Time Series Analysis: Monthly Souvenir Shop Sales

Amy Kuang

Abstract

In this project, I examined monthly sales data for a souvenir shop on the wharf at a beach resort town in Queensland, Australia spanning between January 1987- December 1993. In Australia, Queensland is a notable tourist spot for its beaches and tropical islands; this project attempts to predict sales in a souvenir shop that can ultimately contribute to customer satisfaction with an unnamed Queensland beach resort as vacation spot. Hence, the goal of this project is to create a time series model to predict future sales. In particular, this project uses time series modeling on sales data collected from January 1987-December 1992 to fit a SARIMA model for forecasting. Forecasting values for January 1993-December 1993, the forecasted values are plotted against actual sales data collected from January 1993-December 1993 for comparison. At the end of the project, the results find that the fitted SARIMA model $SARIMA(1,1,0) \times (0,1,0)$ is generally accurate.

Introduction

The goal of this project is to create a SARIMA model to predict future sales for a souvenir shop. For this project, I used R and RStudio to analyze the time series dataset called "Monthly sales for a souvenir shop on the wharf at a beach resort town in Queensland, Australia. Jan 1987-Dec 1993." The data collected is between January 1987-Deccember 1993, and it can be found in the tsdl-package R library created by Rob Hyndman, Professor of Statistics at Monash University, Australia. This particular dataset contains 84 datapoints which is seven years of monthly sales data. The impact of this project is that it may help with forecasting future sales that may ultimately improve customer satisfaction.

To begin, I partitioned the monthly sales data into two: a 'training dataset' with data from January 1987-December 1992 for model fitting (called `train`) and a 'testing dataset' with data from January 1993-December 1993 (called `testing`) to compare forecast values with actual values to conclude model accuracy. Through analysis of time series plots, ACF and PACF, histograms, and a box-cox transformation of the data, I was able to construct a SARIMA model fit for subsequent testing. Following, a series of diagnostics check testing was conducted on the model and residuals to check for model accuracy. After constructing and testing the fitted model, the model forecasts sales values for January 1993-December 1993. Plotting the forecasted values alongside the actual data from January 1993-December 1993 from the original dataset for comparison, the results find that the SARIMA model SARIMA(1,1,0) x (0,1,0) is concluded to be generally accurate and may be suitable to use with forecasting future sales.

Data

This is sales time series data:

```
##          Jan      Feb      Mar      Apr      May      Jun      Jul
## 1987 1664.81  2397.53  2840.71  3547.29  3752.96  3714.74  4349.61
## 1988 2499.81  5198.24  7225.14  4806.03  5900.88  4951.34  6179.12
## 1989 4717.02  5702.63  9957.58  5304.78  6492.43  6630.80  7349.62
## 1990 5921.10  5814.58 12421.25  6369.77  7609.12  7224.75  8121.22
## 1991 4826.64  6470.23  9638.77  8821.17  8722.37 10209.48 11276.55
```

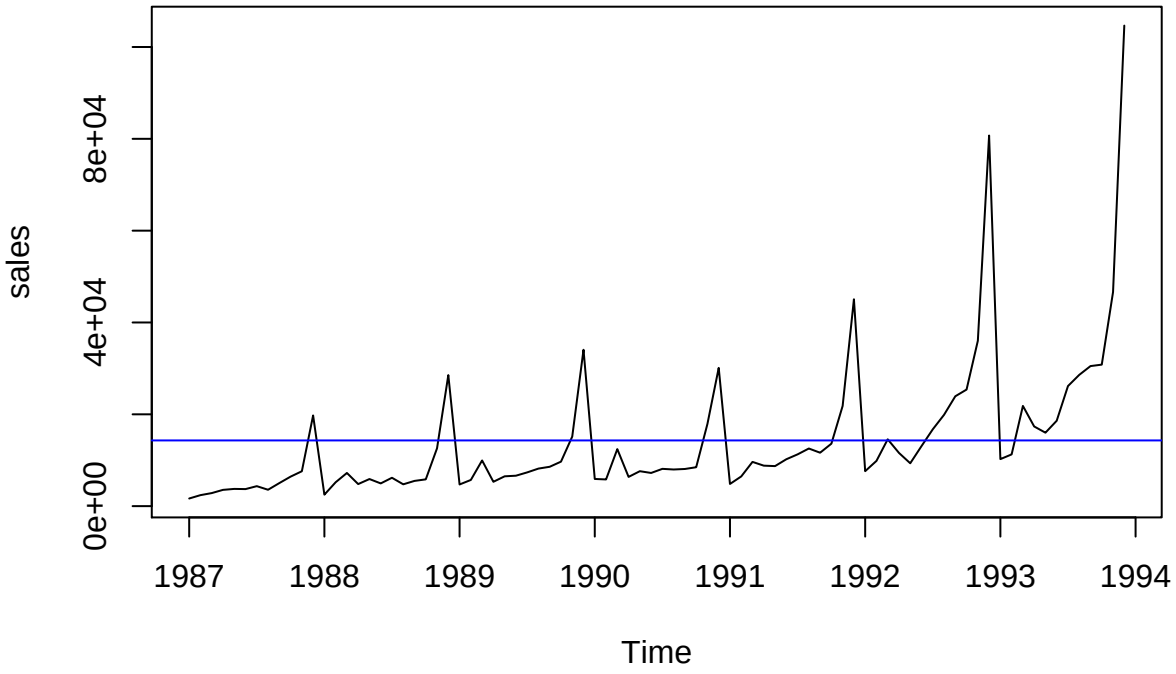
```
## 1992   7615.03   9849.69  14558.40  11587.33   9332.56  13082.09  16732.78
## 1993  10243.24  11266.88  21826.84  17357.33  15997.79  18601.53  26155.15
##            Aug       Sep       Oct       Nov       Dec
## 1987   3566.34   5021.82   6423.48   7600.60  19756.21
## 1988   4752.15   5496.43   5835.10  12600.08  28541.72
## 1989   8176.62   8573.17   9690.50  15151.84  34061.01
## 1990   7979.25   8093.06   8476.70  17914.66  30114.41
## 1991  12552.22  11637.39  13606.89  21822.11  45060.69
## 1992  19888.61  23933.38  25391.35  36024.80  80721.71
## 1993  28586.52  30505.41  30821.33  46634.38 104660.67
```

As shown above, the data collected contains monthly sales from January 1987 to December 1993. Next, I will plot the time series data to analyze.

Time Series Data Plot

Below is a plot of the time series sales data. The data indicates an estimated mean of 14315.59.

```
## [1] 14315.59
```



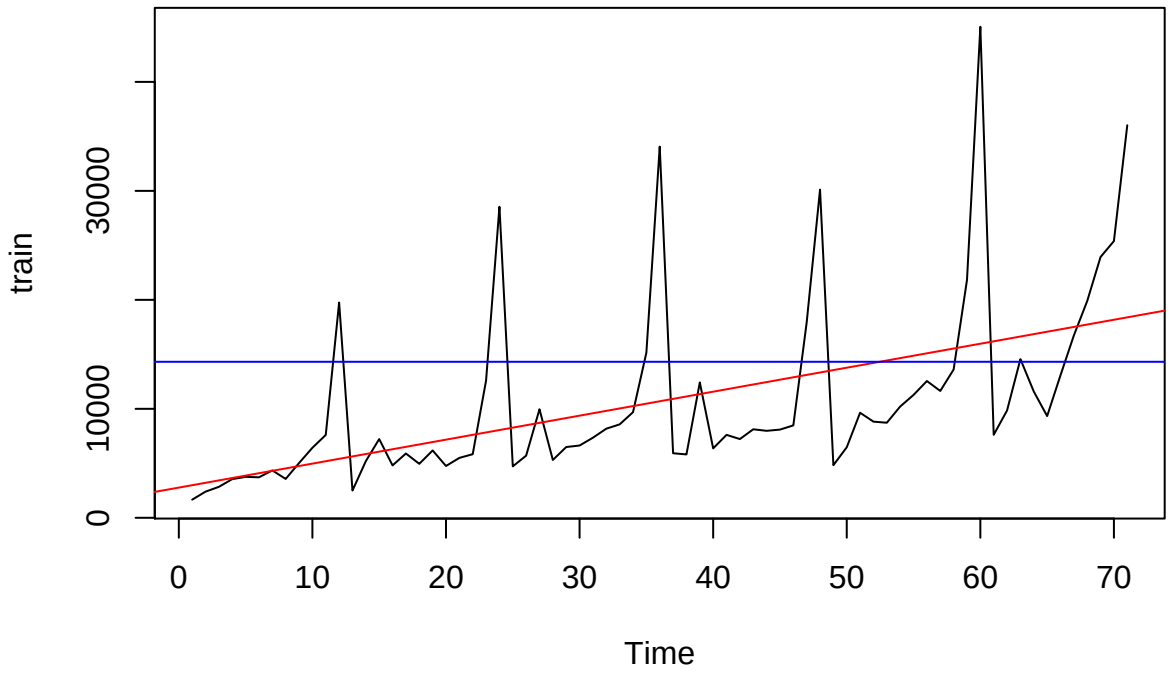
According to the time series plot, there appears to be a sharp-high and increasing spike each year that suggests a possible (yearly) seasonal component and a linear trend. Based on these observations, it appears that a constant mean and variance is very unlikely.

Partitioning Data

To begin time series modeling for fitting, I partitioned the monthly sales data into 2 datasets: a "training dataset" called `train` for model fitting with datapoint observations 1-71 and a "test dataset" called `testing` containing the last 12 datapoint observations that will be used to check the final fitted model's forecasting accuracy.

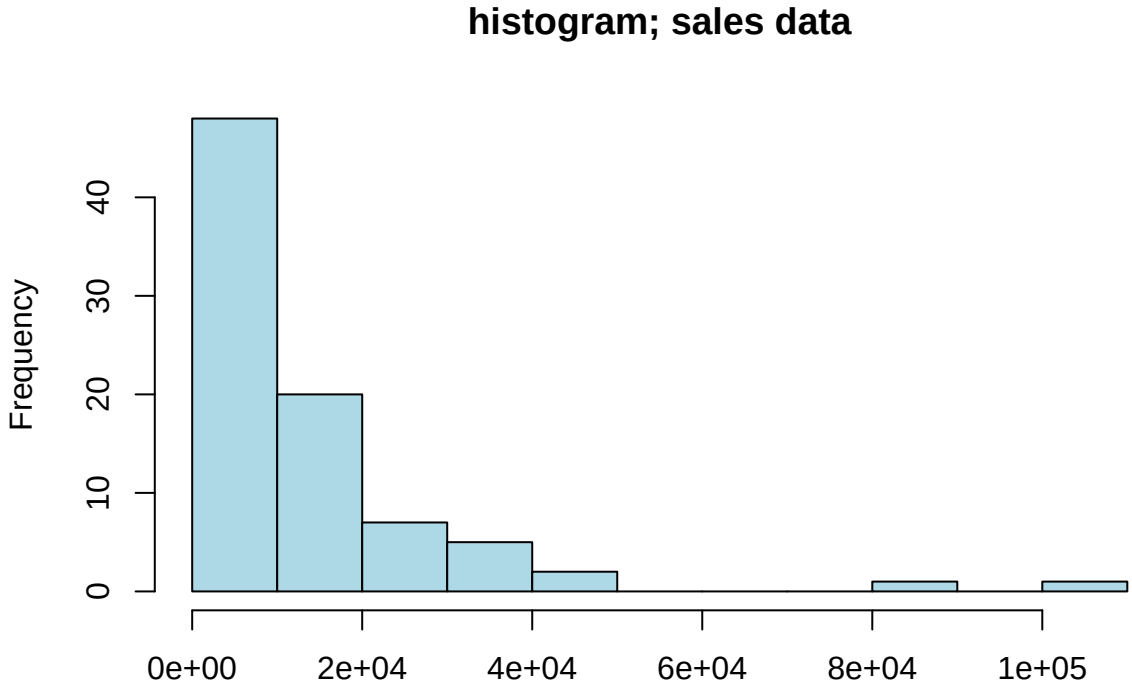
```
# partition data set for model training and model validation
train = sales[1:71]       # training dataset
testing = sales[72:84]    # test dataset
plot.ts(train)
```

```r
fit <- lm(train ~ as.numeric(1:length(train)))
abline(fit, col="red")
abline(h=mean(sales), col="blue")
```



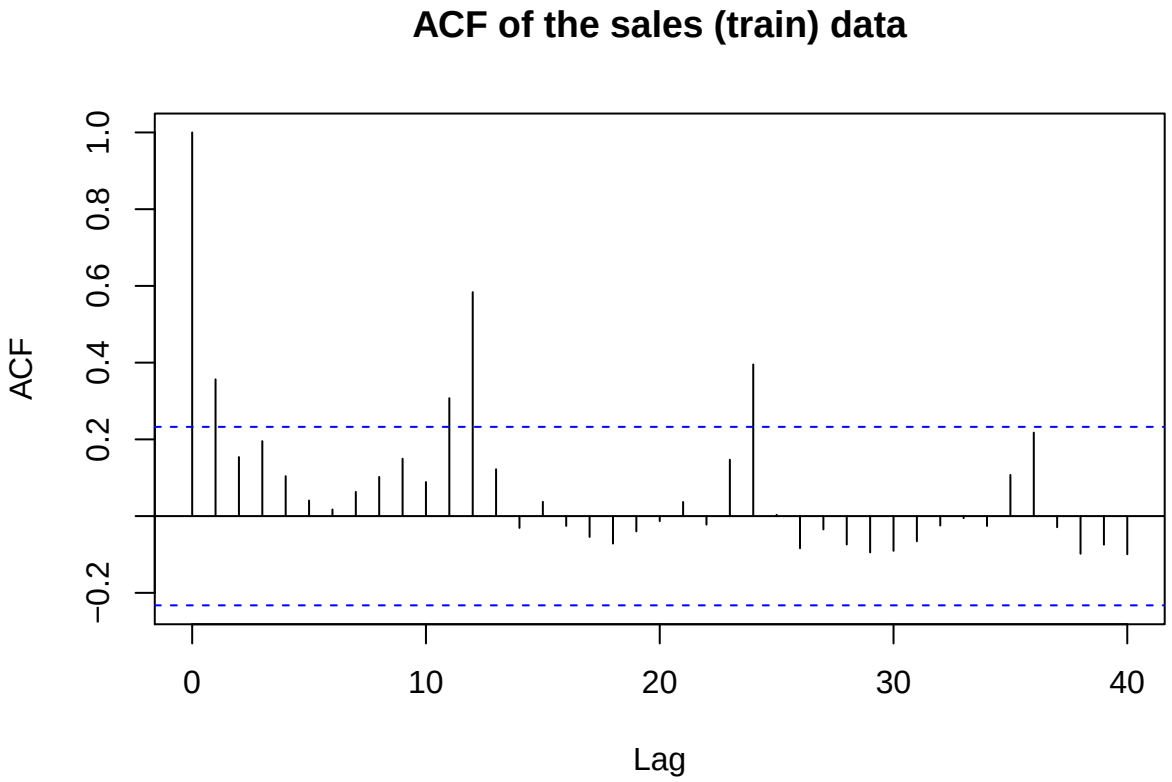
In the partitioned training dataset above, there appears to be a positive-linear trend, (yearly) seasonal component with frequency 12. A non-constant mean and variance appears to be highly likely as well. With large seasonal up-spikes at frequencies of 12 (at time 12,24,36,48,60), positive-linear trend, non-constant mean and variance, we conclude that the data is highly non-stationary.

To reaffirm the observation of non-constant mean and variance, a histogram is also plotted of the original sales data.



As shown above in the plot, the histogram appears to be rightly skewed and does not appear to look normally distributed. With this, we conclude that there is definitely non-constant mean and variance.

Next, the ACF is plotted with `lag.max = 40` for a closer look at possible trend and seasonality.

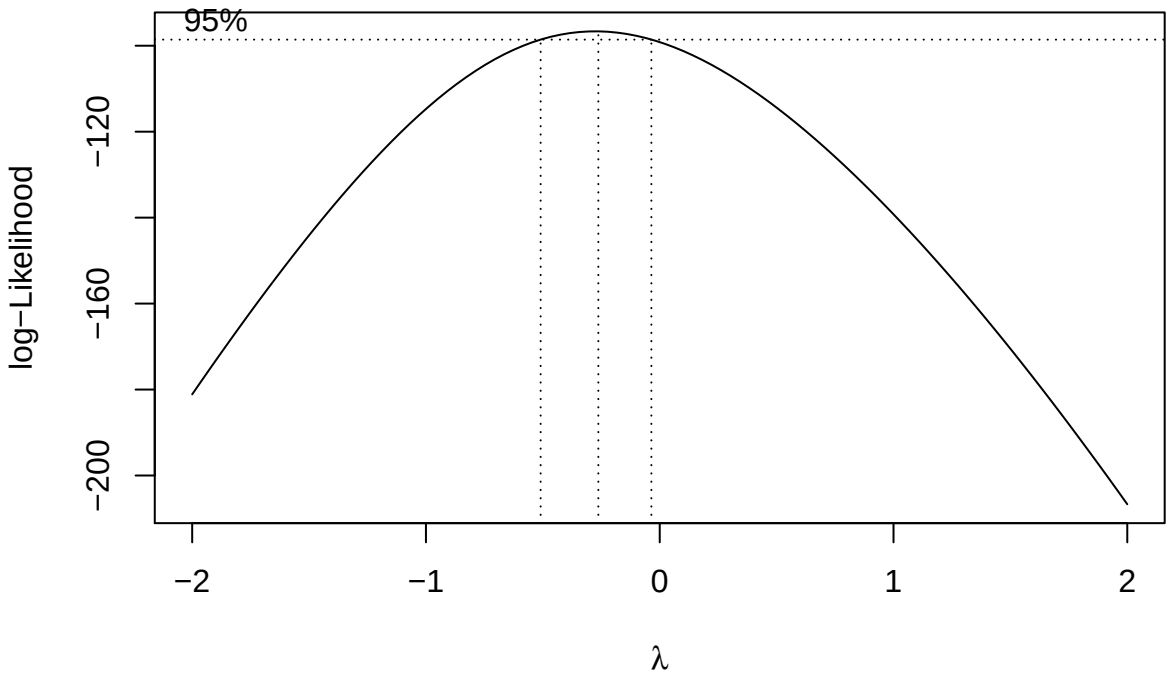


Based on the ACF, the ACF appears to have a (negative) linear trend. There is slight indication of a seasonal component with lag $12, 24$. Moving foward, data transformation should be conducted to stabilize variance, and differencing should be done to remove trend and seasonality.

