GAS PRODUCTION – Time Series Forecasting

SUBMITTED BY – KANUPRIYA MITTAL

MADE UNDER THE GUIDANCE OF – MR. DEEPAK GUPTA

DATE OF SUBMISSION:8TH March,2020

CONTENT:

1. PROJECTIVE OBJECTIVE

We are supposed to download the **Forecast** package in R. The package contains methods and tools for displaying and analysing univariate time series forecasts including exponential smoothing via state space models and automatic ARIMA modelling. We have to explore the **gas** (Australian monthly gas production) dataset in Forecast package to do analysing and forecasting for 12 months more using the best validated model.

2. IMPORTING THE TIME SERIES IN R

The dataset Gas is present in the Forecast Package.

The gas is already a time series. We'll assign it to an object called data.

It initially contains data from 1956 to 1996.

Rcode:

```
data<- forecast::gas
```

Forecast is package in the R language. We download it to access the time series object *gas* from it

3. EXPLORATORY DATA ANALYSIS

It is done to understand the basic data structure as well as to visualize the dataset before performing any modelling.

- STRUCTURE- To find the structure of the Dataset.

str(data)

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

- SUMMARY- Returns the details of the dataset regarding the measures of central tendency.

summary(data)

```
summary(data)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1646    2675   16788   21415   38629   66600
```

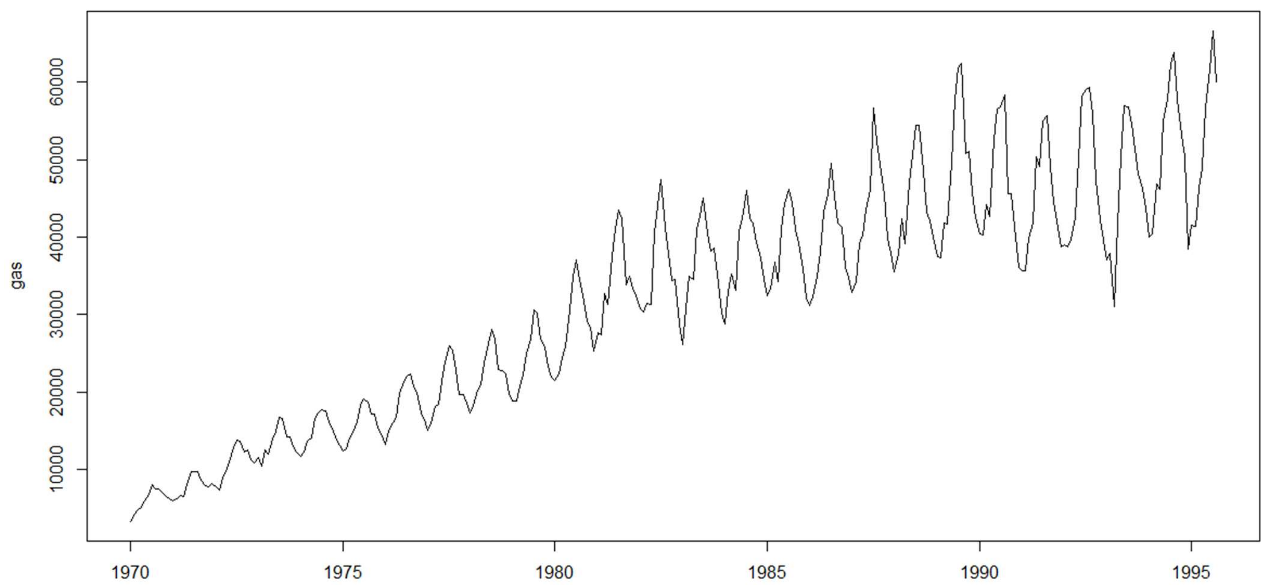
- CLASS- Returns the class of the dataset object.

class(data)

```
> class(data)

[1] "ts"
```

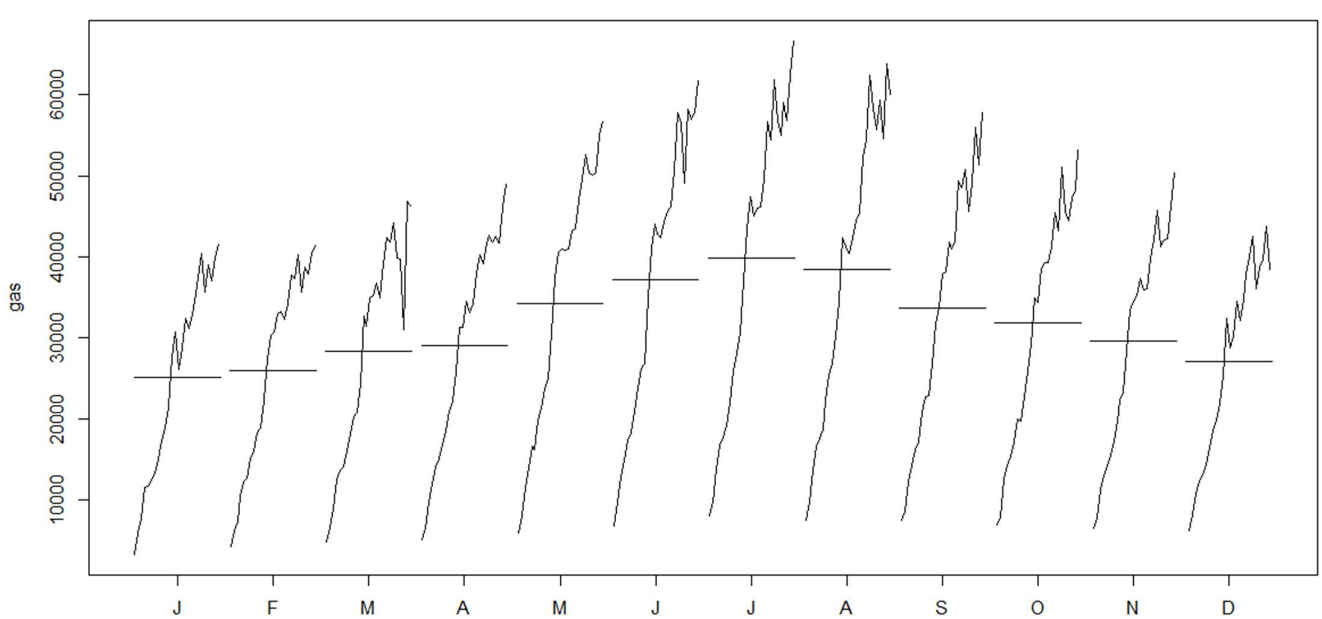
➔ We have to use the data from 1970 onwards. So, considering the time series from (1970,1) till (1995,8)

```
gas = window(data, start=c(1970,1), end= c(1995,8))
plot(gas)
```



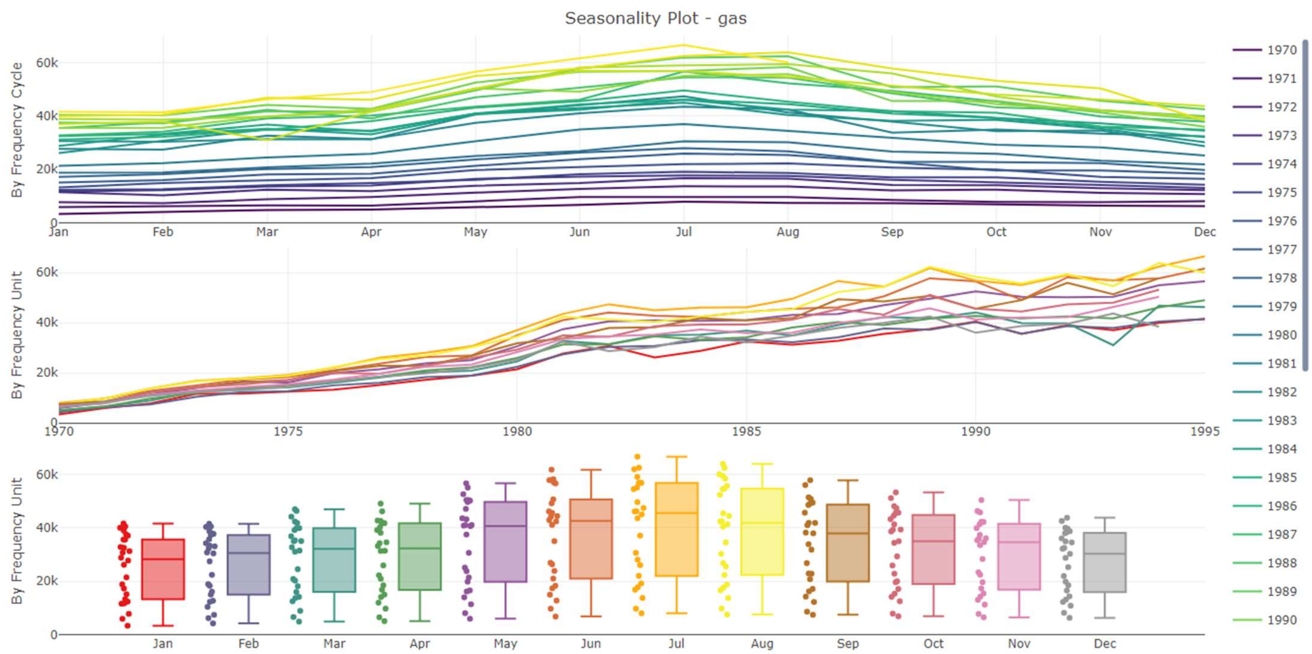
- MONTH PLOT- To visualize each month's trend and seasonality individually

```
monthplot(gas)
```



- SEASONAL PLOT – To visualize the seasonality from every perspective possible.

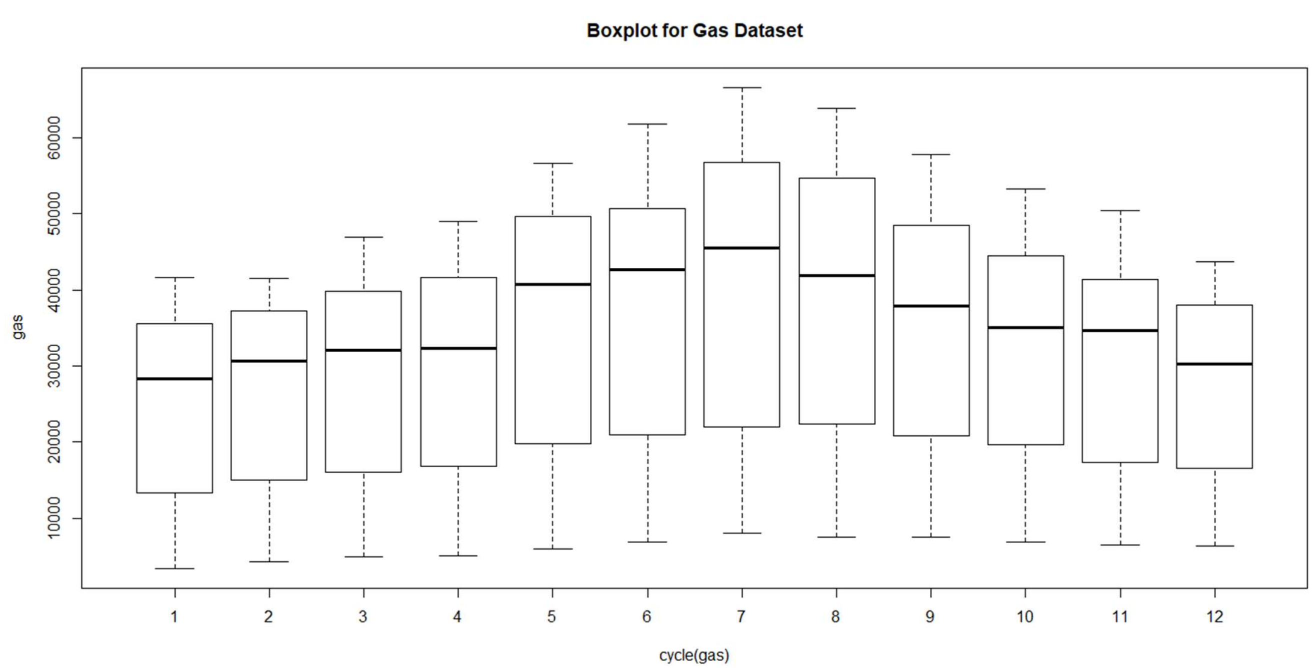
ts_seasonal(gas,type = "all")



- BOXPLOT

This function will make the boxplots of all the month individually on a single chart.

boxplot(gas~cycle(gas), main = "Boxplot for Gas Dataset")



4. IDENTIFYING THE COMPONENTS OF THE TIME SERIES

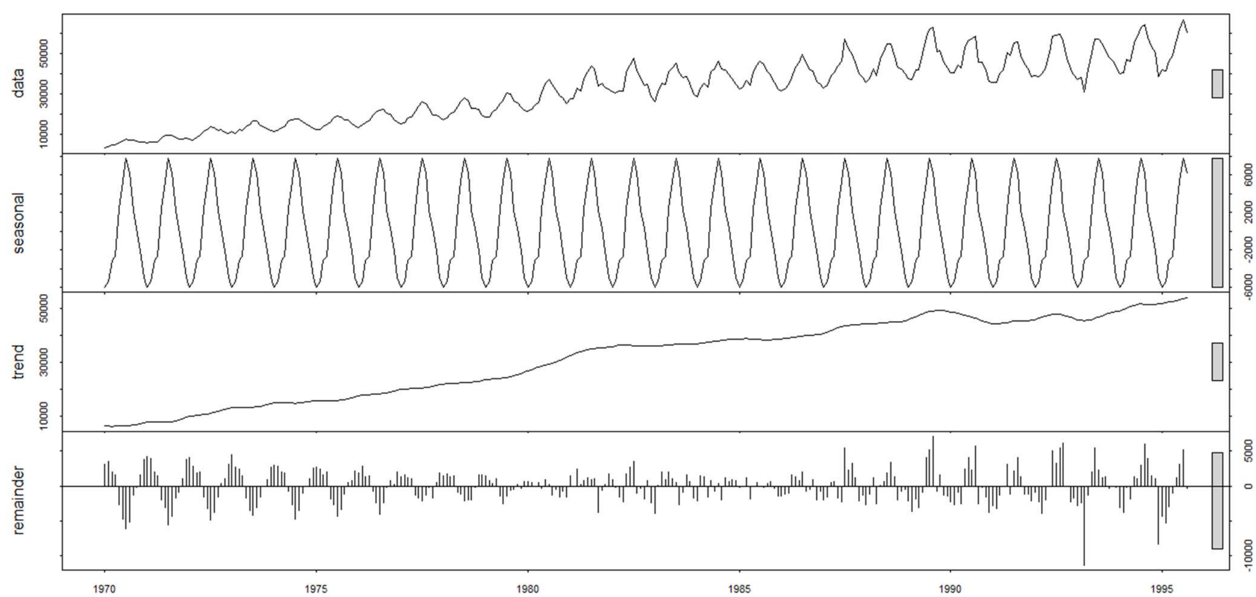
There are in general three components of a time series:

- Seasonality
- Trend
- Residuals

Decomposition of the time series into its different components can be done using stl() .

