

# Mini Project – Australian monthly gas production

Project Report

Jyosmitha M

## Contents

1. Project Objective.....	2
2. Assumptions.....	2
3. Environment Set up and Data Import.....	2
3.1 Install necessary Packages and Invoke Libraries.....	2
3.2 Functions used in R-code: .....	2
3.3 Import and Read the Dataset.....	3
4. Meta Data: .....	3
5. Exploratory Data Analysis .....	3
5.1 Rows and columns: .....	3
5.2 EDA on the data: .....	4
5.2 Missing Values:.....	4
5.3 Structure of the dataset:.....	4
5.4 Five point Summary: .....	4
5.5 Outliers:.....	4
6. Plotting and Observations:.....	5
6.1 Time Series plot:.....	5
6.2 Seasonal plot:.....	6
6.2 Subseries plot:.....	8
6.3 Boxplot: .....	8
6.4 Lagplot:.....	8
7. Periodicity: .....	9
8. Subsetting the data:.....	10
8.1 Various charts with the subset of dataset: .....	11
8.2 Decomposing the dataset: .....	13
9. Is the time series Stationary?.....	14
9.1 Dickey Fuller Test: .....	15
9.2 Removing seasonality: .....	15
9.3 Removing Trend Component: .....	17
10. Building ARIMA Model Manually: .....	18
11. Building model with auto.arima: .....	24
12. Accuracy of the models:.....	27

13.	Forecasting model for 12 periods: .....	29
-----	---	----

## 1. Project Objective

To forecast the Australian Monthly Gas Production for the given time duration by developing a Machine learning model using Time Series data. Below are the items that will be

- Read the data as a time series object in R. Plot the data
- What do you observe? Which components of the time series are present in this dataset?
- What is the periodicity of dataset?
- Is the time series Stationary? Inspect visually as well as conduct an ADF test? Write down the null and alternate hypothesis for the stationarity test? De-seasonalise the series if seasonality is present?
- Develop an ARIMA Model to forecast for next 12 periods. Use both manual and auto.arima (Show & explain all the steps)
- Report the accuracy of the model

## 2. Assumptions

### Time Series:

- All the observations are dependent on time
- Observations must be in ascending order
- Data is auto correlated
- No missing values should be present
- Forecast should not be done for a really long period
- Data should be sufficient to make a forecast

## 3. Environment Set up and Data Import

### 3.1 Install necessary Packages and Invoke Libraries

- `library(forecast)`
- `library(tseries)`
- `library(xts)`
- `library(ggplot2)`

### 3.2 Functions used in R-code:

- `cycle`
- `start`
- `end`
- `frequency`
- `aggregate`
- `decompose`
- `window`
- `seasadj`
- `acf`
- `adf.test`

- pacf
- accuracy
- forecast

### 3.3 Import and Read the Dataset

We need to import forecast library. This package contains methods and tools for displaying and analyzing univariate time series forecasts including exponential smoothing via state space models and automatic ARIMA modelling.

The imported Dataset is already of timeseries format. Hence no conversion is needed.

```
#Importing the Library
library(forecast)
library(tseries)
#reading the dataset
data<- forecast::gas
```

```
> str(data)
Time-Series [1:476] from 1956 to 1996: 1709 1646 1794 1878 2173 ...
> #Summary of entire Series
> summary(data)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1646   2675   16788   21415   38629   66600
> #Checking for null values
> anyNA(data)
[1] FALSE
```

## 4. Meta Data:

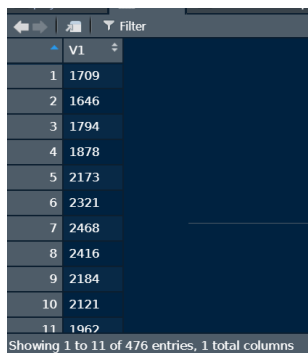
Only 2 columns are present in gas dataset

Column Name	Description
Year	Year of Gas Production
Gas Produced	Gas Produced

## 5. Exploratory Data Analysis

### 5.1 Rows and columns:

```
> View(data)
```



	V1
1	1709
2	1646
3	1794
4	1878
5	2173
6	2321
7	2468
8	2416
9	2184
10	2121
11	1962

Showing 1 to 11 of 476 entries, 1 total columns

The dataset contains 476 rows and one feature column

### 5.2 EDA on the data:

```
> #head and tail
> head(data)
      Jan  Feb  Mar  Apr  May  Jun
1956 1709 1646 1794 1878 2173 2321
> tail(data)
      Mar  Apr  May  Jun  Jul  Aug
1995 46287 49013 56624 61739 66600 60054
> #start and end of time series
> start(data)
[1] 1956    1
> end(data)
[1] 1995    8
```

### 5.2 Missing Values:

There are no missing values in the dataset.

```
> anyNA(data)
[1] FALSE
```

### 5.3 Structure of the dataset:

The data is in timeseries format from the year 1956 to 1996

```
> #Structure of the data
> str(data)
Time-Series [1:476] from 1956 to 1996: 1709 1646 1794 1878 2173 ...
```

### 5.4 Five point Summary:

Summary of the dataset

```
> #Summary of entire Series
> summary(data)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1646    2675   16788   21415   38629   66600
```

### 5.5 Outliers:

There are outliers present in the dataset. Below are the given indexes where they are present

```
> tsoutliers(data)
$index
 [1] 295 300 303 305 306 307 308 311 312 317 318 319 324 325 329 336 345 347
360
[20] 363 367 368 371 376 378 379 380 381 392 394 398 399 402 403 404 405 406
407
[39] 413 414 415 416 417 419 420 421 422 423 426 428 430 432 435 438 439 440
441
[58] 444 445 446 447 448 450 452 458 461 463 464 465 468 469 470 471 474 475
476

$replacements
```

[1]	35616.41	28874.17	30621.71	35026.08	35095.50	35865.60	34919.84	33436.75
[9]	33187.47	36969.79	38277.22	40636.03	33712.75	29899.63	41026.81	33795.84
[17]	40715.58	36226.35	35443.68	37058.81	45441.24	43856.07	37431.36	38730.97
[25]	44142.22	45673.21	45717.80	45260.06	51755.88	45927.37	39492.64	42583.49
[33]	49074.29	49375.88	48197.51	46444.82	45504.33	43425.40	47311.83	47268.97
[41]	48051.91	47368.09	46074.38	43710.87	42913.33	37672.22	39707.07	42739.80
[49]	52306.06	52373.52	46142.15	42784.18	42615.27	49866.51	50429.96	49526.46
[57]	47986.78	42416.59	38095.85	41164.54	45150.21	45151.38	53163.66	54408.51
[65]	42994.00	54310.17	57858.24	56452.10	54396.09	49647.39	44446.88	46721.22
[73]	49849.64	56453.70	57135.07	56352.24				

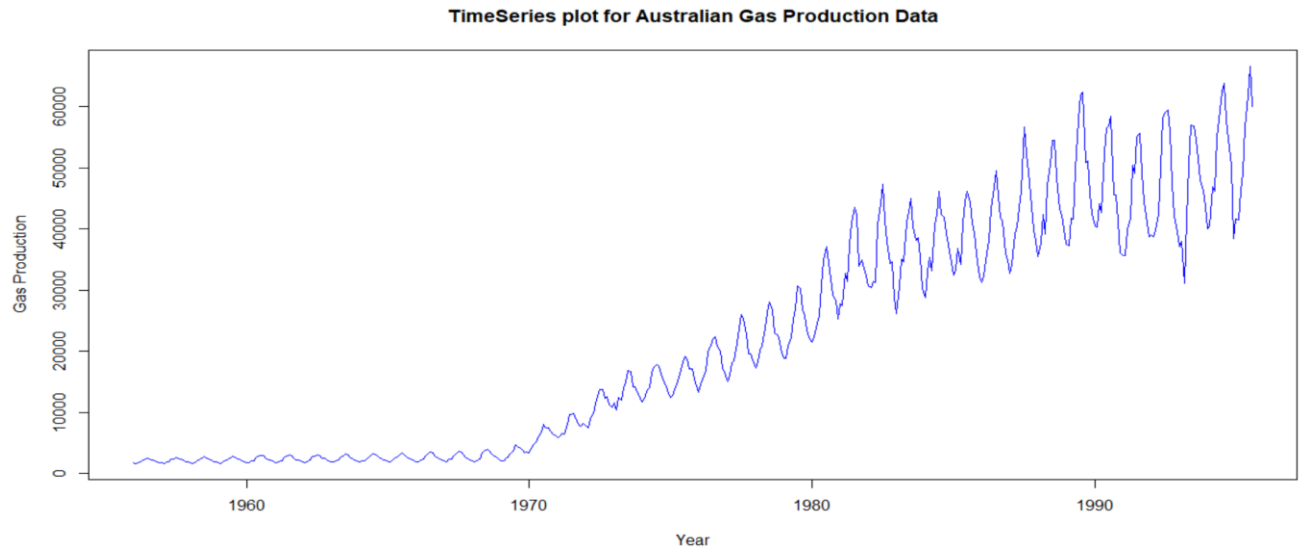
## 6. Plotting and Observations:

The components of time series are as below:

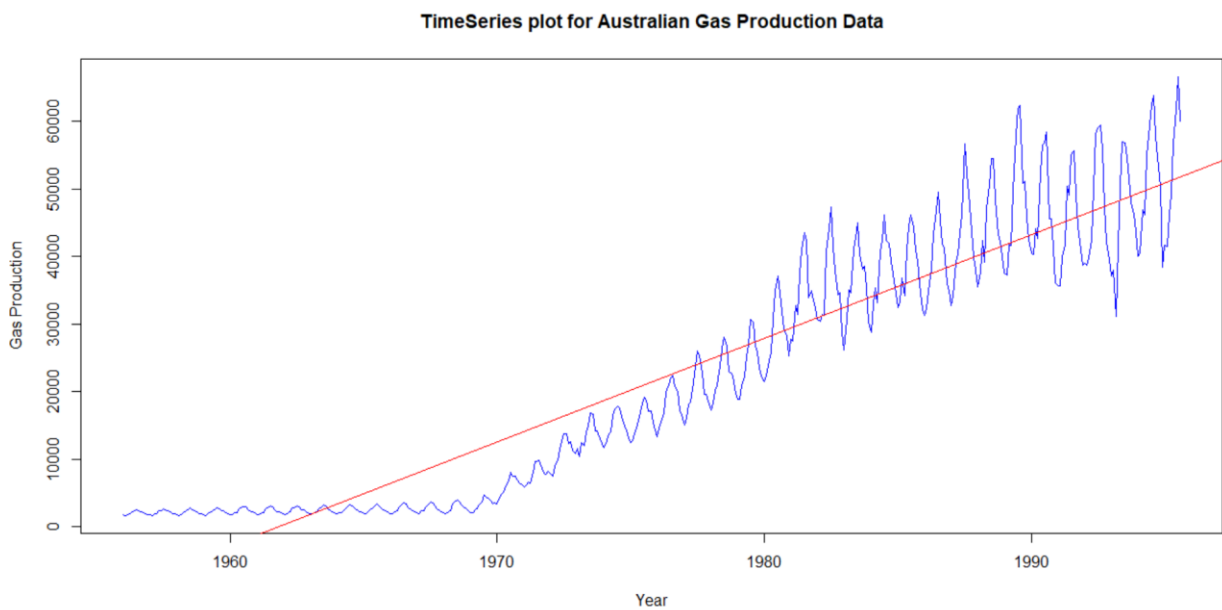
1. **Trend**: The increase of the series for a long term in a direction of increase or decrease and makes the pattern predictable
2. **Seasonality**: The systematic/ calendar related movements that repeat over a specific period such as day/week/month/season etc. This is inter-year changes.
3. **Cyclic**: These are intra-year variations. This oscillatory movement has a period of oscillation of more than a year. One complete period is a cycle.
4. **Irregular**: These fluctuations are unforeseen, uncontrollable, unpredictable, and are erratic.

### 6.1 Time Series plot:

- From the below plot of timeseries we observe that data is recorded from somewhere in January 1956 till August 1995.
- As the years pass by, the production of Gas has drastically increased.
- In the initial years until 1970, there was no trend observed in the data. Also, the seasonality is not quite visible
- After 1970, **there is an increase in trend and seasonality** started increasing in magnitude as well.
- From, 1980s the data is showing great fluctuations each year till the end
- The cyclic component of the data is not visible.