Data Transformation

To make the data more suitable for use, we need to make the time series stationary. Hence, a Box-Cox test is conducted on the data (`train`) to determine what kind of data transformation may be appropriate.

```
# Box-Cox test for data transformation
bcTransform <- boxcox(train~ as.numeric(1:length(train))) # plots the graph
```




```
bcTransform$x[which(bcTransform$y == max(bcTransform$y))] # gives the value of lambda
```

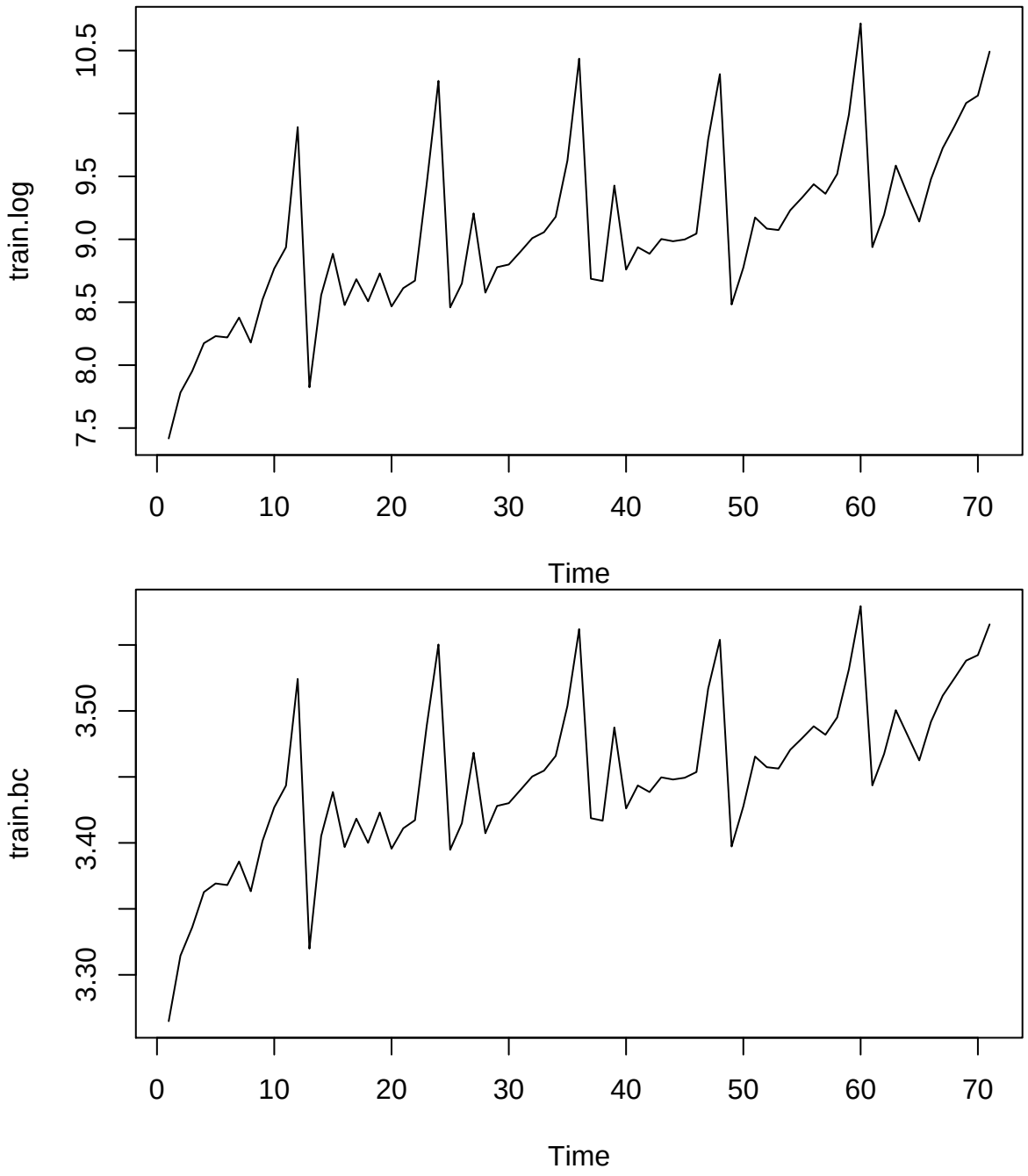
```
## [1] -0.2626263
```

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

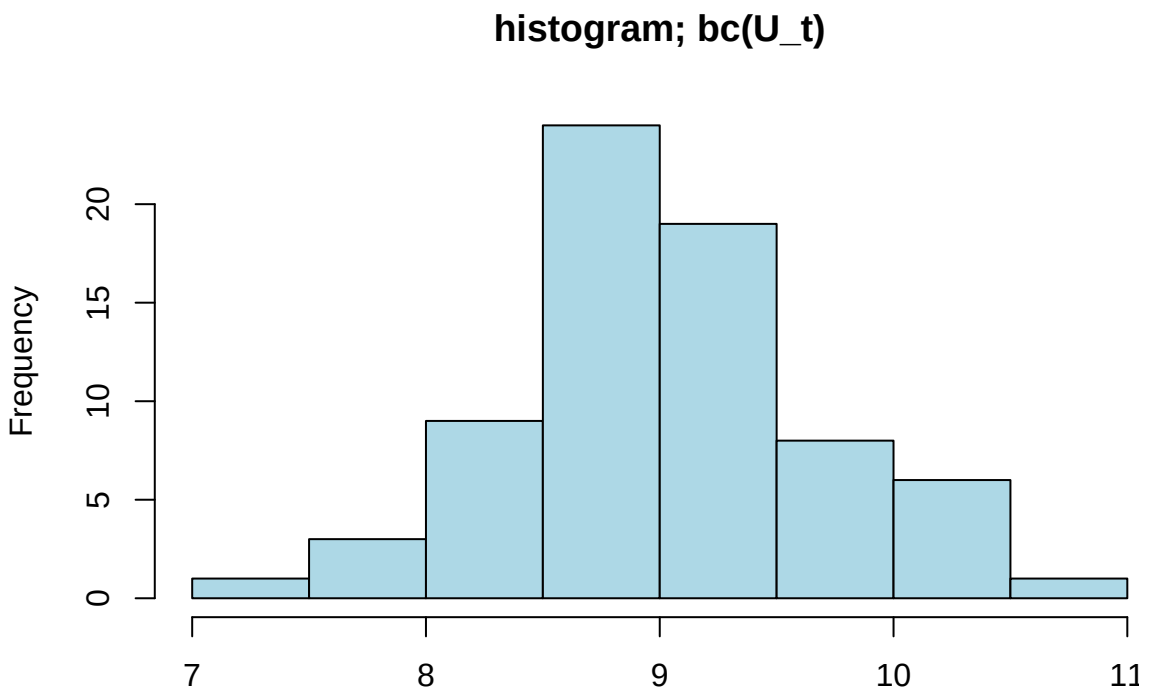
According to the Box-Cox test, a λ value of -0.2626263 is suggested for data transformation by Box-Cox.

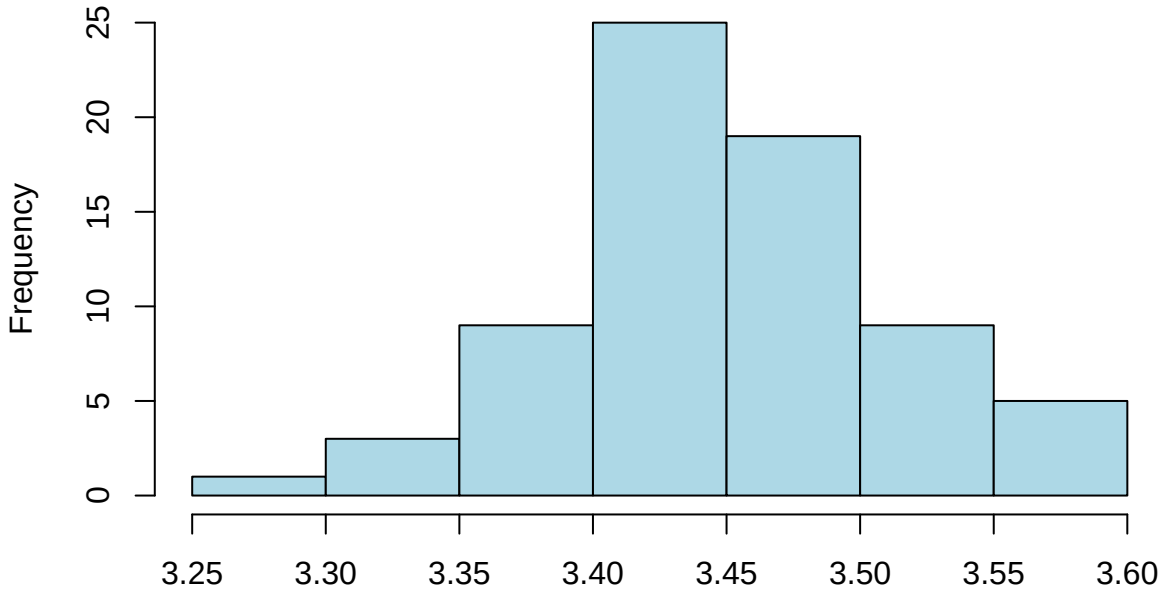
However, let's look at time series plots and histograms of the data being log-transformed and Box-Cox transformed with $\lambda = -0.2626263$ first. I denoted `train.log` or $bc(U_t)$ for log-transformed data and `train.bc` or $\text{bc}(U_t)$ for Box-Cox transformed data.

```
# Perform transformations, plot transformed data, histograms:
lambda=bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
train.bc = (1/lambda)*(train^lambda-1)
train.log <- log(train)
```



According to the time series plots, the Box-Cox transformed data appears to be less varied – more stable variance– than the log-transformed data as indicated through the y-axis values. The mean also appears to be smaller in the Box-Cox transformed data compared to the log-transformed data. Aiming for little to no and or small variance and mean, let's analyze their histograms for further observation.



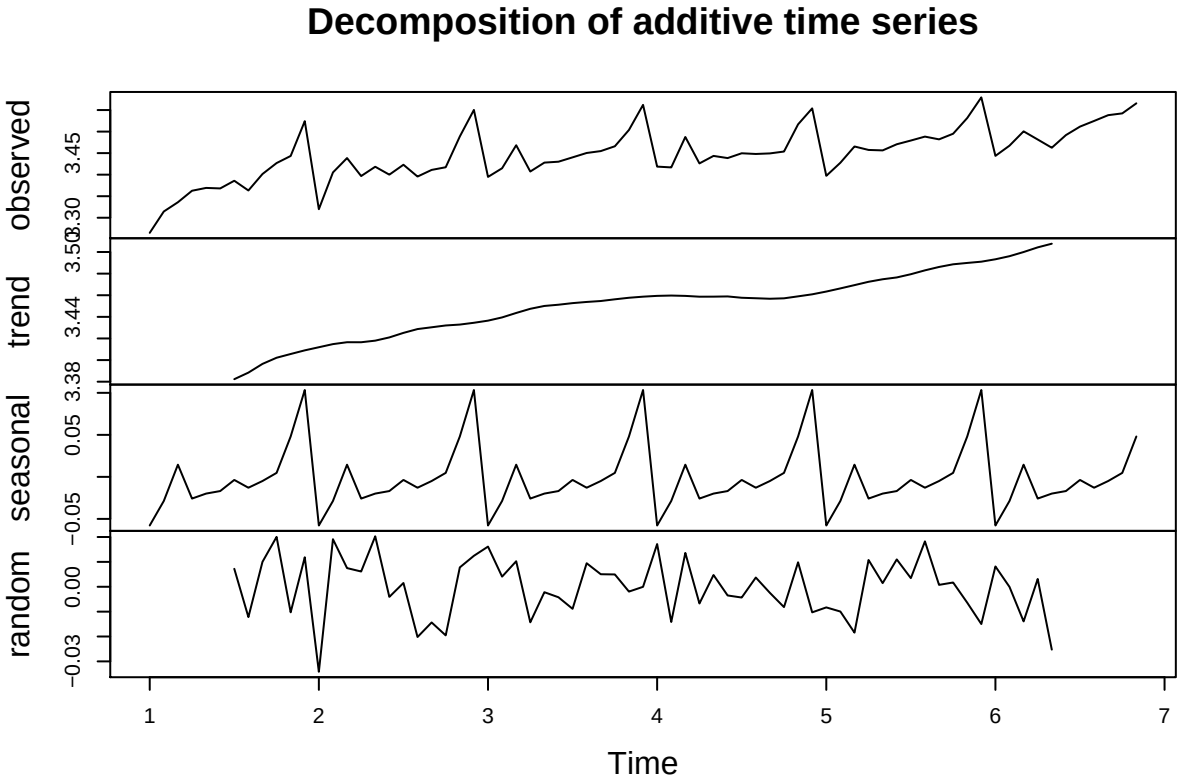


According to the histograms of the transformed data, the Box-Cox transformed data appears to have a smaller and more consistent mean than the log-transformed data. Furthermore, the Box-Cox transformed data's histogram appears to indicate a smaller mean and less variance compared to the log-transformed data's histogram. With this result, it makes sense to proceed with Box-Cox transformed data over the log-transformed data since we would want to get a stationary series for model fitting.

Thus, I choose to proceed with the Box-Cox transformed data.

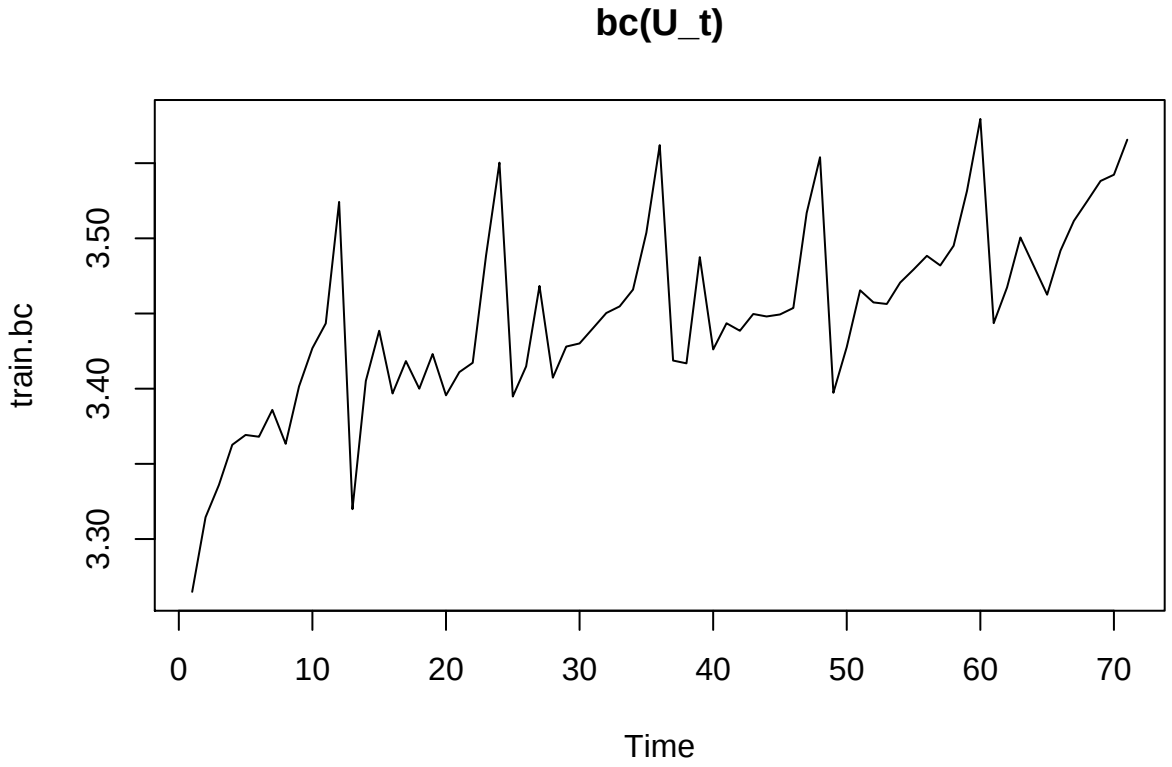
Decomposition of Box-Cox Transformed Data

Next, I proceed with a decomposition of $\text{bc}(U_t)$ (the Box-Cox transformed data) to analyze if there is trend and or seasonality present.



According to the decomposition of $\text{bc}(U_t)$ (Box-Cox transformed data) above, we see an almost linear trend and evident seasonality.

