Seasonal

Suyog

2023-04-03

Analyzing and Forecasting Seasonal CO2 Emissions in Delhi using Time Series Analysis

by Suyog Sunil Khadke

20009192

Submitted to: Professor Hadi Safari

Course: MA 641 Time Series Analysis

Stevens Institute of Technology

Date: May 7th 2023

Abstract:

This project report aims to forecast the CO2 emissions in Delhi using time series analysis. The study utilizes the Box-Jenkins Method to model and forecast the CO2 emissions. The data exhibits seasonal behavior, which is addressed in the analysis. The main findings and conclusions of the study are presented in the report.

Keywords: Time Series Analysis, Box-Jenkins Method, Seasonality, CO2 Emissions, Forecasting, SARIMA, ARIMA

Introduction:

The air quality in Delhi, India has been a major concern for several years, with pollution levels frequently reaching hazardous levels. One of the primary sources of air pollution in the city is the burning of stubble, which is the leftover straw from the previous season’s crops. Farmers in the surrounding states of Punjab, Haryana, and Uttar Pradesh often burn stubble to prepare their fields for the next crop, which leads to a significant increase in air pollution in Delhi.

The impact of stubble burning on air quality in Delhi is a complex problem that requires a thorough understanding of the underlying trends and patterns in the data. Time series analysis can provide valuable insights into the nature of this problem by analyzing the temporal patterns of air quality measurements over time. It could also be used to build models that can predict future pollution levels based on historical data, which could be used to inform policy decisions and interventions to mitigate the problem.

Step 1: Collecting Data

```
library(readr)
library(TSA)
```

```
##
## Attaching package: 'TSA'
```

```
## The following object is masked from 'package:readr':
##
##     spec
```

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

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

```
require(tseries)
```

```
## Loading required package: tseries
```

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

```
library(MASS)
library(forecast)
```

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

```
#CO2 Emission Per Millon Metric Ton. Monltly form 2006 to 2017
library(readr)
final_data_set_seasonal <- read_table("C:/Users/win/Desktop/final_data_set_seasonal.txt",
    col_names = FALSE, col_types = cols(X1 = col_datetime(format = "%Y-%m-%d ")))
na.omit(final_data_set_seasonal)
```

```
## # A tibble: 62 × 2
##    X1                     X2
##    <dttm>              <dbl>
##  1 1990-01-01 00:00:00  20.5
##  2 1990-02-01 00:00:00  17.8
##  3 1990-03-01 00:00:00  19.0
##  4 1990-04-01 00:00:00  21.8
##  5 1990-05-01 00:00:00  25.6
##  6 1990-06-01 00:00:00  28.9
##  7 1990-07-01 00:00:00  31.4
##  8 1990-08-01 00:00:00  32.1
##  9 1990-09-01 00:00:00  29.5
## 10 1990-10-01 00:00:00  25.7
## # ℹ 52 more rows
```

```
tsmonthly <- ts(as.vector(t(as.matrix(final_data_set_seasonal$X2))),
                  start=c(1990,1), end=c(1995,2), frequency=12)



tsmonthly
```

```
##        Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
## 1990 20.51 17.85 18.98 21.84 25.63 28.92 31.41 32.09 29.47 25.72 22.35 20.38
## 1991 22.27 19.54 20.68 23.89 28.03 31.62 34.49 35.19 32.20 28.02 24.35 22.18
## 1992 23.78 20.81 22.01 25.36 29.74 33.52 36.54 37.27 34.02 29.53 25.64 23.34
## 1993 25.18 22.01 23.24 26.76 31.38 35.28 38.48 39.29 35.85 31.12 27.00 24.59
## 1994 26.60 23.26 24.57 28.35 33.17 37.34 40.64 41.46 37.79 32.86 28.54 26.00
## 1995 27.99 24.43
```

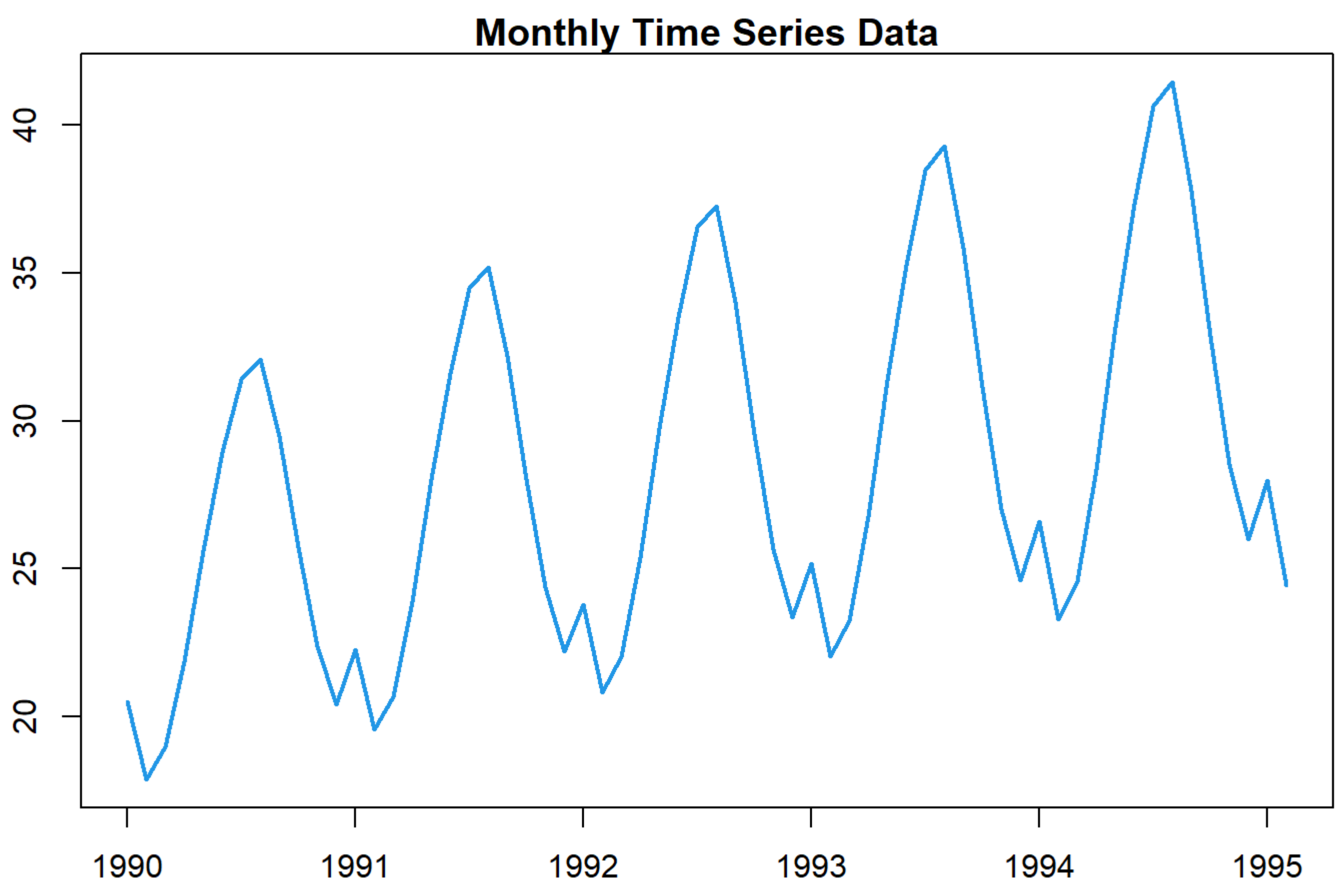
```
library(seastests)
#dropping NA values
tsmonthly<-na.omit(tsmonthly)

isSeasonal(tsmonthly, test = "combined", freq = 12)
```