RCode:

```
decomp_gas= stl(gas_production, s.window = "periodic")
plot(decomp_gas)
```



Here, we see that Trend plays a very big role in the making of this time series. The Residuals are also a significant part if our Time series. But there is no seasonality component in here. Therefore, we can say that only Trend and residual component are present in our time series.

5. CHECKING THE PERIODICITY OF THE TIME SERIES DATA

It is a fundamental characteristic of time series data. The frequency of observations against time.

```
> periodicity(gas)
Monthly periodicity from Jan 1970 to Aug 1995
```

Therefore, in this case, we can observe that there is monthly periodicity in this time series of gas data.

6. CHECKING THE STATIONARITY OF THE SERIES

6.1 Using Augmented Dicky Fuller Test

Formation of the Hypothesis of Augmented Dickey Fuller Test

- Null Hypothesis : H0: The time series is Non-Stationary
- Alternate Hypothesis: H1: The time series is Stationary

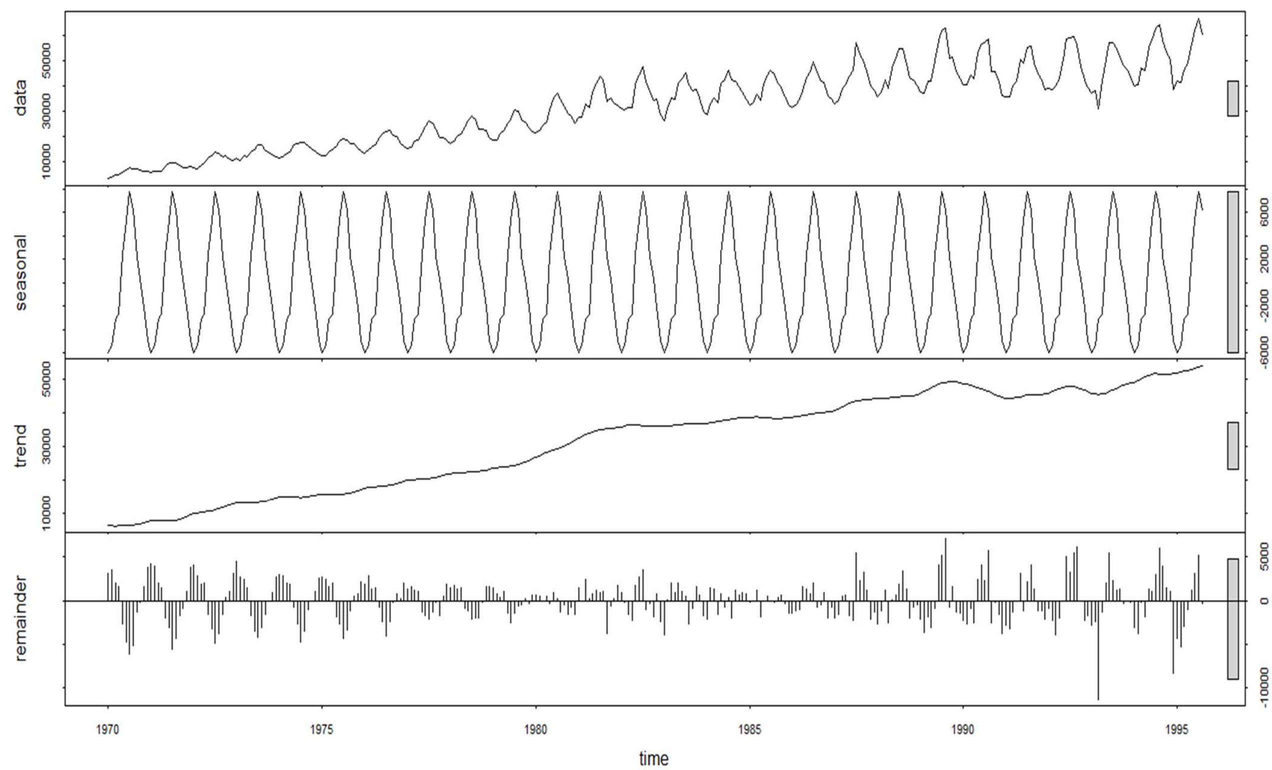
adf.test(gas, alternative = "stationary",k=12)

```
>   adf.test(gas, alternative = "stationary",k=12)

        Augmented Dickey-Fuller Test

data:  gas
Dickey-Fuller = -1.3538, Lag order = 12, p-value = 0.8488
alternative hypothesis: stationary

>
```

The p-value comes out to be .84 which is much bigger than the Alpha of 0.05. Therefore, we fail to reject the null hypothesis of the series being non stationary. We'll accept the Null Hypothesis stating that the time series is non stationary.

6.1 Using Visualization Of Decomposed Series

```
plot(decomp_gas)
```



Had the series been stationary, there would have been no Trend or Seasonality. But there is a presence of Trend in the series. Therefore, also visually talking, the series is not stationary.

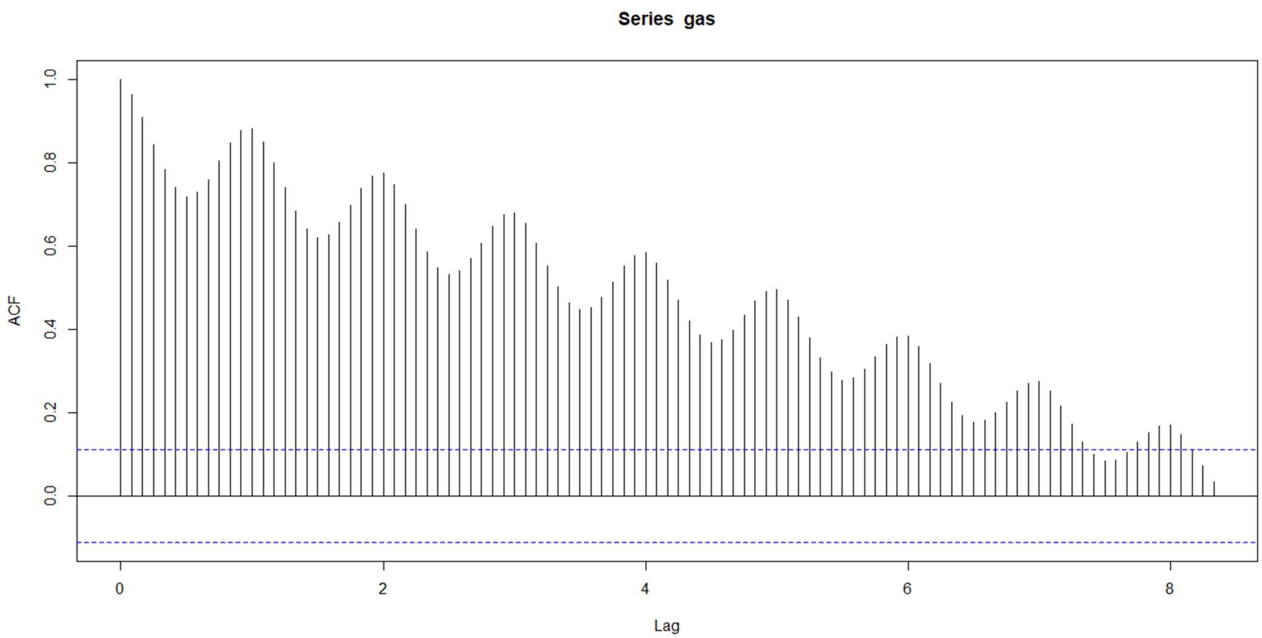
Therefore, we don't need to deseasonalize the time series because it's already unaffected by the seasonal component.

The series is non-stationary.

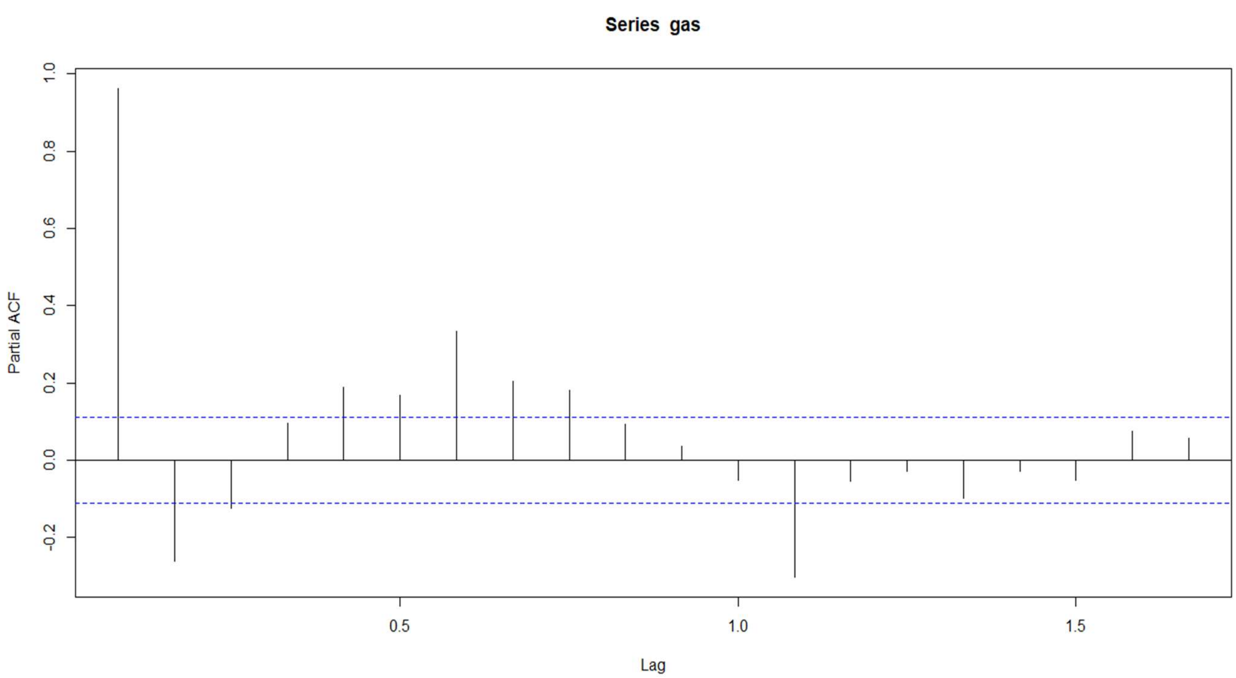
➔ Checking The Auto Correlations Of The Time Series

Autocorrelation is the mathematical representation of the degree of similarity between a given time series and a lagged version of itself over successive time intervals.