Differencing

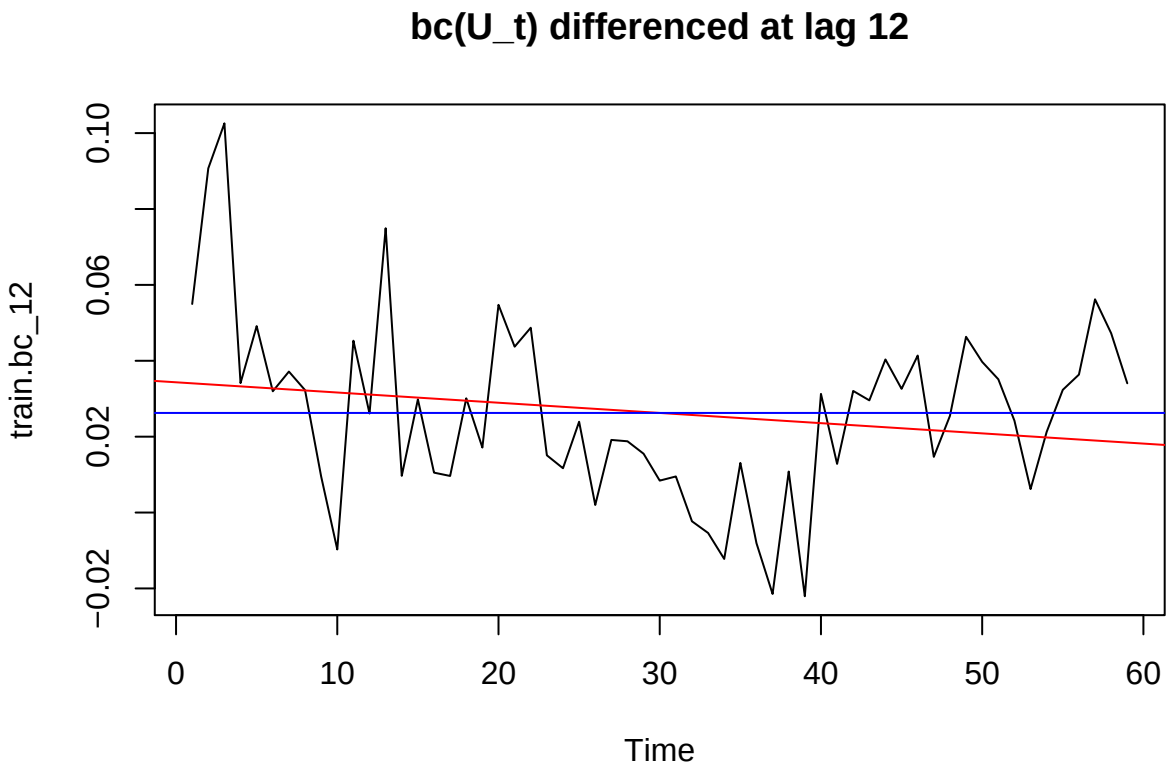



```
## [1] 0.003854942
```

Prior to differencing, bc(U_t) appears to have seasonality, a slightly positive trend, and variance of 0.003854942.

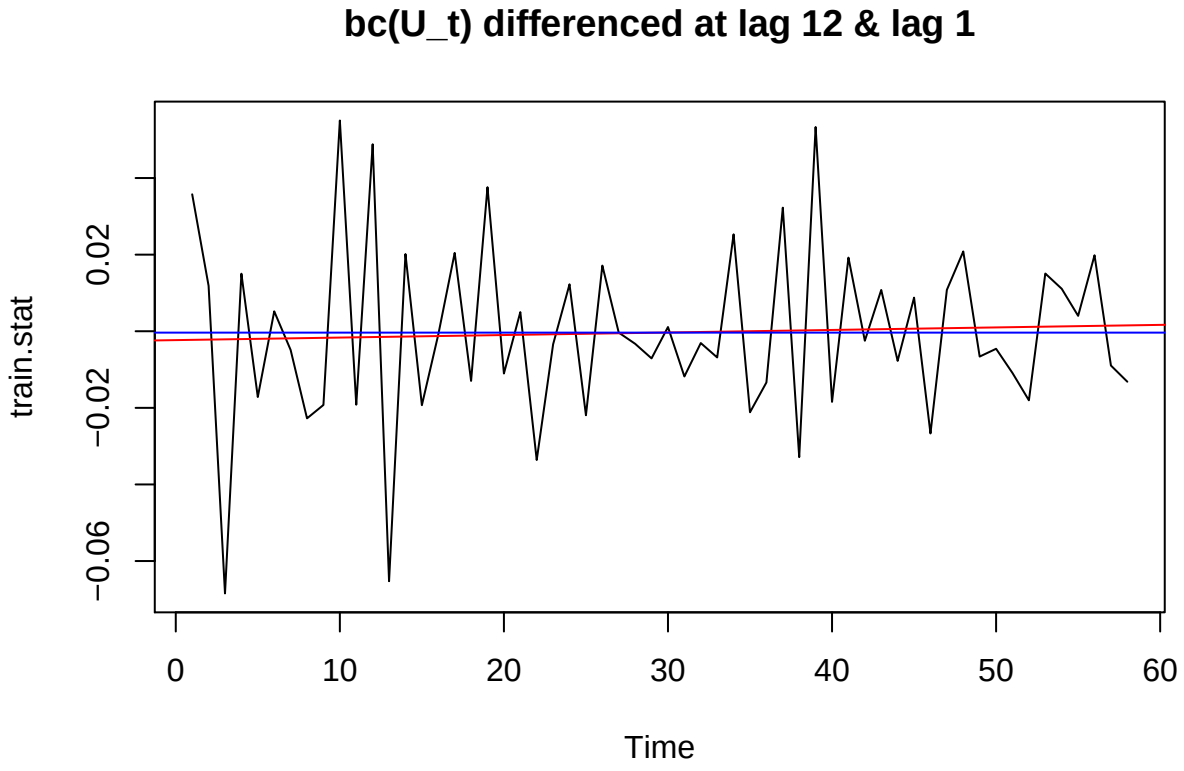
Moving forward, I use differencing at lag 12 to remove seasonality and then at lag 1 to remove trend.

```
## [1] 0.000571439
```



By differencing at lag 12, I remove seasonality. According to the plot of bc(U_t) differenced at lag 12, seasonality no longer as evident. The variance has also decreased, going from 0.003854942 to 0.000571439; the mean is 0.02624927. However, there is still a slight trend present as shown by the red line.

Next, I difference at lag 1 to remove trend.



```
## [1] 0.0005760136
```

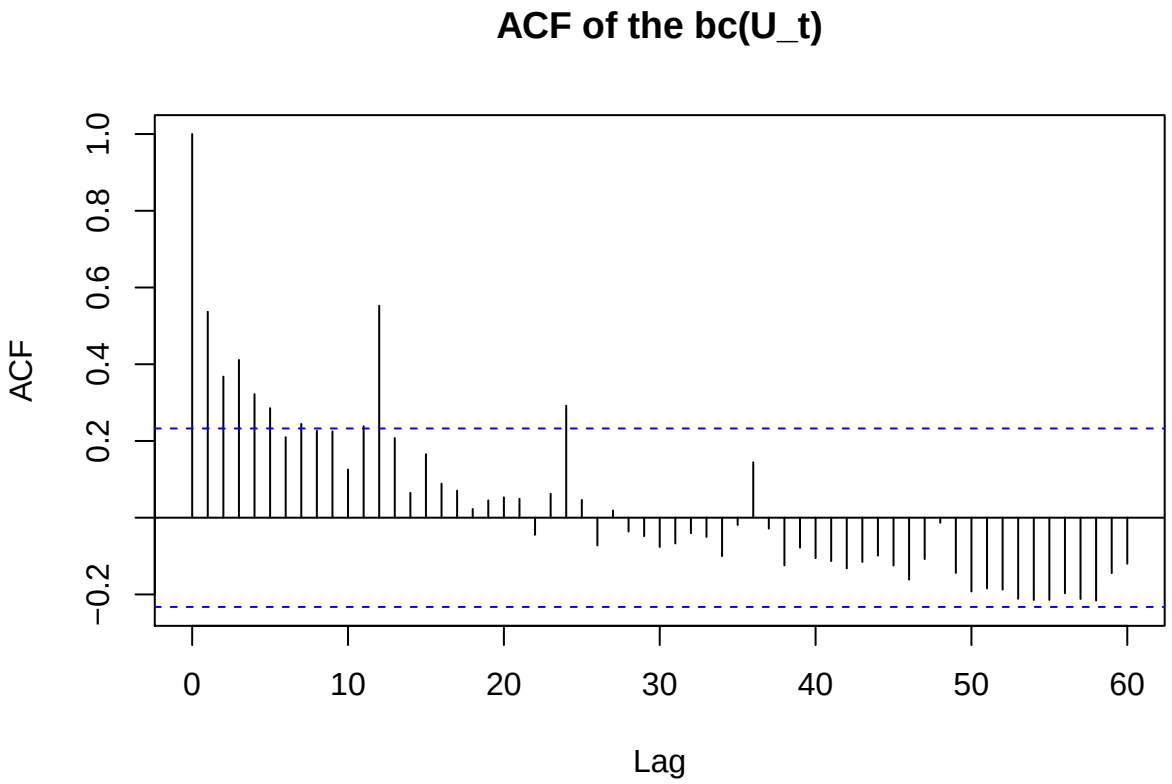
According to the plot of bc(U_t) differenced at lag 12 and 1, there is no more seasonality. The variance has also decreased, going from 0.000571439 to 0.0005760136. There also appears to be no more trend (red line) as the mean appears to be very close to zero (-0.0003603261).

The data now looks stationary and no further differencing should be done since the change in variance from differencing after 12 appears to be very small; this suggests that a second round of differencing may not be required. With trend and seasonality removed and a very small variance, the series appears to be stationary. To prevent over-differencing that could increase the variance, no further differencing is conducted.

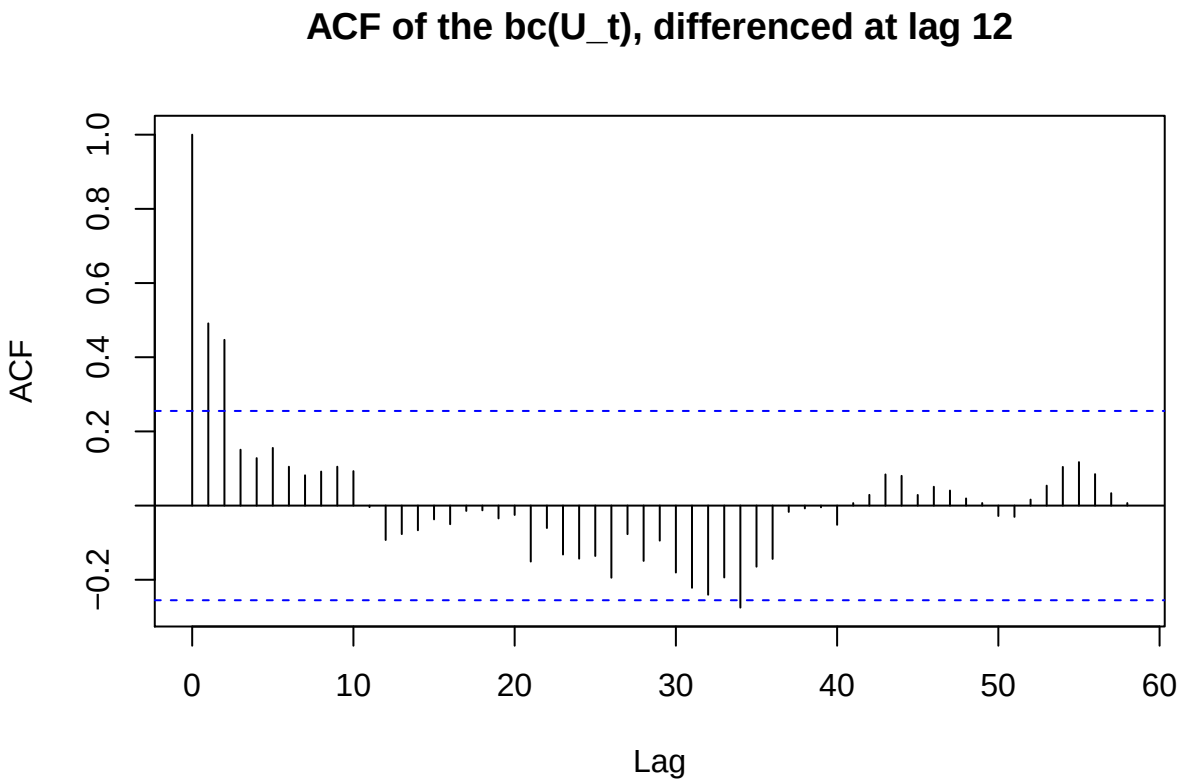
ACF and PACF; Model Estimation

Having differenced our Box-Cox transformed data, let's analyze the ACF and PACF of bcU_t, bcU_t after differenced at lag 12, and bc(U_t) after differenced at lag 12 and 1 in order to proceed with model estimation for parameters p, q, P, Q, s, D, d. Note: differencing at lag $12 = D = 1$, differencing at lag $1 = d = 1$.

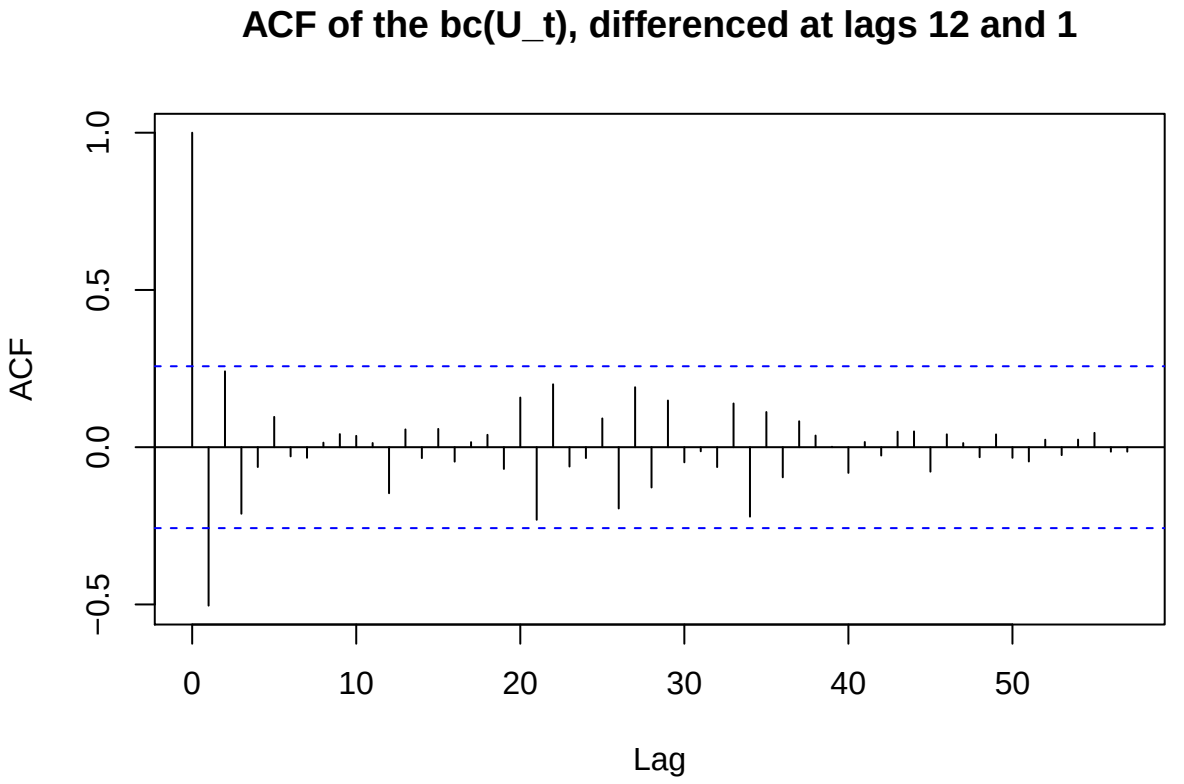
First, we will look at ACF as we attempt to estimate p, Q values.



According to the ACF plot of bc(U_t) above, the ACF appears to show a slight decreasing-like decay and a little bit of seasonality – which indicates non-stationarity.



After differencing at lag 12, the ACF plot of bc(U_t) differenced at lag 12 apperas to show fewer instances in lag that surpass the 95% confidence interval. The seasonality is still slighly apparent, but the downward decay from before is slowly going away. However, the ACF plot here still indicates some non-stationarity.

