Mini Project – Time Series

Table of Contents

1. Project Objective 3
2. Step by step approach 4
   2.1. Environment Set up and Data Import 4
   2.2. Variable Identification 4
   2.3. Identify Components of Time Series 5
   2.4. Identify Periodicity of Time Series 5
   2.5. Perform Stationary Test 5
   2.6. Develop ARIMA Modal 9
   2.7. Test the accuracy of the model 12
3. Appendix A – Source Code 14

1. Project Objective

The objective of the report is to explore **gas** (Australian monthly gas production) dataset in Forecast package in R and prepare a Managerial Report by explaining the following points. This Managerial report consists of the following:

- Business Objective Statement
- Read the data as time series Object data
- Components of the time series object data
- Periodicity of the time series object data
- Time series Stationary test
- De-seasonalise Time Series data
- Develop ARIMA model
- Report the accuracy of the model

The sample gas dataset is:

Year	Month	Gas Data
1956	Jan	1709
1956	Feb	1646
1956	Mar	1794
1956	Apr	1878
1956	May	2173
1956	Jun	2321
1956	Jul	2468
1956	Aug	2416
1956	Sep	2184
1956	Oct	2121
1956	Nov	1962
1956	Dec	1825
1957	Jan	1751
1957	Feb	1688
1957	Mar	1920
1957	Apr	1941
1957	May	2311
1957	Jun	2279
1957	Jul	2638
1957	Aug	2448
1957	Sep	2279
1957	Oct	2163
1957	Nov	1941
1957	Dec	1878

Note: The provided data set has 476 rows of Monthly data from 1956 to 1995.

Business Objective: To build a model that helps to predict future Australian gas production for the next 12 periods.

2. Step by step approach

We shall follow a step by step approach to arrive to the final conclusion as follows:

1. Environment set up and Data import
2. Read the data as time series Object data
3. Identify the components of time series data
4. Identify the periodicity of time series data
5. Perform Stationary test of time series data
6. Develop ARIMA model
7. Test the accuracy of the model
8. Conclusion

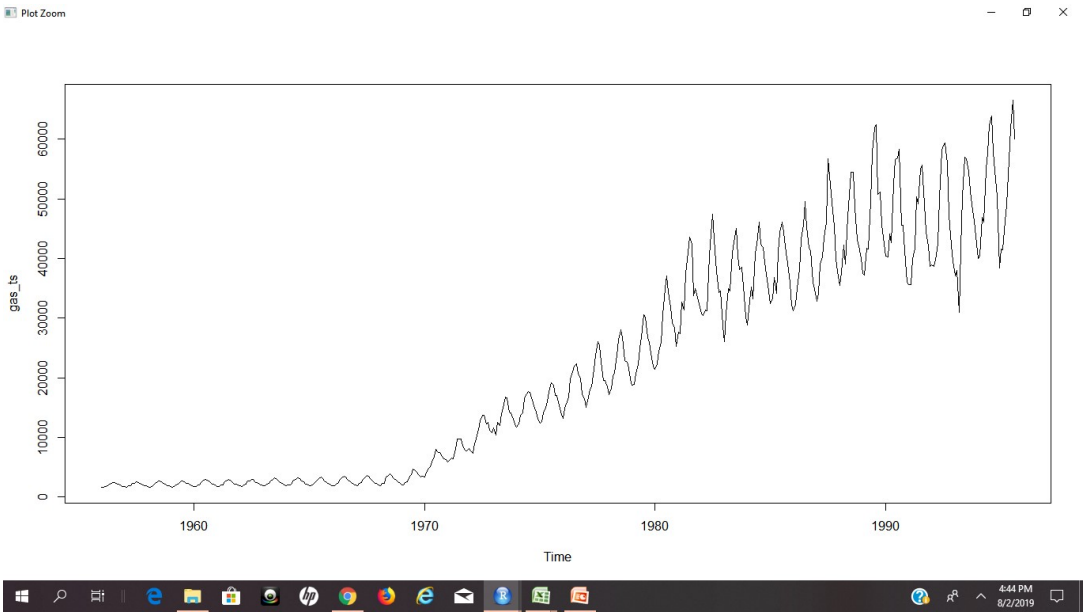
2.1. Environment Set up and Data Import

Please refer Appendix A for Source Code.