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

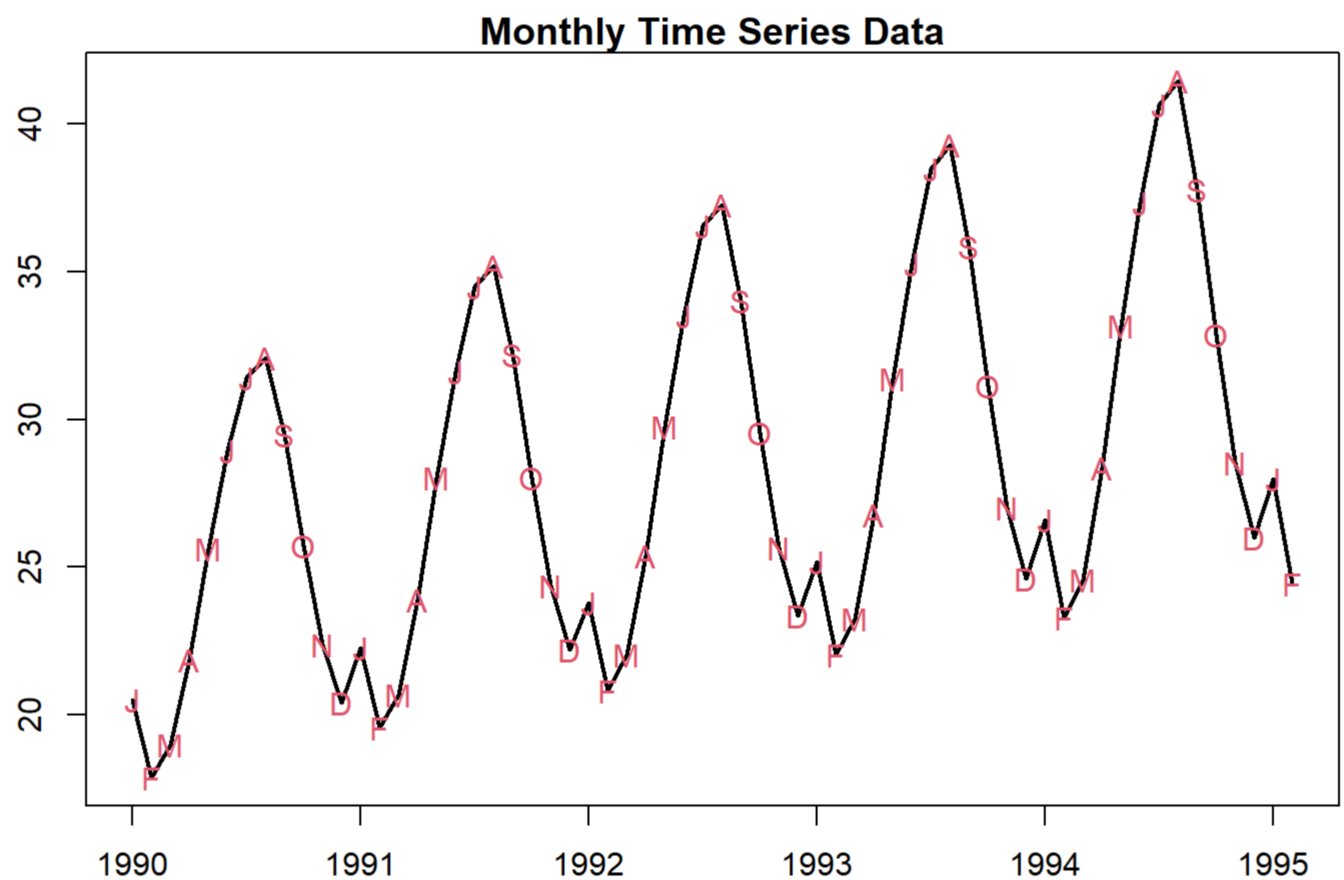
CO2 emissions (in million metric tons) for each month of several years. The emissions are recorded for each year from 1990 to 1995

```
par(mfrow = c(1, 1), mar = c(1, 0, 1, 0) + 0.2, oma = c(1, 2, 2, 0))
plot(tsmonthly, main = "Monthly Time Series Data",type='l',lwd=2,col=4)
```



Seasonal data and a slightly upward trend suggests that there is a cyclical pattern in the data, but there is also a gradual increase in the overall trend.

```
par(mfrow = c(1, 1), mar = c(1, 0, 1, 0) + 0.2, oma = c(1, 2, 2, 0))
plot(tsmonthly, main = "Monthly Time Series Data",type='l',lwd=2)
points(y = tsmonthly, x = time(tsmonthly), pch = as.vector(season(tsmonthly)), lwd = 1, col = 2,
bg = "blue")
```



This is an additional graph that depicts the values on a monthly basis.

```
library(seastests)
#dropping NA values
tsmonthly<-na.omit(tsmonthly)

isSeasonal(tsmonthly, test = "combined", freq = 12)
```

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

By default, the WO-test combines the results of the QS-test and the kw-test, both calculated on the residuals of an automatic non-seasonal ARIMA model. If the p-value of the QS-test is below 0.01 or the p-value of the kw-test is below 0.002, the WO-test will classify the corresponding time series as seasonal.

```
library(seastests)
combined_test(tsmonthly,freq = 12)
```

```
## Test used:  WO
##
## Test statistic:  1
## P-value:  0 0 1.028458e-07
```