**Regression line across the series:**

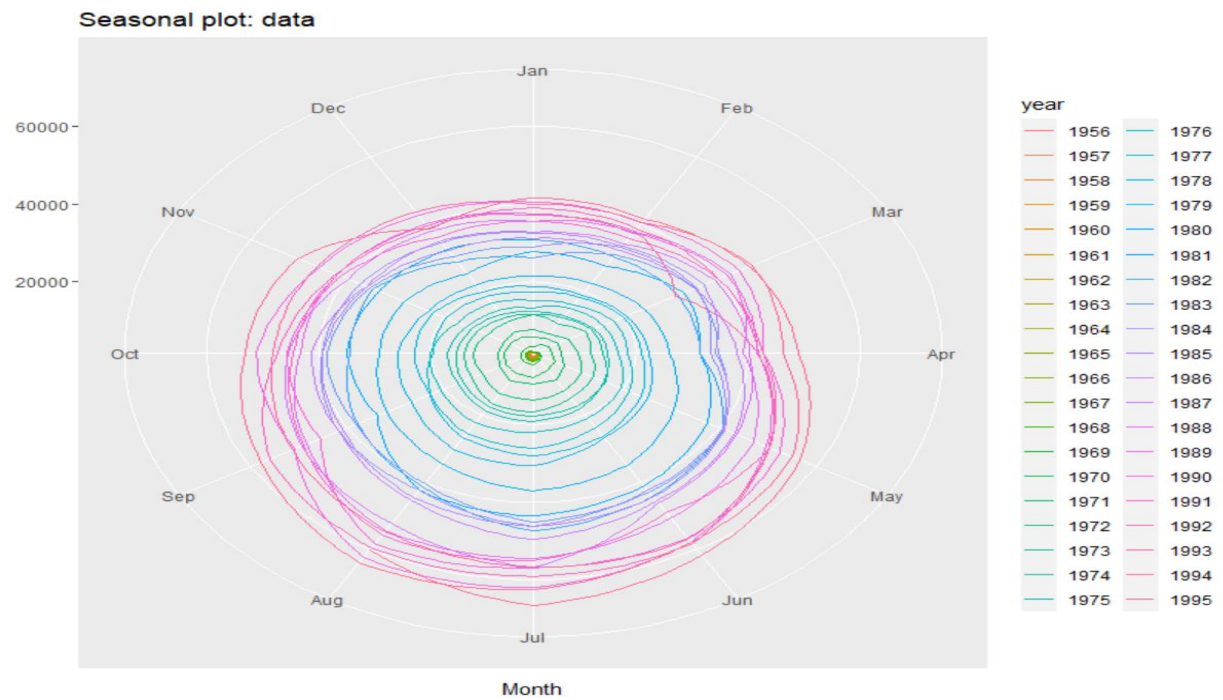
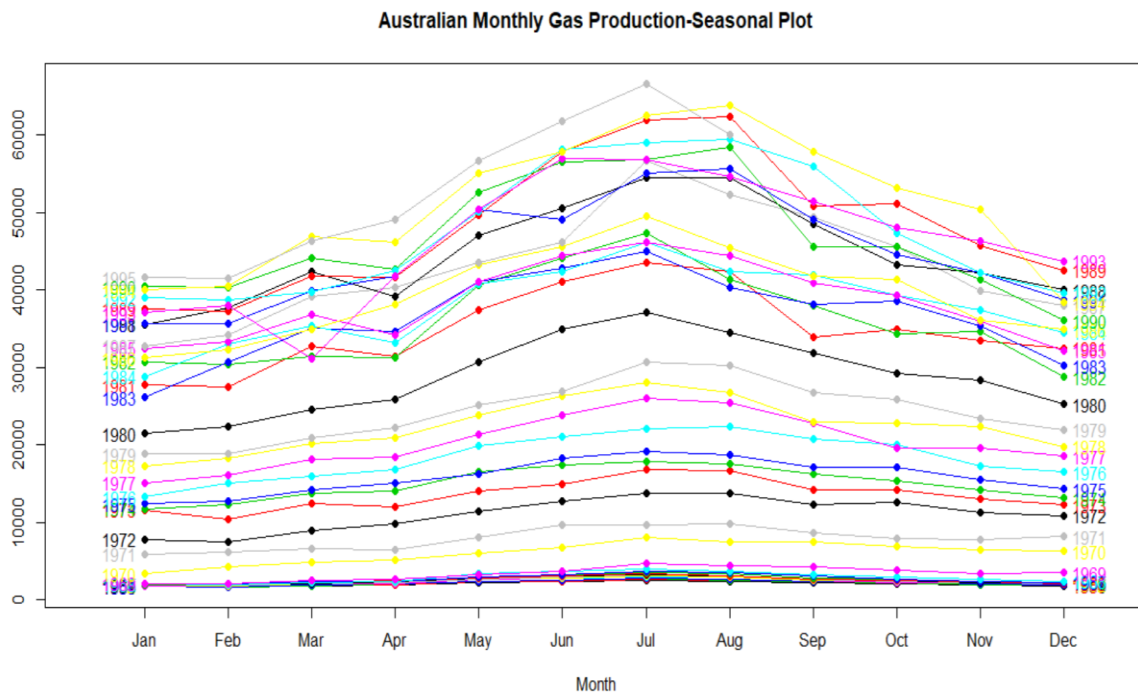


## 6.2 Seasonal plot:

- The seasonal plot shows the trend of gas production of each year across all the months.
- The gas production have increased over time with each year which may be indicative of an increasing linear trend, perhaps due to increasing demand for gas in that time period
- As the year increases, the gas production is increasing as well.
- July seems to be the month of highest gas produced month in the entire year for all the years of observation



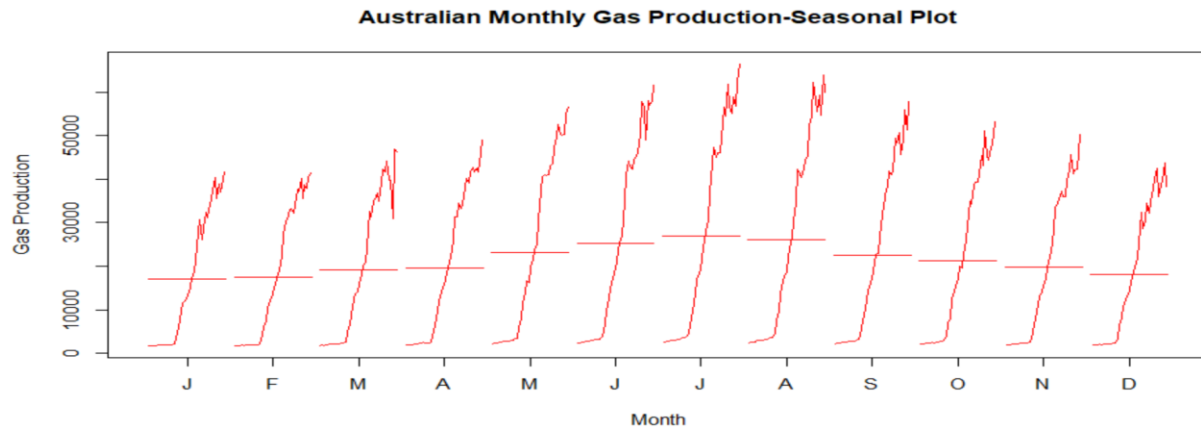
- In the initial years, the gas production is the same across all the months
- The least gas production happened in 1956 and highest in 1995



## 6.2 Subseries plot:

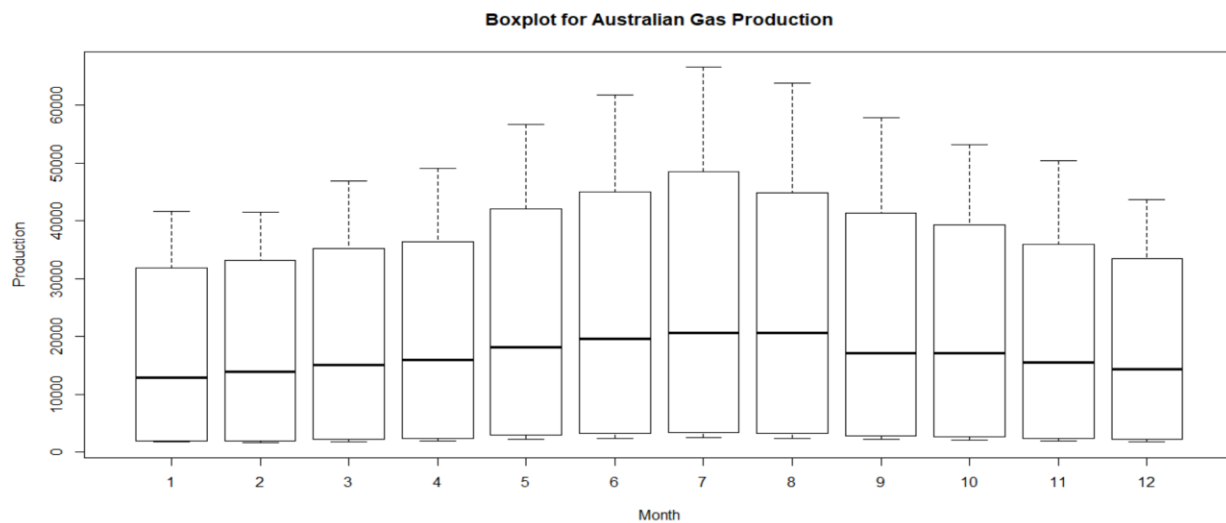
The Monthly plot shows that gas production is more in July followed by August and the June.

The production rate systematically increases from Jan till the mid of the year and then slowly starts to decline.



## 6.3 Boxplot:

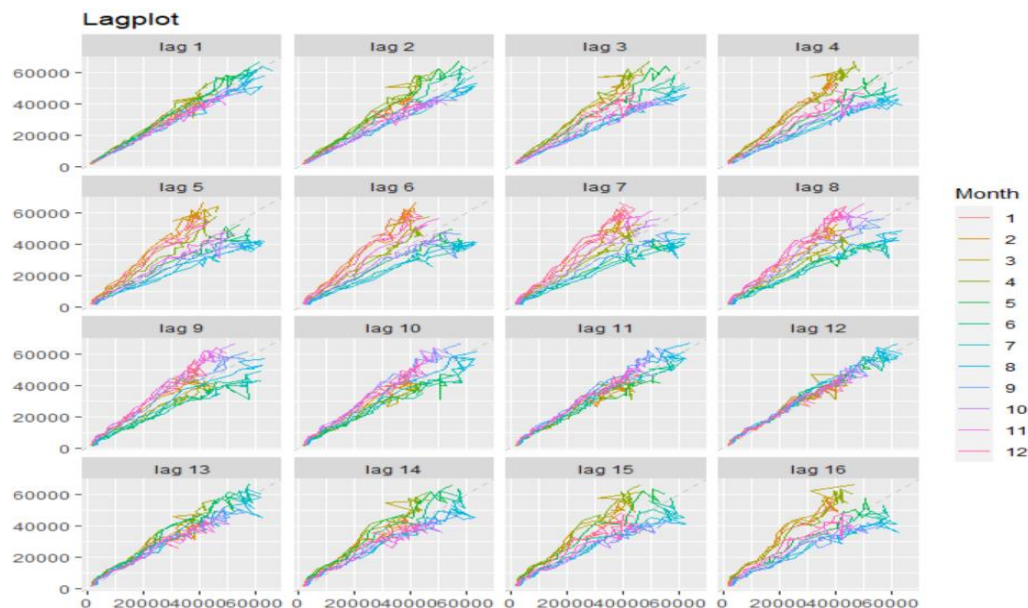
We observe there are not outliers when Month wise data is considered for the years. The rest of the description on the production of gases across the months remains same as above.



## 6.4 Lagplot:

A lag plot is used to help evaluate whether the values in a dataset or time series are random. If the data are random, the lag plot will exhibit no identifiable pattern. If the data are not random, the lag plot will demonstrate a clearly identifiable pattern.

From the below graph we see there is a strong correlation at lag 12. Hence the data is not random and is exhibiting some pattern.



**To conclude that the dataset “gas” has trend and seasonality component present in its time series.**

## 7. Periodicity:

Periodicity is a pattern in a time series that occurs at regular time intervals. It is the sampling frequency.