- acf(gas, lag.max = 100)



- pacf(gas,lag.max = 20)



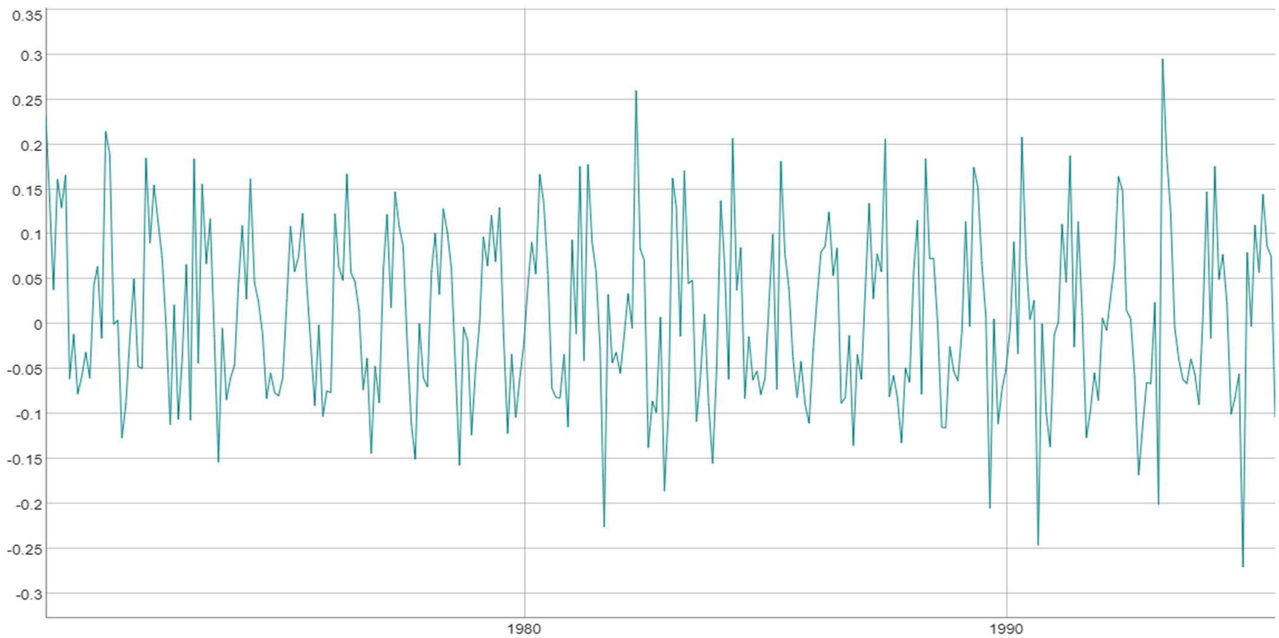
7. DIFFERENCING THE TIME SERIES DATA

A non-stationary time series can be converted into a stationary series by differencing the time series data. It is dne by subtracting the previous observation from the current observation.

Converting the time series into Stationary series.

RCode:

```
diff_gas= diff(log(gas), differences = 1)

dygraph(diff_gas)
```



➔ Checking The stationarity of the Differenced Time Series:

```
adf.test(diff_gas, alternative = "stationary", k=12)
```

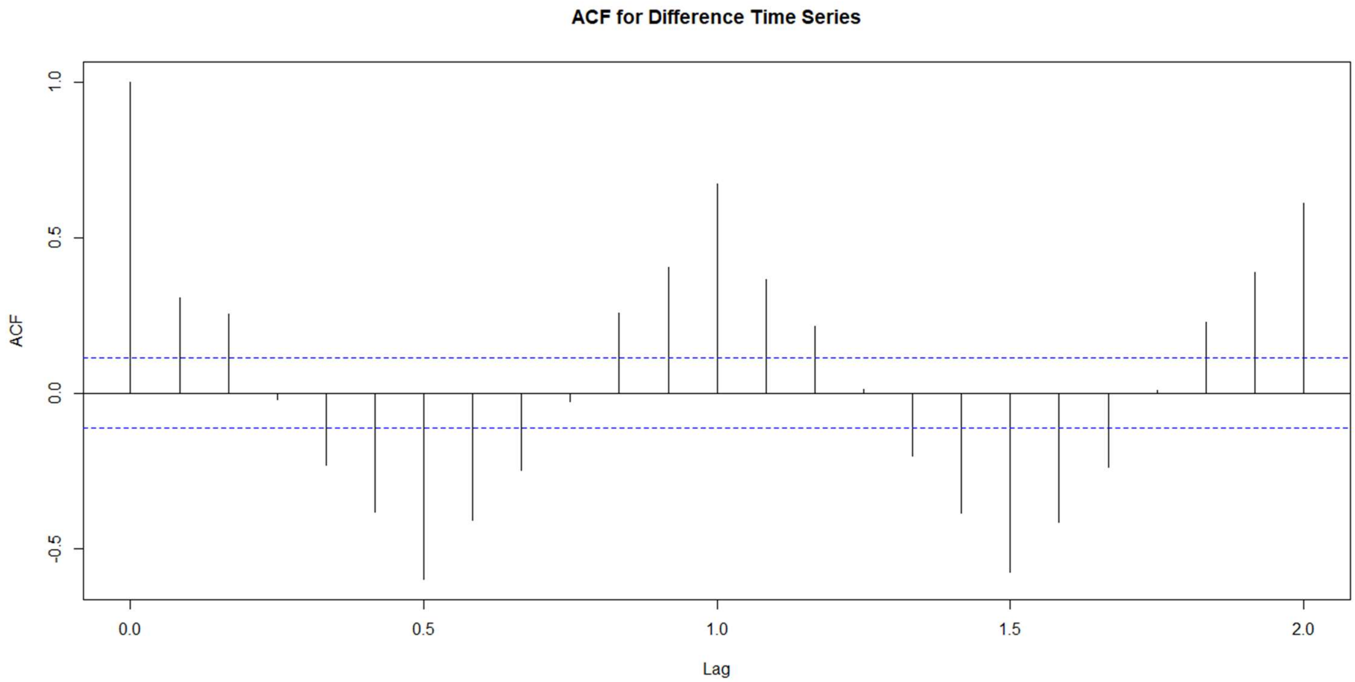
```
        Augmented Dickey-Fuller Test

data:  diff_gas
Dickey-Fuller = -5.4776, Lag order = 12, p-value = 0.01
alternative hypothesis: stationary
```

Here, we see that the p-value is much lower than our alpha of 0.05. Therefore, we know that if P-value is less than Alpha, then we reject the Null hypothesis and accept the alternative Hypothesis of the series being Stationary.

Hence, by doing the differencing of the time series one time, we are able to convert the non-stationary time series into stationary time-series.

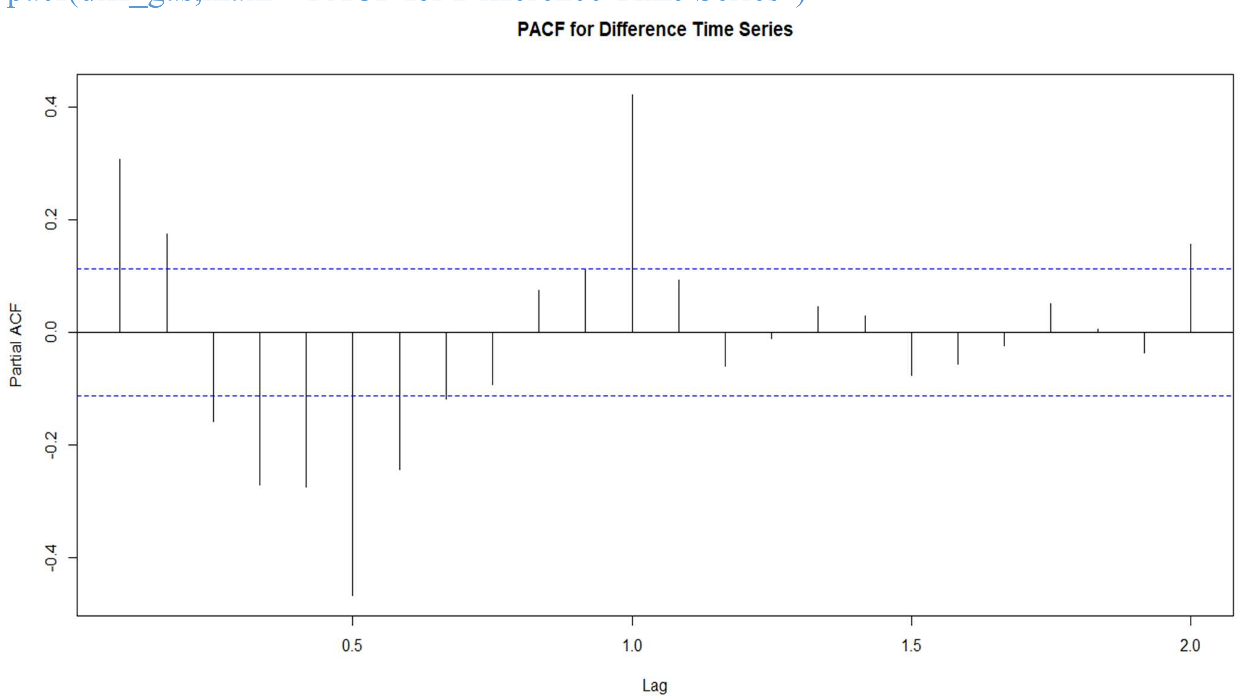
➔ Now, we'll find out the ACF and PACF representations of the differenced time series.

- acf(diff_gas,main="ACF for Difference Time Series")



#Value of q= 3

- pacf(diff_gas,main="PACF for Difference Time Series")



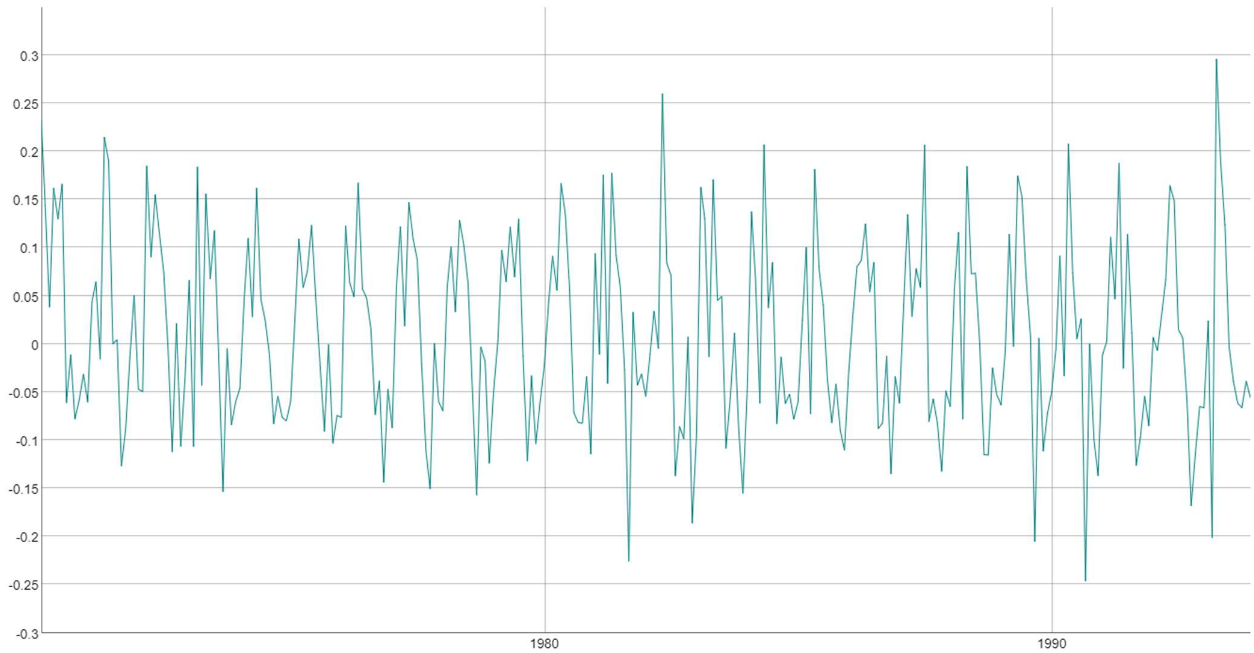
#Value of P= 2

8. SPLITTING THE TIME SERIES INTO TRAIN AND TEST PARTS

We'' use the data from Jan,1970 till Dec,1993 in the Training part.
We'll use the data from Jan,1994 till Aug,1995 in the test data.

RCode:

```
train_gas= window(diff_gas,end= c(1993,12))
dygraph(train_gas)
```

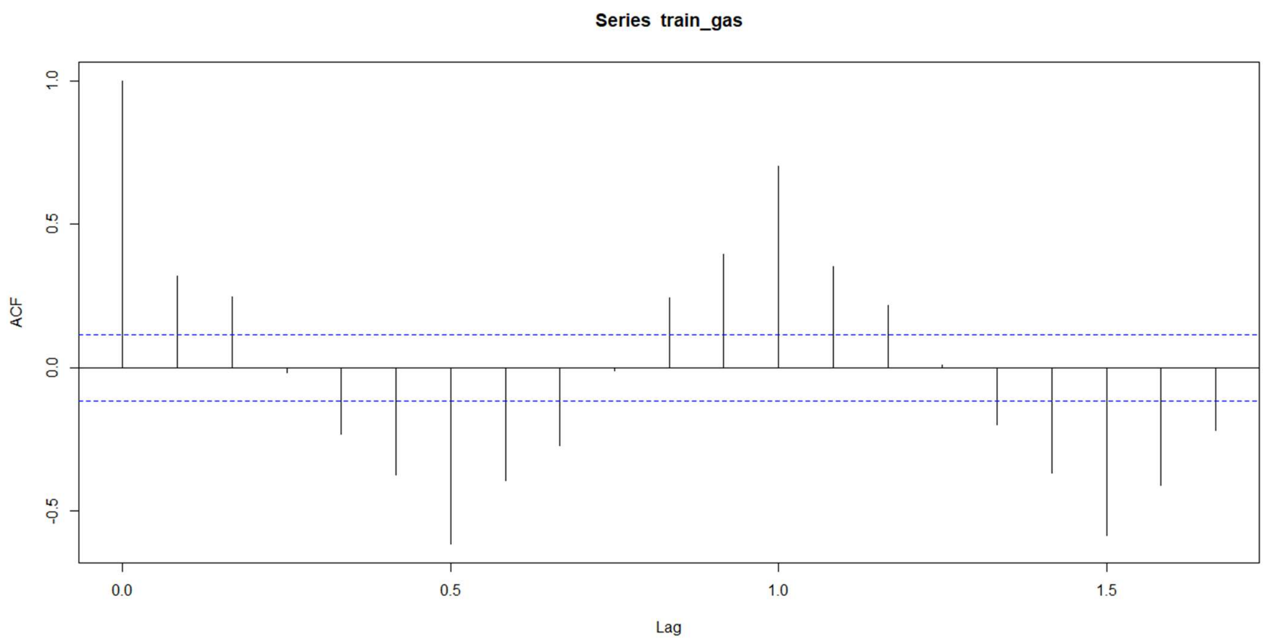



```
test_gas = window(diff_gas, start=c(1994,1))
dygraph(test_gas)
```