All the above test classifies the data as seasonal.

```
adf.test(tsmonthly)
```

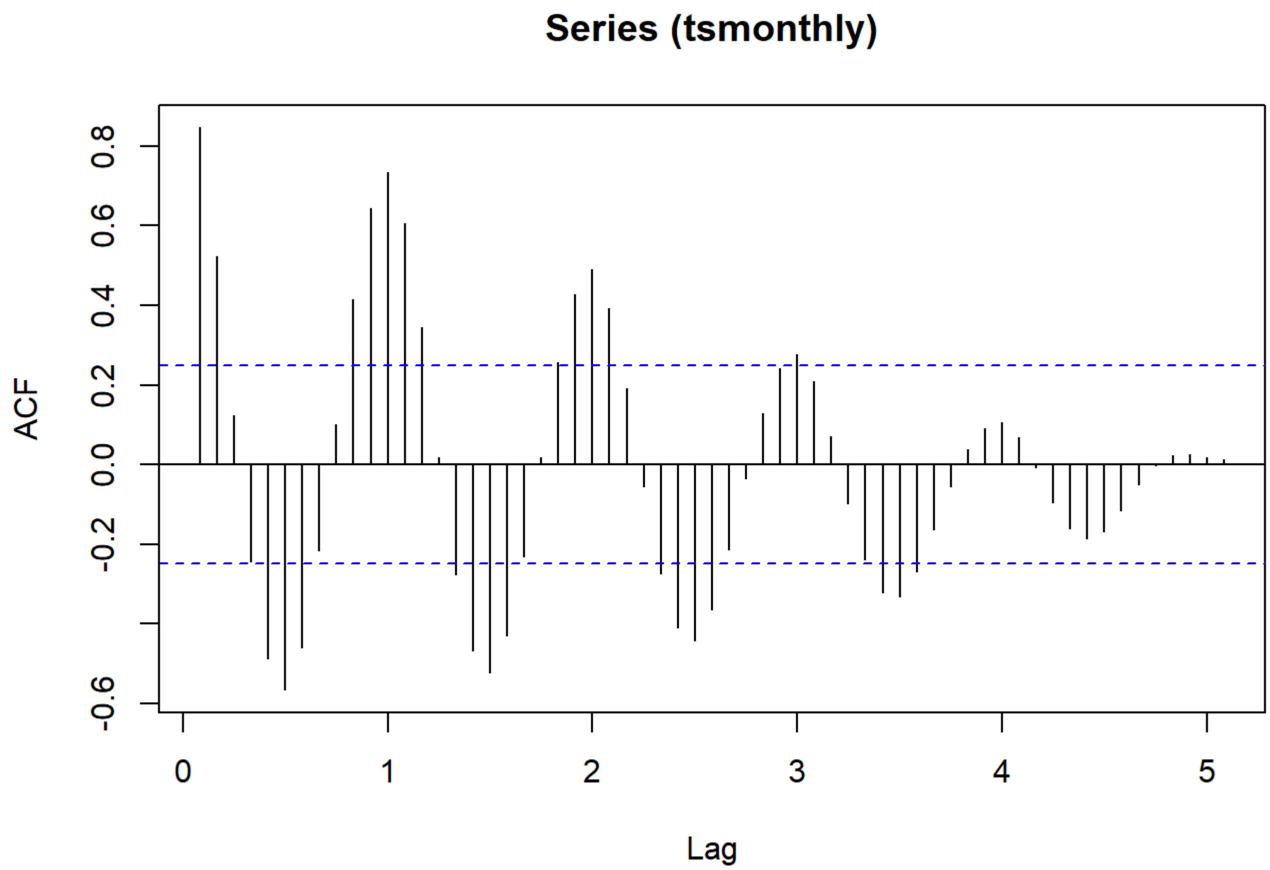
```
## Warning in adf.test(tsmonthly): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  tsmonthly
## Dickey-Fuller = -6.4808, Lag order = 3, p-value = 0.01
## alternative hypothesis: stationary
```

According to ADF test the time series is stationary.

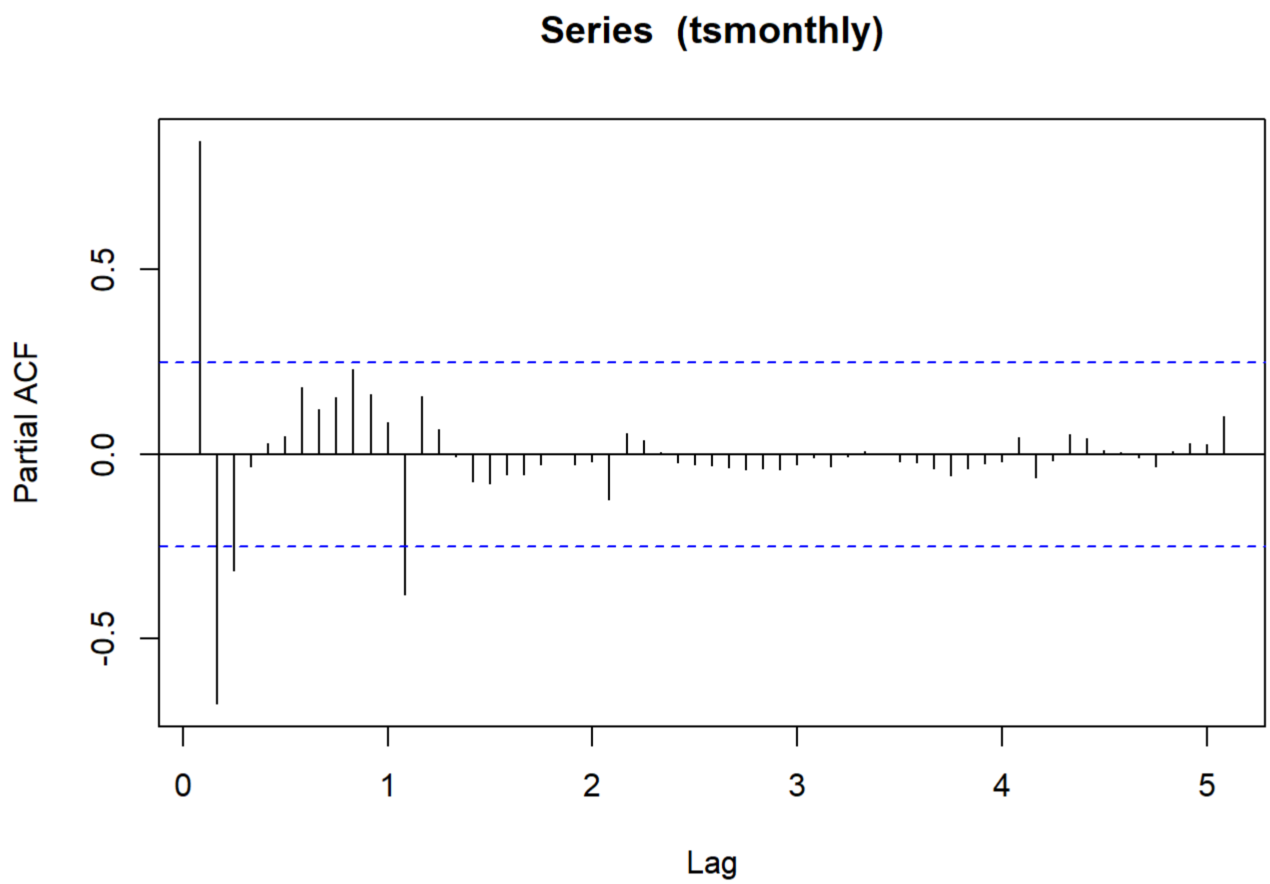
Step 2: Finding Models

```
acf((tsmonthly), lag.max = 70)
```



We got Tailing off ACF. Now let us see the PACF

```
pacf((tsmonthly), lag.max = 70)
```



PACF suggests that there might be an AR(2) process.

```
eacf(tsmonthly)
```

```
## AR/MA
##   0 1 2 3 4 5 6 7 8 9 10 11 12 13
## 0 x x o o x x x o o x x  x  x  x
## 1 x x o x x x x o o x x  x  x  x
## 2 x o o o o o o o o o o  x  o  o
## 3 o o o o o o o o o o o  x  o  o
## 4 o o o o o o o o o o o  x  o  o
## 5 o o o o o o o o o o o  x  o  o
## 6 o o o x o o o o o o o  x  o  o
## 7 x o o x o o o o o o o  x  x  o
```

From EACF we have ARMA(2,1). We will consider this model as well.