From lagplot we noticed that periodicity is 12. Because at lag =12, there exists a huge correlation of the data

Gas dataset is having a periodicity of 12 months.

```
> periodicity(data)
Monthly periodicity from Jan 1956 to Aug 1995
> periodicity(DataAnalysis)
Monthly periodicity from Jan 1970 to Aug 1995
```

```
> #frequency of time series
> frequency(data)
[1] 12
> #Checking the cycle/periodicity of the data
> cycle(data)
      Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
1956   1   2   3   4   5   6   7   8   9  10  11  12
1957   1   2   3   4   5   6   7   8   9  10  11  12
1958   1   2   3   4   5   6   7   8   9  10  11  12
1959   1   2   3   4   5   6   7   8   9  10  11  12
```

1960	1	2	3	4	5	6	7	8	9	10	11	12
1961	1	2	3	4	5	6	7	8	9	10	11	12
1962	1	2	3	4	5	6	7	8	9	10	11	12
1963	1	2	3	4	5	6	7	8	9	10	11	12
1964	1	2	3	4	5	6	7	8	9	10	11	12
1965	1	2	3	4	5	6	7	8	9	10	11	12
1966	1	2	3	4	5	6	7	8	9	10	11	12
1967	1	2	3	4	5	6	7	8	9	10	11	12
1968	1	2	3	4	5	6	7	8	9	10	11	12
1969	1	2	3	4	5	6	7	8	9	10	11	12
1970	1	2	3	4	5	6	7	8	9	10	11	12
1971	1	2	3	4	5	6	7	8	9	10	11	12
1972	1	2	3	4	5	6	7	8	9	10	11	12
1973	1	2	3	4	5	6	7	8	9	10	11	12
1974	1	2	3	4	5	6	7	8	9	10	11	12
1975	1	2	3	4	5	6	7	8	9	10	11	12
1976	1	2	3	4	5	6	7	8	9	10	11	12
1977	1	2	3	4	5	6	7	8	9	10	11	12
1978	1	2	3	4	5	6	7	8	9	10	11	12
1979	1	2	3	4	5	6	7	8	9	10	11	12
1980	1	2	3	4	5	6	7	8	9	10	11	12
1981	1	2	3	4	5	6	7	8	9	10	11	12
1982	1	2	3	4	5	6	7	8	9	10	11	12
1983	1	2	3	4	5	6	7	8	9	10	11	12
1984	1	2	3	4	5	6	7	8	9	10	11	12
1985	1	2	3	4	5	6	7	8	9	10	11	12
1986	1	2	3	4	5	6	7	8	9	10	11	12
1987	1	2	3	4	5	6	7	8	9	10	11	12
1988	1	2	3	4	5	6	7	8	9	10	11	12
1989	1	2	3	4	5	6	7	8	9	10	11	12
1990	1	2	3	4	5	6	7	8	9	10	11	12
1991	1	2	3	4	5	6	7	8	9	10	11	12
1992	1	2	3	4	5	6	7	8	9	10	11	12
1993	1	2	3	4	5	6	7	8	9	10	11	12
1994	1	2	3	4	5	6	7	8	9	10	11	12
1995	1	2	3	4	5	6	7	8				

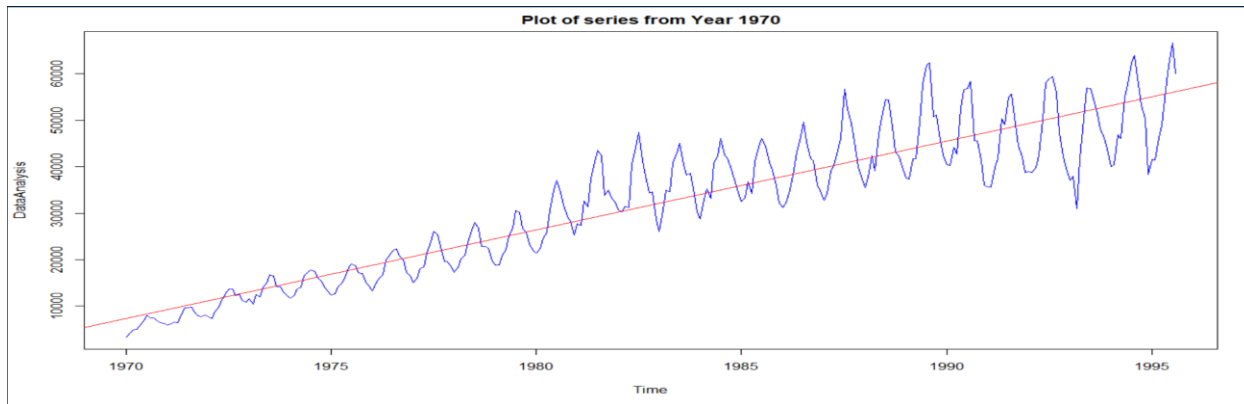
## 8. Subsetting the data:

From the timeseries graph, we see that from 1956-1970, there is no information that we can get as nothing is happening. Hence considering the dataset from 1970 till the end which is 1995 August.

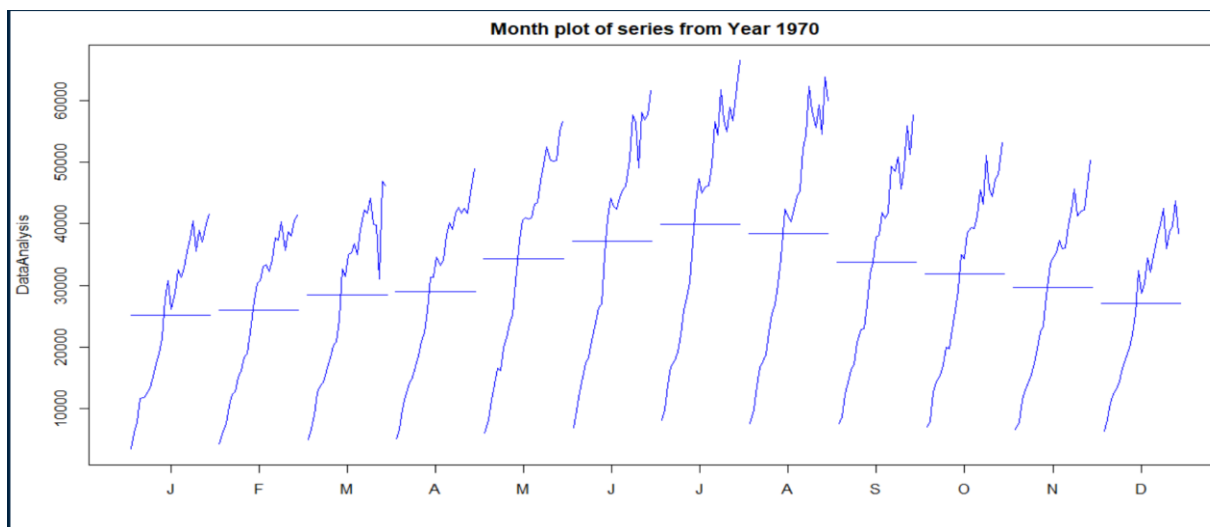
```
> DataAnalysis=window(data,start=1970)
> start(DataAnalysis)
[1] 1970      1
> end(DataAnalysis)
[1] 1995      8
> frequency(DataAnalysis)
[1] 12
> end(dataTrain)
[1] 1993     12
[1] 1995      8
```

## 8.1 Various charts with the subset of dataset:

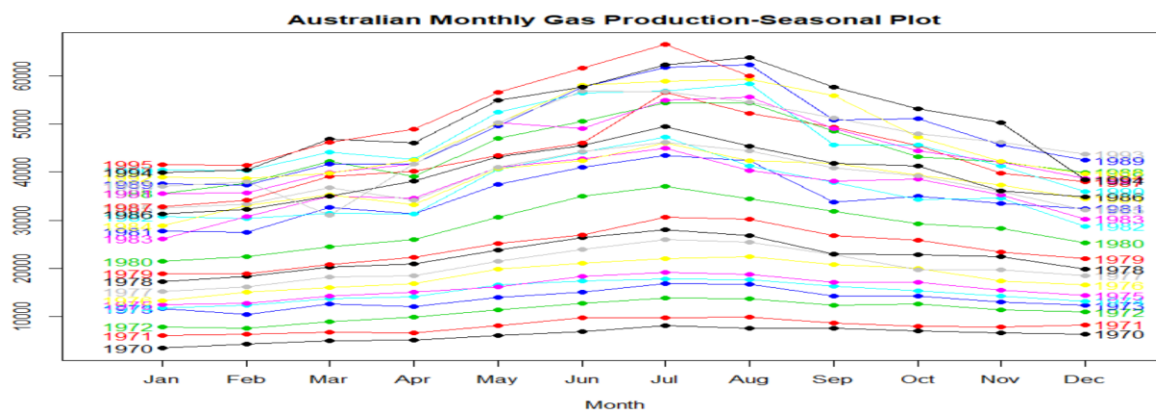
### Series plot:



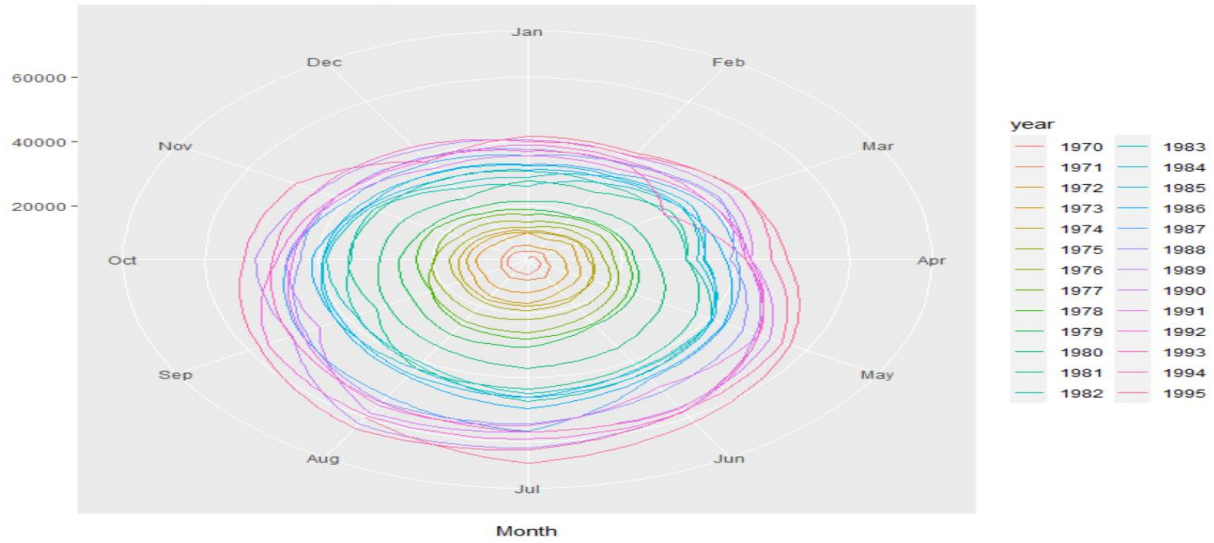
### Month plot:



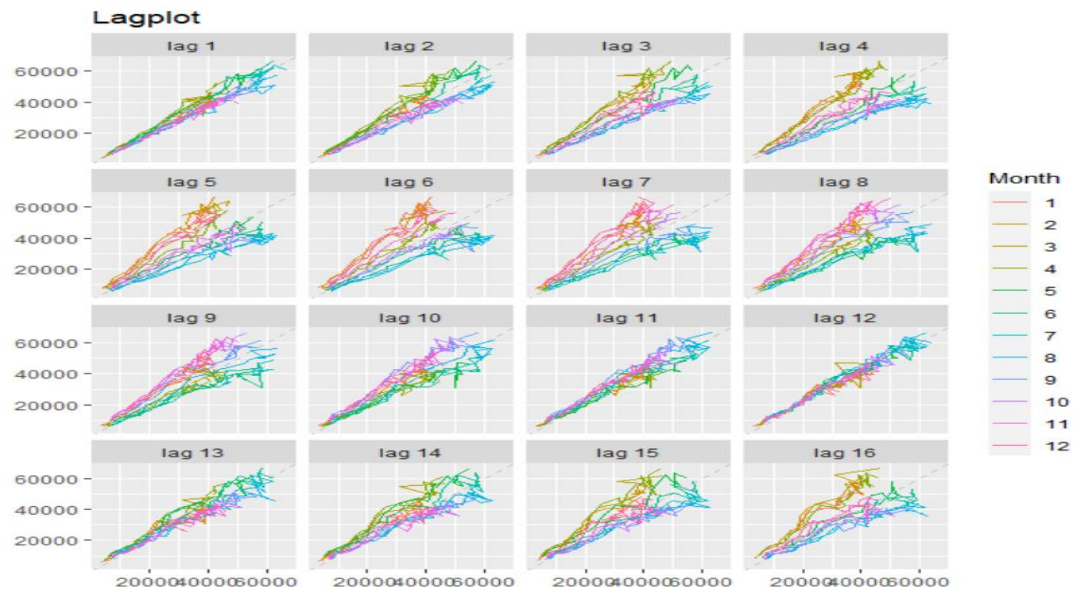
### Seasonal plot:



