PSTAT 274 Final Project

Yang Hu

2023-12-09

Abstract

In this project, we analyzed the time series data of candy production in the United States with the objective of developing a reliable model for future forecasting. The dataset was divided, allocating 90% for training purposes and the remaining 10% for validation. During the training phase, a Box-Cox transformation and differencing at lags 12 and 1 were applied to the data to achieve stationarity.

We analyzed the ACF and PACF plots of the transformed data set and identified suitable SARIMA model candidates. We compared the candidates with maximum likelihood estimation and found the best fit based on its lowest AICc values: a SARIMA(3,1,4)(1,1,1)[12] model. This model was both stationary, invertible, and passed all diagnostic checking tests. By spectral analysis, we showed that the time series has a period of 12 month and the residuals could be seen as a white noise process.

Forecasting was then performed on the latter 10% of the data using this model. The forecast included both predicted values and 95% prediction confidence intervals. The confidence intervals covered all the actual values in the test set, indicating the success of the model. This successful outcome highlights the model's potential as a reliable tool for forecasting future trends in candy production.

1 Introduction

1.1 Motivation and Objectives:

The cyclical nature of candy consumption in the United States, peaking around major holidays like Halloween and Christmas, presents a unique opportunity for time series analysis. This project aims to model the candy production in the US using monthly data to understand production patterns and forecast future demands.This analysis could be beneficial for stakeholders in the candy industry for planning and decision-making purposes.

1.2 Data set description

The data set studied in the project is the "U.S. Candy Production by Month" data set form Kaggle. Spanning from January 1972 to August 2017, it tracks monthly candy production in the United States and offers a detailed view of over 45 years of cnady production amounts.

```
library(dplyr)

raw_data <- read.csv('candy_production.csv') %>%
  mutate(Date = as.Date(observation_date, format = "%Y-%m-%d"), Production = IPG3113N) %>%
  dplyr::select(Date, Production) %>%
  arrange(Date)
```

2 Data Cleaning

2.1 Initial checking

```
null_values <- sapply(raw_data, function(x) sum(is.na(x)))
print("Null Values in Each Column:")
```

```
## [1] "Null Values in Each Column:"
```

```
print(null_values)
```

```
##       Date Production
##          0          0
```

```
duplicates <- raw_data[duplicated(raw_data),]
print("Duplicate Rows:")
```

```
## [1] "Duplicate Rows:"
```

```
print(duplicates)
```

```
## [1] Date       Production
## <0 rows> (or 0-length row.names)
```

We checked that the data set contains no Null values nor duplicate values.

2.2 Data set split

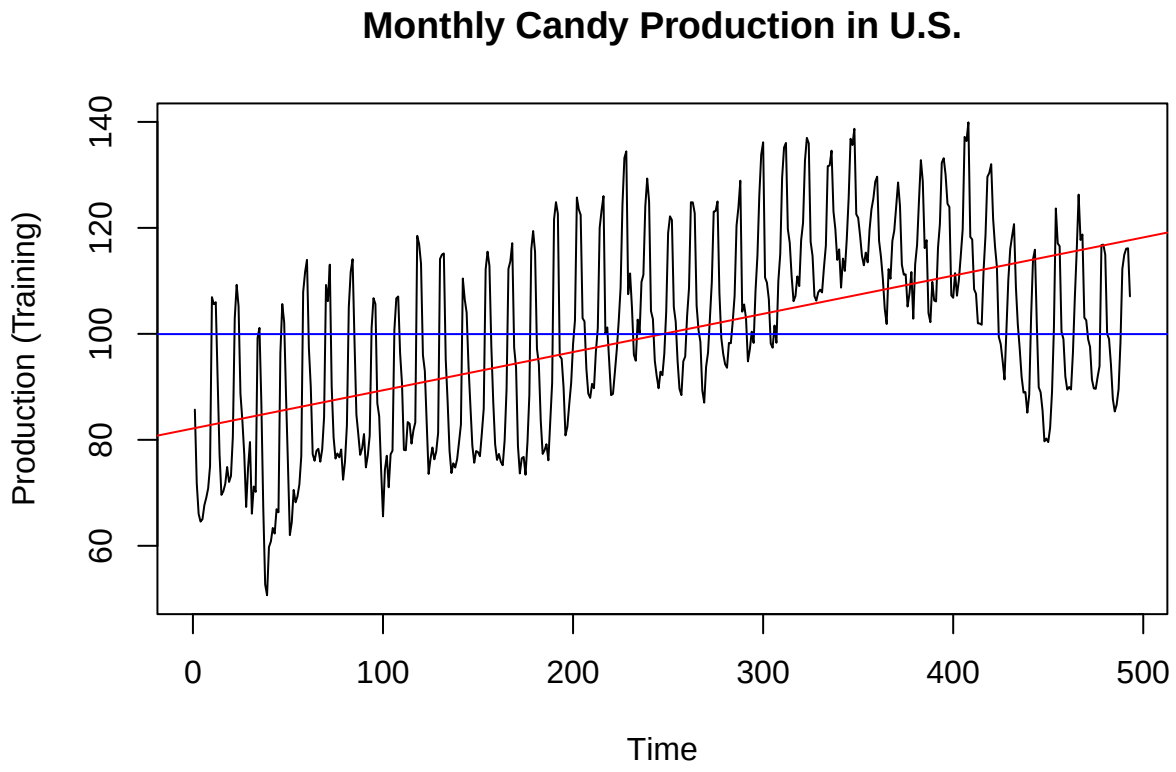
The data set consists of 548 observations spread over the 42 years from 1972-01-01 to 2017-08-01, and we denote the original truncated data as $\{U_t, t = 1, 2, ..., 548\}$. We apply a 90-10 training-test split, building model based on the first 493 observations, leaving 55 observations for validation.

```
start_year <- format(min(raw_data$Date), "%Y")
start_month <- format(min(raw_data$Date), "%m")

cp.ts <- ts(raw_data$Production, start = c(as.numeric(start_year), as.numeric(start_month)), frequency =


# Calculate the split index with first 90% of the data for training
split_index <- round(length(cp.ts) * 0.9)

# Split into training and test set
cp_train = cp.ts[c(1: split_index)]
cp_test= cp.ts[c(split_index+1:length(cp.ts))]
```

2.3 Initial anaylsis on the original training data set

2.3.1 The original training data

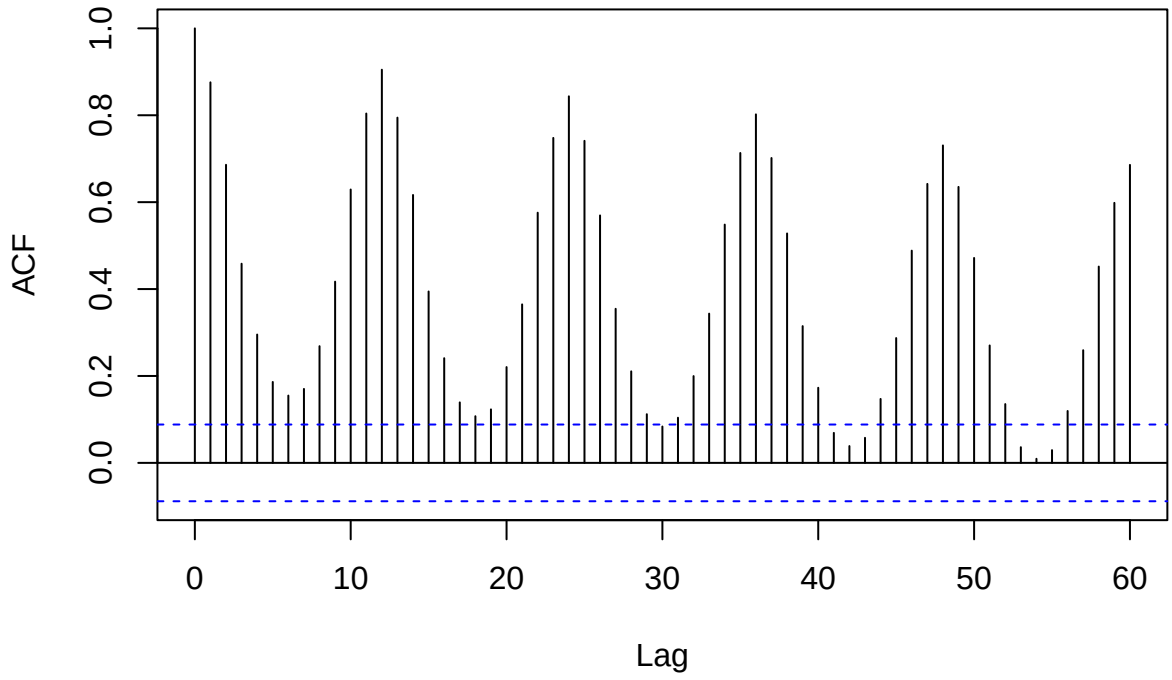
```r
plot(cp_train, main="Monthly Candy Production in U.S.", ylab="Production (Training)", xlab="Time", type=

#Add mean and trend lines
ntr=length(as.numeric(cp_train))
fit_train <- lm(as.numeric(cp_train) ~ as.numeric(1:ntr))
abline(fit_train, col="red")
abline(h=mean(as.numeric(cp_train)), col="blue")
```



We plot and analyze the training set time series and find that: -(i) Trend : There is a long-term increase in production levels. The general movement is upward, particularly noticeable from the early 1970s until the early 2000s. -(ii) Seasonality: The graph exhibits a clear seasonal pattern, with peaks and troughs occurring regularly each year. The peaks likely correspond to periods of high demand, such as Halloween and the end-of-year holiday season, which typically see increased candy consumption.
-(iii) Sharp changes: There are no obvious sharp, sudden shifts in production.

2.3.2 The acf plot of the original training data

```r
acf(cp_train,lag.max = 60,main = "")
```



From the above plot, Acfs remain large and periodic, indicating non-stationarity,

In conclusion, U_t (train) is not stationary and we need to apply transformation and differencing methods to make it stationary for later time series analysis.

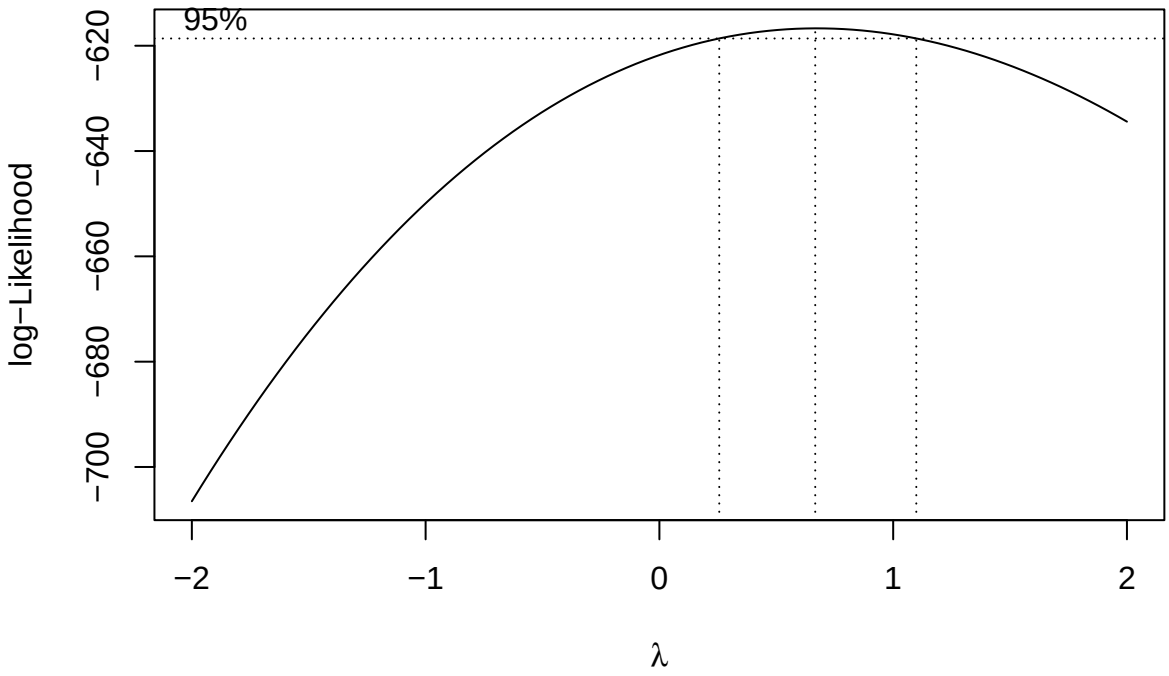
2.4 Transformations

In order to get a stationary time series data, we perform the following transformation.