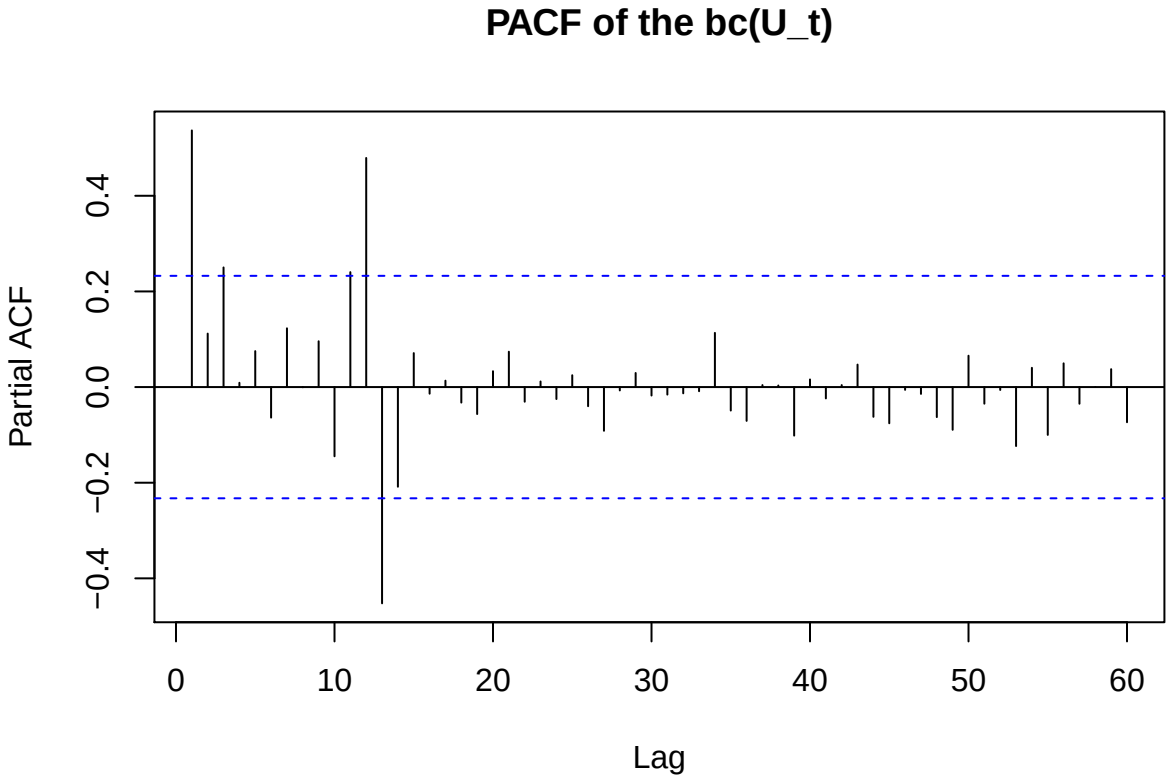


After differencing at lag 12 and 1, the ACF decay appears to follow a stationary process. The ACF also

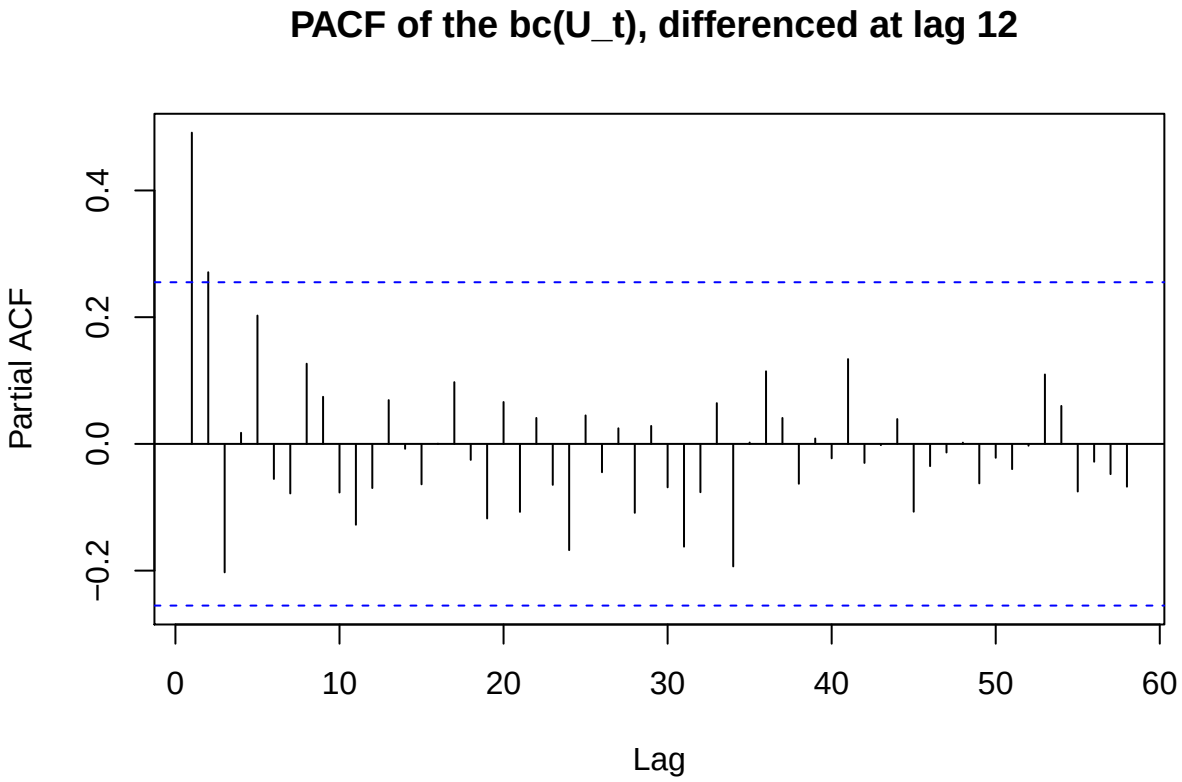
appears to be within the 95% confidence interval.

For model estimation, the ACF of $\text{bc}(U_t)$ difference at lags 12 and 1 have no major spikes at lag $12, 24, 36, 48, 60$, so it is suitable to estimate $Q = 0$. Looking at the lags from $0 - 11$, since our frequency $s = 12$ (given by the dataset), a suitable choice in p could be $p = 0$ or 1.

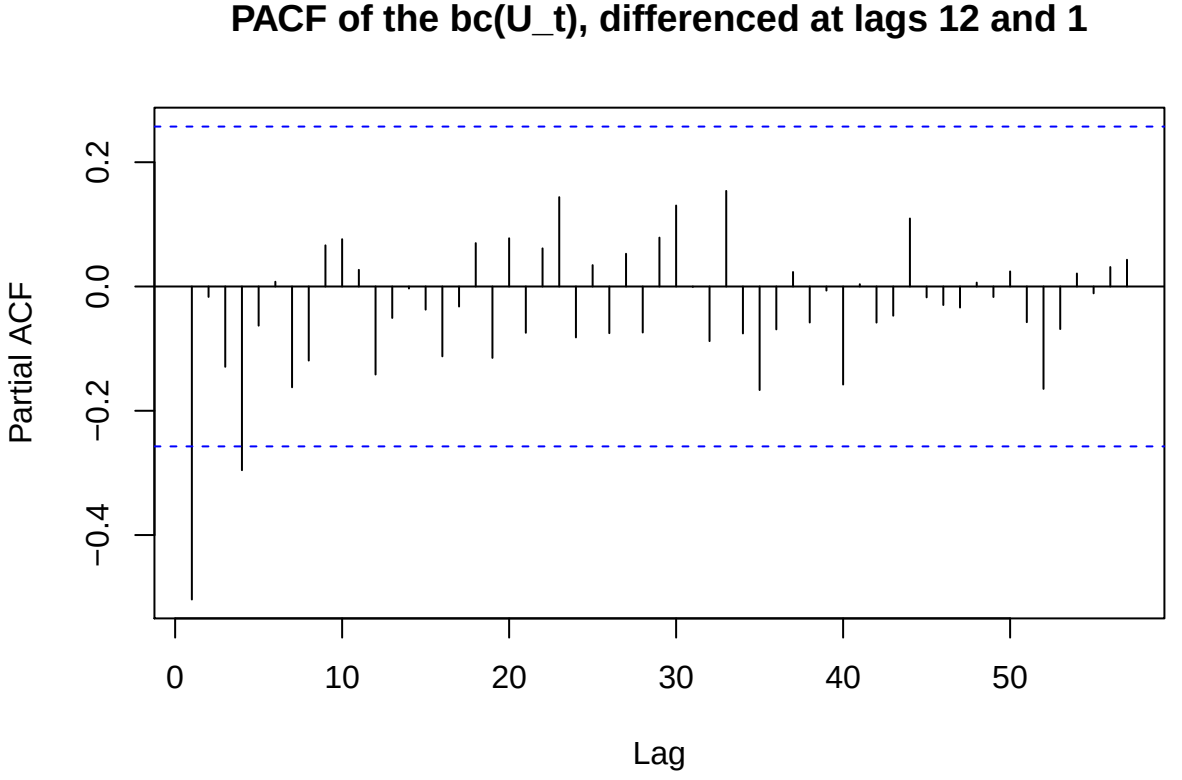
Next, we analyze PACF undergoing before and after differencing – to help estimate P, q values.



According to the PACF of the $\text{bc}(U_t)$, there appears to be major spikes around lags $0, 12, 13$. Further, the data appears to be within the 95% confidence interval bounds; with a few at lag 3 and 11 that crossing the 95% confidence interval by a little bit. Elsewise, in terms of seasonality, the seasonality is not that apparent; the decay is somewhat slow and apparent in the beginning around lags $0 - 12$. Else, the PACF relatively follows a stationary process.



According to the PACF bc(U_t) differenced at lag 12, there only appears to be one major spike at lag 1 and one a tiny spike that crosses the 95% confidence interval at lag 2 – this tiny spike is neglible as it does appear to be significant. The seasonality and decay is not very apparent as the decay is barely noticeable in lags 0-12. Thus, the PACF appears to follow a relatively stationary process.

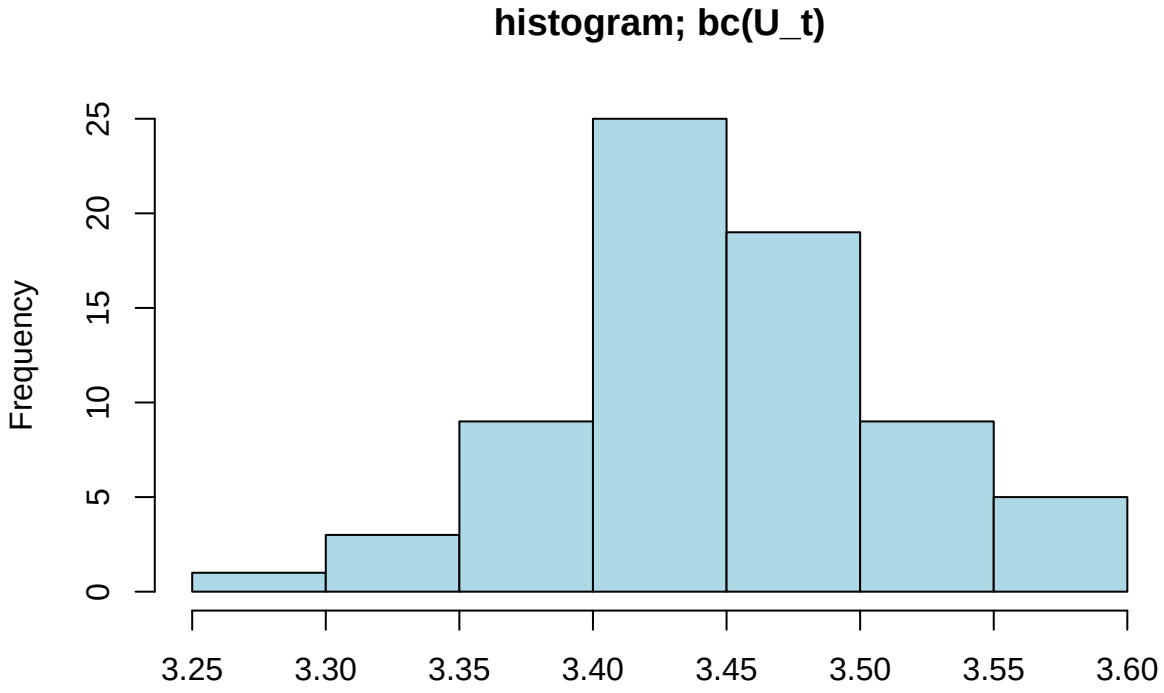


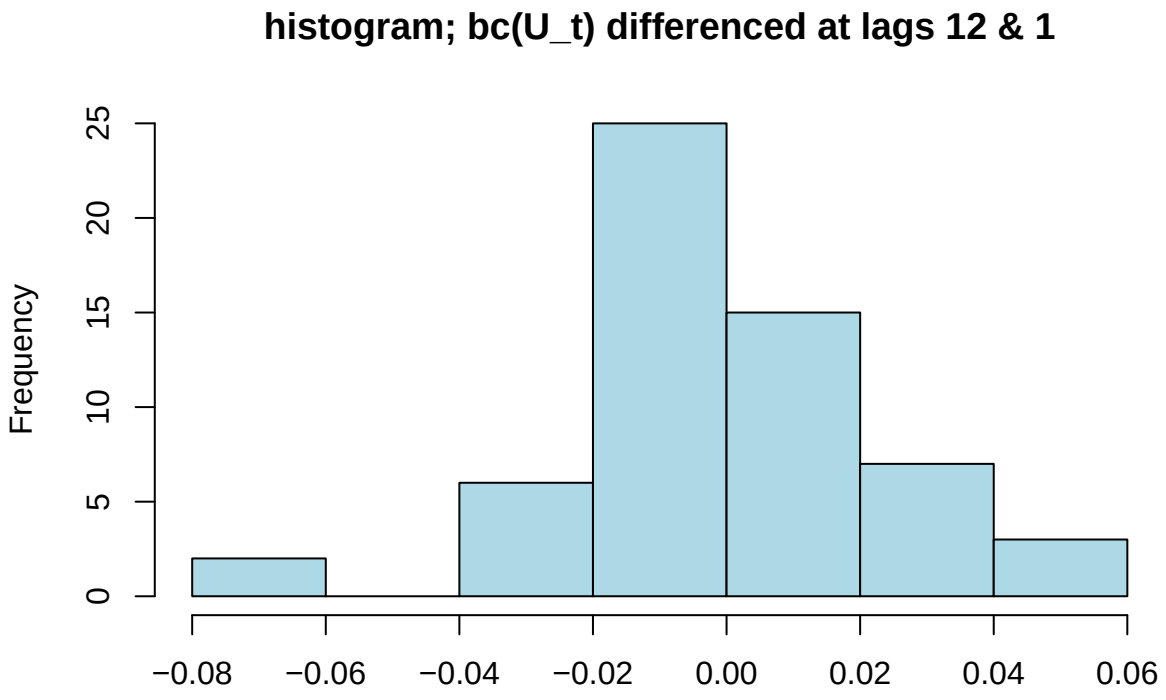
After differencing at lag 12 and 1, there is no seasonality and trend; the PACF decay appears to follow a stationary process. The PACF also appear to mostly be within the 95% confidence interval. There is major spike at lag 1 and small spike at lag 4 that can be considered. A suitable P may be $P = 0$ since there are no major spikes at around lag $12, 24, 36, 48, 60$. Looking at lags $0 - 11$, a suitable q may be $q = 1$ or 4.

Looking at the ACF and PACFs of the time series before and throughout differencing at lag 12 and 1, we conclude that we will work with $\nabla_1 \nabla_{12} bc(U_t)$ as the seasonality and decay are not apparent AND the ACF and PACF correspond to a stationary process. Note: the variances before and after differencing have already indicated that working with $\nabla_1 \nabla_{12} bc(U_t)$ will bring about a more constant mean and variance (that is smaller).

Histogram

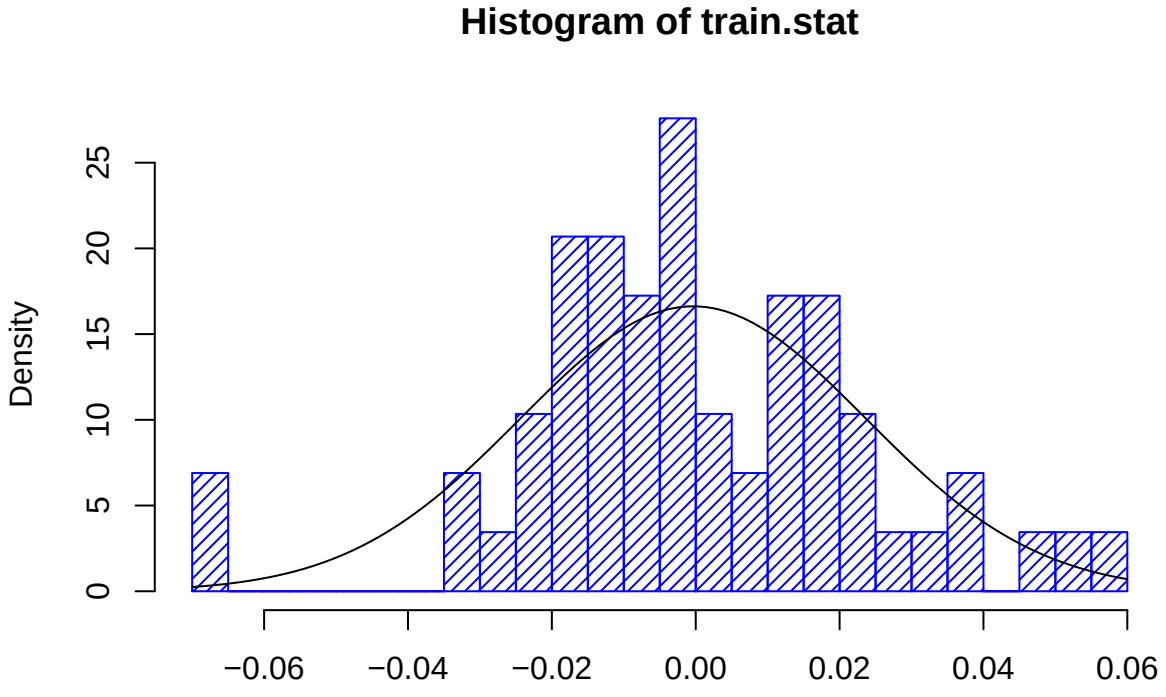
Next, we look at the histograms to analyze the effects of differencing at lag 12 and 1 on variance and mean:





Compared to the histogram of $\text{bc}(U_t)$ without differencing at lags 12 and 1, the histogram of $\text{bc}(U_t)$ differenced at lags 12 and 1 showcases smaller variance and (nearly) constant zero mean.

```
# Histogram of transformed and differenced data with normal curve:
hist(train.stat, density=20,breaks=20, col="blue", xlab="", prob=TRUE)
m <- mean(train.stat)
std <- sqrt(var(train.stat))
curve( dnorm(x,m,std), add=TRUE )
```



Plotting a curve above $\nabla_1\nabla_{12}bc(U_t)$ ($\text{bc}(U_t)$ differenced at lags 12 and 1), we see that the histogram looks symmetric and almost Gaussian – appearing to following a relatively normal distribution.