Seasonal plot: DataAnalysis



**Lagplot:**

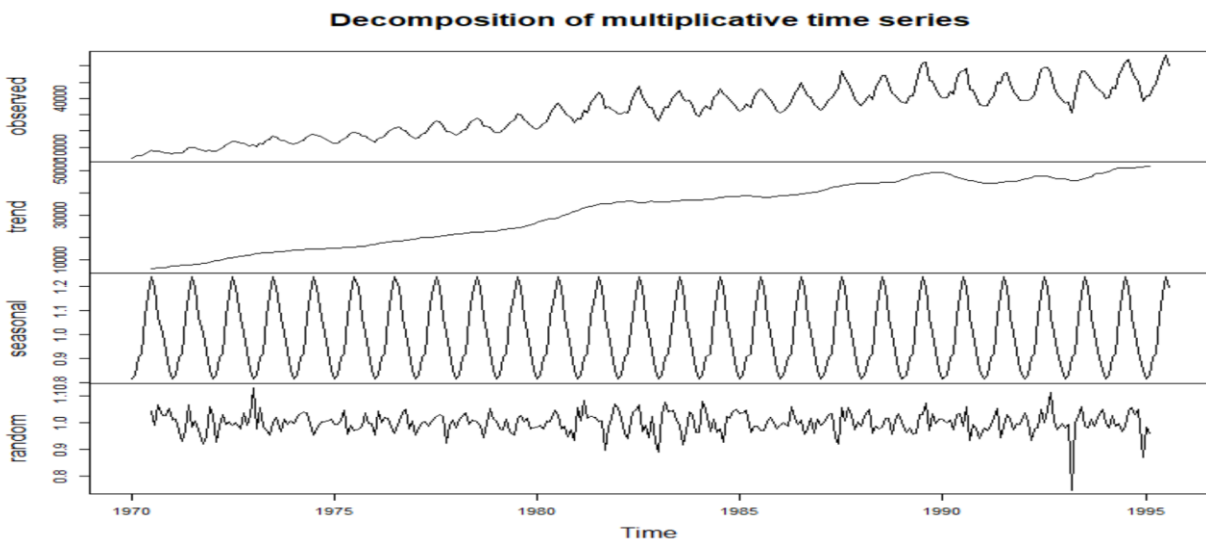
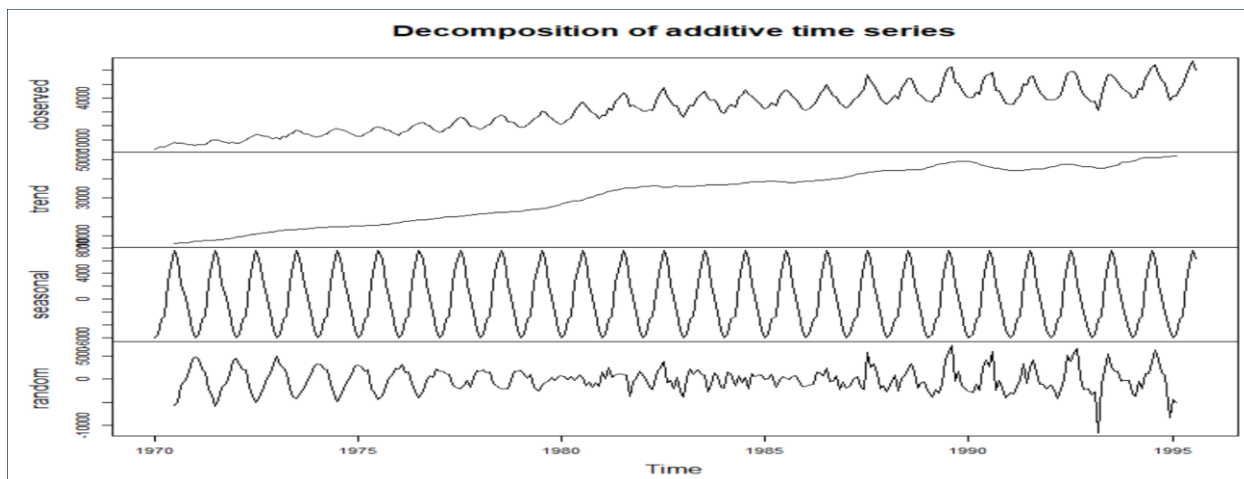


## 8.2 Decomposing the dataset:

We must check if the model is additive or multiplicative by decomposing the series and plotting the same. We can use decompose function to perform the same.

```
{r}  
#Decomposing the data into trend,seasonality and randomness  
decomAdd=decompose(data,type = "additive")  
plot(decomAdd)  
decomMul=decompose(data,type="multiplicative")  
plot(decomMul)  
}
```

Below are the charts for additive and multiplicative timeseries plotted on the same dataset. We notice that there is no change in the seasonality in both. Hence, we can say **that our timeseries is additive**.





## 9. Is the time series Stationary?

Fitting an ARIMA model requires the series to be stationary. So, before developing an ARIMA model we need to be sure if the data is stationary or not. A series is said to be stationary when its mean, variance, autocovariance are not dependent on time. If it is not a stationary series, the model will be improbable to give a proper forecast. If it is not a stationary series, it can be converted to one by taking the difference of the series

Below are the methods with which we can check if the series is stationary:

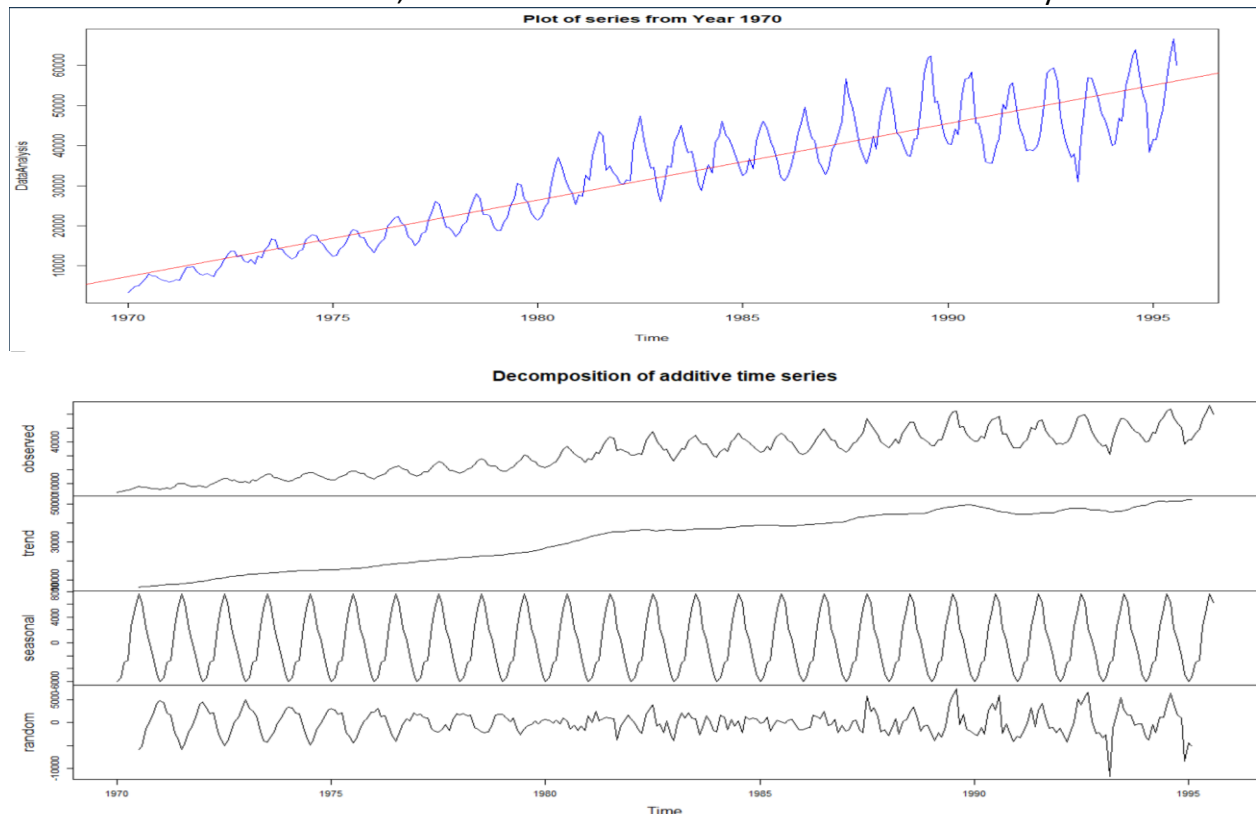
1. Plots
2. Dickey Fuller Test

### 1. Observing through Plots:

From the below charts where we have plotted the series separately in terms of trend, seasonality, observed and randomness, we see below:

- There is an increasing trend as the time pass by
- There are timely peaks and troughs through the series, there by we can say there is seasonality
- We see the variance is not constant as well. It is changing across the months.

When there is trend and seasonality, the mean and variance cannot be constant. If mean and variance are not constant, then we can conclude that series is not stationary.





### 9.1 Dickey Fuller Test:

If the Augmented Dickey-Fuller test is having a p-value less than 0.05, it means that the null hypothesis is not true and if the p-value is more than 0.05, the data is stationary and no need of any modification

**Null Hypothesis:** for ADF is Time series is Non-Stationary

**Alternate Hypothesis:** Time series is stationary

Rejection of null hypothesis implies that series is stationary

#### **Dickey Fuller test on the entire series from 1956 to 1995:**

```
> adf.test(data)
p-value smaller than printed p-value

Augmented Dickey-Fuller Test

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

Here from our hypothesis test, timeseries is not stationary when we do ADF on the entire dataset.

#### **Dickey Fuller test on the entire series from 1970 to 1995:**

```
> adf.test(DataAnalysis)
p-value smaller than printed p-value
Augmented Dickey-Fuller Test

data: DataAnalysis
Dickey-Fuller = -4.7629, Lag order = 6, p-value = 0.01
alternative hypothesis: stationary
```

Here our p value is  $<0.05$ , which means we reject the null hypothesis and accept the alternate hypothesis which says that time series is stationary.

From the above plots, we see there is a trend and seasonality in the data. We have to DE seasonalize the data to make is stationary.

### 9.2 Removing seasonality:

**Stl:** Decompose a time series into seasonal, trend and irregular components using loess method. It takes 2 mandatory arguments as below:

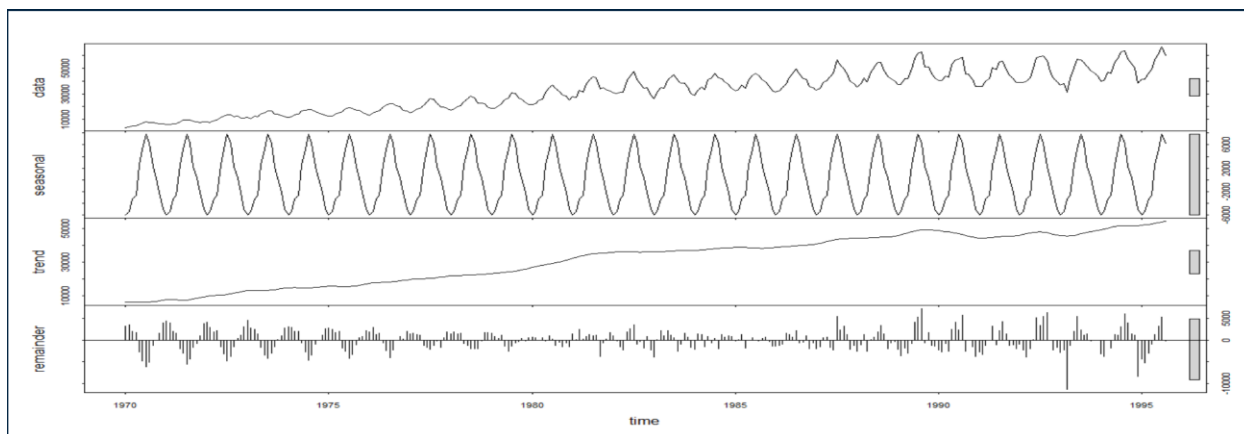
- The univariate data as first argument and it should be of time series object.
- S.window is the span (in lags) of the loess window for seasonal extraction.

**Seasadj:** extracts seasonally adjusted data by removing the seasonal components from the result of STR decomposition.

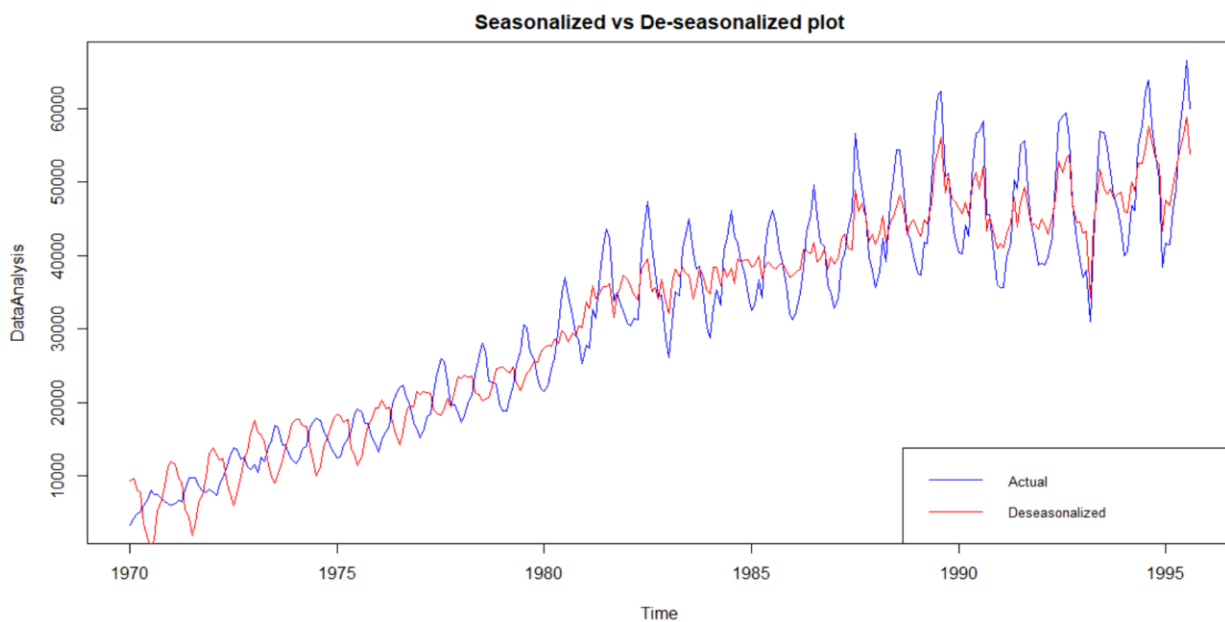
### Removing seasonality