2.4.1 Box-Cox transformation

We first apply the Box-Cox transformation to stabilize the variance.

```
library(MASS)
bcTransform <- boxcox(as.numeric(cp_train)~as.numeric(1:length(cp_train)))
```



The dashed vertical lines in the plot above correspond to a 95% confidence interval for the true value of λ in the Box-Cox transformation. From the plot, the best λ to maximize the log-likelihood is around 0.67. Thus, we transform the training data with the following equation to stabilize the variance: $Y_t = \lambda X_t^{\lambda-1}$

```
#Get the optimal lambda
lambda = bcTransform$x[which(bcTransform$y == max(bcTransform$y))]

#Transform the data
cp_train.bc = (1/lambda)*(cp_train^lambda-1)
```

```
# Calculate and print the variances
cat("Variance of original training data:", var(cp_train), "\n")
```

Analysis of the effectiveness of the Box-Cox transformation:

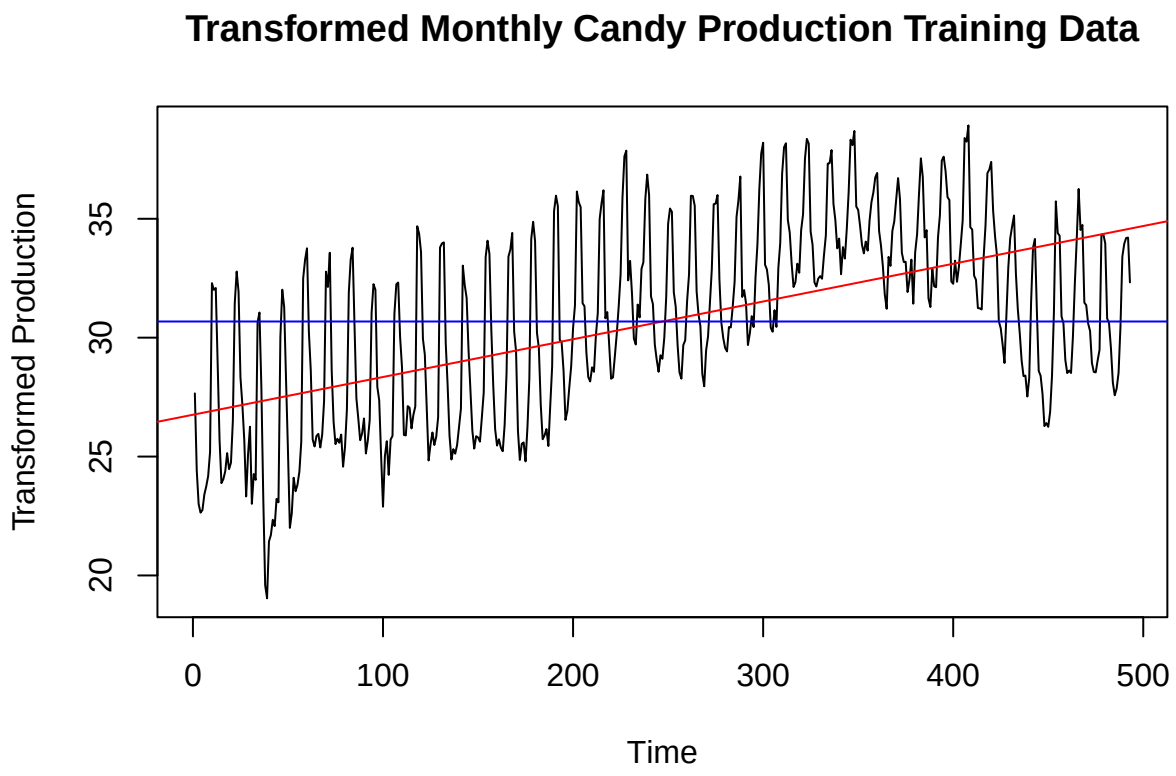
```
## Variance of original training data: 345.6475
```

```
cat("Variance of transformed data:", var(cp_train.bc), "\n")
```

```
## Variance of transformed data: 16.34518
```

The variance largely reduced from 345.6475 to 16.34518 after the transformation.

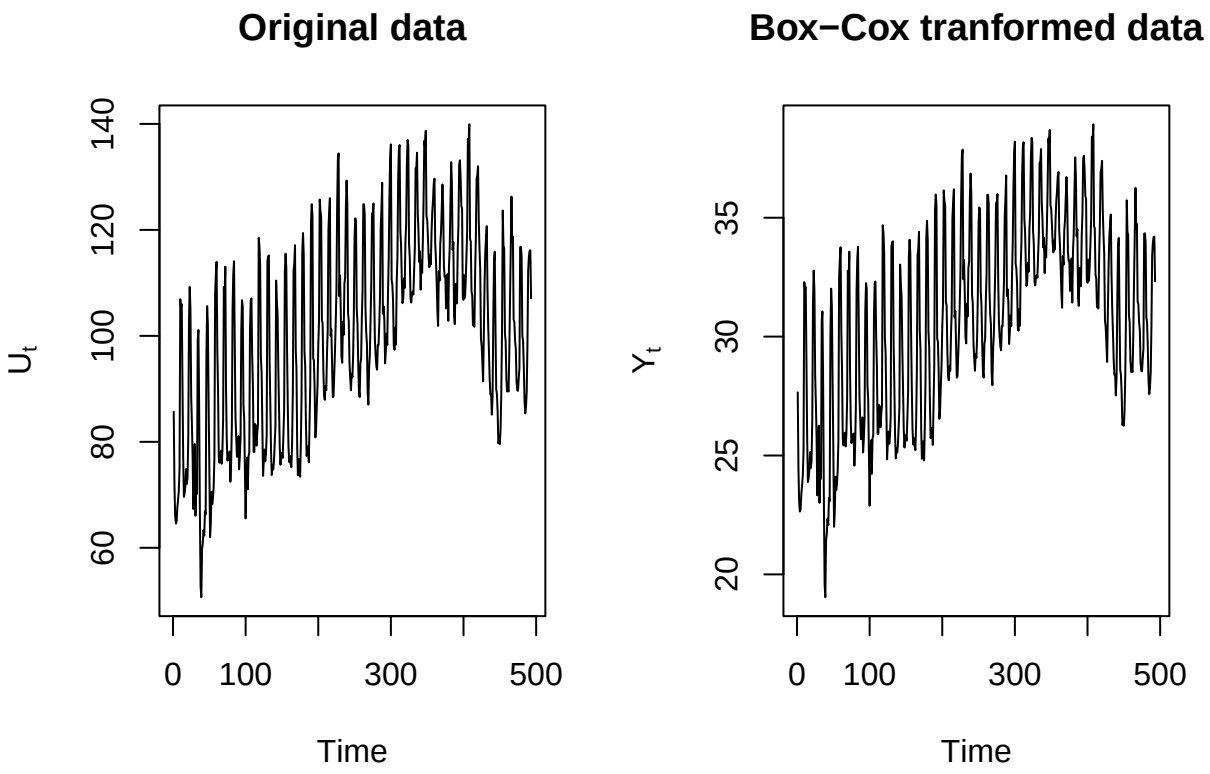
```r
# cp.bc_raw = as.numeric(cp_train.bc)
plot.ts(cp_train.bc, main="Transformed Monthly Candy Production Training Data", ylab="Transformed Produ
nt=length(cp_train.bc)

fit <- lm(cp_train.bc ~ as.numeric(1:nt))
abline(fit, col="red")
abline(h=mean(cp_train.bc), col="blue")
```



The variance stabilizes as time goes by but the positive trend is still presented in the data.

Comparison of the time series data: We now Box-Cox transformed data(Y_t):

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

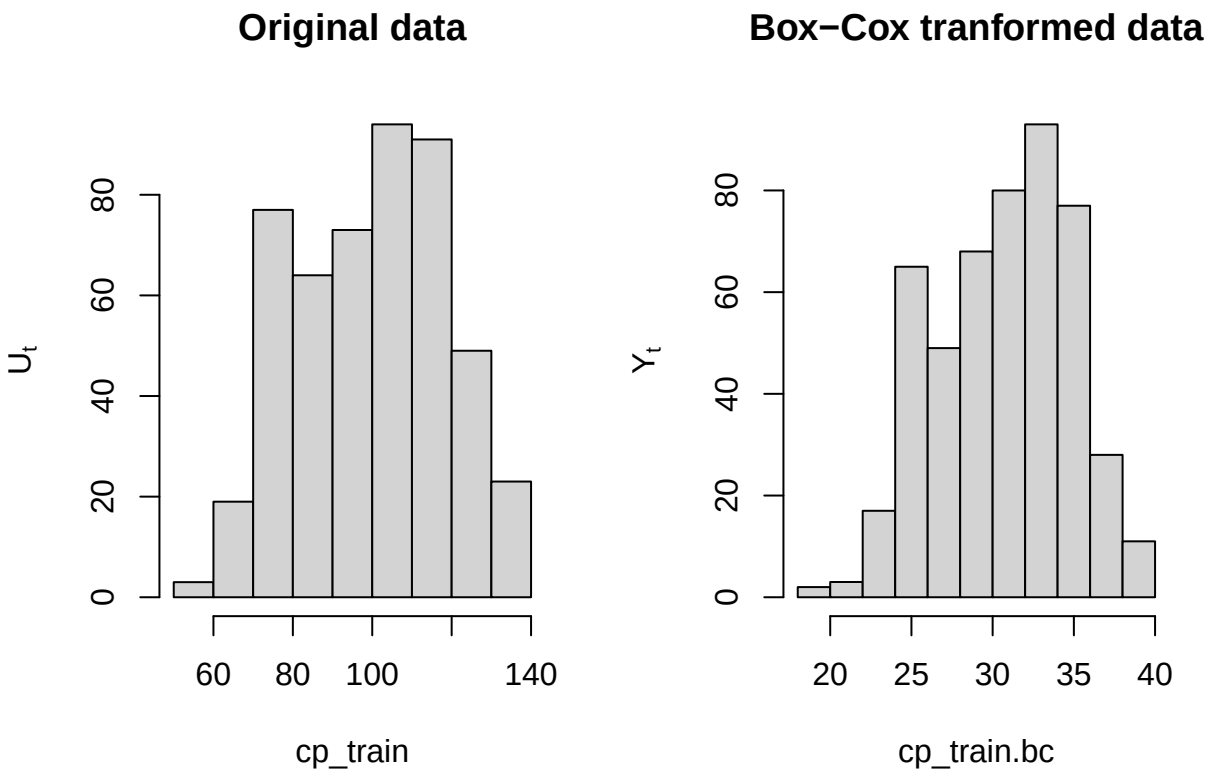



```
par(op)
```

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

```
par(mfrow = c(1,2))
hist(cp_train,main = "Original data",ylab = expression(U[t]))
hist(cp_train.bc,main = "Box-Cox tranformed data", ylab = expression(Y[t]))
```

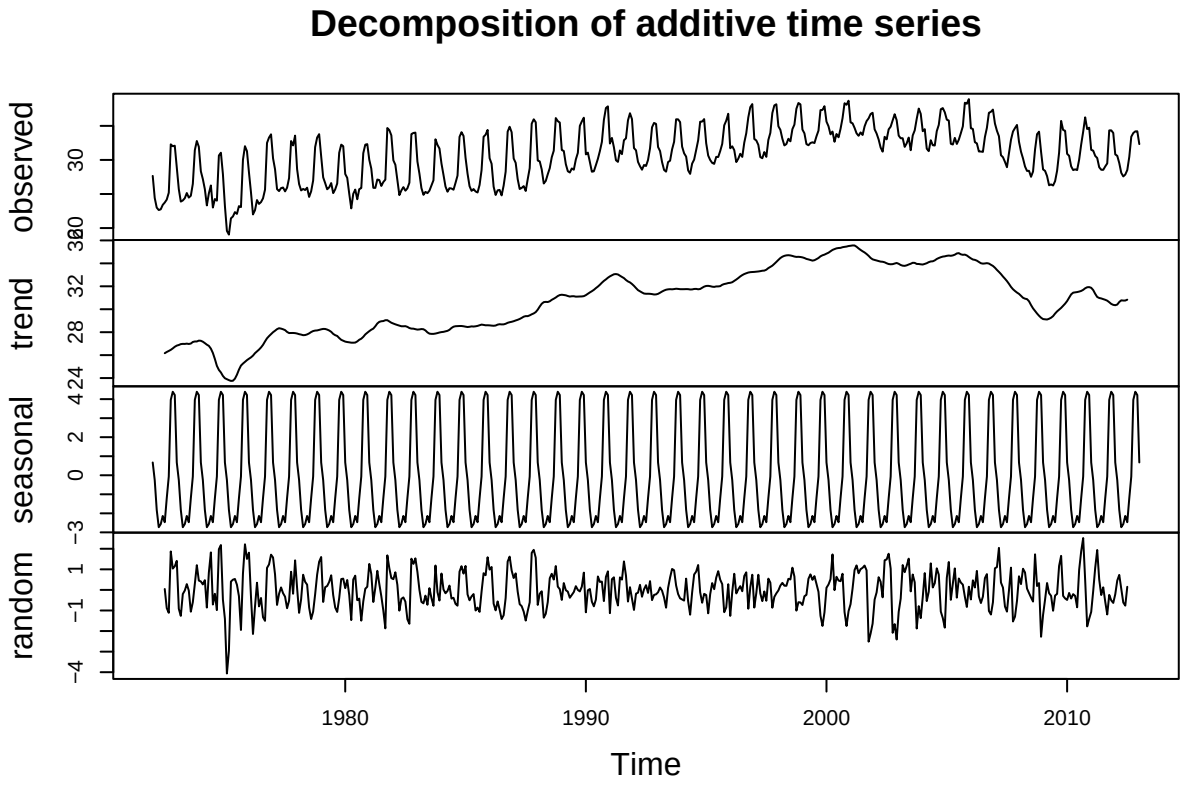
Comparison of the histogram plots:



Compared with the histogram of original data, the transformed data is more Gaussian. Therefore, we will use the transformed data(Y_t) for further analysis.

```
#Convert the transformed data into a time seris object
cp_train.bc.ts <- ts(cp_train.bc, start = c(start_year, start_month), frequency = 12)

# Decompose the time series into components
cp_train_decomposed <- decompose(cp_train.bc.ts, type = "additive")

# Plot the decomposed components
plot(cp_train_decomposed)
```

