Time Series Analysis in R for ARMA, ARIMA and Seasonal ARIMA models

—

Time Series analysis has become a topic of interest for many research work at present. Hence, Following codebook aim to analyse the three different time series related to 3 different topics (Leisure and tourism, Gross Domestic Product at market prices:(£m), and Statistics of Women in full time employment) with quarterly, annual, and monthly frequencies, respectively to provide insights specially in the context of forecasting time series for future values.

Initially, it is required to install relevant Packages and Libraries for the analysis. As such following Tasks has been performed;

Install Packages needed for the analysis. Then, comment out the code after run this once, not to execute it again.

```
#install.packages("magrittr")
#install.packages("forecast")
#install.packages("fpp2")
#install.packages("tseries")
#install.packages("TSA")
```

InstallCall/Load the libraries that is required for the analysis.(Massage and warning has been eliminated when taking the final report out to have a smooth presentable output)

```
library(magrittr) # For %>% operators
library(forecast) # For forecasts (BIC criteria, ARIMA)
```

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

```
library(fpp2) # For forecasts
```

```
## -- Attaching packages --------------------------------------------- fpp2 2.4 --
```

```
## v ggplot2   3.3.5     v expsmooth 2.3
## v fma       2.4
```

```
##
```

```
library(tseries)# for adf.test (Dicky fuller test)
library(TSA) # To compute simple extended acf
```

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

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

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

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

```
#Kintr function produce R-Mark Down Report - ready to be customized to process other file format
knitr::opts_chunk$set(echo = TRUE)

#Set working directory
#setwd("D:/Desktop/USW/Data Mining/cw2/OS_visits_to_UK.csv")
```

#APPLICATION OF SEASONAL ARIMA FOR A QUARTERLY DATA SET

##Task 1 – Exploratory Data Analysis

The data set provided with this assignment is called `OS visits to UK:All visits Thousands-NSA', which is about the number of overseas visits to UK during the period of 1980 to 2019. The quarterly data available in this data set has been used in the analysis below. However, monthly,quarterly & annual values are also separately available in the data set. Thus, the functions have performed below.

Source : (https://www.ons.gov.uk/peoplepopulationandcommunity/leisureandtourism/timeseries/gmaa/ott (https://www.ons.gov.uk/peoplepopulationandcommunity/leisureandtourism/timeseries/gmaa/ott))

1.1 Import the file saved locally in the computer by using read.csv from base R.The first seven raw's of the data set has been eliminated since those consist with the Meta data of the file.

```
all_data <- read.csv("D:/Desktop/USW/Data Mining/cw2/OS_visits_to_UK.csv", skip=7)
```

1.2 View of imported data

```
#View structure of imported data
str(all_data)
```

```
## 'data.frame':    684 obs. of  2 variables:
##  $ Important.notes: chr  "1980" "1981" "1982" "1983" ...
##  $ X              : int  12419 11451 11638 12464 13642 14450 13897 15565 15799 17339 ...
```

This data set includes 684 observations and 2 variables.

```
# View first 5 observations (Annual data)
all_data %>% head(5)
```

```
##   Important.notes     X
## 1            1980 12419
## 2            1981 11451
## 3            1982 11638
## 4            1983 12464
## 5            1984 13642
```

Selection of the raws which contains quarterly data that is raw 41 to 201 as below.

```
#Select only for quarterly overseas visits data and transpose it to a vector
quaterly_data <- as.vector(t(all_data[41:201,2]))
quaterly_data
```

```
##   [1]  2081  3240  4738  2360  1920  3008  4261  2262  2013  3174  4255  2196
##  [13]  2013  3200  4714  2537  2153  3582  5180  2727  2336  3958  5405  2751
##  [25]  2579  3320  5065  2933  2641  4047  5618  3259  2777  4013  5549  3460
##  [37]  3337  4265  5961  3776  3323  4537  6306  3851  2836  4297  6012  3980
##  [49]  3345  4897  6188  4103  3611  5222  6700  4333  3840  5269  6891  4794
##  [61]  4217  5952  7747  5622  4719  6680  8130  5631  4940  6446  8168  5961
##  [73]  4804  6834  8028  6080  5046  6800  7912  5636  4993  6733  7943  5538
##  [85]  4862  6279  7098  4593  4558  6509  7619  5706  4926  6085  7716  6147
##  [97]  5451  7015  8412  6748  6171  7868  8858  7073  6351  8476 10296  7592
## [109]  6737  8511  9564  7966  7195  8406  9358  6929  6490  7971  9261  7350
## [121]  6296  7660  8974  7469  6582  8462  9405  7438  6646  8622  9098  7856
## [133]  6554  9113  9874  8027  7047  9837 10305  8148  7536  9989 10512  8755
## [145]  8199 10138 10892  9900  8847 11012 11899  9322  8547 10521 11536  9679
## [157]  8332 10364 11864 10297  6994
```

```
class(all_data)
```

```
## [1] "data.frame"
```

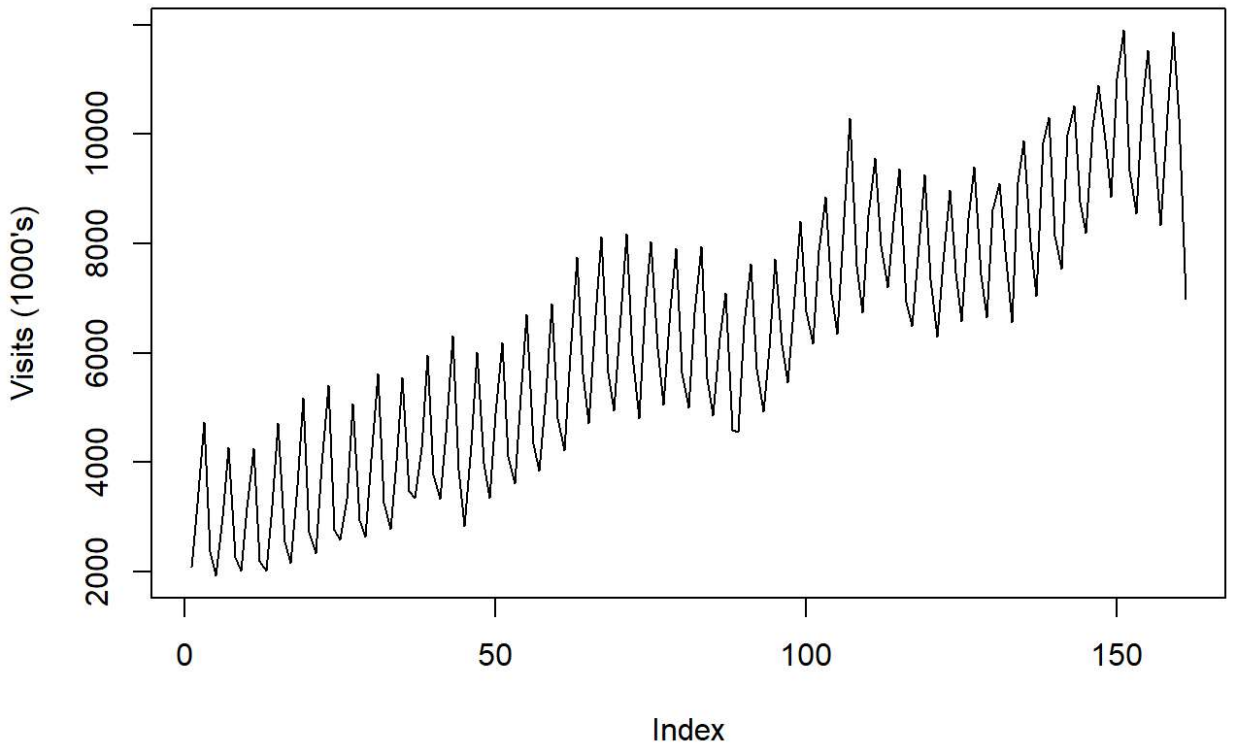
```
class(quaterly_data)
```

```
## [1] "integer"
```

Above output indicate that the Overseas visitors to UK for each quarter from 1980 up to first quarter (Qtr) of 2020.

1.3 Plotting

```
# Plotting quarterly_data vector
quaterly_data %>% plot(type="l", ylab= "Visits (1000's)")
```



1.4 Treatment to outliers and missing values

Missing values have not been examined in the data set. Further, the time plot reveals that there is a massive drop or a outlier at the end of the plot. This is relevant to the data of first quarter (Qtr) of year 2020, which is possibly due to travel restrictions due to Covid 19. Since, the data set capture only a one quarter details relevant to Covid 19 (Qtr1 of 2020), it was removed by considering it as an outlier at this point.

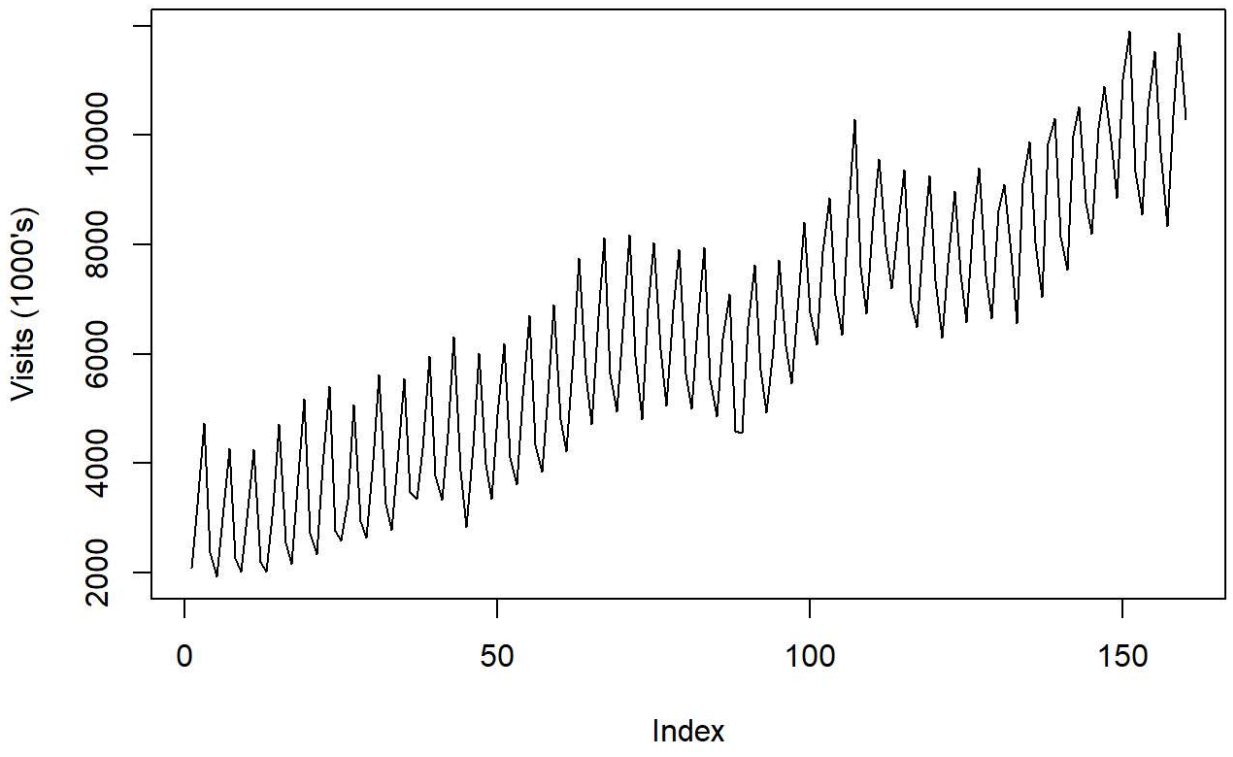
1.5 Transformation

Transformation mechanisms have not been used since the above time series doesn't indicate abnormal variations and pattern. Hence, Boxcox and log transformations were not used.

```
# Elimination of outliers
cleaned_qtr_data <-quaterly_data[c(1:160)]
cleaned_qtr_data
```

```
##   [1]  2081  3240  4738  2360  1920  3008  4261  2262  2013  3174  4255  2196
##  [13]  2013  3200  4714  2537  2153  3582  5180  2727  2336  3958  5405  2751
##  [25]  2579  3320  5065  2933  2641  4047  5618  3259  2777  4013  5549  3460
##  [37]  3337  4265  5961  3776  3323  4537  6306  3851  2836  4297  6012  3980
##  [49]  3345  4897  6188  4103  3611  5222  6700  4333  3840  5269  6891  4794
##  [61]  4217  5952  7747  5622  4719  6680  8130  5631  4940  6446  8168  5961
##  [73]  4804  6834  8028  6080  5046  6800  7912  5636  4993  6733  7943  5538
##  [85]  4862  6279  7098  4593  4558  6509  7619  5706  4926  6085  7716  6147
##  [97]  5451  7015  8412  6748  6171  7868  8858  7073  6351  8476 10296  7592
## [109]  6737  8511  9564  7966  7195  8406  9358  6929  6490  7971  9261  7350
## [121]  6296  7660  8974  7469  6582  8462  9405  7438  6646  8622  9098  7856
## [133]  6554  9113  9874  8027  7047  9837 10305  8148  7536  9989 10512  8755
## [145]  8199 10138 10892  9900  8847 11012 11899  9322  8547 10521 11536  9679
## [157]  8332 10364 11864 10297
```

```
# Plotting
cleaned_qtr_data %>% plot(type="l", ylab= "Visits (1000's)")
```



1.6 Creation of time series object

```
# Time series object
cleaned_qtr_data %>% ts(frequency = 4, start = c(1980,1)) -> cleaned_qtr_data # frequency 4 => Quar
terly Data
cleaned_qtr_data %>% head (20)
```

```
##      Qtr1 Qtr2 Qtr3 Qtr4
## 1980 2081 3240 4738 2360
## 1981 1920 3008 4261 2262
## 1982 2013 3174 4255 2196
## 1983 2013 3200 4714 2537
## 1984 2153 3582 5180 2727
```

```
class(cleaned_qtr_data)
```

```
## [1] "ts"
```

```
#time series object details
class(cleaned_qtr_data)
```

```
## [1] "ts"
```

```
start(cleaned_qtr_data)
```

```
## [1] 1980    1
```

```
end(cleaned_qtr_data)
```

```
## [1] 2019    4
```

1.7 Summary statistics

```
#Summary statistics
summary(cleaned_qtr_data)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1920    4264    6288    6270    8054   11899
```

As per the above output data, the average number of Overseas visits to UK is 6,270 (’Thousands), while minimum is 1,920 (’Thousands) and maximum is 11,899 (’Thousands).

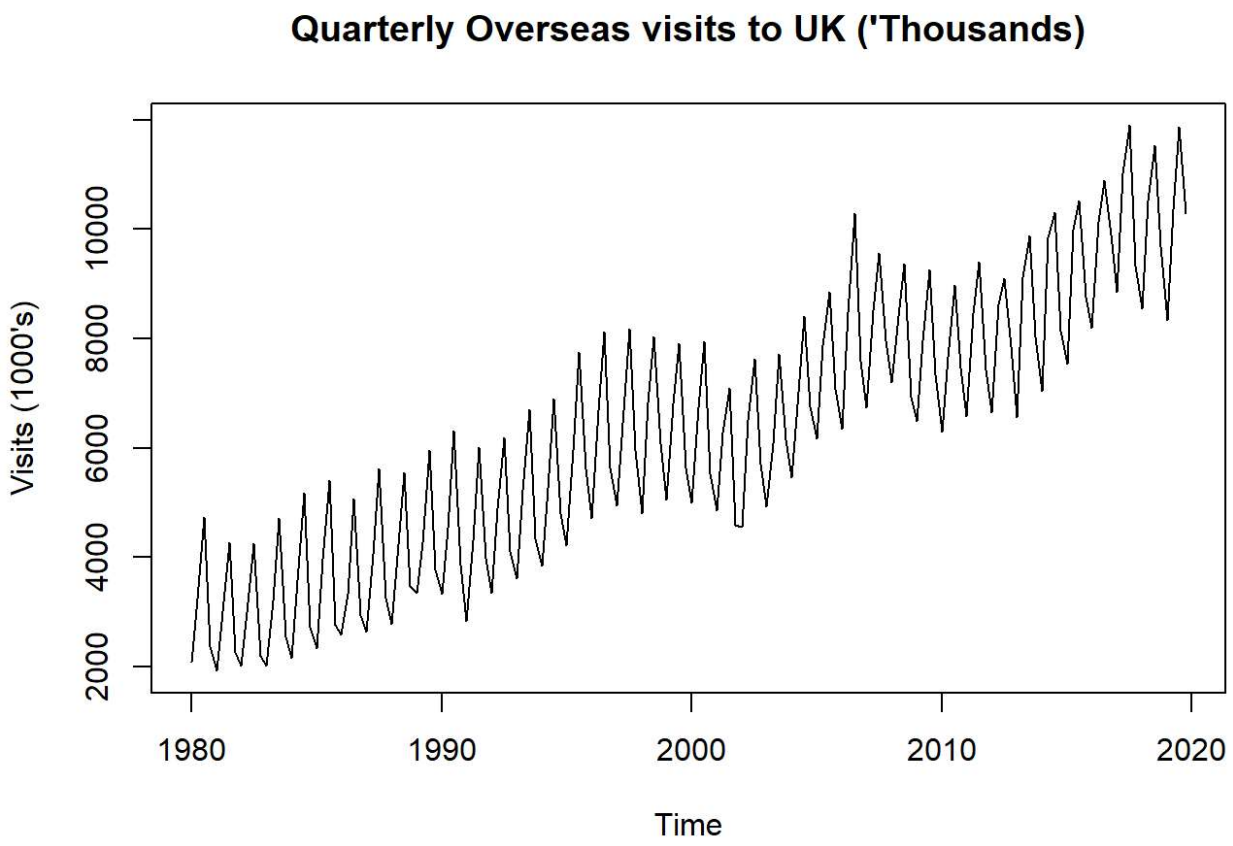
```
#Provides cycle across years
cycle(cleaned_qtr_data)
```

```
##      Qtr1 Qtr2 Qtr3 Qtr4
## 1980    1    2    3    4
## 1981    1    2    3    4
## 1982    1    2    3    4
## 1983    1    2    3    4
## 1984    1    2    3    4
## 1985    1    2    3    4
## 1986    1    2    3    4
## 1987    1    2    3    4
## 1988    1    2    3    4
## 1989    1    2    3    4
## 1990    1    2    3    4
## 1991    1    2    3    4
## 1992    1    2    3    4
## 1993    1    2    3    4
## 1994    1    2    3    4
## 1995    1    2    3    4
## 1996    1    2    3    4
## 1997    1    2    3    4
## 1998    1    2    3    4
## 1999    1    2    3    4
## 2000    1    2    3    4
## 2001    1    2    3    4
## 2002    1    2    3    4
## 2003    1    2    3    4
## 2004    1    2    3    4
## 2005    1    2    3    4
## 2006    1    2    3    4
## 2007    1    2    3    4
## 2008    1    2    3    4
## 2009    1    2    3    4
## 2010    1    2    3    4
## 2011    1    2    3    4
## 2012    1    2    3    4
## 2013    1    2    3    4
## 2014    1    2    3    4
## 2015    1    2    3    4
## 2016    1    2    3    4
## 2017    1    2    3    4
## 2018    1    2    3    4
## 2019    1    2    3    4
```

Original time series can be express as a function that explain the trend, seasonality effect and a stationary component(random component) as white noise. Hence, it was performed an analysis below to identify the trend and seasonality components in the created time series object simply by plotting the original time series data as below.

1.8 Time series object visualisation

```
# Plotting time series object
ts.plot(window(cleaned_qtr_data, start =c(1980,1)),  ylab="Visits (1000's)", main="Quarterly Overse
as visits to UK ('Thousands)")
```

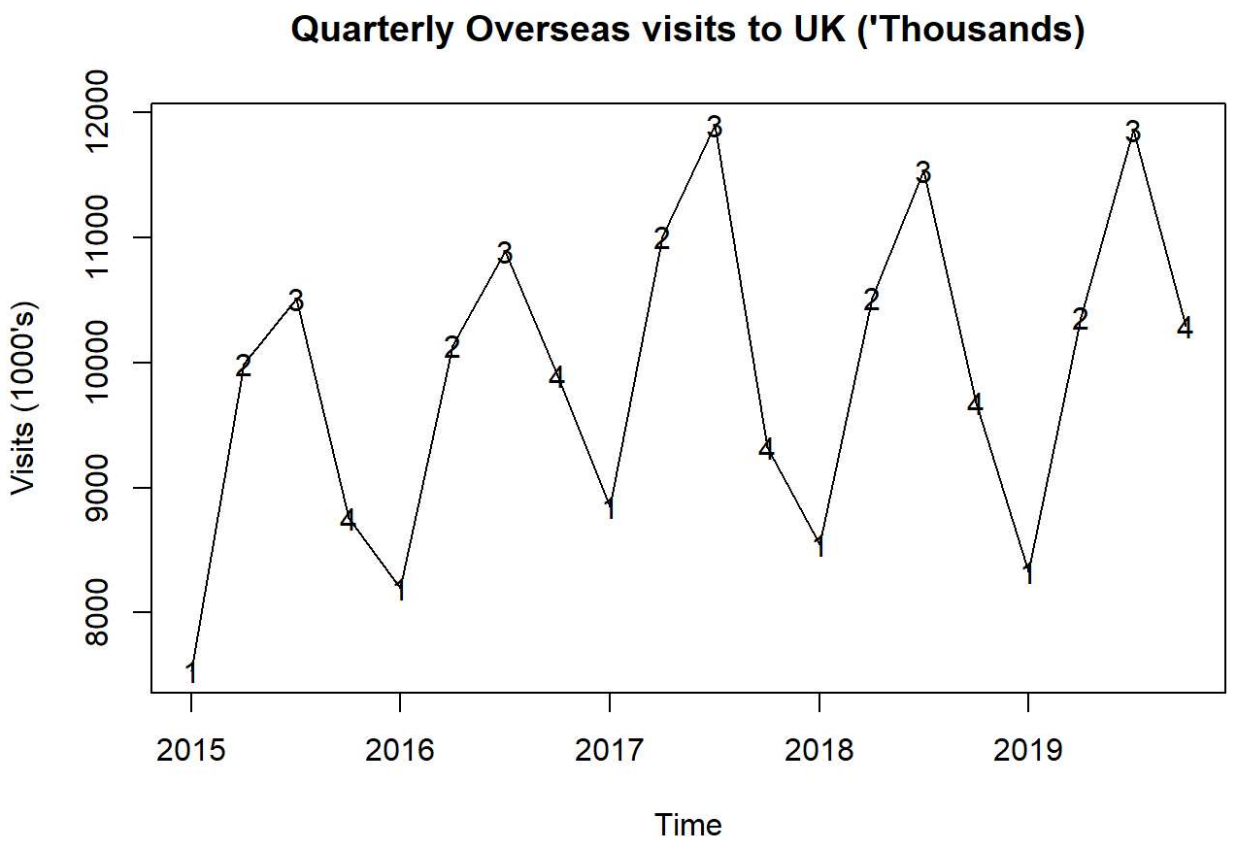


Above Time plot illustrate followings;

There is a increasing trend (upward trend) as the time goes up, the number of visits also growing up in general. The time series displays a very regular pattern called seasonality. Further, seasonality component indicate a strong seasonal pattern between the quarters.This has been analyse in detailed at the plots relevant to seasonality effect.

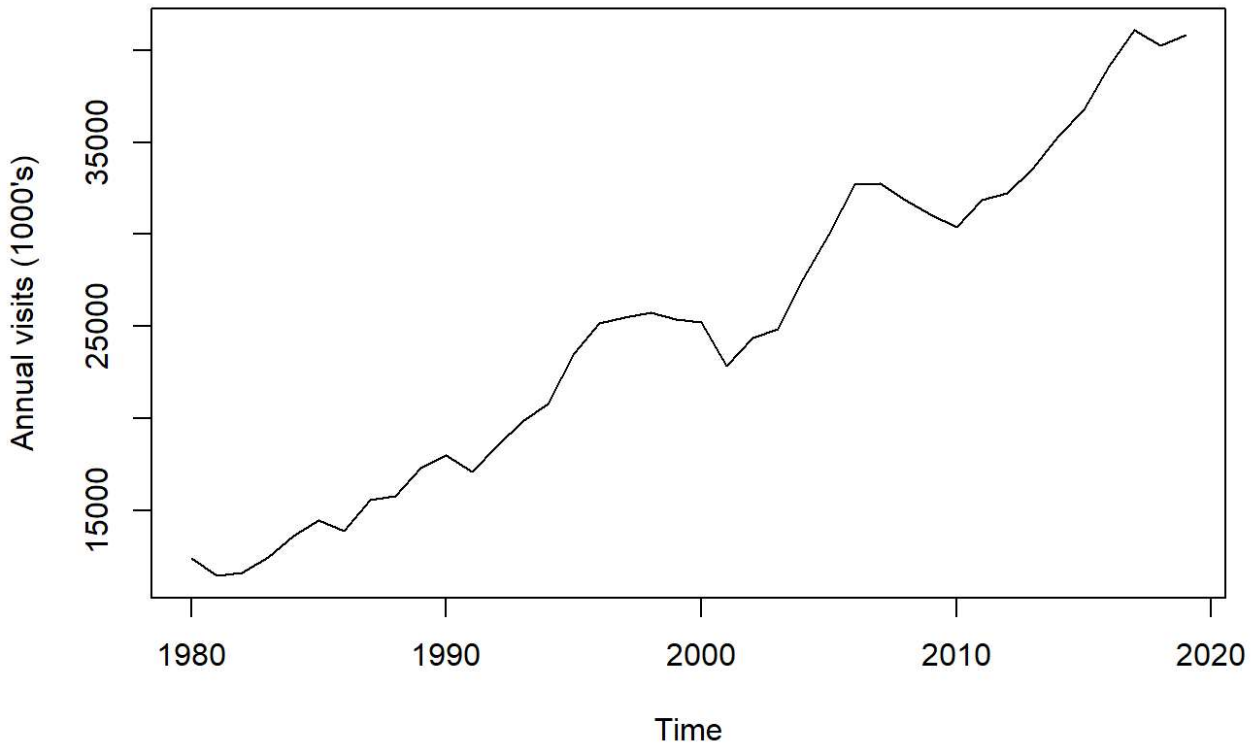
The variance (size of the visits) is not increasing as the time (level of the series) increases. Hence, Box-Cox transformation has not been applied for this time series. All the peaks and drops are not in same size, as there is some viability. As such, it can assume that the Time series variance is not constant over the time.

```
# Plotting a portion of time series object for a recent time slot.
ts.plot(window(cleaned_qtr_data, start =c(2015,1)), ylab="Visits (1000's)", main="Quarterly Oversea
s visits to UK ('Thousands)")
qtr=c("1","2","3","4")
points(window(cleaned_qtr_data,start=c(2015,1)),pch=qtr)
```



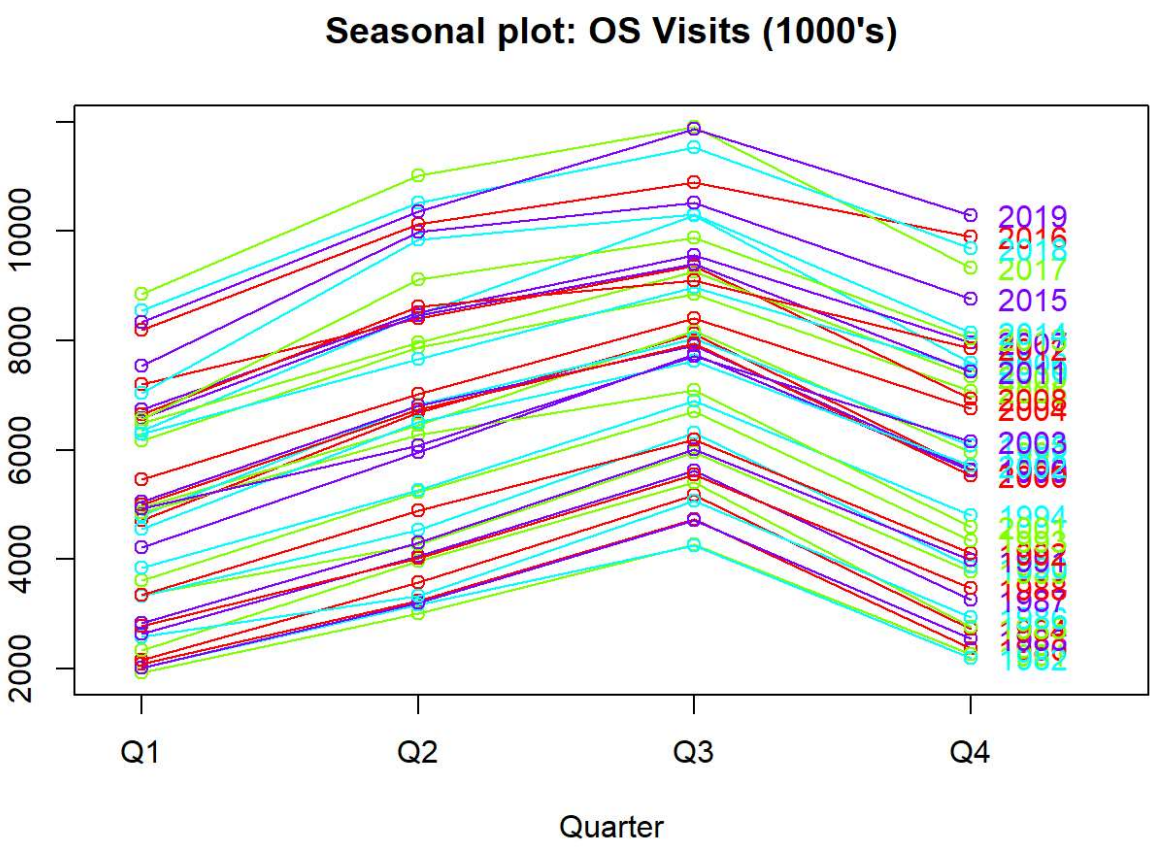
Time series object has been plotted for a portion of recent time slots to visibly see the pattern of the time series more clearly between quarters. Above time plot indicates a regularly repeating pattern of highs and lows related to quarters of the year. In this case, it could clearly seen a drop in first quarter (1), then a gradual increase upto third quarter (3). The graph also shows that there is a drop again in fourth quarter (4).As such, it provides a sense of seasonality component associated with this time series.

```
# Plotting for aggregated data
plot(aggregate(cleaned_qtr_data), ylab="Annual visits (1000's)")
```



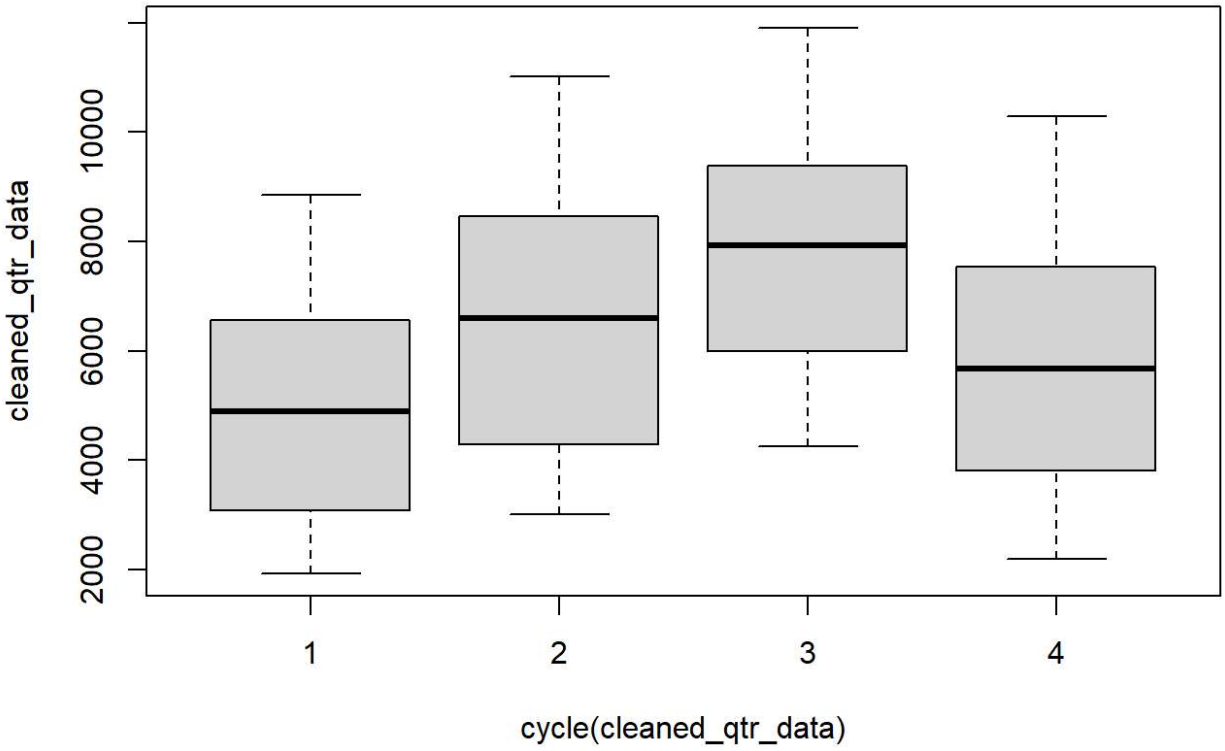
The aggregate (annual) data was plotted to see the trend pattern of annual data and which indicates a upward trend (slighly closer to a deterministic trend) as the time goes up the visits also growing up.

```
#Plot for sense of seasonality
seasonplot(cleaned_qtr_data, col=rainbow(4), year.labels=TRUE, main="Seasonal plot: OS Visits (100
0's)")
```



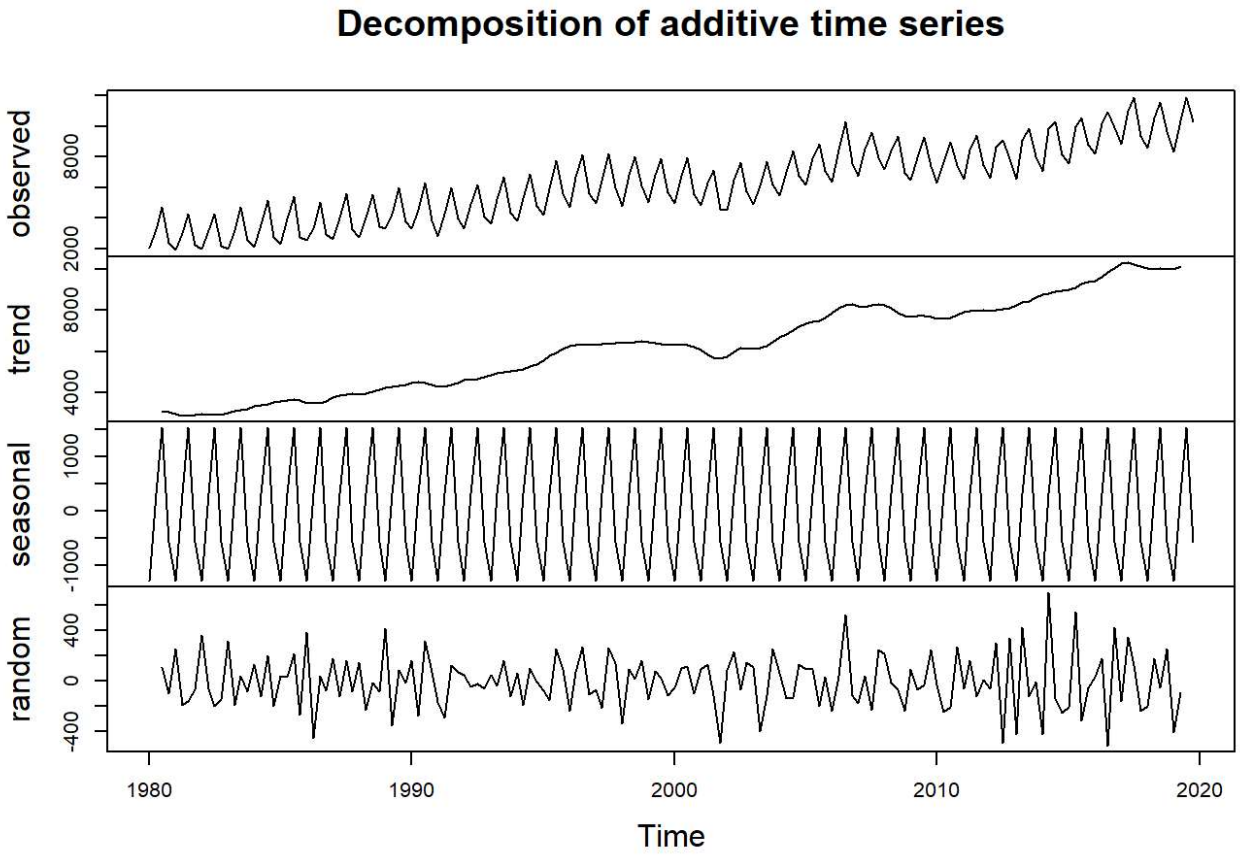
Above seasonal plot indicate the underlying seasonal pattern more clearly. The data from each season are overlapped for four quarters. There is a drop in first quarter (Qtr), then a gradual increase upto third Qtr and again a drop in fourth Qtr. The seasonal pattern between quarters as an increase in summer and then decrease in winter for the number of visits.

```
#Boxplot for cycles
boxplot(cleaned_qtr_data ~cycle(cleaned_qtr_data))
```



The Boxplot is a form of plot enables the underlying seasonal pattern to be seen clearly, and also shows the changes in seasonality over time. The above Boxplot indicates the positions in the cycle of four frequencies. There is a pattern when observing the median line of the each Boxplots which at the beginning of the year tend to decrease. Then at the middle of the year which correspond to summer time there is a increase and again move down at the end of the year which is winter. Highest median value and the range is for the third frequency (Qtr 3) where as the lowest is for the first frequency (Qtr 1).

```
# Decompossition using additive components
Qtr_decom <- decompose(cleaned_qtr_data, type="additive")
plot(Qtr_decom)
```



Additive model was used as the time series variance (size of the visits) is not increasing as the time (level of the series) increases by (variance volatility is not increasing with the time).Decomposition plot illustrate the trend, seasonal and random component of the time series separately in a graphical manner as above. The underlying time series is a non-stationary time series as it has a trend or a seasonal component and requires differencing to transform it to a stationary time series.

Task 2 – Model fitting and Forecasting

Stationary time series is devoid of trend,seasonal patterns and white noise. A time series is said to be stationary if it holds the following conditions true;

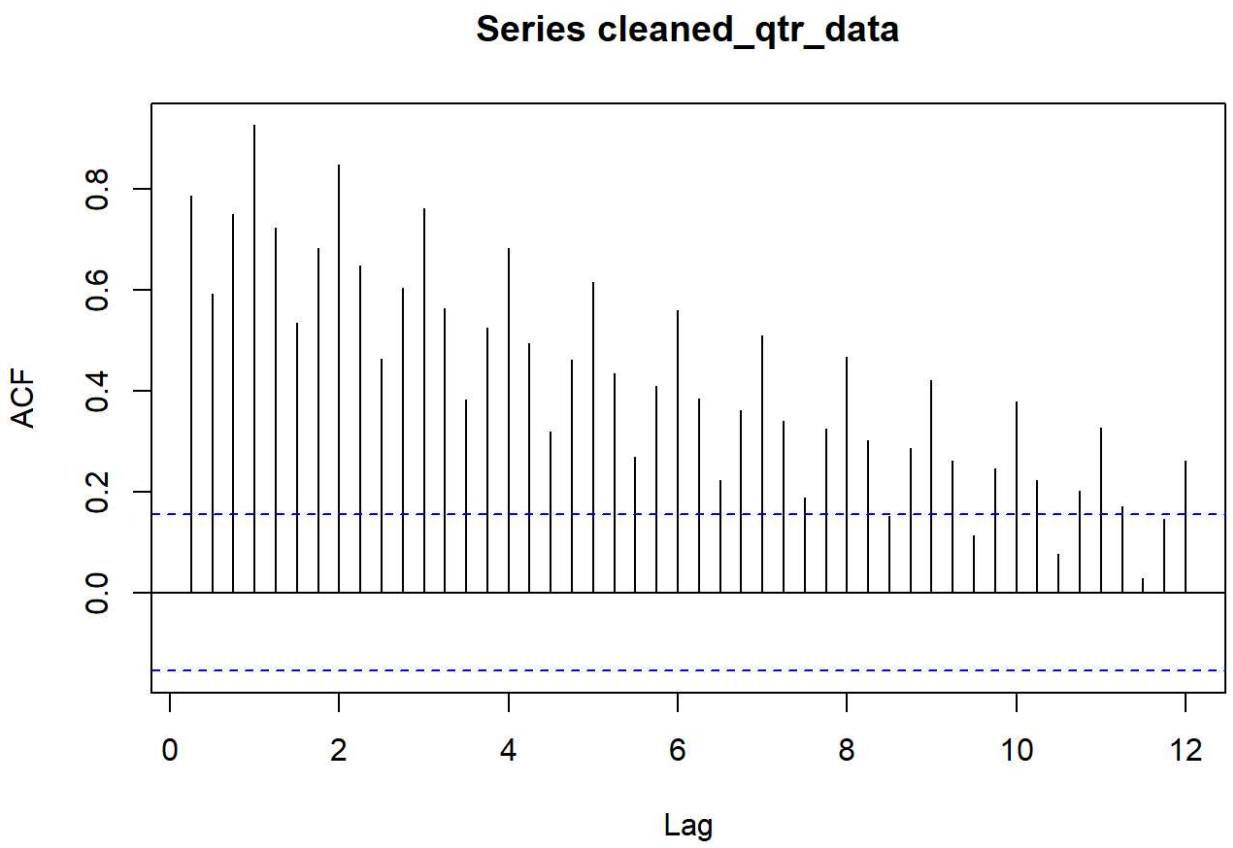
The the expected value (mean value) of time-series is constant over time (which implies, the trend component is nullified) variance is constant which oscillate around a constant value, and the correlation depends on the time lag, but not the absolute.

