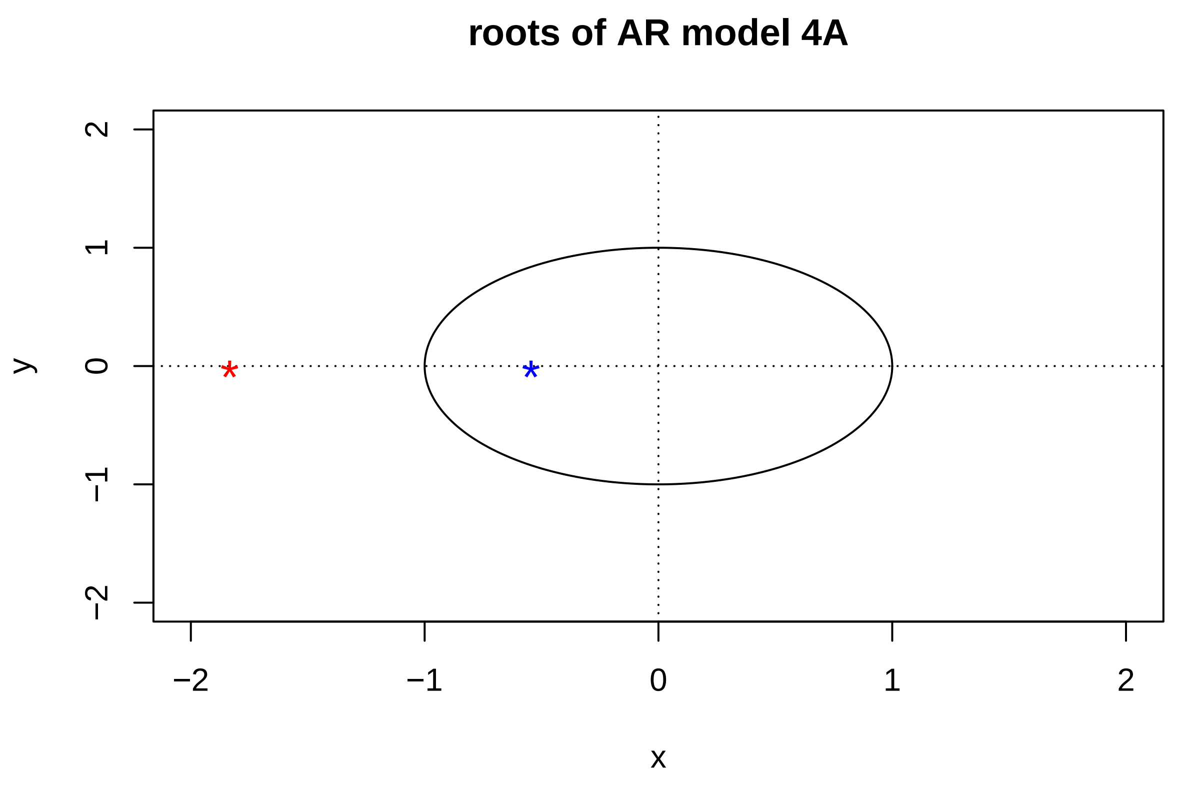
$(1-0.5452B)(1+0.7148B^{12})(1-B)(1-B^{12})Y_t = (1-0.9585B)(1-0.6133B^{24})Z_t$

```
AICc(fit4a)
```