2.2. Variable Identification

Read the data as time series object data and plot the time series data.

```
gas_ts = ts(gasData[,3], start = c(1956,1), frequency = 12)

plot(gas_ts)
```



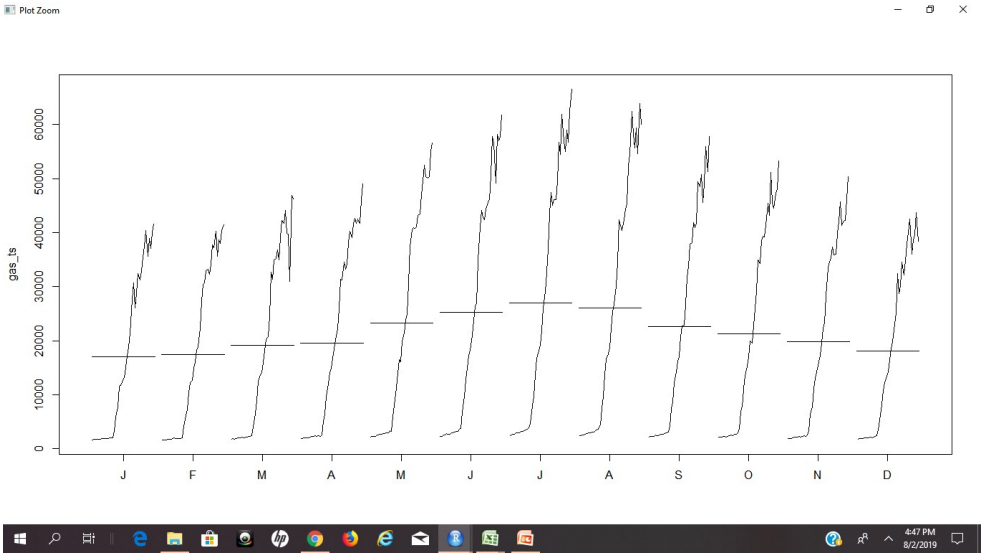
2.3. Identify Components of Time Series data

As per the above graph, time series data shows both trend and seasonality.

Please refer Appendix A for Source Code.

2.4. Identify the periodicity of Time Series data

Given time series data is depicted as monthly time series data and the below monthplot shows the monthwise trend and seasonality of the time series data.

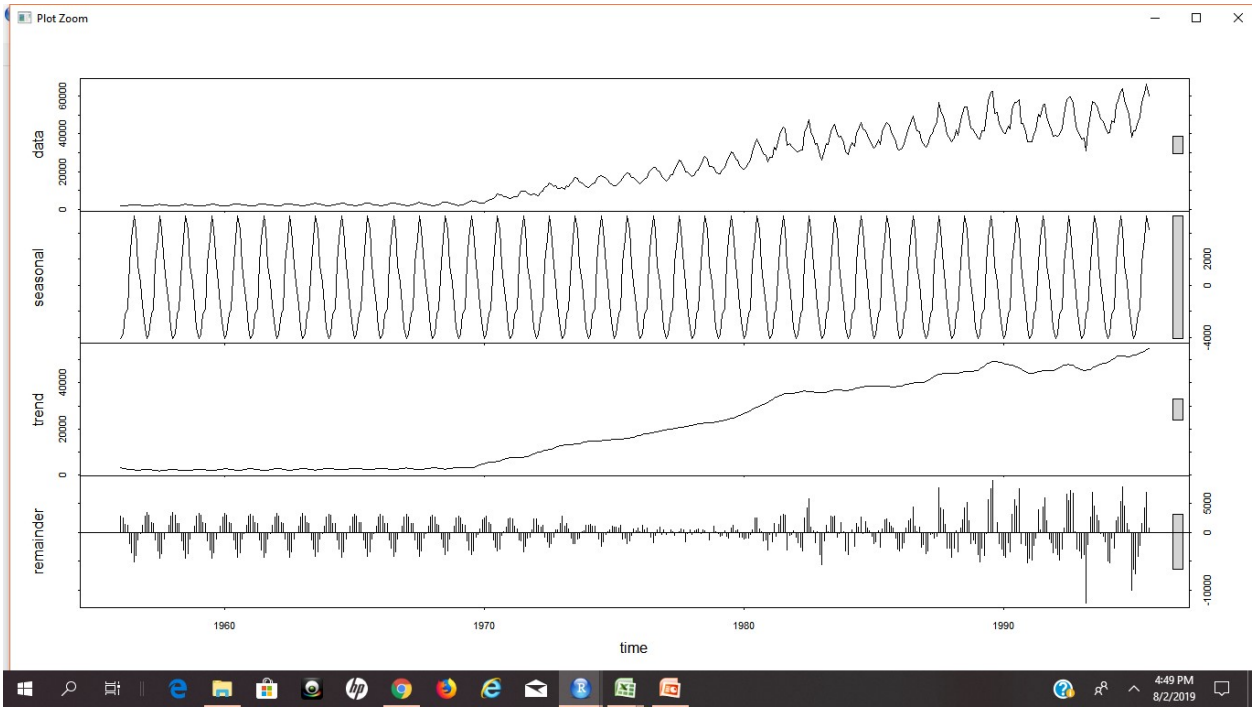


Please refer Appendix A for Source Code.