ARMA(2,1)

```
first_fit<-(Arima(tsmonthly, order = c(2, 0, 1), seasonal = list(order = c(1, 0, 1), period = 1
2)))

first_fit
```

```
## Series: tsmonthly
## ARIMA(2,0,1)(1,0,1)[12] with non-zero mean
##
## Coefficients:
```

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

```
##          ar1      ar2     ma1  sar1    sma1     mean
##       1.4777  -0.4983  0.1137     1  0.8049  29.2578
## s.e.  0.1874   0.1863  0.2104   NaN  0.3899      NaN
##
## sigma^2 = 0.01359:  log likelihood = 31.59
## AIC=-49.18   AICc=-47.1   BIC=-34.29
```

We are getting AIC=-49.18 so far we will record this value and compare it with further values.

ARIMA(4,1,1):

```
sarima_model <- auto.arima(tsmonthly, seasonal = TRUE, stepwise = FALSE,
                           approximation = FALSE, D = 0, max.order = 5,
                           max.P = 5, max.D = 0, max.Q = 3)

# Print the summary of the fitted SARIMA model
summary(sarima_model)
```

```
## Series: tsmonthly
## ARIMA(4,1,1)
##
## Coefficients:
##          ar1      ar2      ar3      ar4      ma1
##       1.0499  -0.2024  -0.1163  -0.2428  -0.8686
## s.e.  0.1422   0.2066   0.2064   0.1411   0.0535
##
## sigma^2 = 3.875:  log likelihood = -126.89
## AIC=265.78   AICc=267.33   BIC=278.44
##
## Training set error measures:
##                     ME     RMSE      MAE      MPE     MAPE     MASE       ACF1
## Training set 0.3976677 1.870855 1.442459 1.083918 5.509568 0.819672 -0.1271707
```

We are getting AIC=265.78. As we have better model before this we should not consider this model for now.

$

$$(1-\phi_1 L-\phi_2 L^2-\phi_3 L^3-\phi_4 L^4)(1-L)y_t=\theta_1\varepsilon_{t-1}+\varepsilon_t$$

$

Step 3/4: Parameter Reduency | Parameter Estimation

```
library(forecast)
best=99999999999
best_Index=0
for (p in c(0,1,2,3)){

    for (q in c(0,1,2,3)){
      for (P in c(0,1,2,3)){
          for (Q in c(0,1,2,3)){
    sarima_model <- tryCatch({
  # Code block to execute
  (Arima(tsmonthly, order = c(p, 0, q), seasonal = list(order = c(P, 0, Q), period = 12)))
}, error = function(e) {
  # Handler for errors
  #print("Error")
  return((Arima(tsmonthly, order = c(0, 0, 0), seasonal = list(order = c(0, 0, 0), period = 1
2))))
})
    #print(c(AIC(sarima_model),p,q,P,Q))
    if(AIC(sarima_model)<best){
      best=AIC(sarima_model)
      best_Index=c(p,q,P,Q)
    }
  }
}
}}
```

```
cat("Best Index",best_Index)
```

```
## Best Index 3 3 1 1
```

Best AIC is -54. something at (3,3 1,1)

###SARMA(3,3)(1,1) S=12

So this is our Final Best model so far.

```
 final_fit<-(Arima(tsmonthly, order = c(3, 0, 3), seasonal = list(order = c(1, 0, 1), period = 1
2)))

final_fit
```

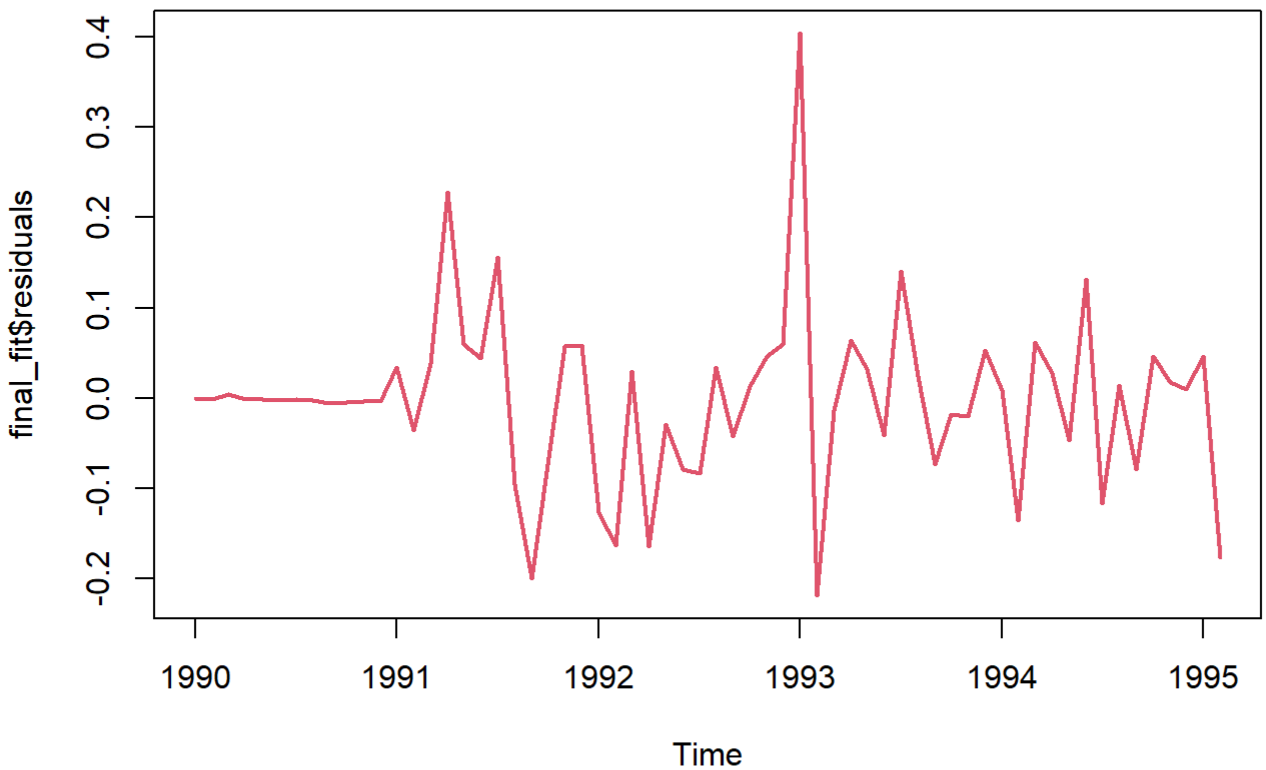
```
## Series: tsmonthly
## ARIMA(3,0,3)(1,0,1)[12] with non-zero mean
##
## Coefficients:
```