Model Fitting

Based on the ACF and PACF plots, a list of candidate models that can be tried are: SARIMA for $\text{bc}(U_t)$: $s = 12, D = 1, d = 1, Q = 0, P = 0, p = 0$ or $1, q = 0$ or 4. We are now ready for model fitting:

First, we try SAR models: $p = 0$ or $1, P = 0$:

```
# SAR models: p=0 or 1, P=0
# Model 1
sar1 <- arima(train.bc, order=c(0,1,0), seasonal = list(order = c(0,1,0),
                                                        period = 12), method="ML")
sar1
```

```
##
## Call:
## arima(x = train.bc, order = c(0, 1, 0), seasonal = list(order = c(0, 1, 0),
##     period = 12), method = "ML")
##
##
## sigma^2 estimated as 0.0005662:  log likelihood = 134.52,  aic = -267.04
```

```
AICc(sar1) # AICc
```

```
## [1] -266.9711
```

```
# Model 2
sar2 <- arima(train.bc, order=c(1,1,0), seasonal = list(order = c(0,1,0),
                                                        period = 12), method="ML")
sar2
```

```
##
## Call:
## arima(x = train.bc, order = c(1, 1, 0), seasonal = list(order = c(0, 1, 0),
##     period = 12), method = "ML")
##
## Coefficients:
##           ar1
##       -0.5171
## s.e.   0.1135
##
## sigma^2 estimated as 0.0004163:  log likelihood = 143.29,  aic = -282.57
```

```
AICc(sar2) # AICc
```

```
## [1] -282.3536
```

```
-0.5171 < -1.96*0.1135 # checking if values are within 95% CI
```

```
## [1] TRUE
```

According to the two models above, the SAR model that produced the smallest AICc was $SARIMA(0,1,0) \times (0,1,0)$. However, because $SARIMA(0,1,0) \times (0,1,0)$ does not have coefficients, we cannot later check to the model for invertibility or stationarity. Thus, we will consider Model 2: $SARIMA(1,1,0) \times (0,1,0)$ as our "lowest AICc" model for comparison.

Next, we try SMA models: $q = 1$ or $4, Q = 0$:

```
# SMA models: q=1 or 4, Q=0
# Model 3
sma1 <- arima(train.bc, order=c(0,1,1), seasonal = list(order = c(0,1,0),
```

```
                                                    period = 12), method="ML")
sma1
```

```
##
## Call:
## arima(x = train.bc, order = c(0, 1, 1), seasonal = list(order = c(0, 1, 0),
##     period = 12), method = "ML")
##
## Coefficients:
##           ma1
##       -0.5847
## s.e.   0.1194
##
## sigma^2 estimated as 0.0004155:  log likelihood = 143.29,  aic = -282.57
```

```
AICc(sma1)
```

```
## [1] -282.3549
```

```
-0.5847 < -1.96*0.1194 # checking if values are within 95% CI
```

```
## [1] TRUE
```

```
# Model 4
sma2 <- arima(train.bc, order=c(0,1,4), seasonal = list(order = c(0,1,0),
                                                    period = 12), method="ML")
sma2
```

```
##
## Call:
## arima(x = train.bc, order = c(0, 1, 4), seasonal = list(order = c(0, 1, 0),
##     period = 12), method = "ML")
##
## Coefficients:
##           ma1     ma2      ma3      ma4
##       -0.5532  0.3412  -0.5206  -0.0393
## s.e.   0.1573  0.1634   0.1428   0.2181
##
## sigma^2 estimated as 0.0003555:  log likelihood = 147.34,  aic = -284.67
```

```
AICc(sma2)
```

```
## [1] -283.5179
```

```
# checking if values are within 95% CI
c(-0.5532,0.3412,-0.5206,-0.0393) < -1.96*c(0.1573,0.1634,0.1428,0.2181)
```

```
## [1]  TRUE FALSE  TRUE FALSE
```

```
AICc(arima(train.bc, order=c(0,1,4), seasonal = list(order = c(0,1,0),
                                                    period = 12),fixed=c(NA,0,NA,0), method="ML"))
```

```
## [1] -284.1219
```

According to the two SMA models, the SARIMA model that produces the lowest AICc between the two is: Model 3: $SARIMA(0,1,1) \times (0,1,0)$.

Thus, the two lowest AICc models we compare are: Model 2's $SARIMA(1,1,0) \times (0,1,0)$ with AICc -282.3536 and Model 3's $SARIMA(0,1,1) \times (0,1,0)$ with AICc -282.3549.

Models

Let these two final models for comparison be:

Model (A): $SARIMA(1,1,0) \times (0,1,0)$

$$(1-\phi_1 B)\nabla_1\nabla_{12}\frac{1}{-0.2626263} * (U_t^{-0.2626263}-1) = Z_t$$

$$= (1+0.4325_{(0.0568)}B)\nabla_1\nabla_{12}\frac{1}{-0.2626263} * (U_t^{-0.2626263}-1) = Z_t$$

Model (B): $SARIMA(0,1,1) \times (0,1,0)$

$$\nabla_1\nabla_{12}\frac{1}{-0.2626263} * (U_t^{-0.2626263}-1) = (1+\theta_1 B)Z_t$$

$$= \nabla_1\nabla_{12}\frac{1}{-0.2626263} * (U_t^{-0.2626263}-1) = (1-0.6747_{(0.0598)}B)Z_t$$

To determine which model is our "best" model, we now need to check for invertibility and stationarity before proceeding to diagnostic checks.

Invertibility and Stationarity

Since Model (A) is pure AR, Model (A) is invertible. With $|\phi_1| = 0.4325 < 1$, Model (A) is also stationary.

For Model (B), since Model (B) is pure MA, Model (B) is stationary. Because $|\theta_1| = 0.6747 < 1$, Model (B) is also invertible.

Diagnostic Checks

Moving forward, we now conduct diagnostic checks by checking/analyzing: histograms, distributions, residuals, residuals^2, normality, PACF/ACF, Shapiro-Wilk test, Ljung Box test, and Box-Pierce testt o determine which of our two models are "best" suited to proceed for forecasting.

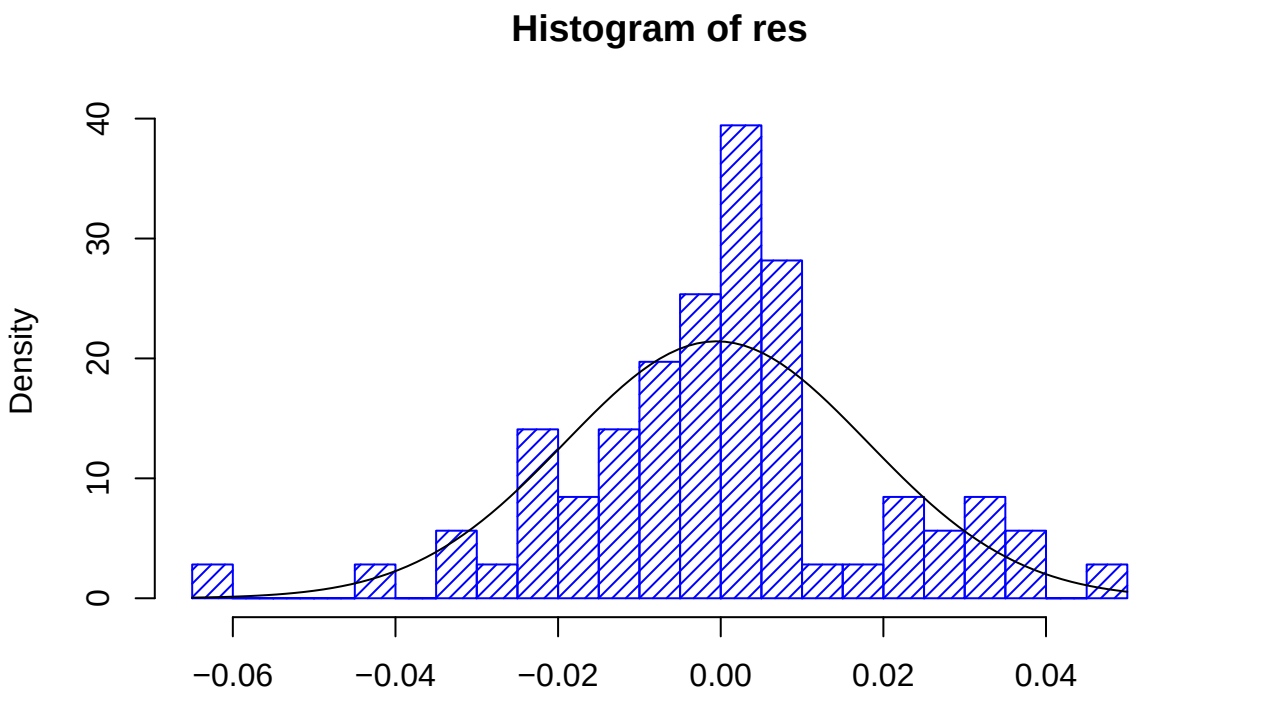
Diagnostic Checking for Model (A):

To begin, we will conduct diagnostic checking with Model (A) first.

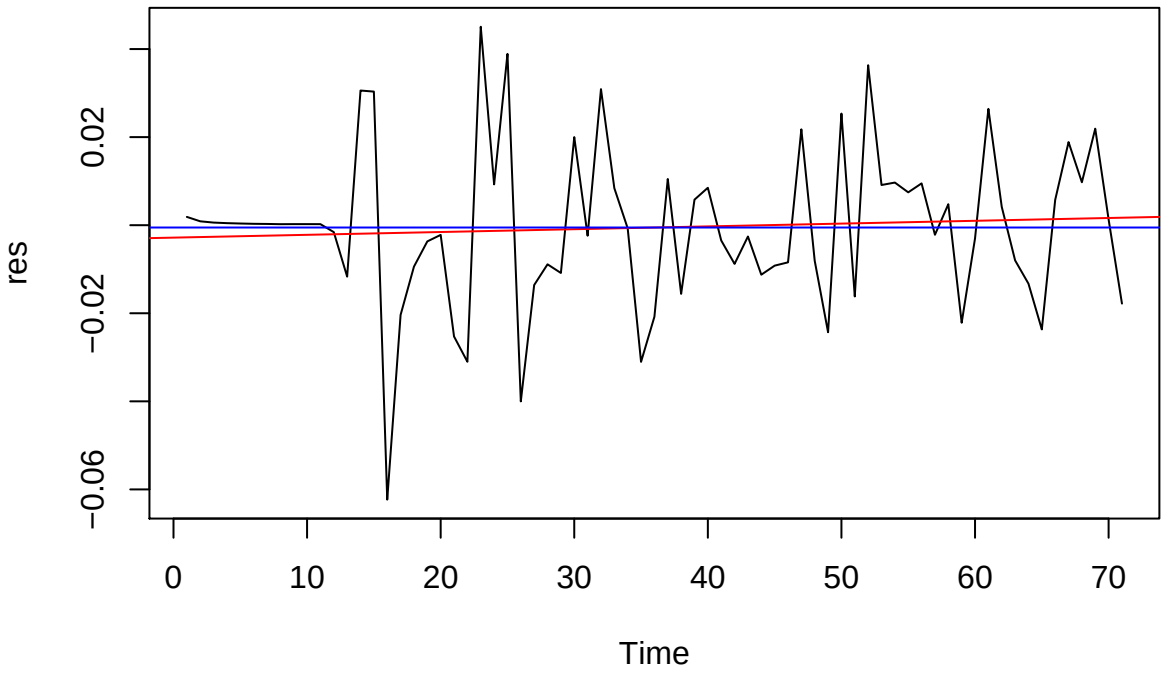
$$(1+0.4325_{(0.0568)}B)\nabla_1\nabla_{12}\frac{1}{-0.2626263} * (U_t^{-0.2626263}-1) = Z_t$$

```
# Model (A)
arima(x = train.bc, order = c(1, 1, 0), seasonal = list(order = c(0, 1, 0),
    period = 12), method = "ML")
```

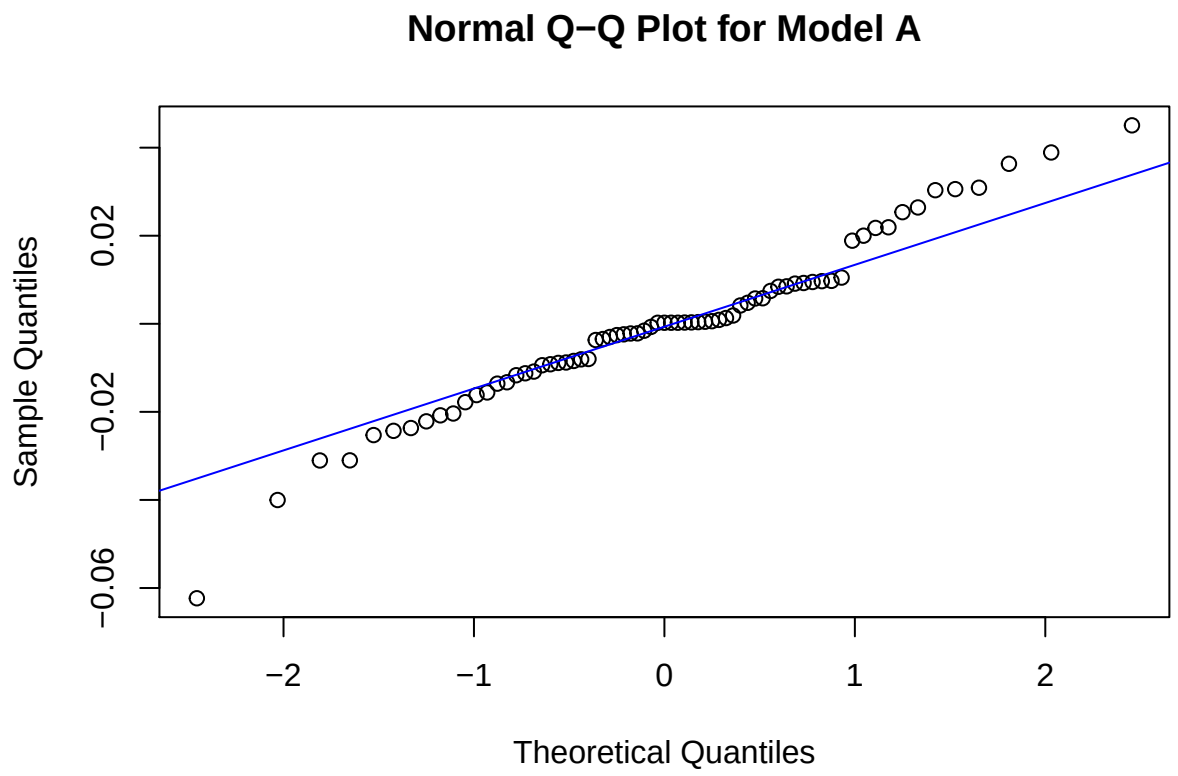
```
##
## Call:
## arima(x = train.bc, order = c(1, 1, 0), seasonal = list(order = c(0, 1, 0),
##     period = 12), method = "ML")
##
## Coefficients:
##           ar1
##       -0.5171
## s.e.   0.1135
##
## sigma^2 estimated as 0.0004163:  log likelihood = 143.29,  aic = -282.57
```



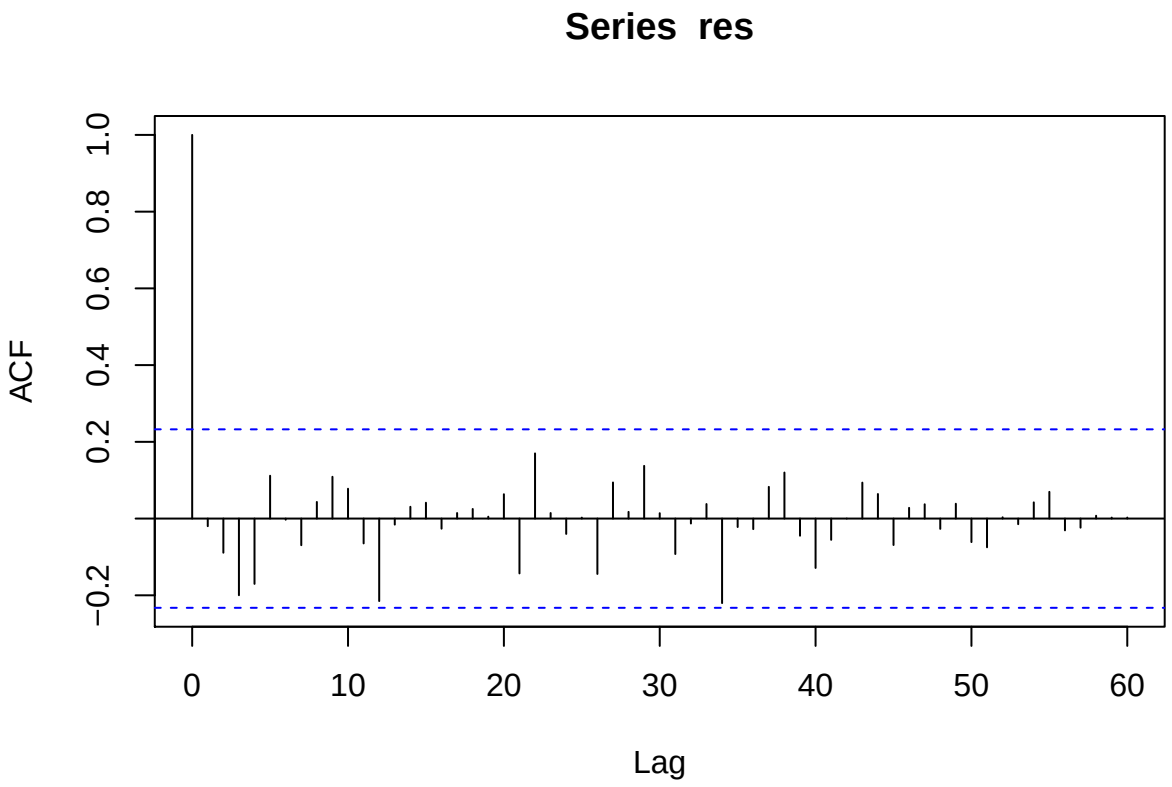

```
plot.ts(res)
fitt <- lm(res ~ as.numeric(1:length(res))); abline(fitt, col="red")
abline(h=mean(res), col="blue")
```