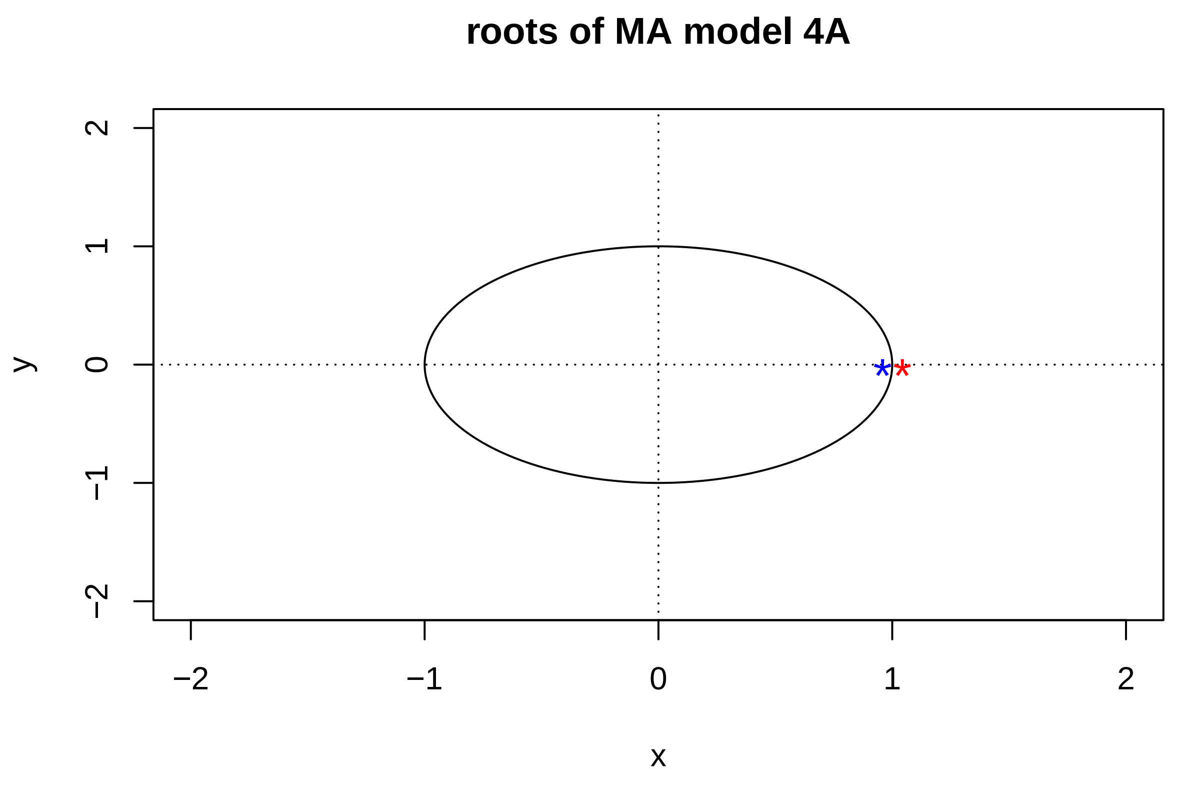
```
## [1] -327.8048
```

Now fit4a has the lowest AICc value out of all the models

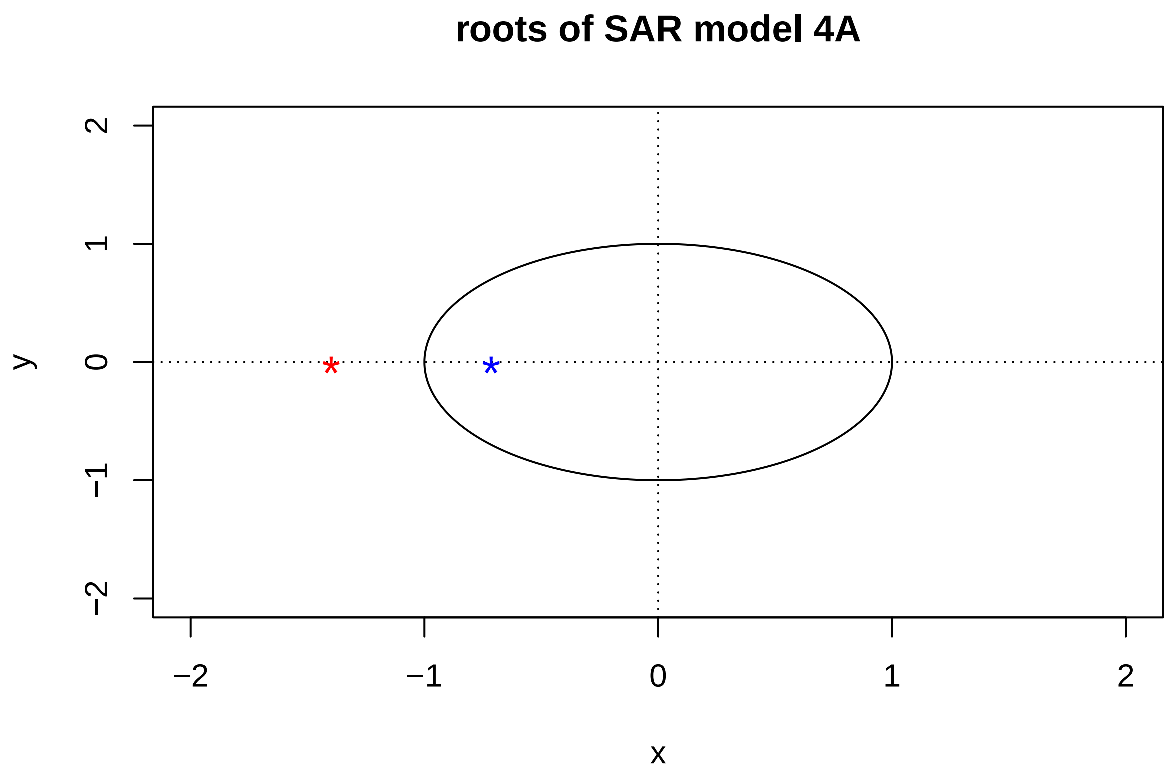
Check Model4A stationarity/invertibility:



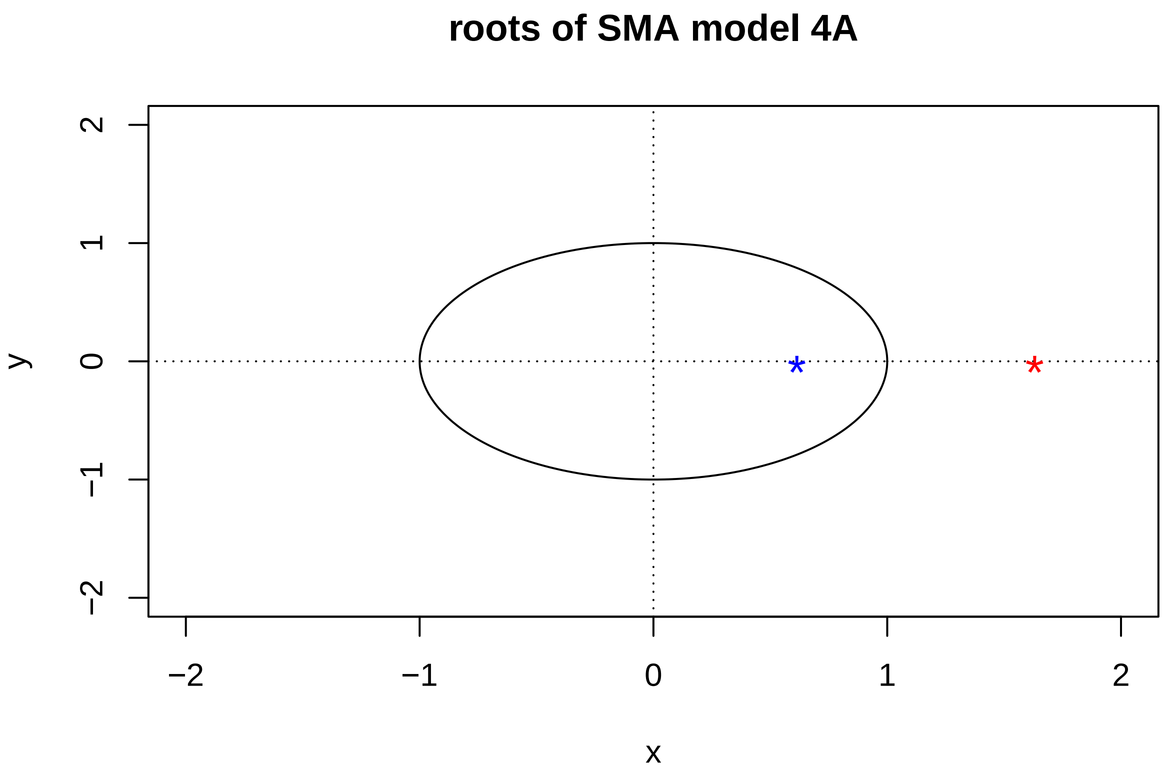
```
plot.roots(NULL,polyroot(c(1, -0.9585)), main="roots of MA model 4A"
)
```



```
plot.roots(NULL,polyroot(c(1, 0.7148)), main="roots of SAR model 4A"
)
```