2.5. Perform Stationary test

Since, seasonality is present in the data, decompose the data to calculate the seasonality component using smoothing and then adjust data by removing seasonality.

Below plot shows the decomposed data.



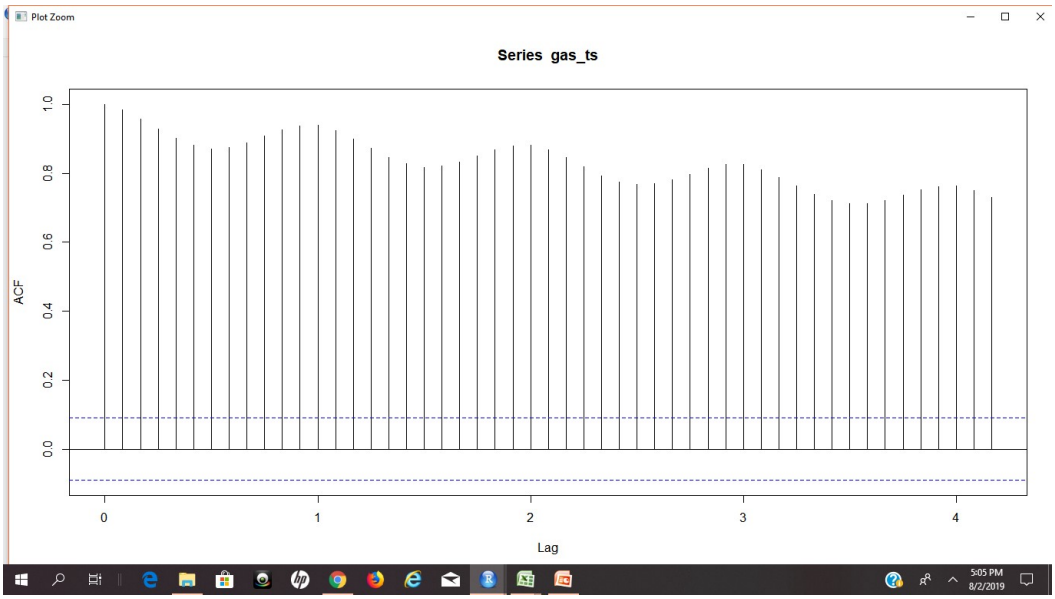
Now, Check whether the series is stationary or not using Augmented Dickey-Fuller Test. Formulate the hypothesis as:
H0: Data is not Stationary
Ha: Data is stationary

```
> adf.test(gas_ts)

        Augmented Dickey-Fuller Test

data:  gas_ts
Dickey-Fuller = -2.7131, Lag order = 7, p-value = 0.2764
alternative hypothesis: stationary
```

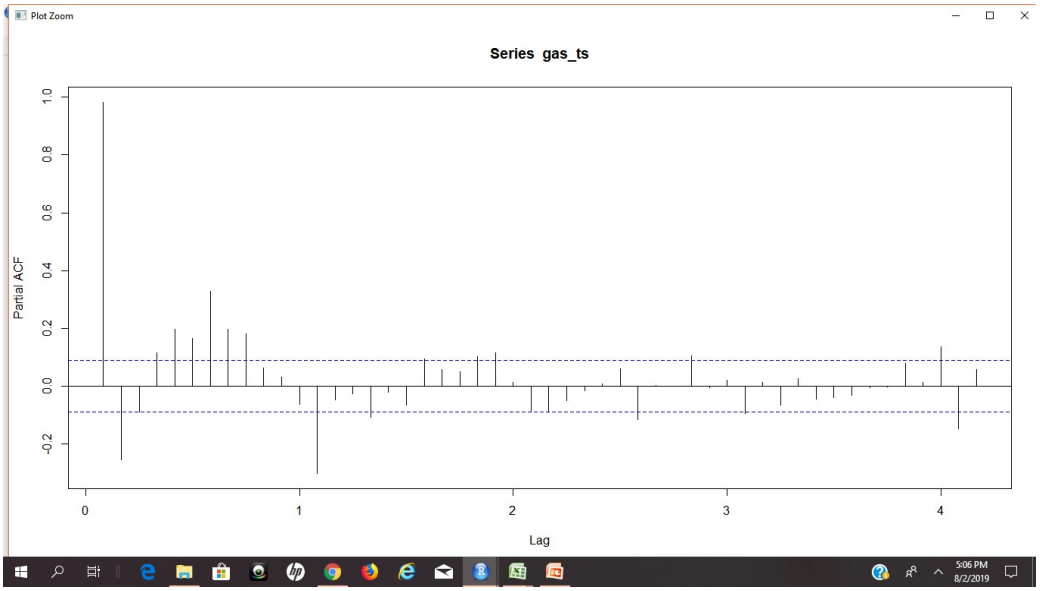
As p-value: 0.2764 > 0.05, Null hypothesis is accepted
That is Data is not stationary.