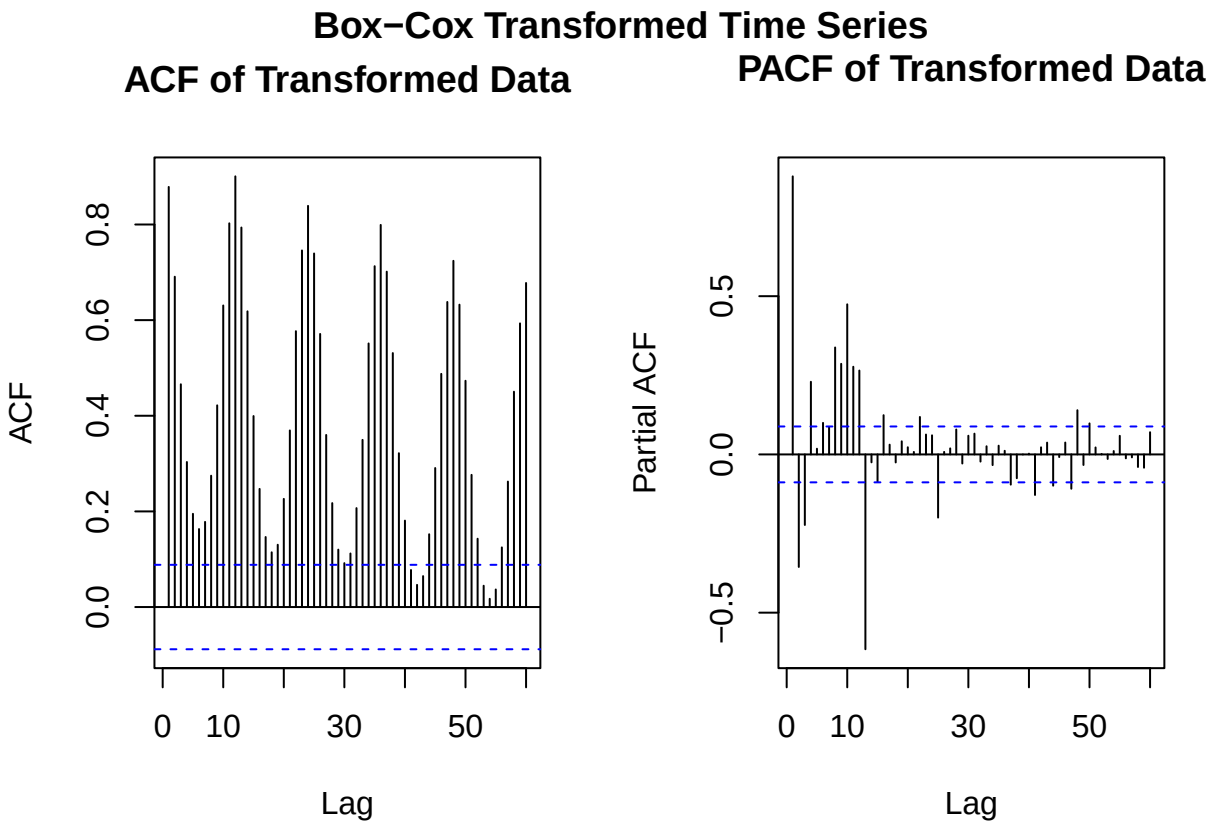
The Decomposition of the transformed time series



Decomposition of the transformed time series shows clear seasonality, indicating a need for differencing.

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

ACF and PACF of the transformed data



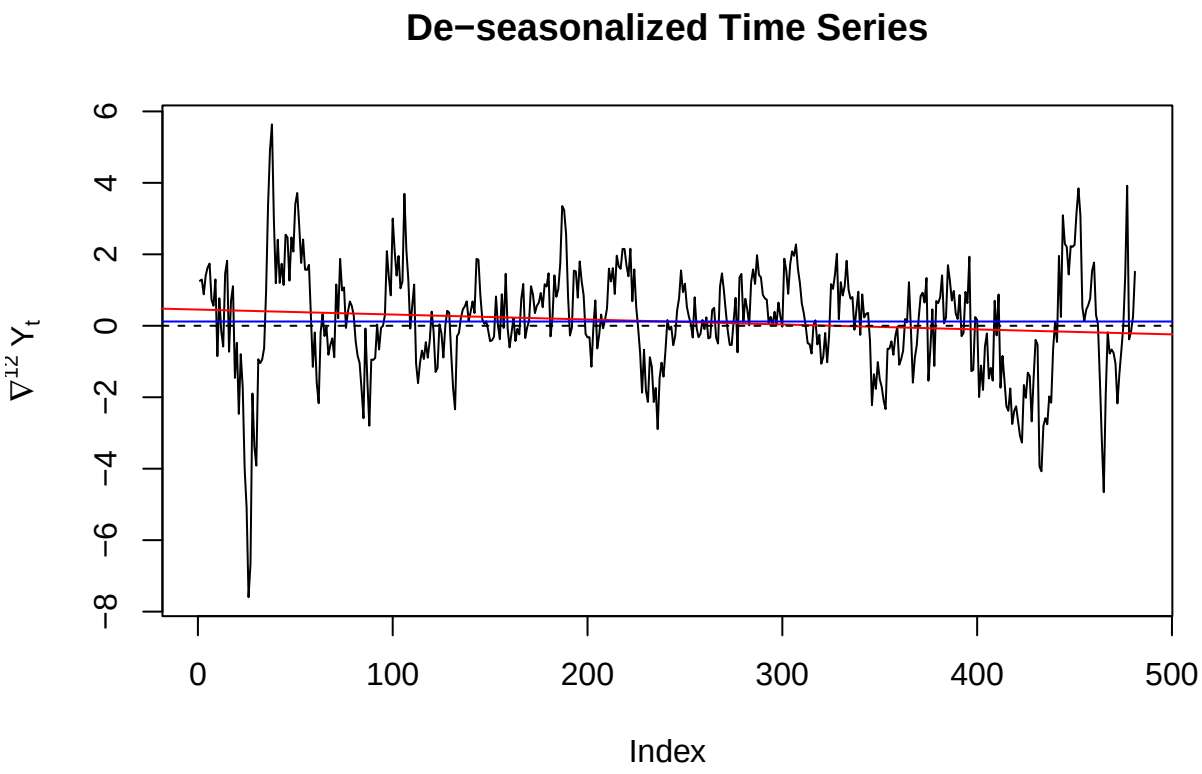
We notice the cyclical behavior in the ACF of the transformed data. Also, there are significant correlations with values moving proportionally every 12 lags. Therefore, we can see that the period of the seasonal component is given by $s = 12$.

2.4.2 Remove seasonal components by differencing

Difference at Lag 12 We remove seasonal components by differencing the transformed time series(Y_t at **lag 12**.

We plot the data after differencing at lag 12 with the regression and mean lines:

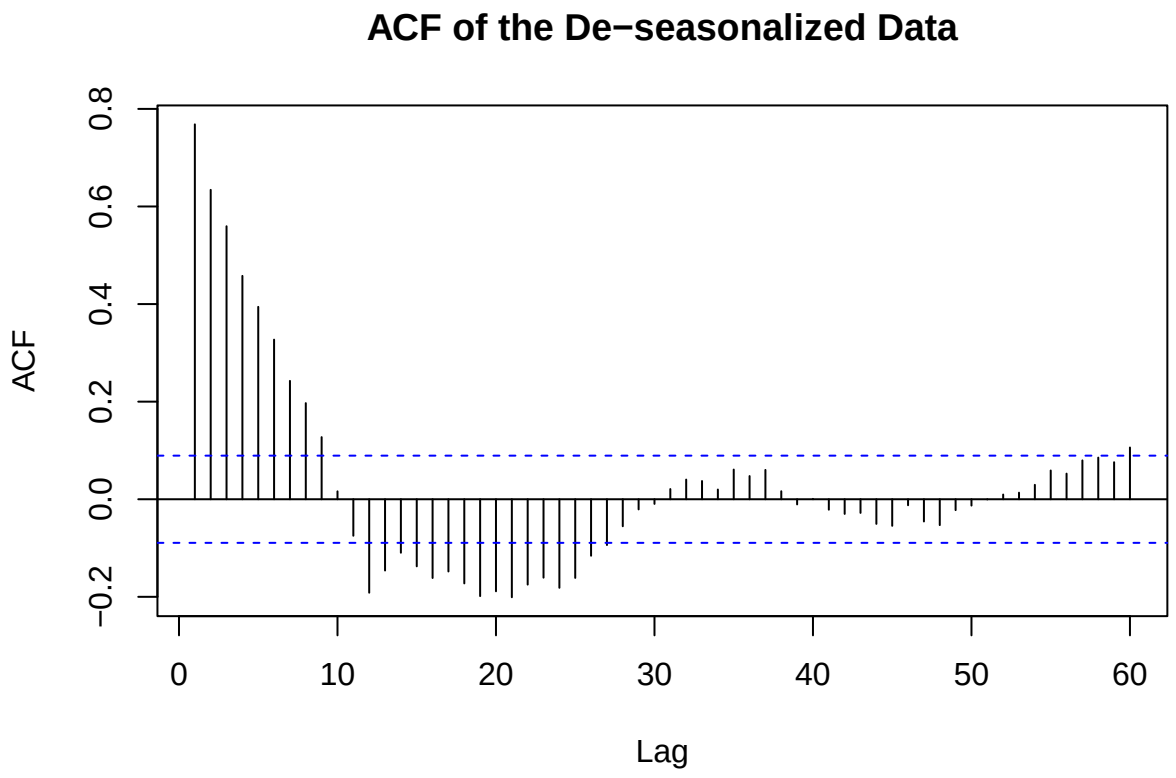
```
# Difference at lag = 12 to remove seasonal component
y1 = diff(cp_train.bc, 12)

plot(y1,main = "De-seasonalized Time Series",ylab = expression(nabla^{12}~Y[t]),type='l')

index = 1: length(y1)
abline(lm(y1 ~ index), col="red")
abline(h=mean(y1) , col='blue')
abline(h = 0,lty = 2)
```



From the plot we see that there is a negative trend in the de-seasonalized data.

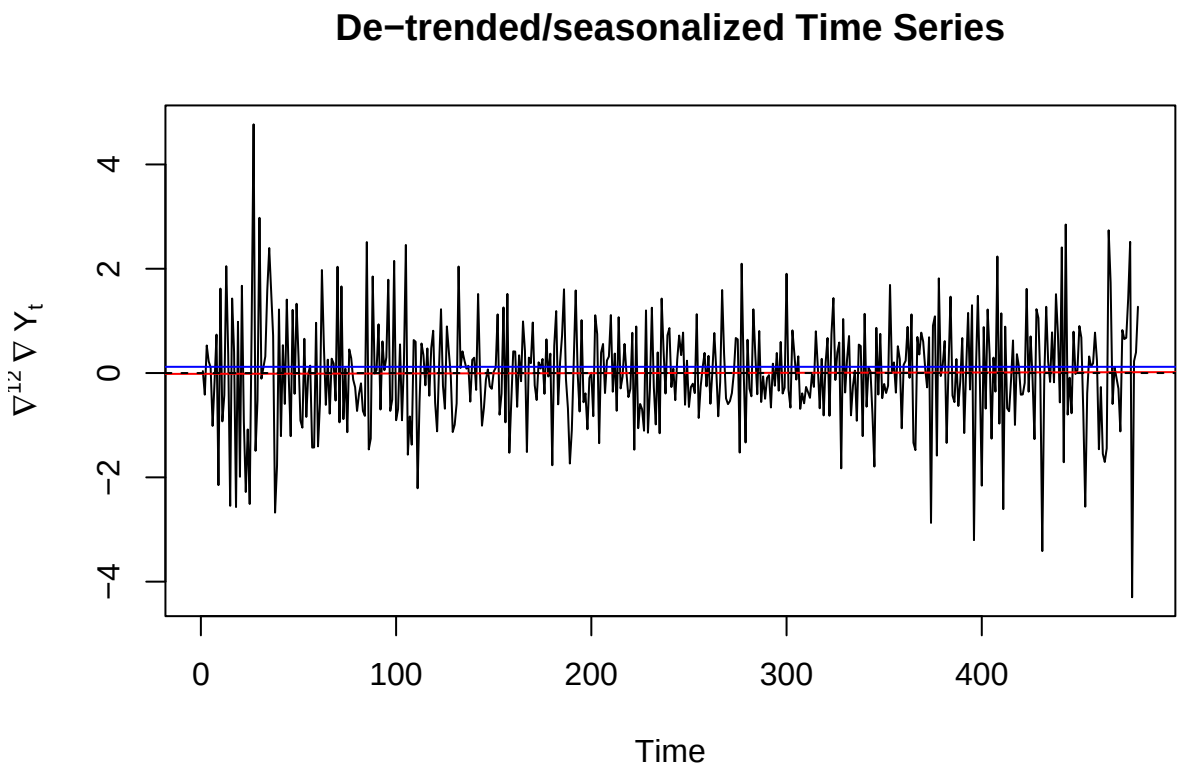
```
acf(y1,lag.max = 60,main = "ACF of the De-seasonalized Data")
```