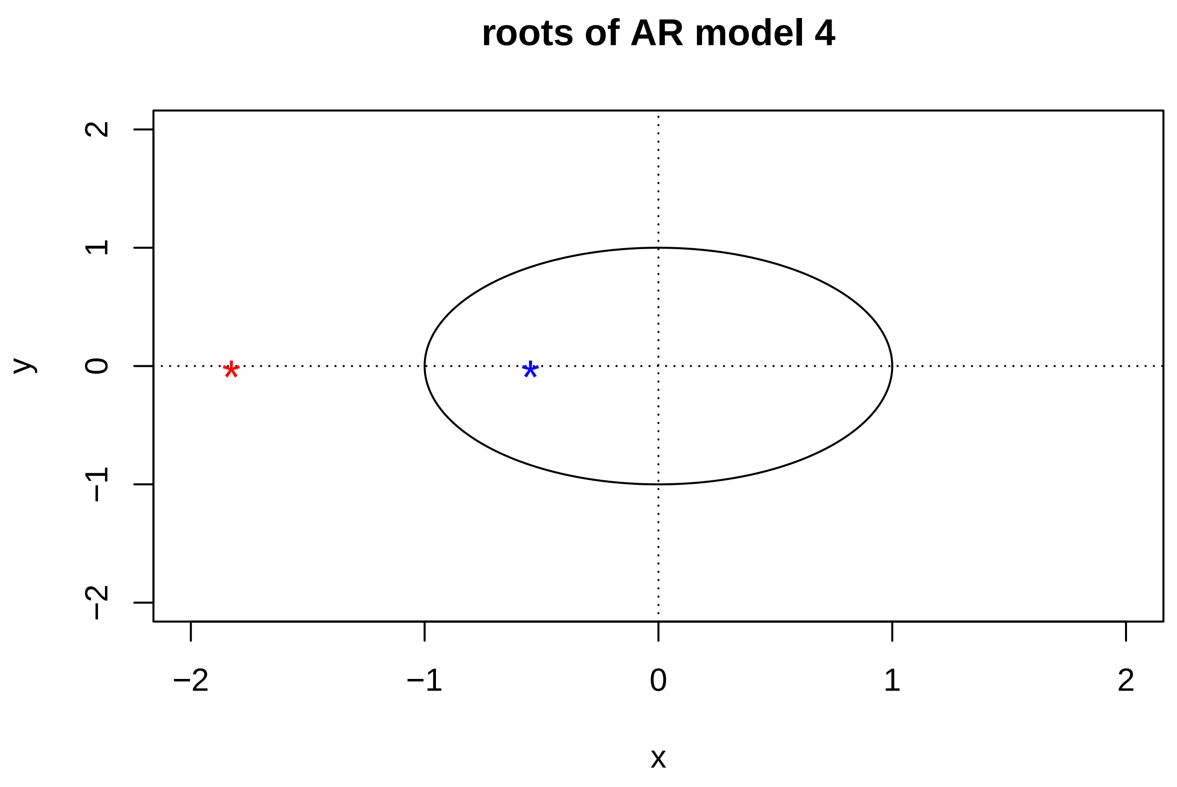


```
plot.roots(NULL,polyroot(c(1, -0.6133)), main="roots of SMA model 4A"
)
```