Below Autocorrelation plots also confirm that the time series data is not stationary.

ACF Plot of time series data



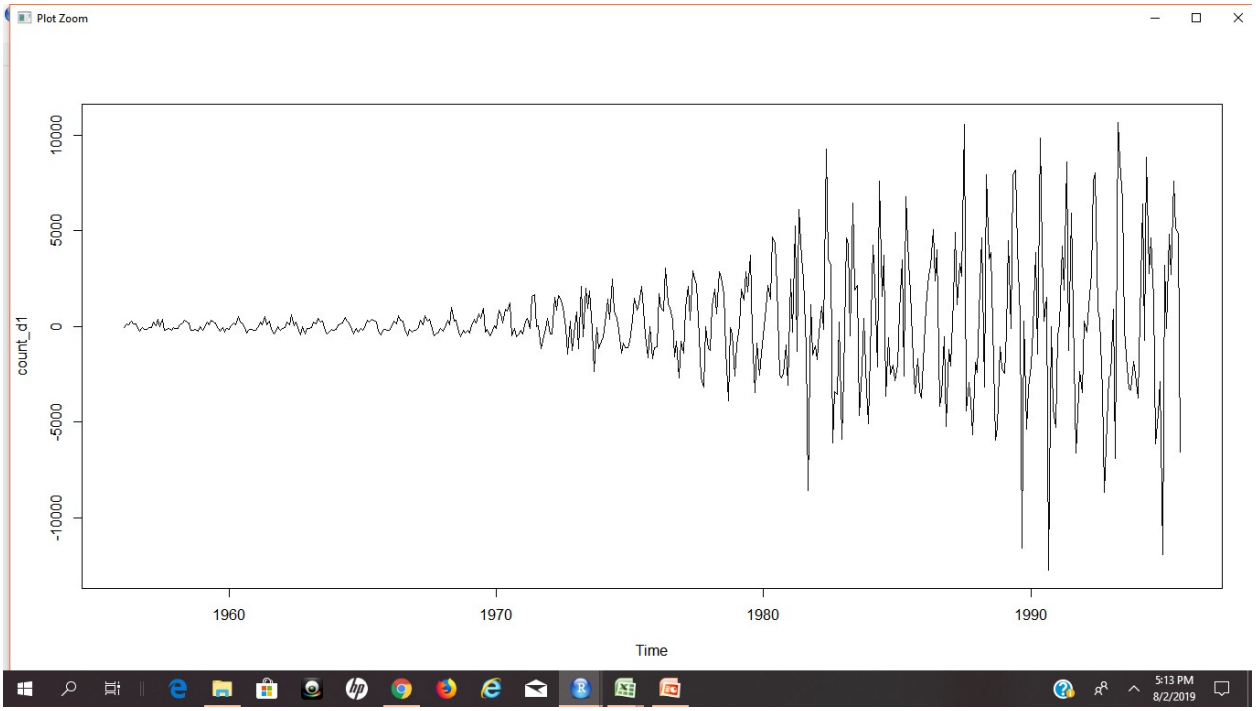
PACF plot of Time series data



Though, there are some significant periods outside the control line, the obvious significant line is the first period, hence, difference the series in the order of 1 to remove the trend and seasonality from the data.

```
count_d1 = diff(gas_ts, differences = 1)
plot(count_d1)
```

The time series plot after removing the trend and seasonality of data is



Plot shows that series is now having constant mean.

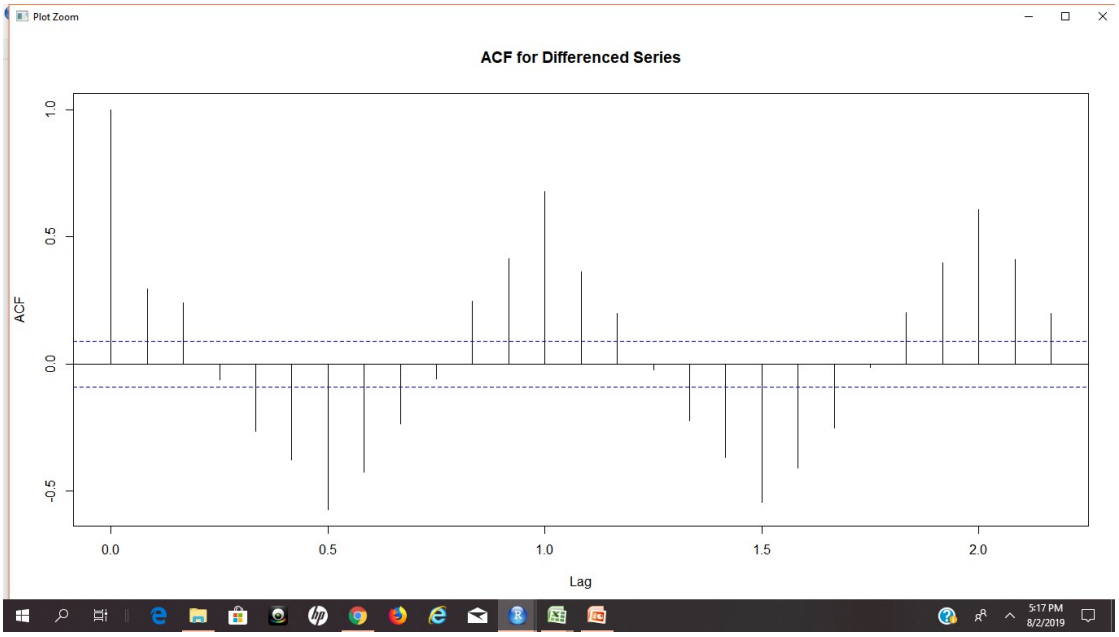
Now, test if the differenced data is stationary or not.