Difference at Lag 1



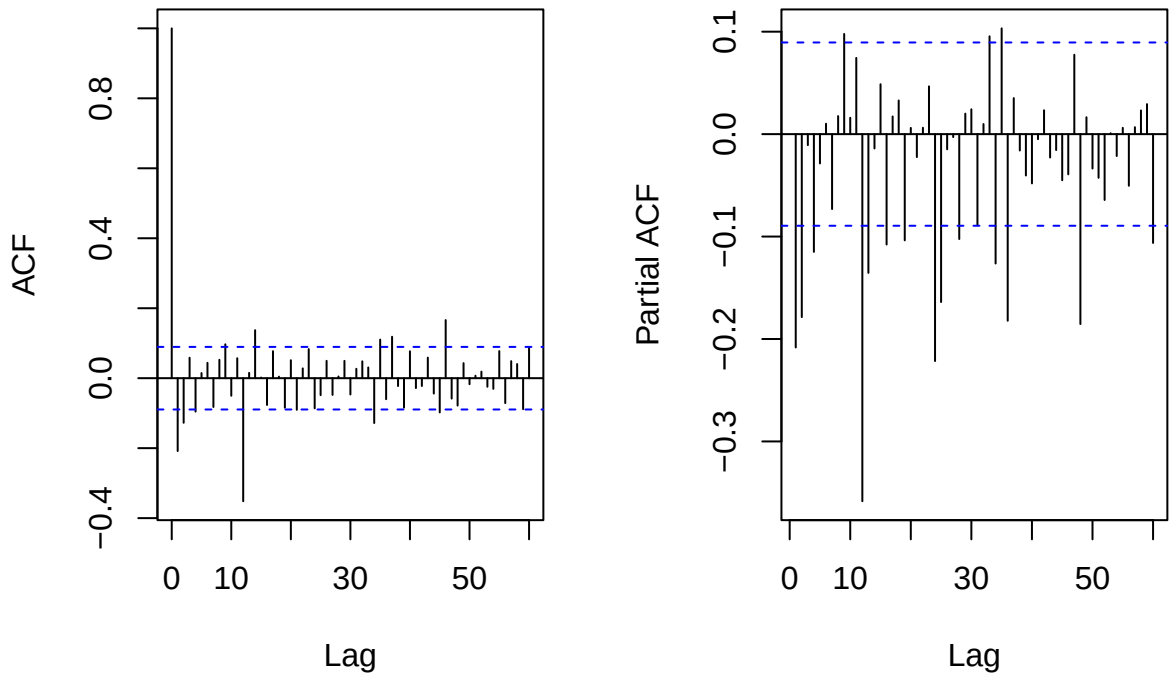
The ACF values decrease slowly. Therefore, we consider taking difference at lag 1 to eliminate the trend.

```
# Difference y1 at lag = 1 to remove trend
y2 = diff(y1, 1)

# Plot the time series with mean and trend line
ts.plot(y2,main = "De-trended/seasonalized Time Series",ylab = expression(nabla^{12}~nabla~Y[t]), type =

index2 = 1: length(y2)
abline(lm(y2 ~ index2), col="red")
abline(h=mean(y1) , col='blue')
abline(h = 0,lty = 2)
```



From the plot above, the regression line is quite horizontal, meaning that we may not have trend now.

```
# Examine the ACF and PACF
par(mfrow = c(1,2))
acf(y2,lag.max = 60,main = "")
pacf(y2,lag.max = 60,main = "")
title("De-trended/seasonalized Time Series",line = -1, outer=TRUE)
```



The ACF values now decay corresponds to a stationary process.

Comparison of the variances We calculate the variance at each step:

```
# Variance of the transformed data
cat("Variance of the transformed data:", var(cp_train.bc), "\n")
```

```
## Variance of the transformed data: 16.34518
```

```
# Variance of the de-seasonalized data
cat("Variance of the de-seasonalized data:", var(y1), "\n")
```

```
## Variance of the de-seasonalized data: 2.291027
```

```
# Variance of the de-trended/seasonalized data
cat("Variance of the de-trended/seasonalized data:", var(y2), "\n")
```

```
## Variance of the de-trended/seasonalized data: 1.05693
```

The variance after both differences at lag 12 and lag 1 has the lowest variance and looks stationary.Therefore, we will use the de-trended/seasonalized data($\nabla_1 \nabla_{12} Y_t$) for further analysis.

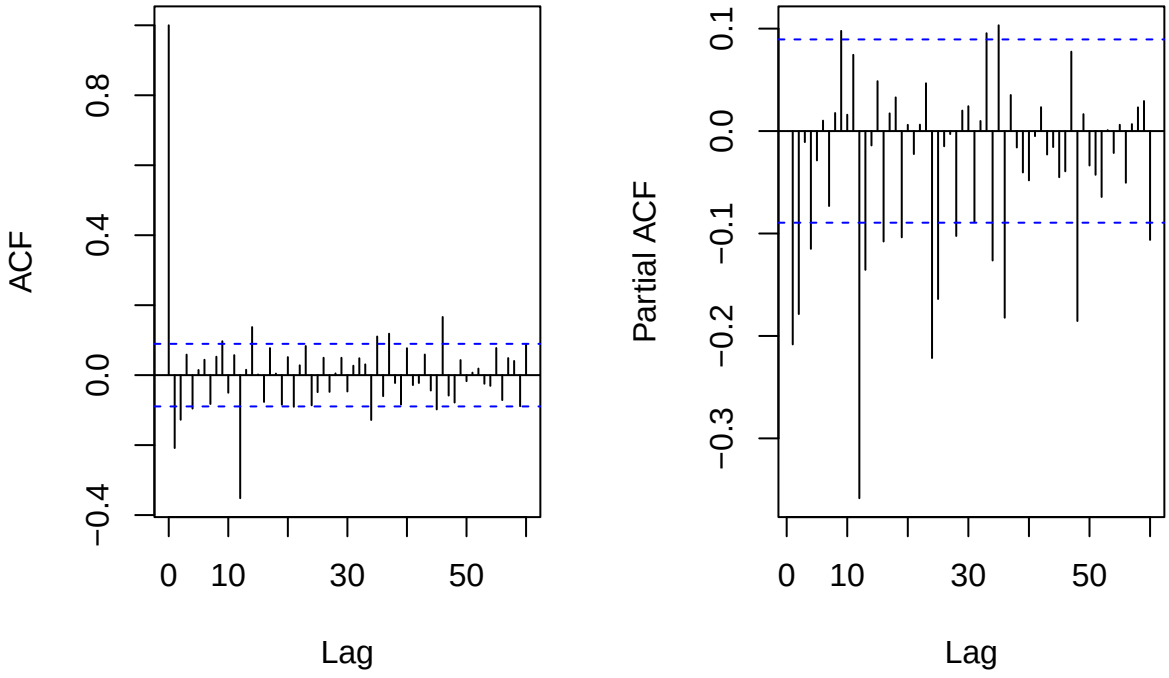
3 Model building

3.1 Model Identification by ACF/PACF

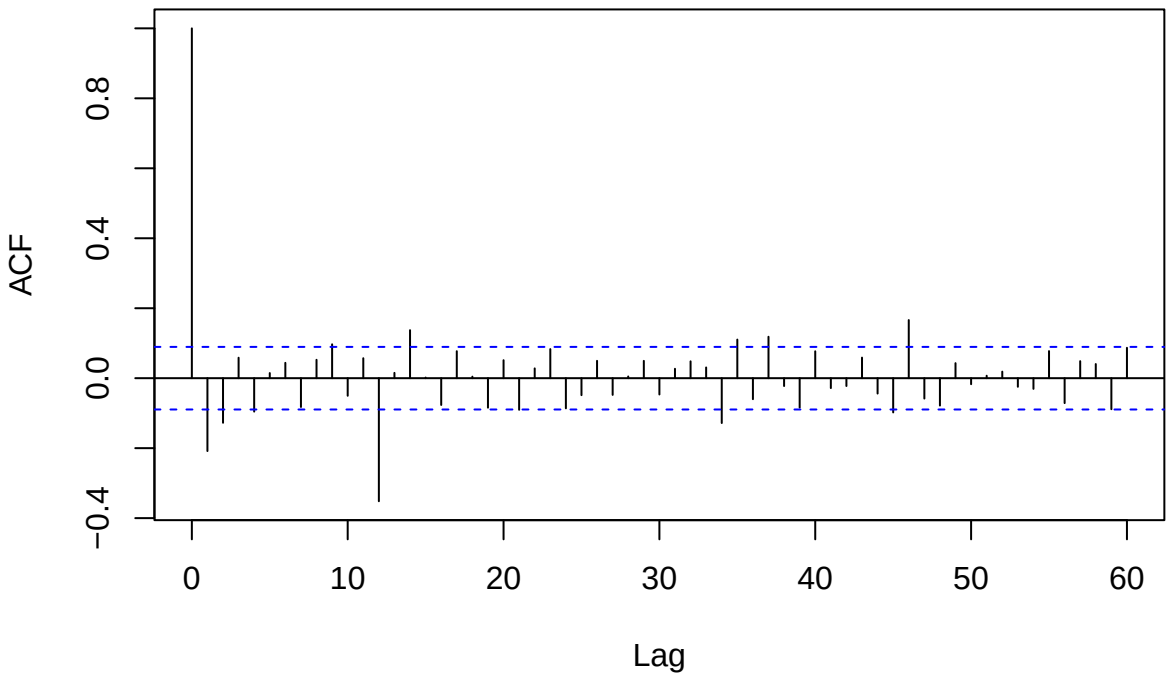
3.1.1 ACF/PACF plots and candidates

We preliminary identify this model by plotting and analyzing the ACF and PACF plots. Given the seasonality and trend of the original data, We first find a suitable SARIMA model to the data.

```
par(mfrow = c(1,2))
acf(y2,lag.max = 60,main = "")
pacf(y2,lag.max = 60,main = "")
```




```
acf(y2,lag.max = 60,main = "")
```



- s = 12

Modeling the seasonal part (P, D, Q): For this part, focus on the seasonal lags h = 1s, 2s, etc. - D = 1 Since we applied one seasonal differencing. - Q: We pick Q = 1 or 2 since the ACF shows a strong peak at h = 1s, 2s and smaller peaks appearing at 3s, 4s ... - P: We can try P = 1 since the PACF shows a strong peak at h = 1s, and smaller peaks appearing at 2s, 3s, 4s, 5s...

Modeling the non-seasonal part (p , d, q): In this case focus on the within season lags, h = 1,. . . ,11. - d = 1 since We applied one differencing to remove the trend. - q: The ACF cuts off at lag h = 1, 2 or 4. We choose q = 1, 2, or 4. - p: The PACF cuts off at lag 2, 3 or 5. We choose p = 2, 3 or 5.

Based on the above info and principle of parsimony, we choose two candidates: -SARIMA(1,1,1)(1,1,1)[12] -SARIMA(2,1,1)(1,1,1)[12]

3.1.2 Coefficients estimation

Now we calculates the AICc values for the candidate models:

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

SARIMA(1,1,1)(1,1,1)[12]

```
##
## Call:
## arima(x = cp_train.bc, order = c(1, 1, 1), seasonal = list(order = c(1, 1, 1),
##     period = 12), method = "ML")
##
## Coefficients:
##          ar1      ma1    sar1     sma1
##       0.5020  -0.7707  0.1326  -0.7779
## s.e.  0.1325   0.1040  0.0607   0.0374
##
## sigma^2 estimated as 0.6839:  log likelihood = -594.42,  aic = 1198.84
```

The model has no coefficient fall in the confidence interval of its standard error.

$\nabla\nabla_{12}Y_t = (1-0.5020(0.1325)B)(1-0.7707(0.1040)B)(1-0.1326(0.0607)B^{12})(1-0.7779(0.0374)B^{12})Z_t, \sigma_Z^2 = 0.6839$

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

SARIMA(2,1,1)(1,1,1)[12]

```
##
## Call:
## arima(x = cp_train.bc, order = c(2, 1, 1), seasonal = list(order = c(1, 1, 1),
##     period = 12), method = "ML")
##
## Coefficients:
##           ar1      ar2     ma1    sar1     sma1
##       -0.6409  -0.2695  0.4002  0.1016  -0.7487
## s.e.   0.1906   0.0508  0.1970  0.0630   0.0400
##
## sigma^2 estimated as 0.6874:  log likelihood = -595.25,  aic = 1202.51
```

The model has no coefficient fall in the confidence interval of its standard error. It can be written as:

$\nabla\nabla_{12}Y_t = (1-0.6409(0.1906)B-0.2695(0.0508)B^2)(1-0.1016(0.0630)B^{12})(1-0.7487(0.0400)B^{12})Z_t, \sigma_Z^2 = 0.6874$

3.2 Find suitable models using maximum likelihood estimation

Q = 1 P=1; p=0,1,2,3,4,5; q=0,1,2,3,4

```
library("qpcR")
```

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

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

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

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

```
aiccs_sarima1 <- matrix(NA, nr = 6, nc = 5)
dimnames(aiccs_sarima1) <- list(p = 0:5, q = 0:4)

for (p in 0:5) {
  for (q in 0:4) {
    model <- try(arima(cp_train.bc, order = c(p, 1, q),
                       seasonal = list(order = c(1, 1, 1), period = 12),
                       method = "ML"), silent = TRUE)

    if (class(model) != "try-error") {
      aiccs_sarima1[p+1, q+1] <- AICc(model)
    } else {
      aiccs_sarima1[p+1, q+1] <- NA  # Assign NA if the model fails
    }
  }
}

aiccs_sarima1
```

```
##    q
## p          0        1        2        3        4
##   0 1238.693 1207.813 1199.088 1200.820 1194.072
##   1 1218.436 1198.924 1198.865 1199.471 1196.129
##   2 1202.213 1202.631 1197.933 1190.625 1173.645
##   3 1204.099 1191.101 1189.105 1170.418 1166.753
##   4 1195.668 1190.943 1191.026 1168.357       NA
##   5 1196.732 1188.290 1190.350 1166.253       NA
```

The best SARIMA model(low AICc value) with Q=1, P=1 is: p=5; q=3 with AICc = 1166.253 or p=3; q=4 with AICc = 1166.753.