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

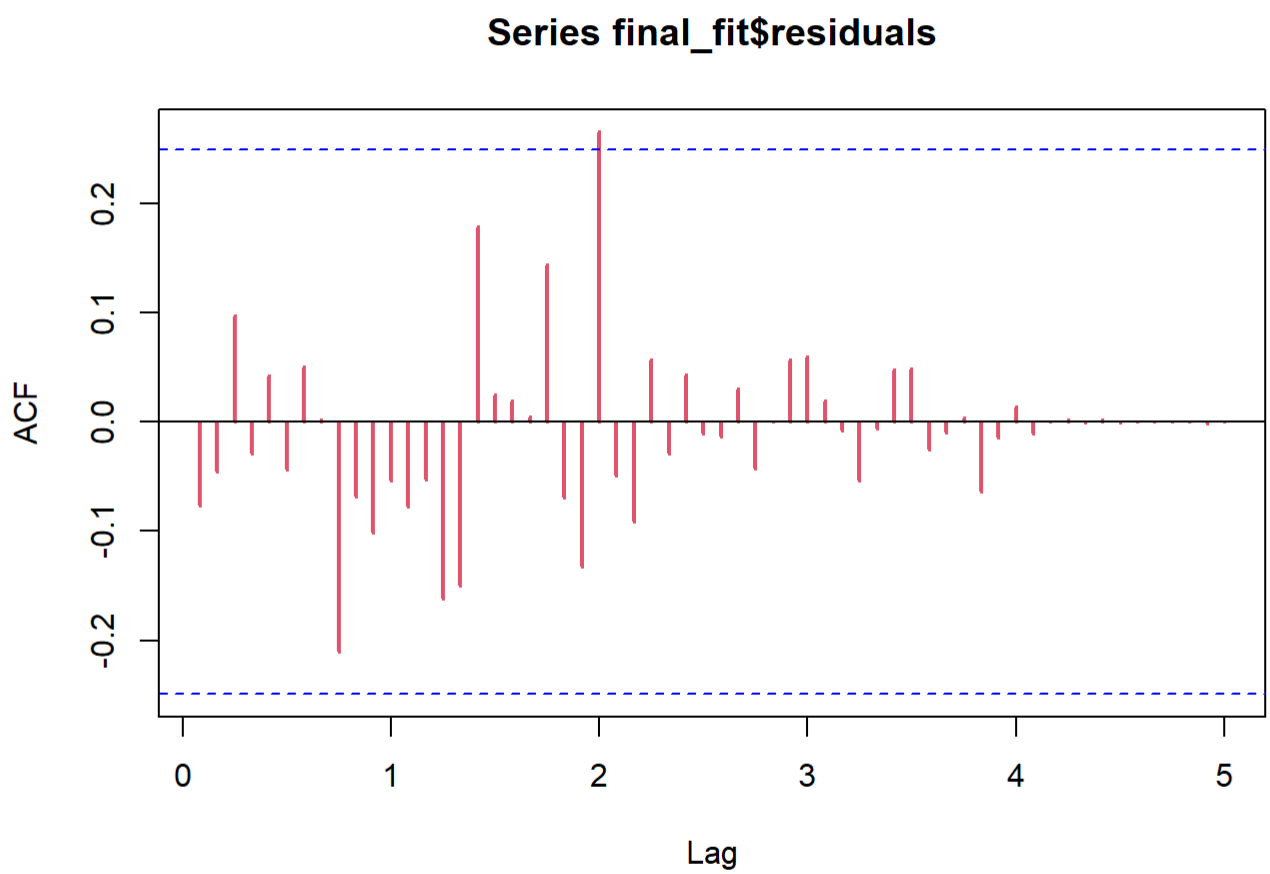
```
##          ar1      ar2     ar3      ma1     ma2     ma3  sar1    sma1       mean
##       2.7183  -2.7066  0.9881  -1.3367  0.3972  0.2679     1  0.6375    28.2103
## s.e.     NaN      NaN     NaN      NaN     NaN     NaN   NaN     NaN  8599.4628
##
## sigma^2 = 0.01109:  log likelihood = 39.03
## AIC=-58.06   AICc=-53.75   BIC=-36.79
```

Step 5: Residule Analysis

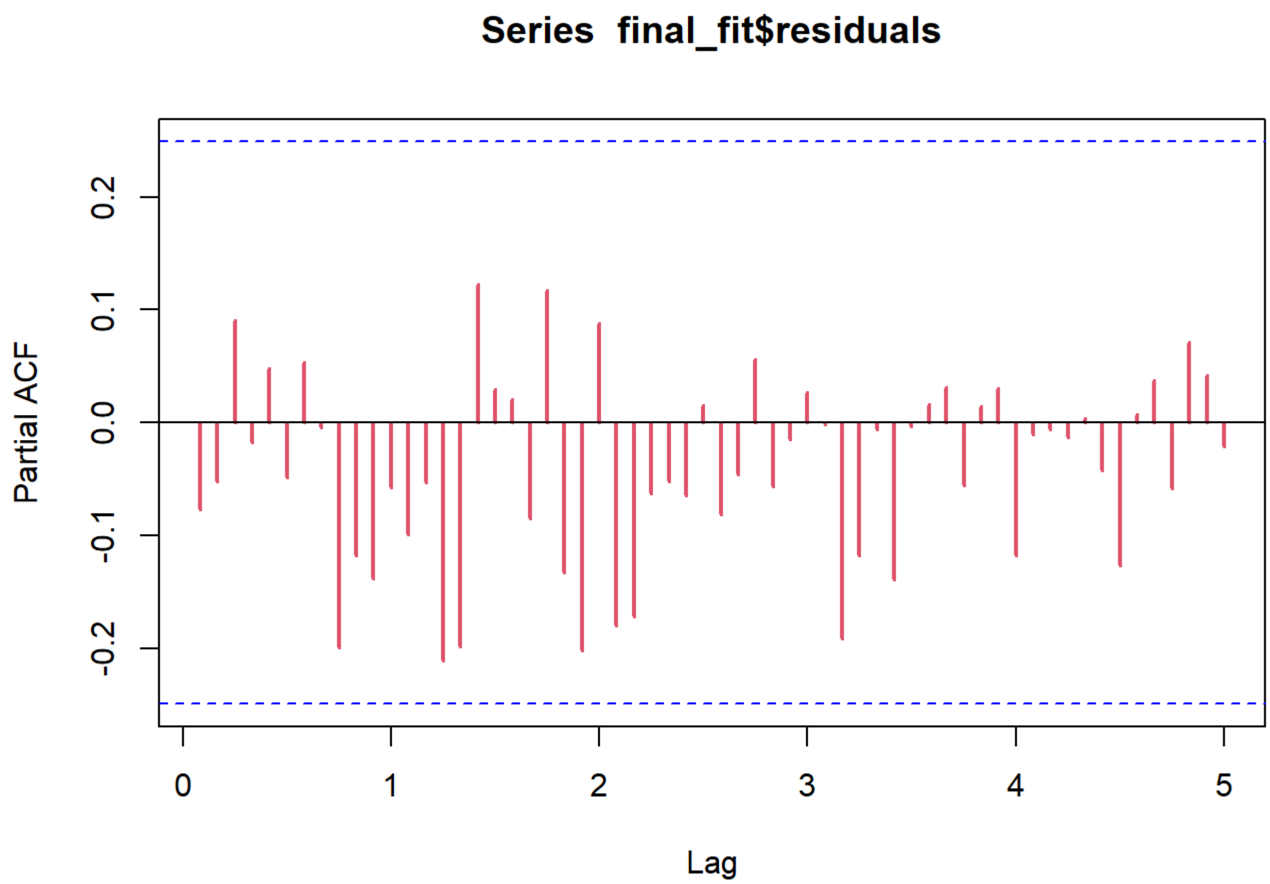
```
plot(final_fit$residuals,col=10,lwd=2)
```