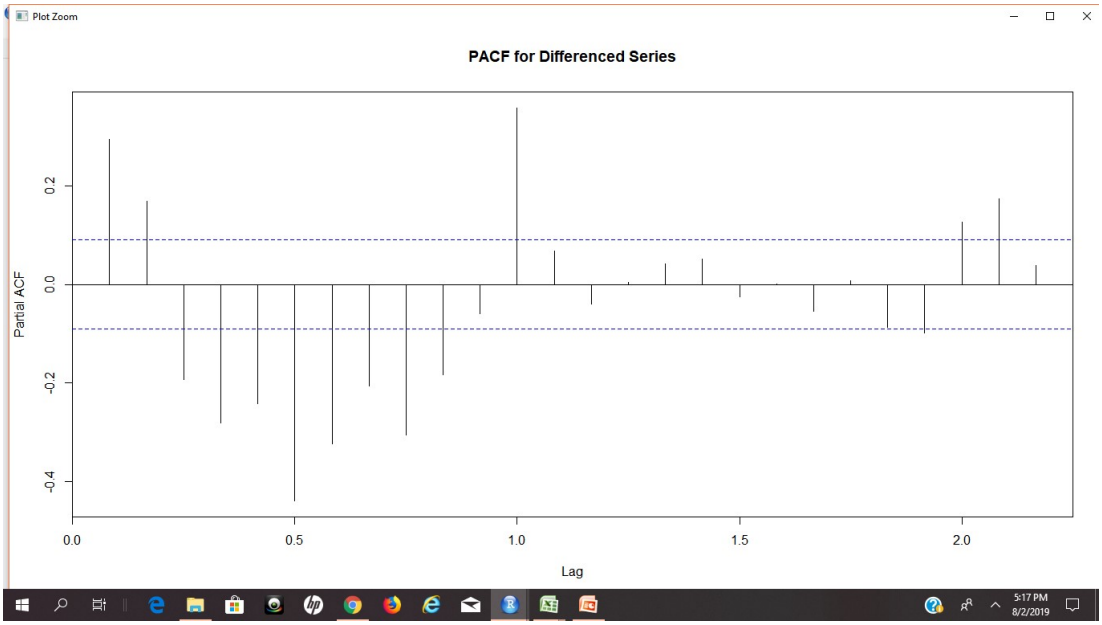
```
> adf.test(count_d1)

        Augmented Dickey-Fuller Test

data:  count_d1
Dickey-Fuller = -19.321, Lag order = 7, p-value = 0.01
alternative hypothesis: stationary

Warning message:
In adf.test(count_d1) : p-value smaller than printed p-value
```

p-Value : 0.01 < 0.05, hence reject the null hypothesis and accept the alternate hypothesis. Now the series is Stationary.

ACF and PACF plots also confirm the series is stationary.

```
acf(count_d1, main="ACF for Differenced Series")
pacf(count_d1, main="PACF for Differenced Series")
```





However, as per the plots, there exists correlation and seasonality between the lags.

Please refer Appendix A for Source Code.