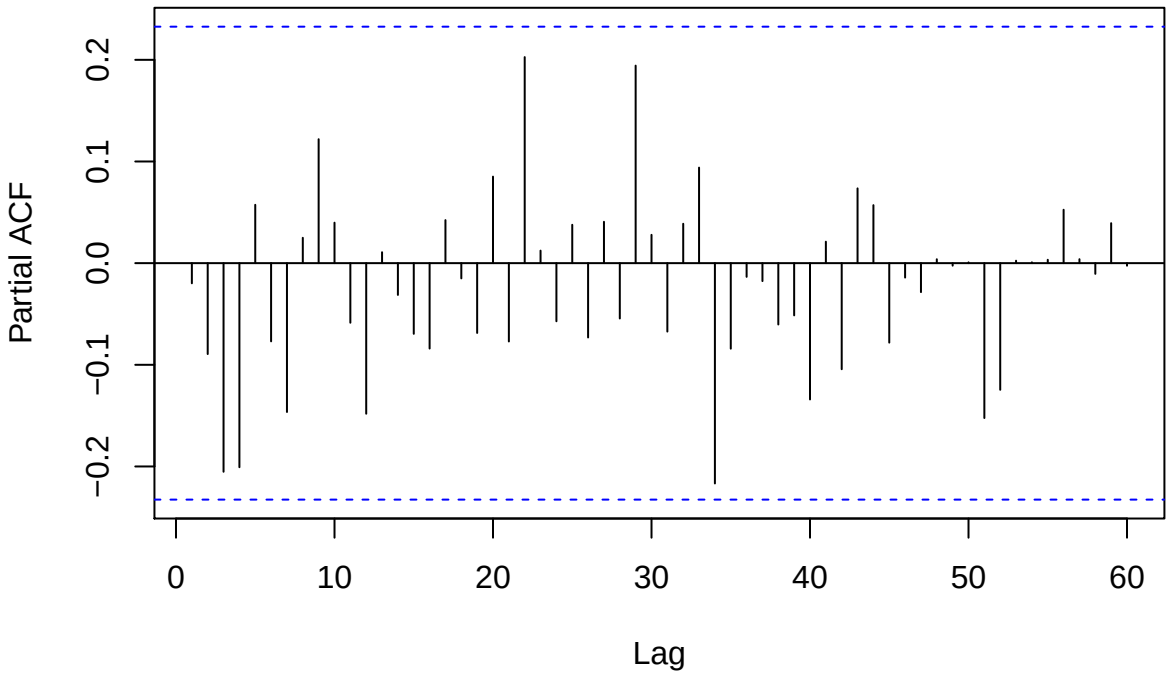



```
# Checking for normal distribution, ACF, PACF
qqnorm(res,main= "Normal Q-Q Plot for Model A")
qqline(res,col="blue")
```



Despite not being perfectly Gaussian, the histogram of the residuals show a relatively normal distribution. The plot of the residuals also show no trend, no visible apparent change in variance, and no seasonality. The sample mean is also almost zero: -0.0005290401. The Q-Q plot also looks okay and appears to follow a relatively normal distribution.





According to the residual ACF and PACF plots, all ACF and PACF appears to be within the 95% confidence interval, and can be counted zeros.

```
# Diagnostic checking tests
shapiro.test(res)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res
## W = 0.9717, p-value = 0.1088
```

```
Box.test(res, lag = sqrt(nt), type = c("Box-Pierce"), fitdf = 1) # nt is 84 (# observations in sales da
```

```
##
##  Box-Pierce test
##
## data:  res
## X-squared = 7.6875, df = 8.1652, p-value = 0.4816
```

```
Box.test(res, lag = sqrt(nt), type = c("Ljung-Box"), fitdf = 1) # nt is 84 (# observations in sales dat
```

```
##
##  Box-Ljung test
##
## data:  res
## X-squared = 8.4268, df = 8.1652, p-value = 0.4093
```

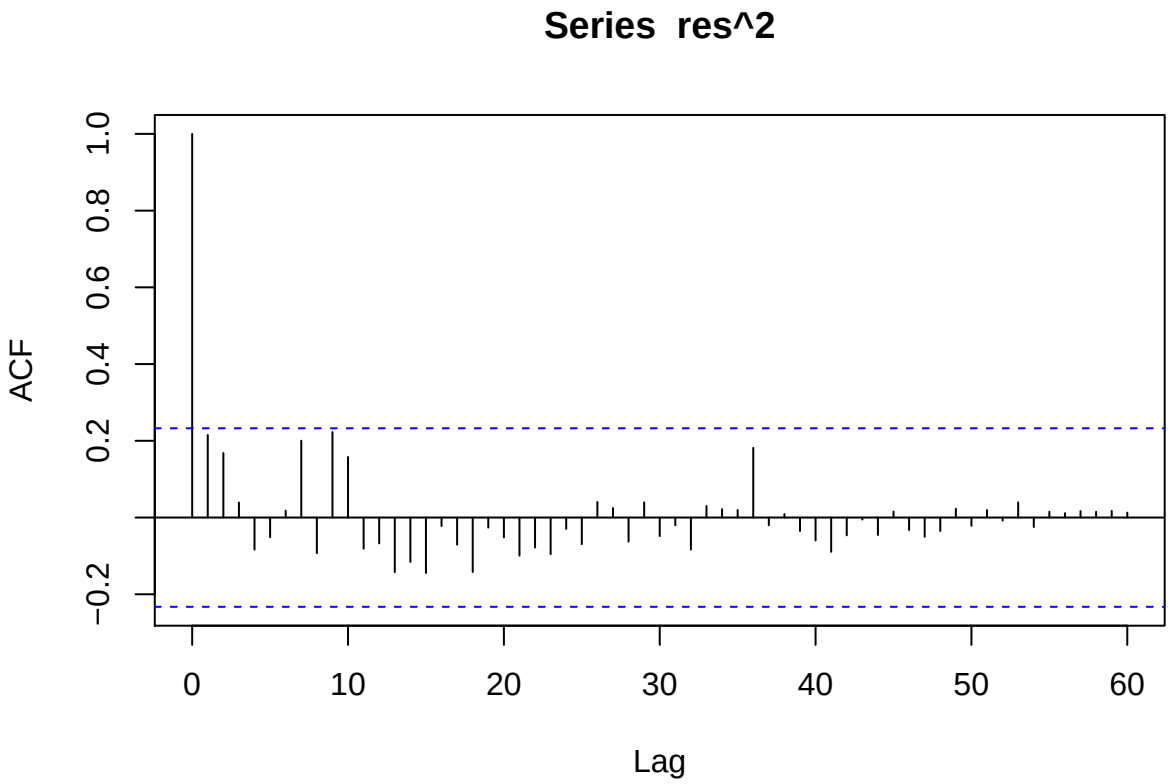
```
Box.test(res^2, lag = sqrt(nt), type = c("Ljung-Box"), fitdf = 0) # nt is 84 (# observations in sales d
```

```
##
##  Box-Ljung test
```

```
##
## data:  res^2
## X-squared = 14.542, df = 9.1652, p-value = 0.1111
```

According to the diagnostic checking tests for Model (A), we get that all p-values are larger than 0.05 which means that the result is statistically significant.

```
# check residuals
acf(res^2, lag.max=60) # ACF of residuals^2; appears to be zero as all ACF are within 95% CI
```




```
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  0.0003467
```

Checking the residuals, all ACF residuals^2 are all within the 95% confidence interval – which can be considered as zero. We also find that the order selected is 0 with σ^2 estimated as 0.0003467 – thus $AR(0)$, i.e. WN! Hence, the Model (A) is satisfactory! Passing diagnostic checking, Model (A) is ready to be used for forecasting.

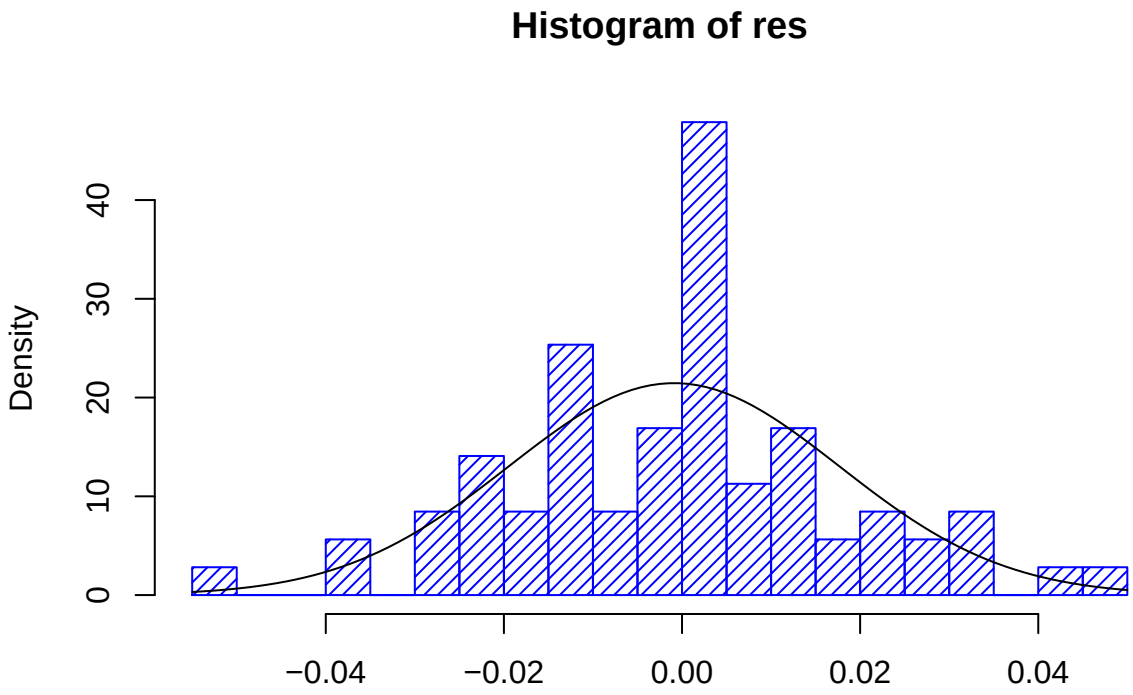
Diagnostic Checking for Model (B):

We now perform diagnostic checking with Model (B):

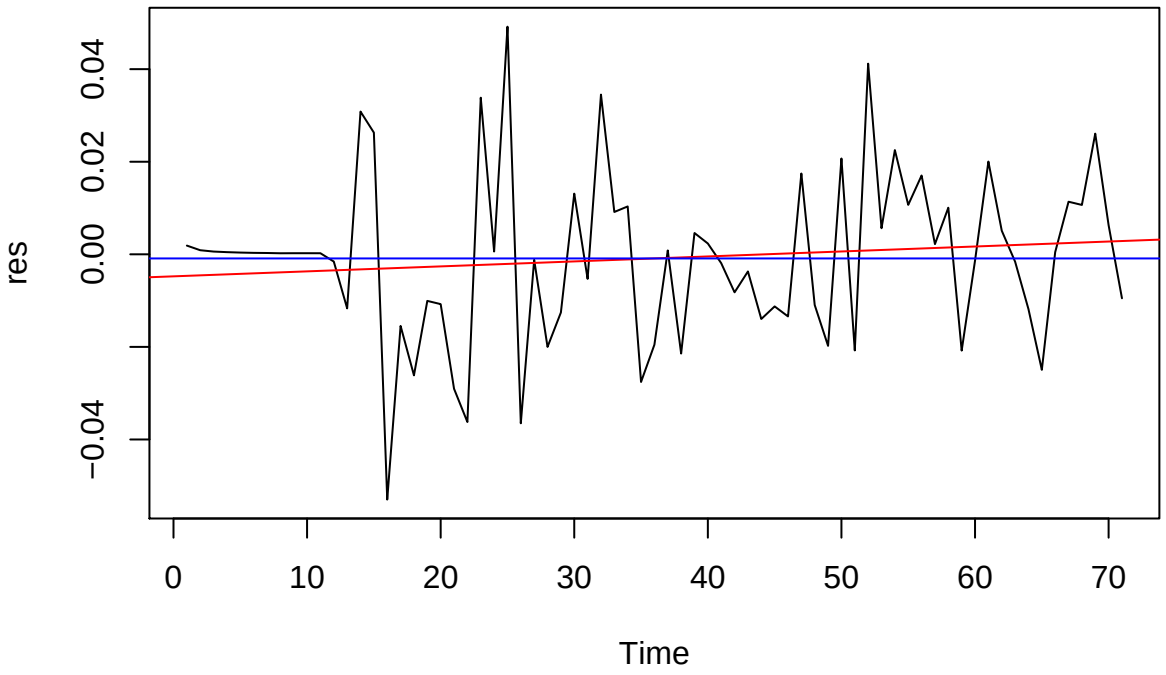
$$\nabla_1\nabla_{12}\frac{1}{-0.2626263} * (U_t^{-0.2626263} - 1) = (1 - 0.6747_{(0.0598)}B)Z_t$$

```r
# Model (B)
arima(x = train.bc, order = c(0, 1, 1), seasonal = list(order = c(0, 1, 0),
    period = 12), method = "ML")
```

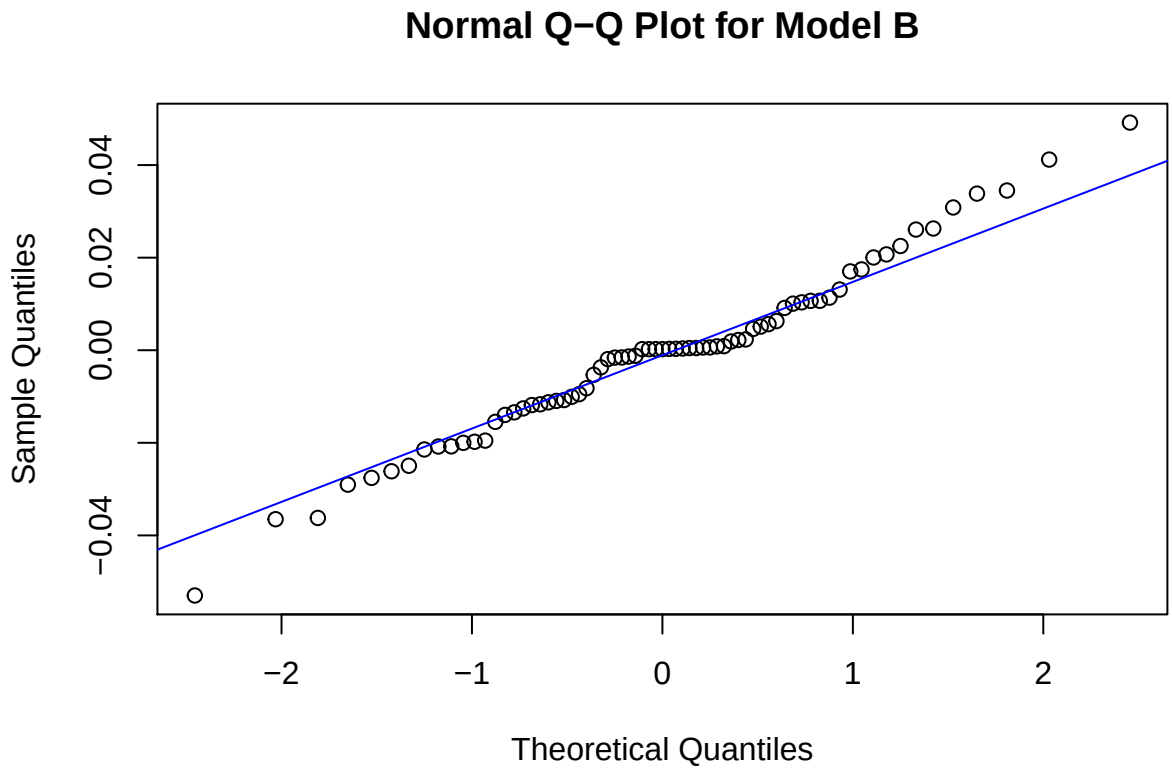
```
##
## Call:
## arima(x = train.bc, order = c(0, 1, 1), seasonal = list(order = c(0, 1, 0),
##     period = 12), method = "ML")
##
## Coefficients:
##           ma1
##       -0.5847
## s.e.   0.1194
##
## sigma^2 estimated as 0.0004155:  log likelihood = 143.29,  aic = -282.57
```