It is important to note that the process defined in this way is weakly stationary. ACF plot and hypothesis testing (ADF & KPSS) was performed to identify that the time series of Overseas visits to UK is stationary or not.

1.9.1 Checking for stationary - Autocorrelation function (ACF)

The autocorrelation coefficients are plotted to show the autocorrelation function or ACF. ACF plots observe if the data has a trend and a seasonal component. The plot is also known as a correlogram which is in below.

```
#Autocorrelation function for original time series
acf(cleaned_qtr_data, lag.max=48)
```



Above plot of ACF indicates significant positive correlations. The autocorrelations decreases slowly (irrespective of seasonal pattern) as the number of lags increases. Further, there is a clear seasonal pattern is visible as there are some peaks at every four lags. The autocorrelations are gradually decreasing after the every peak as the lags increases. A combination of these effects could be seen, when data are both trended and seasonal.

Hence, ACF indicates trended time series with a seasonality effect. In a stationary time series, the ACF will drop to zero relatively quickly as it doesn't depends on time, while the ACF of non-stationary data decreases slowly as in the above plot. Further, more than 5% of spikes are outside the bounds. As such, the above ACF indicates a that the attributes of non-stationary time series or not a white noise.

Further, it is also useful to quantify the evidence of non-stationarity via hypothesis testing. Hence, the Dickey -Fuller test and KPSS tests has been used in below.

1.9.2 Checking for stationary - Augmented Dickey-Fuller unit-root test (ADF)

```
#Dickey-Fuller test
adf.test(cleaned_qtr_data)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  cleaned_qtr_data
## Dickey-Fuller = -2.9089, Lag order = 5, p-value = 0.1973
## alternative hypothesis: stationary
```

H0: Time series is not a stationary, H1: Time series is a stationary

The p-value of above ADF test is 0.1973, which is not significant at the 5% level. Hence, the null hypothesis should not be rejected, since that there is enough evidence to suggest that there the time series is not stationary.

1.9.3 Checking for stationary - Kwiatkowski-Phillips-SchmidtShin (KPSS) test

```
#KPSS test
kpss.test(cleaned_qtr_data)
```

```
## Warning in kpss.test(cleaned_qtr_data): p-value smaller than printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  cleaned_qtr_data
## KPSS Level = 3.1489, Truncation lag parameter = 4, p-value = 0.01
```

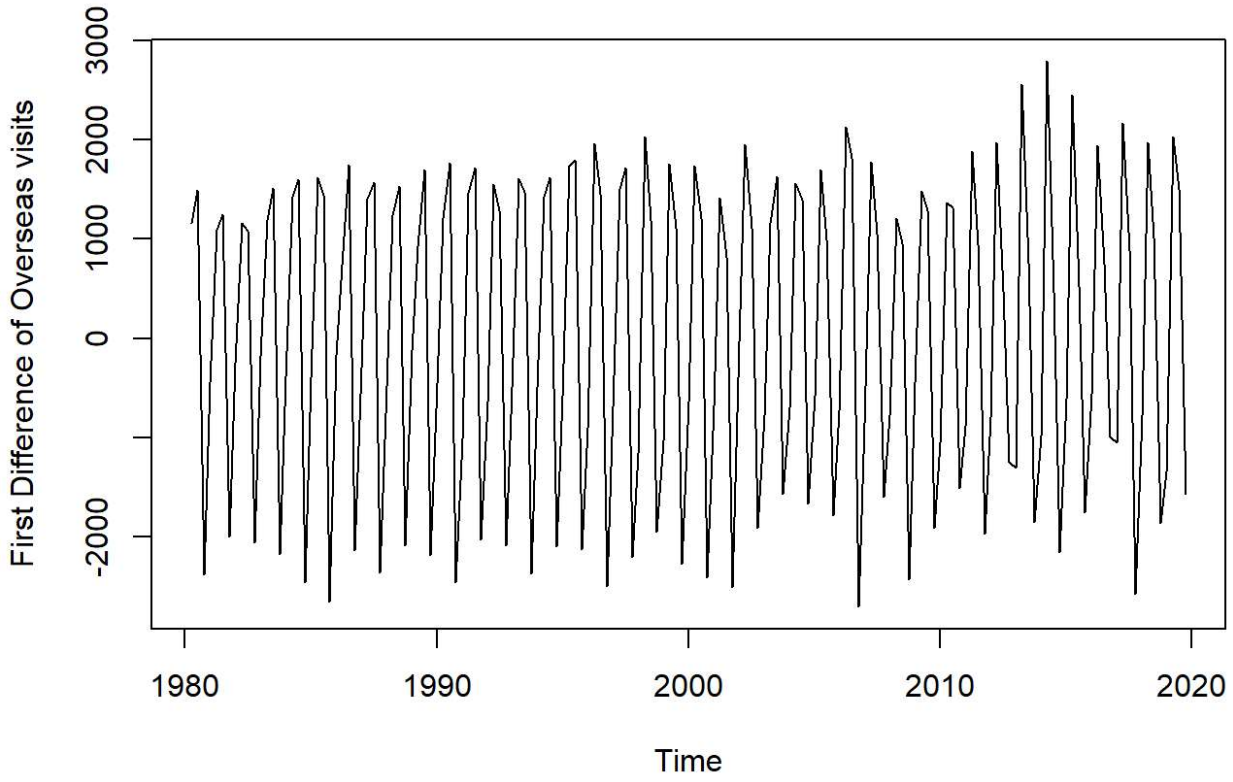
H0: Time series is a stationary, H1: Time series is not a stationary

The p-value of the KPSS test is 0.01, which is significant at the 5% level. Hence, the null hypothesis should be rejected, since there is enough evidence in favor of alternative hypothesis (H1) and conclude that the time series is not a stationary.

It is apparent that the above time series is a non-stationary since it has a trend and seasonal component.Hence, it is required to convert to a stationary time series to construct a appropriate model for forecasting.Decompose the series into the components trend, seasonal effect, and residuals, and plot the decomposed series in below.

1.10 Stationarity Through Differencing

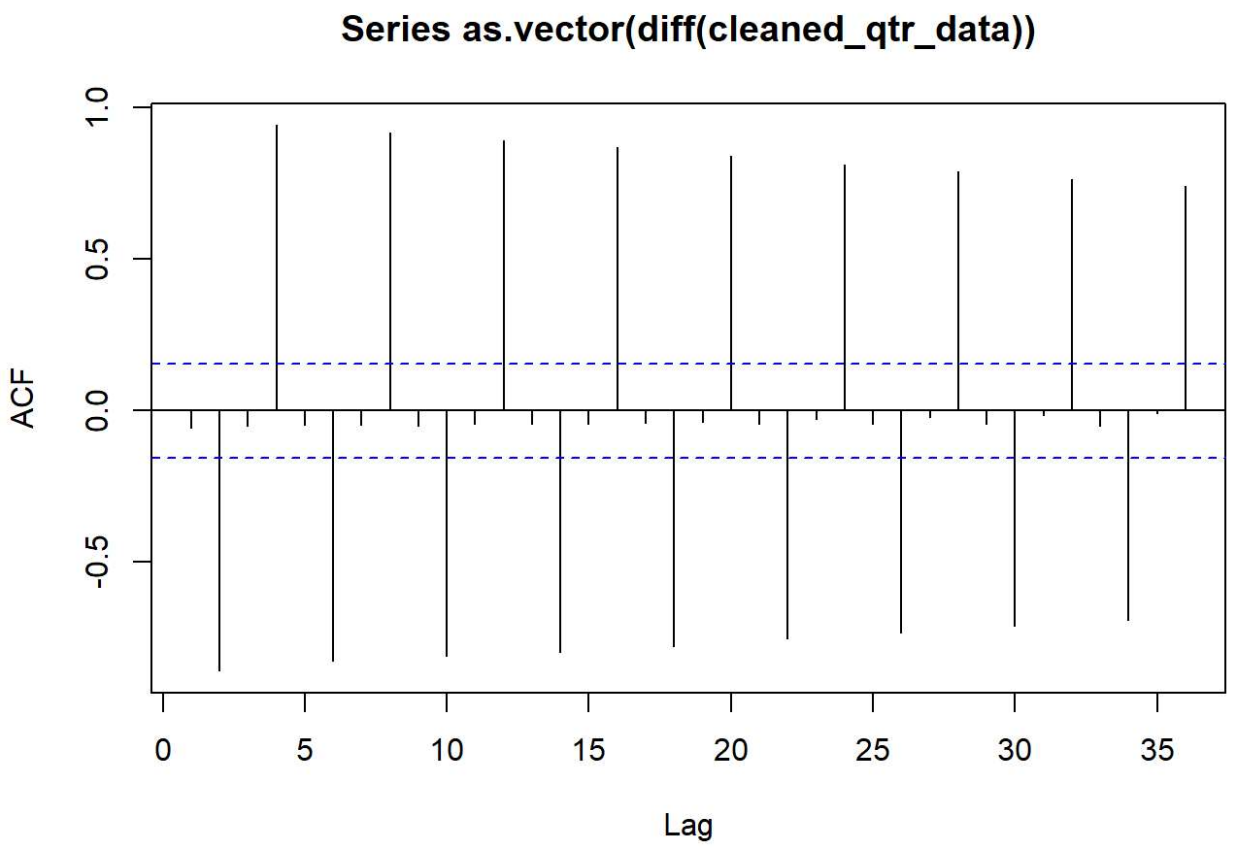
```
#First difference
plot(diff(cleaned_qtr_data),ylab="First Difference of Overseas visits",xlab="Time")
```



A time plot after the first difference is illustrate above. In this case, it is clear that the trend component has been eliminated from the series and values oscillate around a constant level. However, there is still regular pattern is visible which repeat every year. That is an indication of the seasonal pattern which require seasonal differencing to convert the time series to a white noise.

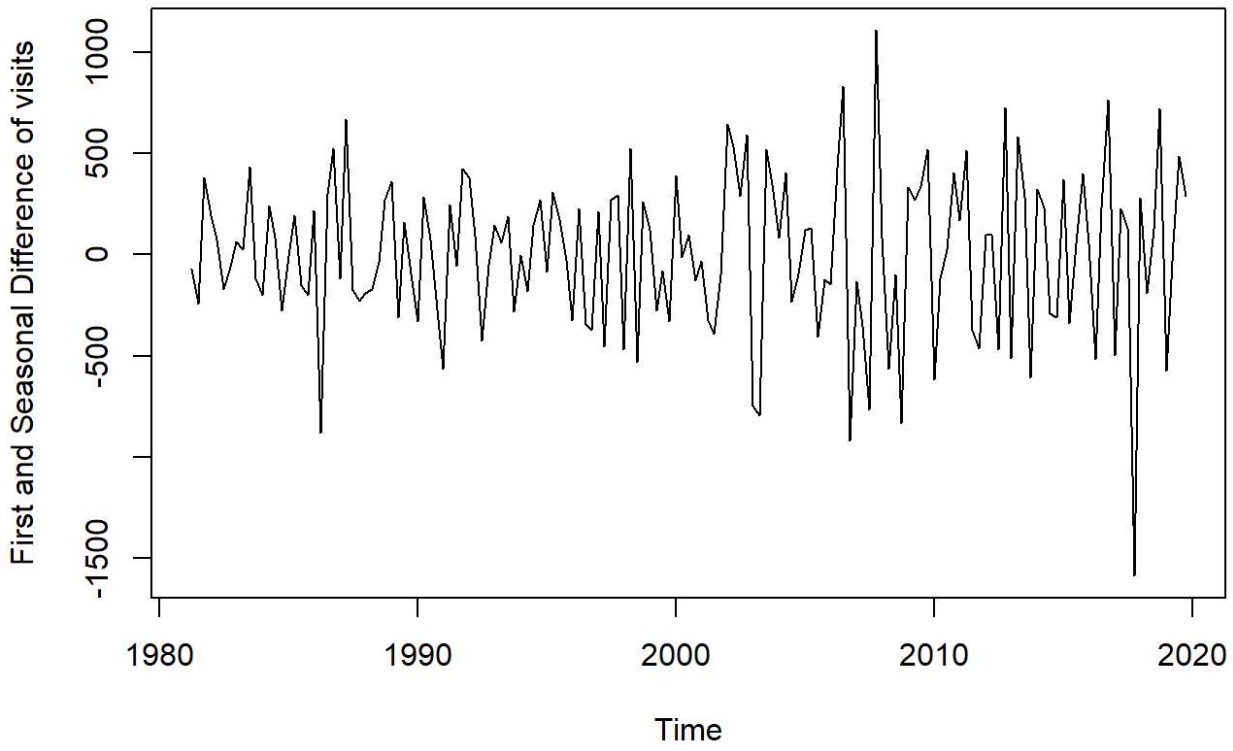
-

```
#Autocorrelation function for the first Difference
acf(as.vector(diff(cleaned_qtr_data)),lag.max=36)
```



Above ACF plot indicates autocorrelation function for the first Difference. There are many significant spikes with a pattern. Hence, second difference is applied to the time series to remove the seasonal effect.

```
# First and Seasonal Difference
plot(diff(diff(cleaned_qtr_data),lag=4),xlab="Time", ylab="First and Seasonal Difference of visits"
)
```



The above figure displays the time plot after application of both first difference and seasonal difference.As such, there is no trend component, since the values are oscillate around a constant level. Further, the seasonal pattern has been eliminated and a random pattern is visible. Accordingly, both the trend and seasonal component has been eliminated from the time series. Further, the evidence of non-stationarity could be quantify via hypothesis testing. Thus, Dickey - Fuller test has been applied below.

Checking for stationary - Augmented Dickey-Fuller unit-root test (ADF)

```
#Dickey-Fuller test
adf.test(diff(diff(cleaned_qtr_data),lag=4))
```

```
## Warning in adf.test(diff(diff(cleaned_qtr_data), lag = 4)): p-value smaller than
## printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(diff(cleaned_qtr_data), lag = 4)
## Dickey-Fuller = -6.7978, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

H0: Time series is not a stationary, H1: Time series is a stationary

The p-value of above ADF test is 0.01, which is significant at the 5% level. Hence, the null hypothesis should be rejected, since there is enough evidence in favor of alternative hypothesis (H1) and conclude that the time series is a stationary.

```
#KPSS test
kpss.test(diff(diff(cleaned_qtr_data),lag=4))
```

```
## Warning in kpss.test(diff(diff(cleaned_qtr_data), lag = 4)): p-value greater
## than printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  diff(diff(cleaned_qtr_data), lag = 4)
## KPSS Level = 0.021788, Truncation lag parameter = 4, p-value = 0.1
```

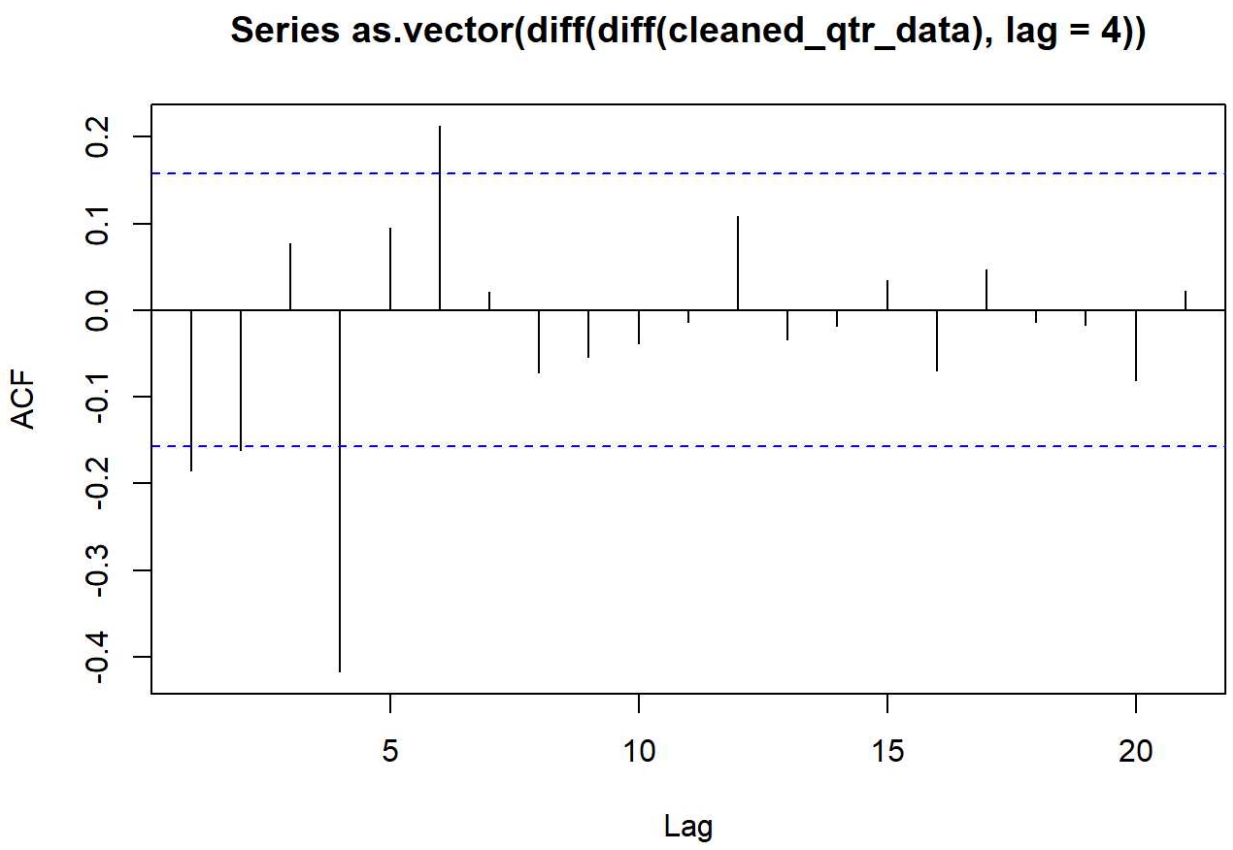
H0: Time series is a stationary, H1: Time series is not a stationary

The p-value of above KPSS test is 0.1, which is not significant at the 5% level. Hence, the null hypothesis should not be rejected, since that there is enough evidence to suggest that there the time series is a stationary.

1.11 Model Specification

The following ACF, PACF and EACF plots have been analysed to determine the parameters initially.

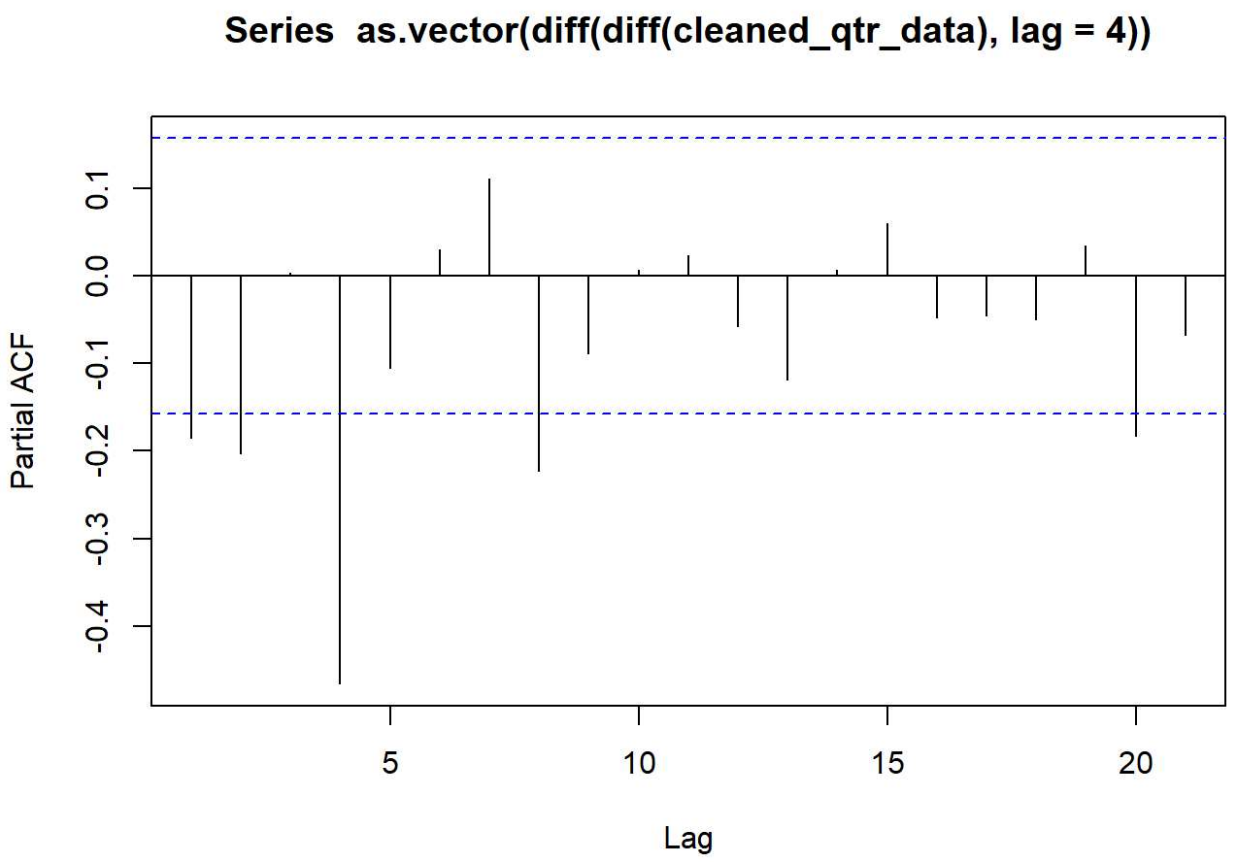
```
#Autocorrelation function stationary time series
acf(as.vector(diff(diff(cleaned_qtr_data),lag=4)))
```



Above ACF function illustrate that approximately 95% of the spikes are within the interval of confidence. However, there are very few significant autocorrelations are out from the interval of confidence. The spikes at lag 1,4, and 6 are significantly different from zero and rest of all lags are zero or closer to zero. Hence, it could assume that the time series is a is a stationary.

Above ACF plot has been used to determine moving average (MA) component of the model specification. As such, ACF illustrate both a non-seasonal moving average component (q) and seasonal moving average component (Q). Hence, it could assume 'q' as 1 or 2, and 'Q' as 1. The first difference (d) was set as 1.

```
#Partial Autocorrelation function stationary time series
pacf(as.vector(diff(diff(cleaned_qtr_data),lag=4)))
```



PACF function indicates that very few significant autocorrelations are out from the interval of confidence. However, it can assume that the 95% is within the interval of confidence. The spikes at lag 1,2,4,8,and 20 are significantly different from zero and rest of all lags are zero or closer to zero. Hence, it could assume that the time series is a is a stationary.

PACF plot illustrate autoregressive (AR) component of the of model specification. The time series in this analysis consist with both non-seasonal autoregressive component (p) and seasonal autoregressive component (Q). Significant Peaks tend to be four lags apart. Hence, by examining the above PACF plot,it could assume 'p' as 1 or 2, and 'P' as 1 or 2. (Lag 20 has not considered as not all the 20 parameters are significant and it will lead to increase the errors in the model). The seasonal difference (D) was set as 1.

```
#eacf for stationary time series
eacf(as.vector(diff(diff(cleaned_qtr_data))),ar.max=10, ma.max=10)
```

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

By examining the EACF plot to determine a appropriate model is complicated as it does not have a clear visible pattern.

1.12 Parameter Estimation

Seasonal ARIMA(p,d,q)×(P,D,Q)s model would be appropriate to apply for the time series in this analysis where nonseasonal orders (p, d, q),seasonal orders (P, D, and Q) and seasonal period s. Thus, ACF, PACF plots and AIC, BIC criteria has been used in the process of model specification & parameter estimation of the model.

```
# Model development
pq0101 =arima(cleaned_qtr_data,order=c(0,1,1),seasonal=list(order=c(0,1,1), period=4))
pq0101
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(0, 1, 1), seasonal = list(order = c(0,
##     1, 1), period = 4))
##
## Coefficients:
##           ma1     sma1
##       -0.3585  -0.7691
## s.e.   0.0892   0.0617
##
## sigma^2 estimated as 98806:  log likelihood = -1113.13,  aic = 2230.26
```

The out put illustrate two parameters in this model, and coefficients of two parameters (MA1 and SMA1) is -0.3585 and -0.7691 respectively. Square error also quite small for the parameters (0.0892 & 0.0617 respectively) and AIC value as 2,230.26. Accordingly, list of possible models have been evaluated in below to identify the most appropriate model for the time series.

```
#Evaluatuin of List of possible models and Parameter estimation
pq0101 =arima(cleaned_qtr_data,order=c(0,1,1),seasonal=list(order=c(0,1,1), period=4))
pq0101
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(0, 1, 1), seasonal = list(order = c(0,
##     1, 1), period = 4))
##
## Coefficients:
##           ma1     sma1
##       -0.3585  -0.7691
## s.e.   0.0892   0.0617
##
## sigma^2 estimated as 98806:  log likelihood = -1113.13,  aic = 2230.26
```

```
pq1010=arima(cleaned_qtr_data,order=c(1,1,0),seasonal=list(order=c(1,1,0), period=4))
pq1010
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(1, 1, 0), seasonal = list(order = c(1,
##     1, 0), period = 4))
##
## Coefficients:
##           ar1     sar1
##       -0.1737  -0.4172
## s.e.   0.0796   0.0729
##
## sigma^2 estimated as 123099:  log likelihood = -1128.69,  aic = 2261.38
```

```
pq0201=arima(cleaned_qtr_data,order=c(0,1,2),seasonal=list(order=c(0,1,1), period=4))
pq0201
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(0, 1, 2), seasonal = list(order = c(0,
##     1, 1), period = 4))
##
## Coefficients:
##           ma1      ma2     sma1
##       -0.3031  -0.1229  -0.7351
## s.e.   0.0819   0.0780   0.0728
##
## sigma^2 estimated as 97460:  log likelihood = -1111.89,  aic = 2229.78
```

```
pq2221=arima(cleaned_qtr_data,order=c(2,1,2),seasonal=list(order=c(2,1,1), period=4))
pq2221
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(2, 1, 2), seasonal = list(order = c(2,
##     1, 1), period = 4))
##
## Coefficients:
##          ar1      ar2      ma1     ma2    sar1    sar2     sma1
##       0.9196  -0.4334  -1.2266  0.5472  0.1294  0.0261  -0.8060
## s.e.  0.2964   0.2165   0.3018  0.2169  0.1386  0.1079   0.0753
##
## sigma^2 estimated as 95231:  log likelihood = -1110.09,  aic = 2234.19
```

```
pq2111=arima(cleaned_qtr_data,order=c(2,1,1),seasonal=list(order=c(1,1,1), period=4))
pq2111
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(2, 1, 1), seasonal = list(order = c(1,
##     1, 1), period = 4))
##
## Coefficients:
##          ar1      ar2      ma1    sar1     sma1
##       0.2705  -0.0614  -0.5862  0.1660  -0.8086
## s.e.  0.2222   0.1135   0.2141  0.1204   0.0663
##
## sigma^2 estimated as 95955:  log likelihood = -1110.77,  aic = 2231.54
```

```
pq1221=arima(cleaned_qtr_data,order=c(1,1,2),seasonal=list(order=c(2,1,1), period=4))
pq1221
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(1, 1, 2), seasonal = list(order = c(2,
##     1, 1), period = 4))
##
## Coefficients:
##          ar1      ma1      ma2    sar1    sar2     sma1
##       0.1640  -0.4815  -0.0895  0.1808  0.0280  -0.8214
## s.e.  0.4273   0.4293   0.1854  0.1193  0.0966   0.0714
##
## sigma^2 estimated as 95945:  log likelihood = -1110.77,  aic = 2233.54
```

```
pq2211=arima(cleaned_qtr_data,order=c(2,1,2),seasonal=list(order=c(1,1,1), period=4))
pq2211
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(2, 1, 2), seasonal = list(order = c(1,
##     1, 1), period = 4))
##
## Coefficients:
##          ar1      ar2      ma1     ma2    sar1     sma1
##       0.9472  -0.4255  -1.2567  0.5542  0.1268  -0.7979
## s.e.  0.3061   0.1955   0.3128  0.2069  0.1380   0.0701
##
## sigma^2 estimated as 95248:  log likelihood = -1110.12,  aic = 2232.25
```

```
pq1121=arima(cleaned_qtr_data,order=c(1,1,1),seasonal=list(order=c(2,1,1), period=4))
pq1121
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(1, 1, 1), seasonal = list(order = c(2,
##     1, 1), period = 4))
##
## Coefficients:
##          ar1      ma1    sar1    sar2     sma1
##       0.3400  -0.6681  0.1869  0.0237  -0.8271
## s.e.  0.1818   0.1490  0.1190  0.0956   0.0688
##
## sigma^2 estimated as 95990:  log likelihood = -1110.88,  aic = 2231.76
```

```
pq1111=arima(cleaned_qtr_data,order=c(1,1,1),seasonal=list(order=c(1,1,1), period=4))
pq1111
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(1, 1, 1), seasonal = list(order = c(1,
##     1, 1), period = 4))
##
## Coefficients:
##          ar1      ma1    sar1     sma1
##       0.3350  -0.6629  0.1792  -0.8181
## s.e.  0.1805   0.1480  0.1166   0.0618
##
## sigma^2 estimated as 96019:  log likelihood = -1110.91,  aic = 2229.82
```

```
pq0111=arima(cleaned_qtr_data,order=c(0,1,1),seasonal=list(order=c(1,1,1), period=4))
pq0111
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(0, 1, 1), seasonal = list(order = c(1,
##     1, 1), period = 4))
##
## Coefficients:
##           ma1    sar1     sma1
##       -0.3717  0.1215  -0.8188
## s.e.   0.0932  0.1029   0.0609
##
## sigma^2 estimated as 97882:  log likelihood = -1112.45,  aic = 2230.89
```

```
pq1001=arima(cleaned_qtr_data,order=c(1,1,0),seasonal=list(order=c(0,1,1), period=4))
pq1001
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(1, 1, 0), seasonal = list(order = c(0,
##     1, 1), period = 4))
##
## Coefficients:
##           ar1     sma1
##       -0.2394  -0.7625
## s.e.   0.0817   0.0636
##
## sigma^2 estimated as 101847:  log likelihood = -1115.37,  aic = 2234.75
```

```
pq1100=arima(cleaned_qtr_data,order=c(1,1,1),seasonal=list(order=c(0,1,0), period=4))
pq1100
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(1, 1, 1), seasonal = list(order = c(0,
##     1, 0), period = 4))
##
## Coefficients:
##          ar1      ma1
##       0.5630  -1.0000
## s.e.  0.0676   0.0208
##
## sigma^2 estimated as 120940:  log likelihood = -1128.82,  aic = 2261.64
```

```
pq1102=arima(cleaned_qtr_data,order=c(1,0,1),seasonal=list(order=c(0,1,2), period=4))
pq1102
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(1, 0, 1), seasonal = list(order = c(0,
##     1, 2), period = 4))
##
## Coefficients:
##          ar1      ma1     sma1     sma2
##       0.9742  -0.3437  -0.6807  -0.0861
## s.e.  0.0243   0.1025   0.0876   0.0835
##
## sigma^2 estimated as 96842:  log likelihood = -1118.51,  aic = 2245.02
```

```
pq2012=arima(cleaned_qtr_data,order=c(2,0,0),seasonal=list(order=c(0,1,2), period=4))
pq2012
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(2, 0, 0), seasonal = list(order = c(0,
##     1, 2), period = 4))
##
## Coefficients:
##          ar1     ar2     sma1     sma2
##       0.7396  0.2004  -0.6591  -0.0769
## s.e.  0.0789  0.0872   0.0933   0.0851
##
## sigma^2 estimated as 99131:  log likelihood = -1120.25,  aic = 2248.5
```

More than 10 possible models were tested. As per the those output results,the lowest AIC value which is 2,229.78 produce by “pq0201” model and then the second lowest AIC value of 2,229.82 for “pq1111”. Hence, it can assume ‘pq0202’ as the best model based on the AIC value and the second best as ‘pq111’. The attributes of the model are describe below.

```
#AIC lowest models in details

pq0201
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(0, 1, 2), seasonal = list(order = c(0,
##     1, 1), period = 4))
##
## Coefficients:
##           ma1      ma2     sma1
##       -0.3031  -0.1229  -0.7351
## s.e.   0.0819   0.0780   0.0728
##
## sigma^2 estimated as 97460:  log likelihood = -1111.89,  aic = 2229.78
```

```
pq1111
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(1, 1, 1), seasonal = list(order = c(1,
##     1, 1), period = 4))
##
## Coefficients:
##          ar1      ma1    sar1     sma1
##       0.3350  -0.6629  0.1792  -0.8181
## s.e.  0.1805   0.1480  0.1166   0.0618
##
## sigma^2 estimated as 96019:  log likelihood = -1110.91,  aic = 2229.82
```

Model attributes comparrison :

The lowest AIC value of 2,229.78 is given by ‘pp0201’ model which has 3 parameters and the coefficients of parameters (ma1, ma2 and sma1) is -0.3031, -0.1229 and and -0.7351 respectively. Square error also quite small for all the parameters (0.0819, 0.0780 and 0.0728 respectively).

The second lowest AIC value of 2,229.82 is given by ‘pp1111’ which has 4 parameters and the coefficients of parameters (ar1, ma1, sar1 and sma1) is 0.3350, -0.6629, 0.1792 and -0.8181 respectively. However, square error also quite small for all parameters but slightly above to the previous model.

Further, the number of parameters are lower in the model ‘pp0201’ and lesser the parameters is better. By comparing two models, it could assume that the pp0201 is the best model in terms of AIC value and number of parameters.

As such, two models (pp0201 and pp1111) has been specified as below.

Best model out of the evaluated list of models : Seasonal ARIMA(0,1,2)×(0,1,1)4 Second best model out of the evaluated list of models : Seasonal ARIMA(1,1,1)×(1,1,1)4

```
#Model evaluation-BIC criteria
BIC(pq0201)
```

```
## [1] 2243.953
```

```
BIC(pq1111)
```

```
## [1] 2247.037
```

```
BIC(pq1010)
```

```
## [1] 2272.511
```

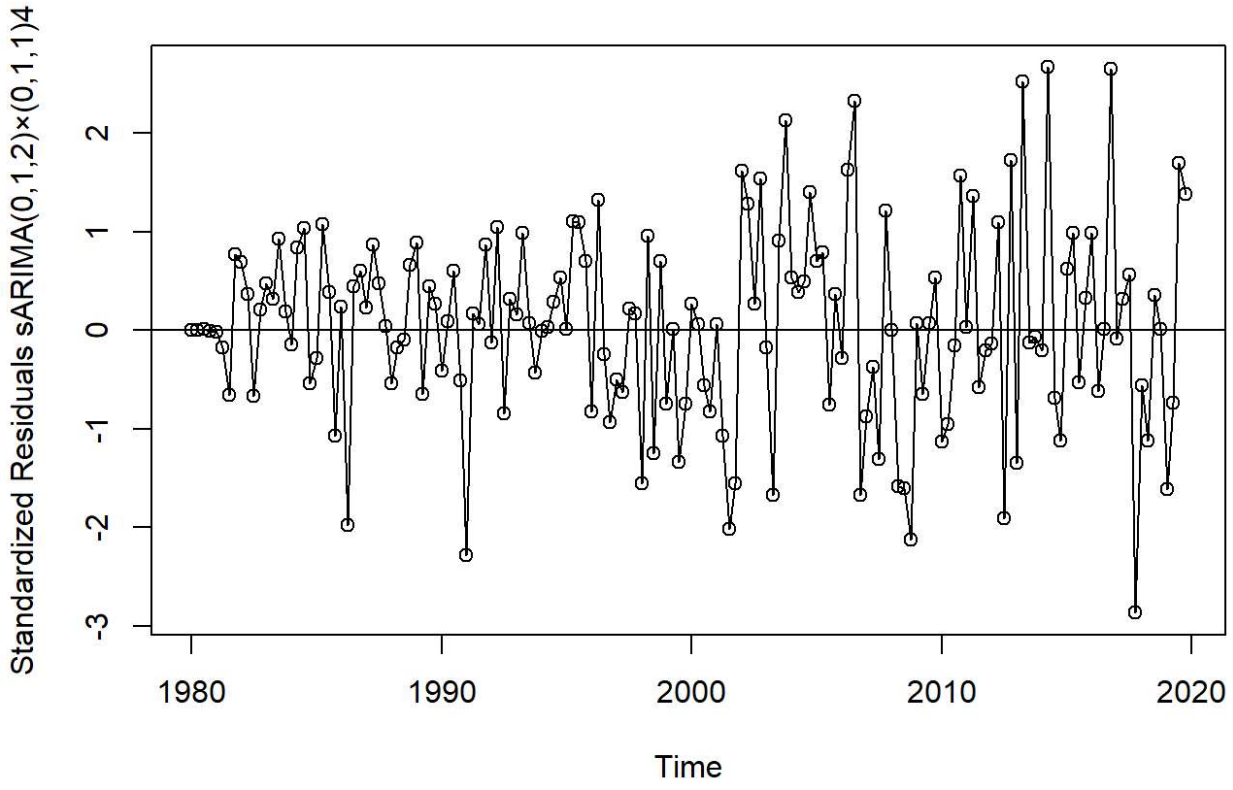
Further, BIC criteria has used to evaluate the models which has the lowest AIC values. As such, BIC criteria provides lowest value for “pq0201” which is 2,243.953. Hence, both tests suggest the “pq0201” appears to be an more appropriate model for the time series that is seasonal ARIMA(0,1,2)×(0,1,1)4

1.13 Residual Analysis (Estimate of White noise)

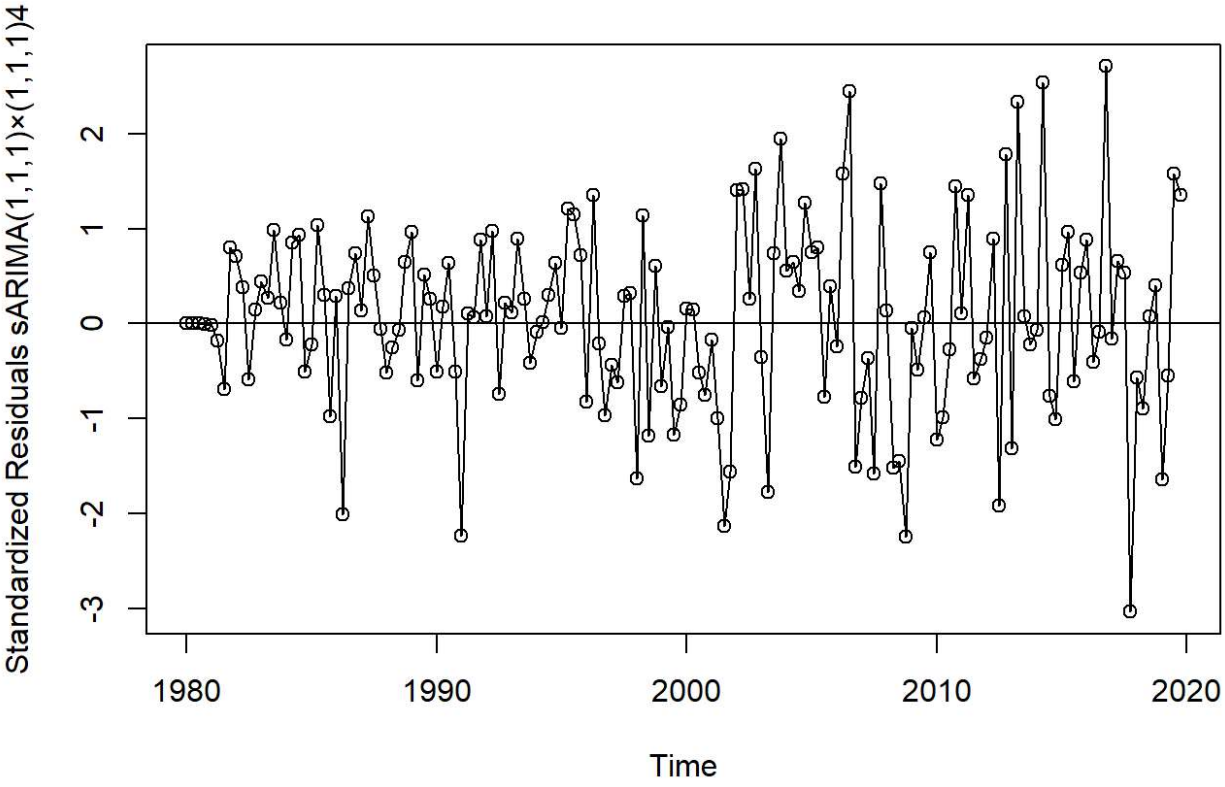
Residuals should be a white noise which is a weaker condition of stationary. Accordingly analysis is performed for residuals as below for the above selected two models; Seasonal ARIMA(0,1,2)×(0,1,1)4 and Seasonal ARIMA(1,1,1)×(1,1,1)4

1.13.1 Plots of the Residuals