2.6. Develop ARIMA Model

Divide the Stationary series into Train and Test Data.

gasTrain = window(ts(deseasonal_demand[c(1:465)]))

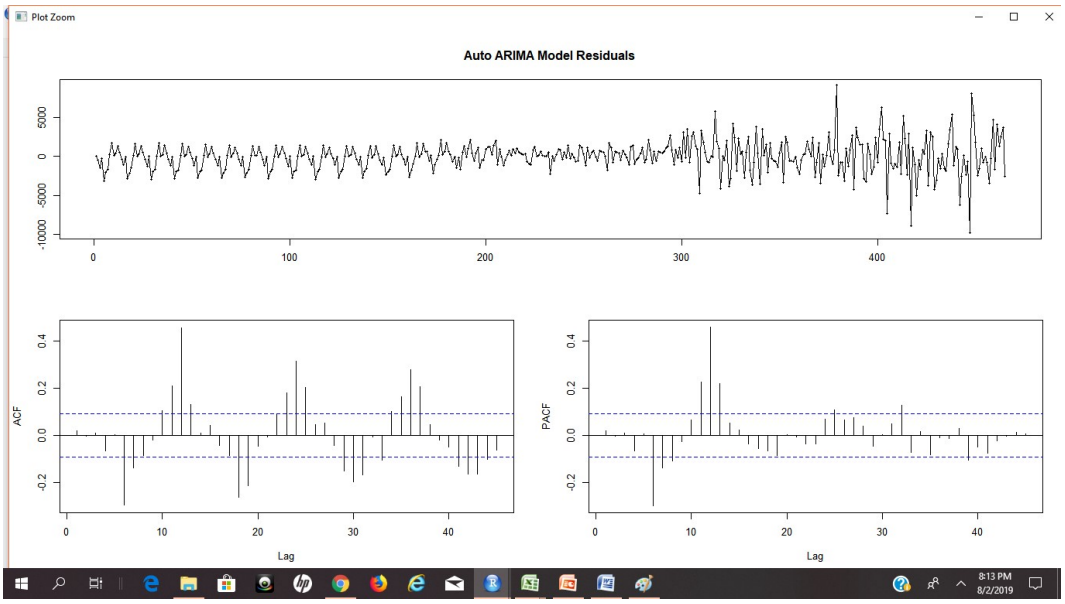
gasTest= window(ts(deseasonal_demand), start=466)

Use auto.arima function to automatically generate optimal (p,d,q) order to build the model.

fit<-auto.arima(gasTrain, seasonal=FALSE)

```
Series: gasTrain
ARIMA(1,1,5) with drift

Coefficients:
          ar1      ma1     ma2      ma3      ma4      ma5     drift
      0.4931  -0.5640  0.0984  -0.2225  -0.0449  -0.1296  107.8536
s.e.  0.0913   0.0931  0.0639   0.0703   0.0701   0.0534   24.7937

sigma^2 estimated as 3725403:  log likelihood=-4165.56
AIC=8347.11   AICc=8347.43   BIC=8380.23
```

To Evaluate the fitted model, Draw the ACF and PACF plots of model residuals

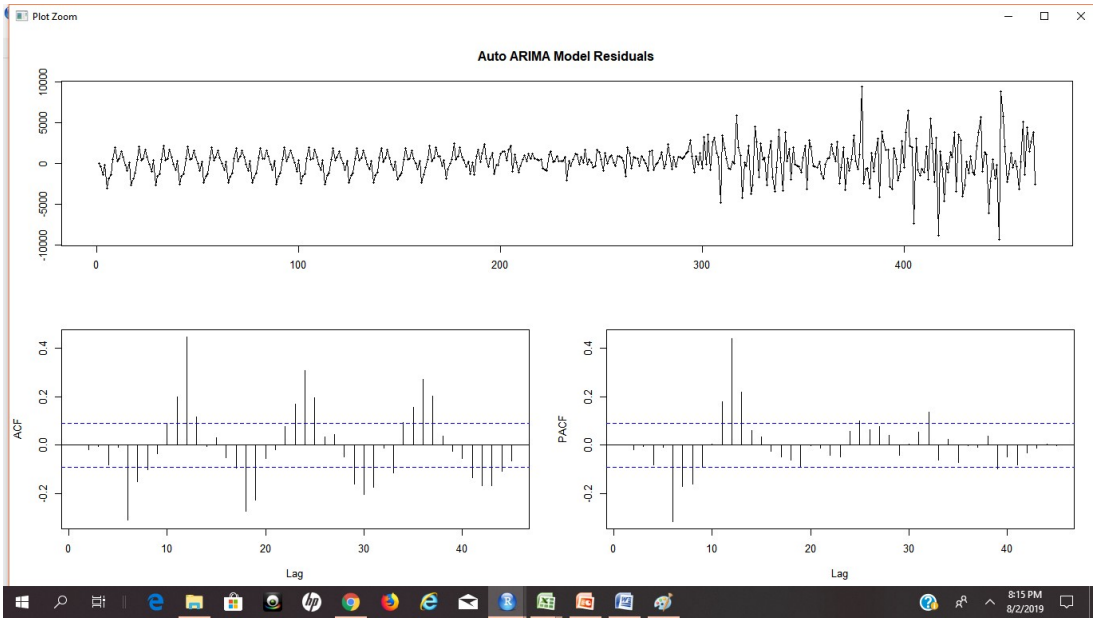


Auto ARIMA Model suggests the order to be ARIMA(1,1,5) with drift.

fit1<-arima(gasTrain, order=c(1,1,5))

tsdisplay(residuals(fit1), lag.max=45, main='Auto ARIMA Model Residuals')

Now the residual plots are:



Also, the residuals are normally distributed as per Ljung box test.