Q = 2 P=1; p=0,1,2,3,4,5; q=0,1,2,3,4

```
aiccs_sarima2 <- matrix(NA, nr = 6, nc = 5)
dimnames(aiccs_sarima2) <- list(p = 0:5, q = 0:4)

for (p in 0:5) {
  for (q in 0:4) {
    model <- try(arima(cp_train.bc, order = c(p, 1, q),
                       seasonal = list(order = c(1, 1, 2), period = 12),
                       method = "ML"), silent = TRUE)

    if (class(model) != "try-error") {
      aiccs_sarima2[p+1, q+1] <- AICc(model)
    } else {
      aiccs_sarima2[p+1, q+1] <- NA  # Assign NA if the model fails
    }
  }
}
```

```
aiccs_sarima2
```

```
##    q
## p          0        1        2        3        4
##   0 1239.716 1208.784 1200.161 1202.006 1195.078
##   1 1219.430 1199.504 1200.225 1200.491 1197.145
##   2 1203.584 1198.306 1198.602 1191.625 1174.917
##   3 1205.517 1192.373 1190.577 1170.567 1167.844
##   4 1197.096 1192.365 1192.523       NA 1168.466
##   5 1198.040       NA       NA 1167.917       NA
```

The best SARIMA model(low AICc value) with Q=2, P=1 is: p=5; q=3 with AICc = 1167.917 or p=3; q=4 with AICc = 1168.466.

Among all the models, We choose 4 candidates based on the AICC values:

-Model A: SARIMA(5,1,3)(1,1,1)[12] -Model B: SARIMA(3,1,4)(1,1,1)[12] -Model C: SARIMA(5,1,3)(1,1,2)[12] -Model D: SARIMA(3,1,4)(1,1,2)[12]

Coefficients estimation for the candidate models by AICc

Model A: SARIMA(5,1,3)(1,1,1)[12] We estimate its coefficients:

```
arima(cp_train.bc, order = c(5, 1, 3),
                     seasonal = list(order = c(1, 1, 1), period = 12),
                     method = "ML")
```

```
##
## Call:
## arima(x = cp_train.bc, order = c(5, 1, 3), seasonal = list(order = c(1, 1, 1),
##     period = 12), method = "ML")
##
## Coefficients:
##           ar1      ar2     ar3     ar4     ar5     ma1      ma2      ma3
##       -0.4250  -0.1130  0.8665  0.1437  0.0851  0.1754  -0.1392  -0.9820
## s.e.   0.0452   0.0491  0.0283  0.0499  0.0452     NaN      NaN   0.0146
##         sar1     sma1
##       0.0611  -0.7186
## s.e.  0.0660   0.0444
##
## sigma^2 estimated as 0.6141:  log likelihood = -571.9,  aic = 1165.8
```

NaNs in the standard errors suggests that the model may be over-parameterized, thus we give up this candidate.

Model B: SARIMA(3,1,4)(1,1,1)[12] We estimate its coefficients:

```
arima(cp_train.bc, order = c(3, 1, 4),
                     seasonal = list(order = c(1, 1, 1), period = 12),
                     method = "ML")
```

```
##
## Call:
## arima(x = cp_train.bc, order = c(3, 1, 4), seasonal = list(order = c(1, 1, 1),
##     period = 12), method = "ML")
##
## Coefficients:
##           ar1     ar2     ar3     ma1      ma2      ma3     ma4    sar1
##       -0.2728  0.0170  0.8804  0.0056  -0.1731  -0.9532  0.1650  0.0567
## s.e.   0.0398  0.0451  0.0399  0.0669   0.0200   0.0372  0.0679  0.0667
##          sma1
##       -0.7166
## s.e.   0.0466
##
## sigma^2 estimated as 0.6197:  log likelihood = -573.19,  aic = 1166.38
```

The coefficients of ar2, ma1, sar1 fall into the confidence interval of its standard error, thus might be necessary.

We re-estimate it correspondingly with out those coefficients: Based on tests, eliminating the coefficients of ar2 or sar1 would result in failed estimation, thus we only set the coefficient of ma1 to 0.

```
arima(cp_train.bc, order = c(3, 1, 4),
      seasonal = list(order = c(1, 1, 1), period = 12),
      fixed = c(NA, NA, NA, 0, NA, NA, NA, NA, NA),
      method = "ML")
```

```
##
## Call:
## arima(x = cp_train.bc, order = c(3, 1, 4), seasonal = list(order = c(1, 1, 1),
##     period = 12), fixed = c(NA, NA, NA, 0, NA, NA, NA, NA, NA), method = "ML")
##
## Coefficients:
##           ar1     ar2     ar3  ma1      ma2      ma3     ma4    sar1     sma1
##       -0.2755  0.0144  0.8782    0  -0.1720  -0.9527  0.1722  0.0782  -0.7309
## s.e.   0.0291  0.0328  0.0289    0   0.0112   0.0270  0.0212  0.0660   0.0434
##
## sigma^2 estimated as 0.6192:  log likelihood = -573.19,  aic = 1164.38
```

Now we have a lower AICc value of 1164.38

```
arima(cp_train.bc, order = c(5, 1, 3),
                    seasonal = list(order = c(1, 1, 2), period = 12),
                    method = "ML")
```

Model C: SARIMA(5,1,3)(1,1,2)[12]

```
##
## Call:
## arima(x = cp_train.bc, order = c(5, 1, 3), seasonal = list(order = c(1, 1, 2),
##     period = 12), method = "ML")
```

```
##
## Coefficients:
##           ar1      ar2     ar3     ar4     ar5     ma1      ma2      ma3
##       -0.4270  -0.1127  0.8655  0.1440  0.0832  0.1763  -0.1385  -0.9823
## s.e.   0.5242   0.8175  1.1991  0.7091  0.4370  0.9212   0.9001   0.0254
##          sar1     sma1     sma2
##       -0.4175  -0.2362  -0.3376
## s.e.   0.7387   0.8835   0.3585
##
## sigma^2 estimated as 0.613:  log likelihood = -571.68,  aic = 1167.37
```

Every coefficients except ma3 fall into the confidence interval of its standard error.

We try SMA with q = 3, Q = 0:

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

```
##
## Call:
## arima(x = cp_train.bc, order = c(0, 1, 3), seasonal = list(order = c(0, 1, 0),
##     period = 12), method = "ML")
##
## Coefficients:
##           ma1      ma2     ma3
##       -0.2527  -0.1552  0.0234
## s.e.   0.0457   0.0507  0.0449
##
## sigma^2 estimated as 0.9707:  log likelihood = -674.03,  aic = 1356.05
```

It does not have a lower AICc.

```
arima(cp_train.bc, order = c(3, 1, 4),
                   seasonal = list(order = c(1, 1, 2), period = 12),
                   method = "ML")
```

Model D: SARIMA(3,1,4)(1,1,2)[12]

```
##
## Call:
## arima(x = cp_train.bc, order = c(3, 1, 4), seasonal = list(order = c(1, 1, 2),
##     period = 12), method = "ML")
##
## Coefficients:
##           ar1     ar2     ar3     ma1      ma2      ma3     ma4     sar1
##       -0.2818  0.0067  0.8719  0.0198  -0.1660  -0.9577  0.1557  -0.4069
## s.e.   0.0504  0.0587  0.0506  0.1086   0.0357   0.0341  0.0811   0.5547
##          sma1     sma2
##       -0.2446  -0.3319
```

```
## s.e.   0.5525   0.3777
##
## sigma^2 estimated as 0.6161:  log likelihood = -572.69,  aic = 1167.39
```

The coefficients of ar2, ma1, sar1, sma1, sma2 fall into the confidence interval of its standard error, thus might be necessary.

Based on tests, set the coefficients of ma1 to 0 results in a lower AICc of 1165.55.

```
arima(cp_train.bc, order = c(3, 1, 4),
      seasonal = list(order = c(1, 1, 2), period = 12),
      fixed = c(NA, NA, NA, 0, NA, NA, NA, NA, NA, NA),
      method = "ML")
```

```
##
## Call:
## arima(x = cp_train.bc, order = c(3, 1, 4), seasonal = list(order = c(1, 1, 2),
##     period = 12), fixed = c(NA, NA, NA, 0, NA, NA, NA, NA, NA, NA), method = "ML")
##
## Coefficients:
##           ar1     ar2     ar3  ma1      ma2      ma3     ma4     sar1     sma1
##       -0.2727  0.0169  0.8807    0  -0.1713  -0.9590  0.1760  -0.4231  -0.2282
## s.e.   0.0276  0.0320  0.0276    0   0.0130   0.0228  0.0185   0.5042   0.5017
##          sma2
##       -0.3412
## s.e.   0.3440
##
## sigma^2 estimated as 0.6148:  log likelihood = -572.77,  aic = 1165.55
```

The Best model The Best model we get from AICc value is Model B: SARIMA(3,1,4)(1,1,1)[12] which can be written as:

$$\nabla\nabla_{12}Y_t = (1-0.2755(0.0291)B-0.0144(0.0328)B^2+0.8782(0.0289)B^3)(1-0.1720(0.0112)B^2-0.9527(0.0270)B^3+0.1722(0.02$$

3.3 Diagnostic Checking

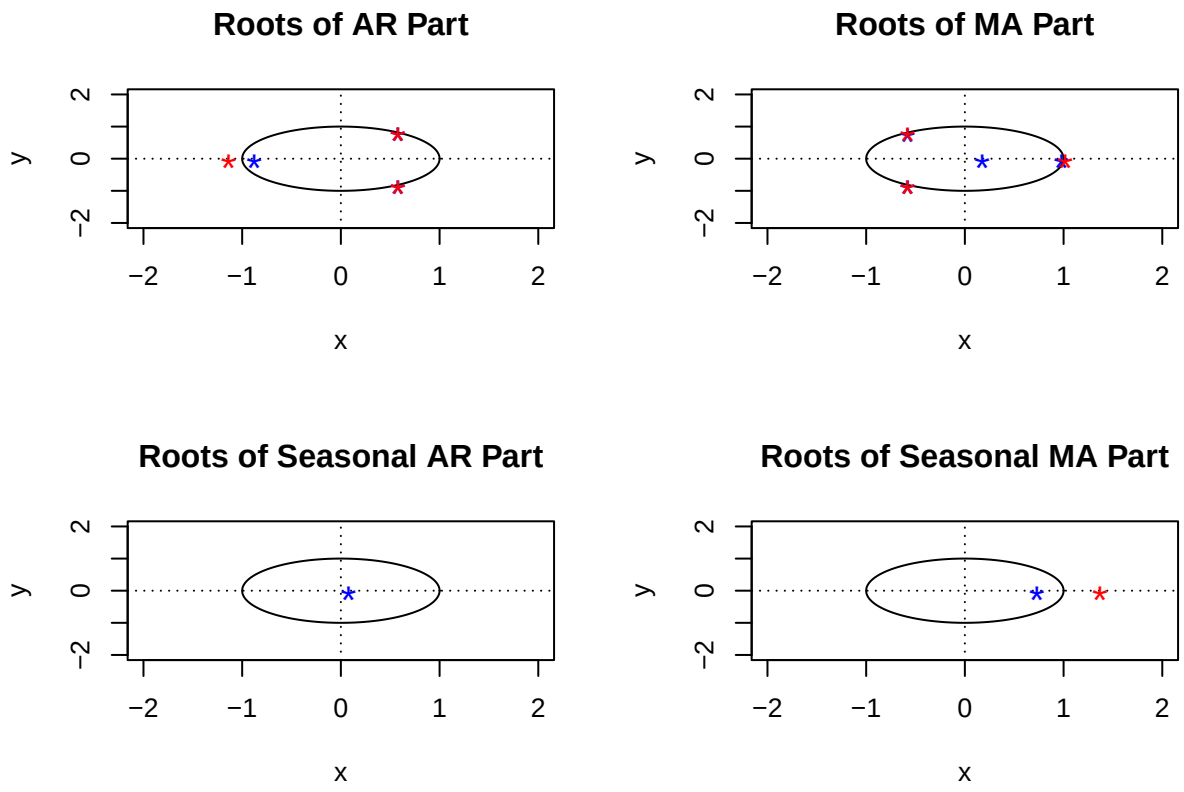
We perform the checking with the lowest AICc.

Model A: SARIMA(3,1,4)(1,1,1)[12]

Invertibility and stationarity Check We check the invertibility and stationarity of the model by calculating and plotting the roots of the 4 parts.

```
source("plot.roots.R")
# AR part roots
ar_poly <- c(1, -0.2755, -0.0144, 0.8782)
ar_roots <- polyroot(ar_poly)

# MA part roots
ma_poly <- c(1, 0, -0.1720, -0.9527, 0.1722)
ma_roots <- polyroot(ma_poly)
```

```r
# Seasonal AR part roots
sar_poly <- c(1, -0.0782)
sar_roots <- polyroot(sar_poly)

# Seasonal MA part roots
sma_poly <- c(1, -0.7309)
sma_roots <- polyroot(sma_poly)

par(mfrow = c(2, 2))
plot.roots(NULL, ar_roots, main = "Roots of AR Part")
plot.roots(NULL, ma_roots, main = "Roots of MA Part")
plot.roots(NULL, sar_roots, main = "Roots of Seasonal AR Part")
plot.roots(NULL, sma_roots, main = "Roots of Seasonal MA Part")
```



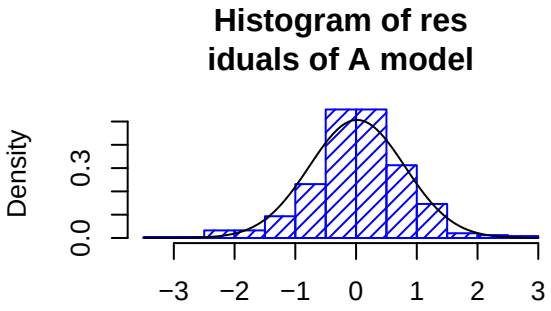
Model A is invertible and stationary, all roots are outside the unit circle.