```
#Reseduals plots for 2 selected models
plot(rstandard(pq0201),ylab ="Standardized Residuals sARIMA(0,1,2)×(0,1,1)4", type="o"); abline(h=0
)
```




```
plot(rstandard(pq1111),ylab ="Standardized Residuals sARIMA(1,1,1)×(1,1,1)4", type="o"); abline(h=0
)
```

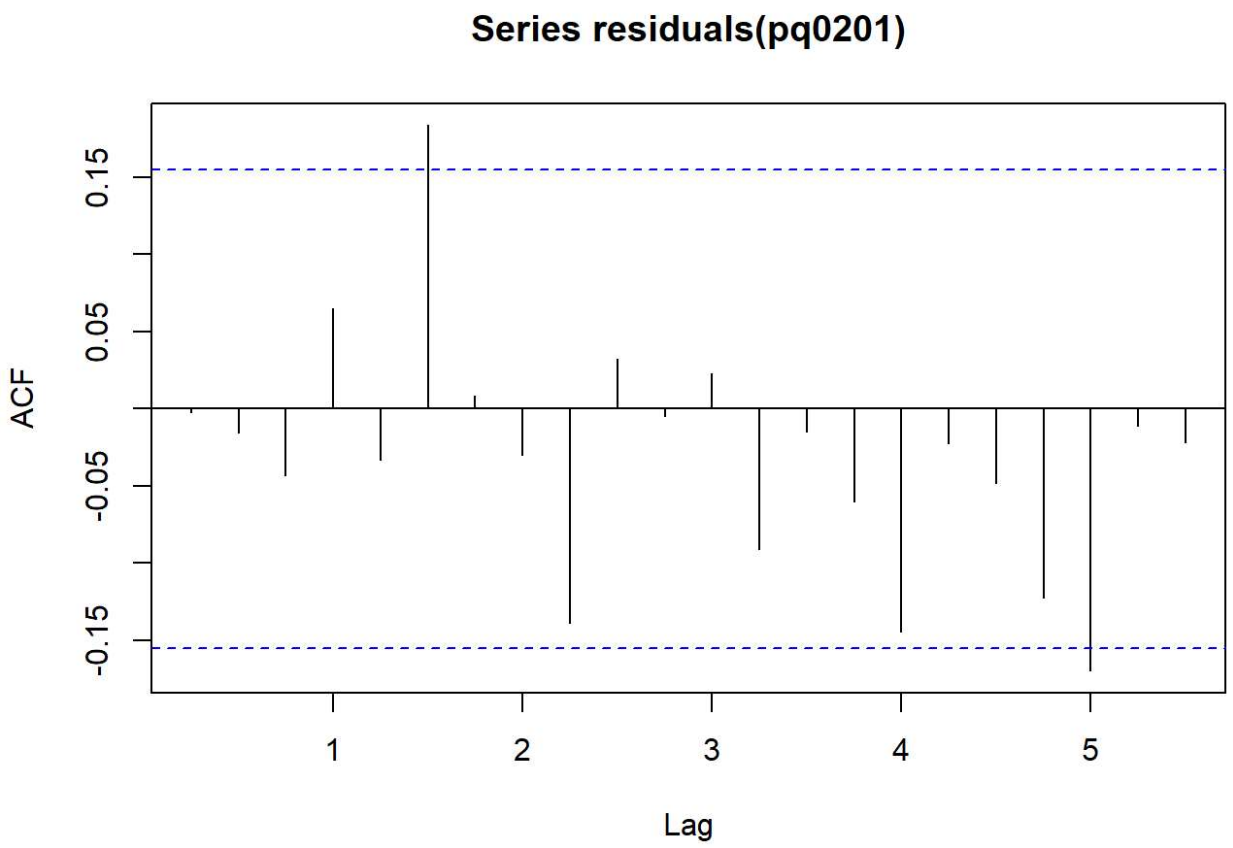


The above residual plots illustrate that the majority of residuals are oscillate around zero and approximately 95% of data fall between +2 & -2.

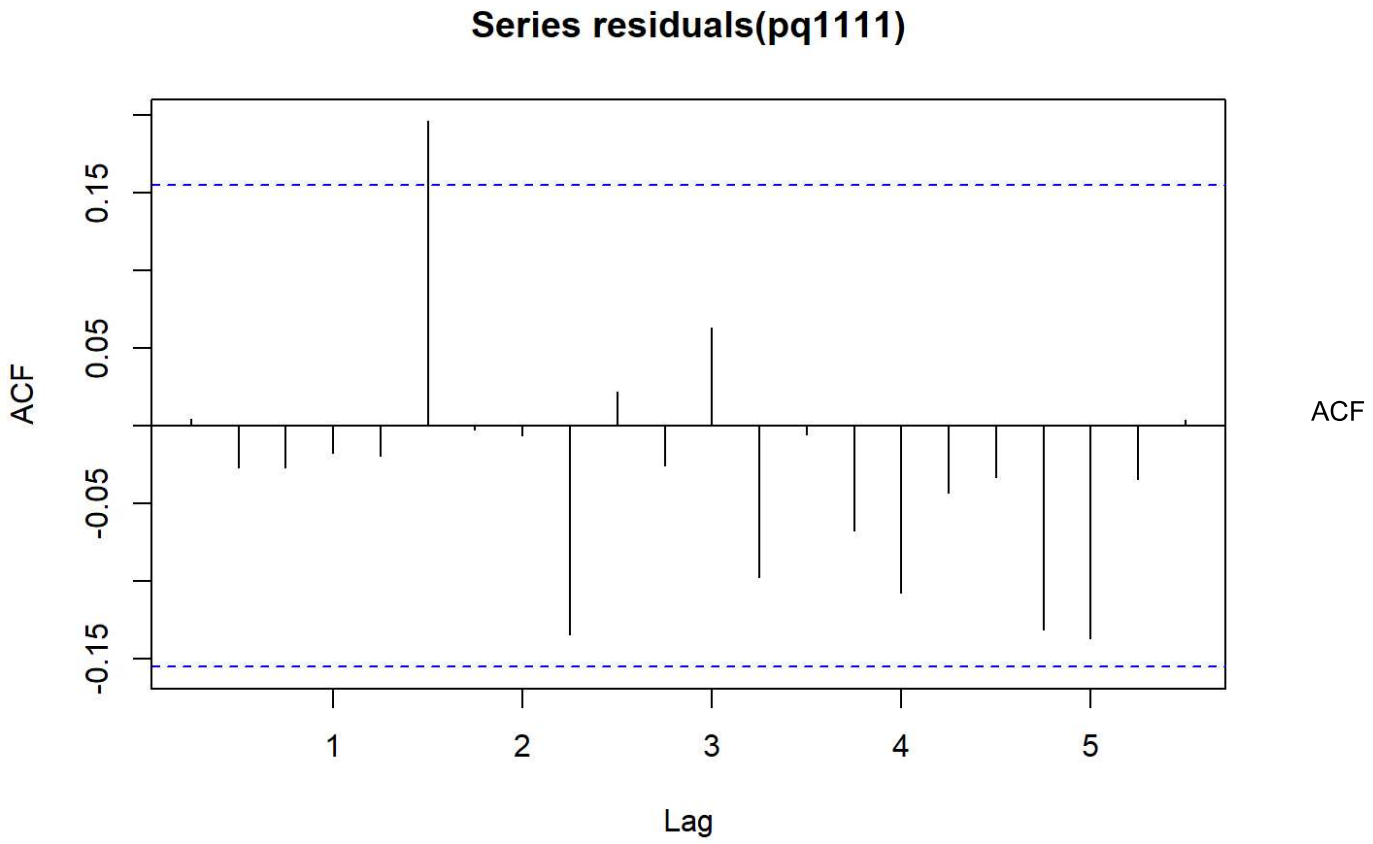
1.13.2 ACF of Residuals

As residuals should be a stationary which is independent and identically distributed. Hence, ACF of residuals should be close to zero and 95% of lags should be inside the interval of confidence.

```
acf(residuals(pq0201)) # Seasonal ARIMA(0,1,2)×(0,1,1)4
```




```
acf(residuals(pq1111)) # Seasonal ARIMA(1,1,1)×(1,1,1)4
```



function for residuals indicates that only one or two autocorrelation are out from the interval of confidence. Hence, that the 95% is within the interval of confidence and are approximately zero or closer to zero. As such, it appears to be no significant autocorrelation in the residuals and the residuals are uncorrelated.

1.13.3 Statistical tests for Residuals - Box-Pierce test and the Ljung-Box test.

(H0 : the data are sample from an iid sequence)

```
# seasonal ARIMA(0,1,2)×(0,1,1)4
Box.test(residuals(pq0201)) #Box-Pierce test
```

```
##
##  Box-Pierce test
##
## data:  residuals(pq0201)
## X-squared = 0.0011166, df = 1, p-value = 0.9733
```

```
Box.test(residuals(pq0201), lag= 4, type=c("Ljung-Box")) #Ljung-Box test
```

```
##
##  Box-Ljung test
##
## data:  residuals(pq0201)
## X-squared = 1.0429, df = 4, p-value = 0.9032
```

```
#seasonal ARIMA(1,1,1)×(1,1,1)4
Box.test(residuals(pq1111)) #Box-Pierce test
```

```
##
##  Box-Pierce test
##
## data:  residuals(pq1111)
## X-squared = 0.0033836, df = 1, p-value = 0.9536
```

```
Box.test(residuals(pq1111),lag= 4, type=c("Ljung-Box")) #Ljung-Box test
```

```
##
##  Box-Ljung test
##
## data:  residuals(pq1111)
## X-squared = 0.29204, df = 4, p-value = 0.9903
```

H0 : Residuals or error terms are uncorrelated

The p-value of above tests are not significant at the 5% level. Hence, the null hypothesis should not be rejected, since that there is enough evidence to suggest for Residuals or error terms are uncorrelated.

1.13.4 Stationary of Residuals - ADF test

```
#Dickey-Fuller test
Res_pq0201 <- residuals(pq0201) # seasonal ARIMA(0,1,2)×(0,1,1)4
adf.test(Res_pq0201)
```

```
## Warning in adf.test(Res_pq0201): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  Res_pq0201
## Dickey-Fuller = -4.0971, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

```
Res_pq1111 <- residuals(pq1111) #seasonal ARIMA(1,1,1)×(1,1,1)4
adf.test(Res_pq1111)
```

```
## Warning in adf.test(Res_pq1111): p-value smaller than printed p-value
```

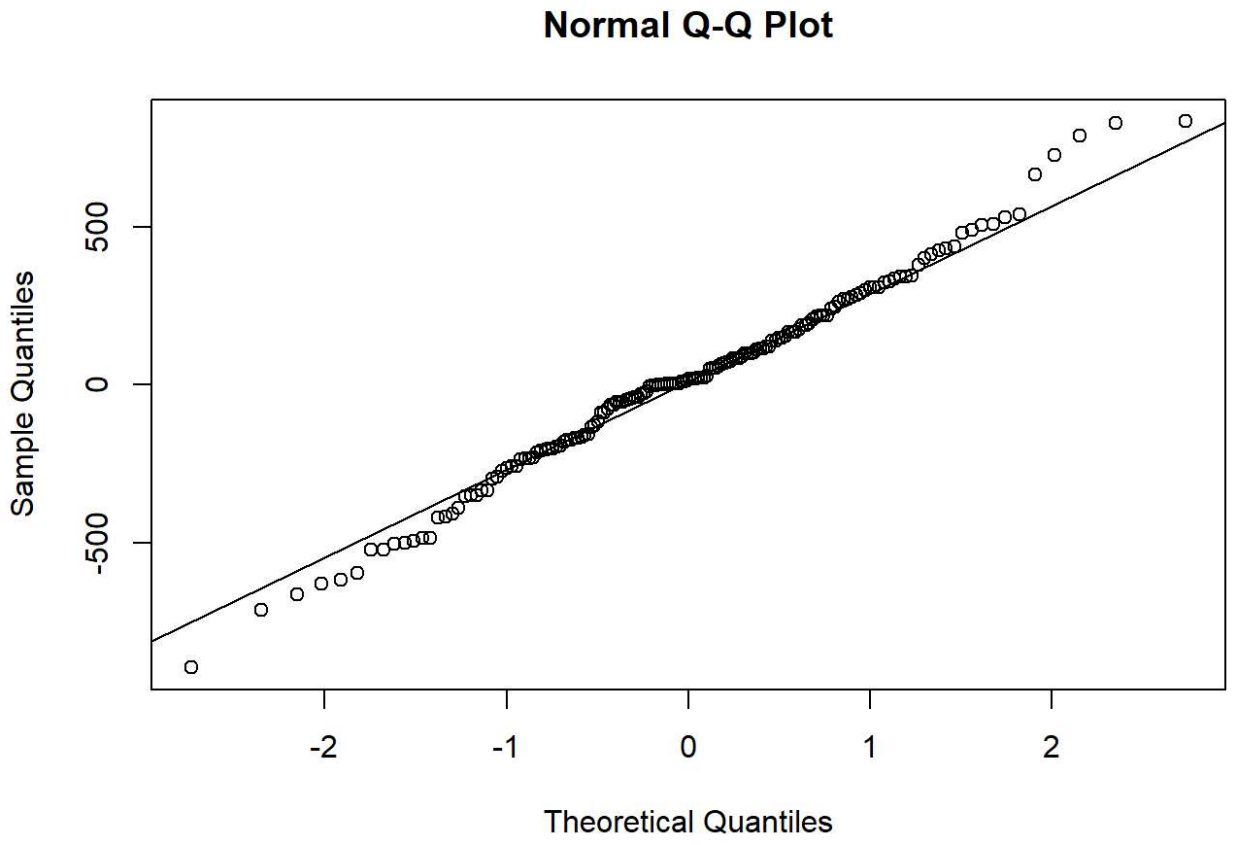
```
##
##  Augmented Dickey-Fuller Test
##
## data:  Res_pq1111
## Dickey-Fuller = -4.2119, Lag order = 5, p-value = 0.01
## alternative hypothesis: stationary
```

H0: Time series is not a stationary, H1: Time series is a stationary

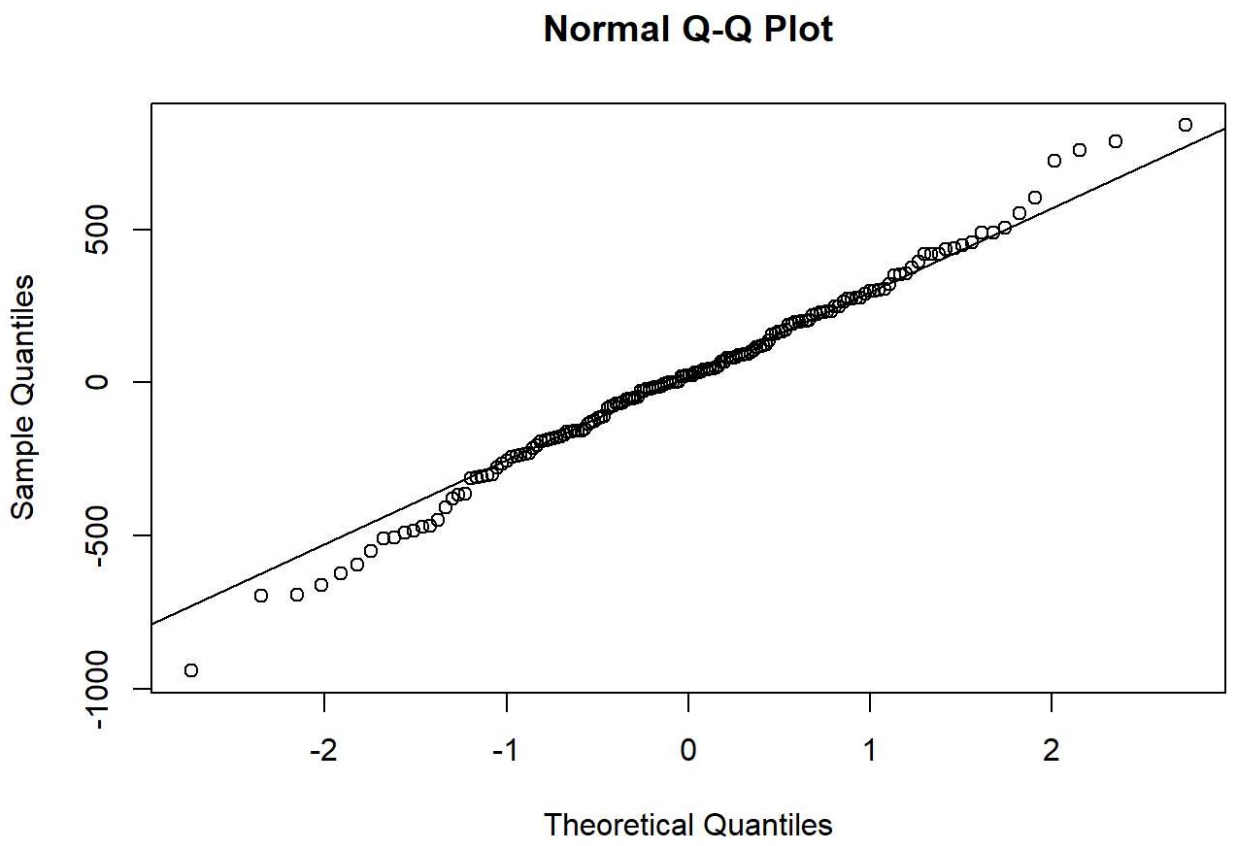
The p-value of both ADF test are below 0.01, which is significant at the 5% level. Hence, the null hypothesis should be rejected, since there is enough evidence in favor of alternative hypothesis (H1) and conclude that the time series is a stationary. This confirms the residuals are stationary or white noise.

1.13.5 Normality of the Residuals - QQ plots

```
qqnorm(residuals(pq0201)); qqline(residuals(pq0201)) # seasonal ARIMA(0,1,2)×(0,1,1)4
```




```
qqnorm(residuals(pq1111)); qqline(residuals(pq1111)) #seasonal ARIMA(1,1,1)×(1,1,1)4
```



Above plots illustrate that the quantiles of the residuals is close to the normal distribution in general.However, few deviations are noticable at the edge of the line. Hence, A statistical test has been performed to chack the normality of the distribution.

1.13.6 Statistical tests - Normality of the Residuals

```
shapiro.test(rstandard(pq0201))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  rstandard(pq0201)
## W = 0.99108, p-value = 0.417
```

```
shapiro.test(rstandard(pq1111))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  rstandard(pq1111)
## W = 0.99216, p-value = 0.5334
```

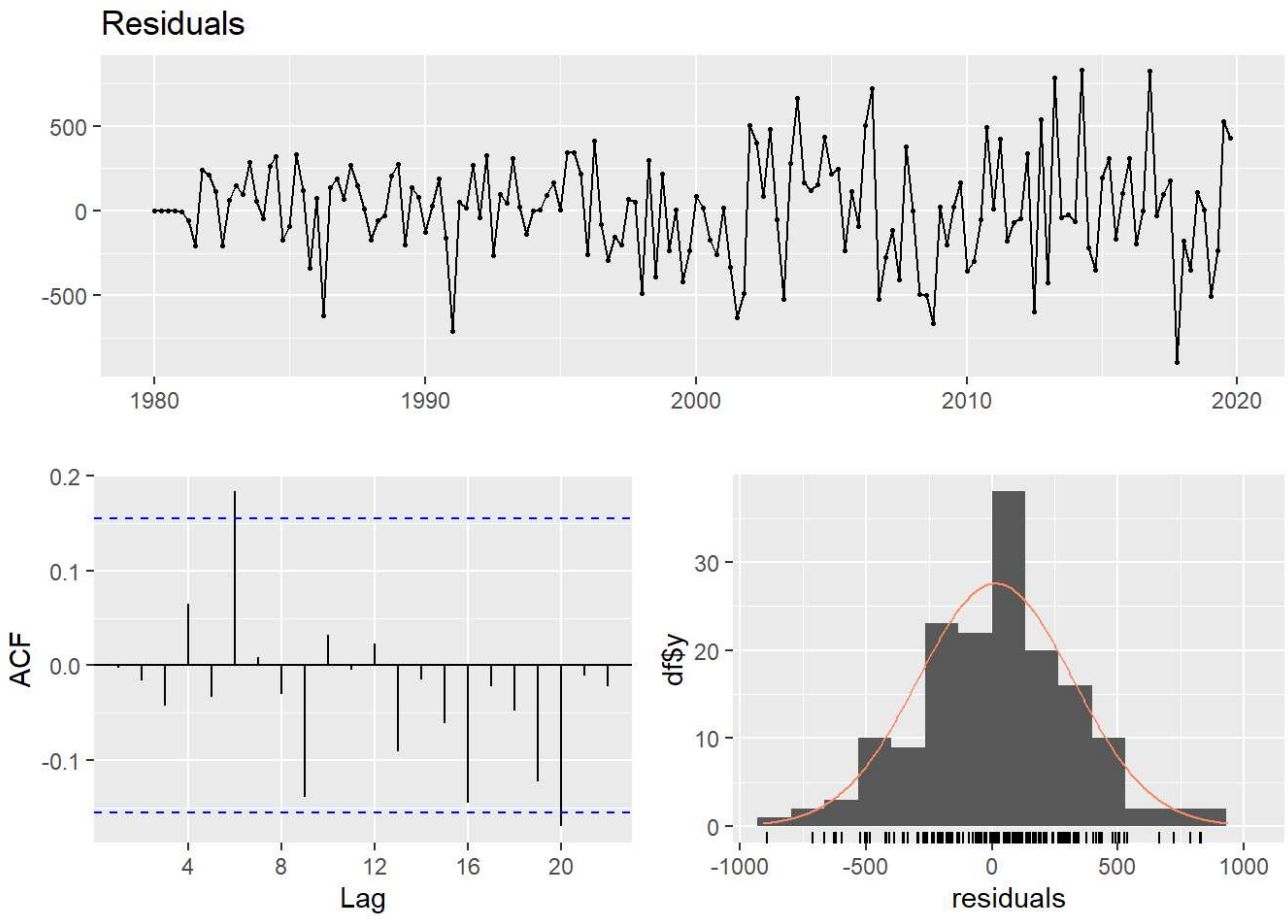
H0: Data is normally distributed

Shapiro-Wilk test of normality has a test statistic p-value is 0.417 and HO should not be rejected. Hence, the residuals are normally distributed.

1.13.7 Summary of residuals analysis

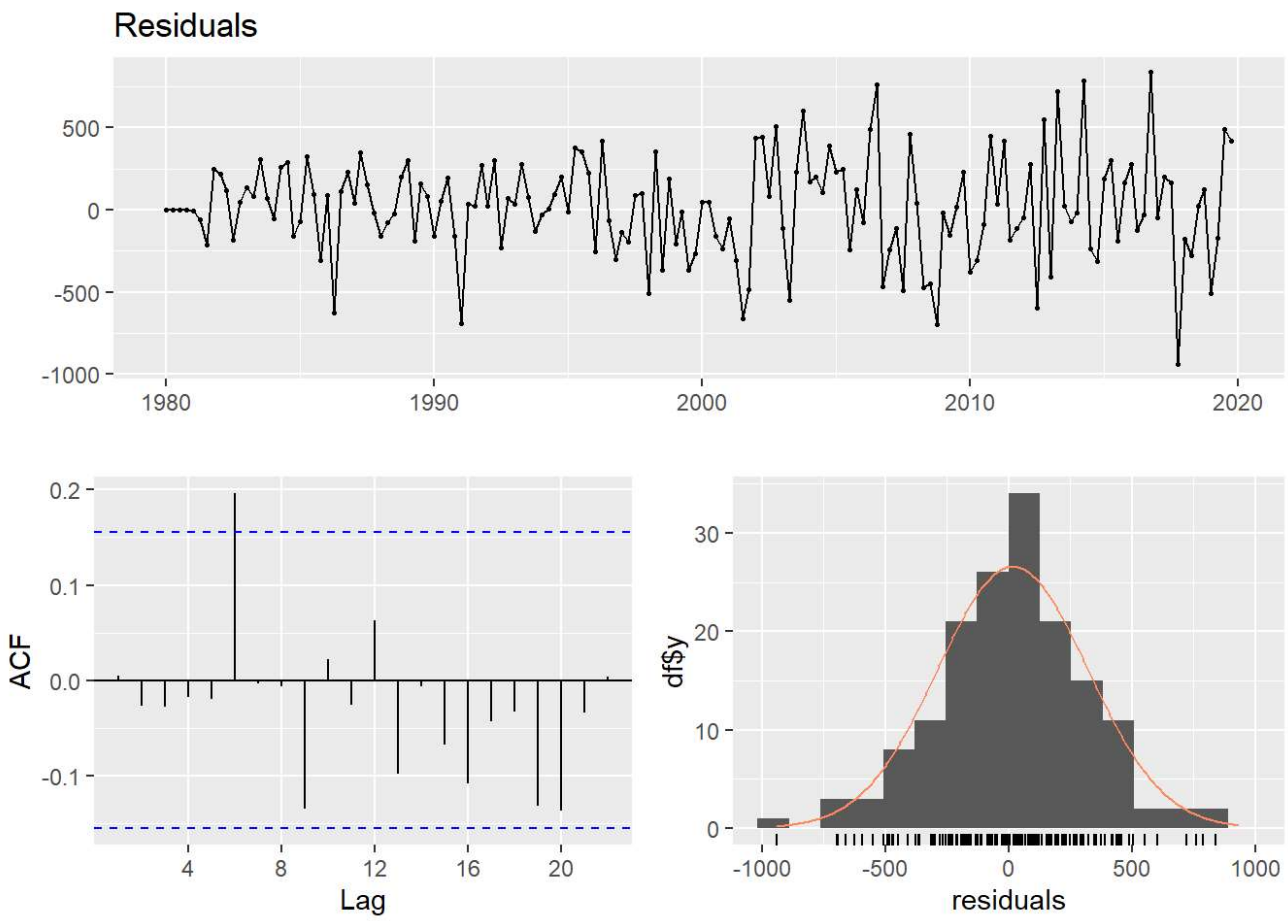
```
checkresiduals(Res_pq0201) # seasonal ARIMA(0,1,2)×(0,1,1)4
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```




```
checkresiduals(Res_pq1111) #seasonal ARIMA(1,1,1)×(1,1,1)4
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```



Residuals analysis illustrate that the residuals are stationary/white noise (weaker conditions), uncorrelated (correlation=0, independent) and normally distributed for both the models.

1.14 Checking for Overfitness

Over fitting exercise was carried out by adding more patameters to evaluate whether the selected model is overfitting. As such, one parameter has added to our moel as below.

```
pq0201=arima(cleaned_qtr_data,order=c(0,1,2),seasonal=list(order=c(0,1,1), period=4)) #Best model-seasonal ARIMA(0,1,2)×(0,1,1)4
pq0201
```

```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(0, 1, 2), seasonal = list(order = c(0,
##     1, 1), period = 4))
##
## Coefficients:
##           ma1      ma2     sma1
##       -0.3031  -0.1229  -0.7351
## s.e.   0.0819   0.0780   0.0728
##
## sigma^2 estimated as 97460:  log likelihood = -1111.89,  aic = 2229.78
```

```
pq0301=arima(cleaned_qtr_data,order=c(0,1,3),seasonal=list(order=c(0,1,1), period=4)) #new model wi
th extra parameter
pq0301
```

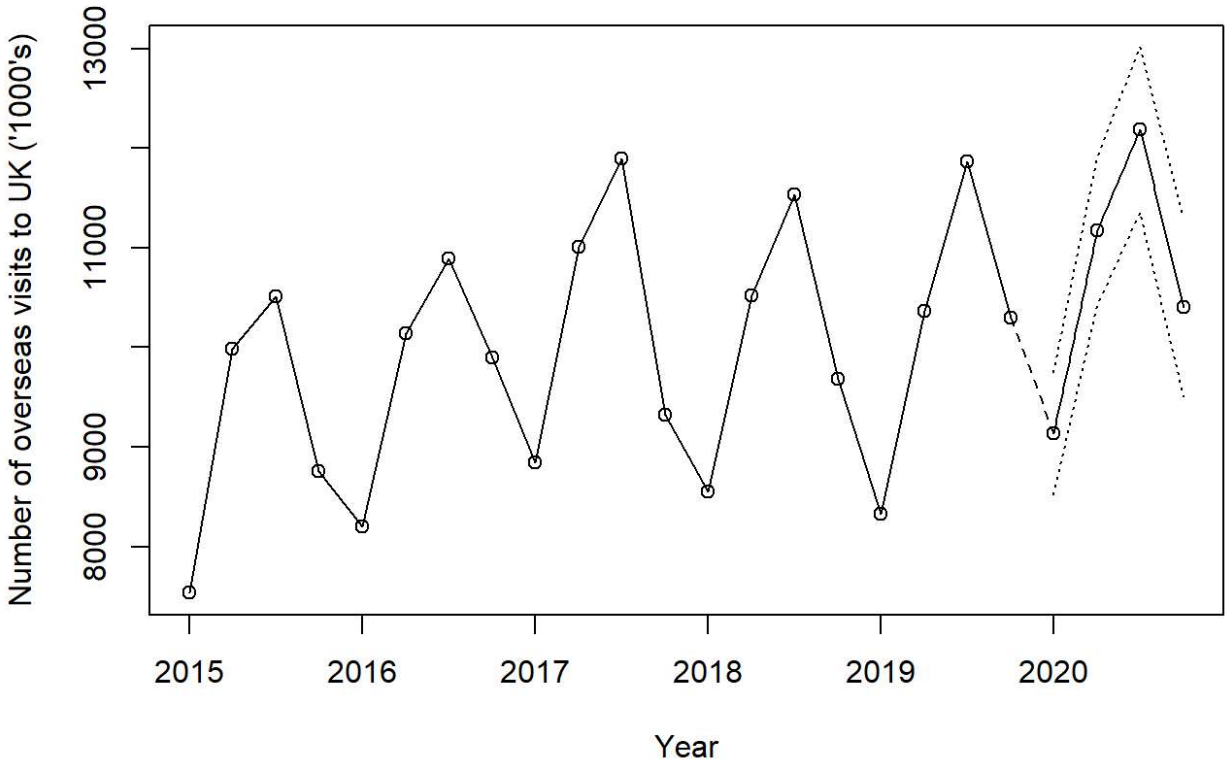
```
##
## Call:
## arima(x = cleaned_qtr_data, order = c(0, 1, 3), seasonal = list(order = c(0,
##     1, 1), period = 4))
##
## Coefficients:
##           ma1      ma2     ma3     sma1
##       -0.3006  -0.1207  -0.008  -0.7358
## s.e.   0.0854   0.0809   0.077   0.0731
##
## sigma^2 estimated as 97441:  log likelihood = -1111.88,  aic = 2231.77
```

New model (pq0301) has four parameters. The coefficient of the extra parameter is -0.008 which is very close to zero. Hence, adding a new parameter (MA3) is insignificant and made the model ineffective. Further, the AIC value for the new model (pq0301) is 2,231,77 which is slightly higher than the original model ‘pq0201’. Hence, It provide a sign that the selected model is a overfitted model.

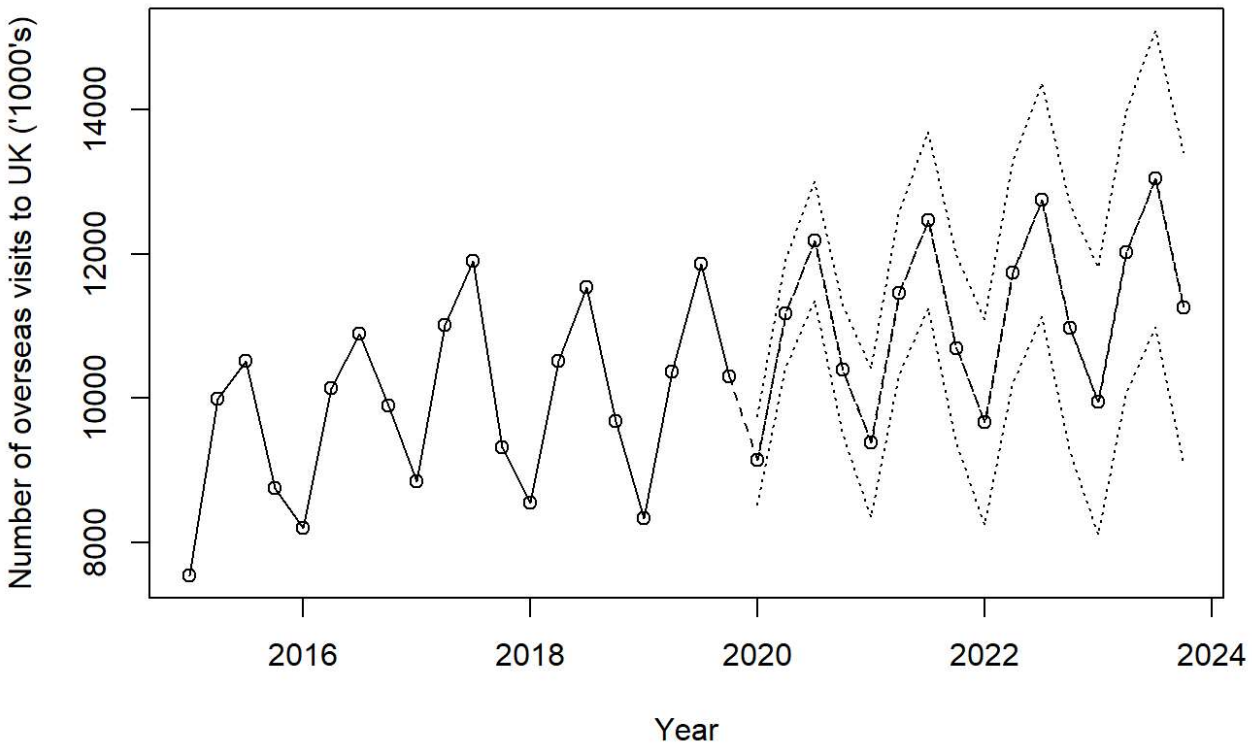
1.15 Forecasting of the model

Forecasting for the model pp0201 which is (Seasonal ARIMA(0,1,2)×(0,1,1)4) is as below.

```
plot(pq0201,n1=c(2015,1),n.ahead=4,xlab="Year",type="o", ylab="Number of overseas visits to UK ('10
00's)")
```




```
plot(pq0201,n1=c(2015,1),n.ahead=16, xlab="Year",type="o", ylab="Number of overseas visits to UK
 ('1000's)")
```



Seasonal ARIMA(0,1,2)×(0,1,1)4 model was used to forecast for next 4 and 16 quarters respectively. It could examine that the forecasted values of the model (Seasonal ARIMA(0,1,2)×(0,1,1)4) is similar to the pattern of the previous quarters of the year. Hence, model is appropriate and fitted to forecast the values of the time series.

The time plot for next 16 quarters indicate that the interval of confidence increases as the period increases which leads predictions become less precise. Hence, forecasting for quite lengthy periods tend to have increased interval of confidence.

1.16 Forecast errors

A forecast “error” is the difference between an observed value and its forecast. Accuracy of the model prediction depends on the length of the trained data. Hence, the data set used for this analysis has quite lengthy period of past data starting from Qtr1 of 1980 to Qtr 4 of 2019.

```
# train and test data
train_data <-window(cleaned_qtr_data,end=c(2018,4))
test_data  <- window(cleaned_qtr_data,start=c(2019,1))


pq0201_train<-arima(train_data,order=c(0,1,2),seasonal=list(order=c(0,1,1), period=4)) # pq0201 - S
ARIMA(0,1,2)×(0,1,1)4
pq1111_train<-arima(train_data,order=c(1,1,1),seasonal=list(order=c(1,1,1), period=4)) # pq1111 - S
ARIMA(1,1,1)×(1,1,1)4
```

```
# predict.Arima: Forecast from ARIMA fits
predict_pq0201 <- predict(pq0201_train,n.ahead = 4)
accuracy(predict_pq0201$pred, test_data)
```

```
##                 ME     RMSE      MAE       MPE     MAPE      ACF1 Theil's U
## Test set -154.9103 429.6566 385.6015 -1.808996 3.997368 0.2701336 0.2457753
```

```
predict_pq1111 <- predict(pq1111_train,n.ahead = 4)
accuracy(predict_pq1111$pred, test_data)
```

```
##                 ME     RMSE      MAE       MPE     MAPE      ACF1 Theil's U
## Test set -136.8498 413.8451 376.2989 -1.638778 3.905309 0.2837756 0.2277057
```

Above two test results is for the forecast errors of the two models nammely pq0201 and the pq1111. Lesser values for forecast error is given by the pq1111 model which is Seasonal ARIMA(1,1,1)×(1,1,1)4. Hence, based on the analysis of forecast errors, it assume Seasonal ARIMA(1,1,1)×(1,1,1)4 as the best model out of the tested models.