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

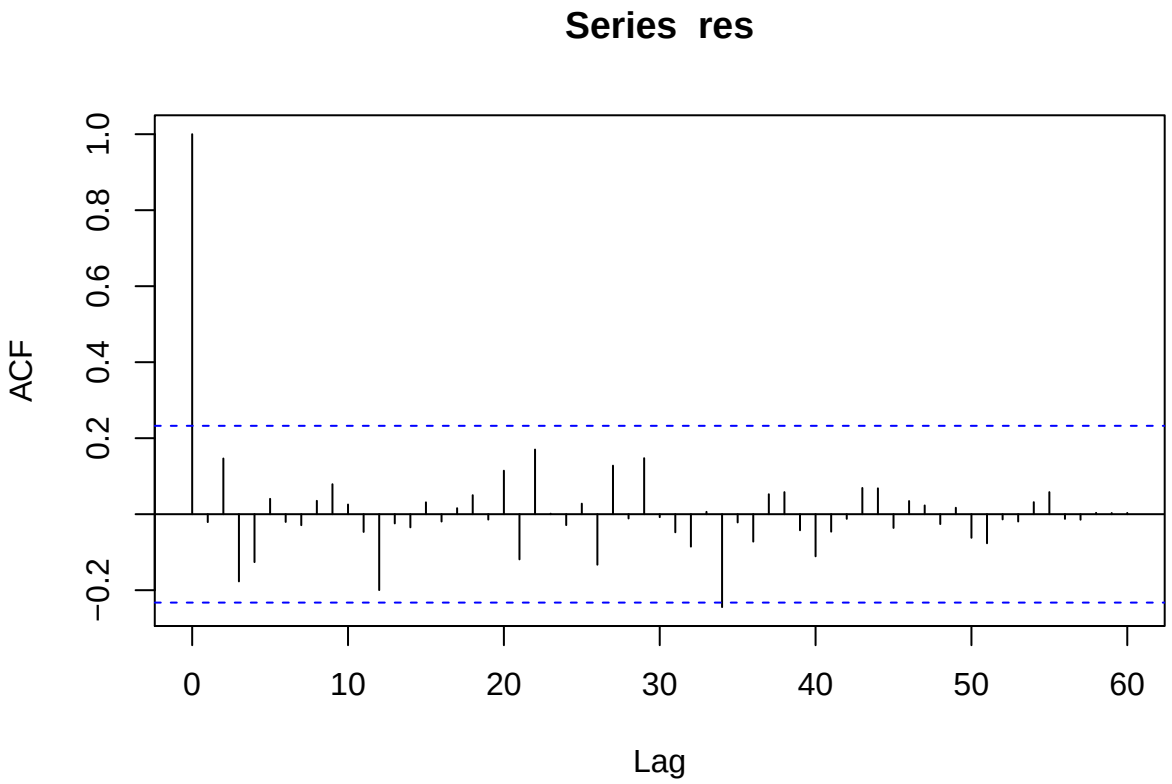



```
# Checking for normal distribution, ACF, PACF
qqnorm(res,main= "Normal Q-Q Plot for Model B")
qqline(res,col="blue")
```

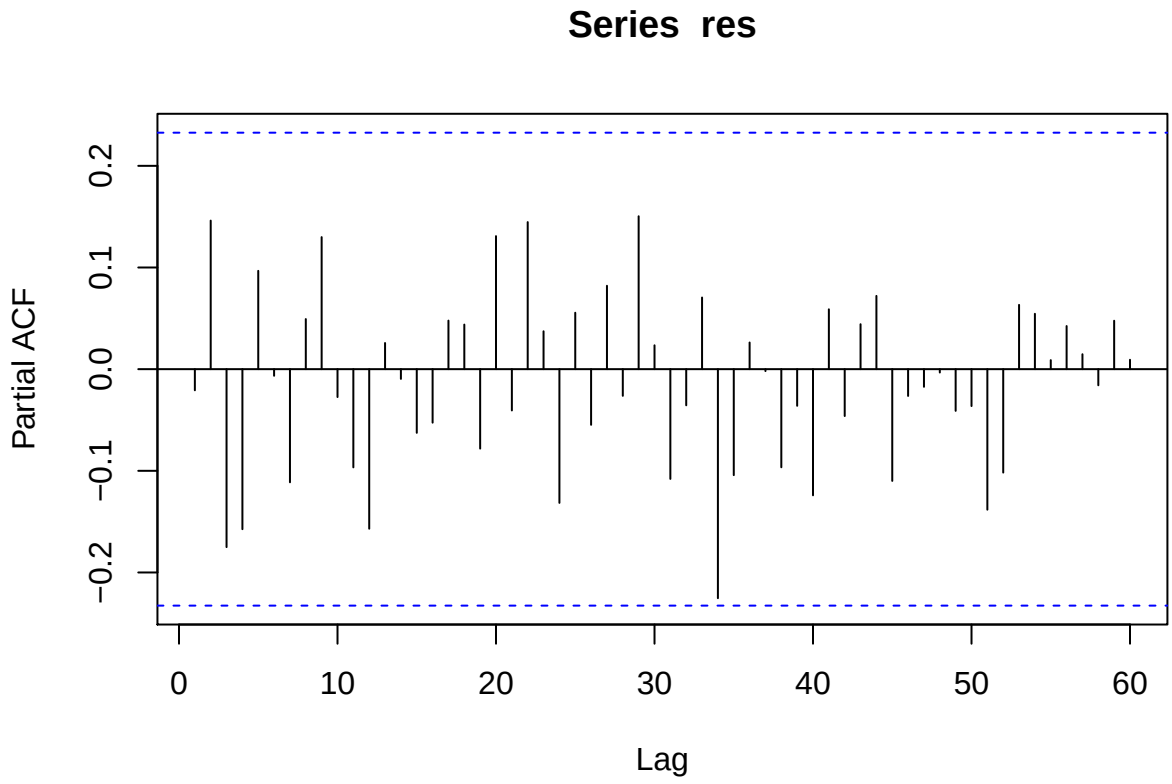


The histogram of the residuals show a relatively normal distribution and appears to be almost Gaussian. The plot of the residuals also showcase no trend, no visible change in variance, and no seasonality. The sample mean is also almost zero: -0.0008906486. The Q-Q plot also somewhat okay and appears to roughly follow normal distribution.

```
acf(res, lag.max=60)
```




```
pacf(res, lag.max=60)
```



Almost all ACF residuals are within the 95% confidence interval except one at lag 34 – which is a bit concerning. The PACF residuals all appear to be within the 95% confidence interval and can be counted as zeros.

```
# Diagnostic checking tests
shapiro.test(res)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res
## W = 0.9817, p-value = 0.3882
```

```
Box.test(res, lag = sqrt(nt), type = c("Box-Pierce"), fitdf = 1) # nt is 84 (# observations in sales da
```

```
##
##  Box-Pierce test
##
## data:  res
## X-squared = 5.6389, df = 8.1652, p-value = 0.7034
```

```
Box.test(res, lag = sqrt(nt), type = c("Ljung-Box"), fitdf = 1) # nt is 84 (# observations in sales dat
```

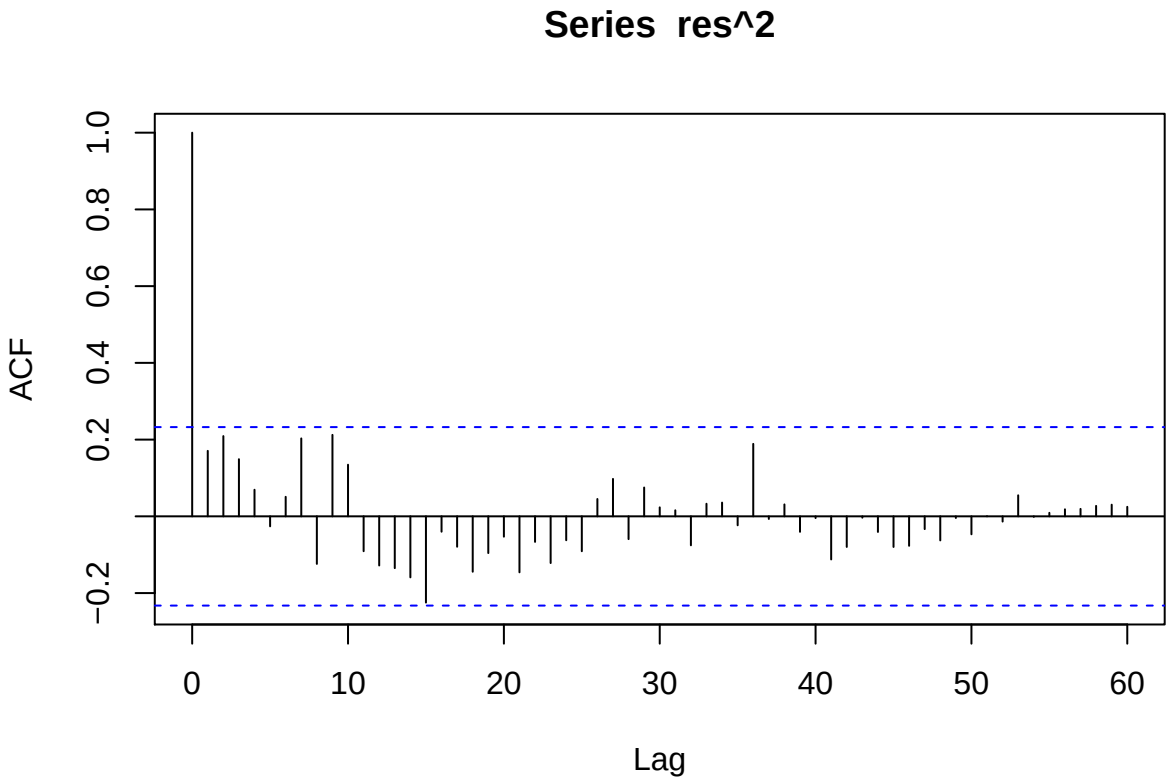
```
##
##  Box-Ljung test
##
## data:  res
## X-squared = 6.1095, df = 8.1652, p-value = 0.6516
```

```
Box.test(res^2, lag = sqrt(nt), type = c("Ljung-Box"), fitdf = 0) # nt is 84 (# observations in sales d
```

```
##
##  Box-Ljung test
##
## data:  res^2
## X-squared = 16.14, df = 9.1652, p-value = 0.06867
```

According to the diagnostic checking tests for Model (B), we get that all p-values are larger than 0.05 which means that the result is statistically significant.

```
# check residuals
acf(res^2, lag.max=60)
```




```
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  0.0003455
```

Checking the residuals, all ACF $residuals^2$ are all within the 95% confidence interval – which can be considered as zero. We also find that the order selected is 0 with σ^2 estimated as 0.0003455 – thus $MA(0)$. However, even though Model (B) passes the diagnostic check tests, I do have some reservations about Model (B) as there is a single point in ACF residuals at lag 34 that passes the 95% confidence interval. Thus, I do not find Model (B) satisfactory.

For this reason, we proceed with Model (A) as the final model for forecasting. (Note: Model (A) also has a lower AICc of -282.3536 compared to Model (B)'s AICc of -282.3549. Hence, choosing Model (A) would also be "best" as the best model minimizes the AICc.)

Final fitted model: $SARIMA(1,1,0) \times (0,1,0)$

$$(1 + 0.4325_{(0.0568)}B)\nabla_1\nabla_{12}\frac{1}{-0.2626263} * (U_t^{-0.2626263} - 1) = Z_t$$

Forecasting

Using Model (A):

$$(1 + 0.4325_{(0.0568)}B)\nabla_1\nabla_{12}\frac{1}{-0.2626263} * (U_t^{-0.2626263} - 1) = Z_t$$

```
## 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
```

```
##
## Attaching package: 'forecast'
```

```
## The following object is masked from 'package:astsa':
##
##     gas
```

Forecast of Box-Cox transformed data using Model (A)

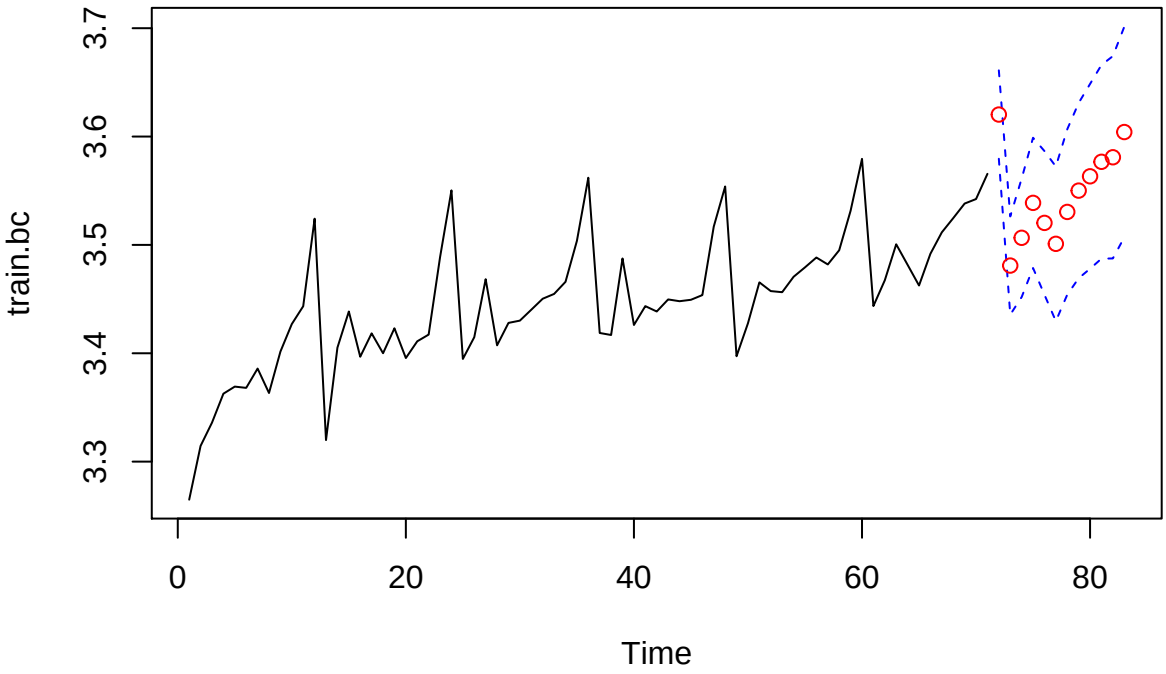
```
# Forecasting using model (A)
fit.A <- arima(train.bc, order=c(1,1,0), seasonal = list(order = c(0,1,0),
                                                         period = 12), method="ML")
forecast(fit.A)  # prints forecasts with prediction bounds in a table
```

```
##    Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
## 72       3.620299 3.594152 3.646447 3.580310 3.660289
## 73       3.480937 3.451900 3.509975 3.436529 3.525346
## 74       3.506552 3.471509 3.541596 3.452958 3.560146
## 75       3.538800 3.500275 3.577325 3.479881 3.597719
## 76       3.520313 3.477844 3.562782 3.455362 3.585264
## 77       3.500991 3.455280 3.546701 3.431083 3.570899
## 78       3.530419 3.481506 3.579331 3.455614 3.605224
## 79       3.550123 3.498292 3.601954 3.470854 3.629392
## 80       3.563294 3.508658 3.617930 3.479735 3.646852
## 81       3.576704 3.519421 3.633987 3.489097 3.664311
## 82       3.580867 3.521044 3.640691 3.489375 3.672360
## 83       3.604156 3.541901 3.666411 3.508945 3.699367
## 84       3.658856 3.582974 3.734739 3.542804 3.774909
## 85       3.519493 3.437946 3.601040 3.394778 3.644208
## 86       3.545109 3.455620 3.634597 3.408248 3.681969
## 87       3.577356 3.481894 3.672818 3.431359 3.723353
## 88       3.558869 3.457156 3.660583 3.403312 3.714427
## 89       3.539547 3.432256 3.646838 3.375460 3.703635
## 90       3.568975 3.456230 3.681720 3.396546 3.741404
## 91       3.588680 3.470808 3.706551 3.408411 3.768948
## 92       3.601850 3.479028 3.724671 3.414011 3.789689
## 93       3.615261 3.487700 3.742821 3.420173 3.810348
## 94       3.619423 3.487284 3.751563 3.417333 3.821514
```

```
## 95       3.642712 3.506152 3.779273 3.433861 3.851564
```

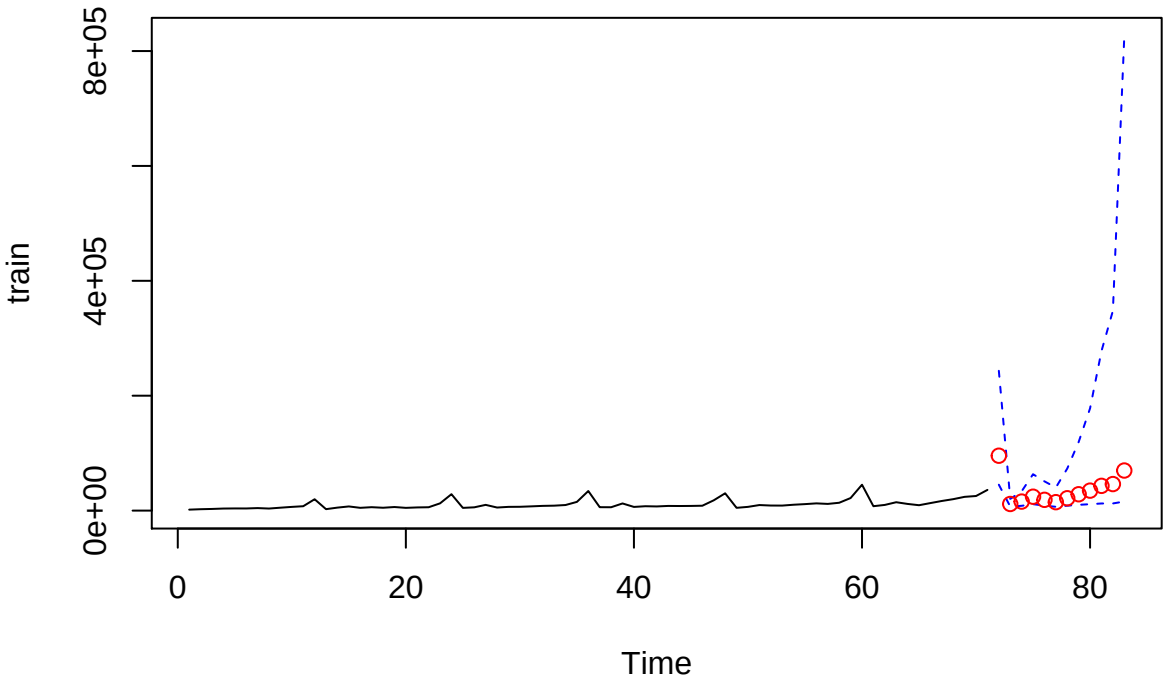
KEY: In subsequent plots: the blue dotted line denotes the 95% confidence interval and the red circles denote forecast values.