```
```{r}  
decomp = stl(DataAnalysis, s.window = "periodic")  
deseasonal_data=seasadj(decomp)  
plot(decomp)  
plot(deseasonal_data)  
```
```

### Before seasonality is removed:



### Seasonalized vs De-seasonalized plots:



From the above chart, we see the **seasonality component is removed and there are no periodic peaks and troughs**. But we still see trend in the series. We must remove the trend as well to make the series stationary.

Now that seasonality is removed, we still see some trend in the data. We must make the means equal by taking the difference in the data. To make data to stationary, we must remove that as well. We can remove the trend by taking the differences as shown below:

### **9.3 Removing Trend Component:**

Trend component can be removed by taking the difference of consecutive observations as shown below

Diff function takes 2 arguments as below:

**A numeric vector as x.** Here we passed the de-seasonalized data as input

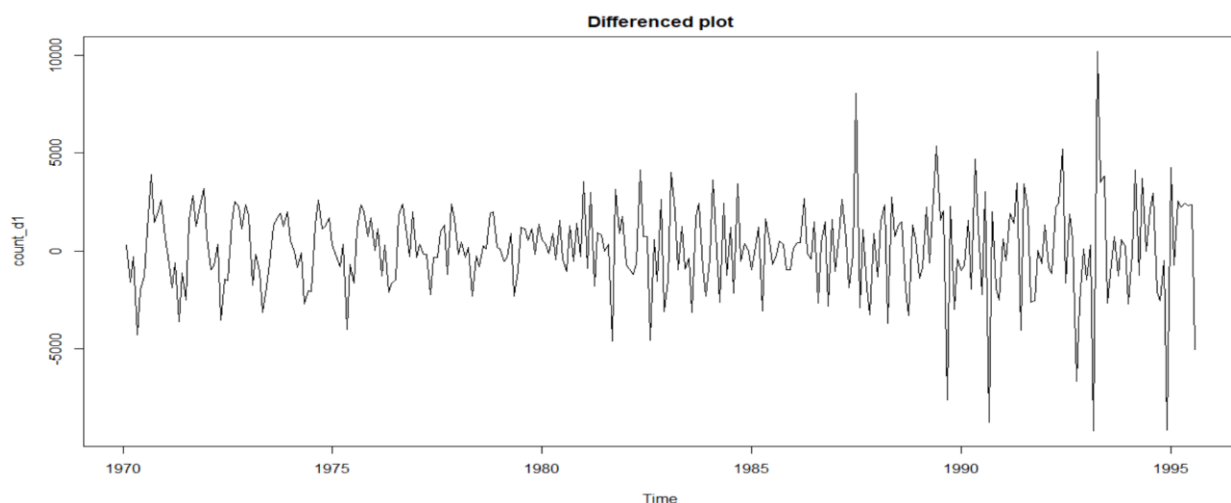
**Differences:** an integer indicating the order of the difference

After removing the trend below is the plot we get:

```
```{r}
count_d1 = diff(deseasonal_data, differences = 1)
plot(count_d1)
adf.test(count_d1, alternative = "stationary")
```
```

### **Plot after removing trend and seasonality:**

From the below plot, we see the trend and seasonality is removed. After 1960, it appears there is a huge variance in the data. Finally, our data is stationary. We can verify the same by performing Augmented Dickey Fuller Test.



### Dicky-Fuller Test again to verify stationarity:

```
> adf.test(DataAnalysis)
p-value smaller than printed p-value
Augmented Dickey-Fuller Test

data: count_d1
Dickey-Fuller = -13.507, Lag order = 6, p-value = 0.01
alternative hypothesis: stationary
```

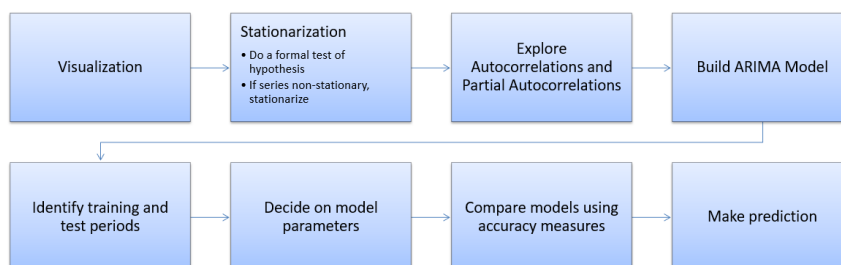
Here p value is  $<0.05$ , the series is stationary

## 10. Building ARIMA Model Manually:

ARIMA stands for Auto Regressive Integrated Moving Average. This model includes an explicit statistical model for the irregular component of a time series, that allows for non-zero autocorrelations in the irregular component

### Steps in building ARIMA Model:

1. **Visualization:** visualize the data with plots for checking trend, seasonality
2. **Stationarize:** Do a format hypothesis test to check if the series is stationary or not. If it is non stationary, then stationarize it.
3. **Explore the p and q** values with Partial Auto correlation and Auto correlation plots respectively and d value from the differencing
4. **Build ARIMA Model:**
  - a. Identify the train and test datasets
  - b. Decide on model parameters from the ACF and PACF plots
  - c. Compare models using accuracy measures
  - d. Make forecast
  - e. Make predictions
5. **Calculate MAPE/RMSE**
6. Auto ARIMA



Source: Great Learning PPT

In the previous section, we have checked for trend and seasonality in the data and removed those components. Also, we made the series to stationary. ARIMA model is built only on stationary series. Now let us explore p and q values from PACF and ACF plots respectively

To build an ARIMA model, it takes 3 parameters namely p, d, q as follows:

- p is the number of autoregressive terms, (PACF plot)
- d is the number of nonseasonal differences needed for stationarity, and
- q is the number of lagged forecast errors in the prediction equation. (ACF plot)

### **Exploring ACF and PACF with Chart:**

ARIMA model can be built only if there is an auto correlation among the data. For time-series it is one of the assumptions.

**ACF:** This is an (complete) auto-correlation function which gives us values of autocorrelation of any series with its lagged values. it describes how well the present value of the series is related with its past values. A time series can have components like trend, seasonality, cyclic and residual. ACF considers all these components while finding correlations

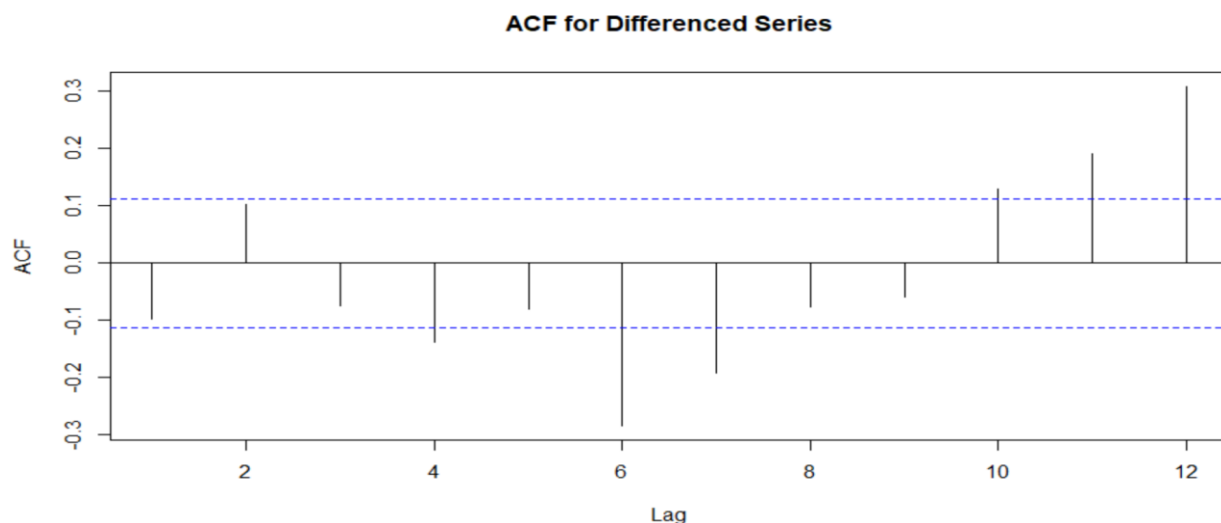
**ACF Graph starts from 0.** The first bar in the graph says about the correlation with it self

The auto correlation of a series with itself will be one. From the below chart, we see that the ACF is slowly decreasing as the lag increases. Here lag represents the trend. After a while, ACF starts increasing and then it decreases again. This happens when there is a periodicity in data.

The autocorrelation will be more for periodic lags when compared to other lags. For Example,

The data in January of an year is similar to data of January in the subsequent year in our case study. Due to this, data shows higher correlation during this period. **Hence periodicity is 12**

So from the below plot, we can conclude that **q=6**

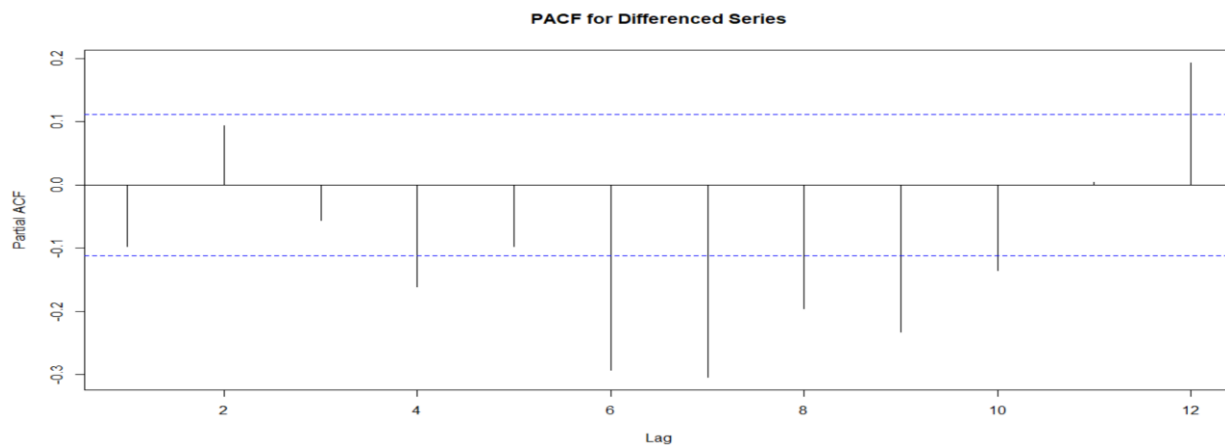


**PACF:** This is a partial auto-correlation function. it finds correlation of the residuals (which remains after removing the effects which are already explained by the earlier lag(s)) with the next lag value hence 'partial' and not 'complete' as we remove already found variations before we find the next correlation

**PACF graph starts from 1.** This is the correlation of the series with its lagged version of order 1.

$ACF(1)=PACF(1)$

So from the above plot, we can conclude that **p=7**



We took the difference of 1<sup>st</sup> order above, hence  $d=1$ . Finally the values of  $p, d, q$  for ARIMA are ARIMA (7,1,6). These are obtained from the plots.

**Splitting the data into train and test:**

```
{r}
dataTrain = window(deseasonal_data, start=1970, end=c(1993, 12), frequency=12)
start(dataTrain)
end(dataTrain)
dataTest= window(deseasonal_data, start=1994)
start(dataTest)
end(dataTest)
plot(deseasonal_data)
```

The R Console output shows the following results for the window function:

|     |      |    |
|-----|------|----|
| [1] | 1970 | 1  |
| [1] | 1993 | 12 |
| [1] | 1994 | 1  |
| [1] | 1995 | 8  |

## Building Manual ARIMA Model:

### Summary of the model:

The AIC value obtained with auto arima is 5123.82

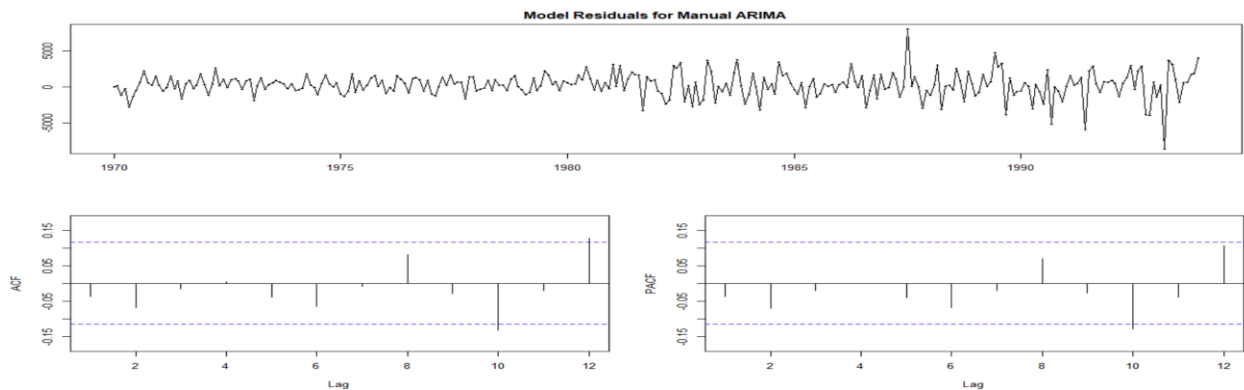
```
Call:
arima(x = dataTrain, order = c(7, 1, 6))