Hypothesis for Ljung box test is:

H0: Residuals are independent

Ha: Residuals are not independent

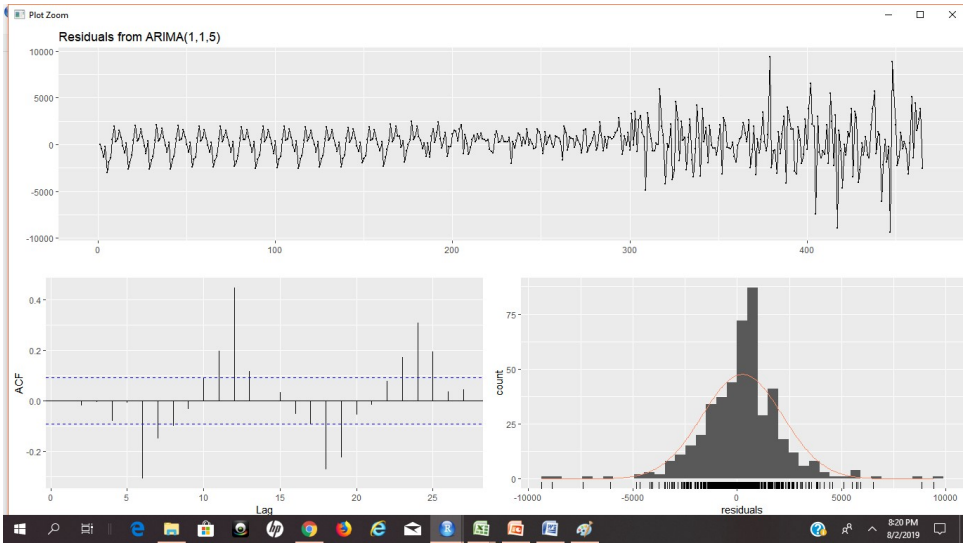
Box.test(fit1$residuals)

```
> Box.test(fit1$residuals)

        Box-Pierce test

data:  fit1$residuals
X-squared = 0.00022843, df = 1, p-value = 0.9879
```

p-value: 0.9879 > 0.05, hence residuals are independant.

Normality of the residuals

checkresiduals(fit1)



Please refer Appendix A for Source Code to build the CART model using Train Data set.

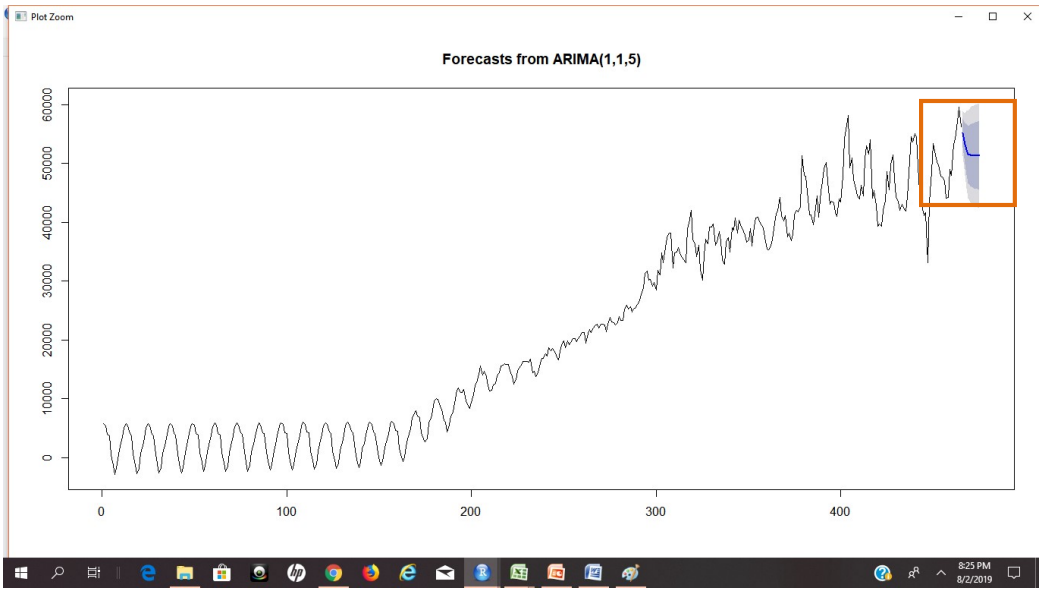
2.7. Test the accuracy of the model

Forecasting with ARIMA model

fcast <- forecast(fit1)

plot(fcast)

The forecasted model is shown in the graph with 80% and 95% confidence intervals.