9. CHECKING ACF AND PACF ON THE TRAIN SET

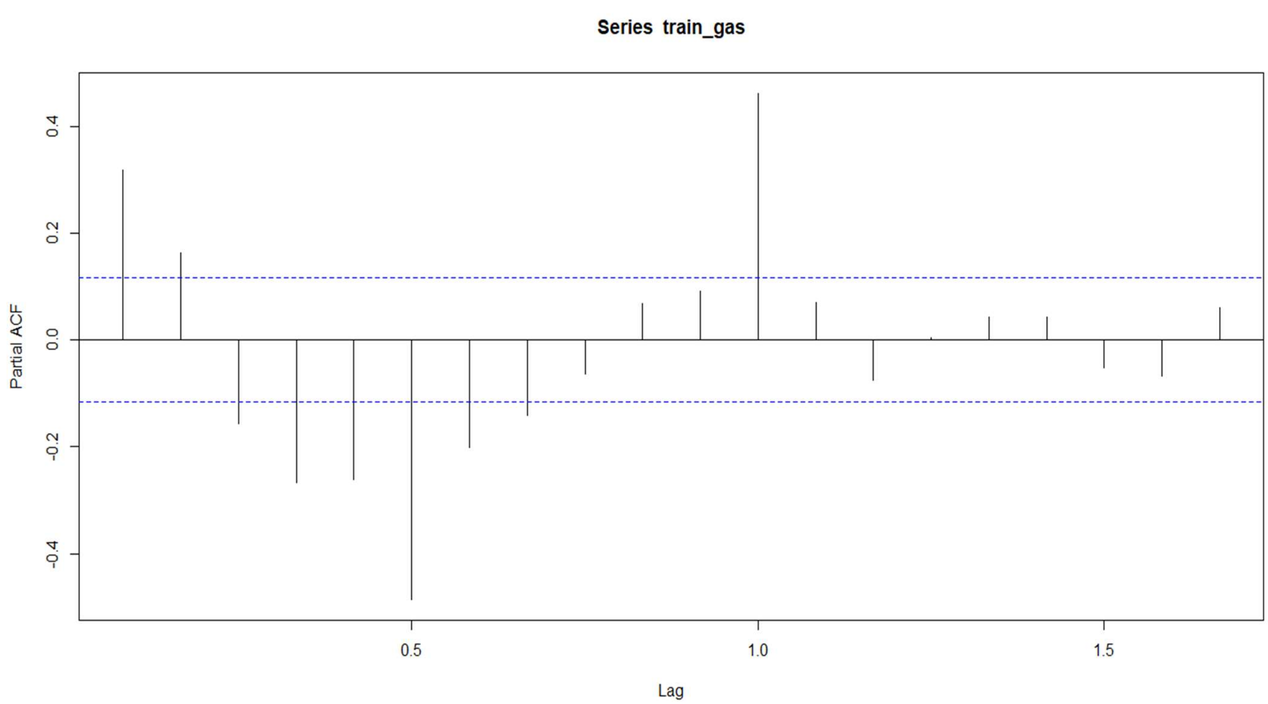
RCode:

- acf(train_gas,lag.max = 20)



#Value of q= 3

- pacf(train_gas,lag.max = 20)



Value of p= 2

10. <u>MANUAL ARIMA MODELLING</u>

It is an Auto Regressive Integrated Moving Average model that explains a given time series based on it past values, its own lags, in order to forecast future values.

10.1 Creating The Manual ARIMA Model

The manual arima takes as argumemts the values of acf , pcaf and differencing in the order of c(p,d,q)

With p=2 nd q=3

RCode:

```
manual_arima= arima(exp(train_gas), order = c(2,0,3))
summary(manual_arima)
```

```
> summary(manual_arima)

Call:
arima(x = exp(train_gas), order = c(2, 0, 3))

Coefficients:
         ar1      ar2      ma1     ma2      ma3  intercept
      1.7315  -0.9996  -2.2374  1.8432  -0.4799     1.0132
s.e.  0.0009   0.0004   0.0528  0.0935   0.0521     0.0017

sigma^2 estimated as 0.003735:  log likelihood = 388.5,  aic = -763.01

Training set error measures:
                       ME       RMSE       MAE        MPE   MAPE      MASE
Training set 0.0006422578 0.06111115 0.0461625 -0.2717222 4.6101 0.5115093
                   ACF1
Training set 0.05126293
>
```

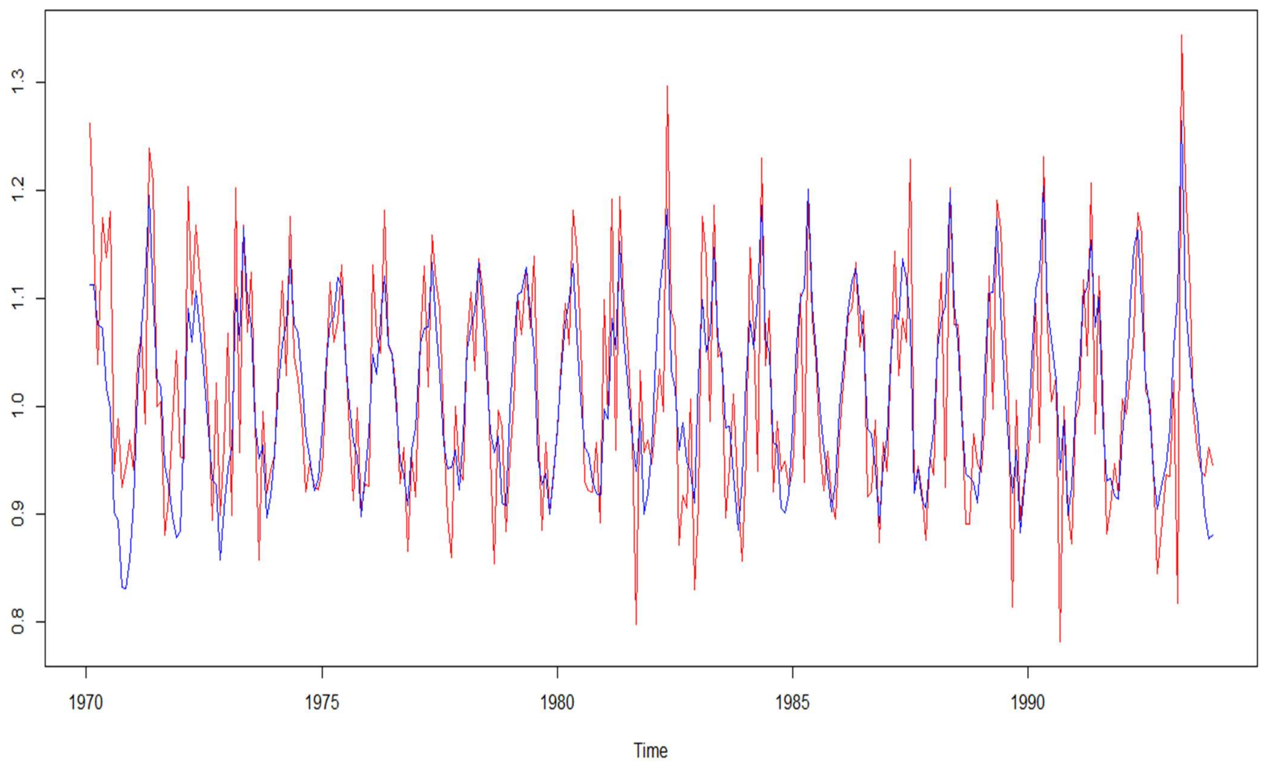
Aic = -763

10.2 Visualizing The Model-Fit

Here, we are visualizing the plots of the actual series alongside the fitted values derived from the model.

Rcode:

```
manual_fit= fitted(manual_arima)
ts.plot(exp(train_gas), manual_fit, col= c("red","blue"))
```



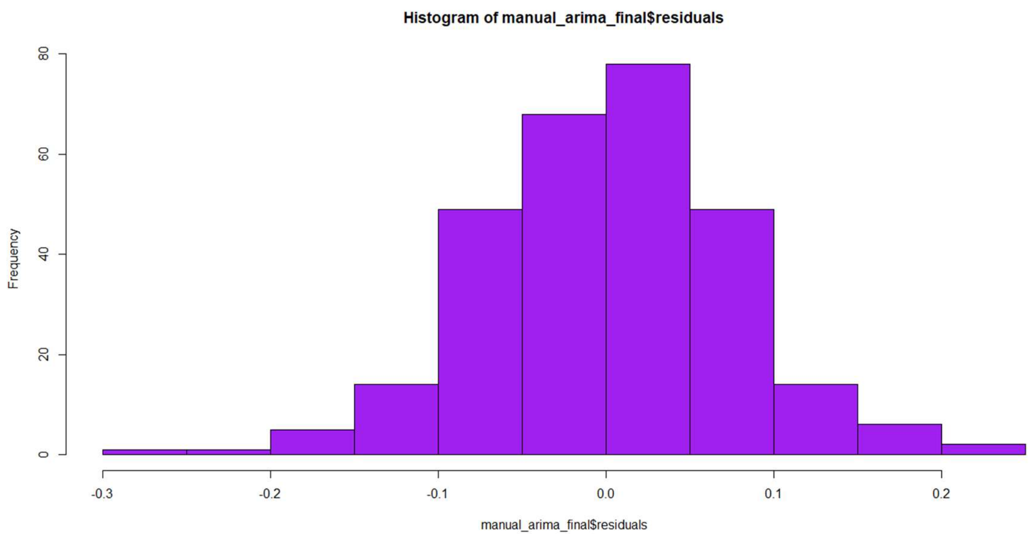
#The model accuracy is quite good as can be observed from the plot. The fitted values are following the curves of the actual values.

10.3 Interpreting The Residuals

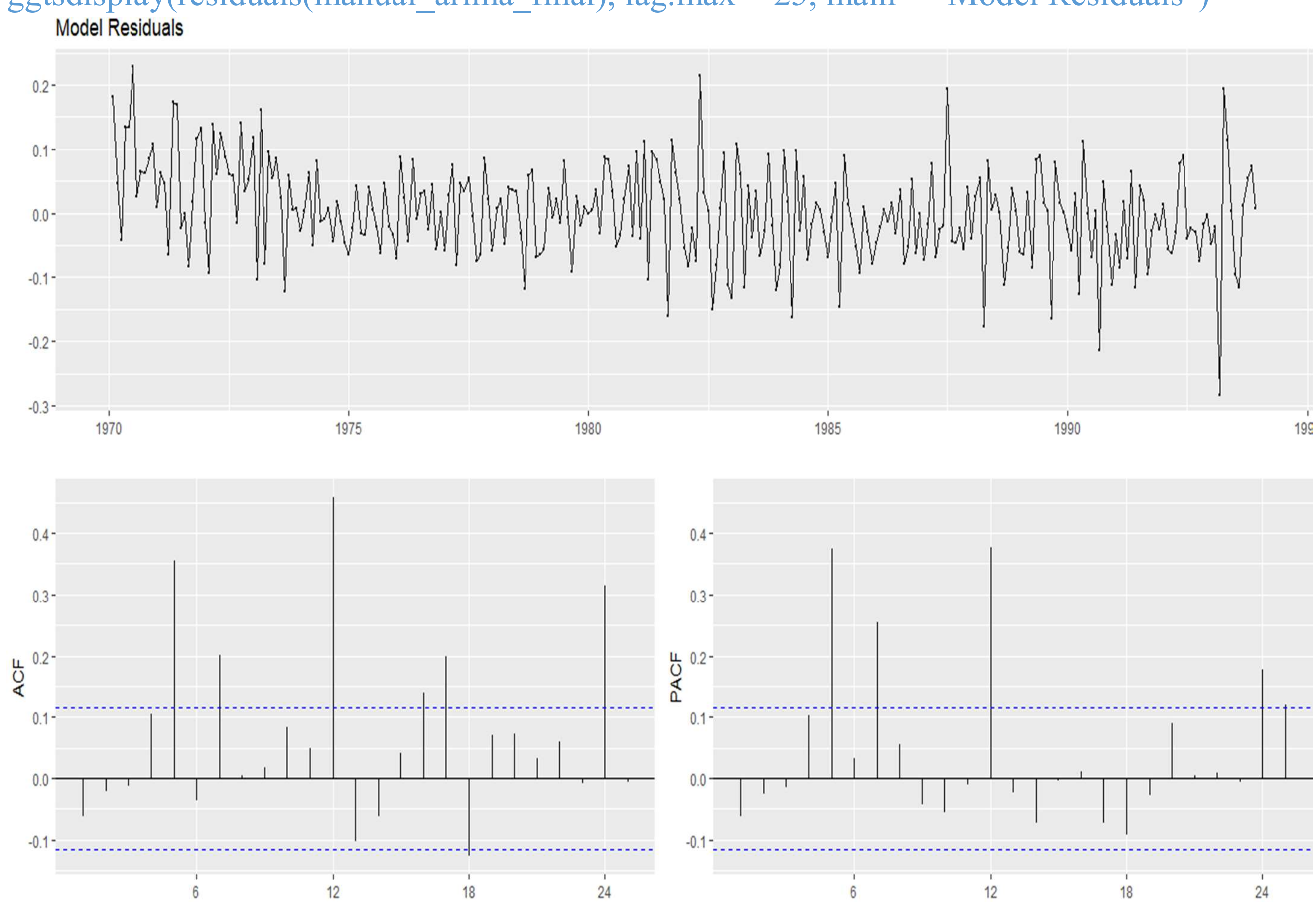
Here, we will visualize the residuals of the manual ARIMA model that we created.