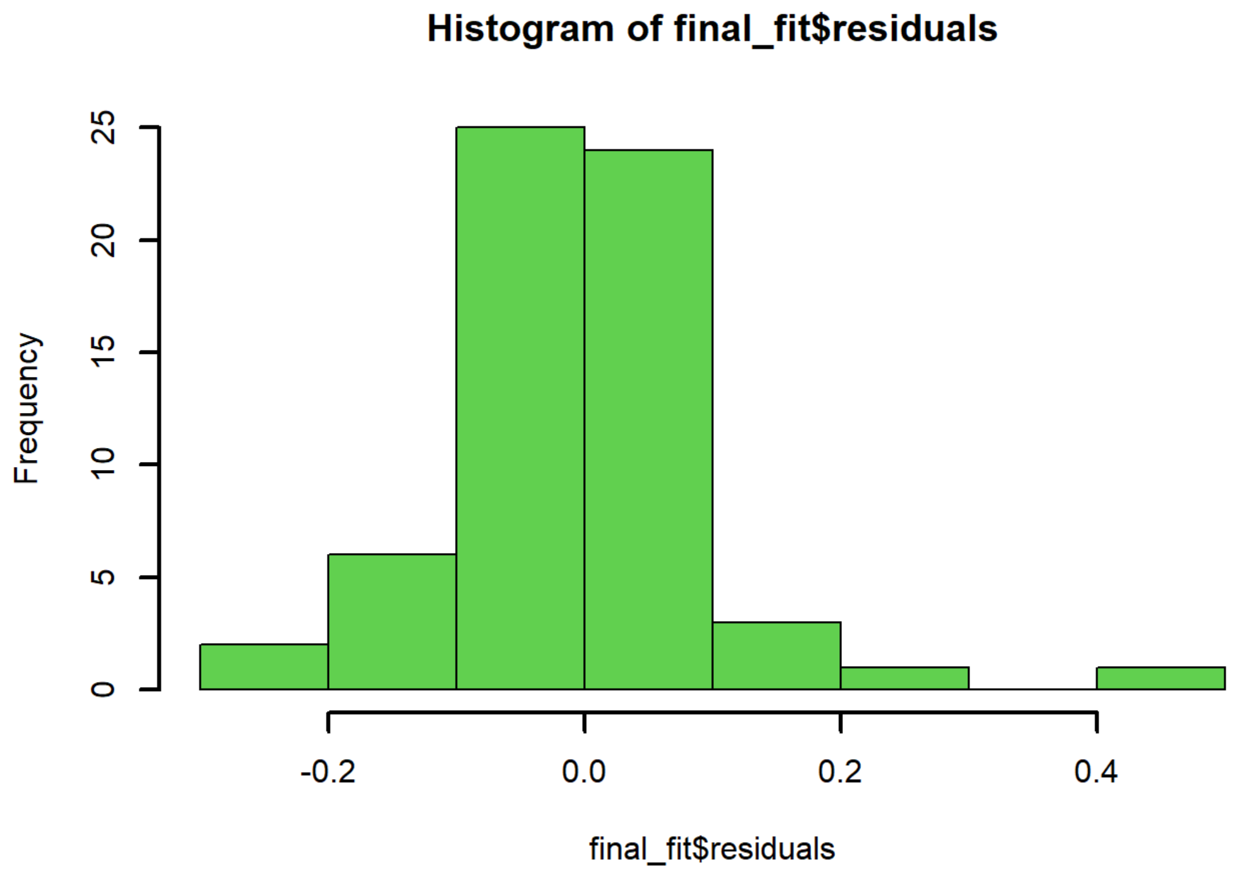

```
acf(final_fit$residuals, lag.max = 60,col=010,lwd=2)
```



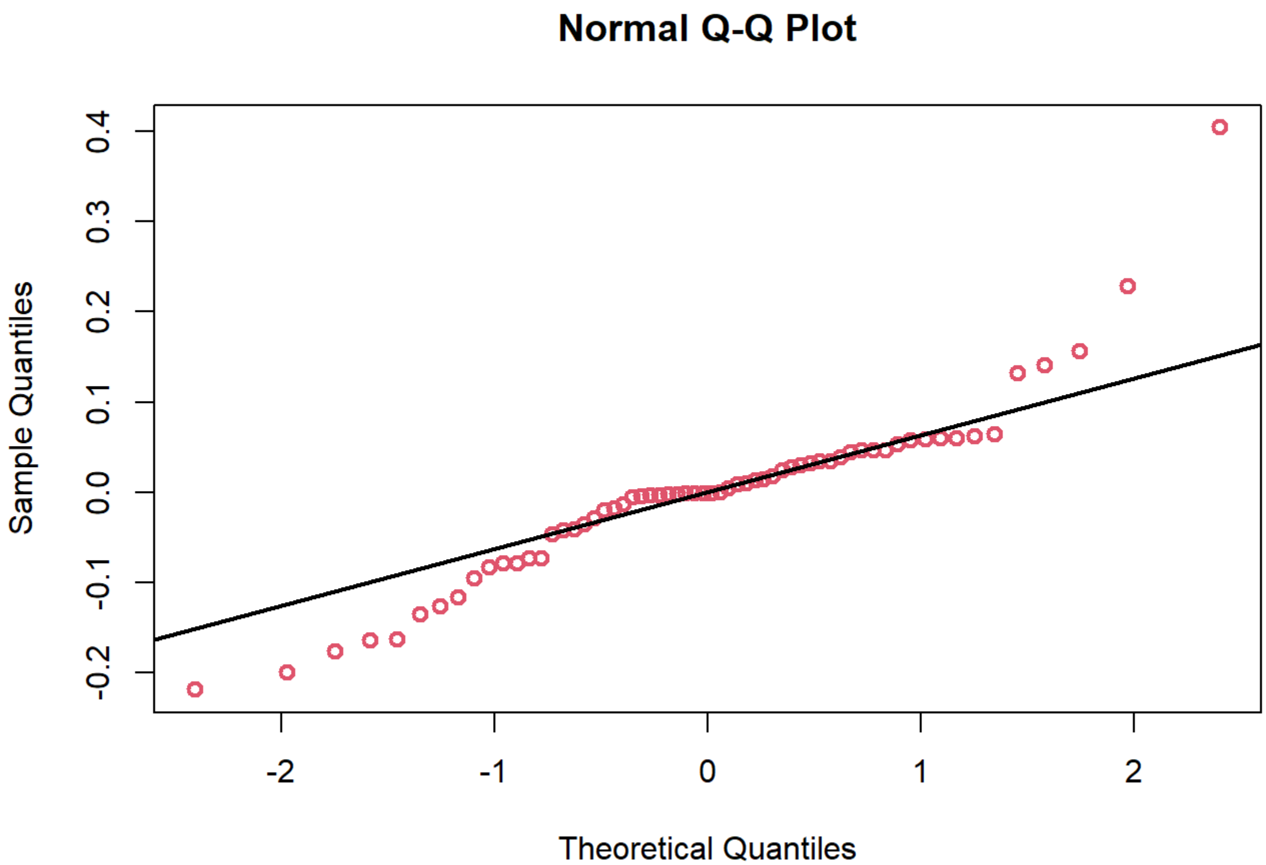

```
pacf(final_fit$residuals, lag.max = 60,col=10,lwd=2)
```




```r
#plot time series data
hist(final_fit$residuals,col = 3,lwd=2)
```




```
qqnorm(final_fit$residuals, col=2,lwd=2)
qqline(final_fit$residuals, col=9,lwd=2)
```