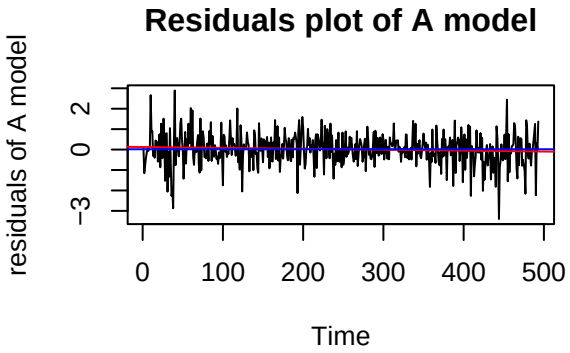
```r
fit1 = arima(cp_train.bc, order = c(3, 1, 4),
             seasonal = list(order = c(1, 0, 1), period = 12),
             fixed = c(NA, NA, NA, 0, NA, NA, NA, NA, NA),
             method = "ML")

res1 = residuals(fit1)
par(mfrow=c(2,2))
```

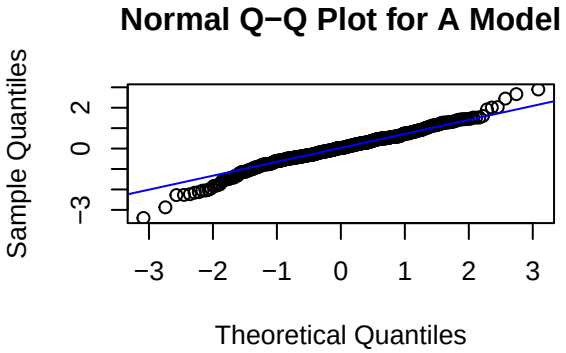
```r
hist(res1,density=20,breaks=20, col="blue", xlab="", prob=TRUE,main="Histogram of res
iduals of A model")
m1 <- mean(res1)
std1 <- sqrt(var(res1))
curve( dnorm(x,m1,std1), add=TRUE )
plot.ts(res1,ylab= "residuals of A model",main="Residuals plot of A model")
fitt1 <- lm(res1 ~ as.numeric(1:length(res1)))
abline(fitt1, col="red")
abline(h=mean(res1), col="blue")
qqnorm(res1,main= "Normal Q-Q Plot for A Model")
qqline(res1,col="blue")
print(m1)
```

Residual Plots

```
## [1] 0.01651673
```



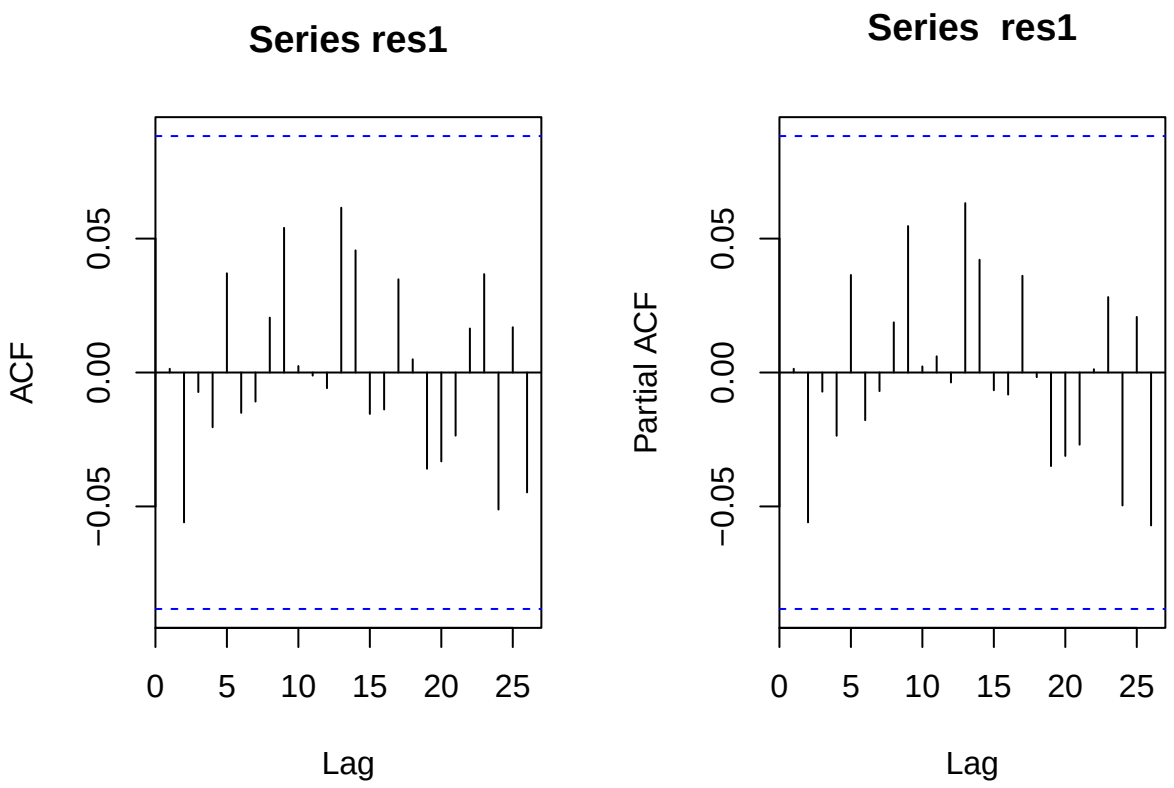




- Based on the above plots, we find No trend, no visible change of variance and no seasonality.
- The Sample mean is close to zero: 0.01651673.
- Histogram suggests that the distribution of the residuals is approximately normal.
- Q-Q plot:The data points are all close along the reference line, suggesting that the residuals follow a normal distribution quite closely.

```
par(mfrow=c(1,2))
acf(res1)
pacf(res1)
```

ACF/PACF Plots



The ACF and PACF plots of the residuals shows that the ACF and PACF values are all in the 95% confidence intervals and can be counted as zeros.

Portmanteau Statistics Then, we check for several independence assumptions by Portmanteau Statistics. lag: Here, we choose lag = 22 because we have n=493 observations and the square root is about 22 fitdf: Also, Model A has 3+4+1+1=9 estimated coefficients with 1 coefficient set to 0. Therefore, fitdf = 9-1=8.

```
#Box-Pierce test
Box.test(res1, lag = 22, type = c("Box-Pierce"), fitdf = 8)
```

```
##
##  Box-Pierce test
##
## data:  res1
## X-squared = 9.5849, df = 14, p-value = 0.7919
```

```r
#Ljung-Box test
Box.test(res1, lag = 22, type = c("Ljung-Box"), fitdf = 8)
```

```
##
##  Box-Ljung test
##
## data:  res1
## X-squared = 9.8485, df = 14, p-value = 0.7732
```

Model A passed the tests with p-value all > 0.05.

```r
ar(res1 , aic = TRUE, order.max = NULL, method = c("yule-walker"))
```

Fitted residuals to AR

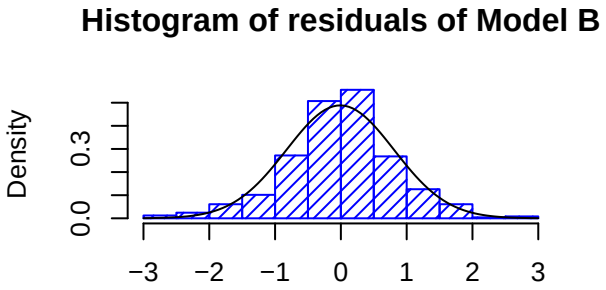
```
##
## Call:
## ar(x = res1, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  0.6191
```

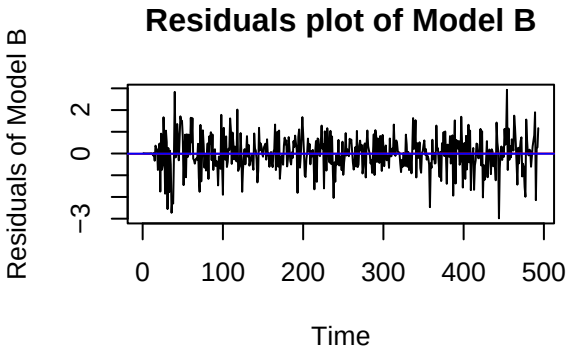
The residual of Model A is fitted to AR(0), indicating it as a WN process and passed the diagnostic checking.

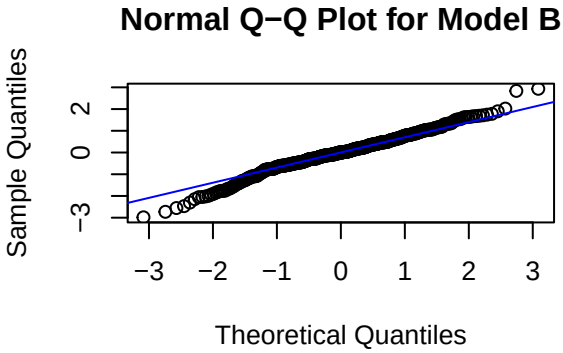
Model B: SARIMA(1,1,1)(1,1,1)[12]

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

res2 = residuals(fit2)
par(mfrow=c(2,2))
hist(res2, density=20, breaks=20, col="blue", xlab="", prob=TRUE, main="Histogram of residuals of Model
m2 <- mean(res2)
std2 <- sqrt(var(res2))
curve( dnorm(x, m2, std2), add=TRUE )
plot.ts(res2, ylab= "Residuals of Model B", main="Residuals plot of Model B")
fitt2 <- lm(res2 ~ as.numeric(1:length(res2)))
abline(fitt2, col="red")
abline(h=mean(res2), col="blue")
qqnorm(res2, main= "Normal Q-Q Plot for Model B")
qqline(res2, col="blue")
print(m2)
```