Coefficients:
      ar1      ar2      ar3      ar4      ar5      ar6      ar7
0.7834  0.0871 -0.6546 -0.3933  0.8895 -0.6652 -0.2067
s.e.  0.0884  0.0910  0.0696  0.0772  0.0558  0.0701  0.0754
      ma1      ma2      ma3      ma4      ma5      ma6
-1.2003  0.2292  0.5926  0.2627 -1.0533  0.7871
s.e.  0.0673  0.0997  0.1038  0.1094  0.0944  0.0606

sigma^2 estimated as 2934529:  log likelihood = -2547.91,  aic = 5123.82
```

Manual arima model is built with c(7,1,6) p,d,q values respectively. We have received aic value as 5123.82 which is quite high. Lets check the residual plots and perform hypothesis test on the same.

Below is the residual plot for Manually built ARIMA model. From the plot, we see the error spread is random and independent. There is no auto correlation in the error.



However, lets perform **Ljung box** test to verify the same.

The Ljung–Box test is a type of statistical test of whether any of a group of autocorrelations of a time series are different from zero. Instead of testing randomness at each distinct lag, it tests the "overall" randomness based on a number of lags, and is therefore a portmanteau test.

Below are the Null and alternate hypothesis for the test.

**H<sub>0</sub>**: Residuals are independent

**H<sub>a</sub>**: Residuals are not independent

### Box-Pierce test

```
data: arimaModel$residuals  
X-squared = 0.37314, df = 1, p-value = 0.5413
```

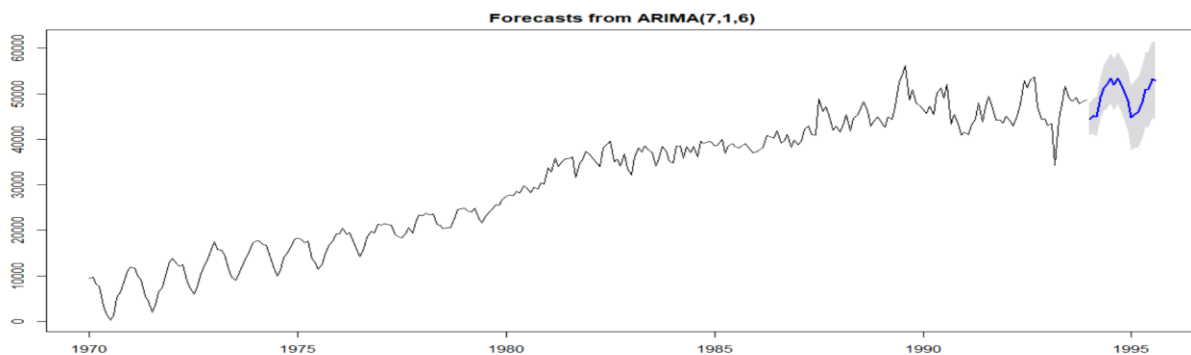
The p-value obtained is  $>0.05$ , hence we must **accept the null hypothesis that residuals are independent.**

Now let us make a forecast with our model.

Below is the forecast plot made with manual ARIMA model. At the end of the plot we have a band with 95% confidence showing approximate forecasting values.

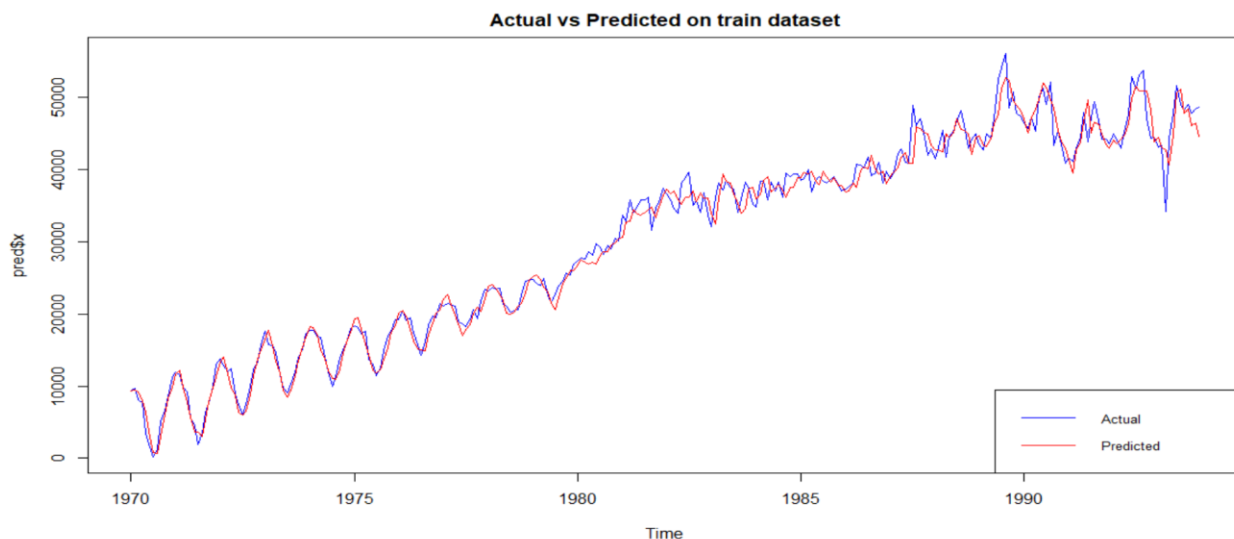
### Forecast with Manual ARIMA:

The blue line gives us the mean value of the forecast surrounded by a confidence band



### Actual vs Predicted on train dataset:

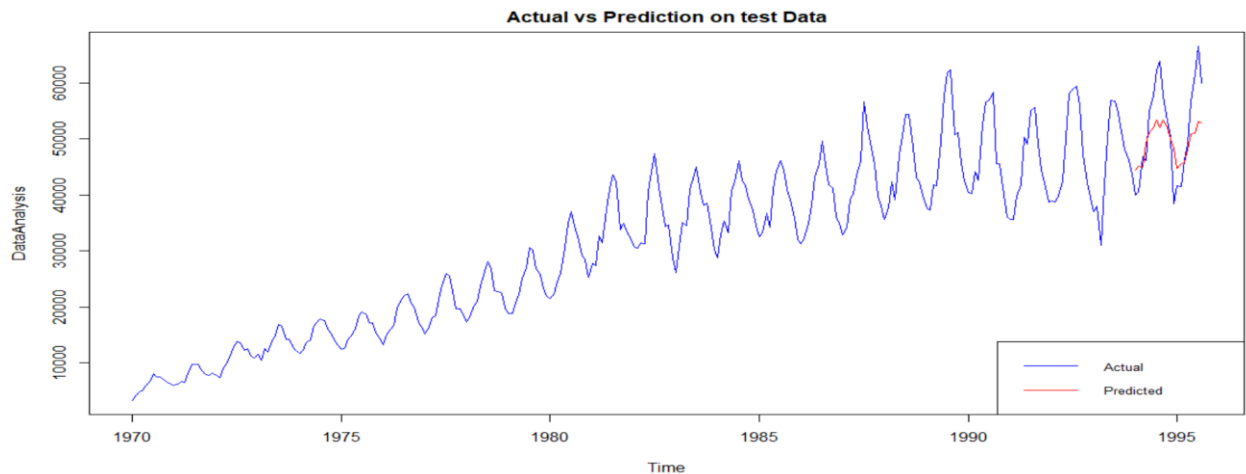
The blue line is the actual plot of deseasonalized data and red line is the prediction. After the model is built, predictions are made on the train dataset. Predictions are showed by red line





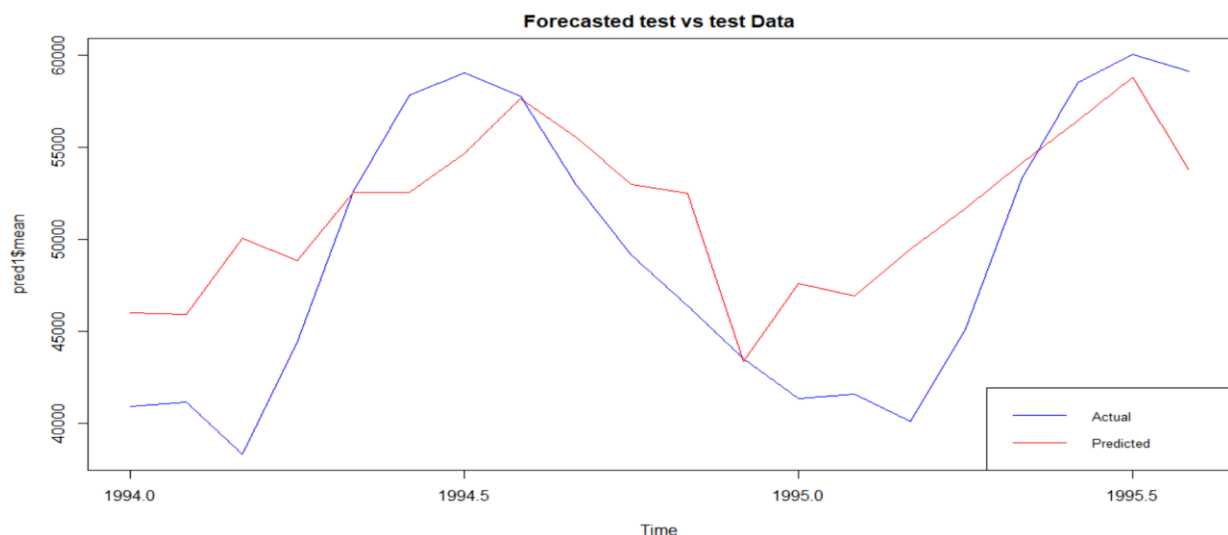
### Actual vs Predicted on test dataset:

The predictions (red) made on the test dataset are compared with the original values and plotted in the graph. The red line is the predicted graph on the test dataset. This is compared against the original data (blue) which have trend and series. The reason why original data is used for comparison is because in real time, data will have both seasonality and trend. The predictions made is not that good from the graph although seasonality is captured to some extent. The variance is missed to some extent.



### Predicted test vs test Data built with manual ARIMA:

The below plot compares only the test data window. Red line is the actual test data and blue line is the prediction made.



### Accuracy when forecast is made for 20 periods using manual ARIMA model:

```
>f7=forecast(arimaModel)
```

```
>forecast::accuracy(fcast, dataTest,h=20)
      ME      RMSE      MAE      MPE      MAPE
Training set 273.5049 1710.070 1225.375 -0.1126082 6.313526
Test set     2152.9943 3244.316 2695.219  3.9521467 5.192869
      MASE      ACF1 Theil's U
Training set 0.4648297 -0.03599488 NA
Test set     1.0223951 -0.03029999 0.9832872
```

We consider the MAPE and RMSE values for checking the performance of the model. Here we got descent MAPE values but the AIC value is high (from earlier)

## 11. Building model with auto.arima:

Auto ARIMA deals with seasonality automatically. Hence, we can pass the seasonalized dataset into the model