10.3.1 Using Plots

- hist(manual_arima_final$residuals,col = "purple")



- ggtsdisplay(residuals(manual_arima_final), lag.max = 25, main = "Model Residuals")



10.3.2 Using Box Ljung Test

This test is used for examining the null hypothesis of independence of residuals in a given time series .

H0 : Residuals are independent
H1 : Residuals are not independent

Box.test(manual_arima$residuals)

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

        Box-Pierce test

data:  manual_arima$residuals
X-squared = 0.7542, df = 1, p-value = 0.3851

> |
```

The p-value is 0.385 which is very big as compared to alpha of 0.05.

If p is greater than alpha , we fail to reject the null hypothesis H0.

This results in signifying that THE RESIDUALS ARE INDEPENDENT.

11. AUTO ARIMA MODELLING

11.1 Creating the Auto ARIMA Model

Whilst using AUTO-ARIMA , we can directly fit our model by skipping the next steps after data preprocessing. It uses AIC and BIC values generated by trying various different combinations of p, d, and q values to fit the model.

Rcode:

```
auto_arima= auto.arima(exp(train_gas),seasonal = TRUE)
summary(auto_arima)
```

```
> summary(auto_arima)
Series: exp(train_gas)
ARIMA(1,0,1)(2,1,1)[12]

Coefficients:
         ar1      ma1    sar1    sar2     sma1
      0.1168  -0.5452  0.1913  0.0349  -0.8869
s.e.  0.1305   0.0998  0.0945  0.0841   0.0749

sigma^2 estimated as 0.003012:  log likelihood=403.54
AIC=-795.07   AICc=-794.76   BIC=-773.37

Training set error measures:
                     ME       RMSE        MAE       MPE     MAPE      MASE
Training set -0.0112713 0.05323167 0.03857387 -1.336179 3.872699 0.7345801
                    ACF1
Training set -0.02972483
>
```

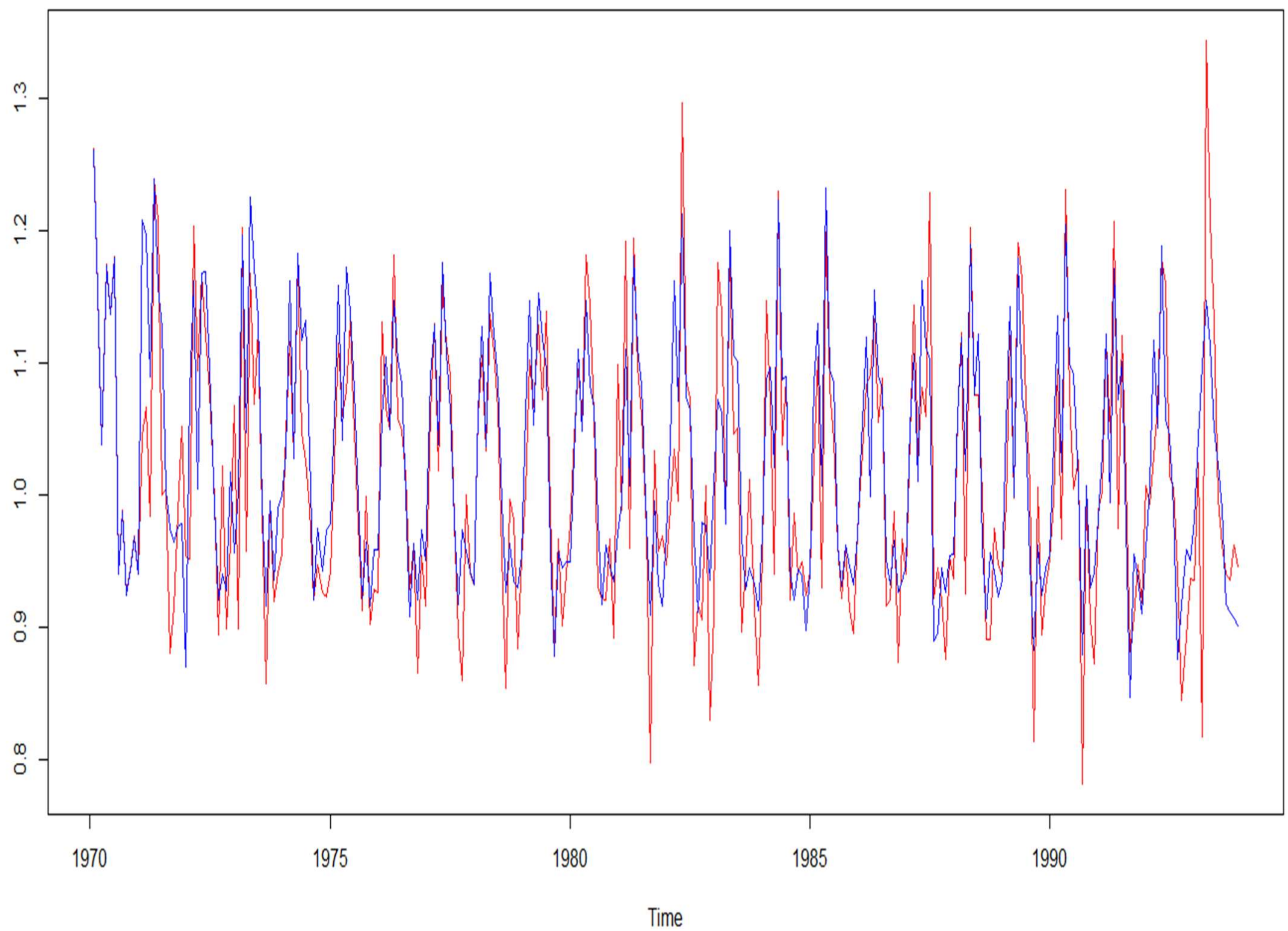
Aic = -795.07

11.2 Visualizing The Model Fit

Here, we are visualizing the plots of the actual series alongside the fitted values derived from the model.

Rcode:

```
auto_fit<- fitted(auto_arima)
ts.plot(exp(train_gas), auto_fit, col= c("red","blue"))
```



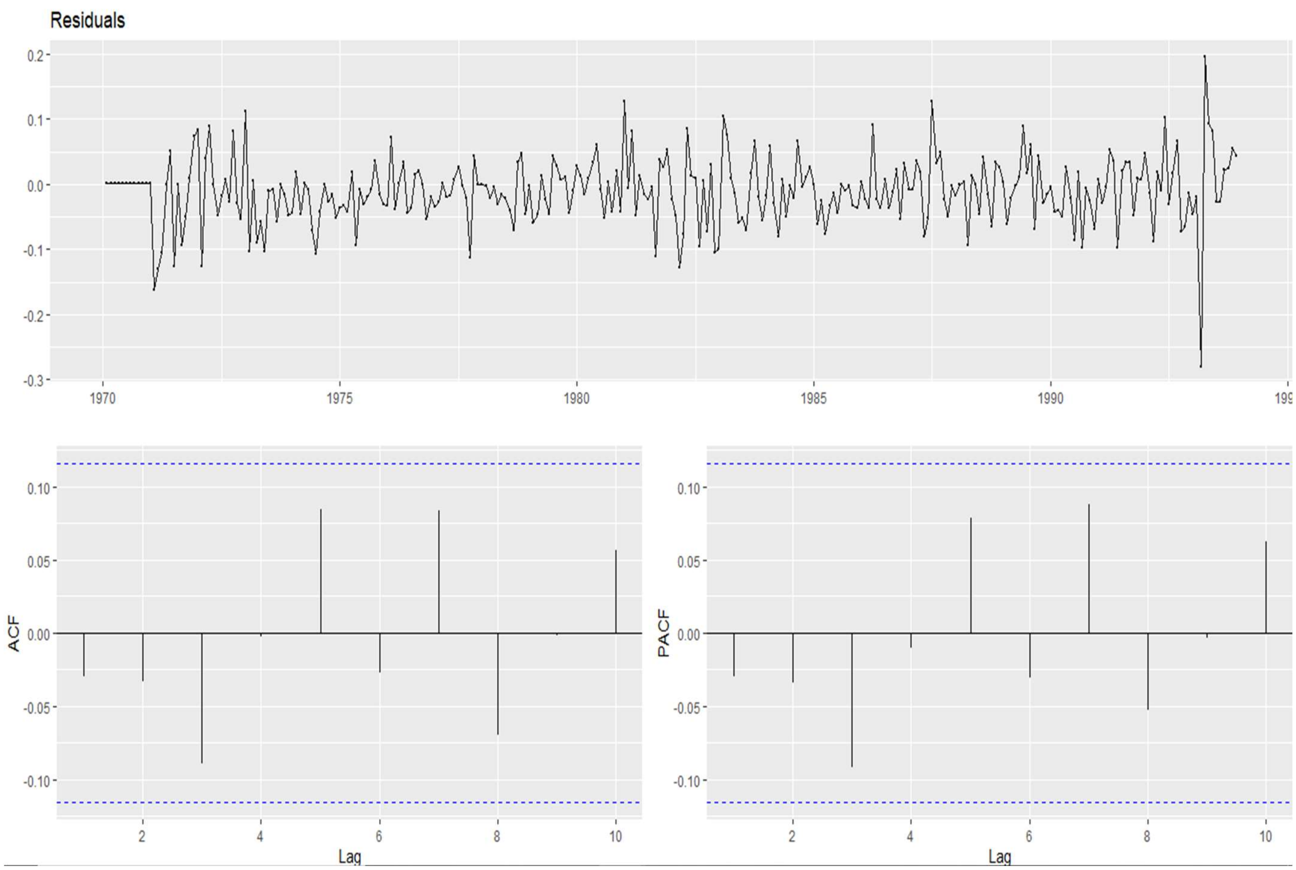
The model fit seems quite good after visualization. The fitted values are following the curves of the actual values.

11.3 Interpreting The Residuals Of The Auto Arima Model

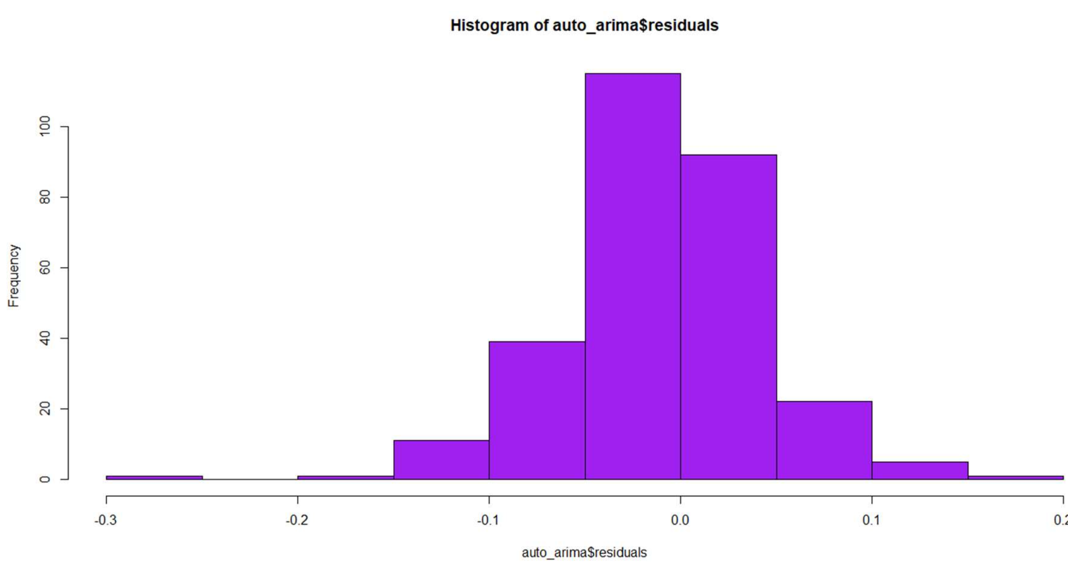
Here, we will visualize the residuals of the manual ARIMA model that we created.

11.3.1 Using Plots

- ggtsdisplay(residuals(auto_arima),lag.max = 10, main = "Residuals")



- hist(auto_arima$residuals,col = "purple")



11.3.2 Using Box-Ljung Test

This test is used for examining the null hypothesis of independence of residuals in a given time series .

H0 : Residuals are independent
H1 : Residuals are not independent

RCode:

Box.test(auto_arima$residuals)

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

        Box-Pierce test

data:  auto_arima$residuals
X-squared = 0.25358, df = 1, p-value = 0.6146

>
```

The p-value is 0.614 which is very big as compared to alpha of 0.05.

If p is greater than alpha , we fail to reject the null hypothesis H0.

This results in signifying that THE RESIDUALS ARE INDEPENDENT.