1.17 Predictions comparrison

```
#Values of fitted model
test_data
```

```
##       Qtr1  Qtr2  Qtr3  Qtr4
## 2019  8332 10364 11864 10297
```

```
predict_pq1111$pred
```

```
##           Qtr1      Qtr2      Qtr3      Qtr4
## 2019  8835.861 10886.436 11772.172  9909.930
```

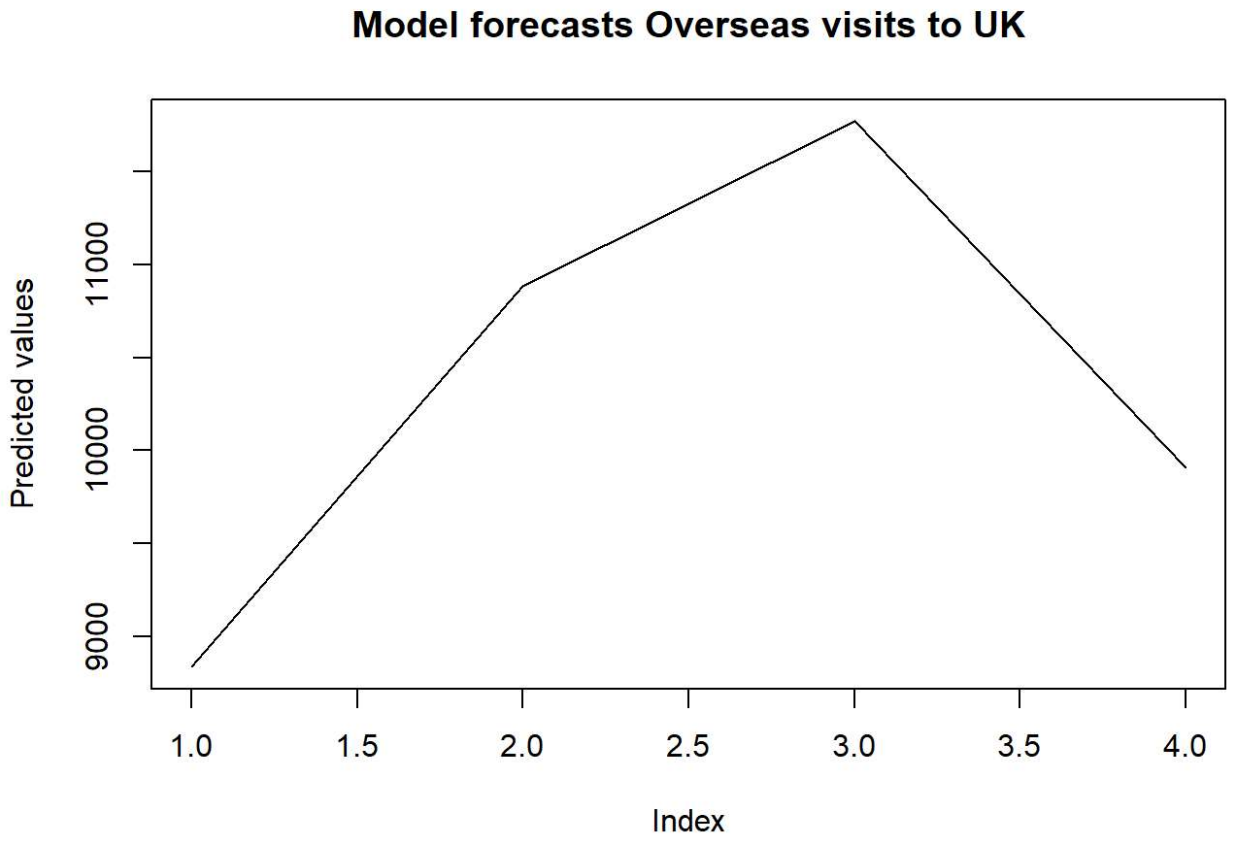
```
# Comparison among test and forecast data
aaa <- test_data
bbb <- predict_pq1111$pred
ccc <- data.frame(aaa,bbb)

names(ccc)<- c("Test data", "Forecast")
ccc
```

```
##   Test data  Forecast
## 1      8332  8835.861
## 2     10364 10886.436
## 3     11864 11772.172
## 4     10297  9909.930
```

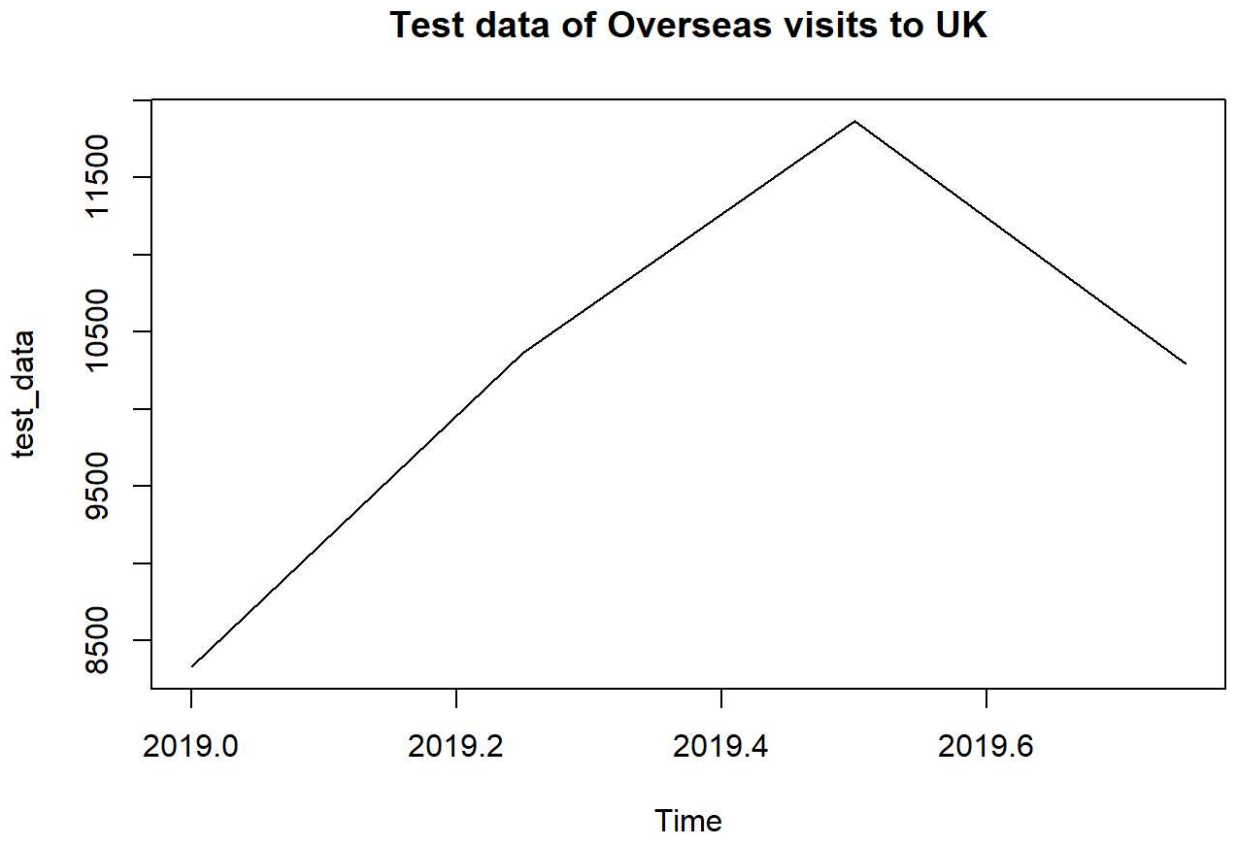
Above output illustrate the values of test data and predicted value of Seasonal ARIMA(1,1,1)×(1,1,1)4 model for the test data quarters of year 2020. As such, it was observed that the predicted values of the model follows similar pattern to the previous data. Further, values predicted of the fitted model is closer to the actual values. Further, below plots illustrate the patterns relevant to the above data.

1.18 Plotting predictions

```
#Ploting
plot(predict_pq1111$pred[1:4], type = "l", ylab="Predicted values" , main = "Model forecasts Overse
as visits to UK")
```




```
plot(test_data, main = "Test data of Overseas visits to UK" )
```



1.19 Conclusion:

Time series for number of overseas visits to UK was analysed in this study. As such, time series was convert to a stationary time series, develop list of possible models, model fitting and , residual analysis was performed. Finally, Seasonal ARIMA(1,1,1)×(1,1,1)4 was fitted to forecast the number of overseas visits to UK. Then, apply that model to train and test data set of the same time series and evaluate the prediction criteria. As such, the model follows the similar pattern to that of past data and predict values closer to the test value set.

Limitations and future developments:

This model can be improved by application of some machine learning prediction algorithms to predict the values almost similar to the test values. Further, this data set is related to the Leisure and Tourism industry, which was affected drastically during recent past. The data set used in this analysis is pre-Covid period. Hence, another model could be develop to track the in_Covid and post_Covid values.

#APPLICATION OF ARIMA FOR A ANNUAL DATA SET

##Task 1 – Exploratory Data Analysis

The data set provided with this assignment is called `Gross Domestic Product at market prices: CP: NSA £m', which is about the number of GDP market value for the period of 1948 to 2021. The annual data available in this data set has been used in the analysis below. However, both quarterly & annual values separately available in the same data set.

Source : (https://www.ons.gov.uk/economy/grossdomesticproductgdp/timeseries/bktl/pn2 (https://www.ons.gov.uk/economy/grossdomesticproductgdp/timeseries/bktl/pn2))

1.1 Import the file saved locally in the computer by using read.csv from base R.The first seven raw's of the data set has been eliminated since those consist with the Meta data of the file.

```
import_data <- read.csv("D:/Desktop/USW/Data Mining/cw2/GDP.csv", skip=7)
```

1.2 View of imported data

```
#View structure of imported data
str(import_data)
```

```
## 'data.frame':    342 obs. of  2 variables:
##  $ Important.notes
: chr  "1948" "1949" "1950" "1951" ...
##  $ Following.a.quality.review.it.has.been.identified.that.the.methodology.used.to.estimate.eleme
nts.of.purchased.software.within.gross.fixed.capital.formation..GFCF..has.led.to.some.double.counti
ng.from.1997.onwards..When.this.issue.is.amended.in.The.Blue.Book.2017.it.will.reduce.the.level.of.
GFCF.across.the.period.by.around.1.1..per.year..The.average.impact.on.quarter.on.quarter.GFCF.growt
h.is.negative.0.02..and.the.average.impact.on.quarter.on.quarter.GDP.growth.is.0.00..: int  11425 1
2168 12737 14298 15528 16675 17589 19160 20832 22093 ...
```

This data set includes 342 observations and 2 variables.

```
# View first 5 observations (Annual data)
import_data %>% head(5)
```

```
##   Important.notes
## 1            1948
## 2            1949
## 3            1950
## 4            1951
## 5            1952
##   Following.a.quality.review.it.has.been.identified.that.the.methodology.used.to.estimate.elemen
ts.of.purchased.software.within.gross.fixed.capital.formation..GFCF..has.led.to.some.double.countin
g.from.1997.onwards..When.this.issue.is.amended.in.The.Bl ...
## 1
11425
## 2
12168
## 3
12737
## 4
14298
## 5
15528
```

1.3 Selection of annual data

Tt was selected the raws which contains annual data that is raw 1 to 74 as below.

```
#Select only for annual data and transpose it to a vector
annual_data <- as.vector(t(import_data[1:74,2]))
annual_data
```

```
##  [1]   11425   12168   12737   14298   15528   16675   17589   19160   20832
## [10]   22093   23210   24378   26189   27905   29224   31090   34038   36818
## [19]   39383   41666   45786   49754   56020   62841   70502   81687   92534
## [28]  114936  136800  159524  185837  220586  259499  289653  318999  350813
## [37]  377386  414161  446361  496226  555980  615125  669873  706821  732295
## [46]  771414  812289  853228  911191  952585  998318 1041970 1098500 1142023
## [55] 1190336 1259970 1322795 1399656 1476722 1552470 1598752 1557120 1612195
## [64] 1669509 1721355 1793155 1876162 1935212 2016638 2097143 2174380 2255283
## [73] 2152646 2317667
```

```
class(all_data)
```

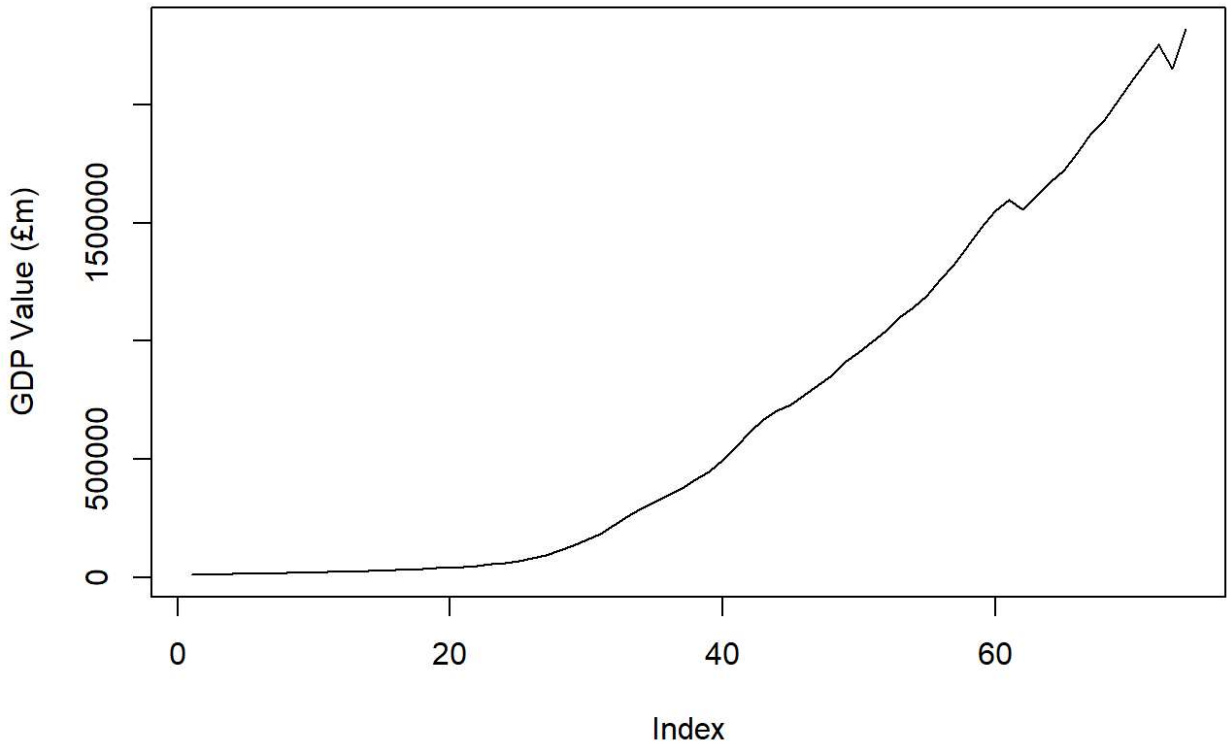
```
## [1] "data.frame"
```

```
class(annual_data)
```

```
## [1] "integer"
```

1.4 Plotting

```
# Plotting vector
annual_data %>% plot(type="l", ylab= "GDP Value (£m)")
```



1.5 Treatment to outliers and missing values

Missing values have not been examined in the data set. Outliers also not examine in this data set as there is a gradual increase over the time period.

1.6 Transformation

The above time series consist with money value which is GBP in millions. It is apparent that the moneytory value is reduced over time. Hence, GDP per-capita, adjusting the data for inflation using CPI are possible transformation options for this data set to smooth the effect of monitory value. However, those transformations were not applied at this point to the time series in this analysis.

1.7 Creation of time series object

```
# Time series object
annual_data %>% ts(frequency = 1, start = c(1948)) -> annual_data

annual_data
```

```
## Time Series:
## Start = 1948
## End = 2021
## Frequency = 1
##  [1]   11425   12168   12737   14298   15528   16675   17589   19160   20832
## [10]   22093   23210   24378   26189   27905   29224   31090   34038   36818
## [19]   39383   41666   45786   49754   56020   62841   70502   81687   92534
## [28]  114936  136800  159524  185837  220586  259499  289653  318999  350813
## [37]  377386  414161  446361  496226  555980  615125  669873  706821  732295
## [46]  771414  812289  853228  911191  952585  998318 1041970 1098500 1142023
## [55] 1190336 1259970 1322795 1399656 1476722 1552470 1598752 1557120 1612195
## [64] 1669509 1721355 1793155 1876162 1935212 2016638 2097143 2174380 2255283
## [73] 2152646 2317667
```

```
class(annual_data)
```

```
## [1] "ts"
```

```
#time series object details
class(annual_data)
```

```
## [1] "ts"
```

```
start(annual_data)
```

```
## [1] 1948    1
```

```
end(annual_data)
```

```
## [1] 2021    1
```

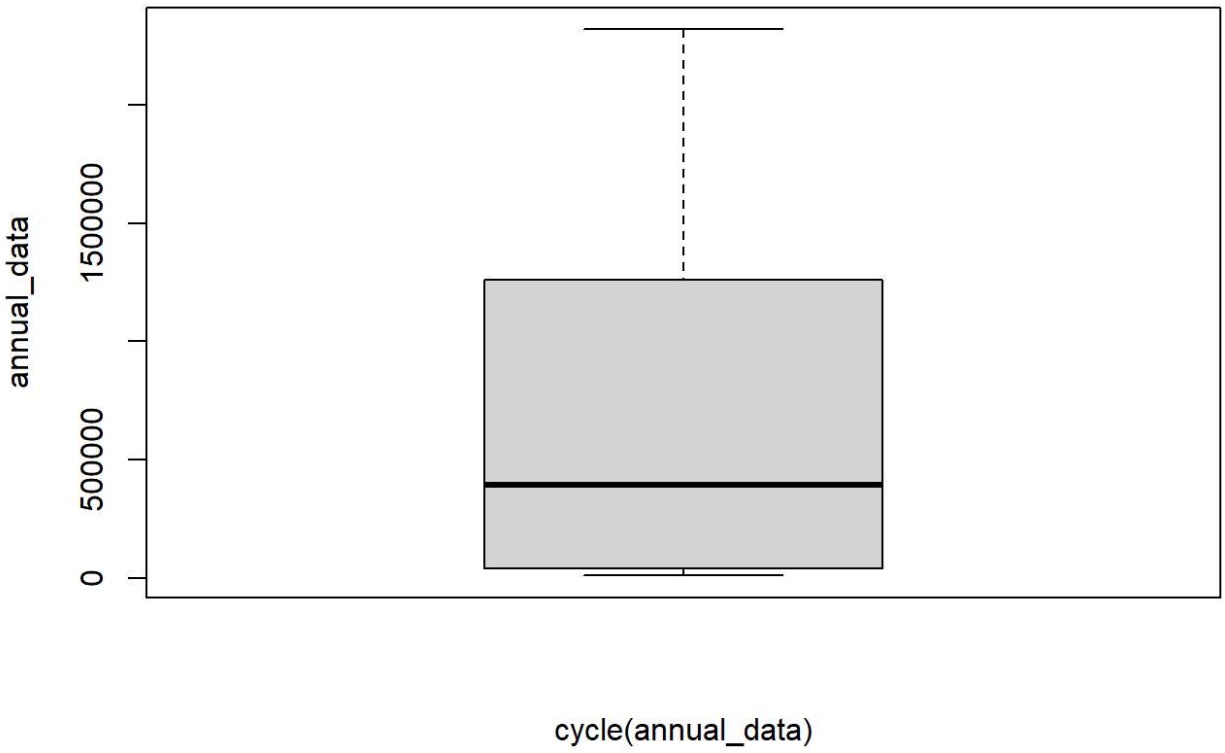
1.8 Summary statistics

```
#Summary statistics
summary(annual_data)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   11425   39954  395774  696582 1242562 2317667
```

As per the above output data, the average of GDP value is 696,582 (£m) for the period given, while minimum is 11,425 ((£m)) and maximum is 2,317,667 (£m).

```
#Boxplot for cycles
boxplot(annual_data ~cycle(annual_data))
```



Above summary statistics are explain in the Boxplot where more number of data points concentrated more arround lower two quartiles, where as there is a significant spread of values in upper quartiles.

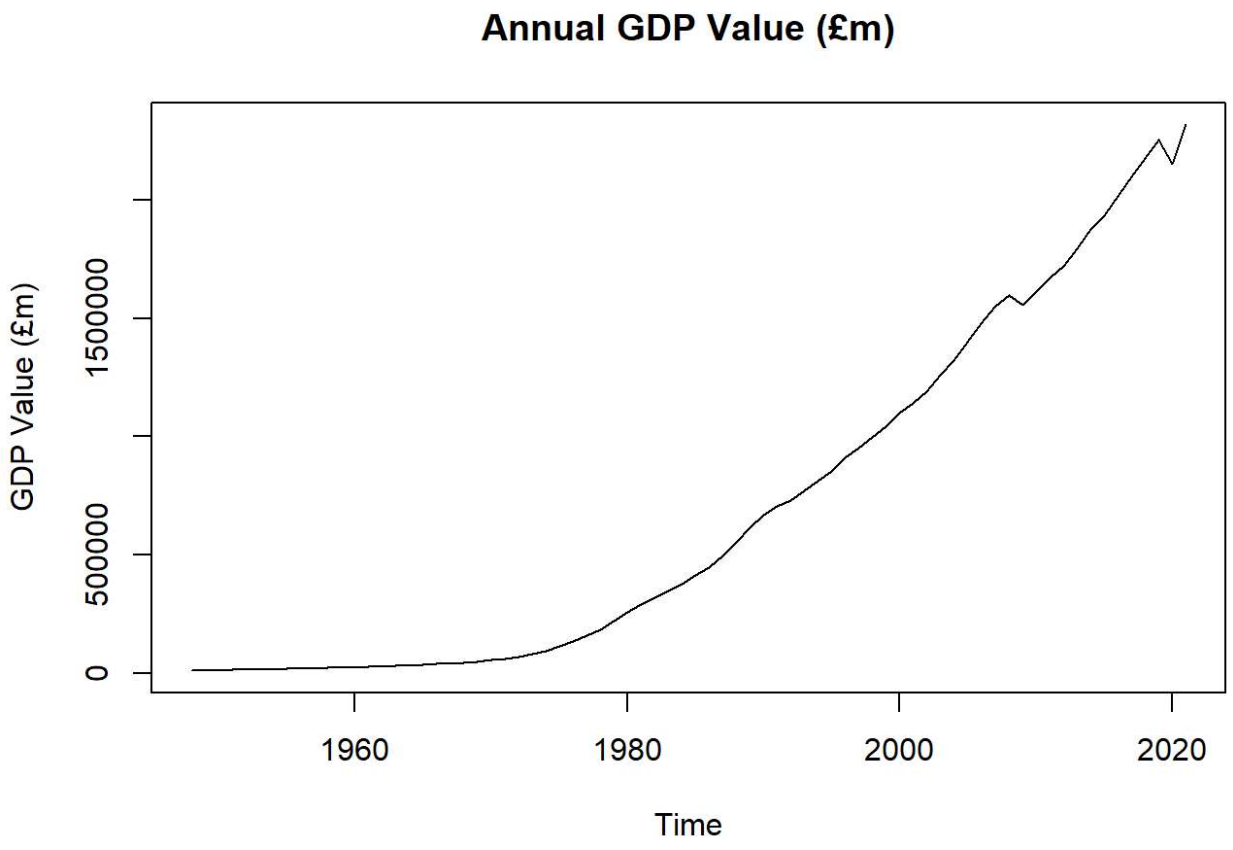
```
#Provides cycle across years
cycle(annual_data)
```

```
## Time Series:
## Start = 1948
## End = 2021
## Frequency = 1
##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [39] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

Original time series can be express as a function that explain the trend, seasonality effect and a stationary component(random component) as white noise. Hence, it was performed an analysis below to identify the trend and seasonality components in the created time series object simply by plotting the original time series data as below.

1.9 Plotting the time series object

```
# Plotting time series object
ts.plot(window(annual_data, start =c(1948)),  ylab="GDP Value (£m)", main="Annual GDP Value (£m)")
```



Above Time plot illustrate followings;

There is a increasing trend (upward trend) as the time goes up, GDP value also growing up in general. It can argue as a quadratic trend also. The time series do not indicate any sense of seasonality. There is a slight drop in the trend pattern in 2009 and 2020. This is mainly due to the economic downturn/ressession in 2009 and Covid 19 effect.

Task 2 – Model fitting and Forecasting

Stationary time series is consist of trend or seasonal patterns and a random noise component. A time series is said to be stationary if it holds the following conditions true;

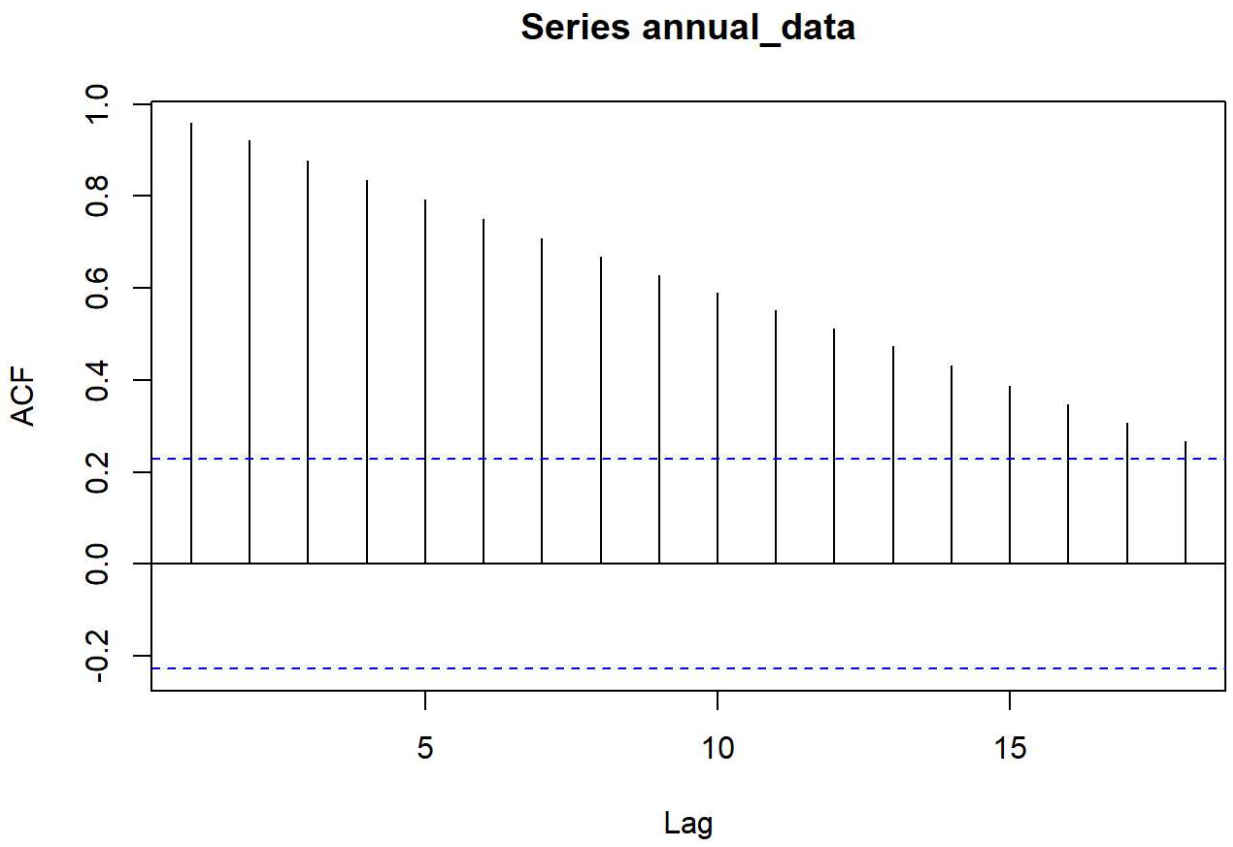
The the expected value (mean value) of time-series is constant over time (which implies, the trend component is nullified) variance is constant which oscillate around a constant value, and the correlation depends on the time lag, but not the absolute.

It is important to note that the process defined in this way is weakly stationary. ACF plot and hypothesis testing (ADF & KPSS) was performed to identify that the time series object is stationary or not.

1.10.1 Checking for stationary - Autocorrelation function (ACF)

The autocorrelation coefficients are plotted to show the autocorrelation function or ACF. ACF plots observe if the data has a trend and a seasonal component. The plot is also known as a correlogram which is in below.

```
#Autocorrelation function for original time series
acf(annual_data)
```



Above plot of ACF indicates significant positive correlations. The autocorrelations decreases slowly as the number of lags increases. Further, there is no any clear seasonal pattern visibly in the ACF. Hence, ACF indicates trended time series. In a stationary time series, the ACF will drop to zero relatively quickly as it doesn't depends on time, while the ACF of non-stationary data decreases slowly as in the above plot. Further, more than 5% of spikes are outside the boundaries. As such, the above ACF indicates the attributes of non-stationary time series or not a white noise.

Further, it is also useful to quantify the evidence of non-stationary via hypothesis testing. Hence, the Dickey -Fuller test and KPSS tests has been used in below.

1.10.2 Checking for stationary - Augmented Dickey-Fuller unit-root test (ADF)

```
#Dickey-Fuller test
adf.test(annual_data)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  annual_data
## Dickey-Fuller = -1.44, Lag order = 4, p-value = 0.8037
## alternative hypothesis: stationary
```

H0: Time series is not a stationary, H1: Time series is a stationary

The p-value of above ADF test is 0.8037 which is not significant at the 5% level. Hence, the null hypothesis should not be rejected, since that there is enough evidence to suggest that there the time series is not stationary.

1.10.3 Checking for stationary - Kwiatkowski-Phillips-SchmidtShin (KPSS) test

```
#KPSS test
kpss.test(annual_data)
```

```
## Warning in kpss.test(annual_data): p-value smaller than printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  annual_data
## KPSS Level = 1.7955, Truncation lag parameter = 3, p-value = 0.01
```

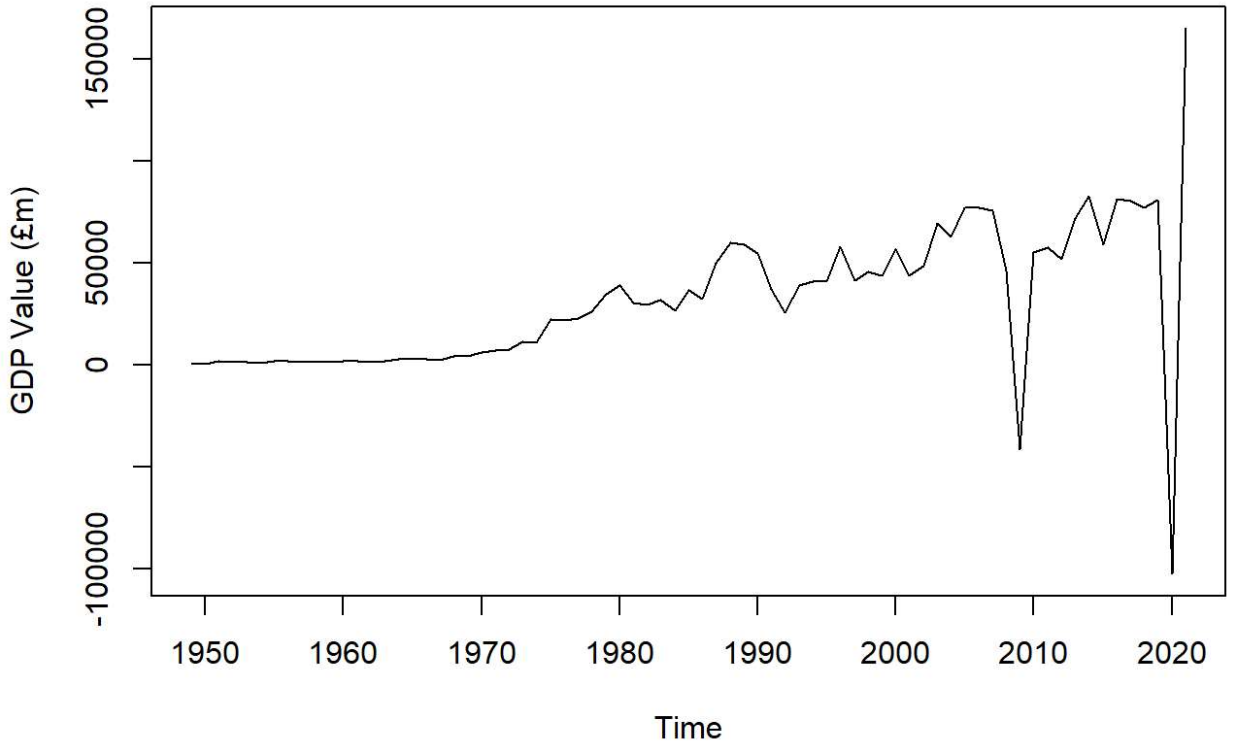
H0: Time series is a stationary, H1: Time series is not a stationary

The p-value of the KPSS test is smaller than 0.01, which is significant at the 5% level. Hence, the null hypothesis should be rejected, since there is enough evidence in favor of alternative hypothesis (H1) and conclude that the time series is not a stationary.

It is apparent that the above time series is a non-stationary since it has a trend component.Hence, it is required to convert to a stationary time series to construct a appropriate model for forecasting.

1.11 Stationary through Differencing

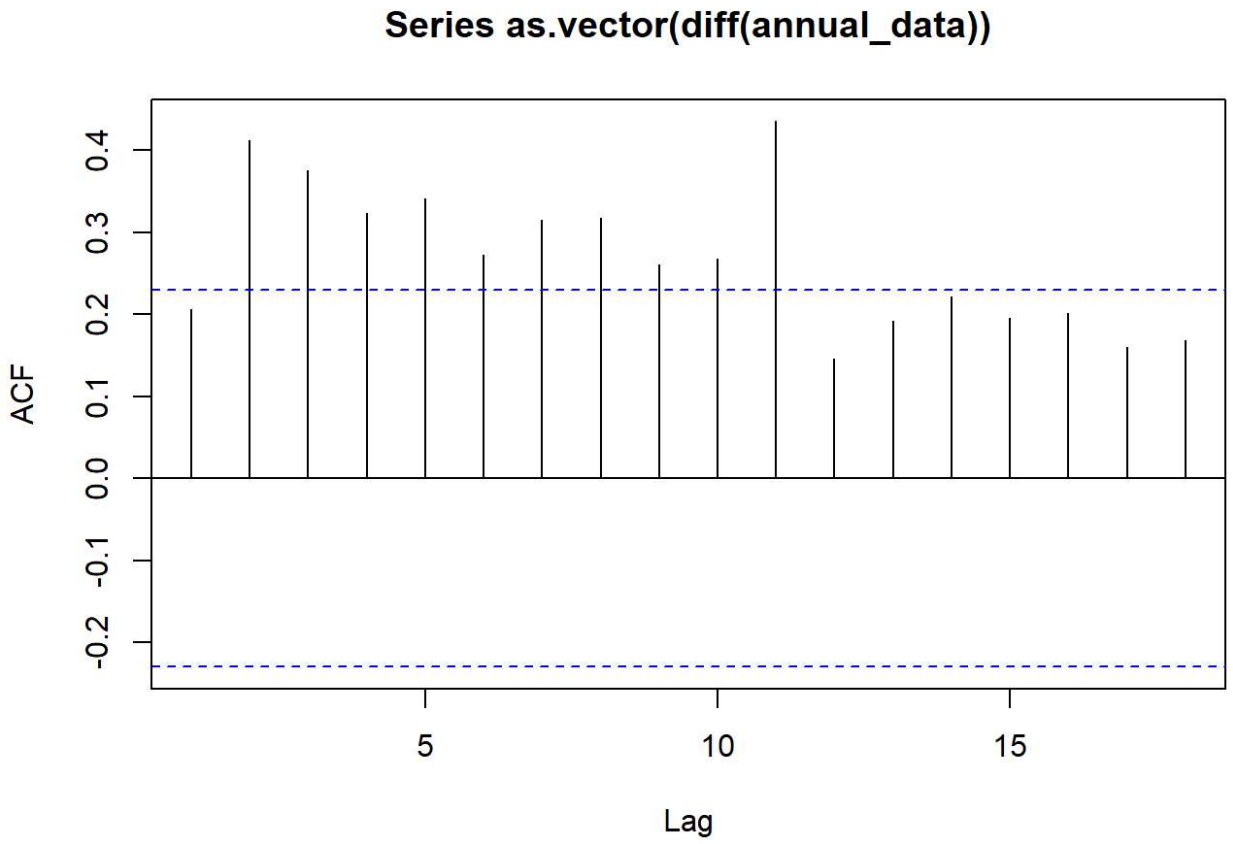
```
#First difference
plot(diff(annual_data),ylab="GDP Value (£m)",xlab="Time")
```



A time plot after the first difference is illustrate above. In this case, there indicate some random pattern, however, trend is still visible in the series. That is an indication of the for the requirement of second differencing to convert the time series to a white noise.

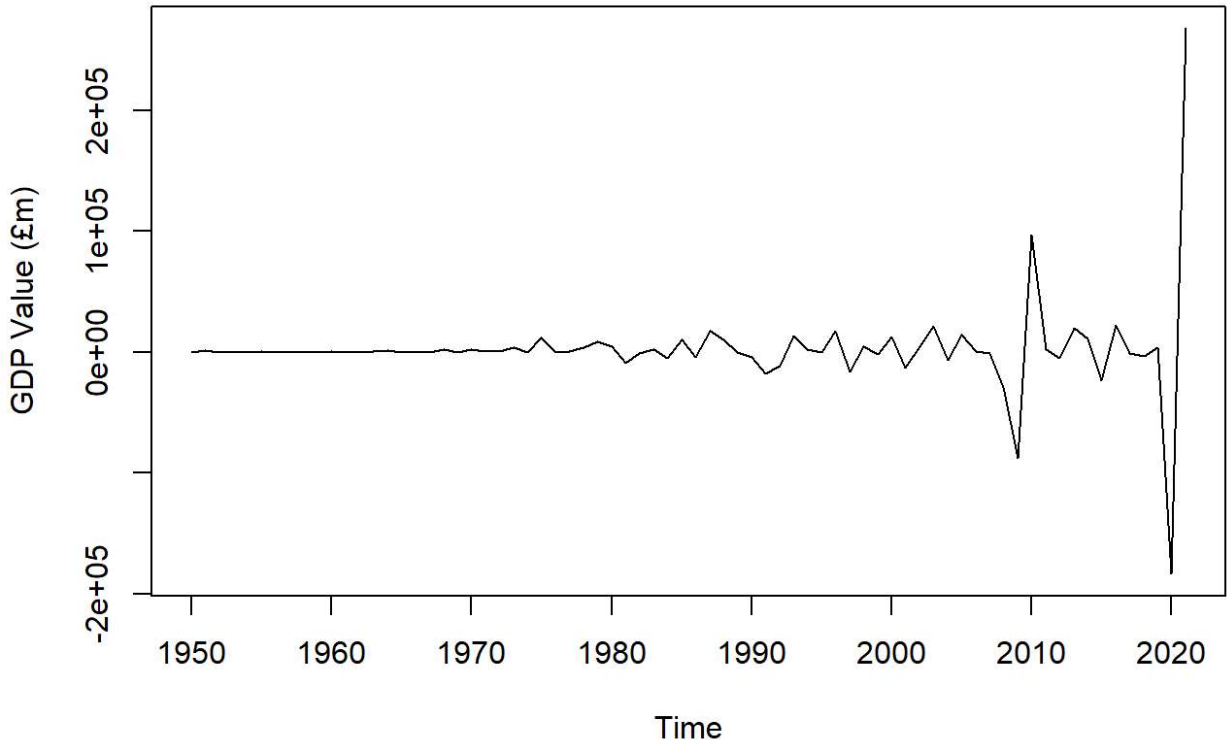
-

```
#Autocorrelation function for the first Difference
acf(as.vector(diff(annual_data)))
```



Above ACF plot indicates autocorrelation function for the first difference. There are many significant spikes beyond the interval of confidence. Hence, second difference is applied to the time series to remove this effect.

```
# First and Second Difference
plot(diff(diff(annual_data)),xlab="Time", ylab="GDP Value (£m)")
```



The above figure displays the time plot after application of both first difference and second difference.As such, there is no trend component, since the values are oscillate around a constant level. Further, few ups and downs are noticed in the time series which is insignificant. The evidence of non-stationarity could be quantify via hypothesis testing further. Thus, Dickey -Fuller test has been applied below.

Checking for stationary - Augmented Dickey-Fuller unit-root test (ADF)

```
#Dickey-Fuller test
adf.test(diff(diff(annual_data)))
```

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

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(diff(annual_data))
## Dickey-Fuller = -4.9389, Lag order = 4, p-value = 0.01
## alternative hypothesis: stationary
```

H0: Time series is not a stationary, H1: Time series is a stationary

The p-value of above ADF test is smaller 0.01, which is significant at the 5% level. Hence, the null hypothesis should be rejected, since there is enough evidence in favor of alternative hypothesis (H1) and conclude that the time series is a stationary.

-

```
#KPSS test
kpss.test(diff(diff(annual_data)))
```

```
## Warning in kpss.test(diff(diff(annual_data))): p-value greater than printed p-
## value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  diff(diff(annual_data))
## KPSS Level = 0.098633, Truncation lag parameter = 3, p-value = 0.1
```

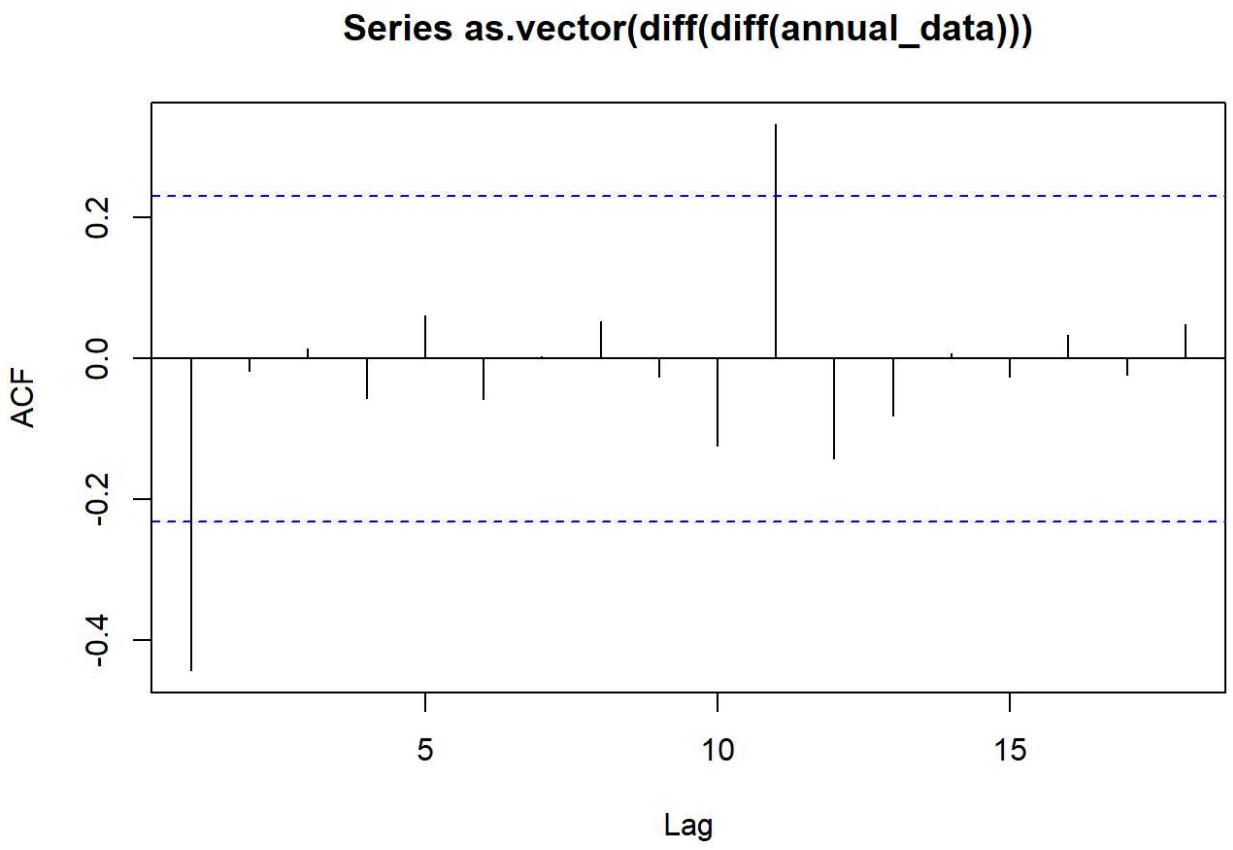
H0: Time series is a stationary, H1: Time series is not a stationary

The p-value of above KPSS test is 0.1, which is not significant at the 5% level. Hence, the null hypothesis should not be rejected, since that there is enough evidence to suggest that there the time series is a stationary.