```
``{r pressure, echo=FALSE}
dataTrain1 = window(DataAnalysis,start=1970,end=c(1993,12),frequency=12)
start(dataTrain1)
end(dataTrain1)
dataTest1= window(DataAnalysis, start=1994,frequency=12)

fit<-auto.arima(dataTrain1 , seasonal=TRUE)
fit
tsdisplay(residuals(fit), lag.max=12, main='Auto ARIMA Model Residuals')
``
```

### Summary of the model:

The AIC value obtained with auto arima is 4939.33

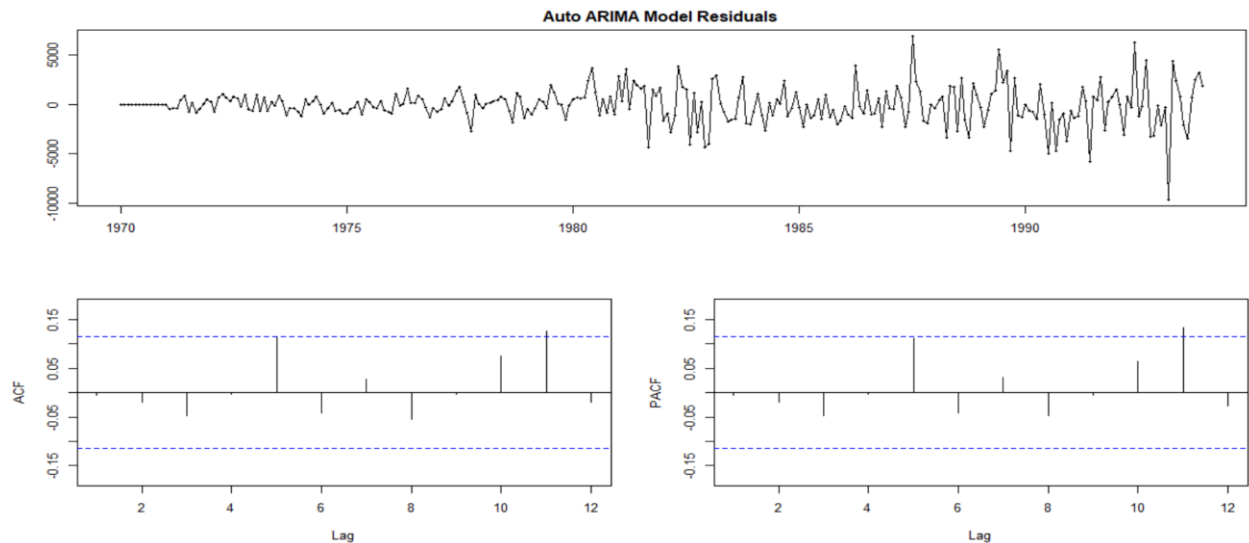
```
Series: dataTrain
ARIMA(2,1,1)(0,1,2)[12]

Coefficients:
      ar1      ar2      ma1      sma1      sma2
      0.5017  0.2057 -0.9583 -0.4404 -0.1236
s.e.    0.0738  0.0722  0.0426  0.0676  0.0639

sigma^2 estimated as 3535011:  log likelihood=-2463.67
AIC=4939.33  AICc=4939.64  BIC=4961.03
```

### Auto ARIMA residual plots:

We see there is no pattern in the residuals from the below plot.



### Residual test with Ljung Boxtest:

Below are the Null and alternate hypothesis for the test.

**H<sub>0</sub>**: Residuals are independent

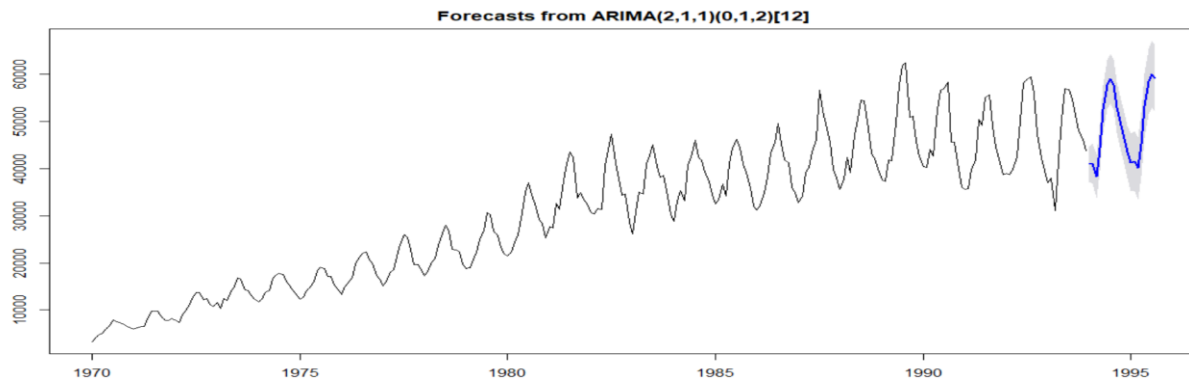
**H<sub>a</sub>**: Residuals are not independent

```
Box-Pierce test
data:  fit2$residuals
X-squared = 0.0098249, df = 1, p-value = 0.921
```

The p-value obtained is  $>0.05$ , hence we must **accept the null hypothesis that residuals are independent.**

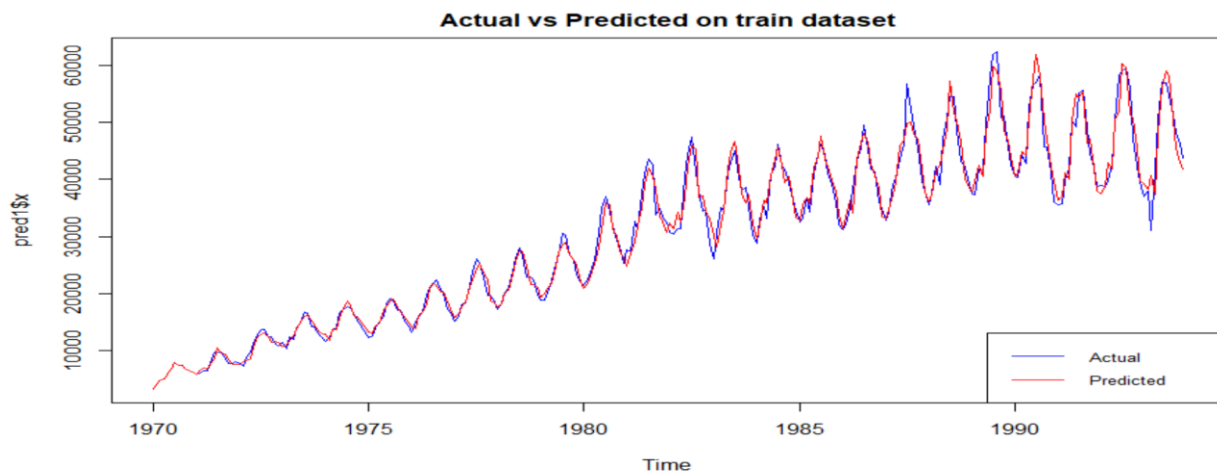
### Forecasting with auto.arima for 20 periods:

From the below graph we see the blue line is the forecast made for 20 more periods. The p,d,q values selected are shown in the below graph. Order ARIMA (2,1,1)(0,1,2)[ 12]



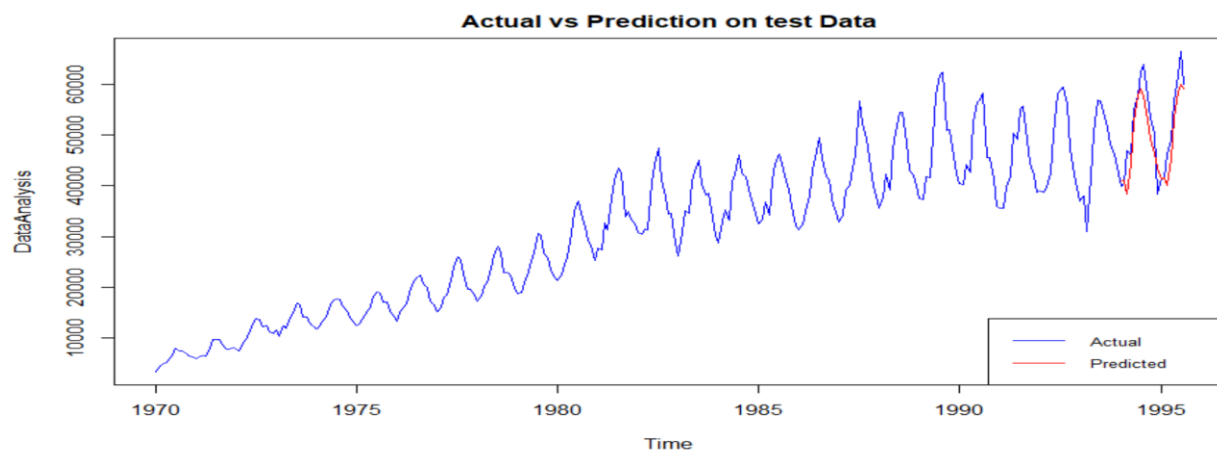
### Actual vs Predicted graph on Train dataset auto.arima:

The seasonality component is automatically handled by auto.arima. The blue line shows the real data and red line shows the predicted. This can be overfitting as well

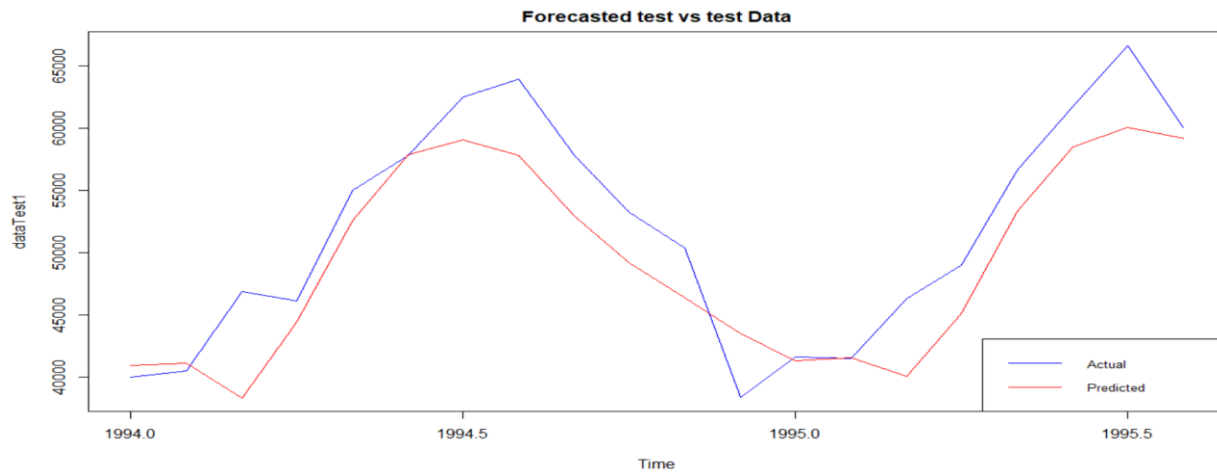


### Actual vs Predicted graph on Test dataset with auto.arima:

The seasonality is captured better here when compared to the model that is built with manual arima. The blue line shows the real data and red line shows the predicted.



### Predicted test vs actual test Data plot built with auto.arima:



From the plots we see that auto.arima is fitting better than what we saw in the plot for manual ARIMA

## 12.Accuracy of the models:

Accuracy is an important measure test the performance of the model we built. All the models can be used for making forecasting and predictions but choosing the best model is important for making accurate predictions and forecasting.

Let's compare the accuracy of the models we built with manual arima and auto.arima

### Manual ARIMA model Accuracy and Summary:

#### Accuracy:

|              | ME        | RMSE        | MAE       | MPE        | MAPE     |
|--------------|-----------|-------------|-----------|------------|----------|
| Training set | 273.5049  | 1710.070    | 1225.375  | -0.1126082 | 6.313526 |
| Test set     | 2152.9943 | 3244.316    | 2695.219  | 3.9521467  | 5.192869 |
|              | MASE      | ACF1        | Theil's U |            |          |
| Training set | 0.4648297 | -0.03599488 | NA        |            |          |
| Test set     | 1.0223951 | -0.03029999 | 0.9832872 |            |          |

#### Summary:

```
Forecast method: ARIMA(7,1,6)

Model Information:

Call:
arima(x = dataTrain, order = c(7, 1, 6))

Coefficients:
      ar1      ar2      ar3      ar4      ar5      ar6
 0.7834  0.0871 -0.6546 -0.3933  0.8895 -0.6652
s.e.    0.0884  0.0910  0.0696  0.0772  0.0558  0.0701
      ar7      ma1      ma2      ma3      ma4      ma5
```

```

      -0.2067  -1.2003  0.2292  0.5926  0.2627  -1.0533
s.e.   0.0754   0.0673  0.0997  0.1038  0.1094   0.0944
      ma6
      0.7871
s.e.   0.0606

sigma^2 estimated as 2934529:  log likelihood = -2547.91,  aic = 5123.82

Error measures:
      ME      RMSE      MAE      MPE      MAPE
Training set 273.5049 1710.07 1225.375 -0.1126082 6.313526
      MASE      ACF1
Training set 0.4648297 -0.03599488

Forecasts:

```

### **Auto.arima Model Accuracy and Summary:**

#### **Accuracy:**

```

      ME      RMSE      MAE      MPE
Training set -79.13092 1820.459 1260.207 -0.3549272
Test set     2637.01573 4082.542 3318.157  4.8017254
      MAPE      MASE      ACF1 Theil's U
Training set 4.218518 0.4780427 -0.005840748 NA
Test set     6.349393 1.2586983 0.034942031 1.256306

```

#### **Summary:**

```

Forecast method: ARIMA(2,1,1)(0,1,2)[12]

Model Information:
Series: dataTrain
ARIMA(2,1,1)(0,1,2)[12]