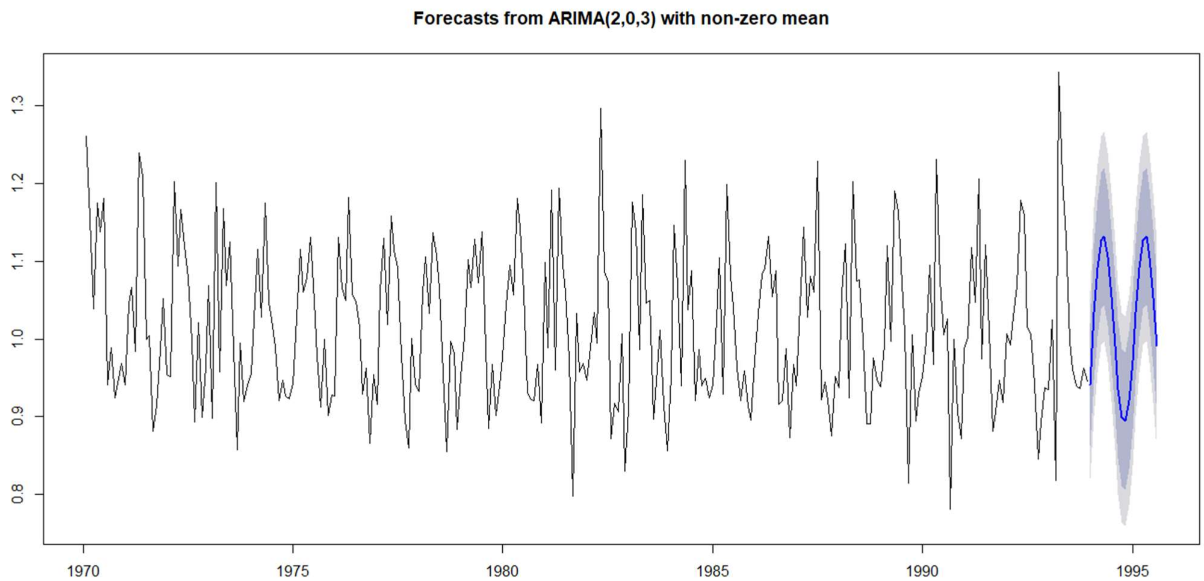
12. FORECASTING ON ARIMA MODELS

We have already analyzed the times series and now we will forecast the data using the ARIMA models that we have built

12.1 Manual ARIMA Forecasting

RCode:

```
manual_forecast= forecast(manual_arima,h=20)
plot(manual_forecast)
```




```
accuracy(manual_forecast,exp(test_gas))
```

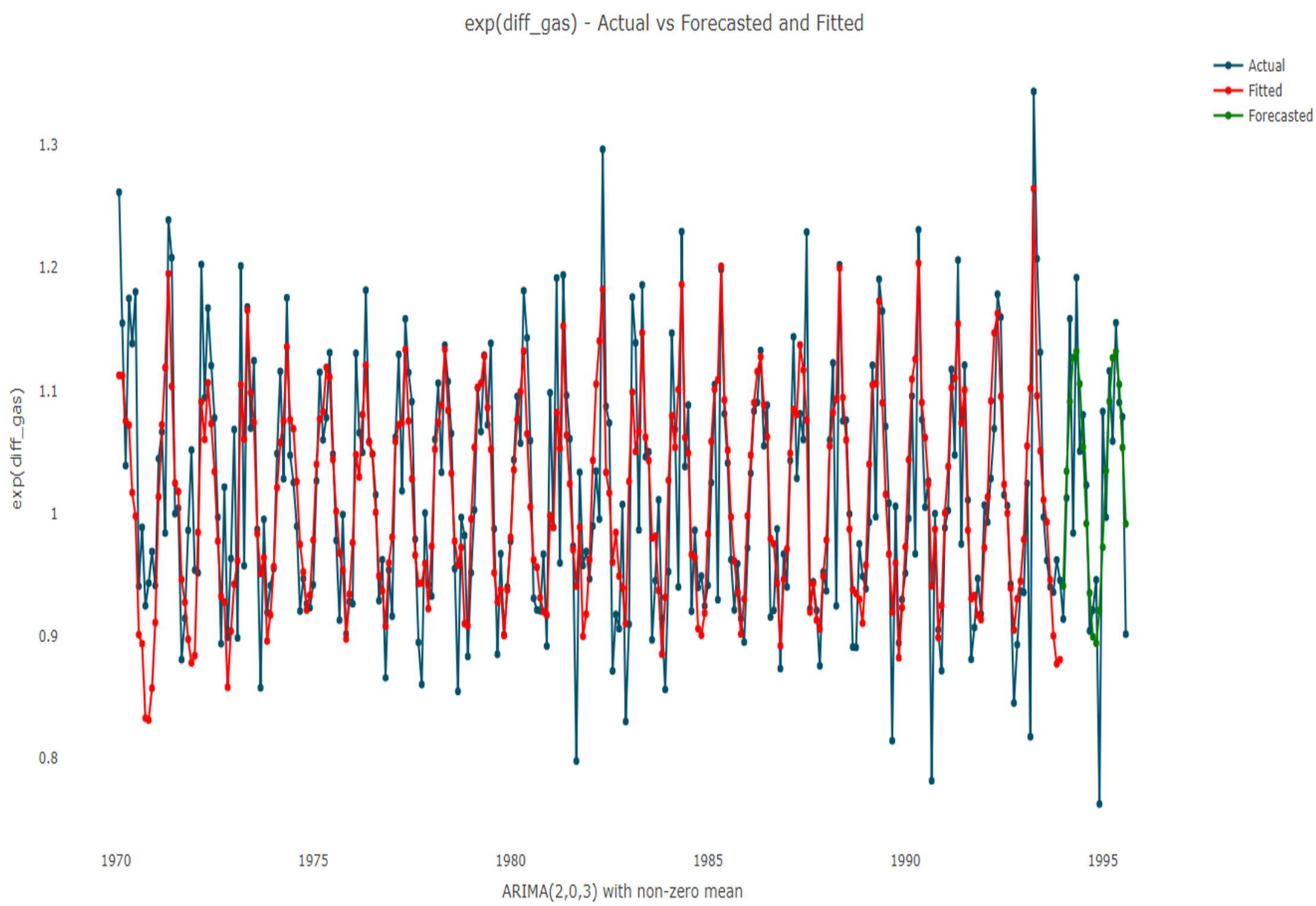
```
> accuracy(manual_forecast,exp(test_gas))
                        ME       RMSE        MAE        MPE    MAPE
Training set  0.0006422578 0.06111115 0.04616250 -0.2717222 4.61010
Test set     -0.0102156701 0.06779089 0.05440361 -1.4820666 5.57312
                  MASE        ACF1 Theil's U
Training set 0.8790939  0.05126293        NA
Test set     1.0360332 -0.68492823 0.4822607
>
```

The Accuracy on:
Train Data= 4.610
Test Data= 5.573

MAPE on TRAIN: 4.610
MAPE on TEST   : 5.573

➔ Visualization Of the Actual, Fitted and Forecasted values

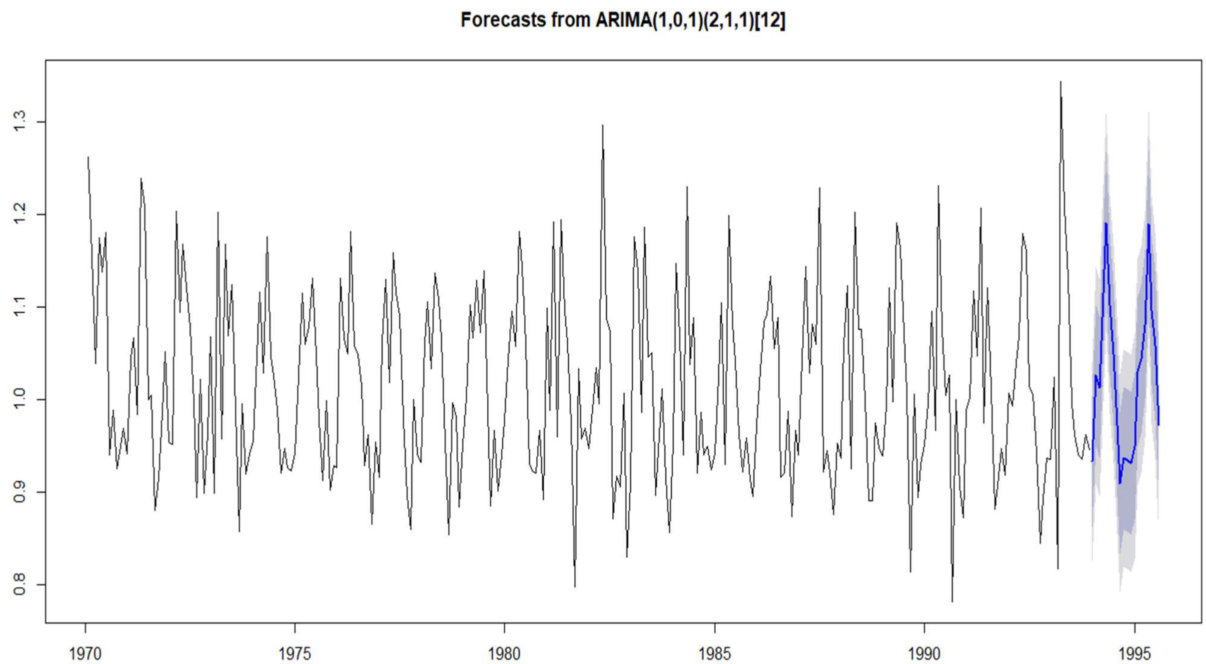
```
test_forecast(forecast.obj = manual_forecast,actual = exp(diff_gas),
              test = exp(test_gas))
```



12.2 Auto ARIMA Forecasting

RCode:

auto_forecast = forecast(auto_arima,h=20)
plot(auto_forecast)



accuracy(auto_forecast,exp(test_gas))

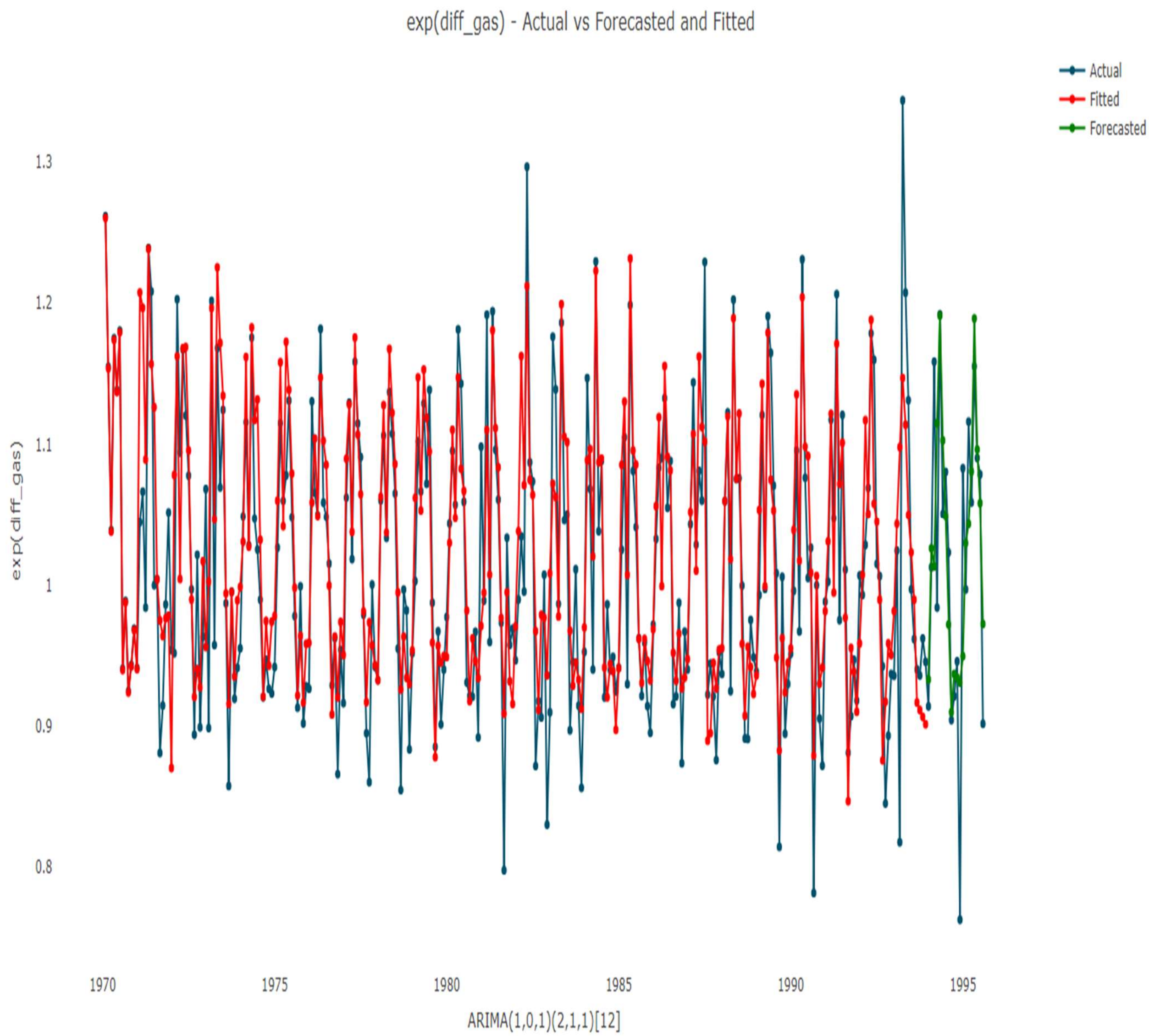
```
> accuracy(auto_forecast,exp(test_gas))
                       ME       RMSE        MAE       MPE     MAPE
Training set -0.011271298 0.05323167 0.03857387 -1.336179 3.872699
Test set     -0.005177004 0.07248664 0.05187003 -1.013439 5.261249
                  MASE        ACF1 Theil's U
Training set 0.7345801 -0.02972483        NA
Test set     0.9877851 -0.52404102 0.5281561
>
```

The Accuracy on:
Train Data= 3.8726
Test Data= 5.261

MAPE on TRAIN: 3.872
MAPE on TEST   : 5.262

➔ Visualization Of the Actual, Fitted and Forecasted values

```
test_forecast(forecast.obj = auto_forecast,actual = exp(diff_gas),
              test = exp(test_gas))
```



13. SELECTING THE FINAL BEST MODEL

We select the best model by comparing the AIC values. Aic is a relative idea. So, the model with the lower Aic value will be selected.