Producing the plot with 12 forecasts on transformed data



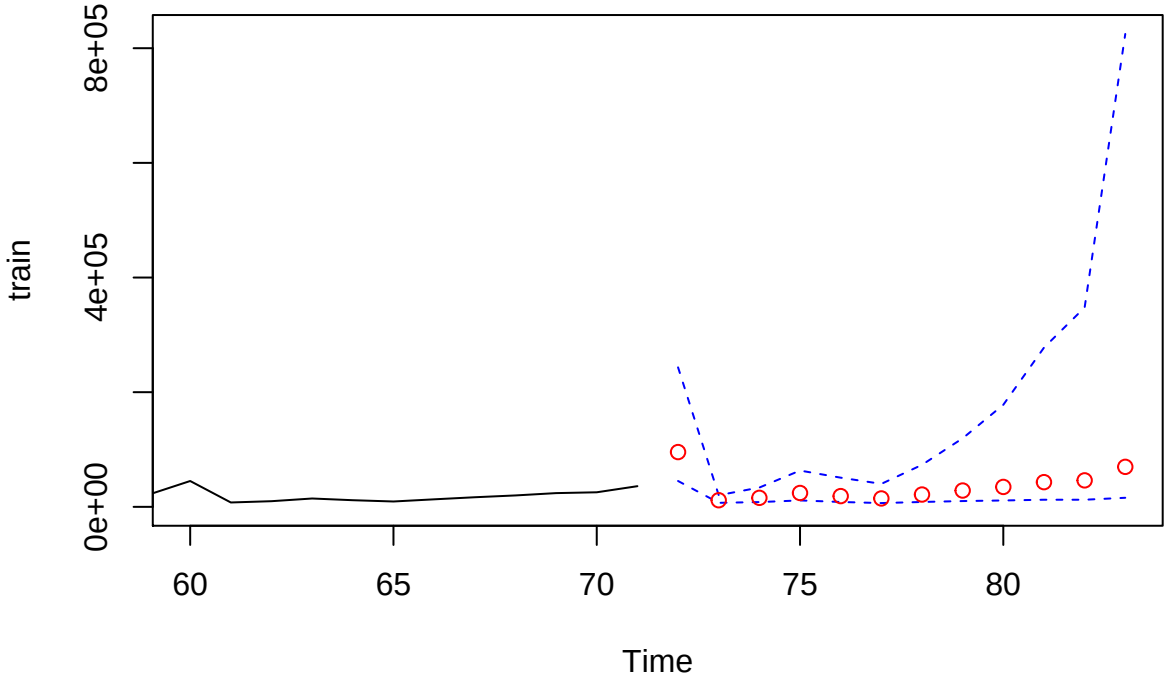
In this plot, the 12 forecasted values are plotted on the Box-Cox transformed data.

Producing the plot with forecasts on original data

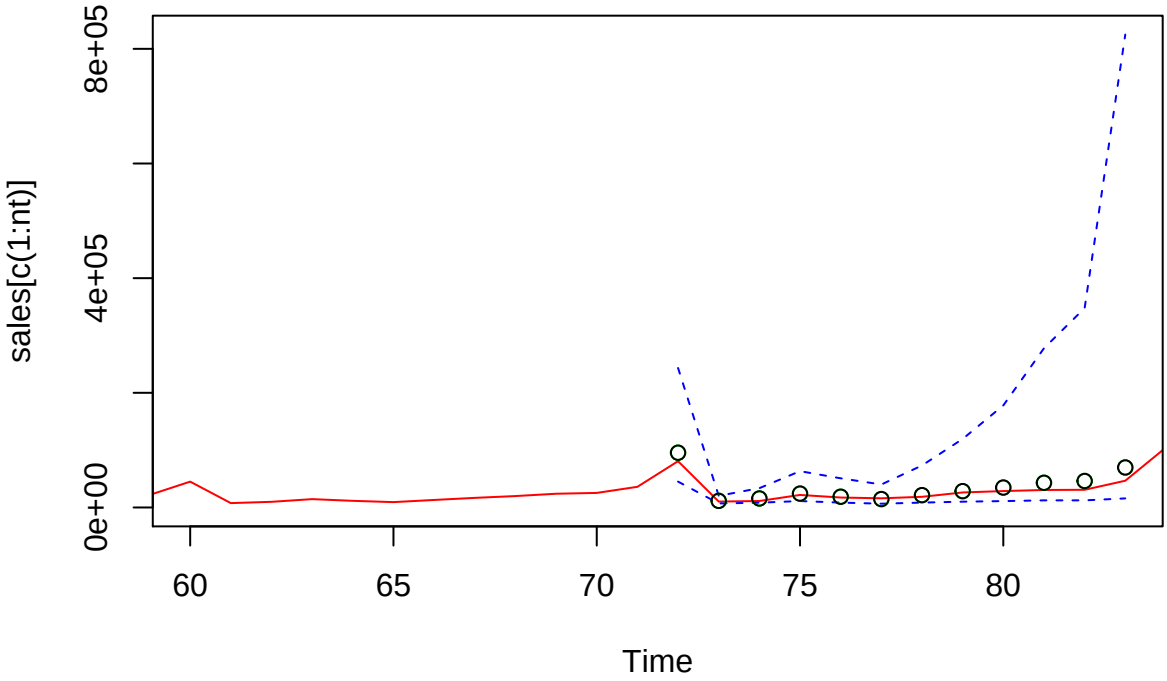


In this plot, the 12 forecasted values are plotted on the original partitioned sales dataset where values from January 1993 - December 1993 are not present.

Zoom the graph, starting from entry 60:



Plot zoomed forecasts and true values (in sales):



In this plot, the forecast values of monthly sales in 1993 from the final model $SARIMA(1,1,0) \times (0,1,0)$ are hollow black circles. The red line denotes the actual values from the original sales data dataset, prior to partitioning. Seeing that the red line appears to pass through nearly all of the black circles and are within

the 95% confidence interval (denoted by the blue dotted line), we can conclude that the final model appears to be generally accurate at predicting future sales.

Conclusion

To recap, the goal of this project was to construct a SARIMA model that would help predict future sales for a souvenir shop on the wharf at a beach resort town in Queensland, Australia. This project goal was achieved through a $SARIMA(1,1,0) \times (0,1,0)$ model:

$$(1 + 0.4325_{(0.0568)}B)\nabla_1\nabla_{12}\frac{1}{-0.2626263} * (U_t^{-0.2626263} - 1) = Z_t$$

According to the forecast plots in the previous section titled *Forecasting*, our final model proved to be generally accurate as most forecast points appear to plot closely with actual values from the original data. Hence, it may be safe to say that the model may be used for future sales forecasting for this souvenir shop.

Last but not least, I would like to give a big, special thanks to Professor Raya Feldman, TAs Jasmine Li and Sunpeng Duan. Without their help, this project would not have been possible. Thank you!!

References

Feldman, Raya. PSTAT174 Lectures 1-15. PSTAT174 Time Series. N.p., Fall 2021. Web.

Hyndman, Rob J, and Yangzhuoran Yang. "Tsdl: Time Series Data Library". Tsdl, 2018, https://pkg.yangzhuoranyang.com/tsdl/.

Appendix

```r
# R libraries used
library(tsdl)
library(astsa)
library(MASS)
library(MuMIn)

# time series data -- the dataset
sales <- subset(tsdl, 12, "Sales")[[12]]
sales

plot.ts(sales)                      # plot data
nt <- length(sales)
fit <- lm(sales ~ as.numeric(1:nt))
abline(fit, col="red")              # added trend to data plot
mean(sales)[1]
abline(h=mean(sales), col="blue")   # added mean (constant) to data plot)

# partition data set for model training and model validation
train = sales[1:71]       # training dataset
testing = sales[72:84]    # test dataset
plot.ts(train)
fit <- lm(train ~ as.numeric(1:length(train)))
abline(fit, col="red")
abline(h=mean(sales), col="blue")
```

```r
hist(sales, col="light blue", xlab="",
     main="histogram; sales data") # plots histogram

acf(train, lag.max=40, main="ACF of the sales (train) data") # plots ACF

# Box-Cox test for data transformation
bcTransform <- boxcox(train~ as.numeric(1:length(train))) # plots the graph
bcTransform$x[which(bcTransform$y == max(bcTransform$y))] # gives the value of lambda
lambda=bcTransform$x[which(bcTransform$y == max(bcTransform$y))] # store lambda value

# Perform transformations, plot transformed data, histograms:
lambda=bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
train.bc = (1/lambda)*(train^lambda-1)
train.log <- log(train)

plot.ts(train.log) # plot log-transformed data
plot.ts(train.bc) # plot box-cox transformed data

hist(train.log, col="light blue", xlab="", main="histogram; bc(U_t)") # plot histogram of log-transform
hist(train.bc, col="light blue", xlab="", main="histogram; bc(U_t)") # plot histogram of box-cox transf

# To produce decomposition of bc(U_t):
#install.packages('ggplot2')
#install.packages('ggfortify')
library(ggplot2)
library(ggfortify)

# choose bc transformation
y <- ts(as.ts(train.bc), frequency = 12)
decomp <- decompose(y)
plot(decomp)
# Differencing before
plot.ts(train.bc, main = "bc(U_t)")  # plot before differencing, only transformed
var(train.bc) [1]  # variance before differencing

# Differencing
train.bc_12 <- diff(train.bc, lag=12)    # difference at lag 12
plot.ts(train.bc_12, main="bc(U_t) differenced at lag 12")
var(train.bc_12) [1]        # variance after differencing at lag 12
fit <- lm(train.bc_12 ~ as.numeric(1:length(train.bc_12)))
abline(fit, col="red")
#mean(train.bc_12) [1]        # mean after differencing at lag 12
abline(h=mean(train.bc_12), col="blue")

train.stat <- diff(train.bc_12, lag=1)    # difference again, but at lag 1
plot.ts(train.stat, main="bc(U_t) differenced at lag 12 & lag 1")  # plot after differencing
fit <- lm(train.stat ~ as.numeric(1:length(train.stat)))
abline(fit, col="red")
#mean(train.stat) [1]     # mean after differencing at lag 12 and 1
abline(h=mean(train.stat), col="blue")
var(train.stat) [1]      # variance after differencing at lag 12 and 1

# train.stat is bc transformed truncated data, differenced at lags 12 and then 1.
```

```r
# ACF
acf(train.bc, lag.max=60, main="ACF of the bc(U_t)")
acf(train.bc_12, lag.max=60,
    main="ACF of the bc(U_t), differenced at lag 12")
acf(train.stat, lag.max=60,
    main="ACF of the bc(U_t), differenced at lags 12 and 1")

# PACF
pacf(train.bc, lag.max=60, main="PACF of the bc(U_t)")
pacf(train.bc_12, lag.max=60, main="PACF of the bc(U_t), differenced at lag 12 ")
pacf(train.stat, lag.max=60, main="PACF of the bc(U_t), differenced at lags 12 and 1")

# histograms of without differencing and with differencing at lag 12 and 1
hist(train.bc, col="light blue", xlab="", main="histogram; bc(U_t)")
hist(train.stat, col="light blue", xlab="", main="histogram; bc(U_t) differenced at lags 12 & 1")
# Histogram of transformed and differenced data with normal curve:
hist(train.stat, density=20,breaks=20, col="blue", xlab="", prob=TRUE)
m <- mean(train.stat)
std <- sqrt(var(train.stat))
curve( dnorm(x,m,std), add=TRUE )

# SAR models: p=0 or 1, P=0
# Model 1
sar1 <- arima(train.bc, order=c(0,1,0), seasonal = list(order = c(0,1,0),
                                                        period = 12), method="ML")
sar1
AICc(sar1) # AICc

# Model 2
sar2 <- arima(train.bc, order=c(1,1,0), seasonal = list(order = c(0,1,0),
                                                        period = 12), method="ML")
sar2
AICc(sar2) # AICc

-0.5171 < -1.96*0.1135 # checking if values are within 95% CI

# SMA models: q=1 or 4, Q=0
# Model 3
sma1 <- arima(train.bc, order=c(0,1,1), seasonal = list(order = c(0,1,0),
                                                        period = 12), method="ML")
sma1
AICc(sma1)

-0.5847 < -1.96*0.1194 # checking if values are within 95% CI

# Model 4
sma2 <- arima(train.bc, order=c(0,1,4), seasonal = list(order = c(0,1,0),
                                                        period = 12), method="ML")
sma2
AICc(sma2)

# checking if values are within 95% CI
c(-0.5532,0.3412,-0.5206,-0.0393) < -1.96*c(0.1573,0.1634,0.1428,0.2181)
```

```r
AICc(arima(train.bc, order=c(0,1,4), seasonal = list(order = c(0,1,0),
                                                     period = 12),fixed=c(NA,0,NA,0), method="ML"))

# Model (A)
arima(x = train.bc, order = c(1, 1, 0), seasonal = list(order = c(0, 1, 0),
    period = 12), method = "ML")
# Check histogram, residuals
fit <- arima(train.bc, order=c(1,1,0), seasonal = list(order = c(0,1,0), period = 12), method="ML")
res <- residuals(fit)
hist(res,density=20,breaks=20, col="blue", xlab="", prob=TRUE)
m <- mean(res)
std <- sqrt(var(res))
curve( dnorm(x,m,std), add=TRUE )
plot.ts(res)
fitt <- lm(res ~ as.numeric(1:length(res))); abline(fitt, col="red")
abline(h=mean(res), col="blue")

# Checking for normal distribution, ACF, PACF
qqnorm(res,main= "Normal Q-Q Plot for Model A")
qqline(res,col="blue")
acf(res, lag.max=60) # ACF of residuals
pacf(res, lag.max=60) # PACF of residuals

# Diagnostic checking tests
shapiro.test(res)
Box.test(res, lag = sqrt(nt), type = c("Box-Pierce"), fitdf = 1) # nt is 84 (# observations in sales da
Box.test(res, lag = sqrt(nt), type = c("Ljung-Box"), fitdf = 1) # nt is 84 (# observations in sales dat
Box.test(res^2, lag = sqrt(nt), type = c("Ljung-Box"), fitdf = 0) # nt is 84 (# observations in sales d

# check residuals
acf(res^2, lag.max=60) # ACF of residuals^2; appears to be zero as all ACF are within 95% CI
ar(res, aic = TRUE, order.max = NULL, method = c("yule-walker"))

# Model (B)
arima(x = train.bc, order = c(0, 1, 1), seasonal = list(order = c(0, 1, 0),
    period = 12), method = "ML")
# Check histogram, residuals
fit <- arima(train.bc, order=c(0,1,1), seasonal = list(order = c(0,1,0), period = 12), method="ML")
res <- residuals(fit)
hist(res,density=20,breaks=20, col="blue", xlab="", prob=TRUE)
m <- mean(res)
std <- sqrt(var(res))
curve( dnorm(x,m,std), add=TRUE )
plot.ts(res)
fitt <- lm(res ~ as.numeric(1:length(res))); abline(fitt, col="red")
abline(h=mean(res), col="blue")

# Checking for normal distribution, ACF, PACF
qqnorm(res,main= "Normal Q-Q Plot for Model B")
qqline(res,col="blue")
acf(res, lag.max=60)
pacf(res, lag.max=60)
```

```r
# Diagnostic checking tests
shapiro.test(res)
Box.test(res, lag = sqrt(nt), type = c("Box-Pierce"), fitdf = 1) # nt is 84 (# observations in sales da
Box.test(res, lag = sqrt(nt), type = c("Ljung-Box"), fitdf = 1) # nt is 84 (# observations in sales dat
Box.test(res^2, lag = sqrt(nt), type = c("Ljung-Box"), fitdf = 0) # nt is 84 (# observations in sales d

# check residuals
acf(res^2, lag.max=60)
ar(res, aic = TRUE, order.max = NULL, method = c("yule-walker"))

# Forecasting
library(forecast)
# Forecasting using model (A)
fit.A <- arima(train.bc, order=c(1,1,0), seasonal = list(order = c(0,1,0),
                                                         period = 12), method="ML")
forecast(fit.A)  # prints forecasts with prediction bounds in a table

# To produce graph with 12 forecasts on transformed data:
pred.tr <- predict(fit.A, n.ahead = 12)
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(train.bc, xlim=c(1,length(train.bc)+12),
        ylim = c(min(train.bc),max(U.tr)))
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(train.bc)+1):(length(train.bc)+12),
       pred.tr$pred, col="red")

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

# To zoom the graph, starting from entry 60:
ts.plot(train, xlim = c(60,length(train)+12), ylim = c(0,max(U)))
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(train)+1):(length(train)+12), pred.orig, col="red")

# To plot zoomed forecasts and true values (in sales):
plot.ts(sales[c(1:nt)], xlim = c(60,length(train)+12), ylim = c(0,max(U)), col="red")
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(train)+1):(length(train)+12), pred.orig, col="green")
points((length(train)+1):(length(train)+12), pred.orig, col="black")
```