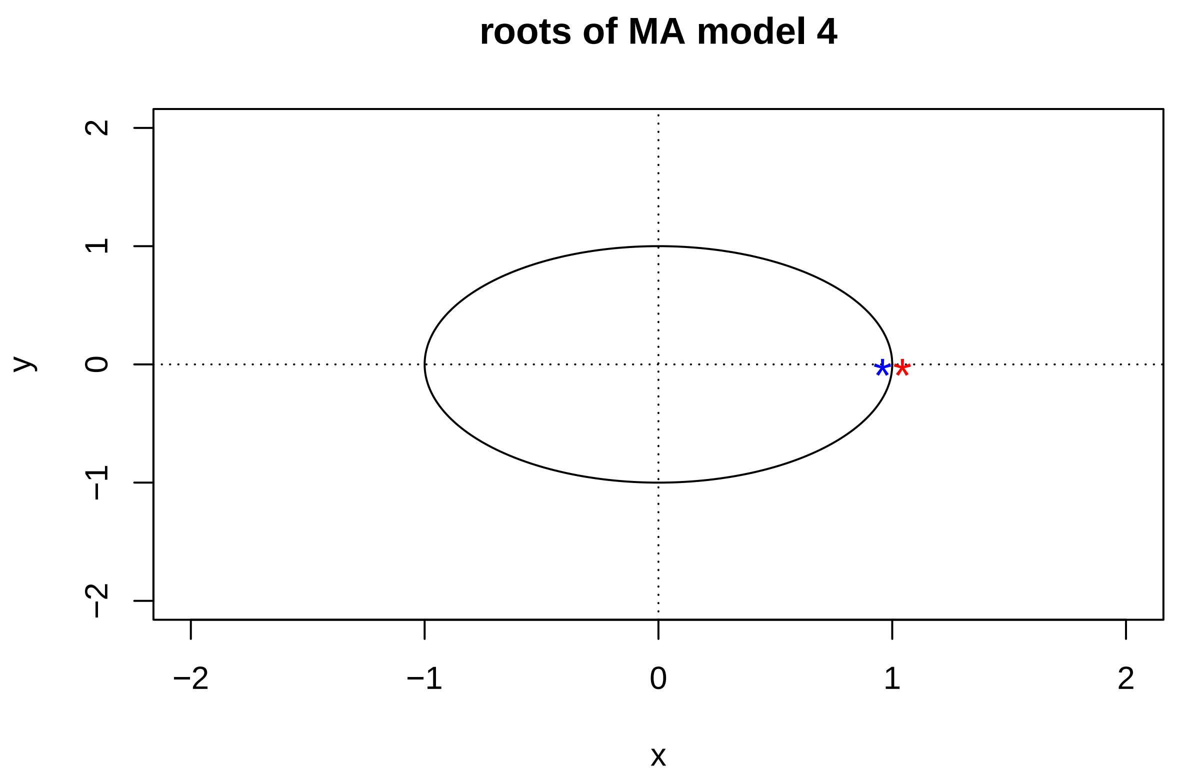


Check Model4 stationarity/invertibility:

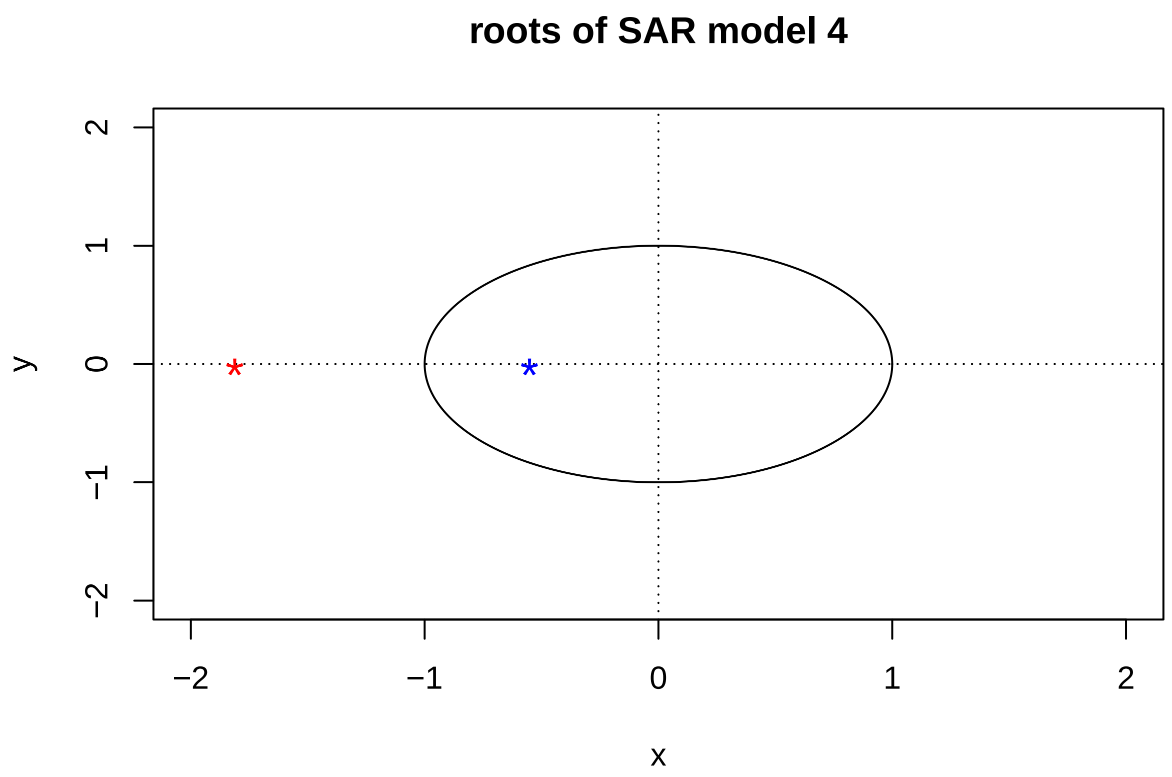
```
plot.roots(NULL,polyroot(c(1, 0.5474)), main="roots of AR model 4"
)
```