1.12 Model Specification

The following ACF, PACF and EACF plots have been analysed to determine the parameters initially.

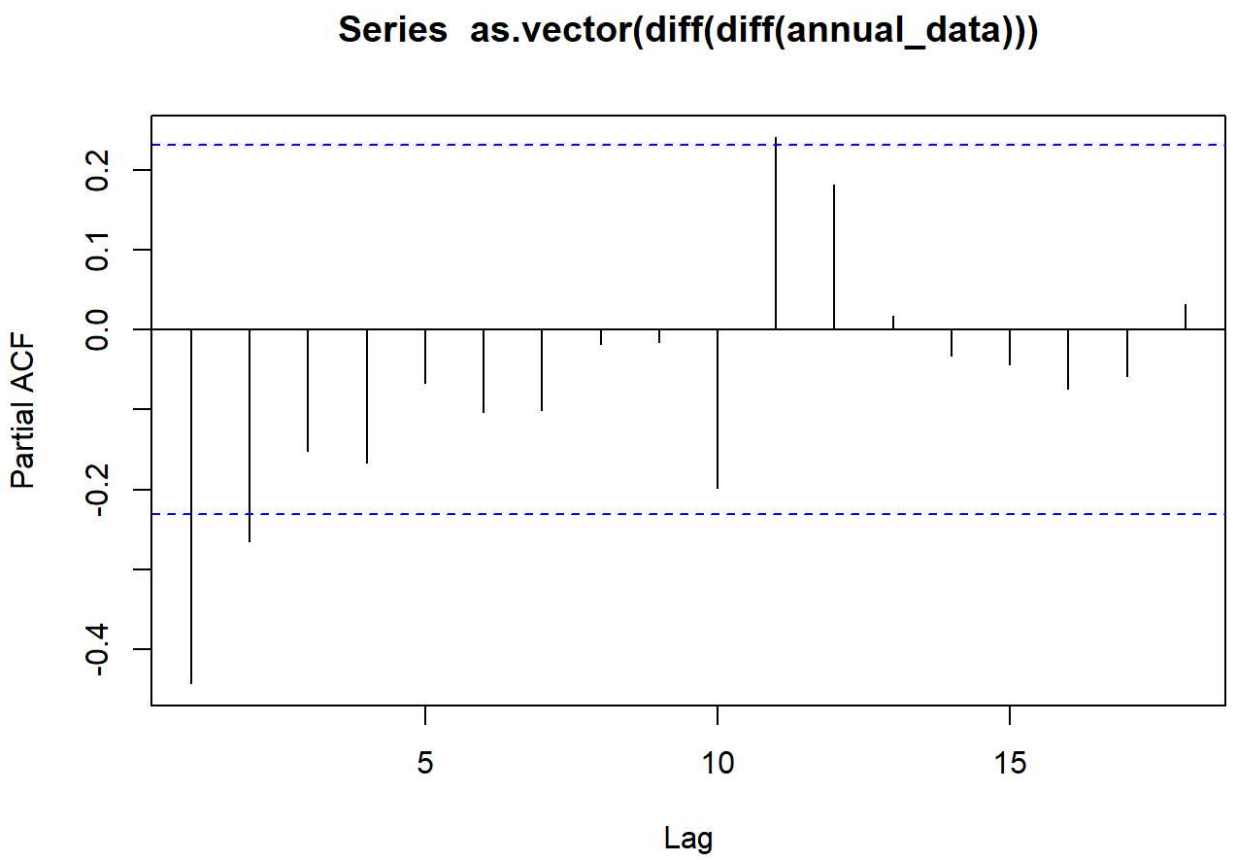
```
#Autocorrelation function stationary time series
acf(as.vector(diff(diff(annual_data))))
```



Above ACF function illustrate that approximately 95% of the spikes are within the interval of confidence. However, there are very few significant autocorrelations are out from the interval of confidence. That is lag 1 and 11 which is significantly different from zero and rest of all lags are zero or closer to zero. Hence, the time series is a is a stationary.

Above ACF plot could be used to determine the moving average (MA) component of model specification. As such, above ACF illustrate non-seasonal moving average component (q) which could be assume as 1. The difference (d) was set as 2 (d=2), since two differencing were applied to the series.

```
#Partial Autocorrelation function stationary time series
pacf(as.vector(diff(diff(annual_data))))
```



PACF function indicates that very few significant autocorrelations are out from the interval of confidence. However, it can assume that the 95% are within the interval of confidence. That is lag 1,2 and 11 which is significantly different from zero and rest of all lags are zero or closer to zero. Hence, the time series is a is a stationary.

Thus,PACF plot illustrate autoregressive component (AR) component of model specification. Hence, by examining the above PACF plot, it could assume non-Seasonal autoregressive component (p) as 1 or 2. The difference (d) was set as 2 (d=2), since two differencing were applied to the series.

```
#eacf for stationary time series
eacf(as.vector(diff(diff(annual_data))),ar.max=10, ma.max=10)
```

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

The EACF table illustrate a triangular pattern of zeroes with the top-0,1 point. Thus, EACF plot illustrate AR/MA as (0,1) is as a appropriate for the model specification.

1.13 Parameter Estimation

Accordingly, ARIMA(p,d,q) model would be appropriate to apply for the time series in this analysis where nonseasonal orders (p, d, q). Thus, ACF, PACF plots and AIC, BIC criteria has been used in the process of model specification & parameter estimation of the model.

Accordingly, list of possible models have been evaluated in below to identify the most appropriate model for the time series.

```
#List of possible models and Parameter estimation - ARIMA (p,d,q)

pq01 =arima(annual_data,order=c(0,2,1),method="ML")
pq01
```

```
##
## Call:
## arima(x = annual_data, order = c(0, 2, 1), method = "ML")
##
## Coefficients:
##           ma1
##       -0.8623
## s.e.   0.0435
##
## sigma^2 estimated as 903409140:  log likelihood = -845.22,  aic = 1692.45
```

```
pq11=arima(annual_data,order=c(1,2,1),method="ML")
pq11
```

```
##
## Call:
## arima(x = annual_data, order = c(1, 2, 1), method = "ML")
##
## Coefficients:
##           ar1      ma1
##       -0.3516  -0.7999
## s.e.   0.1440   0.0721
##
## sigma^2 estimated as 830713046:  log likelihood = -842.35,  aic = 1688.7
```

```
pq21=arima(annual_data,order=c(2,2,1),method="ML")
pq21
```

```
##
## Call:
## arima(x = annual_data, order = c(2, 2, 1), method = "ML")
##
## Coefficients:
##           ar1     ar2      ma1
##       -0.3351  0.1594  -0.8268
## s.e.   0.1426  0.2216   0.0722
##
## sigma^2 estimated as 824368583:  log likelihood = -842.09,  aic = 1690.18
```

```
pq12=arima(annual_data,order=c(1,2,0),method="ML")
pq12
```

```
##
## Call:
## arima(x = annual_data, order = c(1, 2, 0), method = "ML")
##
## Coefficients:
##           ar1
##       -0.9099
## s.e.   0.0918
##
## sigma^2 estimated as 994305735:  log likelihood = -848.88,  aic = 1699.75
```

```
pq20=arima(annual_data,order=c(2,2,0),method="ML")
pq20
```

```
##
## Call:
## arima(x = annual_data, order = c(2, 2, 0), method = "ML")
##
## Coefficients:
##           ar1      ar2
##       -1.0451  -0.4292
## s.e.   0.1291   0.1856
##
## sigma^2 estimated as 927345619:  log likelihood = -846.07,  aic = 1696.14
```

```
pq00=arima(annual_data,order=c(0,2,0),method="ML")
pq00
```

```
##
## Call:
## arima(x = annual_data, order = c(0, 2, 0), method = "ML")
##
##
## sigma^2 estimated as 1.78e+09:  log likelihood = -868.96,  aic = 1737.91
```

More than 5 possible out puts were tested. As per those output results,the lowest AIC value which is 1,688.78 for “pq11”, the second lowest AIC value of 1,690.18 for “pq21” and the third lowest AIC value of 1,692.45 for “pq01”. Hence, it can assume ‘pq11’ as the best model based on the AIC value and the second best as ‘pq21’.

```
#Model evaluation-BIC criteria
BIC(pq11)
```

```
## [1] 1697.527
```

```
BIC(pq21)
```

```
## [1] 1701.289
```

```
BIC(pq01)
```

```
## [1] 1699.003
```

The three models which has the lowest AIC value was tested for BIC criteria. Accordingly, ‘pq11’ has the lowest BIC value, then ‘pq01’ and ‘pq21’ respectively. Thus, both AIC and BIC criteria suggest ‘pq11’ as the best model.

Model Attribute Comparrison

```
#AIC lowest model in details

pq11
```

```
##
## Call:
## arima(x = annual_data, order = c(1, 2, 1), method = "ML")
##
## Coefficients:
##           ar1      ma1
##       -0.3516  -0.7999
## s.e.   0.1440   0.0721
##
## sigma^2 estimated as 830713046:  log likelihood = -842.35,  aic = 1688.7
```

```
pq21
```

```
##
## Call:
## arima(x = annual_data, order = c(2, 2, 1), method = "ML")
##
## Coefficients:
##           ar1     ar2      ma1
##       -0.3351  0.1594  -0.8268
## s.e.   0.1426  0.2216   0.0722
##
## sigma^2 estimated as 824368583:  log likelihood = -842.09,  aic = 1690.18
```

```
pq01
```

```
##
## Call:
## arima(x = annual_data, order = c(0, 2, 1), method = "ML")
##
## Coefficients:
##           ma1
##       -0.8623
## s.e.   0.0435
##
## sigma^2 estimated as 903409140:  log likelihood = -845.22,  aic = 1692.45
```

The attributes of the model are describe below.

The lowest AIC & BIC values given by ‘pq11’ model which has 2 parameters (ar1 and ma1) and the coefficients of parameters are -0.3516, and -0.7999 respectively. Square error also quite small for the two parameters (0.1440, and 0.0721 respectively). Number of parameters are lower in the model ‘pq11’ than the ‘pq21’ model and lesser the parameters is better the model.

The second lowest AIC value is for ‘pq21’ model which has three parameters (ar1, ar2 and ma1) and coefficients of the parameter are -0.3351, 0.1594, -0.8268 respectively. Square error also quite small for the parameters and AIC value as 1690.18.

As such, two models (pq11 and pq01) could be specify as below.

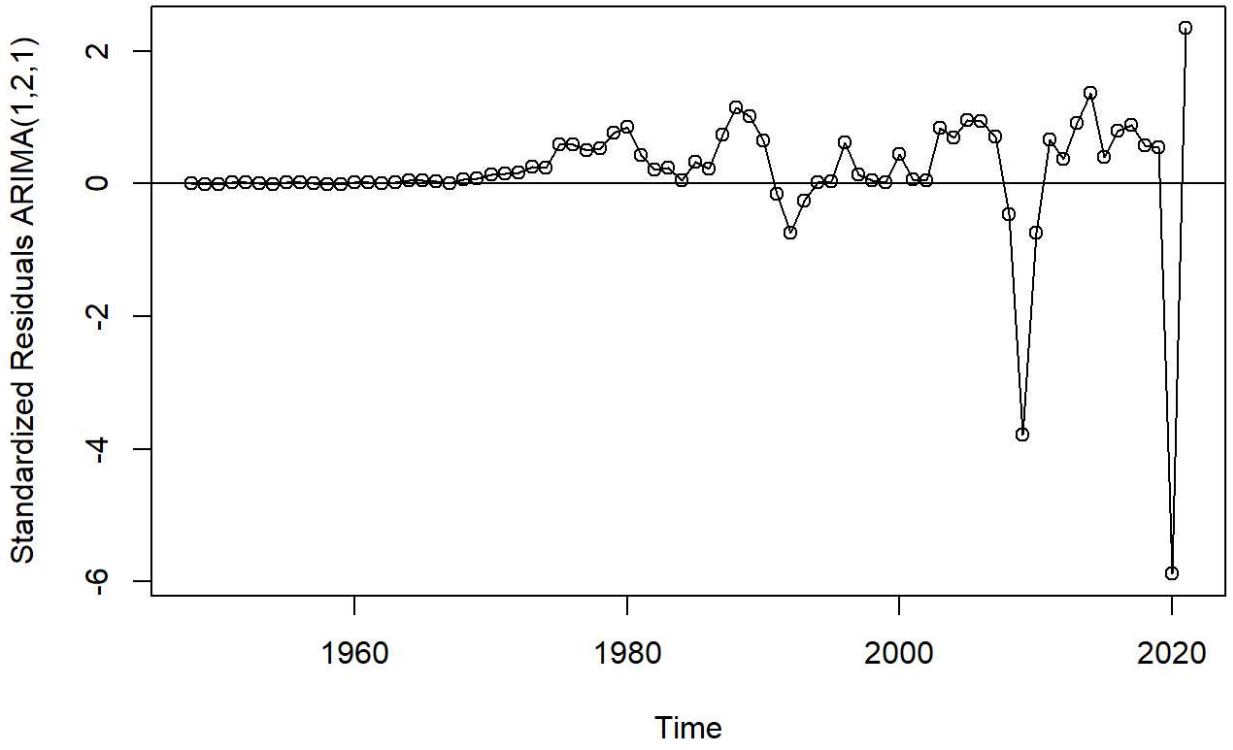
Best model out of the evaluated list of models : ARIMA(1,2,1) Second best model out of the evaluated list of models : ARIMA(2,2,1)

1.14 Residual Analysis (Estimate of White noise)

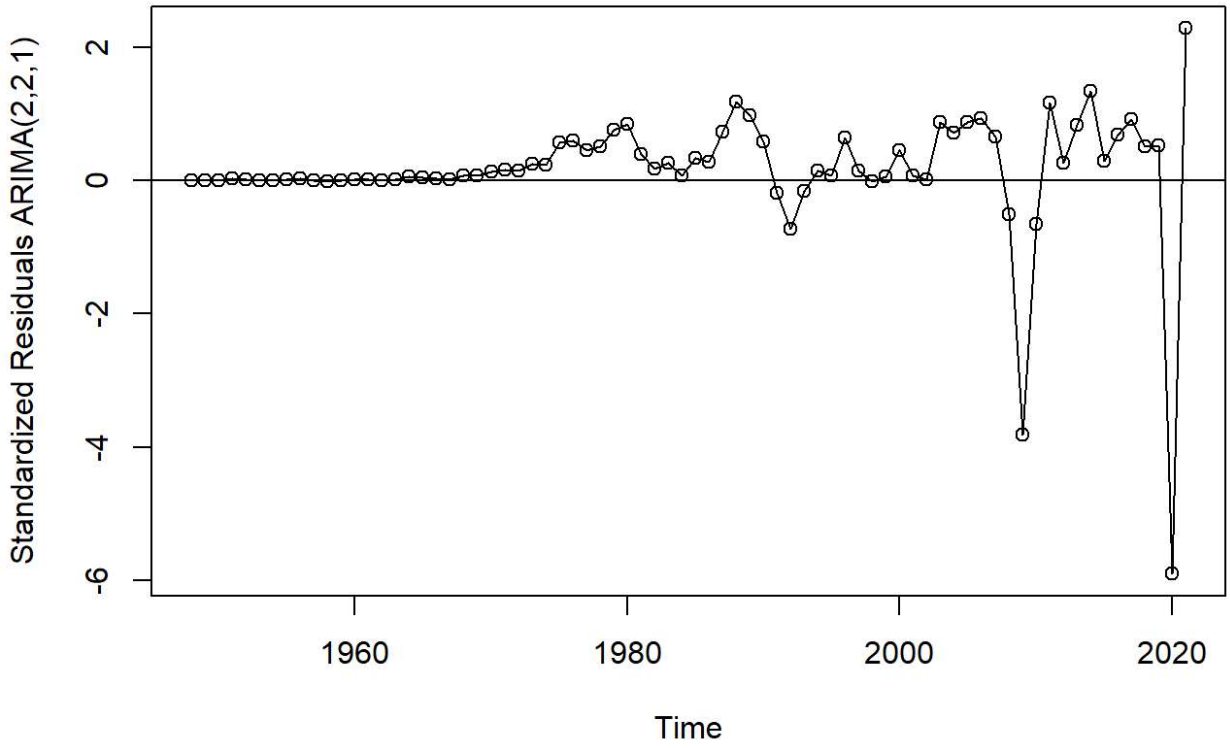
Residuals should be a white noise which is a weaker condition of stationary. Accordingly analysis is performed for residuals as below for the above selected two models; ARIMA(1,2,1) and ARIMA(2,2,1)

1.14.1 Plots of the Residuals

```
#Reseduals plots for 2 selected models
plot(rstandard(pq11),ylab ="Standardized Residuals ARIMA(1,2,1)", type="o"); abline(h=0)
```




```
plot(rstandard(pq21),ylab ="Standardized Residuals ARIMA(2,2,1)", type="o"); abline(h=0)
```

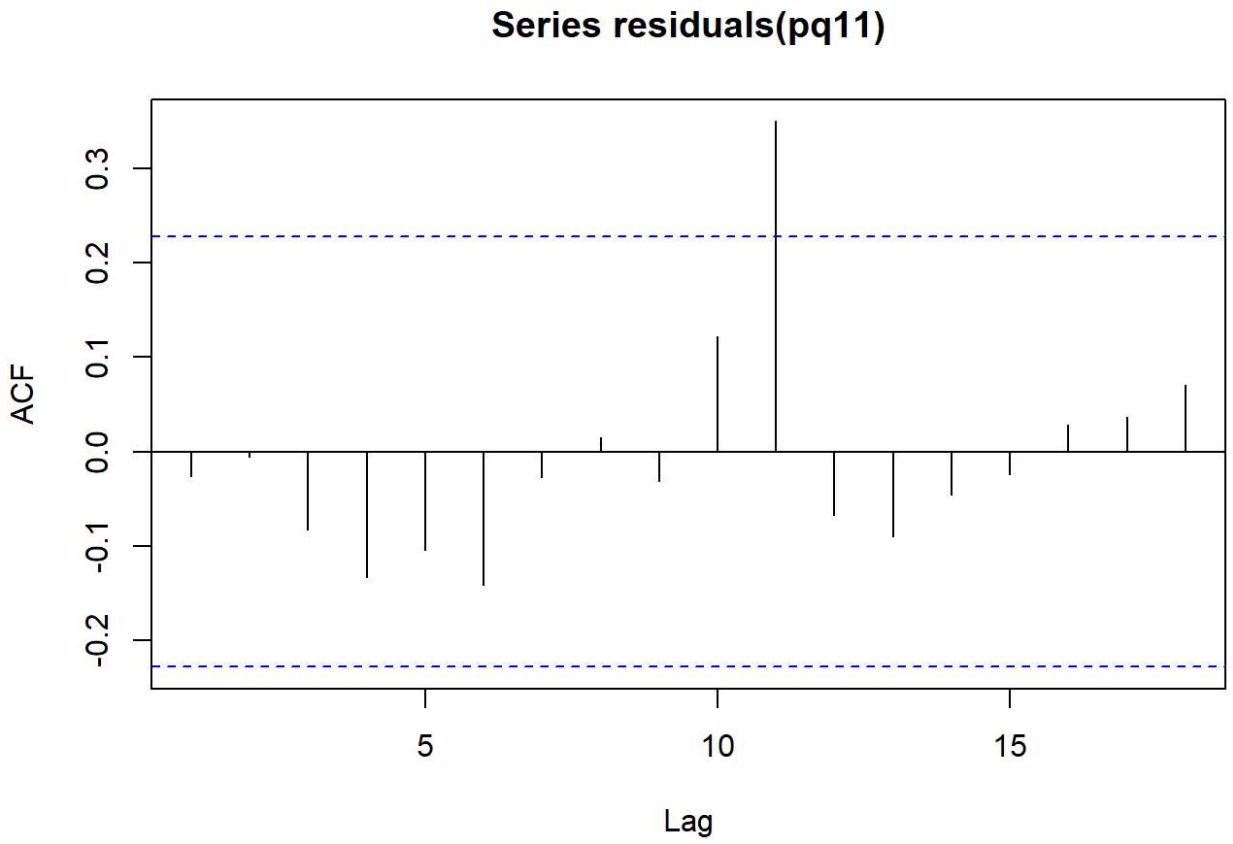


The above residual plots illustrate that the majority of residuals are oscillate around zero and approximately 95% of data fall between +2 & -2. However, the reseduals do not indicate a pattern, but the distributed randomly.

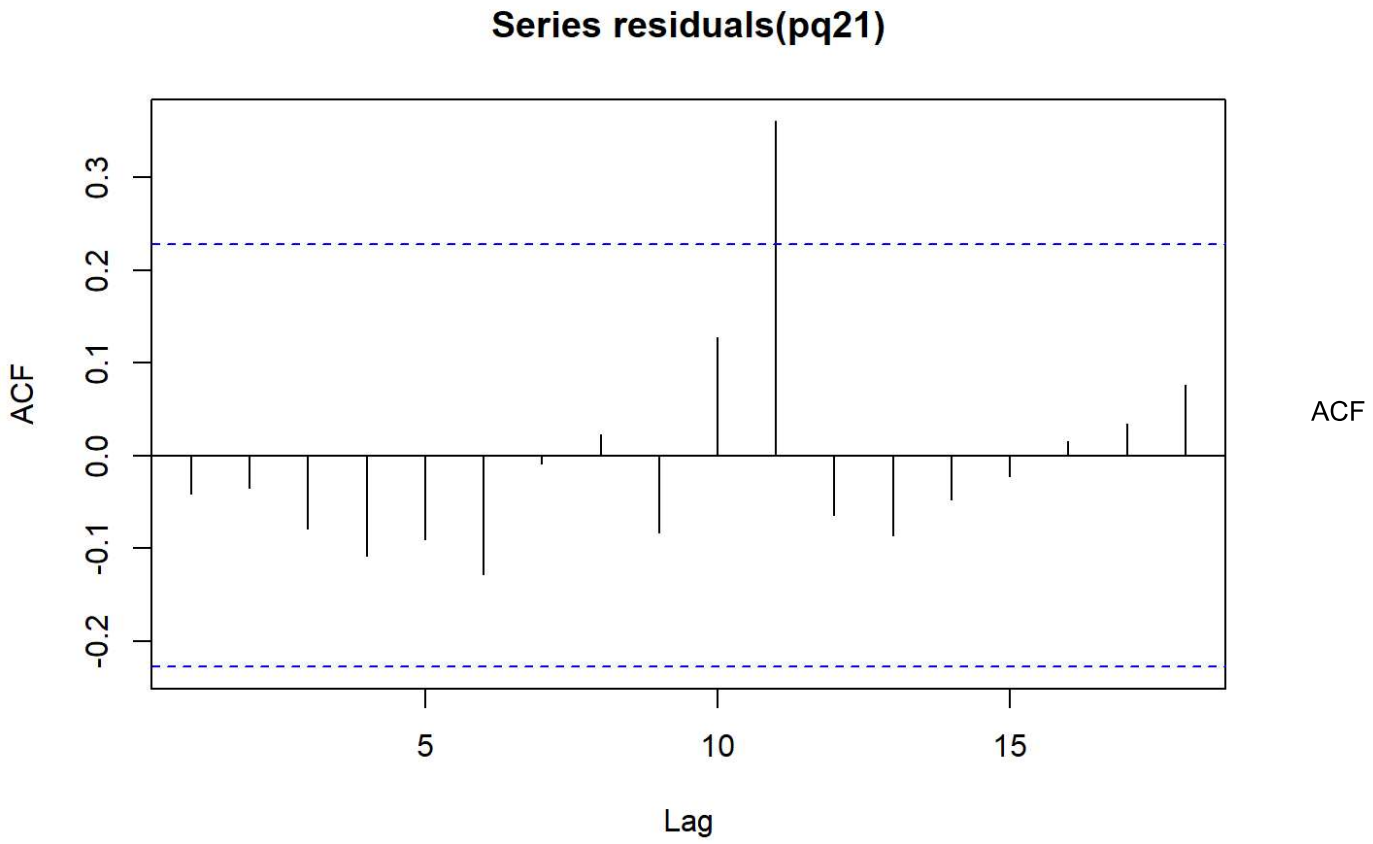
1.14.2 ACF of Residuals

As residuals should be a stationary which is independent and identically distributed. Hence, ACF of residuals should be close to zero and 95% of lags should be inside the interval of confidence.

```
acf(residuals(pq11)) # ARIMA(1,2,1)
```




```
acf(residuals(pq21)) # ARIMA(2,2,1)
```



function for residuals indicates that only one or two autocorrelation are out from the interval of confidence. Hence, that the 95% is within the interval of confidence and are approximately zero or closer to zero. As such, it appears to be no significant autocorrelation in the residuals and the residuals are uncorrelated.

1.14.3 Statistical tests for Residuals - Box-Pierce test and the Ljung-Box test.

(H0 : the data are sample from an iid sequence)

```
# ARIMA(1,2,1)
Box.test(residuals(pq11)) #Box-Pierce test
```

```
##
##  Box-Pierce test
##
## data:  residuals(pq11)
## X-squared = 0.052283, df = 1, p-value = 0.8191
```

```
Box.test(residuals(pq11), type=c("Ljung-Box")) #Ljung-Box test
```

```
##
##  Box-Ljung test
##
## data:  residuals(pq11)
## X-squared = 0.054431, df = 1, p-value = 0.8155
```

```
# ARIMA(2,2,1)
Box.test(residuals(pq21)) #Box-Pierce test
```

```
##
##  Box-Pierce test
##
## data:  residuals(pq21)
## X-squared = 0.12426, df = 1, p-value = 0.7245
```

```
Box.test(residuals(pq21), type=c("Ljung-Box")) #Ljung-Box test
```

```
##
##  Box-Ljung test
##
## data:  residuals(pq21)
## X-squared = 0.12936, df = 1, p-value = 0.7191
```

H0 : Residuals or error terms are uncorrelated

The p-value of above Box-Pierce test and the Ljung-Box tests are not significant at the 5% level. Hence, the null hypothesis should not be rejected, since that there is enough evidence to suggest for Residuals or error terms are uncorrelated.

1.14.4 Stationary of Residuals - ADF test

```
#Dickey-Fuller test
Res_pq11 <- residuals(pq11) # ARIMA(1,2,1)
adf.test(Res_pq11)
```

```
## Warning in adf.test(Res_pq11): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  Res_pq11
## Dickey-Fuller = -4.5191, Lag order = 4, p-value = 0.01
## alternative hypothesis: stationary
```

```
Res_pq21 <- residuals(pq21) # ARIMA(2,2,1)
adf.test(Res_pq21)
```

```
## Warning in adf.test(Res_pq21): p-value smaller than printed p-value
```

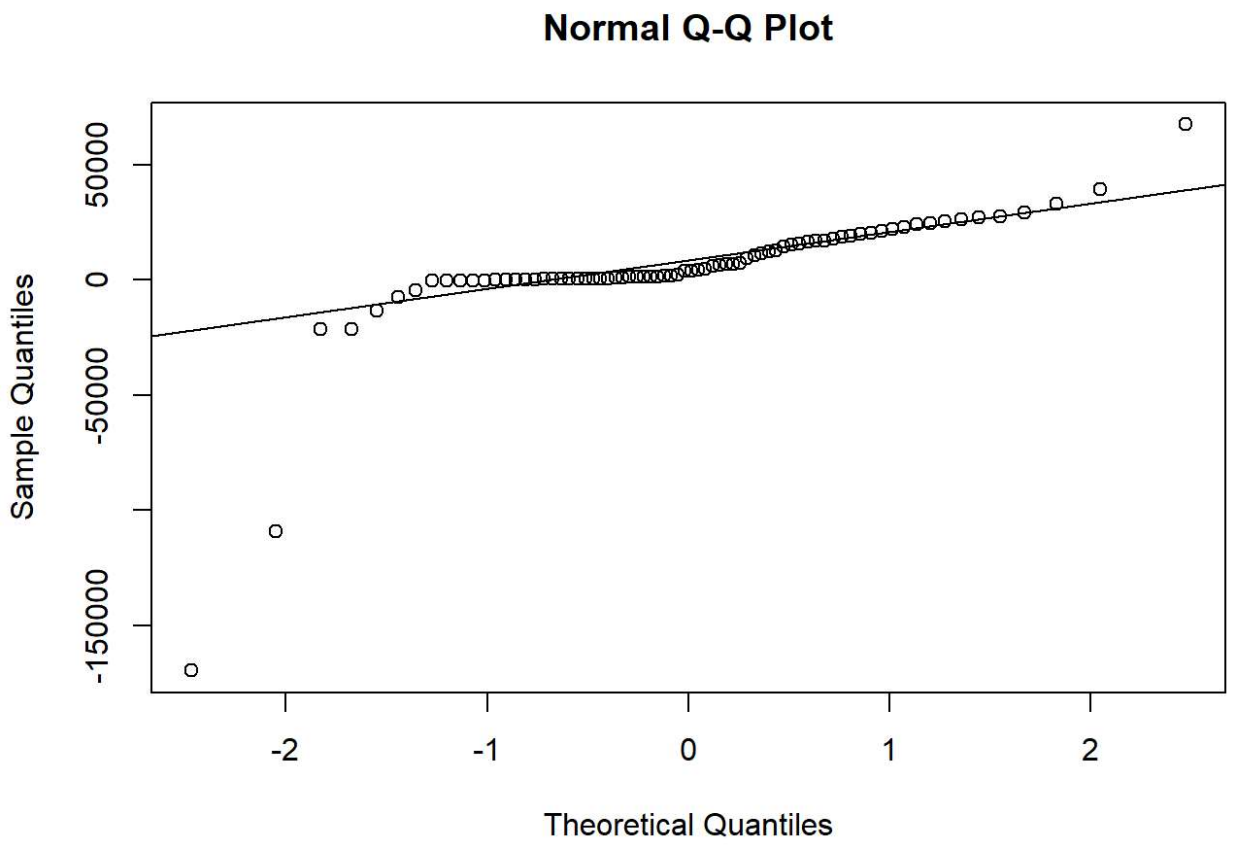
```
##
##  Augmented Dickey-Fuller Test
##
## data:  Res_pq21
## Dickey-Fuller = -4.4727, Lag order = 4, p-value = 0.01
## alternative hypothesis: stationary
```

H0: Time series is not a stationary, H1: Time series is a stationary

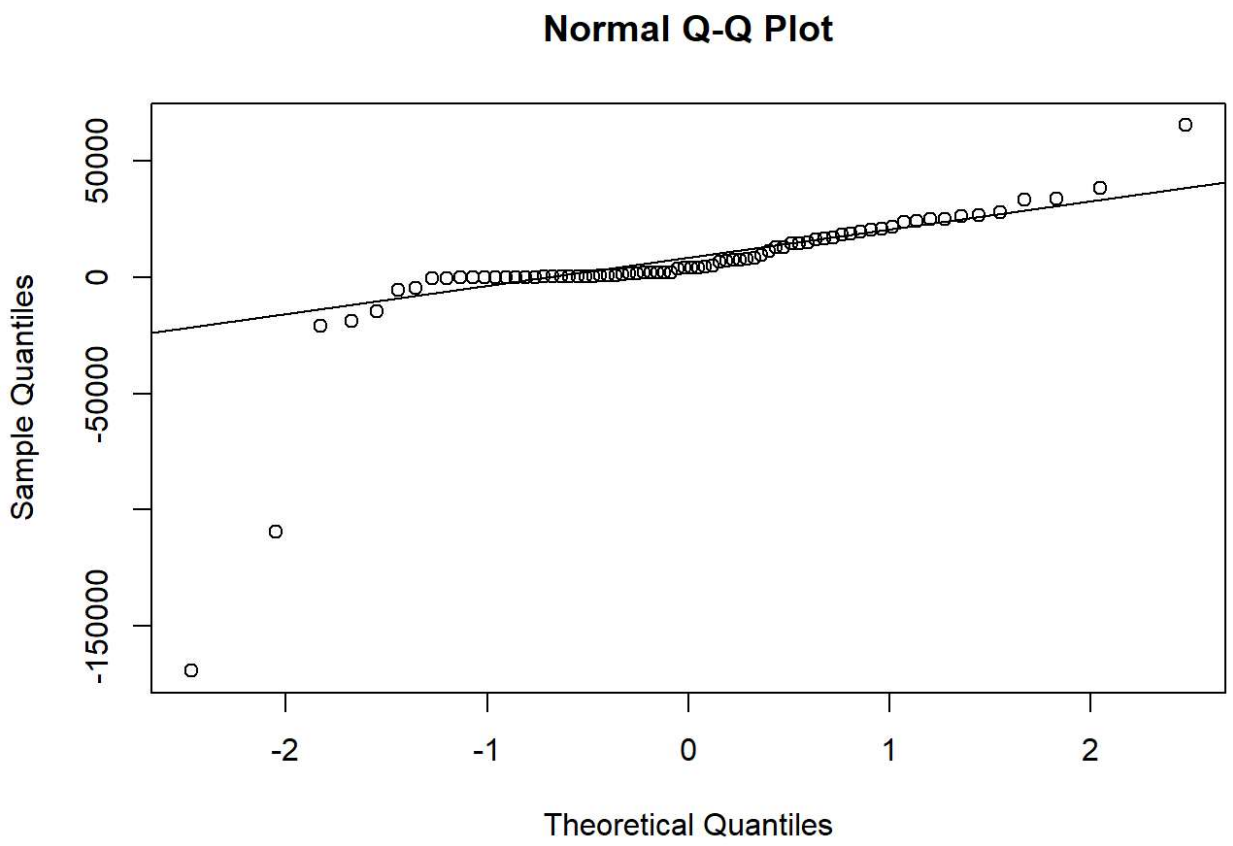
The p-value of both ADF test are below 0.01, which is significant at the 5% level. Hence, the null hypothesis should be rejected, since there is enough evidence in favor of alternative hypothesis (H1) and conclude that the time series is a stationary. This confirms the residuals are stationary or white noise.

1.14.5 Normality of the Residuals - QQ plots

```
qqnorm(residuals(pq11)); qqline(residuals(pq11)) # ARIMA(1,2,1)
```




```
qqnorm(residuals(pq21)); qqline(residuals(pq21)) # ARIMA(2,2,1)
```



Above plots illustrate that the quantiles of the residuals is close to the normal distribution in general.However, few deviations are noticable at the edge of the line. Hence, A statistical test has been performed to check the normality of the distribution.

1.14.6 Statistical tests - Normality of the Residuals

```
shapiro.test(rstandard(pq11)) # ARIMA(1,2,1)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  rstandard(pq11)
## W = 0.57999, p-value = 3.152e-13
```

```
shapiro.test(rstandard(pq21)) # ARIMA(2,2,1)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  rstandard(pq21)
## W = 0.57583, p-value = 2.692e-13
```

H0: Data is normally distributed

Shapiro-Wilk test of normality has a test statistic p-value is lesser than 0.01 and HO should be rejected. Hence, the residuals are not normally distributed. However, the reseduals do not indicate a pattern but distributed in randomly. Because, it is random part that cannot model. Hence, assume it has a normal distribution, but random.

1.15 Checking for Overfitness

Over fitting exercise was carried out by adding more parameters to evaluate whether the selected model is overfitting. As such, two parameter has added to our model as below.

```
pq11=arima(annual_data,order=c(1,2,1),method="ML") #Best model ARIMA(1,2,2)
pq11
```

```
##
## Call:
## arima(x = annual_data, order = c(1, 2, 1), method = "ML")
##
## Coefficients:
##           ar1      ma1
##       -0.3516  -0.7999
## s.e.   0.1440   0.0721
##
## sigma^2 estimated as 830713046:  log likelihood = -842.35,  aic = 1688.7
```

```
pq31=arima(annual_data,order=c(3,2,1),method="ML") #new model with extra parameter
pq31
```

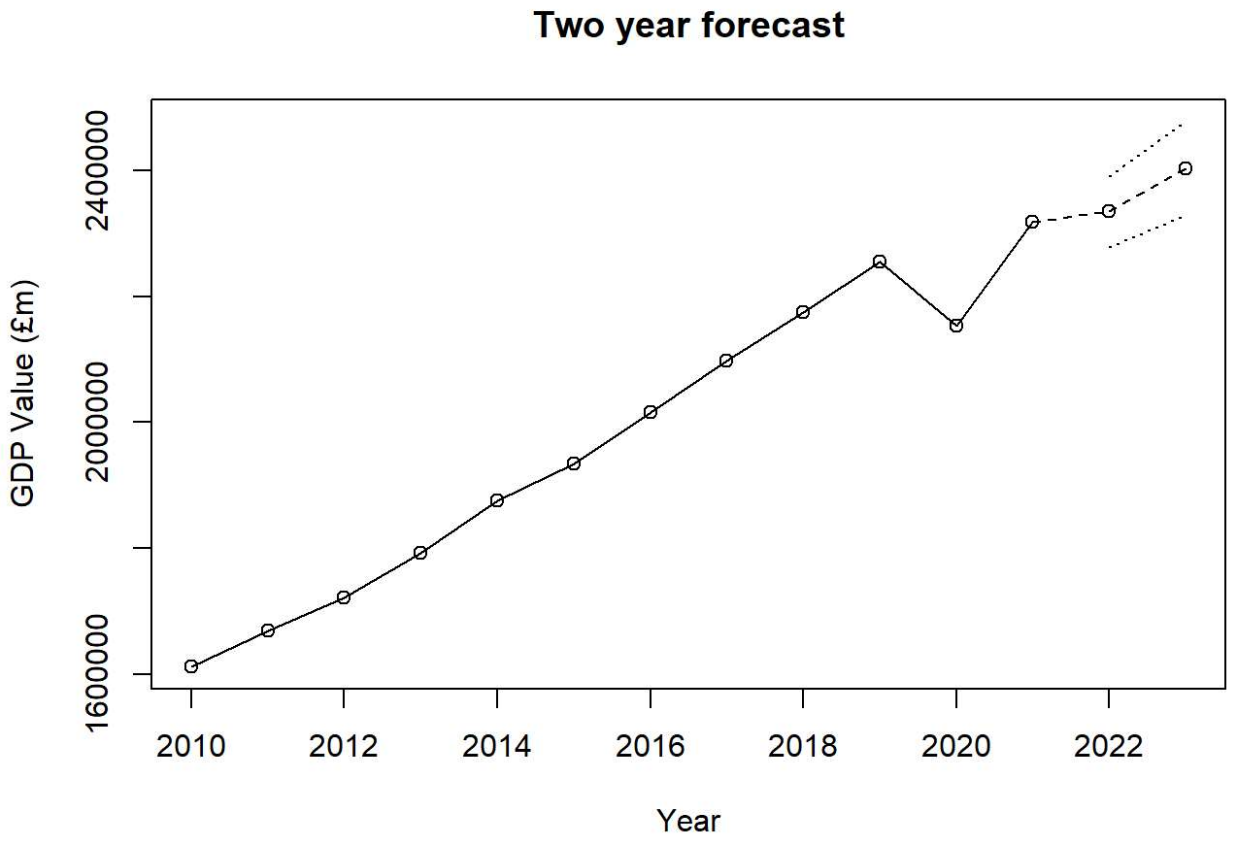
```
##
## Call:
## arima(x = annual_data, order = c(3, 2, 1), method = "ML")
##
## Coefficients:
##           ar1     ar2      ar3      ma1
##       -0.3380  0.1595  -0.0225  -0.8235
## s.e.   0.1448  0.2209   0.2084   0.0792
##
## sigma^2 estimated as 823924016:  log likelihood = -842.08,  aic = 1692.17
```

New model (pq31) has four parameters (ar1, ar2, ar3 and ma1) and the estimation for extra parameter is 0.1595, and -0.0225 which is very close to zero. Hence, adding a new parameter (ar3) is insignificant and made the model ineffective. Further, the AIC value for the new model (pq31) is 1692.17 which is slightly higher than the original model ‘pq11’. Hence, It provide a sign that the selected model is a overfitted model.