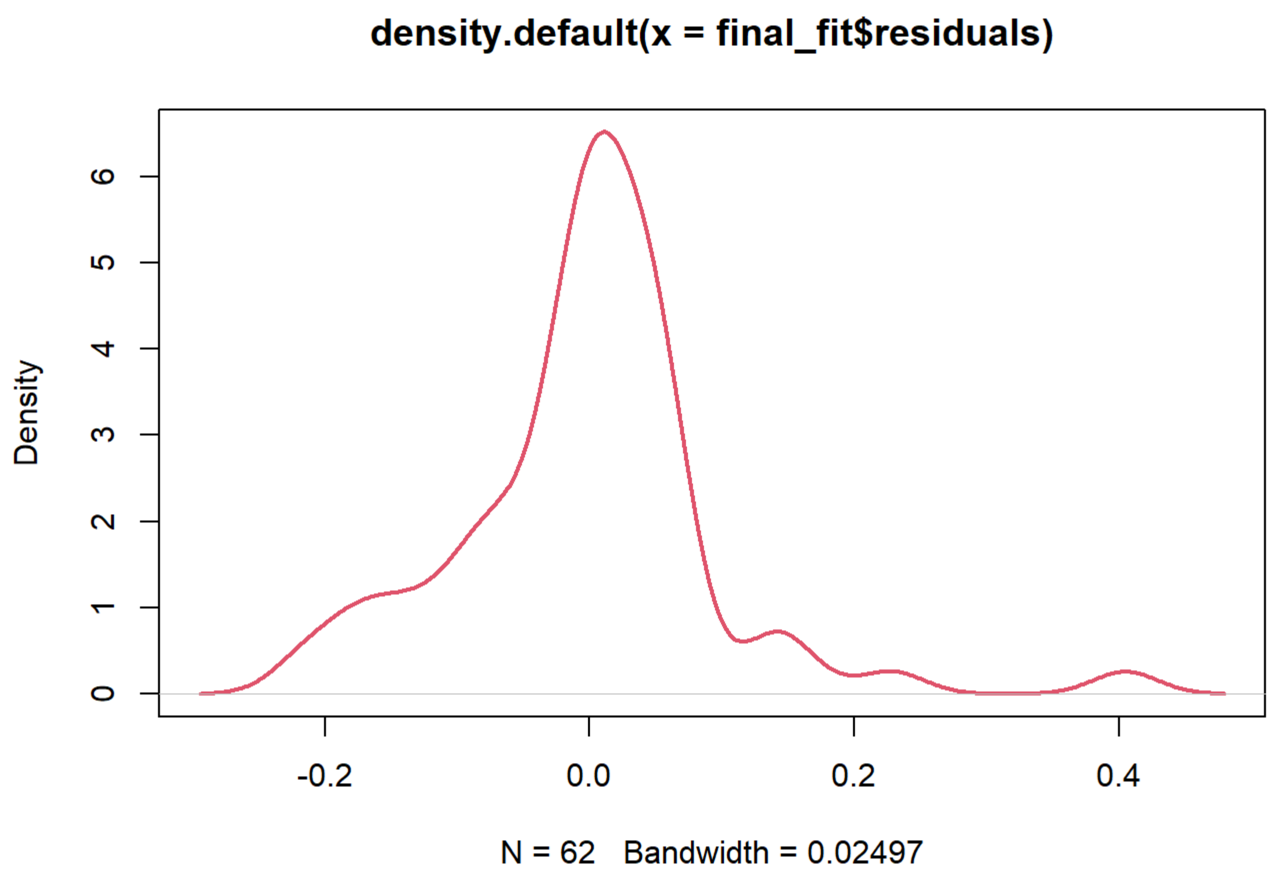



```
shapiro.test(final_fit$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  final_fit$residuals
## W = 0.9043, p-value = 0.0001484
```

The residuals are not normally distributed. overall performance of model seems good. we will move ahed with LB test.

```
plot(density(final_fit$residuals),col=10, lwd=2)
```