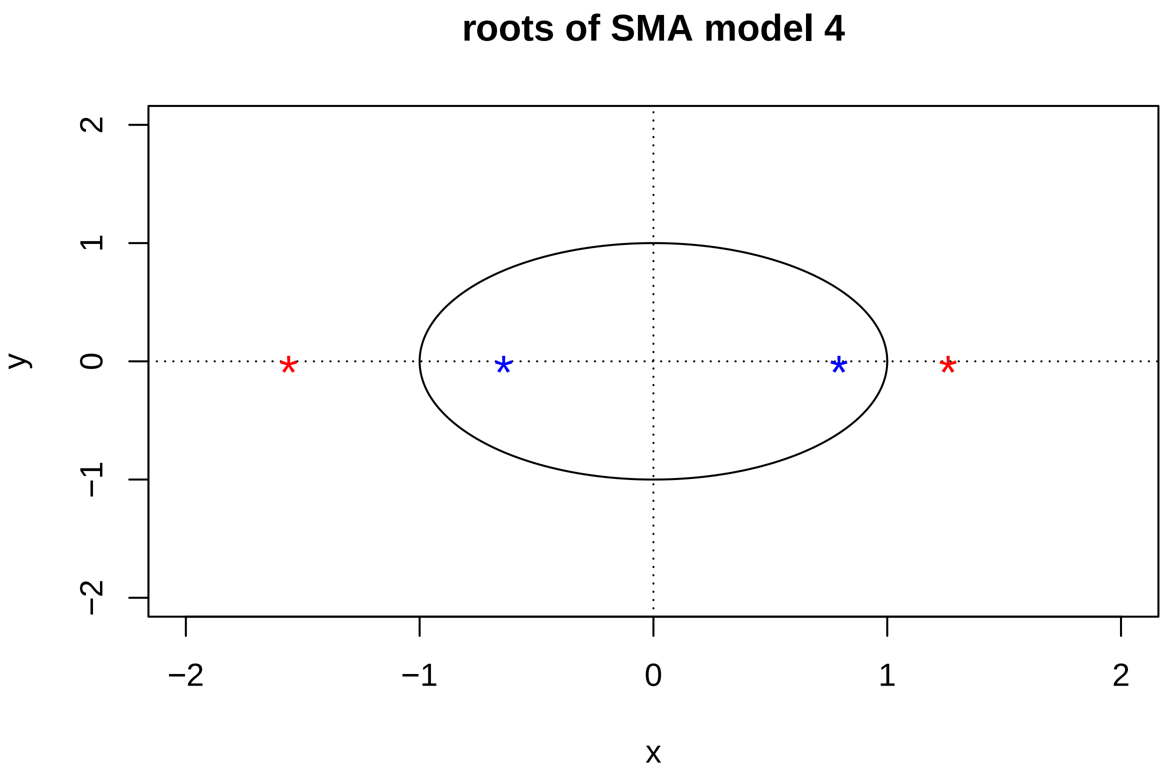
```
plot.roots(NULL,polyroot(c(1, -0.9584)), main="roots of MA model 4"
)
```



```
plot.roots(NULL,polyroot(c(1, 0.5517)), main="roots of SAR model 4"
)
```



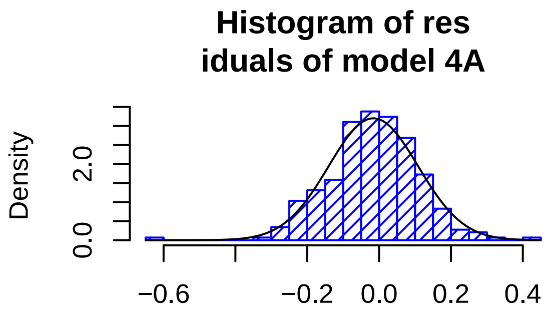
```
plot.roots(NULL,polyroot(c(1, -0.1526, -0.5084)), main="roots of SMA model 4"
)
```

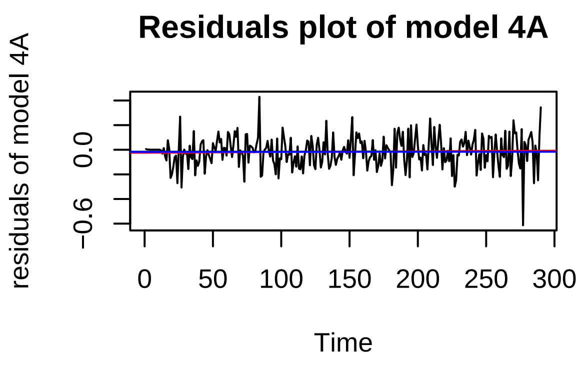


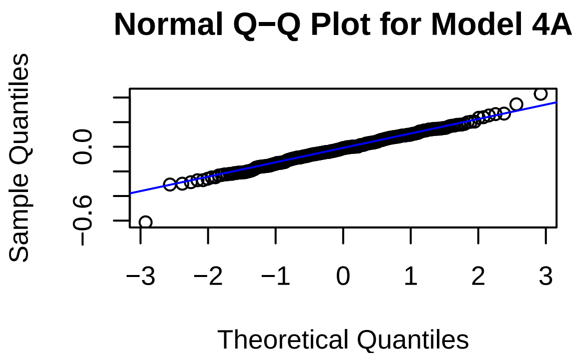
For both plots, blue dots are inside the unit circle, meaning that the roots(red) are outside the unit circle. Therefore, both model pass the check the model stationarity/invertibility.

Diagnostic Tests

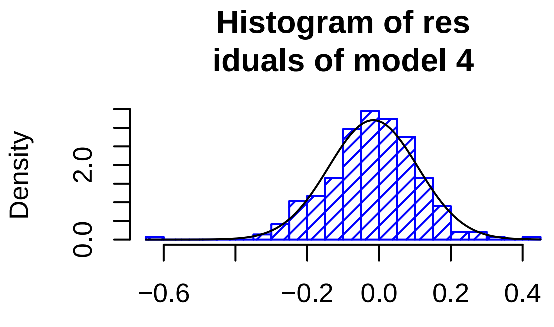
```
## Diagnostic Tests for Fit4A
res4 = residuals(fit4a)
par(mfrow=c(2,2))
hist(res4,density=20,breaks=20, col="blue", xlab="", prob=TRUE,main="Histogram of res
iduals of model 4A")
m <- mean(res4)
std <- sqrt(var(res4))
curve( dnorm(x,m,std), add=TRUE )
plot.ts(res4,ylab= "residuals of model 4A",main="Residuals plot of model 4A")
fitt <- lm(res4 ~ as.numeric(1:length(res4)))
abline(fitt, col="red")
abline(h=mean(res4), col="blue")
qqnorm(res4,main= "Normal Q-Q Plot for Model 4A")
qqline(res4,col="blue")


## Diagnostic Tests for Fit4

res4. = residuals(fit4)
par(mfrow=c(2,2))
```