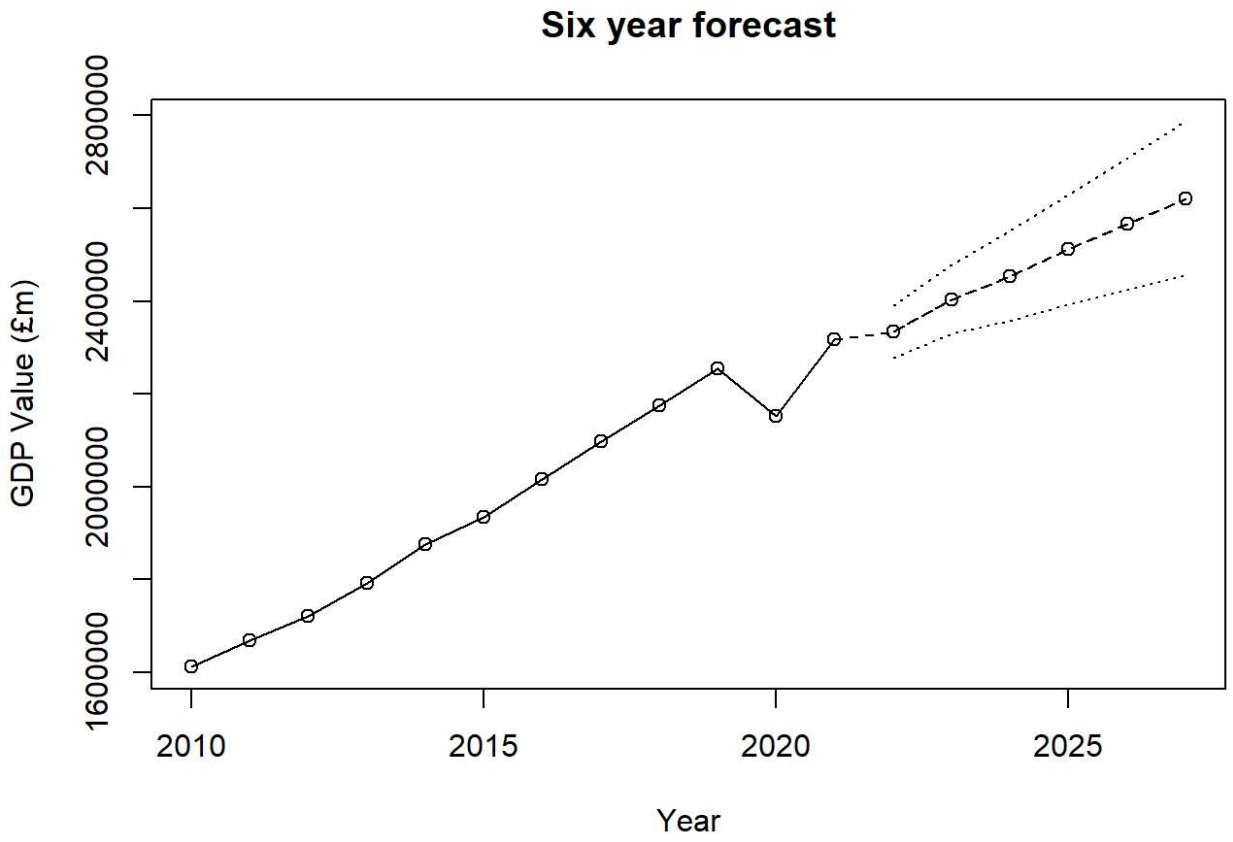
1.16 Forecasting of the model

Forecasting for the model pp0201 (ARIMA(1,2,1)) as below.

```
plot(pq11,n1=c(2010),n.ahead=2,xlab="Year",type="o", ylab="GDP Value (£m)",  main = "Two year forec
ast" )
```




```
plot(pq11,n1=c(2010),n.ahead=6, xlab="Year",type="o", ylab="GDP Value (£m)",  main = "Six year fore
cast" )
```



Forecasting was done for next 2 and 4 years respectively. It could examine that the forecasted values of the model (ARIMA(1,2,1) is similar to the pattern of the previous years irrespective the random drop in 2020. Hence, model was fitted to forecast the values.

The time plot for next 4 years indicate that the interval of confidence increases as the period increases which leads predictions become less precise. Hence, forecasting for quite lengthy periods tend to have increased interval of confidence.

1.17 Forecast errors

A forecast “error” is the difference between an observed value and its forecast. Accuracy of the model prediction depends on the length of the trained data. Hence, the data set used for this analysis has quite lengthy period of past data starting from 1948 to 2021. Train and test data was allocated based on the 80:20.

```
# train and test data
train_data1 <-window(annual_data,end=c(2011))
test_data1  <- window(annual_data,start=c(2012))


pq11_train<-arima(train_data1,order=c(1,2,1),method="ML") # pq11 - ARIMA(1,2,1)
pq21_train<-arima(train_data1,order=c(2,2,1),method="ML") # pq21 - ARIMA(2,2,1)
```

```
# predict.Arima: Forecast from ARIMA fits
predict_pq11 <- predict(pq11_train,n.ahead = 10)
accuracy(predict_pq11$pred, test_data1)
```

```
##                ME     RMSE      MAE      MPE     MAPE      ACF1 Theil's U
## Test set 101078.6 120132.8 101078.6 4.744706 4.744706 0.3179994   1.32507
```

```
predict_pq21 <- predict(pq21_train,n.ahead = 10)
accuracy(predict_pq21$pred, test_data1)
```

```
##                ME     RMSE      MAE      MPE     MAPE      ACF1 Theil's U
## Test set 114294.3 133759.1 114294.3 5.373024 5.373024 0.3598498  1.475783
```

In general, above two test results of forecast errors for the two models namely pq11 and pq21. The lesser values for forecast error is given by the pq21 model which is ARIMA(2,2,1). Hence, based on the analysis of forecast errors, it assume ARIMA(2,2,1) model as the best medel out of the tested models.

1.18 Prediction Comparrisson

```
#Values of fitted model
test_data1
```

```
## Time Series:
## Start = 2012
## End = 2021
## Frequency = 1
##  [1] 1721355 1793155 1876162 1935212 2016638 2097143 2174380 2255283 2152646
## [10] 2317667
```

```
predict_pq11$pred
```

```
## Time Series:
## Start = 2012
## End = 2021
## Frequency = 1
##  [1] 1719457 1767424 1814857 1862147 1909398 1956639 2003877 2051115 2098352
## [10] 2145589
```

```
# Comparison among test and forecast data
a <- test_data1
b <- predict_pq11$pred
c <- data.frame(a,b)

names(c)<- c("Test data", "Forecats")
c
```

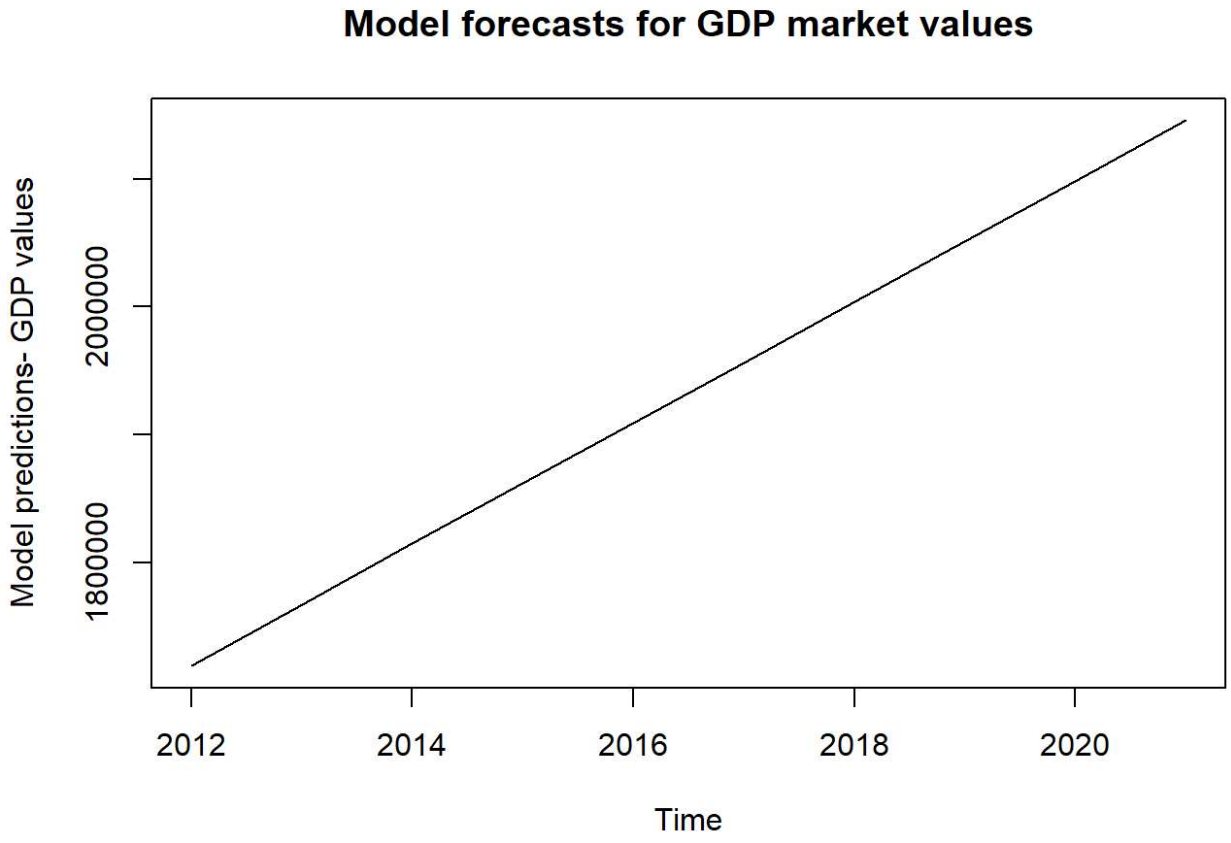
```
##    Test data Forecats
## 1    1721355  1719457
## 2    1793155  1767424
## 3    1876162  1814857
## 4    1935212  1862147
## 5    2016638  1909398
## 6    2097143  1956639
## 7    2174380  2003877
## 8    2255283  2051115
## 9    2152646  2098352
## 10   2317667  2145589
```

Above output illustrate, the values for the test data and ARIMA(2,2,1) model predictions for test data for the period of 2015 to 2021. As such, there the predicted values of the fitted model is closer to the actual values.

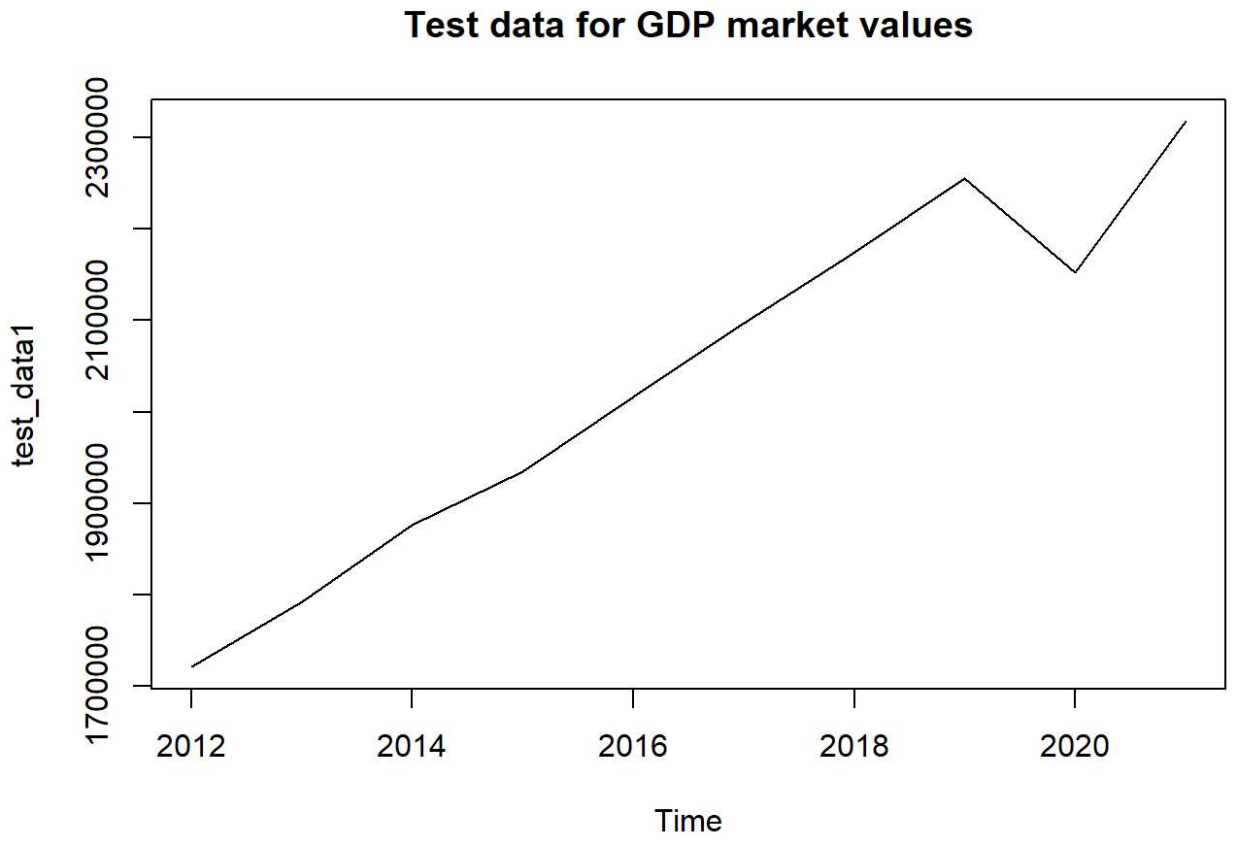
Further, below plots illustrate the patterns relevant to the above data.

1.19 Plotting predictions

```
#Ploting
plot(predict_pq11$pred, type = "l",  ylab="Model predictions- GDP values" , main = "Model forecasts
for GDP market values" )
```




```
plot(test_data1, main = "Test data for GDP market values" )
```



1.20 Conclusion:

Time series for number of annual GDP at market prices ((£m) of UK was analysed in this study. As such, time series was convert to a stationary time series, develop list of possible models, model fitting and residual analysis was performed. Finally, ARIMA(2,2,1) was fitted to forecast the Gross Domestic Product at market prices. Later, time series was segregated to train and test data based on 80:20 proportion. Then, ARIMA(2,2,1) was tested for train and test data set and evaluate the prediction criteria. As such, the model follows the similar pattern to that of past data and predicted values closer to the test value set.

Limitations and future developments:

This model can be improved by application of some machine learning prediction algorithms to predict the values almost similar er to the test values. Further, this data set has monitory values and should be adjusted for inflation which has not captured in this model development. Hence, another model could be develop to track the inflationary impact over the GDP overtime.

#APPLICATION OF SEASONAL ARIMA FOR A MONTHLY DATA SET

##Task 1 – Exploratory Data Analysis

The data set provided with this assignment is called `UK: Women Employment Fulltime: Aged 16 and over (Thousands): NSA', which is basically about the statistics on fulltime employed women in UK during the period of 1993 to 2020. The monthly data available in this data set has been used in the analysis below.

Source :
(https://www.ons.gov.uk/employmentandlabourmarket/peopleinwork/employmentandemployeetypes/timeseries/i46h/lms (https://www.ons.gov.uk/employmentandlabourmarket/peopleinwork/employmentandemployeetypes/timeseries/i46h/lms))

1.1 Import the file saved locally in the computer by using read.csv from base R.The first seven raw's of the data set has been eliminated since those consist with the Meta data, annual data and quarterly data.

```
m_data <- read.csv("D:/Desktop/USW/Data Mining/cw2/employment-fulltime-women.csv", skip=164)
```

1.2 View of imported data

```
#View structure of imported data
str(m_data)
```

```
## 'data.frame':    347 obs. of  2 variables:
##  $ X1992.DEC: chr  "1993 JAN" "1993 FEB" "1993 MAR" "1993 APR" ...
##  $ X6417    : int  6383 6367 6372 6376 6344 6378 6419 6459 6457 6389 ...
```

This data set includes 347 observations and 2 variables.

```
# View first 5 observations (Monthly)
m_data %>% head(5)
```

```
##   X1992.DEC X6417
## 1  1993 JAN  6383
## 2  1993 FEB  6367
## 3  1993 MAR  6372
## 4  1993 APR  6376
## 5  1993 MAY  6344
```

1.3 Selection of annual data

```
#Select only for monthly data and transpose it to a vector
monthly_data <- as.vector(t(m_data[1:336,2]))
monthly_data
```

```
##   [1] 6383 6367 6372 6376 6344 6378 6419 6459 6457 6389 6401 6356 6360 6343 6355
##  [16] 6362 6359 6400 6441 6464 6450 6452 6449 6420 6438 6456 6462 6452 6456 6499
##  [31] 6535 6564 6572 6520 6517 6494 6485 6433 6456 6485 6504 6515 6606 6636 6661
##  [46] 6642 6615 6588 6596 6616 6614 6619 6624 6665 6705 6723 6718 6723 6734 6688
##  [61] 6664 6670 6692 6675 6674 6767 6840 6886 6895 6852 6844 6831 6836 6829 6838
##  [76] 6839 6874 6905 6988 7039 7017 7009 6988 6962 6912 6905 6918 6933 6954 6987
##  [91] 7052 7035 7037 6997 7002 7009 6997 6977 7026 7031 7056 7086 7179 7180 7195
## [106] 7179 7171 7120 7118 7141 7172 7163 7157 7176 7217 7256 7252 7273 7287 7257
## [121] 7223 7209 7187 7184 7188 7237 7263 7281 7308 7290 7276 7281 7245 7251 7264
## [136] 7277 7307 7324 7374 7433 7472 7494 7527 7561 7608 7596 7567 7545 7534 7601
## [151] 7688 7726 7722 7686 7703 7692 7697 7714 7706 7716 7711 7752 7788 7772 7741
## [166] 7718 7704 7681 7676 7694 7695 7725 7735 7800 7833 7895 7898 7881 7894 7892
## [181] 7912 7894 7885 7929 7922 7987 7994 8017 8014 7913 7935 7940 7884 7830 7825
## [196] 7819 7786 7783 7846 7811 7785 7775 7787 7761 7746 7711 7677 7667 7693 7702
## [211] 7720 7728 7722 7690 7702 7702 7717 7730 7692 7712 7748 7809 7820 7807 7802
## [226] 7776 7768 7751 7721 7722 7745 7743 7750 7775 7834 7813 7790 7829 7899 7922
## [241] 7887 7895 7934 7892 7923 7980 8049 8046 8079 8110 8170 8139 8132 8149 8187
## [256] 8189 8148 8173 8296 8302 8328 8317 8344 8344 8321 8332 8380 8339 8361 8352
## [271] 8443 8457 8475 8488 8505 8459 8432 8440 8430 8498 8518 8533 8579 8610 8655
## [286] 8660 8690 8640 8669 8721 8746 8750 8740 8769 8803 8812 8830 8825 8828 8850
## [301] 8809 8859 8868 8926 8916 8938 9017 9029 9036 9023 9057 9060 9045 9051 9118
## [316] 9108 9116 9119 9160 9176 9188 9261 9315 9348 9292 9241 9230 9271 9298 9342
## [331] 9437 9500 9550 9569 9555 9590
```

```
class(m_data)
```

```
## [1] "data.frame"
```

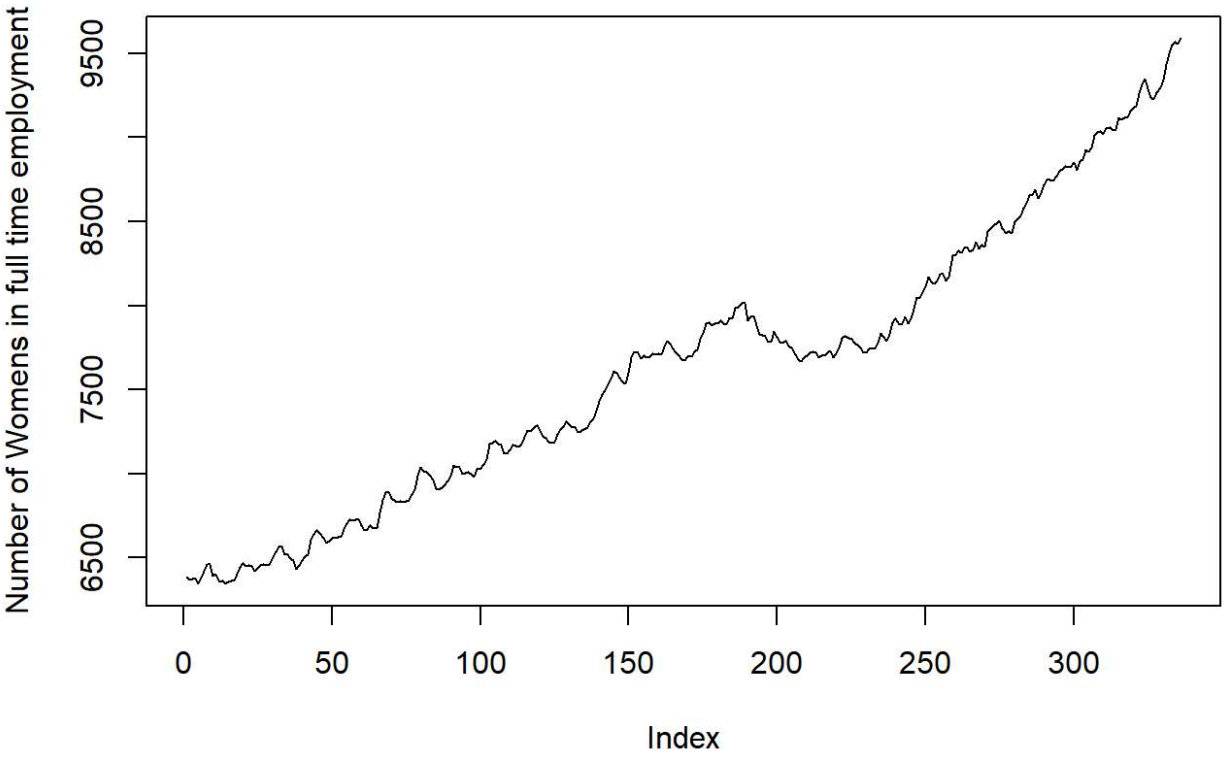
```
class(monthly_data)
```

```
## [1] "integer"
```

Above output indicate that the women fulltime employment statistics of UK for each month from January 1993 up to December 2020.

1.4 Plotting

```
# Plotting monthly_data vector
monthly_data %>% plot(type="l", ylab= "Number of Womens in full time employment")
```



1.5 Treatment to outliers and missing values

Missing values and outliers have not been examined in the data set.

1.6 Transformation

Transformation mechanisms have not been used since the above time series doesn't indicate abnormal variations and pattern. Hence, Boxcox and log transformations were not used.

1.6 Creation of time series object

```
# Time series object
monthly_data %>% ts(frequency = 12, start = c(1993,1)) -> monthly_data # frequency 12 => Monthly Da
ta
monthly_data %>% head (24)
```

```
##       Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec
## 1993 6383 6367 6372 6376 6344 6378 6419 6459 6457 6389 6401 6356
## 1994 6360 6343 6355 6362 6359 6400 6441 6464 6450 6452 6449 6420
```

```
class(monthly_data)
```

```
## [1] "ts"
```

```
#time series object details
class(monthly_data)
```

```
## [1] "ts"
```

```
start(monthly_data)
```

```
## [1] 1993    1
```

```
end(monthly_data)
```

```
## [1] 2020   12
```

1.7 Summary statistics

```
#Summary statistics
summary(monthly_data)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    6343    6973    7705    7637    8141    9590
```

As per the above output data, the average employment count is 7637 (’Thousands), while minimum is 6343 (’Thousands) and maximum is 9590 (’Thousands).

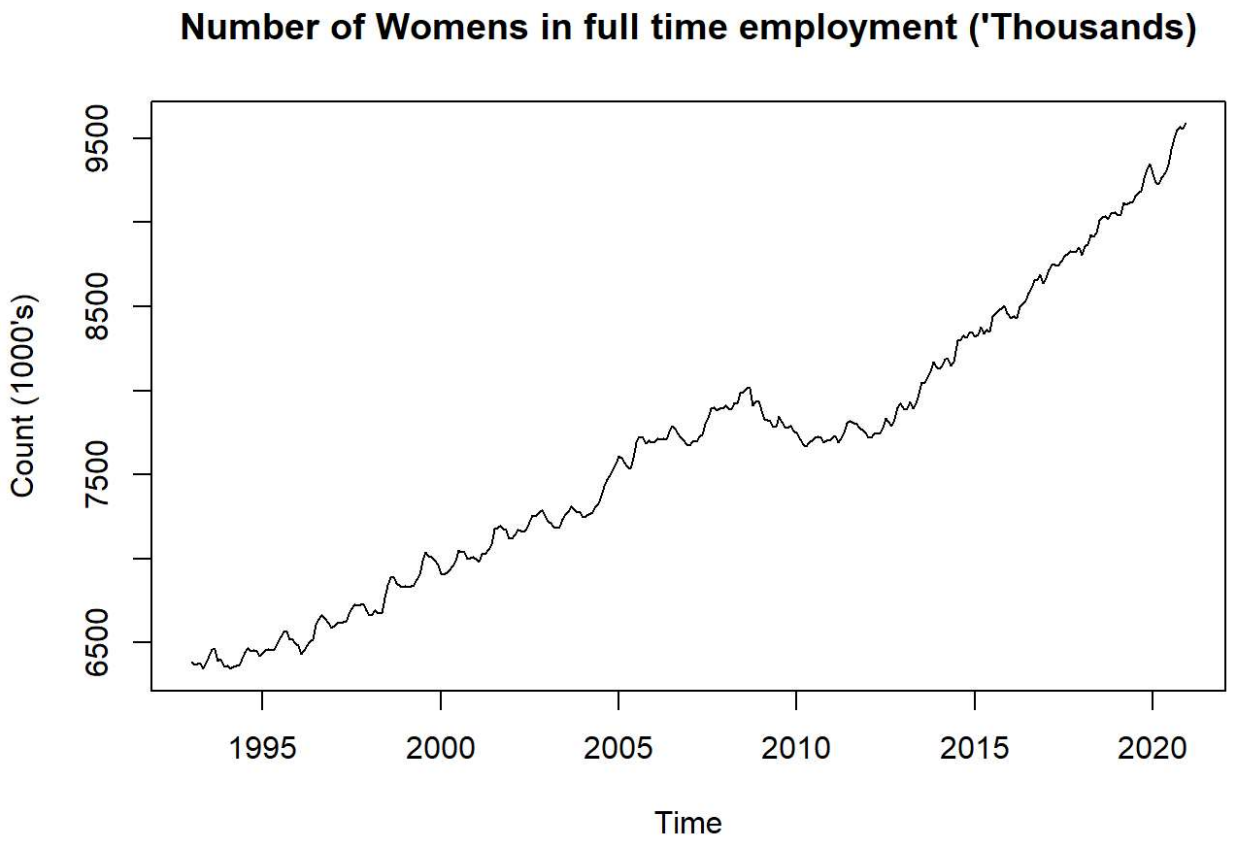
```
#Provides cycle across years
cycle(monthly_data)
```

```
##      Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 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   9  10  11  12
## 1996   1   2   3   4   5   6   7   8   9  10  11  12
## 1997   1   2   3   4   5   6   7   8   9  10  11  12
## 1998   1   2   3   4   5   6   7   8   9  10  11  12
## 1999   1   2   3   4   5   6   7   8   9  10  11  12
## 2000   1   2   3   4   5   6   7   8   9  10  11  12
## 2001   1   2   3   4   5   6   7   8   9  10  11  12
## 2002   1   2   3   4   5   6   7   8   9  10  11  12
## 2003   1   2   3   4   5   6   7   8   9  10  11  12
## 2004   1   2   3   4   5   6   7   8   9  10  11  12
## 2005   1   2   3   4   5   6   7   8   9  10  11  12
## 2006   1   2   3   4   5   6   7   8   9  10  11  12
## 2007   1   2   3   4   5   6   7   8   9  10  11  12
## 2008   1   2   3   4   5   6   7   8   9  10  11  12
## 2009   1   2   3   4   5   6   7   8   9  10  11  12
## 2010   1   2   3   4   5   6   7   8   9  10  11  12
## 2011   1   2   3   4   5   6   7   8   9  10  11  12
## 2012   1   2   3   4   5   6   7   8   9  10  11  12
## 2013   1   2   3   4   5   6   7   8   9  10  11  12
## 2014   1   2   3   4   5   6   7   8   9  10  11  12
## 2015   1   2   3   4   5   6   7   8   9  10  11  12
## 2016   1   2   3   4   5   6   7   8   9  10  11  12
## 2017   1   2   3   4   5   6   7   8   9  10  11  12
## 2018   1   2   3   4   5   6   7   8   9  10  11  12
## 2019   1   2   3   4   5   6   7   8   9  10  11  12
## 2020   1   2   3   4   5   6   7   8   9  10  11  12
```

Original time series can be express as a function that explain the trend, seasonality effect and a stationary component(random component) as white noise. Hence, it was performed an analysis below to identify the trend and seasonality components in the created time series object simply by plotting the original time series data as below.

1.8 Time series object visualisation

```
# Plotting time series object
ts.plot(window(monthly_data, start =c(1993,1)),  ylab="Count (1000's)", main="Number of Womens in f
ull time employment ('Thousands)")
```

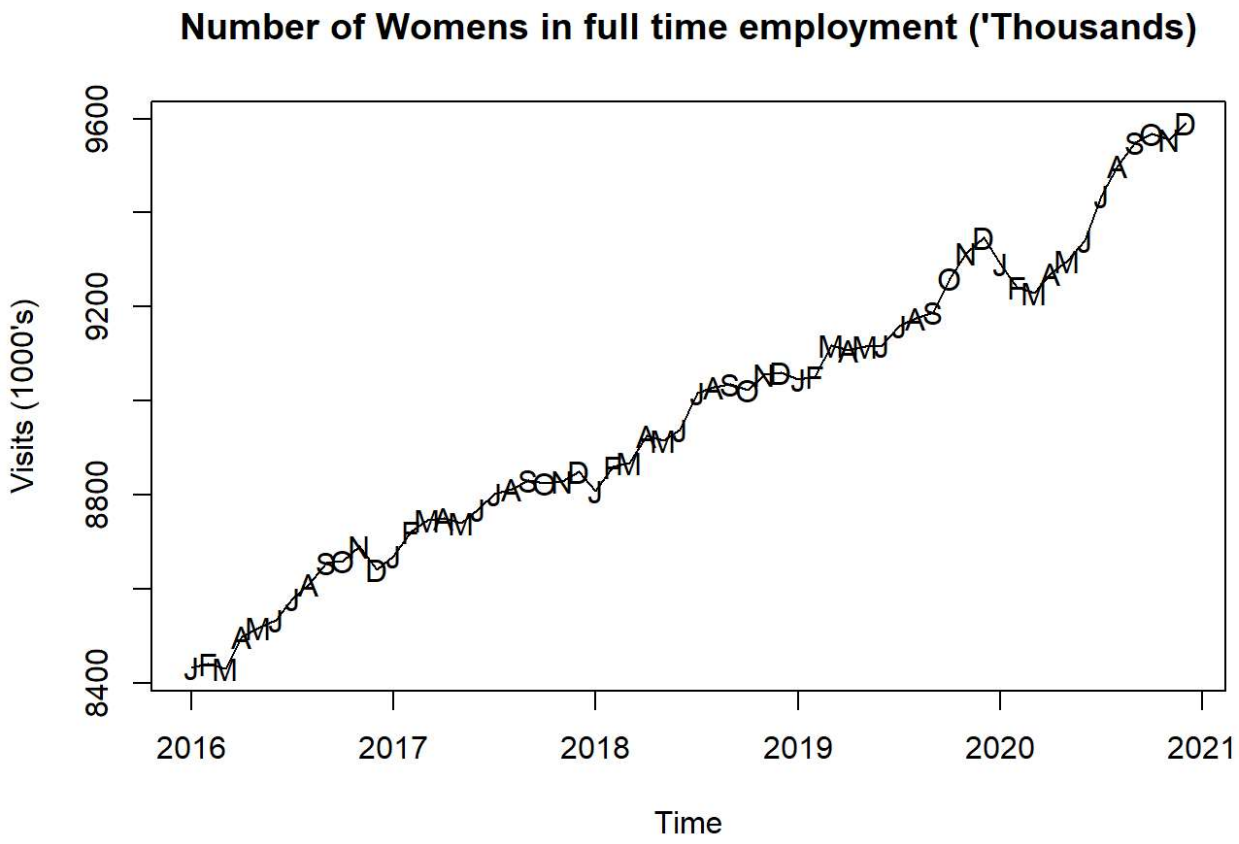


Above Time plot illustrate followings;

There is a increasing trend (upward trend) as the time goes up, the number of visits also growing up in general. The time series displays a some regular pattern. This provides a sense of seasonality. However, further analysis has been performed to study the seasonal effect in below.

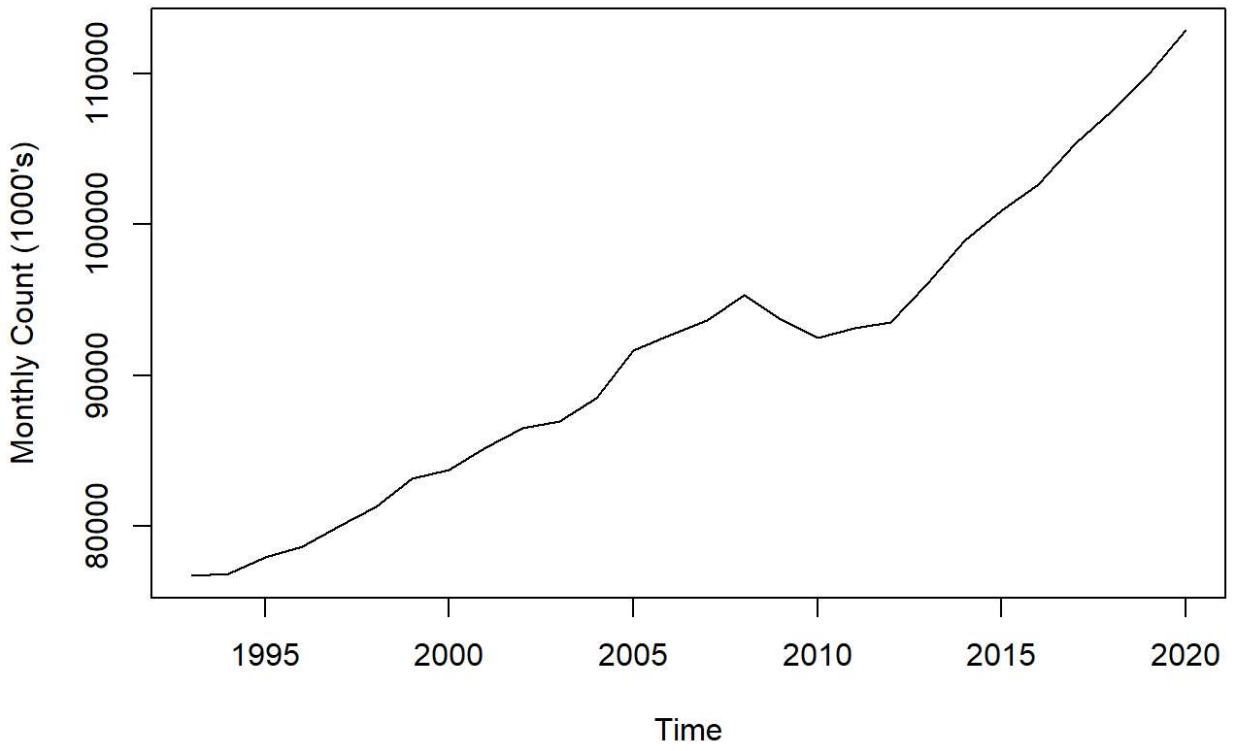
The variance (size of employment count) is not increasing as the time (level of the series) increases. Hence, Box-Cox transformation has not been applied to this time series. All the peaks and drops are not in same size, as there is some viability. As such, it can assume that the Time series variance is not constant over the time.

```
# Plotting a portion of time series object for a recent time slot.
ts.plot(window(monthly_data, start =c(2016,1)), ylab="Visits (1000's)", main="Number of Womens in f
ull time employment ('Thousands)")
qtr=c("J","F","M","A","M","J","J","A","S","O","N","D")
points(window(monthly_data,start=c(2016,1)),pch=qtr)
```



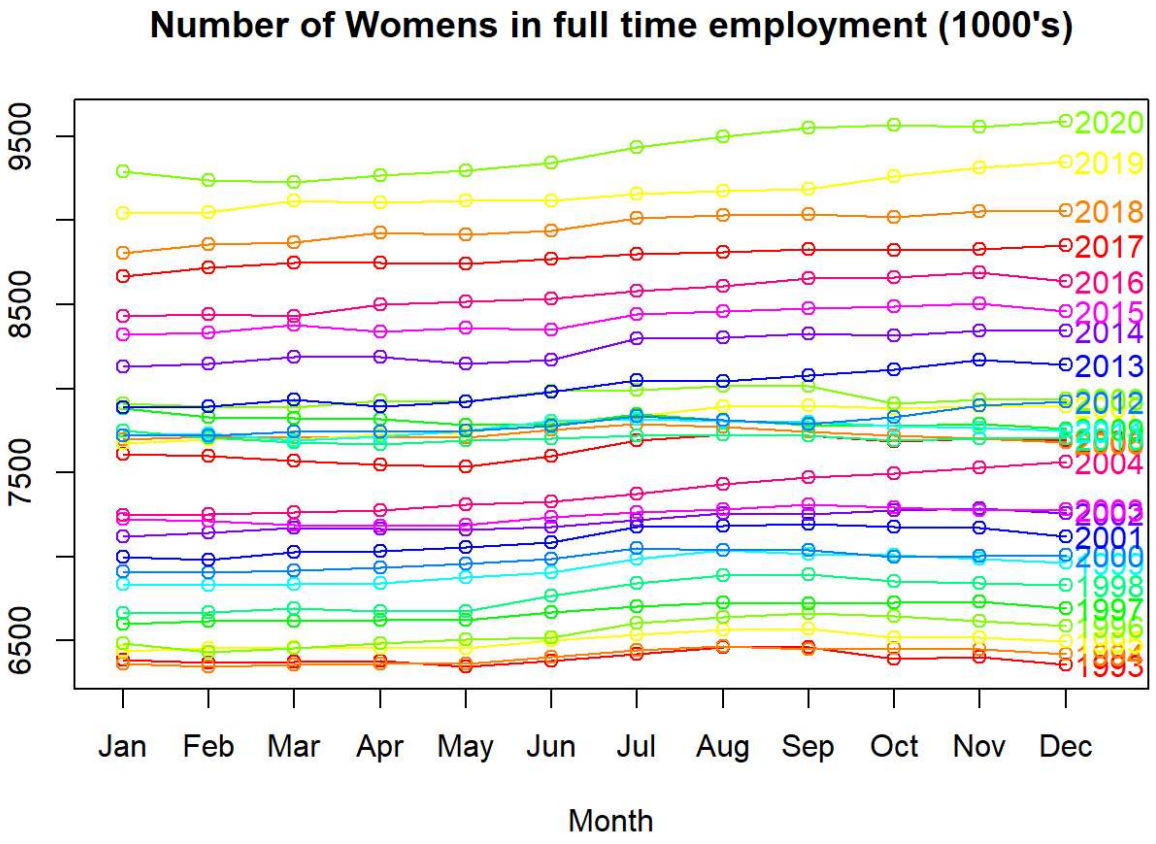
Time series object has been plotted for a portion of recent time slots to see a seasonal pattern of the time series. Above time plot indicates a regularly pattern of highs and lows within the year. Theres a slight drop in each January month and a increase in November and December months. As such, it provides a sense of seasonality component associated with this time series.

```
# Plotting for aggregated data
plot(aggregate(monthly_data), ylab="Monthly Count (1000's)")
```