➔ Comparing the AIC values:

```
> summary(manual_arima)

Call:
arima(x = exp(train_gas), order = c(2, 0, 3))

Coefficients:
         ar1      ar2      ma1     ma2      ma3  intercept
      1.7315  -0.9996  -2.2374  1.8432  -0.4799     1.0132
s.e.  0.0009   0.0004   0.0528  0.0935   0.0521     0.0017

sigma^2 estimated as 0.003735:  log likelihood = 388.5,  aic = -763.01

Training set error measures:
                       ME       RMSE       MAE        MPE   MAPE      MASE
Training set 0.0006422578 0.06111115 0.0461625 -0.2717222 4.6101 0.5115093
                   ACF1
Training set 0.05126293
>
```

MANUAL ARIMA AIC = -763

VS

```
> summary(auto_arima)
Series: exp(train_gas)
ARIMA(1,0,1)(2,1,1)[12]

Coefficients:
         ar1      ma1    sar1    sar2     sma1
      0.1168  -0.5452  0.1913  0.0349  -0.8869
s.e.  0.1305   0.0998  0.0945  0.0841   0.0749

sigma^2 estimated as 0.003012:  log likelihood=403.54
AIC=-795.07   AICc=-794.76   BIC=-773.37

Training set error measures:
                     ME       RMSE        MAE       MPE     MAPE      MASE
Training set -0.0112713 0.05323167 0.03857387 -1.336179 3.872699 0.7345801
                    ACF1
Training set -0.02972483
>
```

AUTO ARIMA AIC = -795

AIC is always the lower the better.

So, we'll chose AUTO ARIMA with (p,d,q) as (1,0,1) as our FINAL MODEL and do the further forecasts of 12 month by using this model.

➔ Comparing MAPE Values

MANUAL ARIMA :

#MAPE on TRAIN: 4.610

#MAPE on TEST : 5.573

AUTO ARIMA:

#MAPE on TRAIN: 3.872

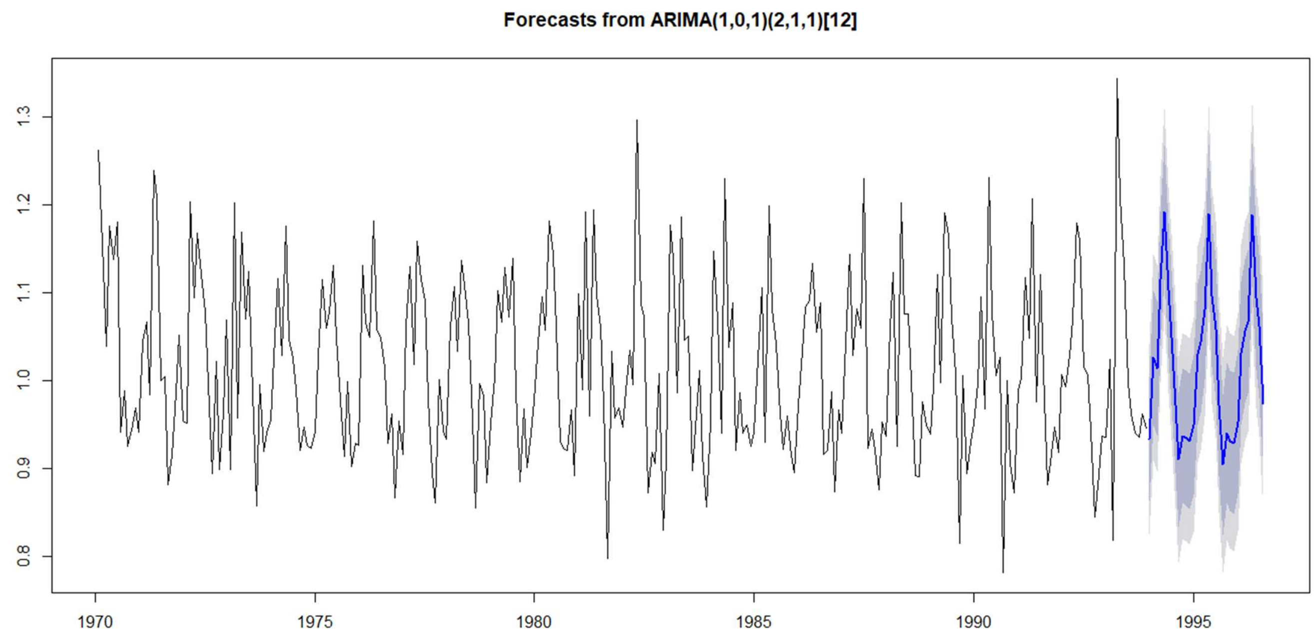
#MAPE on TEST : 5.262

From these MAPE values too, we can interpret that the Auto ARIMA model performs better than the manual ARIMA model.

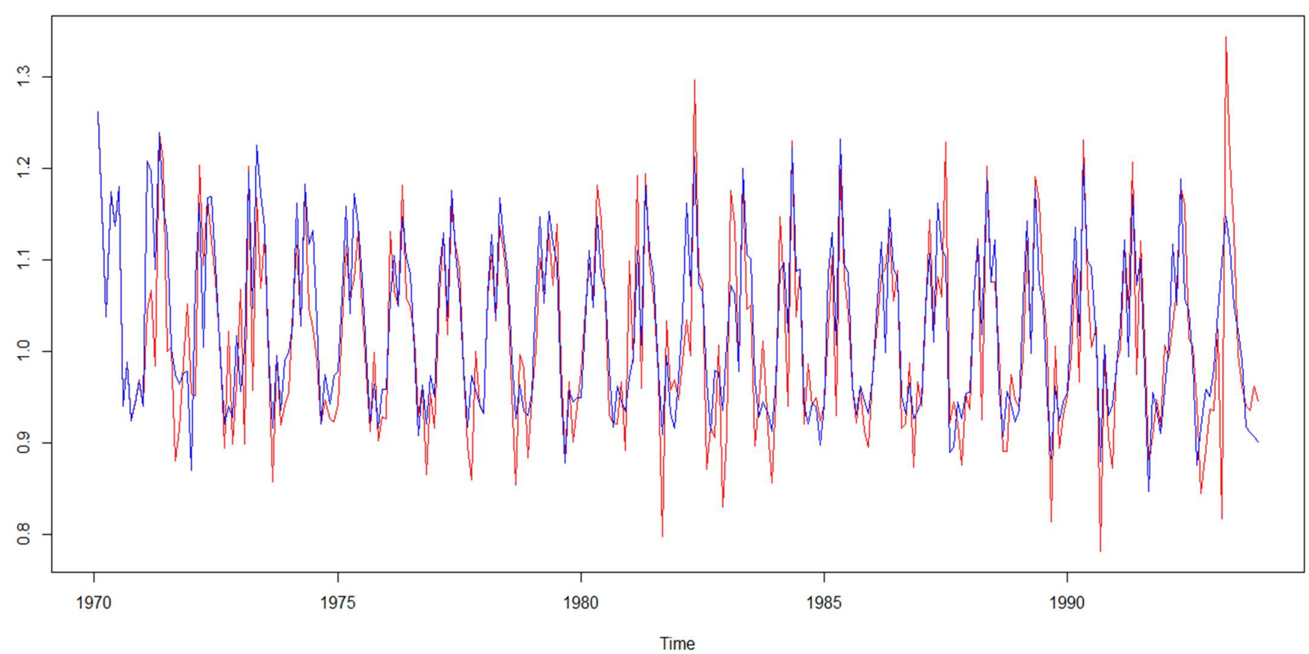
14. USING FINAL MODEL TO FORECAST FOR UNSEEN 12 MONTHS

Rcode:

```
final= forecast(auto_arima,h=32)
plot(final)
```

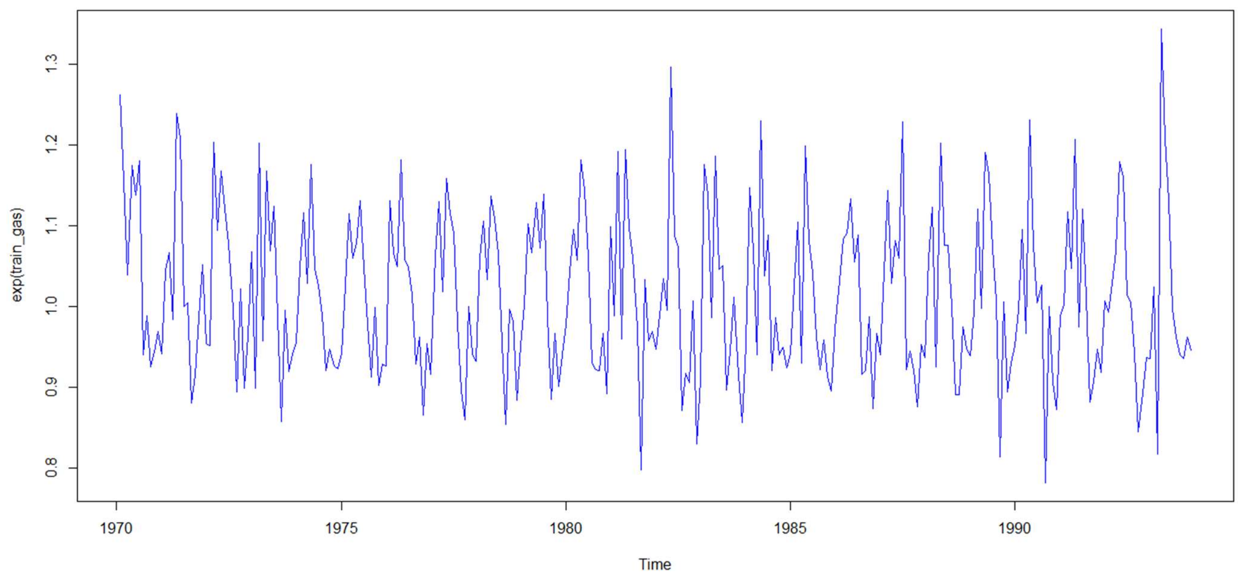


- ts.plot(exp(train_gas),auto_fit,col=c("red","blue"))