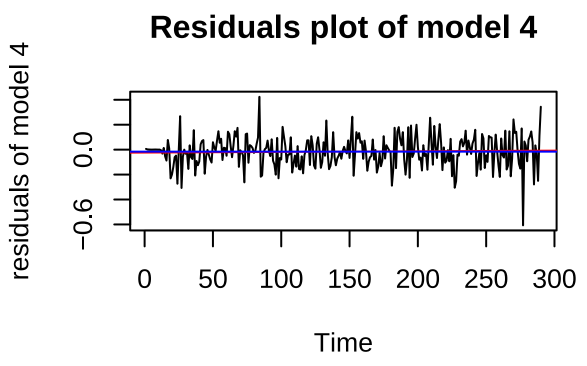


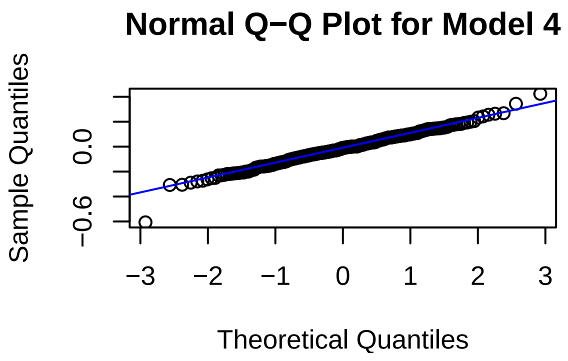




```
hist(res4.,density=20,breaks=20, col="blue", xlab="", prob=TRUE,main="Histogram of res
iduals of model 4")
m <- mean(res4.)
std <- sqrt(var(res4.))
curve( dnorm(x,m,std), add=TRUE )
plot.ts(res4.,ylab= "residuals of model 4",main="Residuals plot of model 4")
fitt <- lm(res4 ~ as.numeric(1:length(res4.)))
abline(fitt, col="red")
abline(h=mean(res4.), col="blue")
qqnorm(res4.,main= "Normal Q-Q Plot for Model 4")
qqline(res4.,col="blue")
```







We can observe that it roughly follow a normal distribution from the histogram and q-q plot. Also, there is no trend or obvious seasonality from the time series plot of the residuals. Then, we also check for several independence assumptions by Portmanteau Statistics.

```
#Portmanteau Statistics for fit4a
shapiro.test(res4)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res4
## W = 0.98527, p-value = 0.004528
```

```
Box.test(res4, lag = 17, type = c("Box-Pierce"), fitdf = 2)
```

```
##
##  Box-Pierce test
##
## data:  res4
## X-squared = 12.006, df = 15, p-value = 0.6786
```

```
Box.test(res4, lag = 17, type = c("Ljung-Box"), fitdf = 2)
```

```
##
##  Box-Ljung test
##
## data:  res4
## X-squared = 12.446, df = 15, p-value = 0.645
```

```
Box.test(res4^2, lag = 17, type = c("Ljung-Box"), fitdf = 0)
```

```
##
##  Box-Ljung test
##
## data:  res4^2
## X-squared = 18.306, df = 17, p-value = 0.3698
```

```
#Portmanteau Statistics for fit4
shapiro.test(res4.)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res4.
## W = 0.98619, p-value = 0.006966
```

```
Box.test(res4., lag = 17, type = c("Box-Pierce"), fitdf = 2)
```

```
##
##  Box-Pierce test
##
## data:  res4.
## X-squared = 12.361, df = 15, p-value = 0.6516
```

```
Box.test(res4., lag = 17, type = c("Ljung-Box"), fitdf = 2)
```

```
##
##  Box-Ljung test
##
## data:  res4.
## X-squared = 12.819, df = 15, p-value = 0.6163
```

```
Box.test(res4.^2, lag = 17, type = c("Ljung-Box"), fitdf = 0)
```

```
##
##  Box-Ljung test
##
## data:  res4.^2
## X-squared = 18.769, df = 17, p-value = 0.342
```

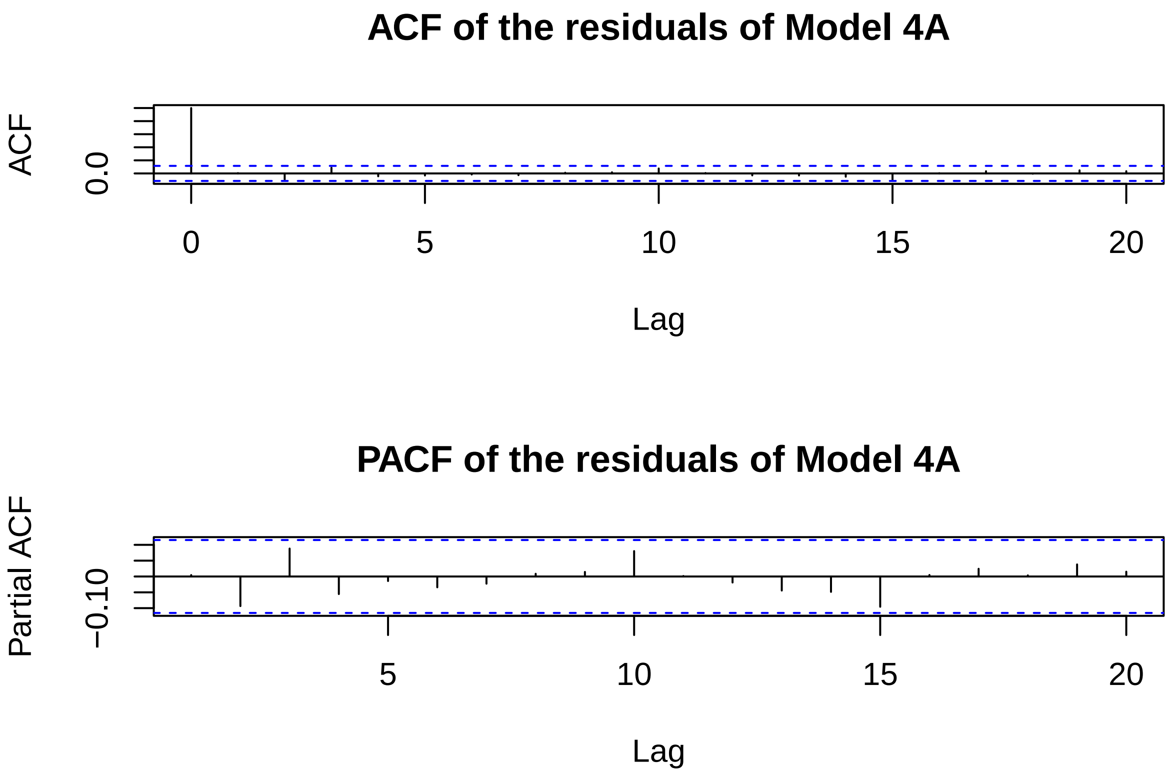
Here, we choose lag = 17 because we have n=290 observations and the square root is about 17. fitdf= p+q so therefore, fitdf = 2. We can see that the p-values are larger than 0.05 for all the tests (except for Shapiro-Wilk) Since fit4A has lower AICc value, we will ultimately go with model fit4A.