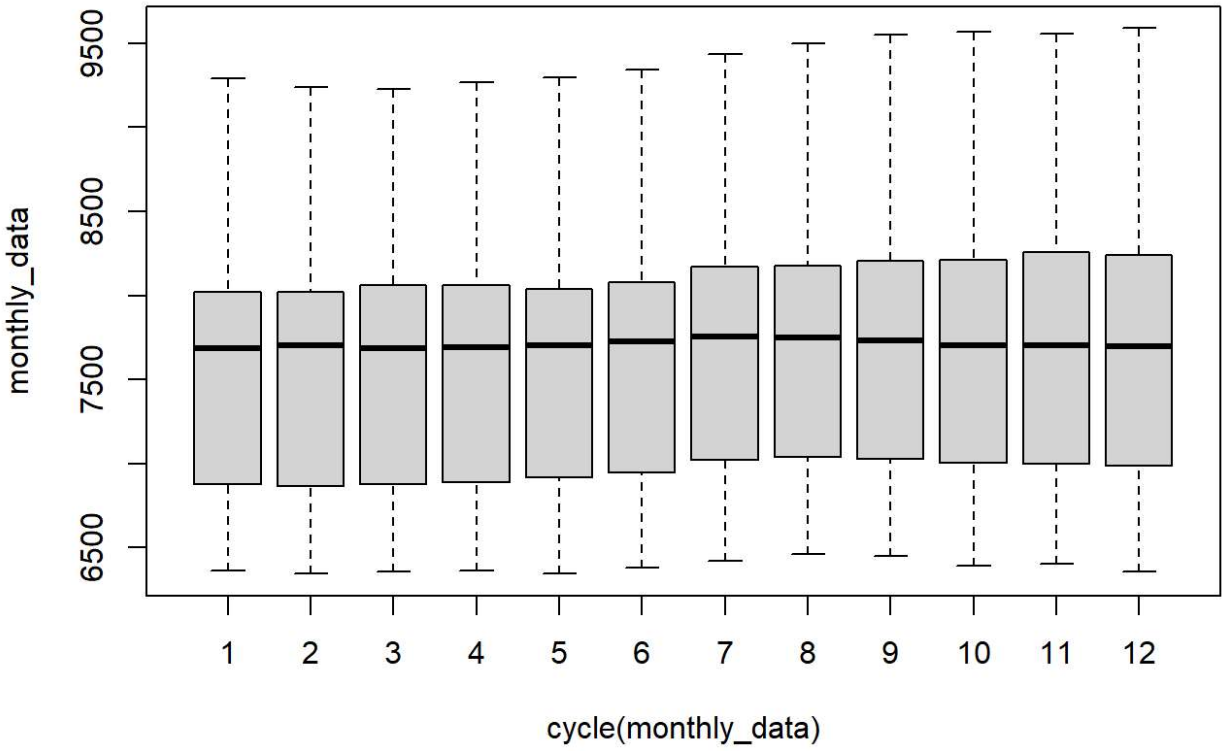
The aggregate (annual) data was plotted to see the trend pattern of annual data and which indicates a upward trend as the time goes up the visits also growing up.

```
#Plot for sense of seasonality
seasonplot(monthly_data, col=rainbow(12), year.labels=TRUE, main="Number of Womens in full time emp
loyment (1000's)")
```



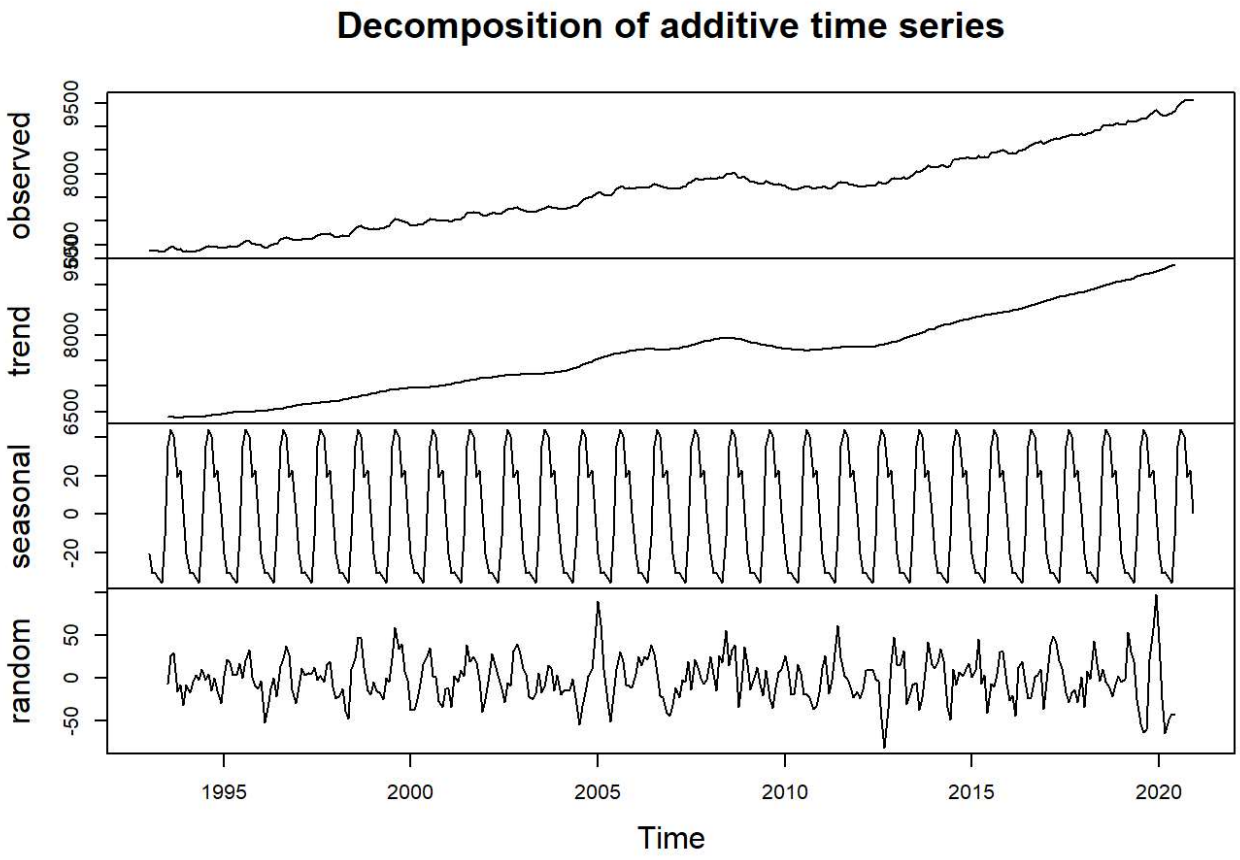
Above seasonal plot indicate the underlying seasonal pattern more clearly. There is no massive ups and downs between months. However, gradual increase in December is visible at the latter part of the period of consideration.

```
#Boxplot for cycles
boxplot(monthly_data ~cycle(monthly_data))
```



The Boxplot is a form of plot enables the underlying seasonal pattern to be seen clearly. The above Boxplot indicates the positions in the cycle of four frequencies. There could observe a slight pattern when observing the median line of the Boxplots which at the beginning of the year tend to decrease. Then at the middle of the year which correspond to summer time there is a increase. Further, the range has been increase gradually from January to December.

```
# Application of additive model to decompose the time series
Qtr_decom <- decompose(monthly_data, type="additive")
plot(Qtr_decom)
```



Additive model was used as the time series variance (size of the visits) is not increasing as the time (level of the series) increases by (variance volatility is not increasing with the time).Decomposition plot illustrate the trend, seasonal and random component of the time series separately in a graphical manner as above. The underlying time series is a non-stationary time series as it has a trend or a seasonal component and requires differencing to transform it to a stationary time series.

Task 2 – Model fitting and Forecasting

Stationary time series is devoid of trend,seasonal patterns and white noise. A time series is said to be stationary if it holds the following conditions true;

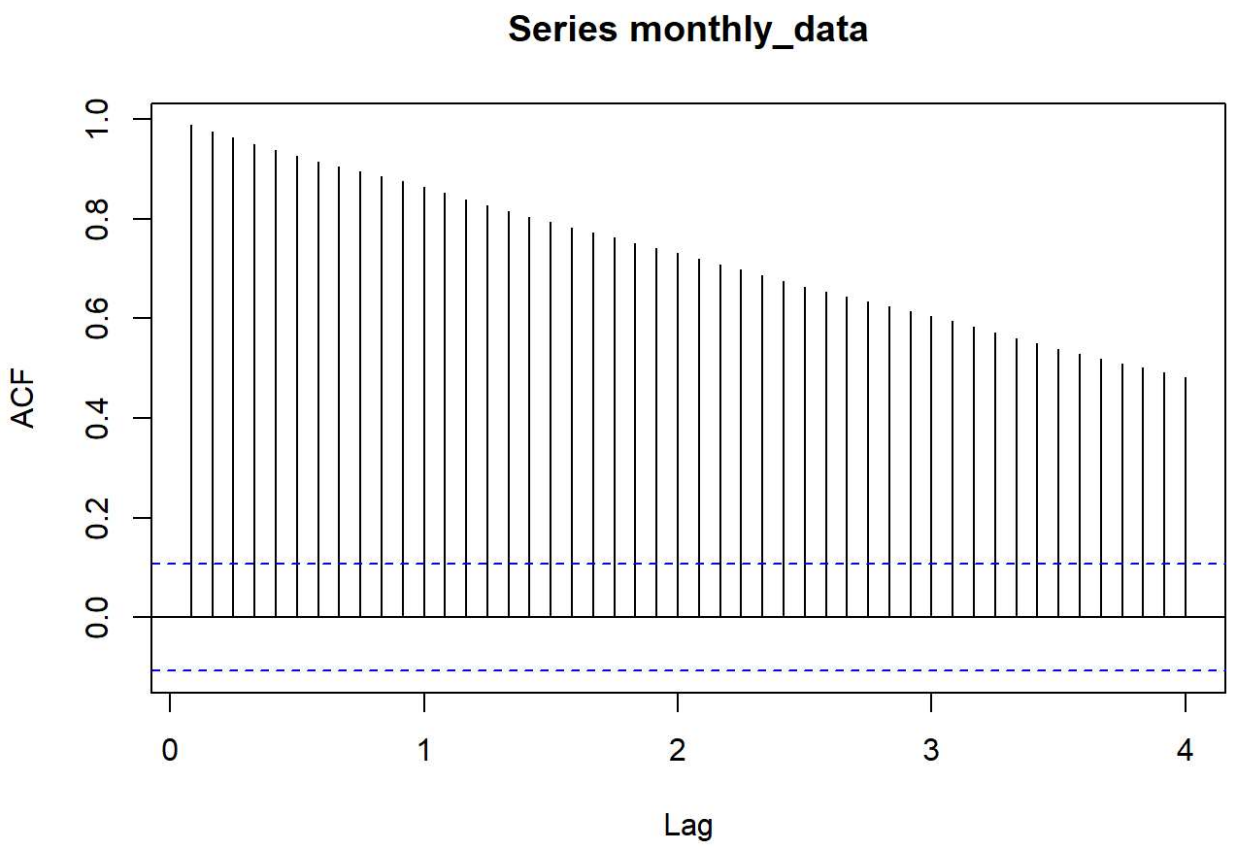
The the expected value (mean value) of time-series is constant over time (which implies, the trend component is nullified) variance is constant which oscillate around a constant value, and the correlation depends on the time lag, but not the absolute.

It is important to note that the process defined in this way is weakly stationary. ACF plot and hypothesis testing (ADF & KPSS) was performed to identify that the time series of Overseas visits to UK is stationary or not.

1.9.1 Checking for stationary - Autocorrelation function (ACF)

The autocorrelation coefficients are plotted to show the autocorrelation function or ACF. ACF plots observe if the data has a trend and a seasonal component. The plot is also known as a correlogram which is in below.

```
#Autocorrelation function for original time series
acf(monthly_data, lag.max=48)
```



Above plot of ACF indicates significant positive many autocorrelations. The autocorrelations decreases slowly as the number of lags increases. The autocorrelations are gradually decreasing after the every peak as the lags increases. A combination of these effects could be seen, when data are both trended and seasonal.

Hence, ACF indicates trended time series. In a stationary time series, the ACF will drop to zero relatively quickly as it doesn't depends on time, while the ACF of non-stationary data decreases slowly as in the above plot. Further, more than 5% of spikes are outside the bounds. As such, the above ACF indicates the attributes of non-stationary time series or not a white noise.

Further, it is also useful to quantify the evidence of non-stationarity via hypothesis testing. Hence, the Dickey -Fuller test and KPSS tests has been used in below.

1.9.2 Checking for stationary - Augmented Dickey-Fuller unit-root test (ADF)

```
#Dickey-Fuller test
adf.test(monthly_data)
```

```
## Warning in adf.test(monthly_data): p-value greater than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  monthly_data
## Dickey-Fuller = 0.2368, Lag order = 6, p-value = 0.99
## alternative hypothesis: stationary
```

H0: Time series is not a stationary, H1: Time series is a stationary

The p-value of above ADF test is 0.99, which is not significant at the 5% level. Hence, the null hypothesis should not be rejected, since that there is enough evidence to suggest that there the time series is not stationary.

1.9.3 Checking for stationary - Kwiatkowski-Phillips-SchmidtShin (KPSS) test

```
#KPSS test
kpss.test(monthly_data)
```

```
## Warning in kpss.test(monthly_data): p-value smaller than printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  monthly_data
## KPSS Level = 5.351, Truncation lag parameter = 5, p-value = 0.01
```

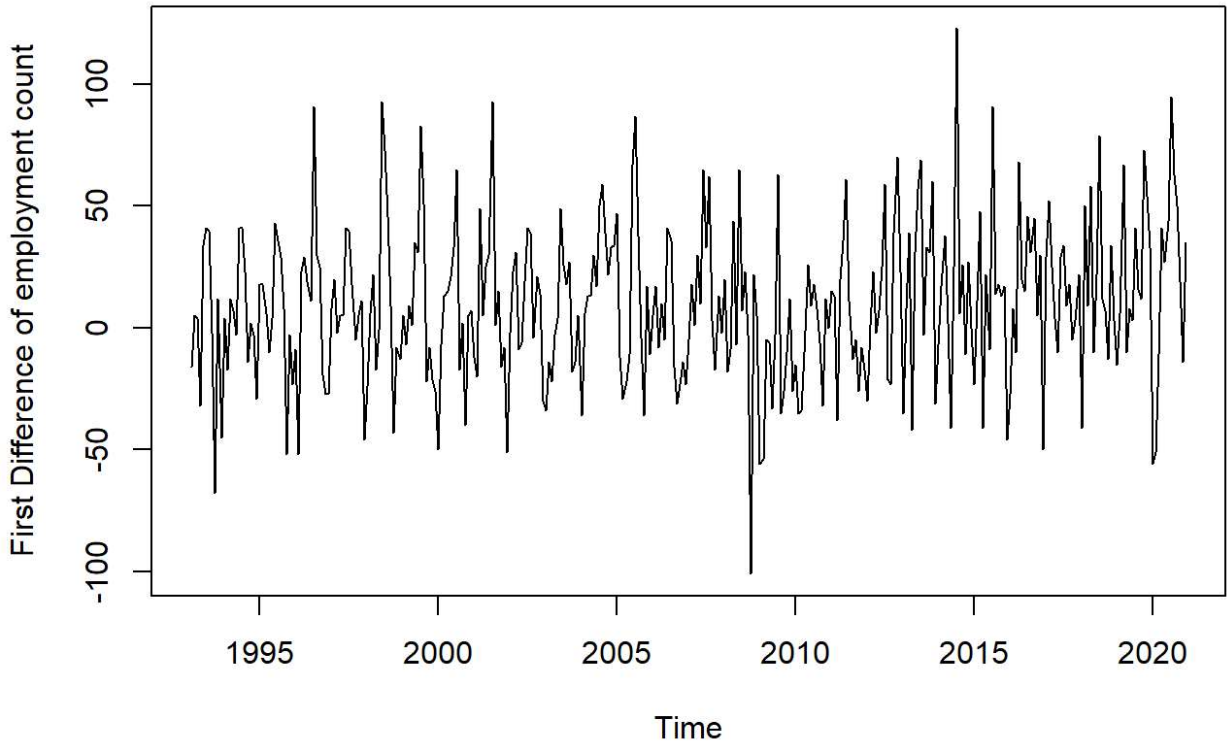
H0: Time series is a stationary, H1: Time series is not a stationary

The p-value of the KPSS test is 0.01, which is significant at the 5% level. Hence, the null hypothesis should be rejected, since there is enough evidence in favor of alternative hypothesis (H1) and conclude that the time series is not a stationary.

It is apparent that the above time series is a non-stationary since it has a trend and seasonal component.Hence, it is required to convert to a stationary time series to construct a appropriate model for forecasting.Decompose the series into the components trend, seasonal effect, and residuals, and plot the decomposed series in below.

1.10 Stationary through Differencing

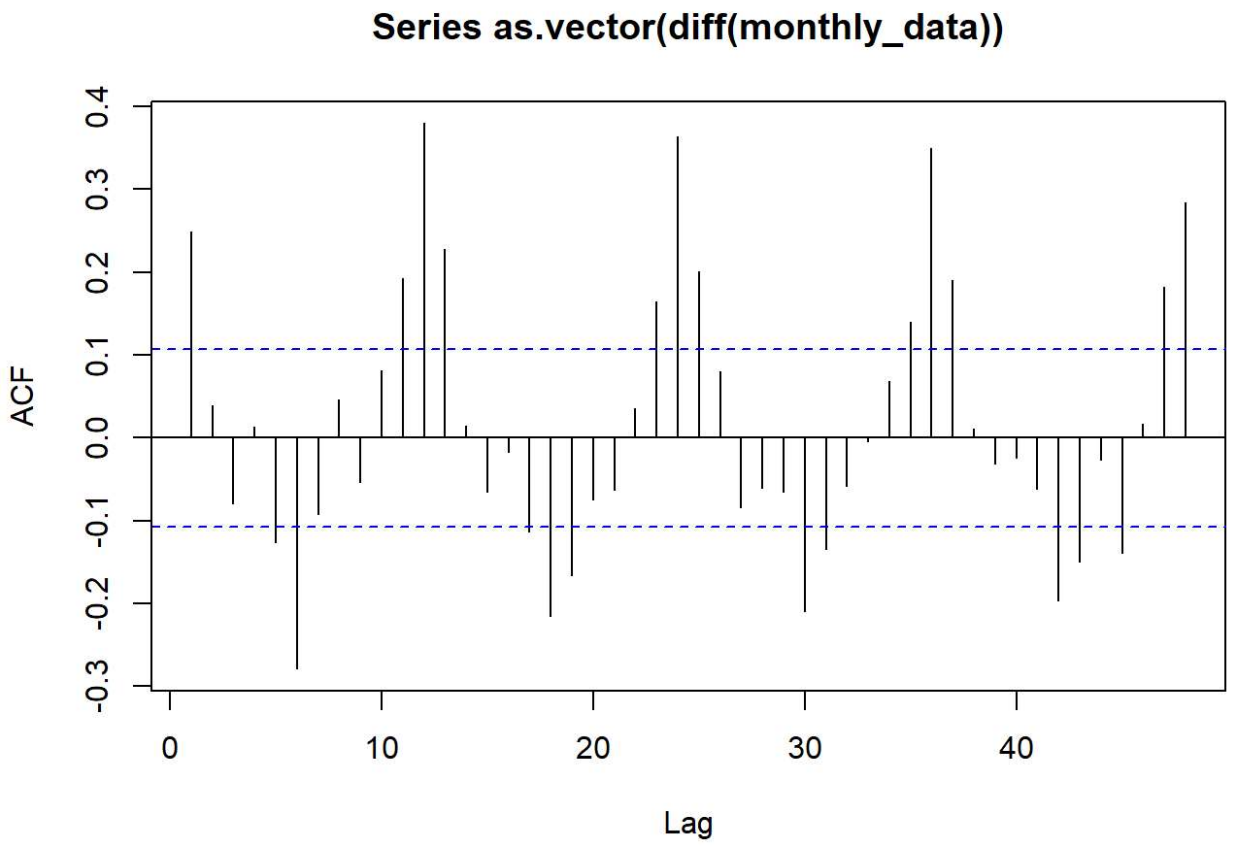
```
#First difference
plot(diff(monthly_data),ylab="First Difference of employment count",xlab="Time")
```



A time plot after the first difference is illustrate above. In this case, it is clear that the trend component has been eliminated from the series and values oscillate around a constant level. Then, seasonal differencing is required to covert the time series to a stationary one or a white noise.

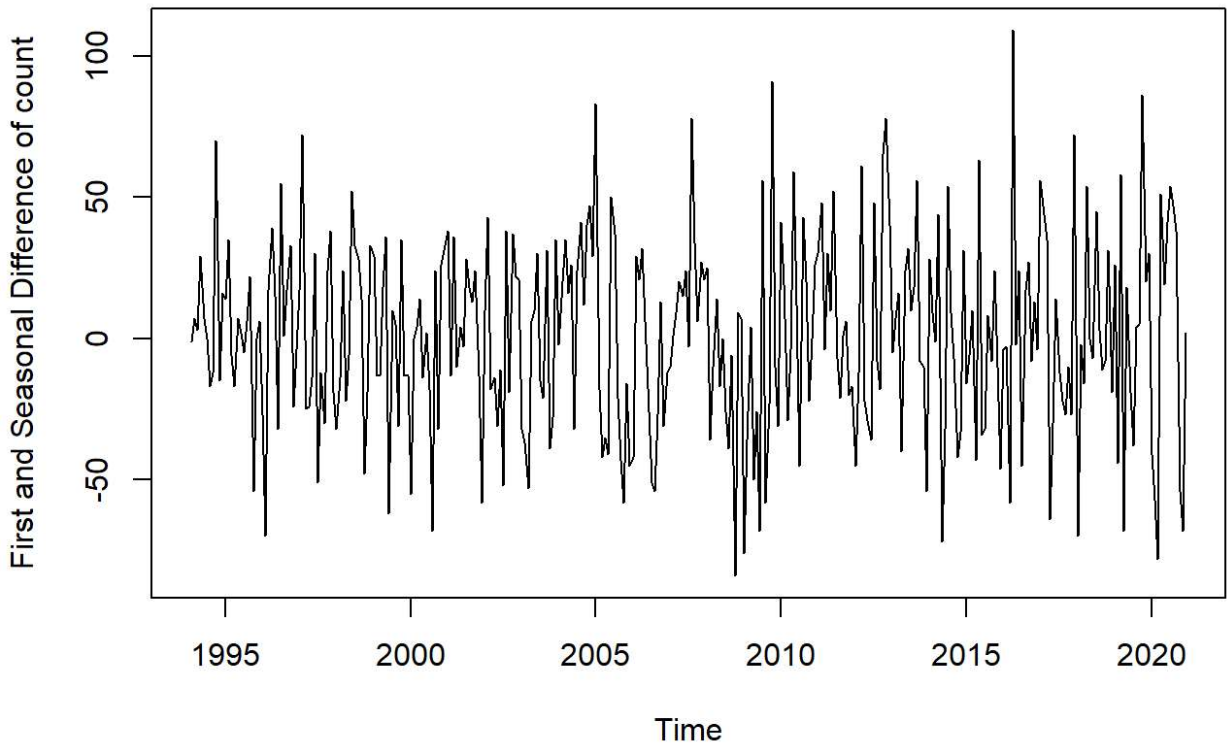
-

```
#Autocorrelation function for the first Difference
acf(as.vector(diff(monthly_data)),lag.max=48)
```



Above ACF plot indicates autocorrelation function for the first Difference. There are many significant spikes with a pattern. Hence, second difference is applied to the time series to remove the seasonal effect.

```
# First and Seasonal Difference
plot(diff(diff(monthly_data),lag=12),xlab="Time", ylab="First and Seasonal Difference of count")
```



The above figure displays the time plot after application of both first difference and seasonal difference.As such, there is no trend component, since the values are oscillate around a constant level. Further, a random pattern is visible. Accordingly, both the trend and seasonal component has been eliminated from the time series. Further, the evidence of non-stationarity could be quantify via hypothesis testing. Thus, Dickey -Fuller test has been applied below.

Checking for stationary - Augmented Dickey-Fuller unit-root test (ADF)

```
#Dickey-Fuller test
adf.test(diff(diff(monthly_data),lag=4))
```

```
## Warning in adf.test(diff(diff(monthly_data), lag = 4)): p-value smaller than
## printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff(diff(monthly_data), lag = 4)
## Dickey-Fuller = -12.062, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

H0: Time series is not a stationary, H1: Time series is a stationary

The p-value of above ADF test is 0.01, which is significant at the 5% level. Hence, the null hypothesis should be rejected, since there is enough evidence in favor of alternative hypothesis (H1) and conclude that the time series is a stationary.

-

```
#KPSS test
kpss.test(diff(diff(monthly_data),lag=4))
```

```
## Warning in kpss.test(diff(diff(monthly_data), lag = 4)): p-value greater than
## printed p-value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  diff(diff(monthly_data), lag = 4)
## KPSS Level = 0.0075099, Truncation lag parameter = 5, p-value = 0.1
```

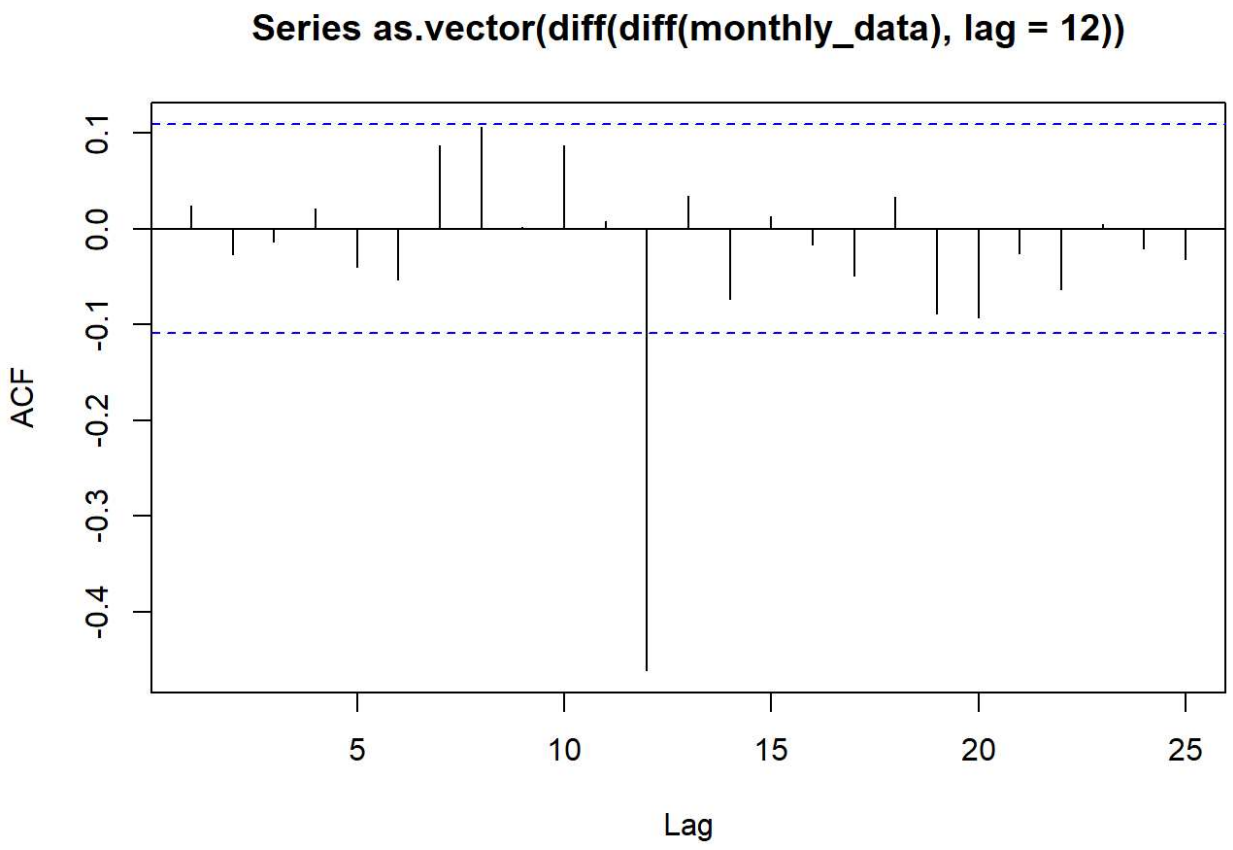
H0: Time series is a stationary, H1: Time series is not a stationary

The p-value of above KPSS test is 0.1, which is not significant at the 5% level. Hence, the null hypothesis should not be rejected, since that there is enough evidence to suggest that there the time series is a stationary.

1.11 Model Specification

The following ACF, PACF and EACF plots have been analysed to determine the parameters initially.

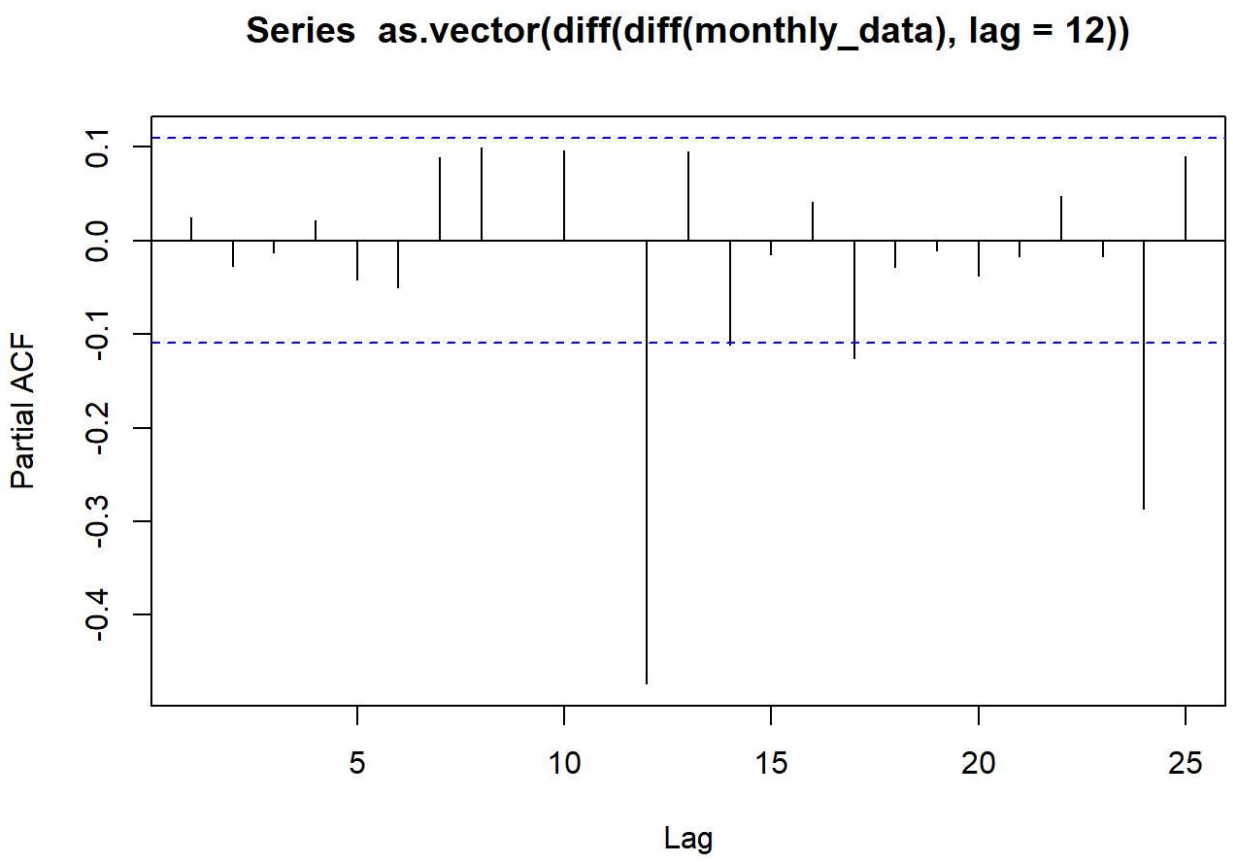
```
#Autocorrelation function stationary time series
acf(as.vector(diff(diff(monthly_data),lag=12)))
```



Above ACF function illustrate that approximately 95% of the spikes are within the interval of confidence. However, there is one significant autocorrelations are out from the interval of confidence. The spikes at lag 12 which is significantly different from zero and rest of all lags are zero or closer to zero. Hence, it could assume that the time series is a is a stationary.

Above ACF plot has been used to determine moving average (MA) component of the model specification. As such, ACF illustrate both a non-seasonal moving average component (q) and seasonal moving average component (Q). Hence, it could assume 'q' as 0 or 1, and 'Q' as 0. The first difference (d) was set as 1.

```
#Partial Autocorrelation function stationary time series
pacf(as.vector(diff(diff(monthly_data),lag=12)))
```



PACF function indicates that very few significant autocorrelations are out from the interval of confidence. However, it can assume that the 95% is within the interval of confidence. The spikes at lag 12 and 24 is significantly different from zero and rest of all lags are zero or closer to zero. Hence, it could assume that the time series is a is a stationary.

PACF plot illustrate autoregressive (AR) component of the of model specification. The time series in this analysis consist with both non-seasonal autoregressive component (p) and seasonal autoregressive component (Q). Significant Peaks tend to be four lags apart. Hence, by examining the above PACF plot,it could assume 'p' as 0 or 1, and 'P' as 1 or 2. The seasonal difference (D) was set as 1.

```
#eacf for stationary time series
eacf(as.vector(diff(diff(monthly_data))),ar.max=10, ma.max=10)
```

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

By examining the EACF plot to determine a appropriate model is complicated as it does not have a clear visible pattern.

1.12 Parameter Estimation

Seasonal ARIMA(p,d,q)×(P,D,Q)s model would be appropriate to apply for the time series in this analysis where nonseasonal orders (p, d, q),seasonal orders (P, D, and Q) and seasonal period s. Thus, ACF, PACF plots and AIC, BIC criteria has been used in the process of model specification & parameter estimation of the model.

```
#List of possible models and Parameter estimation
m1 =arima(monthly_data,order=c(1,1,1),seasonal=list(order=c(1,1,0), period=12))
m1
```

```
##
## Call:
## arima(x = monthly_data, order = c(1, 1, 1), seasonal = list(order = c(1, 1,
##     0), period = 12))
##
## Coefficients:
##           ar1     ma1     sar1
##       -0.1917  0.2561  -0.4927
## s.e.   0.4581  0.4493   0.0495
##
## sigma^2 estimated as 902.9:  log likelihood = -1559.09,  aic = 3124.18
```

```
m2=arima(monthly_data,order=c(0,1,0),seasonal=list(order=c(1,1,0), period=12))
m2
```

```
##
## Call:
## arima(x = monthly_data, order = c(0, 1, 0), seasonal = list(order = c(1, 1,
##     0), period = 12))
##
## Coefficients:
##          sar1
##       -0.4896
## s.e.   0.0496
##
## sigma^2 estimated as 906.9:  log likelihood = -1559.78,  aic = 3121.56
```

```
m3=arima(monthly_data,order=c(1,1,0),seasonal=list(order=c(1,1,0), period=12))
m3
```

```
##
## Call:
## arima(x = monthly_data, order = c(1, 1, 0), seasonal = list(order = c(1, 1,
##     0), period = 12))
##
## Coefficients:
##          ar1     sar1
##       0.0591  -0.4922
## s.e.  0.0556   0.0495
##
## sigma^2 estimated as 903.6:  log likelihood = -1559.21,  aic = 3122.43
```

```
m4=arima(monthly_data,order=c(0,1,1),seasonal=list(order=c(1,1,0), period=12))
m4
```

```
##
## Call:
## arima(x = monthly_data, order = c(0, 1, 1), seasonal = list(order = c(1, 1,
##     0), period = 12))
##
## Coefficients:
##          ma1     sar1
##       0.0639  -0.4925
## s.e.  0.0575   0.0495
##
## sigma^2 estimated as 903.3:  log likelihood = -1559.17,  aic = 3122.34
```

```
m5=arima(monthly_data,order=c(1,1,2),seasonal=list(order=c(1,1,2), period=12))
m5
```

```
##
## Call:
## arima(x = monthly_data, order = c(1, 1, 2), seasonal = list(order = c(1, 1,
##     2), period = 12))
##
## Coefficients:
##           ar1     ma1     ma2     sar1     sma1     sma2
##       -0.6634  0.7625  0.1071  -0.7509  -0.0976  -0.6776
## s.e.   0.4556  0.4490  0.0617   0.4768   0.4676   0.4002
##
## sigma^2 estimated as 696.8:  log likelihood = -1524.07,  aic = 3060.14
```

```
m6=arima(monthly_data,order=c(1,1,2),seasonal=list(order=c(1,1,1), period=12))
m6
```

```
##
## Call:
## arima(x = monthly_data, order = c(1, 1, 2), seasonal = list(order = c(1, 1,
##     1), period = 12))
##
## Coefficients:
##           ar1     ma1     ma2    sar1     sma1
##       -0.6532  0.7515  0.1043  0.0207  -0.8754
## s.e.   0.4841  0.4780  0.0633  0.0700   0.0479
##
## sigma^2 estimated as 697.3:  log likelihood = -1524.2,  aic = 3058.41
```

```
m7=arima(monthly_data,order=c(1,1,1),seasonal=list(order=c(1,1,2), period=12))
m7
```

```
##
## Call:
## arima(x = monthly_data, order = c(1, 1, 1), seasonal = list(order = c(1, 1,
##     2), period = 12))
##
## Coefficients:
##          ar1     ma1     sar1     sma1     sma2
##       0.0832  0.0082  -0.7418  -0.1048  -0.6685
## s.e.  0.4258  0.4256   0.5126   0.5019   0.4281
##
## sigma^2 estimated as 699.2:  log likelihood = -1524.54,  aic = 3059.09
```

```
m8=arima(monthly_data,order=c(0,1,1),seasonal=list(order=c(1,1,2), period=12))
m8
```

```
##
## Call:
## arima(x = monthly_data, order = c(0, 1, 1), seasonal = list(order = c(1, 1,
##     2), period = 12))
##
## Coefficients:
##          ma1     sar1     sma1     sma2
##       0.0895  -0.7386  -0.1074  -0.6654
## s.e.  0.0548   0.5136   0.5034   0.4294
##
## sigma^2 estimated as 699.3:  log likelihood = -1524.56,  aic = 3057.13
```

```
m9=arima(monthly_data,order=c(0,1,2),seasonal=list(order=c(1,1,1), period=12))
m9
```

```
##
## Call:
## arima(x = monthly_data, order = c(0, 1, 2), seasonal = list(order = c(1, 1,
##     1), period = 12))
##
## Coefficients:
##          ma1    ma2    sar1     sma1
##       0.0919  0.014  0.0284  -0.8771
## s.e.  0.0571  0.053  0.0695   0.0478
##
## sigma^2 estimated as 699.1:  log likelihood = -1524.62,  aic = 3057.23
```

As per the above output results,the lowest AIC value which is 3057.13 is provide by “m8” model and then the second lowest AIC value of 3057.23 for “m9”. Hence, it can assume ‘mpq1121’ as the best model based on the AIC value and the second best as ‘m8’. The attributes of the two models are describe below.