Coefficients:
      ar1      ar2      ma1      sma1      sma2
      0.5017  0.2057  -0.9583  -0.4404  -0.1236
s.e.   0.0738  0.0722   0.0426   0.0676   0.0639

sigma^2 estimated as 3535011:  log likelihood=-2463.67
AIC=4939.33  AICc=4939.64  BIC=4961.03

Error measures:
      ME      RMSE      MAE      MPE      MAPE
Training set -79.13092 1820.459 1260.207 -0.3549272 4.218518
      MASE      ACF1
Training set 0.4780427 -0.005840748

Forecasts:

```

The important measure to watch our while selection models are MAPE, RMSE, AIC. Below table compares the values.

| Manual ARIMA | ME        | RMSE     | MAE      | MPE       | MAPE     | MASE      | ACF1       | Theil's U |
|--------------|-----------|----------|----------|-----------|----------|-----------|------------|-----------|
| Training Set | 273.5049  | 1710.07  | 1225.375 | 0.1126082 | 6.313526 | 0.4648297 | 0.03599488 | NA        |
| Test set     | 2152.9943 | 3244.316 | 2695.219 | 3.9521467 | 5.192869 | 1.0223951 | 0.03029999 | 0.9832872 |
| Auto.ARIMA   | ME        | RMSE     | MAE      | MPE       | MAPE     | MASE      | ACF1       | Theil's U |
| Training Set | -79.13092 | 1820.459 | 1260.207 | 0.3549272 | 4.218518 | 0.4780427 | 0.00584075 | NA        |
| Test set     | 2637.0157 | 4082.542 | 3318.157 | 4.8017254 | 6.349393 | 1.2586983 | 0.03494203 | 1.256306  |

### **Model Selection Basis:**

- Choose model with fewer parameters
- Parsimonious model: which prefers to keep more information without losing
- Examine standard errors of forecast values
- Lower SE of predictions
- Compare MSE, AIC, BIC etc to choose between models
- Smaller values preferred
- Practical considerations, if any, or domain input

The MAPE values we got from both the models are descent. The difference is not more. Below are the difference in values of train and test MAPE values for both models.

If we go by the MAPE values then manual arima performed better than auto.arima.

But I choosing model, it is better to choose a model which was built with less information loss. If we see, manual arima was built with mode information loss although the MAPE value difference is low. For Auto arima, the AIC value is less which means the loss of information is relatively less when compared with manual arima model.

AIC will maintain a striking balance between underfit and overfit

| Model        | Diff of MAPE Values of train and test | AIC Value | p,d,q values      |
|--------------|---------------------------------------|-----------|-------------------|
| Manual ARIMA | 1.120657                              | 5123.82   | ARIMA(7,1,6)[12]  |
| Auto.ARIMA   | 2.130875                              | 4939.33   | 2,1,1)(0,1,2)[12] |

**Auto.arima model performed best.**

### **13.Forecasting model for 12 periods:**

Hence the model built with auto.arima is best of the models and lets choose the same for forecasting 12 period duration.

It is always advised not to predict for longer durations due to the uncertainty of external factors.

### Summary:

```

              ME      RMSE      MAE      MPE      MAPE      MASE
Training set -79.06125 1820.459 1260.153 -0.4954352 4.077759 0.4780222
Test set     2370.16895 5095.210 4045.061  4.8788718 8.018354 1.5344398

              ACF1 Theil's U
Training set -0.005838212    NA
Test set     0.517160399  1.509983

Forecast method: ARIMA(2,1,1)(0,1,2)[12]

Model Information:
Series: dataTrain1
ARIMA(2,1,1)(0,1,2)[12]

Coefficients:
      ar1      ar2      ma1      sma1      sma2
      0.5017  0.2057 -0.9583 -0.4404 -0.1236
s.e.  0.0738  0.0722  0.0426  0.0676  0.0639

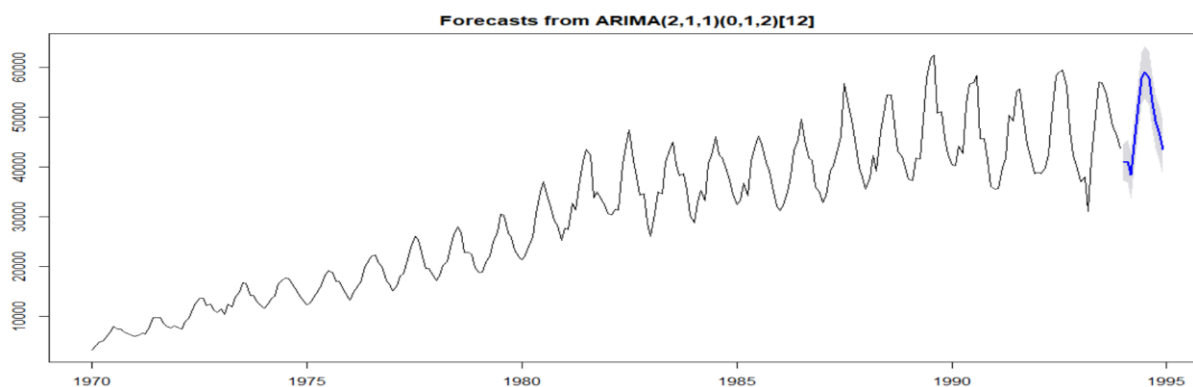
sigma^2 estimated as 3535010:  log likelihood=-2463.67
AIC=4939.33  AICc=4939.64  BIC=4961.03

Error measures:
              ME      RMSE      MAE      MPE      MAPE      MASE
Training set -79.06125 1820.459 1260.153 -0.4954352 4.077759 0.4780222
              ACF1
Training set -0.005838212

```

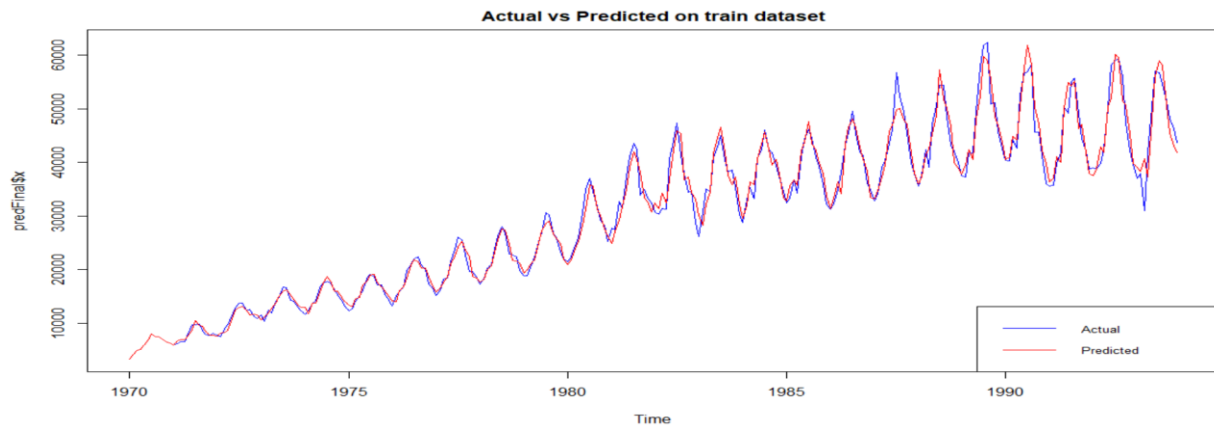
### Forecast Graph:

Below graph, we have forecasted for 12 periods further and forecasted graph is shown below.

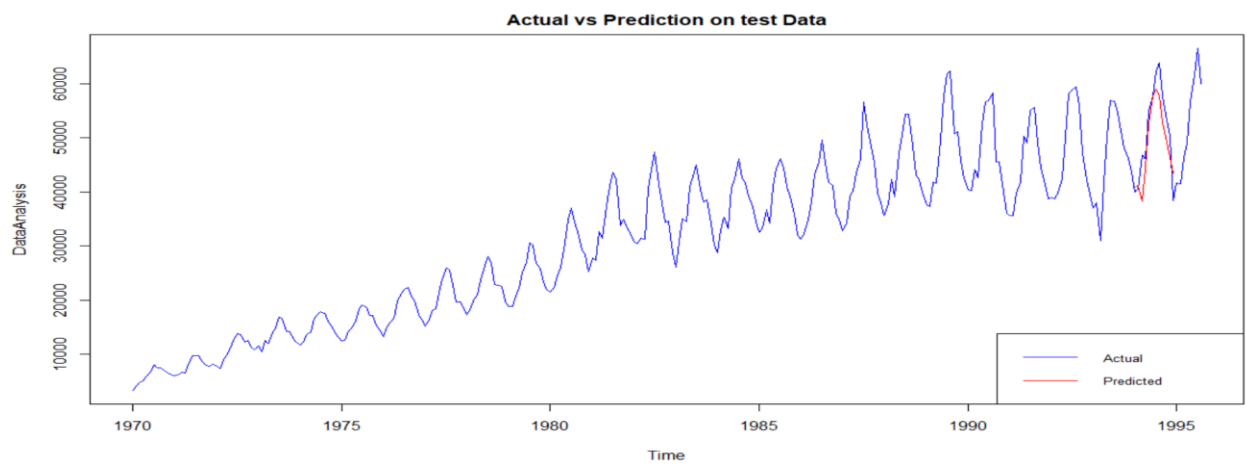




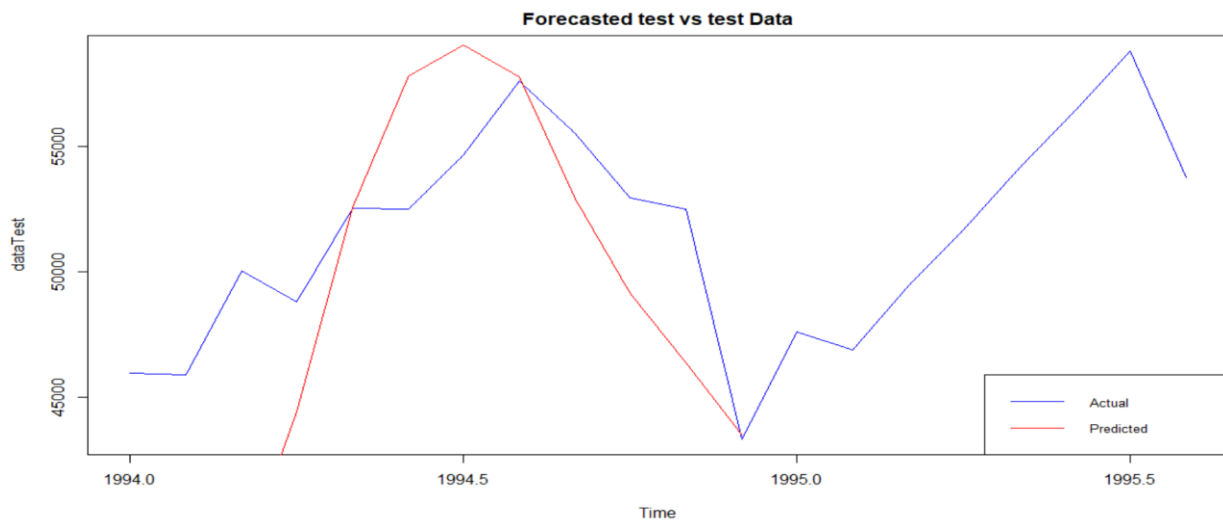
### Actual vs Predictions on Train dataset:



### Actual vs prediction on test data set:



### Predicted Test vs Actual test data:



### Accuracy:

|              | ME             | RMSE     | MAE      | MPE        | MAPE     | MASE      |
|--------------|----------------|----------|----------|------------|----------|-----------|
| Training set | -79.06125      | 1820.459 | 1260.153 | -0.4954352 | 4.077759 | 0.4780222 |
| Test set     | 2370.16895     | 5095.210 | 4045.061 | 4.8788718  | 8.018354 | 1.5344398 |
|              | ACF1 Theil's U |          |          |            |          |           |
| Training set | -0.005838212   | NA       |          |            |          |           |
| Test set     | 0.517160399    | 1.509983 |          |            |          |           |

### Summary:

```
orecast method: ARIMA(2,1,1)(0,1,2)[12]

Model Information:
Series: dataTrain1
ARIMA(2,1,1)(0,1,2)[12]

Coefficients:
      ar1      ar2      ma1      sma1      sma2
      0.5017  0.2057 -0.9583 -0.4404 -0.1236
s.e.  0.0738  0.0722  0.0426  0.0676  0.0639

sigma^2 estimated as 3535010:  log likelihood=-2463.67
AIC=4939.33  AICc=4939.64  BIC=4961.03

Error measures:
      ME      RMSE      MAE      MPE      MAPE      MASE
Training set -79.06125 1820.459 1260.153 -0.4954352 4.077759 0.4780222
      ACF1
Training set -0.005838212
```

From the above the model built with forecasting 12 months period is good and have descent MAPE and AIC values.

\*\*\*\*\*THE END\*\*\*\*\*