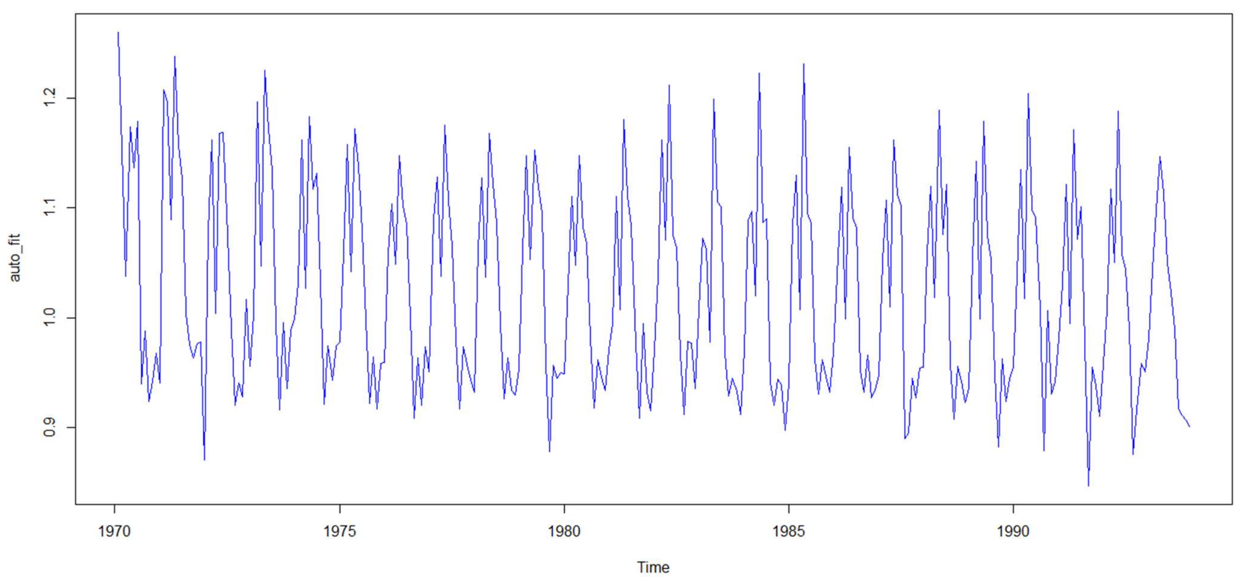


➔ Individual plots of the actual and the fitted values

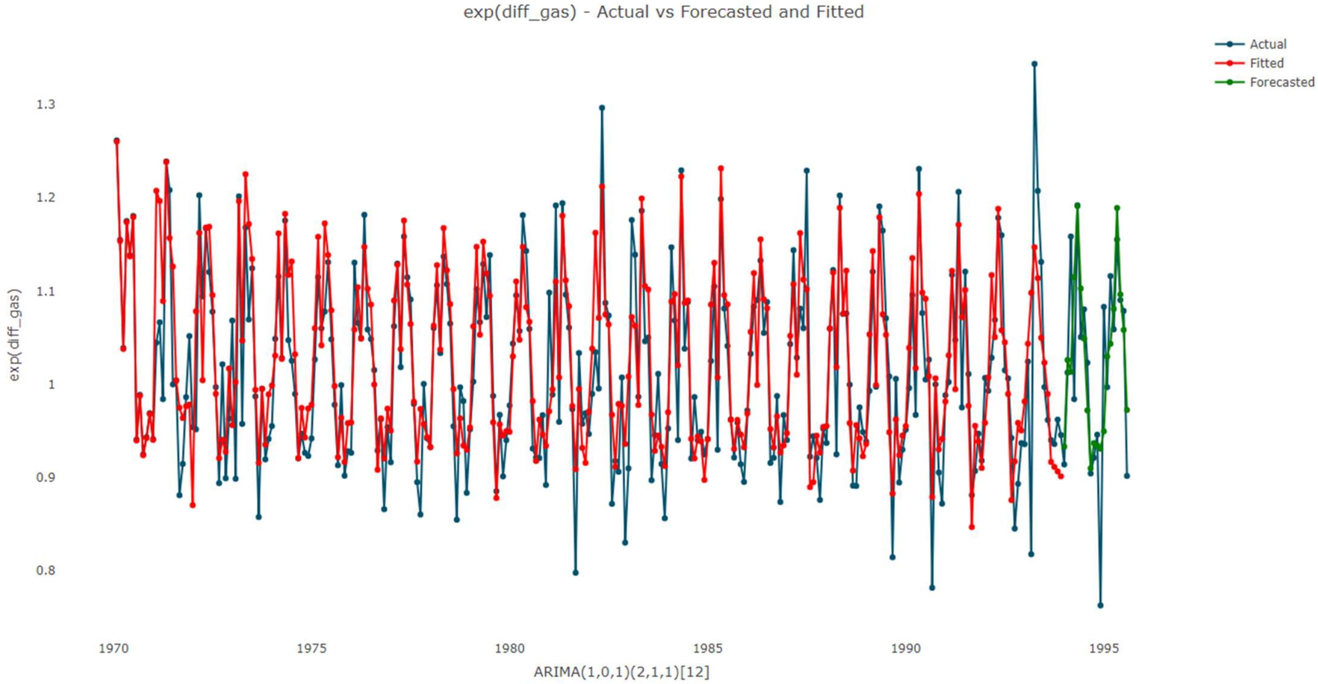
- plot(exp(train_gas),col=c("blue"))



- plot(auto_fit,col=c("blue"))



- test_forecast(forecast.obj = auto_forecast,actual = exp(diff_gas),test = exp(test_gas))



Forecasted values are:

```
> final
         Point Forecast     Lo 80     Hi 80     Lo 95    Hi 95
Jan 1994      0.9331032 0.8627447 1.0034616 0.8254992 1.040707
Feb 1994      1.0260986 0.9495594 1.1026377 0.9090420 1.143155
Mar 1994      1.0131648 0.9365448 1.0897848 0.8959846 1.130345
Apr 1994      1.1149565 1.0383354 1.1915776 0.9977746 1.232138
May 1994      1.1912375 1.1146163 1.2678586 1.0740555 1.308419
Jun 1994      1.1027965 1.0261754 1.1794176 0.9856146 1.219978
Jul 1994      1.0490206 0.9723994 1.1256417 0.9318387 1.166202
Aug 1994      0.9720431 0.8954220 1.0486643 0.8548612 1.089225
Sep 1994      0.9098542 0.8332330 0.9864753 0.7926723 1.027036
Oct 1994      0.9369667 0.8603456 1.0135878 0.8197848 1.054149
Nov 1994      0.9343482 0.8577271 1.0109694 0.8171663 1.051530
Dec 1994      0.9309410 0.8543198 1.0075621 0.8137591 1.048123
Jan 1995      0.9495487 0.8699794 1.0291181 0.8278579 1.071240
Feb 1995      1.0297884 0.9496962 1.1098806 0.9072980 1.152279
Mar 1995      1.0434669 0.9633676 1.1235663 0.9209655 1.165968
Apr 1995      1.0808071 1.0007076 1.1609066 0.9583055 1.203309
May 1995      1.1891417 1.1090422 1.2692412 1.0666401 1.311643
Jun 1995      1.0963517 1.0162522 1.1764511 0.9738501 1.218853
Jul 1995      1.0583431 0.9782436 1.1384426 0.9358415 1.180845
Aug 1995      0.9724575 0.8923581 1.0525570 0.8499559 1.094959
Sep 1995      0.9039704 0.8238709 0.9840699 0.7814688 1.026472
Oct 1995      0.9403399 0.8602404 1.0204394 0.8178383 1.062842
Nov 1995      0.9314419 0.8513424 1.0115413 0.8089403 1.053943
Dec 1995      0.9284330 0.8483336 1.0085325 0.8059315 1.050935
Jan 1996      0.9526047 0.8711933 1.0340161 0.8280967 1.077113
Feb 1996      1.0305538 0.9489111 1.1121965 0.9056921 1.155416
Mar 1996      1.0560842 0.9744382 1.1377301 0.9312174 1.180951
Apr 1996      1.0662925 0.9846465 1.1479385 0.9414257 1.191159
May 1996      1.1881743 1.1065283 1.2698203 1.0633075 1.313041
Jun 1996      1.0941255 1.0124795 1.1757715 0.9692587 1.218992
Jul 1996      1.0619413 0.9802953 1.1435873 0.9370745 1.186808
Aug 1996      0.9728961 0.8912501 1.0545421 0.8480293 1.097763
>
```

The forecasted values for just the new 12 months are:

```
> final
         Point Forecast     Lo 80     Hi 80     Lo 95    Hi 95
Sep 1995      0.9039704 0.8238709 0.9840699 0.7814688 1.026472
Oct 1995      0.9403399 0.8602404 1.0204394 0.8178383 1.062842
Nov 1995      0.9314419 0.8513424 1.0115413 0.8089403 1.053943
Dec 1995      0.9284330 0.8483336 1.0085325 0.8059315 1.050935
Jan 1996      0.9526047 0.8711933 1.0340161 0.8280967 1.077113
Feb 1996      1.0305538 0.9489111 1.1121965 0.9056921 1.155416
Mar 1996      1.0560842 0.9744382 1.1377301 0.9312174 1.180951
Apr 1996      1.0662925 0.9846465 1.1479385 0.9414257 1.191159
May 1996      1.1881743 1.1065283 1.2698203 1.0633075 1.313041
Jun 1996      1.0941255 1.0124795 1.1757715 0.9692587 1.218992
Jul 1996      1.0619413 0.9802953 1.1435873 0.9370745 1.186808
Aug 1996      0.9728961 0.8912501 1.0545421 0.8480293 1.097763
>
```

- Hence, we have derived the forecasted values for next 12 months i.e., from Sep,1995 till Aug,1996
- We used the Auto ARIMA model that deals directly with our original dataset.

The Accuracy of the Model is

*MAPE on TRAIN: 3.872

*MAPE on TEST   : 5.262

RCODE:

```
install.packages("forecast")
install.packages("tseries")
install.packages("dygraphs")
install.packages("TTR")
install.packages("xts")
install.packages("TSstudio")

library(forecast)
library(tseries)
library(dygraphs)
library(TTR)
library(xts)
library(TSstudio)

data<- forecast::gas
str(data)
summary(data)
class(data)
gas = window(data, start=c(1970,1), end= c(1995,8))
plot(gas)



#-------------------------------------------------------------------------


  #Visualizing the time series
plot.ts(gas)

monthplot(gas)
```

```
ts_seasonal(gas,type = "all")

boxplot(gas~cycle(gas), main = "Boxplot for Gas Dataset")
#---------------------------------------------------------------------------

# Decomposition of the time series into its different components

decomp_gas= stl(gas_production, s.window = "periodic")
plot(decomp_gas)
# Here, we see that Trend plays a very big role in the making of this
# time series. The Residuals are also a significant part if our Time series.
# But there is no seasonality component in here. Therefore, we can say that only
# Trend and residual component are present in our time series.




#---------------------------------------------------------------------

# Checking the Periodicity of the Time-series data

periodicity(gas)



#------------------------------------------------------------------------



# Check the Stationarity of the Series

# Using Augmented Dickey Fuller Test
# Formation of The Hypothesis of Augmented Dickey Fuller Test
```

```
#   Null Hypothesis :      H0: The time series is Non-Stationary
#  Alternate Hypothesis:  H1: The time series is Stationary
  adf.test(gas, alternative = "stationary",k=12)
   # The p-value comes oput to be .84 which is much bigger than the Alpha of 0.05.
# Therefore, we fail to reject the null hypothesis of the series being non_stationary.
# We'll accept the Null Hypothesis stating that the time series is
# non-staionary.

  # Using Visual Interpretaion from the Decomposed plots

plot(decomp_gas)

# Had the series been stationary, there would have been no Trend or Seasonality.
# But there is a presence of Trend in the series. Therefore, also visually
# talking , the series is not stationary.

# Therefore, we don't need to deseasonlize the time series because its already
# unaffected by the seasonal component.

# The series is non-stationary.

# Checking the Correaltions
acf(gas, lag.max = 100)
pacf(gas,lag.max = 20)

#-----------------------------------------------------------------
# Converting the time series into Stationary series.
# using Differencing of the data

diff_gas= diff(log(gas), differences = 1)
```

```
dygraph(diff_gas)
adf.test(diff_gas, alternative = "stationary",k=12)



# here, we see that the p-value is much lowwer than our alpha of 0.05.
# Therefore, we know thta if P-value is less than Alpha, then we reject the Null
# hypothesis and accept the laterantive Hypothesis of the series being
# Stationary.

# Hence, by doing the differencing og the time series one time, we are able to convert the
# non-stationary time series into stationary time-series.

# Now, we'll find out the ACF and PACF for the differenced time series.
acf(diff_gas,main="ACF for Difference Time Series")
pacf(diff_gas,main="PACF for Difference Time Series")




# From here , we'll take q=3 as ACf value gives us the value of q to be taken.
# we'll take p=2 as PACF gives us the value of p to be taken.

# Splitting the Time series data set into Train and Test parts.
# We" use the data from 1970 inthe Training part .
# We'll use the data from 1994 for the test data .
train_gas= window(diff_gas,end= c(1993,12))
dygraph(train_gas)
test_gas = window(diff_gas, start=c(1994,1))
dygraph(test_gas)

acf(train_gas,lag.max = 20)
pacf(train_gas,lag.max = 20)
```

```
# Therfore, the value of p= 2, q=3

# Creating Manual Arima Models
# The manual arima takes as argumemts the values of acf , pcaf and
# differencing in the order of c(p,d,q)

# With p=2 nd q=3
manual_arima= arima(exp(train_gas), order = c(2,0,3))
summary(manual_arima)

# Aic= -763

manual_fit= fitted(manual_arima)
ts.plot(exp(train_gas), manual_fit, col= c("red","blue"))

# Interpretation of the Residuals of the mManual Arima Model

hist(manual_arima_final$residuals,col = "purple")

ggtsdisplay(residuals(manual_arima_final), lag.max = 25, main = "Model Residuals")


# Performaning L-jung Box-Test

H0 : Residuals are independent
H1 : Residuals are not independent

Box.test(manual_arima$residuals)

#-------------------------------------------------------------------------
```

```
# Applying Auto Arima

auto_arima= auto.arima(exp(train_gas),seasonal = TRUE)
summary(auto_arima)

# Aic = -795.07



auto_fit<- fitted(auto_arima)

ts.plot(exp(train_gas), auto_fit, col= c("red","blue"))
#----------------------------------------------------------------------

# Interpreting The Residuals of the Auto Arima Model
ggtsdisplay(residuals(auto_arima),lag.max = 10, main = "Residuals")

hist(auto_arima$residuals,col = "purple")



# Performaning L-jung Box-Test

H0 : Residuals are independent
H1 : Residuals are not independent



Box.test(auto_arima$residuals)

# Forecasting with arima model

# Manual Arima Forecasting
```

```
manual_forecast= forecast(manual_arima,h=20)
plot(manual_forecast)
accuracy(manual_forecast,exp(test_gas))

#  The Accuracy on:
# Train Data= 4.610
# Test Data= 5.573

test_forecast(forecast.obj = manual_forecast,actual = exp(diff_gas),
          test = exp(test_gas))

# Auto Arima Forecasting

auto_forecast = forecast(auto_arima,h=20)
plot(auto_forecast)
accuracy(auto_forecast,exp(test_gas))

#  The Accuracy on:
# Train Data= 3.8726
# Test Data= 5.261

test_forecast(forecast.obj = auto_forecast,actual = exp(diff_gas),
          test = exp(test_gas))

#-----------------------------------------------------------------------
# The auto arima performs better than manual Arima
# Forecasting on 12 months ahead
final= forecast(auto_arima,h=32)
plot(final)
ts.plot(exp(train_gas),auto_fit,col=c("red","blue"))
plot(exp(train_gas),col=c("blue"))
```

```
plot(auto_fit,col=c("blue"))
```

```
test_forecast(forecast.obj = auto_forecast,actual = exp(diff_gas),test = exp(test_gas))
```