Next, we can see that ar() function recommended order 0 result for the residuals, which means that the residuals are WN.

```r
ar(res4, aic = TRUE, order.max = NULL, method = c("yule-walker"))
```

```
##
## Call:
## ar(x = res4, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  0.01553
```

```r
par(mfrow=c(2,1))
acf(res4, lag.max=20,main="")
title("ACF of the residuals of Model 4A")
pacf(res4, lag.max=20,main="")
title("PACF of the residuals of Model 4A")
```



Lastly, we create ACF and PACF of the residuals. ACF and PACF values at all lags are within the confidence interval. So the residuals can be seen as White Noise and we will use model fit4A instead of model fit4.

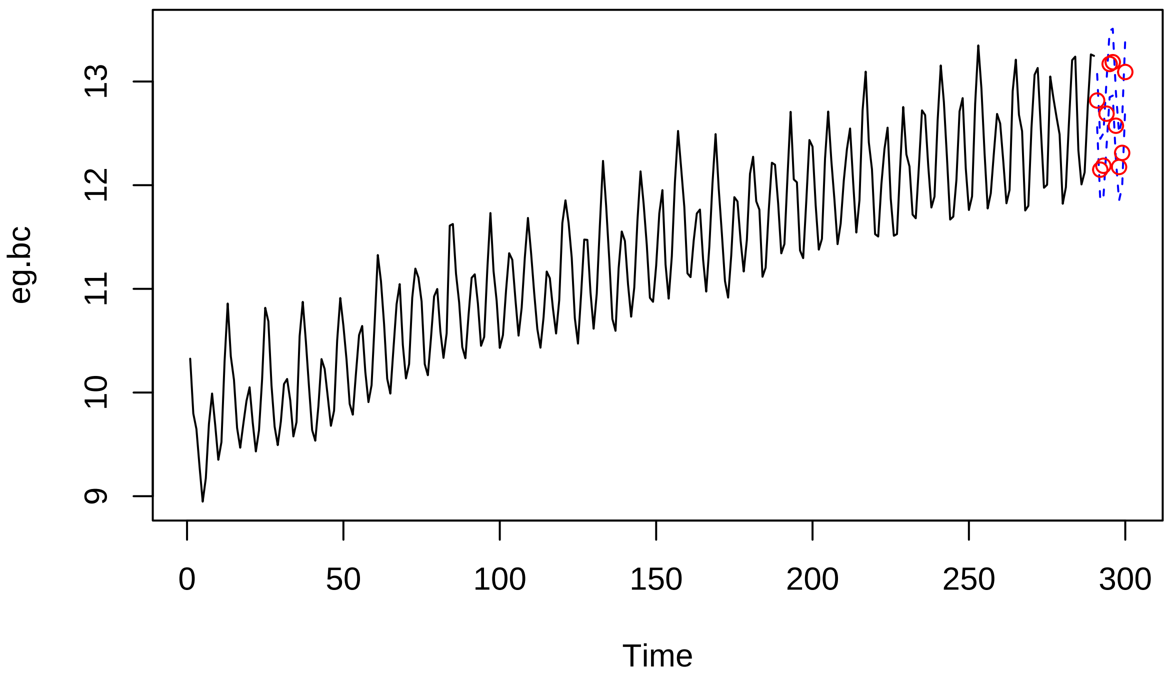
Forecasts on Original data Plot

```r
library(forecast)
```

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

In forecasting section, we predict the armed robberies from March 2008- December 2007 based on our model and then compare with the true values.

```r
pred.tr <- predict(fit4a, n.ahead = 10)
U.tr= pred.tr$pred + 2*pred.tr$se
L.tr= pred.tr$pred - 2*pred.tr$se
ts.plot(eg.bc, xlim=c(1,length(eg.bc)+10), ylim = c(min(eg.bc),max(U.tr)))
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(eg.bc)+1):(length(eg.bc)+10), pred.tr$pred, col="red")
```