```
## [1] -0.009142733
```



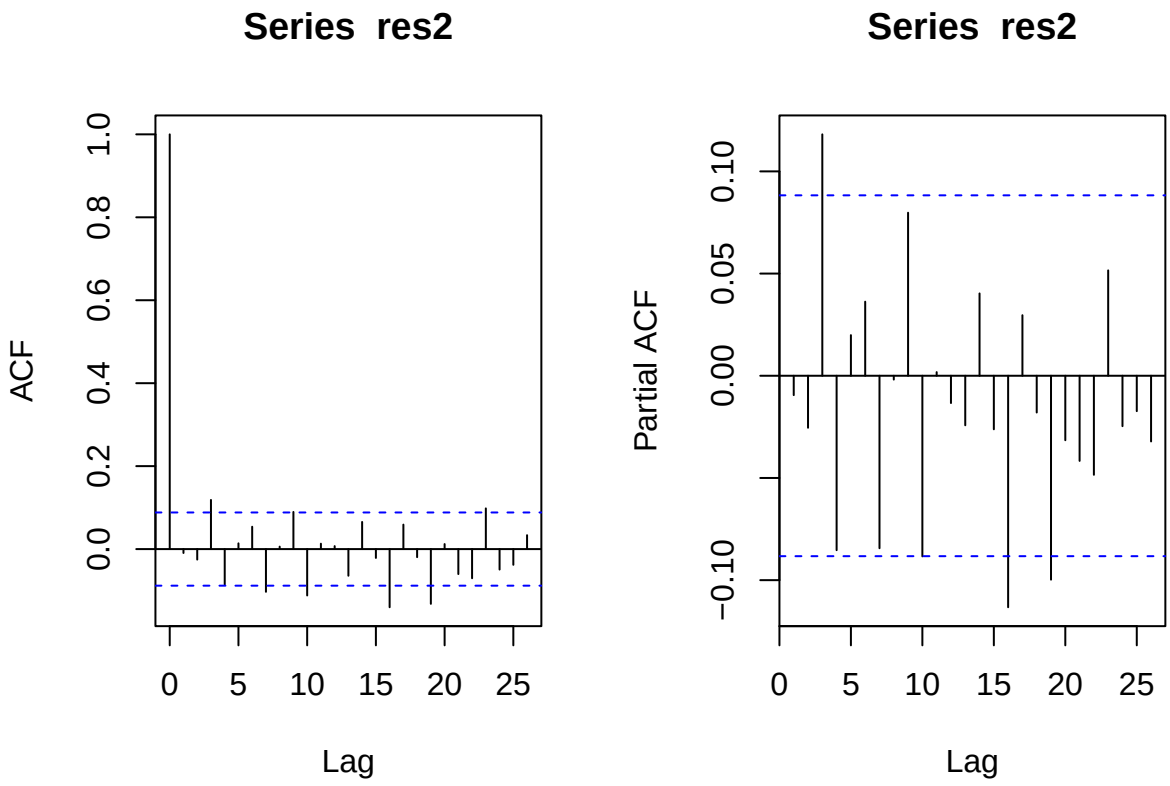




- Based on the above plots, we find No trend, no visible change of variance and no seasonality. - The Sample mean is close to zero: -0.009142733.

- Histogram suggests that the distribution of the residuals is approximately normal.
- Q-Q plot:The data points are all close along the reference line, suggesting that the residuals follow a normal distribution quite closely.

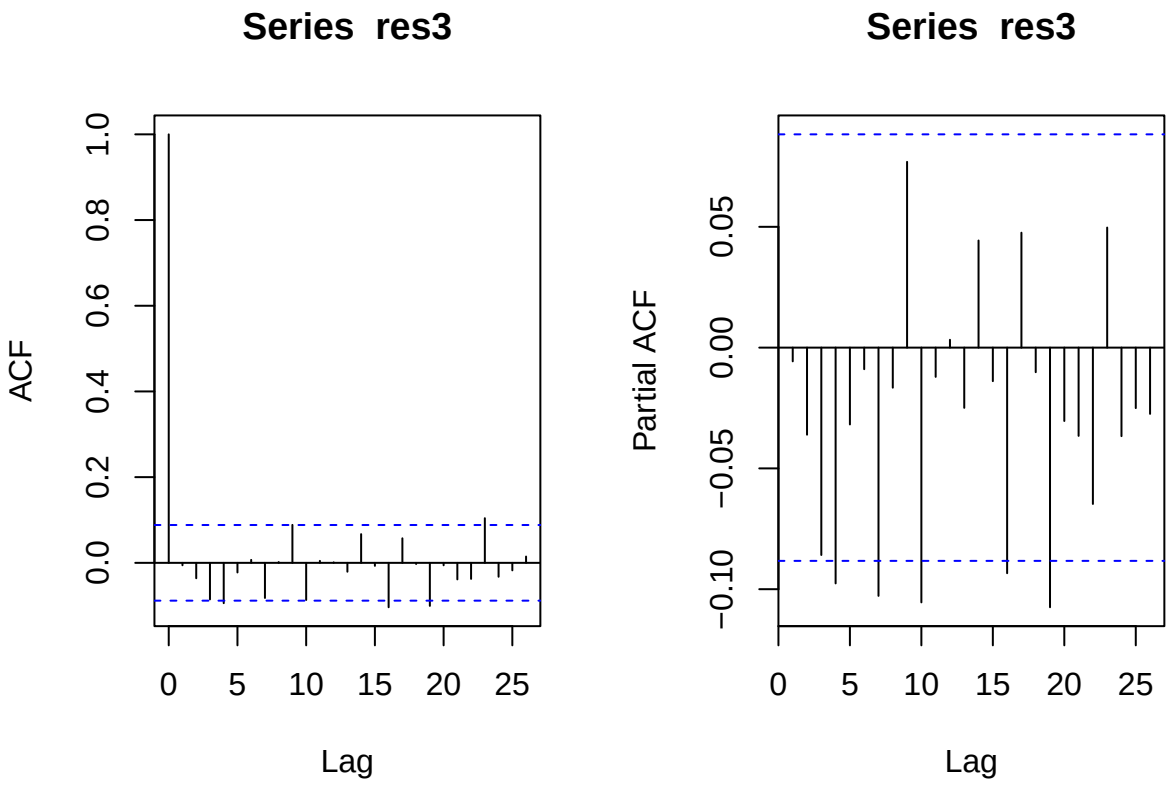
```
par(mfrow=c(1,2))
acf(res2)
pacf(res2)
```



The residual ACF and PACF plots does not show a process close to MA(0).

Model C: SARIMA(2,1,1)(1,1,1)[12]

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

res3 = residuals(fit3)

par(mfrow=c(1,2))
acf(res3)
pacf(res3)
```



The residual ACF and PACF plots does not show a process close to MA(0).

4 Forecasting and validation

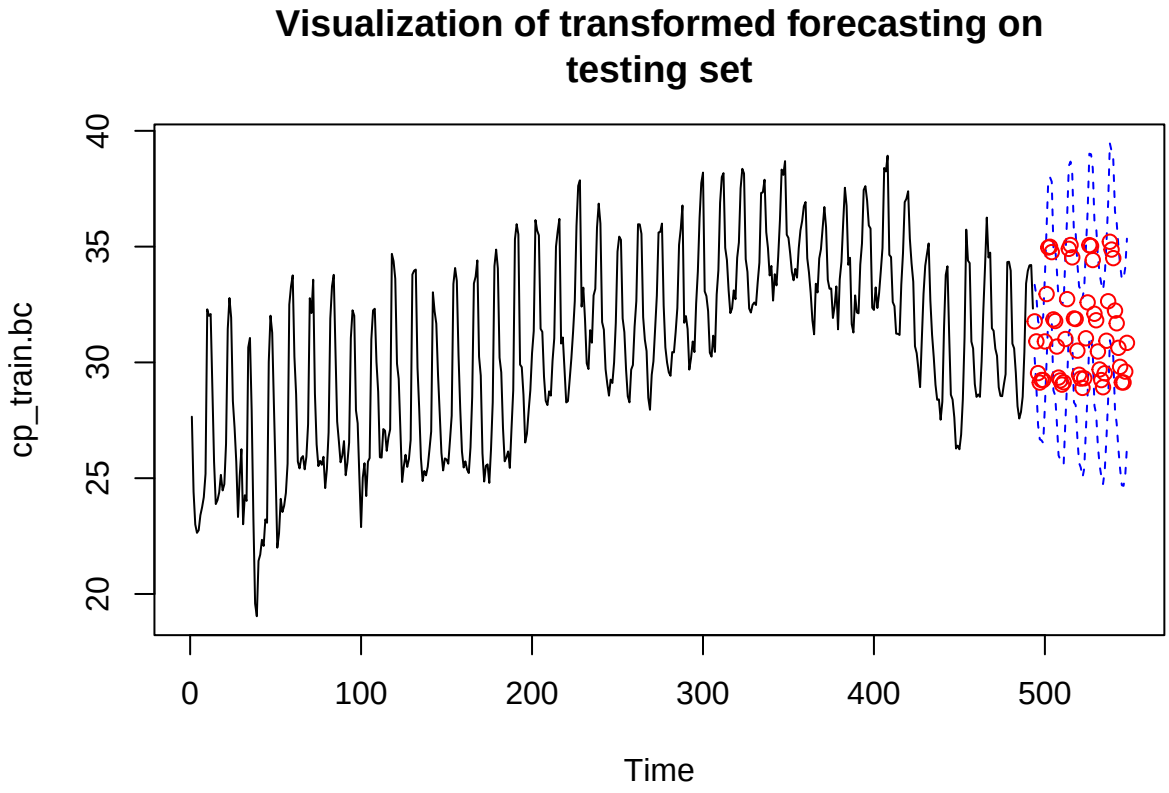
We use Model A SARIMA(3,1,4)(1,1,1)[12] for the final forecasting and validation.

In forecasting section, we predict the last 10% (55) observations of Candy Production in U.S. based on our model and then compare with the true values.

```
library(forecast)
```

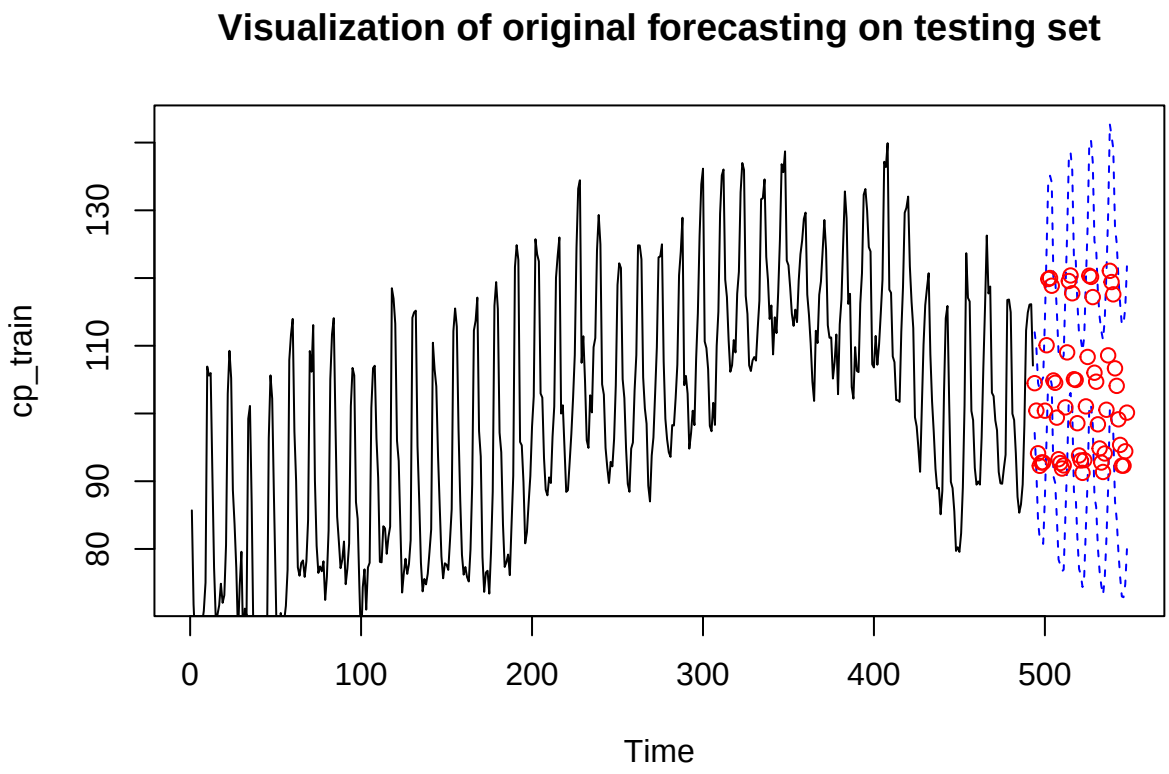
```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
pred.tr <- suppressWarnings(predict(fit1, n.ahead = 55))
U.tr= pred.tr$pred + 2*pred.tr$se
L.tr= pred.tr$pred - 2*pred.tr$se
ts.plot(cp_train.bc, xlim=c(1,length(cp_train.bc)+55), ylim = c(min(cp_train.bc),max(U.tr)), main="Visua
testing set")
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(cp_train.bc)+1):(length(cp_train.bc)+55), pred.tr$pred, col="red")
```



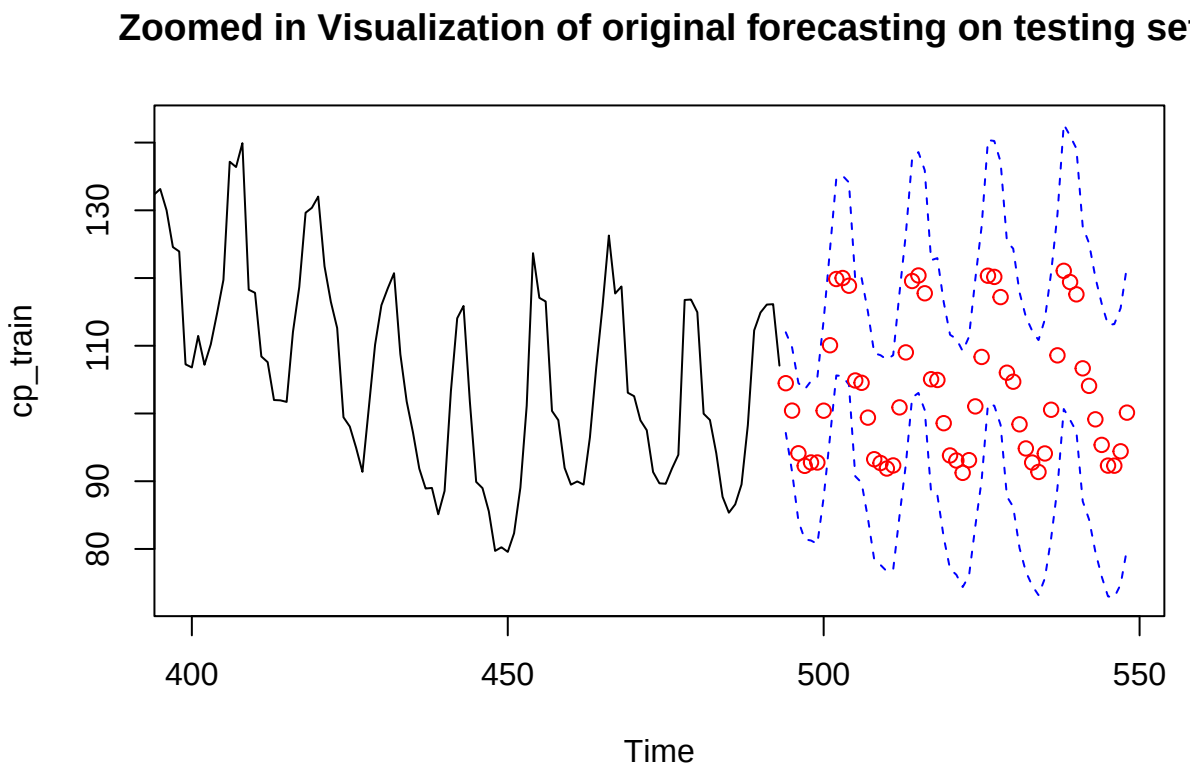
The above figure is the forecast on the transformed data. The true values are within the confidence interval of the forecasting. We convert the forecasting values back to the scale before box-cox transformation to compare with the true values.

```
# Apply the inverse Box-Cox transformation to predictions and confidence intervals
pred.orig <- ((lambda * pred.tr$pred) + 1)^(1/lambda)
U.orig <- ((lambda * U.tr) + 1)^(1/lambda)
L.orig <- ((lambda * L.tr) + 1)^(1/lambda)

# Plotting the transformed time series and the forecasts on the original scale
ts.plot(cp_train, xlim = c(1, length(cp_train) + 55),
        ylim = c(min(L.orig), max(U.orig)), main="Visualization of original forecasting on testing set")
lines(U.orig, col = "blue", lty = "dashed")
lines(L.orig, col = "blue", lty = "dashed")
points((length(cp_train) + 1):(length(cp_train) + 55), pred.orig, col = "red")
```