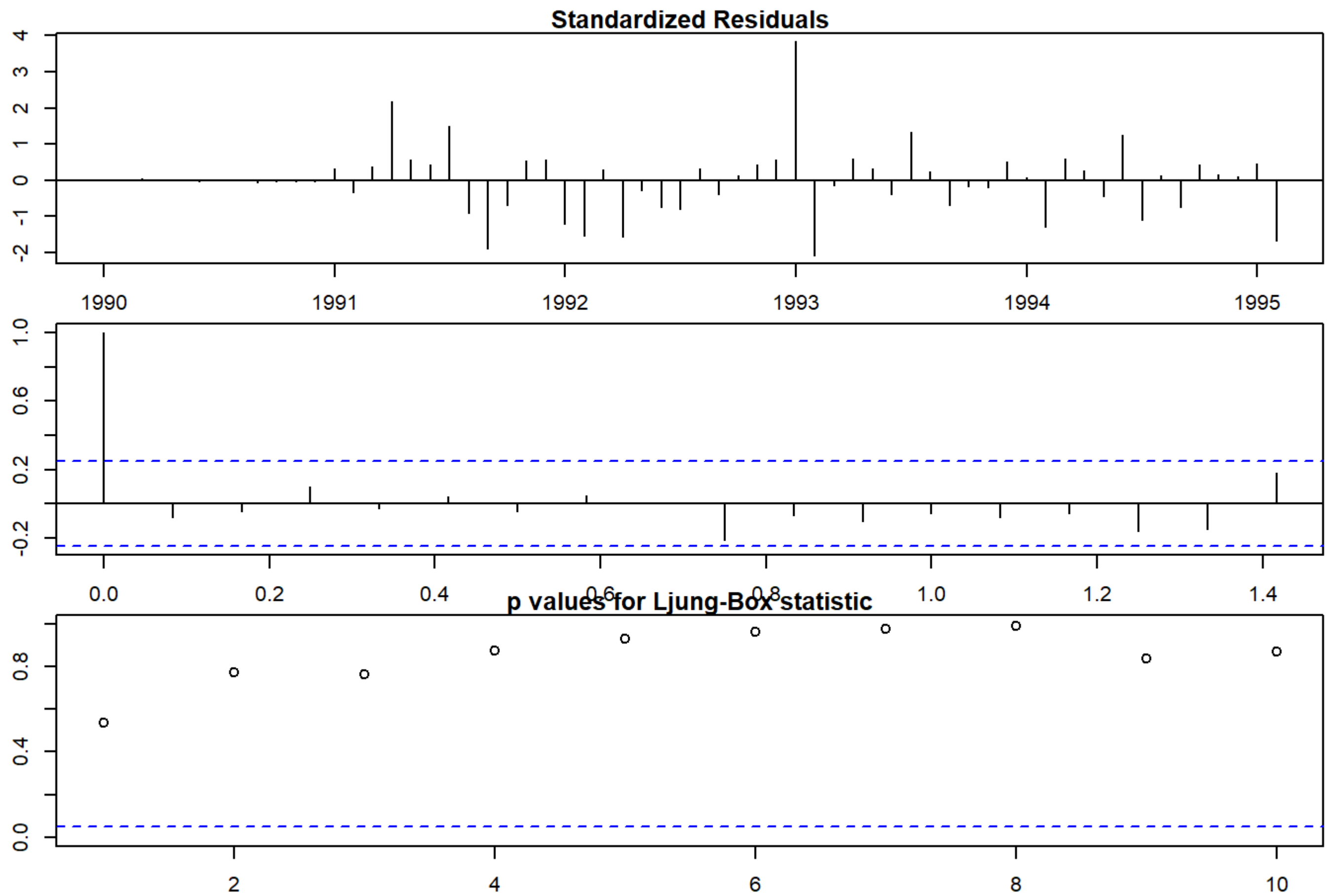



```
Box.test(final_fit$residuals, lag = 20, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  final_fit$residuals
## X-squared = 14.118, df = 20, p-value = 0.8245
```

p-value is 0.8245, indicating that there is no evidence of significant autocorrelation in the residuals. This is a good result as it suggests that the model is adequately capturing the structure of the data and there are no significant patterns left in the residuals.

```
par(mfrow = c(1, 1), mar = c(1, 0, 1, 0) + 0.2, oma = c(1, 2, 2, 0))
tsdiag(final_fit, main = "Residuals of Ljung-Box Test")
```

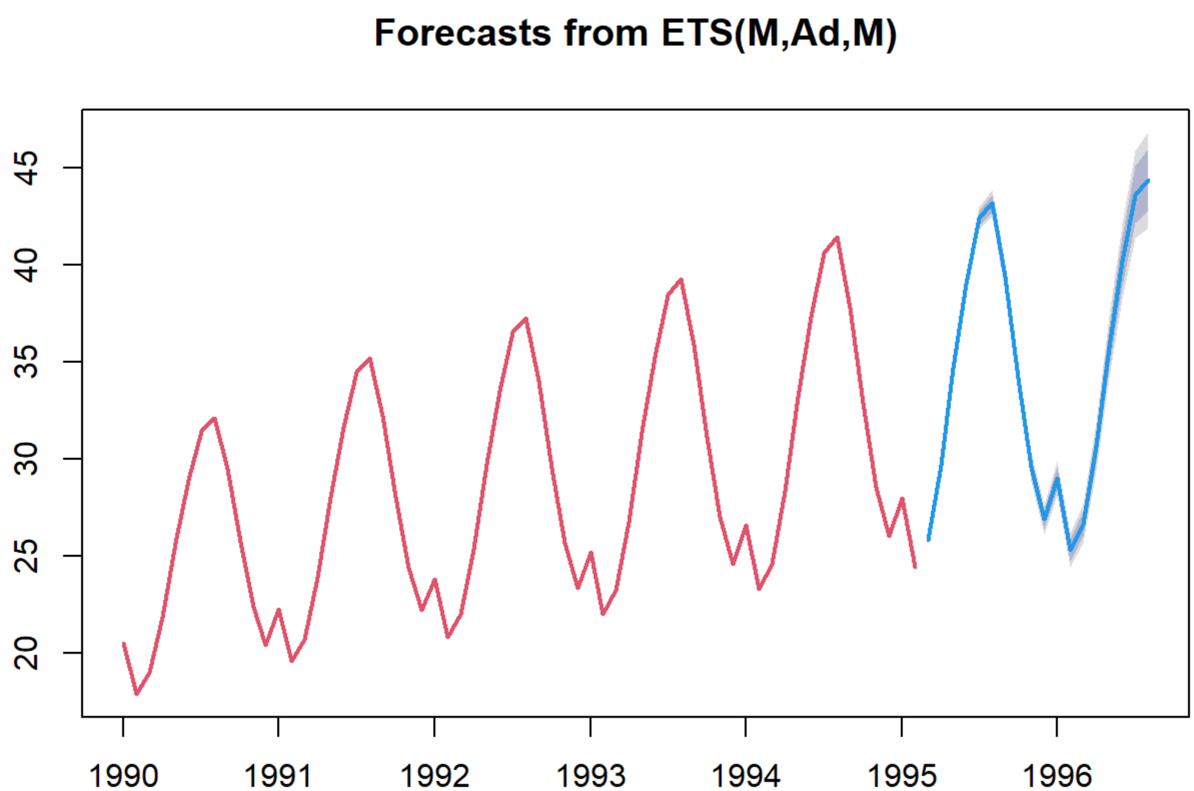


Step 6: Forecasting

```
library(forecast)


# Generate a 18-month forecast from the SARIMA model
forecast_data <- forecast(tsmonthly, h = 18)

# Plot the forecasted values
plot(forecast_data,col = 2,lwd=2)
```



Conclusion: In this project, time series data was analyzed using various models, including AR(2), ARMA(2,1), and SARMA(3,3)(1,1) with a seasonal period of 12. The SARMA(3,3)(1,1) model was found to be the best fit with an AIC value of -54. The model was further evaluated using residual analysis techniques, including ACF plot, histogram, qq plot, Shapiro Wiki test, and Ljung-Box test. The results of these tests showed that the model was a good fit for the data. Finally, the model was used to forecast future values based on the original data. Overall, the results of this analysis suggest that the SARMA(3,3)(1,1) model is a suitable approach for modeling and forecasting the given time series data.