Test Accuracy of the forecast using

accuracy(fcast, gasTest)

```
> accuracy(fcast, gasTest)
                    ME     RMSE      MAE         MPE     MAPE      MASE          ACF1
Training set  255.0977 1939.491 1375.711 -15.167002 53.71842 0.9006448 0.0007008927
Test set     -969.6835 6010.661 5024.363  -3.209061 10.07902 3.2893302 0.5985662117
             Theil's U
Training set        NA
Test set      1.412314
```

High value of MAPE 53.71842 shows that the model is not adequate for future Predictions. Hence, the series further needs to be evaluated to get the better model. Also, the accuracy test shows, the MAPE values of both Train and Test data are not matching. Hence, the model needs to be further evaluated.

Please refer Appendix A for Source Code.

3. Appendix A – Source Code

```
#============================================================================
# Data Analysis - Gas Time Series
#============================================================================
#Environment Set up and Data Import
#Set up working Directory
setwd("C:/Users/Radhika/Desktop/R Programming/Project_TimeSeries")
getwd()
#
#Import the required packages and install them
install.packages("tseries")
library('tseries')
install.packages("forecast")
library("forecast")
#read the data from gas object from forecast package
data("gas")
#Write the data into a CSV file and modify the csv file manually
#To show the yearwise data and read the data from CSV file into gasData object
write.csv(gas, file="gas.csv")
gasData = read.csv("gas.csv")
#Convert the data into Time Series data
gas_ts = ts(gasData[,3], start = c(1956,1), frequency = 12)
#Plot Time Series data
plot(gas_ts)
#Time series plot shows both trend and seasonality in the data
#Draw month plot to check the seasonality of data and to show the periodicity
monthplot(gas_ts)
#Decompose the data to calculate the seasonality component using smoothing
decompGas = stl(gas_ts, s.window = "p")
#Adjust data by removing the seasonality component
#And plot the sesaonality
deseasonal_demand=seasadj(decompGas)
plot(decompGas)
#Check whether the series is stationary or not using Augmented Dickey-Fuller Test
#H0: Data is not Stationary
#Ha: Data is Stationary
adf.test(gas_ts)
#As p-value: 0.2764 > 0.05, Null hypothesis is accepted
#That is Data is not stationary.
#
#Check the autocorrelation plots
acf(gas_ts, lag = 50)
#ACF plot shows the data is not stationary and there exists trend and seasonality
#between the lags
#Check PACF plot to see the correlation between the variable and its lags
pacf(gas_ts, lag = 50)
#As the PACF plot suggests, difference the series in order of 1
#to remove the trend and seasonality from data
count_d1 = diff(gas_ts, differences = 1)
count_d1
plot(count_d1)
#Plot shows that series is now having constant mean
#Test for Stationary
#H0: Data is not Stationary
#Ha: Data is stationary
adf.test(count_d1)
#p-Value : 0.01 < 0.05, hence reject the null hypothesis and
# Accept the alternate hypothesis, now the series is Stationary
#Draw the ACF and PACF plots to confirm the stationary series
acf(count_d1, main="ACF for Differenced Series")
pacf(count_d1, main="PACF for Differenced Series")
# As per the plots, there exists correlation between the lags and sesonality
#Divide the Stationary series into Train and Test Data
gasTrain = window(ts(deseasonal_demand[c(1:465)]))
gasTest= window(ts(deseasonal_demand), start=466)
#Use auto.arima function to automatically generate optimal (p,d,q) order
#To forecast the model.
fit<-auto.arima(gasTrain, seasonal=FALSE)
fit
#It suggests the order to be ARIMA(1,1,5) with drift
#To Evaluate the fitted model, Draw the ACF and PACF plots of model residuals
tsdisplay(residuals(fit), lag.max=45, main='Auto ARIMA Model Residuals')
```

```r
fit1<-arima(gasTrain, order=c(1,1,5))
fit1
tsdisplay(residuals(fit1), lag.max=45, main='Auto ARIMA Model Residuals')
#
#Ljung box test to check whether the residuals are independant or not
#H0: Residuals are independent
#Ha: Residuals are not independent
library(stats)
Box.test(fit1$residuals)
#p-value: 0.9879 > 0.05, hence residuals are independant.
#Normality of the residuals
checkresiduals(fit1)
#Forecasting with ARIMA model
fcast <- forecast(fit1)
#plot the forecast model
plot(fcast)
#Test Accuracy of the forecast
accuracy(fcast, gasTest)
#High value of MAPE 53.71842 shows that the model is not adequate for future
#Predictions. Hence, the series is further needs to be evaluated to get
#the better model. Also, the accuracy test shows, the MAPE values of both
#Train and Test data are not matching. Hence, the model needs to be further
#evaluated.
```