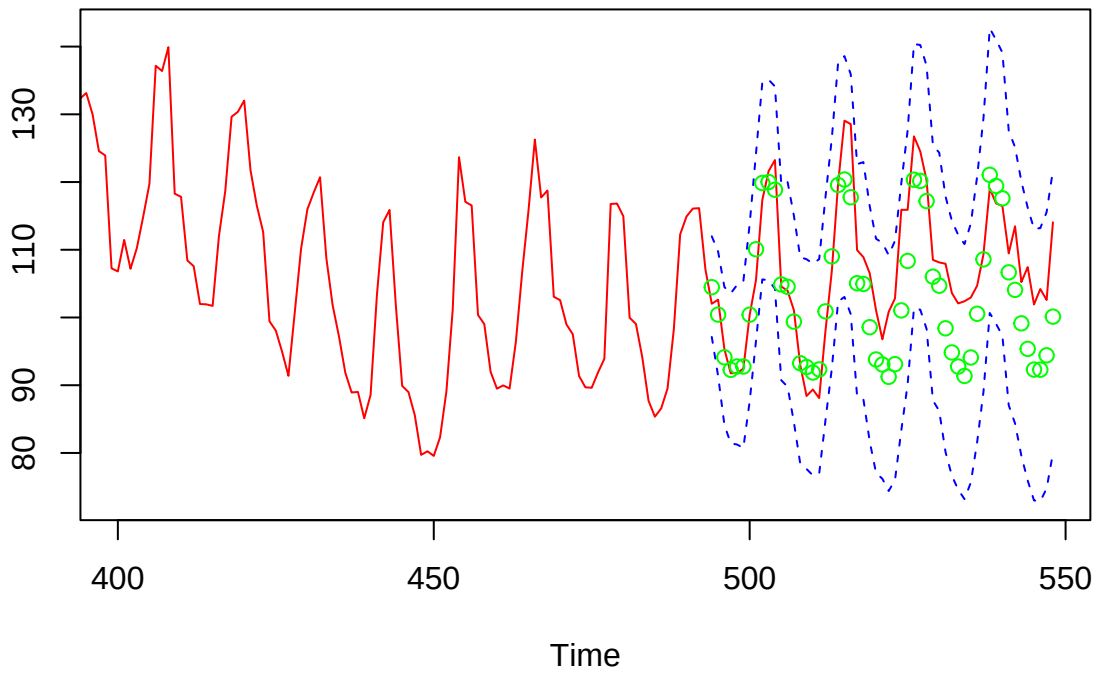
We zoom the graph for better clarity, starting from entry 400:

```
ts.plot(cp_train, xlim = c(400, length(cp_train) + 55),
        ylim = c(min(L.orig), max(U.orig)), main="Zoomed in Visualization of original forecasting on te
lines(U.orig, col = "blue", lty = "dashed")
lines(L.orig, col = "blue", lty = "dashed")
points((length(cp_train) + 1):(length(cp_train) + 55), pred.orig, col = "red")
```



We now show the true values along the predictions:

```
ts.plot(raw_data, xlim = c(400, length(cp_train) + 55),
        ylim = c(min(L.orig), max(U.orig)), main="Zoomed in Visualization of original forecasting with
lines(U.orig, col = "blue", lty = "dashed")
lines(L.orig, col = "blue", lty = "dashed")
points((length(cp_train) + 1):(length(cp_train) + 55), pred.orig, col = "green")
```



-The Original data are indicated by the red line. -Forecasts are represented by the green circles. -We see that the true values of the test set is within the prediction intervals, which shows that our model is successfully forecasting the time series data.

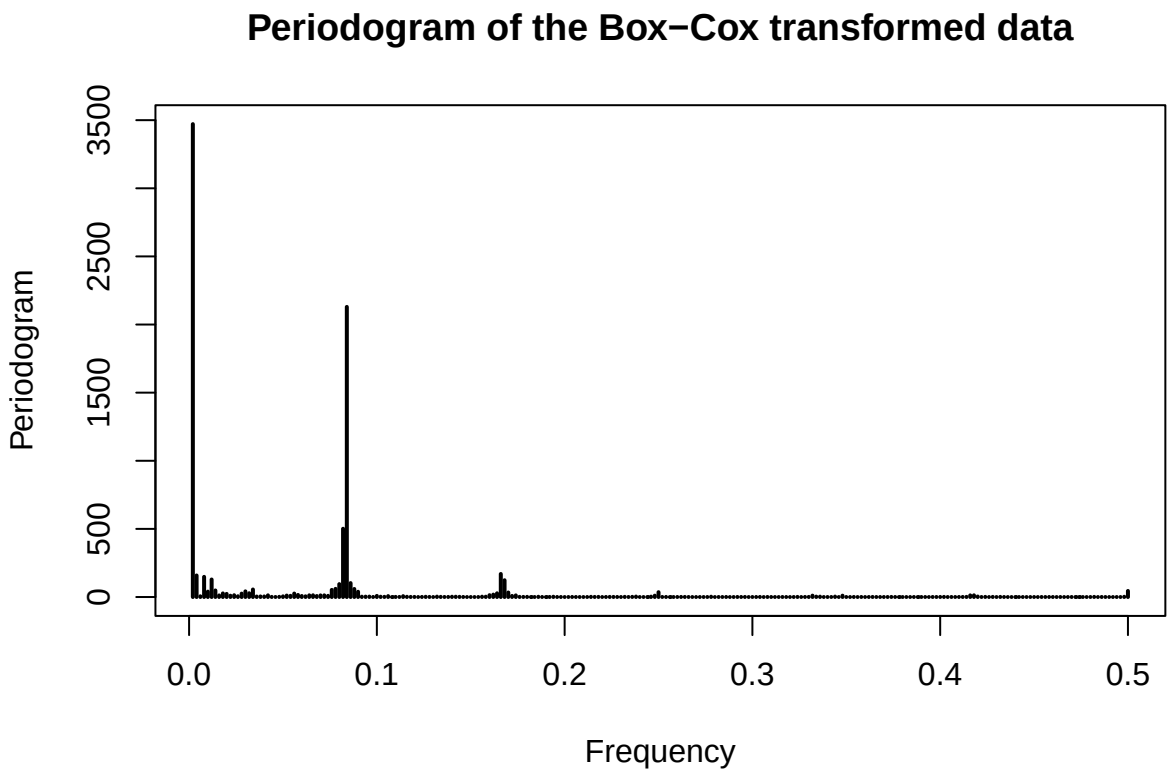
5 Spectral analysis

We use Periodogram to identify period(s) of our seasonal time series.

```
library(TSA)
```

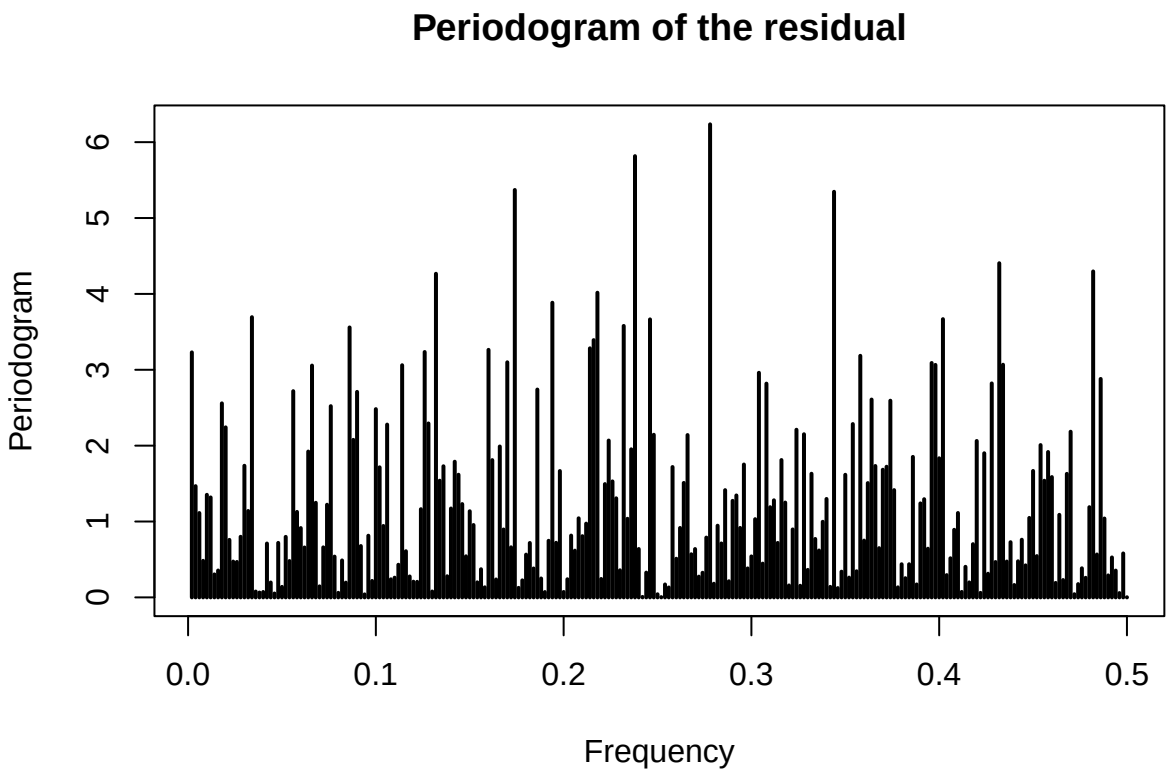
```
## Registered S3 methods overwritten by 'TSA':
##   method       from
##   fitted.Arima forecast
##   plot.Arima   forecast

##
## Attaching package: 'TSA'

## The following objects are masked from 'package:stats':
##
##     acf, arima

## The following object is masked from 'package:utils':
##
##     tar
```

```r
periodogram(cp_train.bc, main="Periodogram of the Box-Cox transformed data")
```



The Periodogram of the Box-Cox transformed data shows a spike at about 1/12, indicating a period of 12.

```r
periodogram(res1, main="Periodogram of the residual")
```



The Periodogram of residuals shows no dominated frequency, indicating the residuals follows a WN.

Fisher's Test

```
library("GeneCycle")
```

```
## Loading required package: longitudinal

## Loading required package: corpcor

## Loading required package: fdrtool

##
## Attaching package: 'GeneCycle'

## The following object is masked from 'package:TSA':
##
##     periodogram

## The following object is masked from 'package:forecast':
##
##     is.constant
```

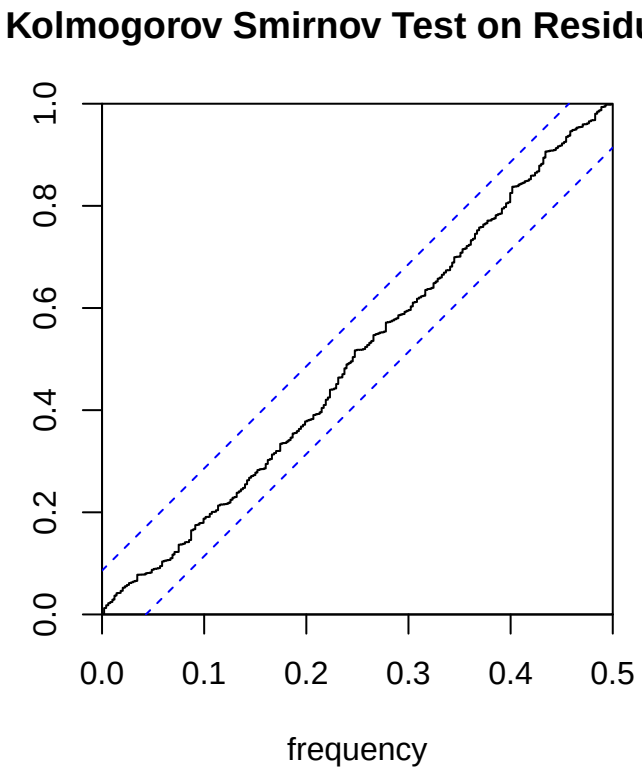
```r
fisher.g.test(res1)
```

```
## [1] 0.9158448
```

The residuals pass the Fisher's test: it is a Gaussian White Noise.

Kolmogorov Smirnov Test

```r
cpgram(res1,main = "Kolmogorov Smirnov Test on Residuals")
```



All data are within the confidence interval, the residuals pass the Kolmogorov Smirnov Test: it is a Gaussian White Noise.

6 Conclusion

In conclusion, our SARIMA(3,1,4)(1,1,1)[12] model successfully forecasted the U.S. candy production and demonstrated a good accuracy of the prediction confidence intervals. This model could be a valuable tool for predicting future trends in the candy industry.

7 References

-1.Data set: https://www.kaggle.com/datasets/rtatman/us-candy-production-by-month/data.

-2.Slides, Notes, and Lab materials.