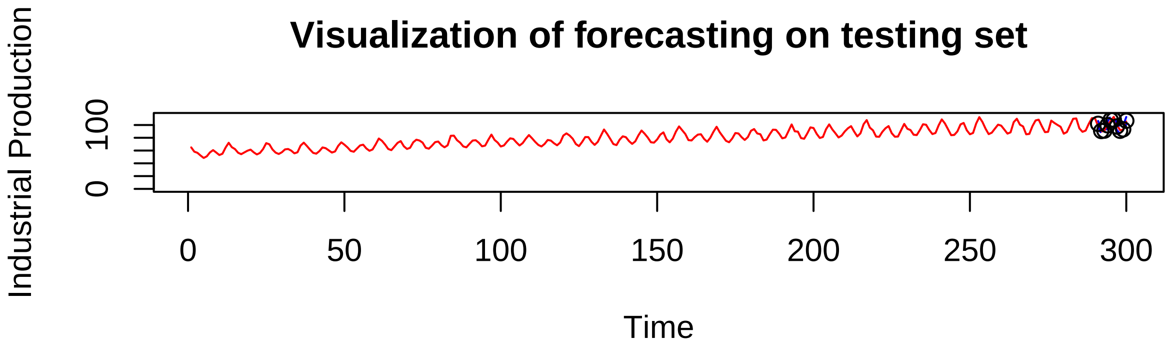


The above figure is the forecast on the transformed data. The true values are within the confidence interval of the forecasting. If we want to compare with the true values of the last ten months, we need to convert the forecasting values back to the scale before box-cox transformation. Next part shows how to convert the data back to the scale before box-cox transformation and compare the true values with predicted values.

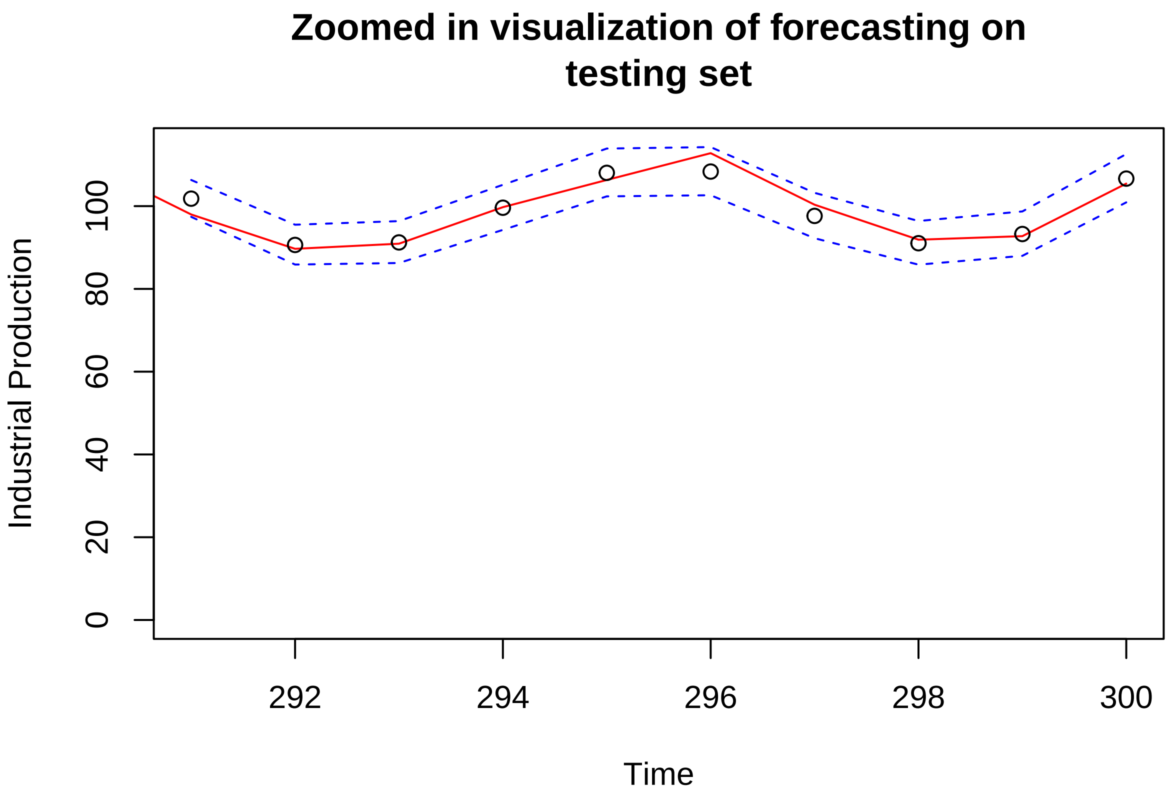
Forecast with original values Zoomed in plot

```r
# Applying inverse Box-Cox transformation
pred.orig <- if (lambda != 0) {
  ((pred.tr$pred * lambda + 1)^(1/lambda)) - 1}


U <- if (lambda != 0) {
  ((U.tr * lambda + 1)^(1/lambda)) - 1}

L <- if (lambda != 0) {
  ((L.tr * lambda + 1)^(1/lambda)) - 1}
```

```r
# Plotting
par(mfrow = c(2, 1))
ts.plot(as.numeric(eg), ylim = c(0, max(U)), col = "red",
        ylab = "Industrial Production", main = "Visualization of forecasting on testing set")
lines(U, col = "blue", lty = "dashed")
lines(L, col = "blue", lty = "dashed")
points((length(train_data) + 1):(length(train_data) + 10), pred.orig, col = "black")
```



```r
ts.plot(as.numeric(eg), xlim = c(291,length(train_data)+10), ylim = c(0,max
(U)),col="red",ylab="Industrial Production",main="Zoomed in visualization of forecasting on
testing set")
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(train_data)+1):(length(train_data)+10), pred.orig, col="black")
```



This is the zoomed in time series plot of the original data with forecasted values in black circles and true values in the red line and confidence intervals in blue dashed lines. The forecasted values are very close to the true values and they are both within the confidence intervals. Model 4A performs well and can predict future values pretty accurately.

```
}
```