```
#AIC lowest models in details

m8
```

```
##
## Call:
## arima(x = monthly_data, order = c(0, 1, 1), seasonal = list(order = c(1, 1,
##     2), period = 12))
##
## Coefficients:
##          ma1     sar1     sma1     sma2
##       0.0895  -0.7386  -0.1074  -0.6654
## s.e.  0.0548   0.5136   0.5034   0.4294
##
## sigma^2 estimated as 699.3:  log likelihood = -1524.56,  aic = 3057.13
```

```
m9
```

```
##
## Call:
## arima(x = monthly_data, order = c(0, 1, 2), seasonal = list(order = c(1, 1,
##     1), period = 12))
##
## Coefficients:
##          ma1    ma2    sar1     sma1
##       0.0919  0.014  0.0284  -0.8771
## s.e.  0.0571  0.053  0.0695   0.0478
##
## sigma^2 estimated as 699.1:  log likelihood = -1524.62,  aic = 3057.23
```

Model attributes comparison :

The lowest AIC value of 3057.13 is given by ‘m8’ model which has 4 parameters and the coefficients of parameters (ma1, sar1, sma1,sma2) is 0.0895, -0.7386, -0.1074 and -0.6654 respectively. Square error also quite small for all the parameters.

The second lowest AIC value of 3057.23 is given by ‘m9’ which also has 4 parameters and the coefficients of parameters (ma1, ma2, sar1, sma1) is 0.0919, 0.014, 0.0284 and -0.8771 respectively.

As such, two models (m8 and m9) has been specified as below.

Best model (m8) out of the evaluated list of models : Seasonal ARIMA(0,1,1)×(1,1,2)12 Second best model (m9) out of the evaluated list of models : Seasonal ARIMA(0,1,2)×(1,1,1)12

```
#Model evaluation-BIC criteria
BIC(m8)
```

```
## [1] 3078.016
```

```
BIC(m9)
```

```
## [1] 3078.123
```

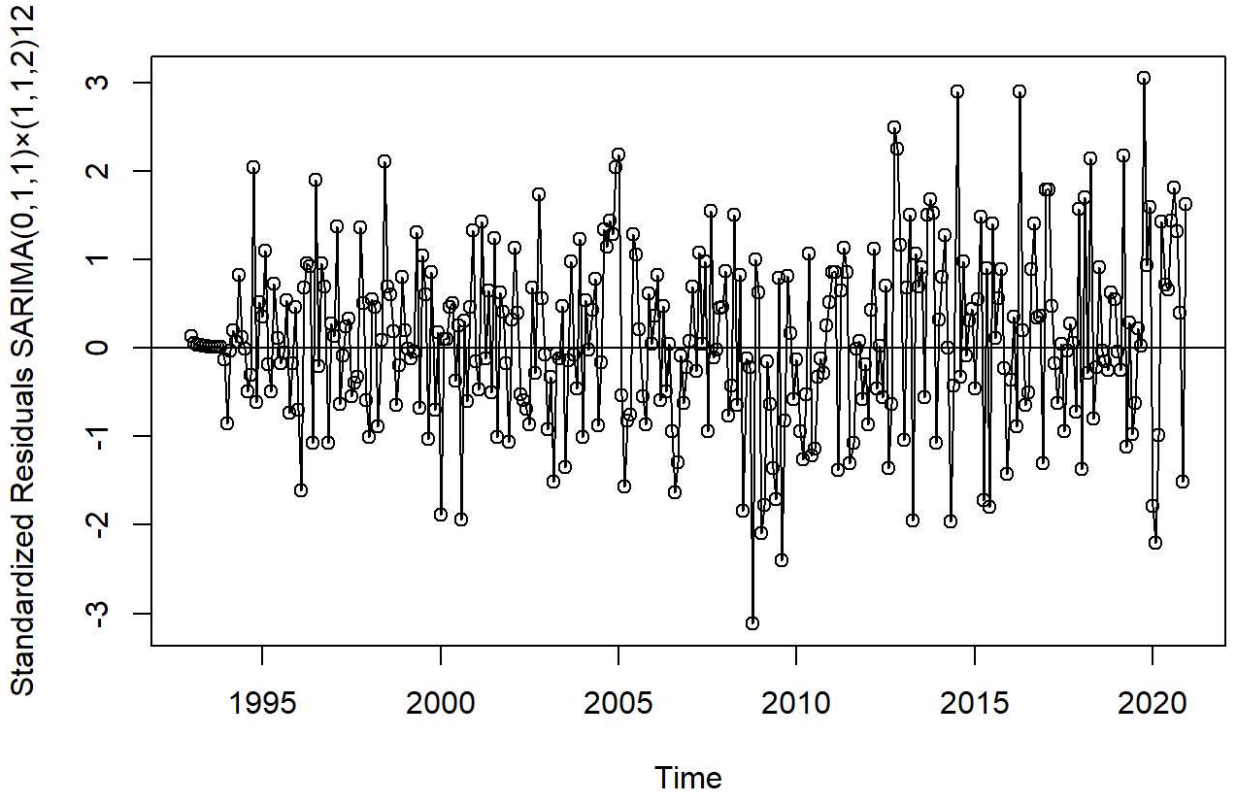
Further, BIC criteria has used to evaluate the models which has the lowest AIC values. As such, BIC criteria provides lowest value for “m8” which is 3078.016 which is slightly higher than the other model. Hence, both tests suggest the “m8” appears to be an more appropriate model for the time series that is seasonal ARIMA(0,1,1)×(1,1,2)12

1.13 Residual Analysis (Estimate of White noise)

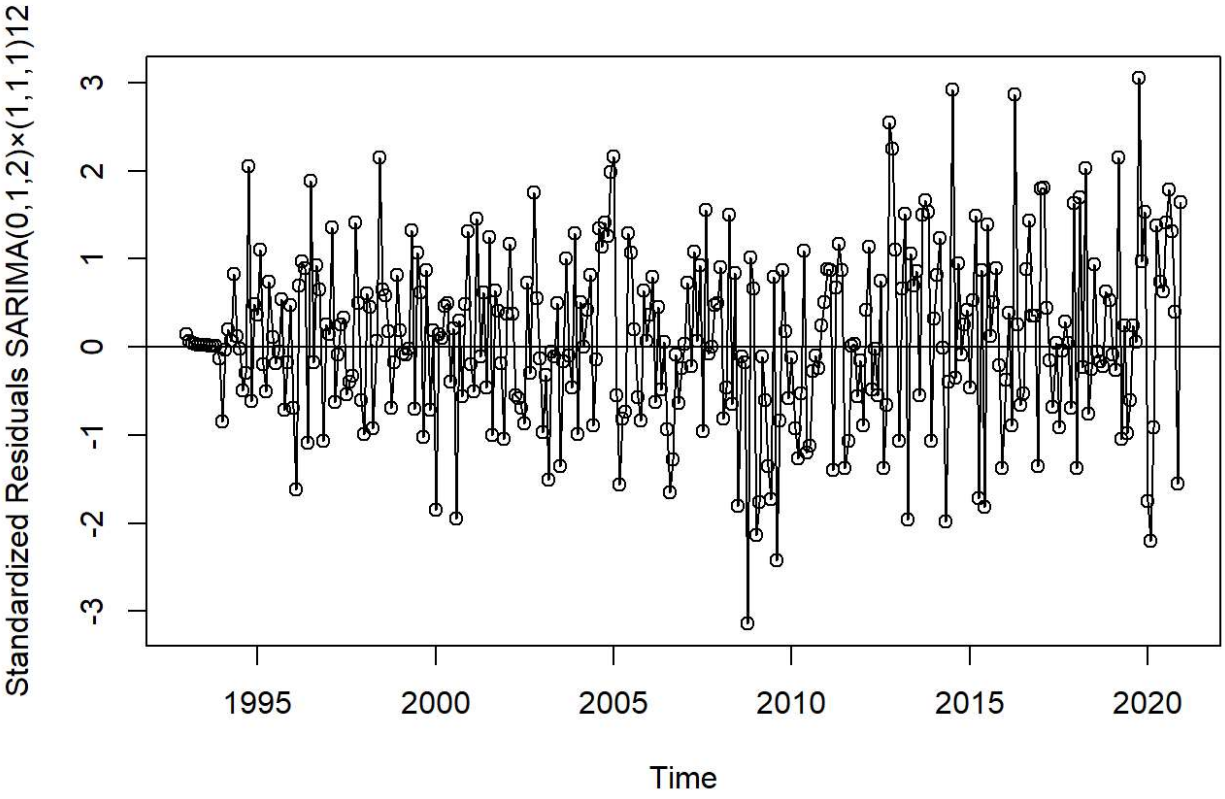
Residuals should be a white noise which is a weaker condition of stationary. Accordingly analysis is performed for residuals as below for the above selected two models; Seasonal ARIMA(0,1,1)×(1,1,2)12 and Seasonal ARIMA(0,1,2)×(1,1,1)12

1.13.1 Plots of the Residuals

```
#Reseduals plots for 2 selected models
plot(rstandard(m8),ylab ="Standardized Residuals SARIMA(0,1,1)×(1,1,2)12", type="o"); abline(h=0)
```




```
plot(rstandard(m9),ylab ="Standardized Residuals SARIMA(0,1,2)×(1,1,1)12", type="o"); abline(h=0)
```

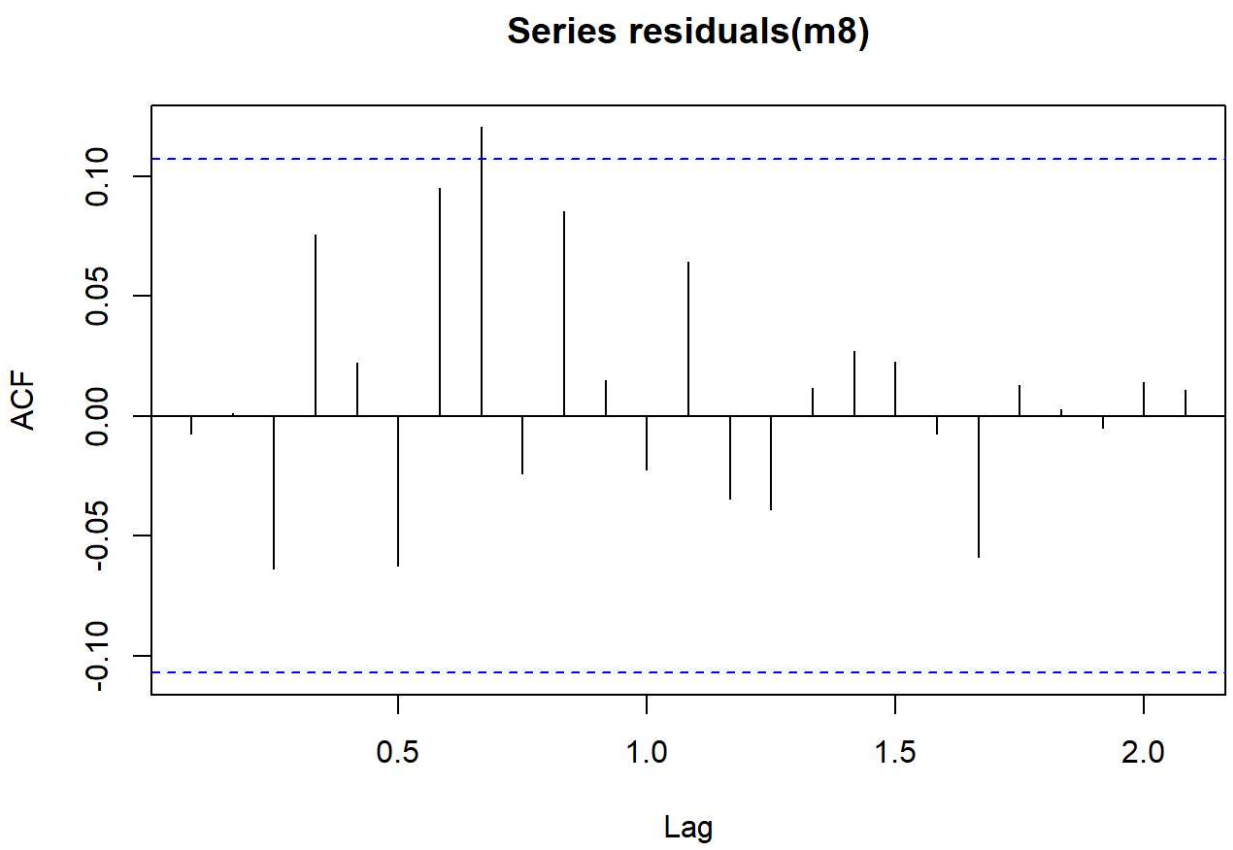


The above residual plots illustrate that the majority of residuals are oscillate around zero and approximately 95% of data fall between +2 & -2.

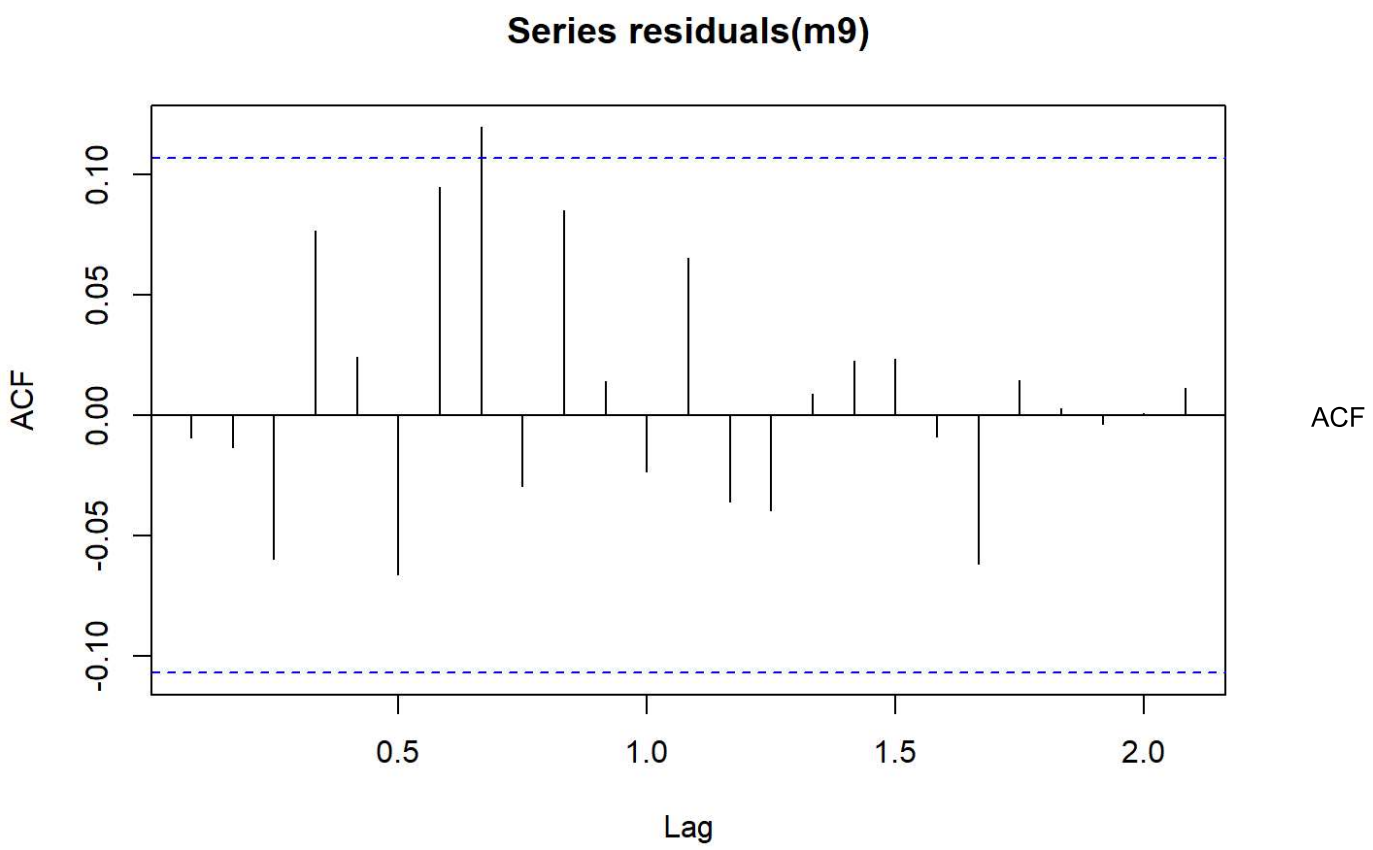
1.13.2 ACF of Residuals

As residuals should be a stationary which is independent and identically distributed. Hence, ACF of residuals should be close to zero and 95% of lags should be inside the interval of confidence.

```
acf(residuals(m8)) # Seasonal ARIMA(0,1,1)×(1,1,2)12
```




```
acf(residuals(m9)) # Seasonal ARIMA(0,1,2)×(1,1,1)12
```



function for residuals indicates that only one or two autocorrelation are out from the interval of confidence. Hence, that the 95% is within the interval of confidence and are approximately zero or closer to zero. As such, it appears to be no significant autocorrelation in the residuals and the residuals are uncorrelated.

1.13.3 Statistical tests for Residuals - Box-Pierce test and the Ljung-Box test.

(H0 : the data are sample from an iid sequence)

```
# Seasonal ARIMA(0,1,1)×(1,1,2)12
Box.test(residuals(m8)) #Box-Pierce test
```

```
##
##  Box-Pierce test
##
## data:  residuals(m8)
## X-squared = 0.017653, df = 1, p-value = 0.8943
```

```
Box.test(residuals(m8), lag= 12, type=c("Ljung-Box")) #Ljung-Box test
```

```
##
##  Box-Ljung test
##
## data:  residuals(m8)
## X-squared = 15.977, df = 12, p-value = 0.1923
```

```
#seasonal ARIMA(0,1,2)×(1,1,1)12
Box.test(residuals(m9)) #Box-Pierce test
```

```
##
##  Box-Pierce test
##
## data:  residuals(m9)
## X-squared = 0.029426, df = 1, p-value = 0.8638
```

```
Box.test(residuals(m9),lag= 12, type=c("Ljung-Box")) #Ljung-Box test
```

```
##
##  Box-Ljung test
##
## data:  residuals(m9)
## X-squared = 16.132, df = 12, p-value = 0.1853
```

H0 : Residuals or error terms are uncorrelated

The p-value of above Box-Pierce test and the Ljung-Box tests are not significant at the 5% level. Hence, the null hypothesis should not be rejected, since that there is enough evidence to suggest for Residuals or error terms are uncorrelated.

1.13.4 Stationary of Residuals - ADF test

```
#Dickey-Fuller test
Res_m8 <- residuals(m8) # seasonal ARIMA(0,1,1)×(1,1,2)12
adf.test(Res_m8)
```

```
## Warning in adf.test(Res_m8): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  Res_m8
## Dickey-Fuller = -6.1871, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

```
Res_m9 <- residuals(m9) #seasonal ARIMA(0,1,2)×(1,1,1)12
adf.test(Res_m9)
```

```
## Warning in adf.test(Res_m9): p-value smaller than printed p-value
```

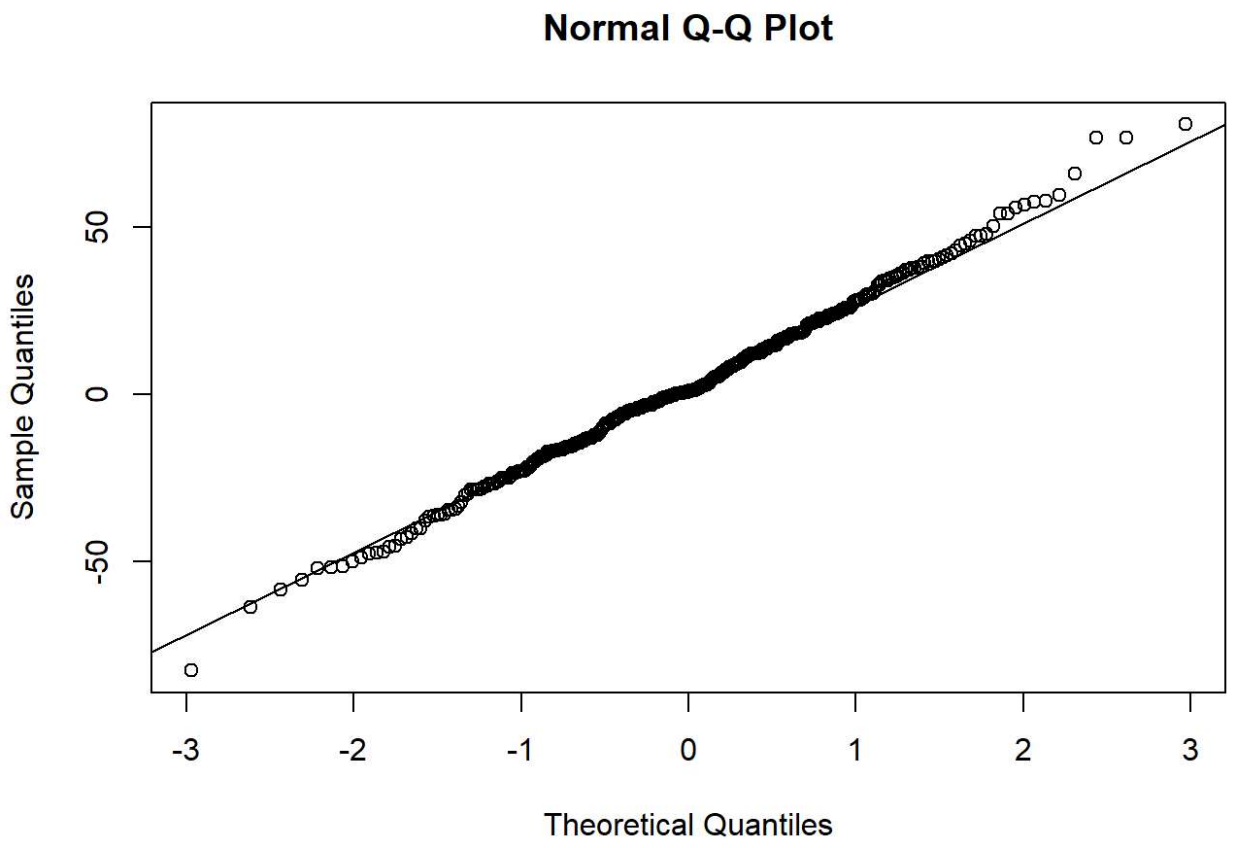
```
##
##  Augmented Dickey-Fuller Test
##
## data:  Res_m9
## Dickey-Fuller = -6.2118, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

H0: Time series is not a stationary, H1: Time series is a stationary

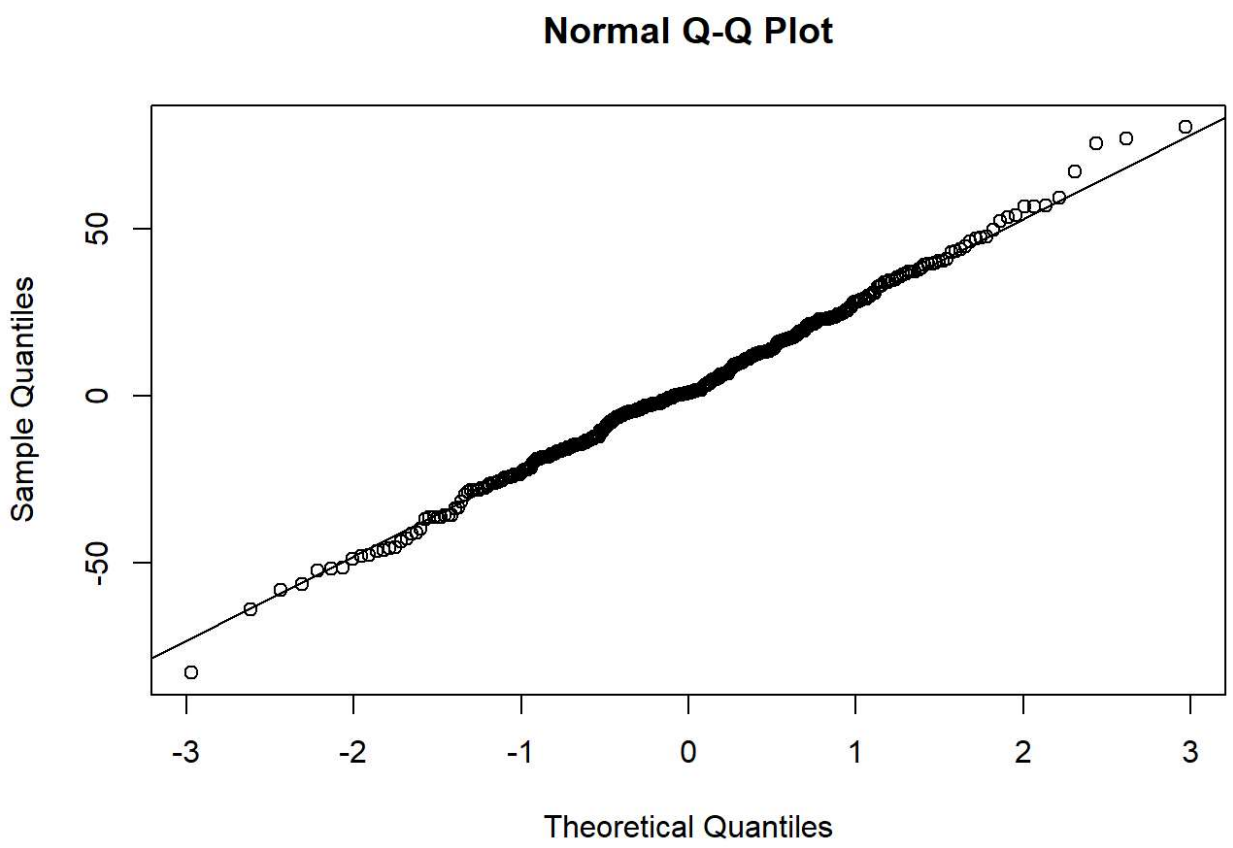
The p-value of both ADF test are below 0.01, which is significant at the 5% level. Hence, the null hypothesis should be rejected, since there is enough evidence in favor of alternative hypothesis (H1) and conclude that the time series is a stationary. This confirms the residuals are stationary or white noise.

1.13.5 Normality of the Residuals - QQ plots

```
qqnorm(residuals(m8)); qqline(residuals(m8)) # seasonal ARIMA(0,1,1)×(1,1,2)12
```




```
qqnorm(residuals(m9)); qqline(residuals(m9)) #seasonal ARIMA(0,1,2)×(1,1,1)12
```



Above plots illustrate that the quantiles of the residuals is close to the normal distribution in general.However, few deviations are noticable at the edge of the line. Hence, A statistical test has been performed to chack the normality of the distribution.

1.13.6 Statistical tests - Normality of the Residuals

```
shapiro.test(rstandard(m8)) # seasonal ARIMA(0,1,1)×(1,1,2)12
```

```
##
##  Shapiro-Wilk normality test
##
## data:  rstandard(m8)
## W = 0.99713, p-value = 0.8217
```

```
shapiro.test(rstandard(m9)) #seasonal ARIMA(0,1,2)×(1,1,1)12
```

```
##
##  Shapiro-Wilk normality test
##
## data:  rstandard(m9)
## W = 0.99709, p-value = 0.8133
```

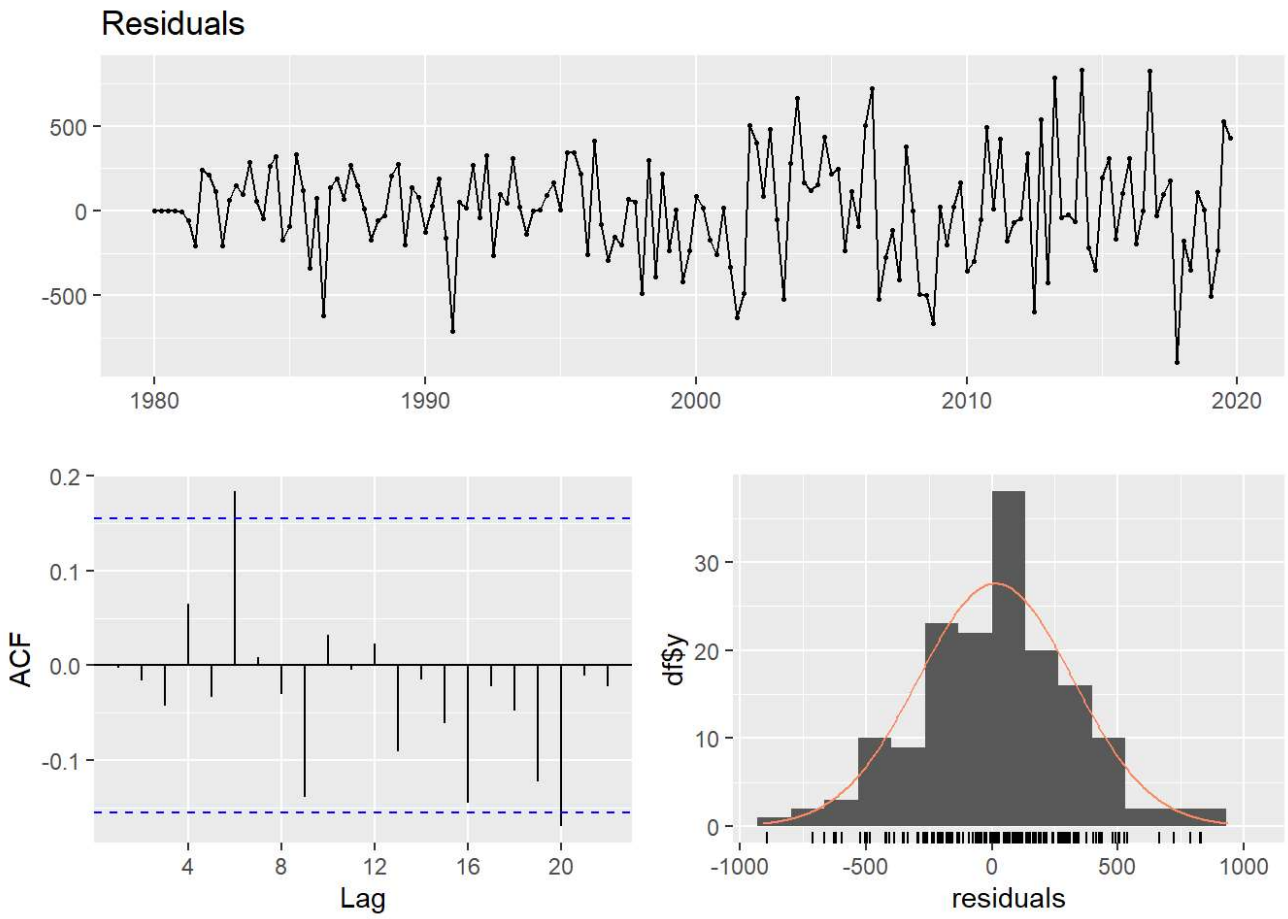
H0: Data is normally distributed

Shapiro-Wilk test of normality has a test statistic p-value is 0.8217 and HO should not be rejected. Hence, the residuals are normally distributed.

Summary of residuals analysis

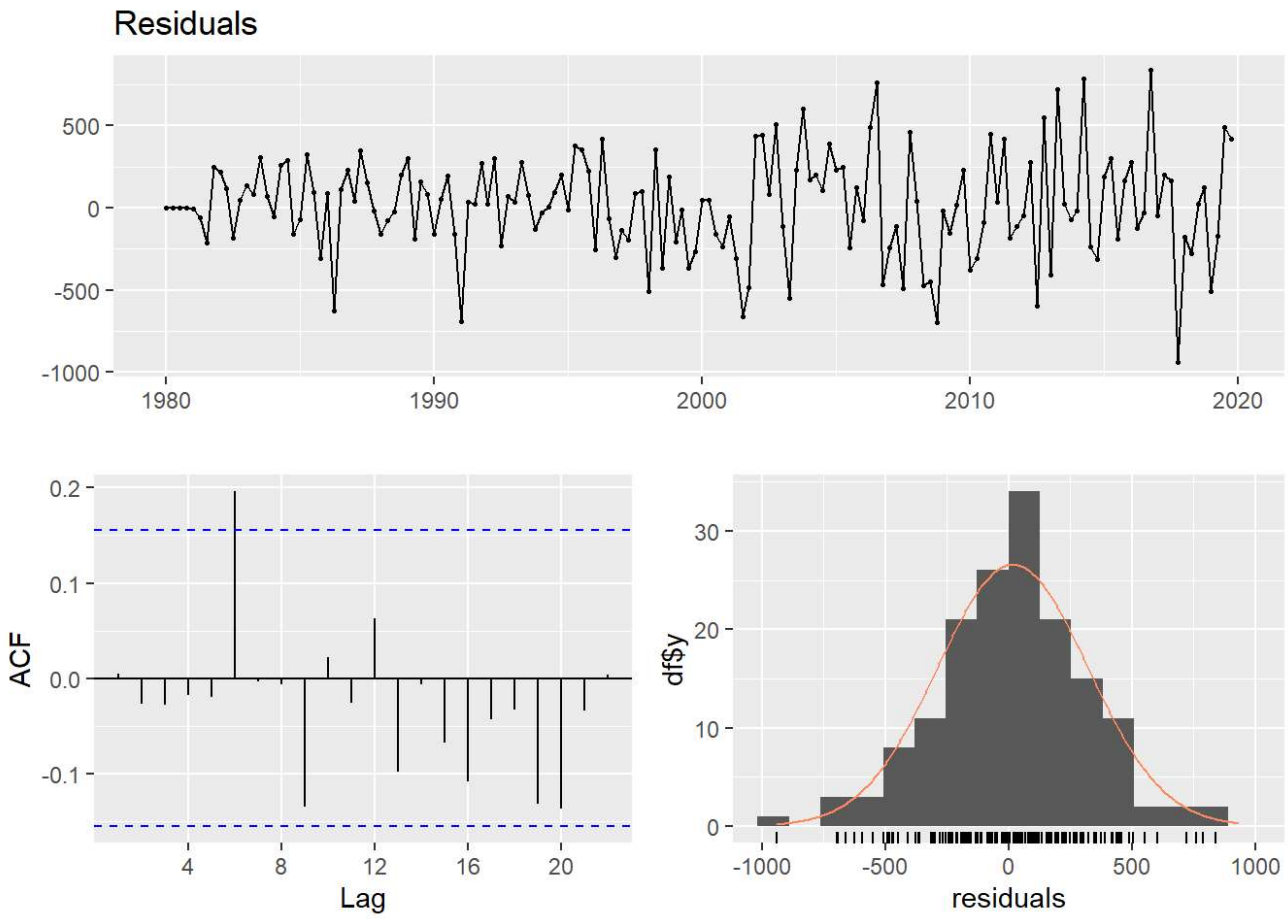
```
checkresiduals(Res_pq0201) # seasonal ARIMA(0,1,1)×(1,1,2)12
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```




```
checkresiduals(Res_pq1111) #seasonal ARIMA(0,1,2)×(1,1,1)12
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```

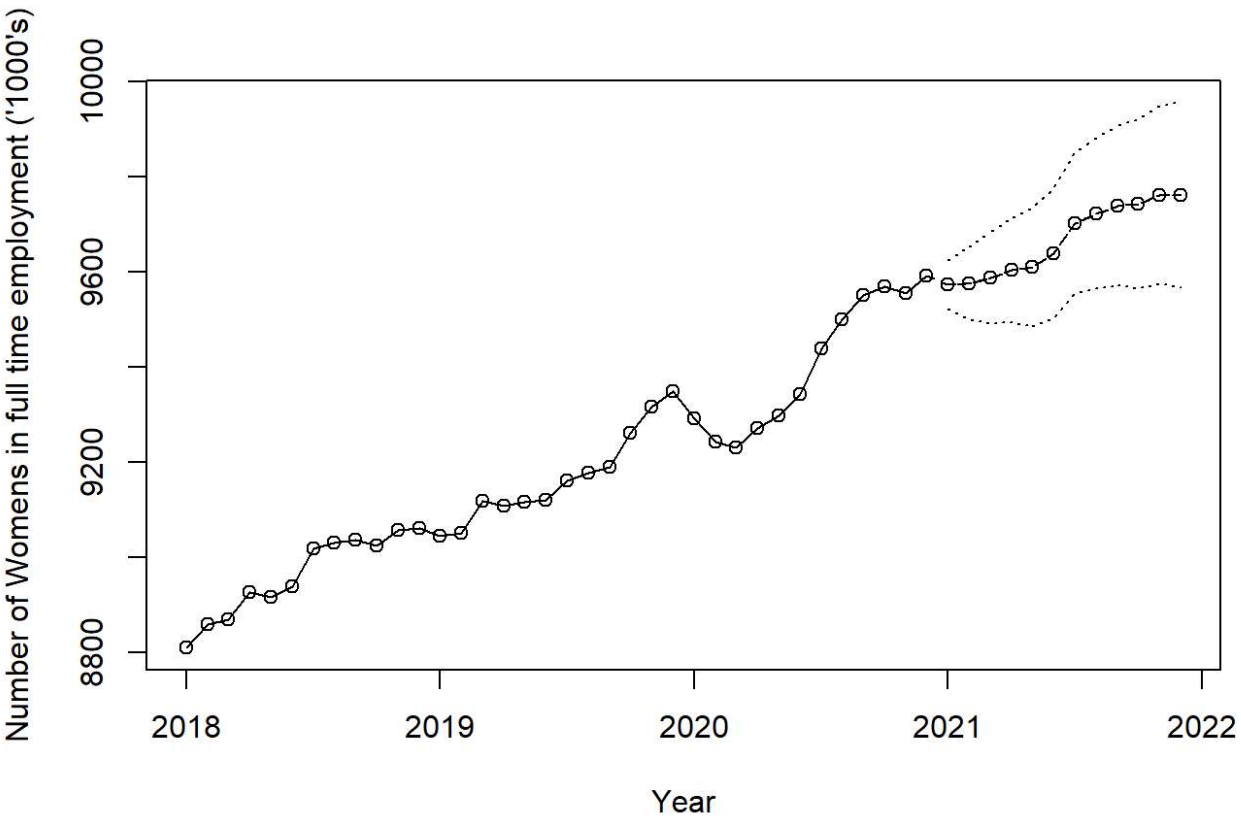


Residuals analysis illustrate that the residuals are stationary/white noise (weaker conditions), uncorrelated (correlation=0, independent) and normally distributed for both the models.

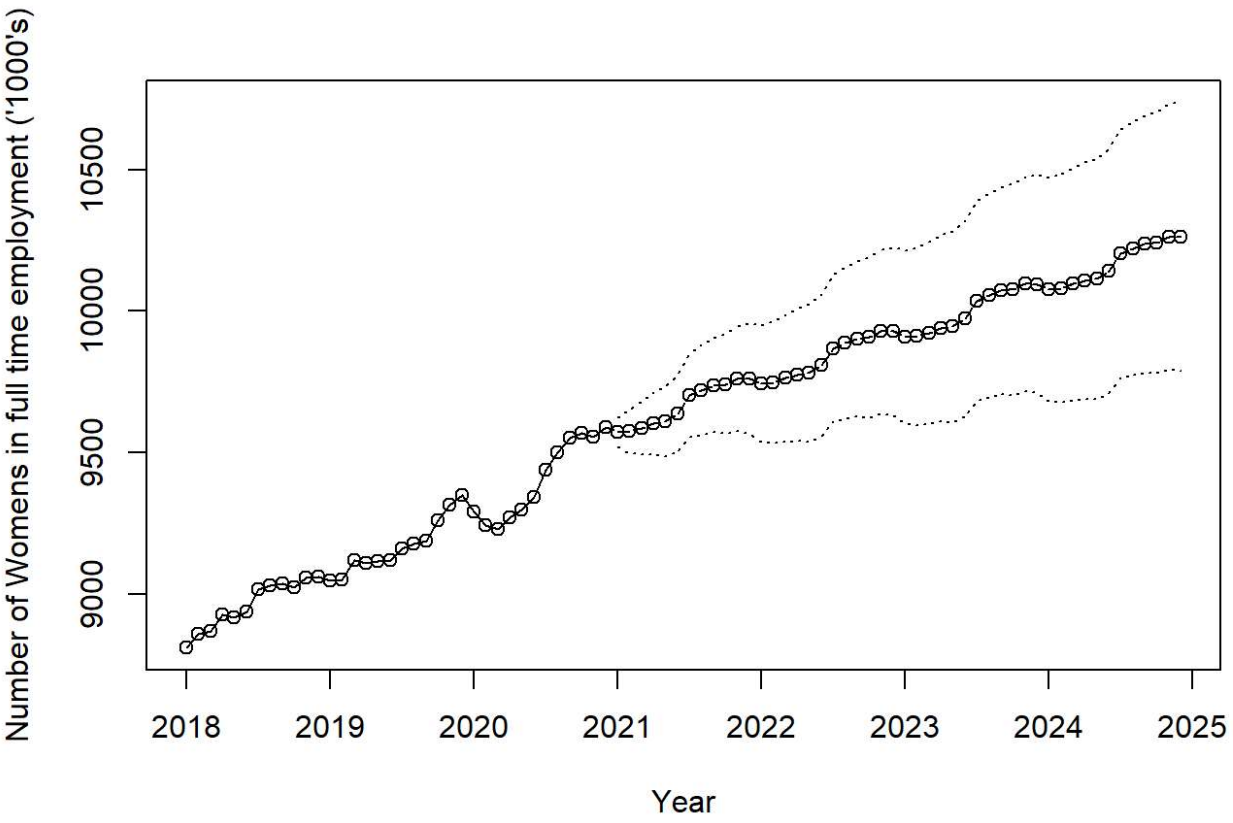
1.14 Forecasting of the model

Forecasting for the model pp0201 which is (Seasonal ARIMA(0,1,1)×(1,1,2)12) is as below.

```
plot(m8,n1=c(2018,1),n.ahead=12,xlab="Year",type="o", ylab="Number of Womens in full time employmen
t ('1000's)")
```




```
plot(m8,n1=c(2018,1),n.ahead=48, xlab="Year",type="o", ylab="Number of Womens in full time employme
nt ('1000's)")
```



Seasonal ARIMA(0,1,1)×(1,1,2)12 model was used to forecast for next 12 and 48 months respectively. It could examine that the forecasted values of the model (Seasonal ARIMA(0,1,1)×(1,1,2)12 is similar to the pattern of the previous months of the year. Hence, model is appropriate and fitted to forecast the values of the time series.

The time plot for next 48 months indicate that the interval of confidence increases as the period increases which leads predictions become less precise. Hence, forecasting for quite lengthy periods tend to have increased interval of confidence.

1.15 Forecast errors

A forecast “error” is the difference between an observed value and its forecast. Accuracy of the model prediction depends on the length of the trained data. Hence, the data set used for this analysis has quite lengthy period of past data starting from Qtr1 of 1980 to Qtr 4 of 2019.

```
# train and test data
train_data2 <-window(monthly_data,end=c(2019,12))
test_data2  <- window(monthly_data,start=c(2020,1))

m8_train<-arima(train_data2,order=c(0,1,1),seasonal=list(order=c(1,1,2), period=12)) # SARIMA(0,1,
1)×(1,1,2)12
m9_train<-arima(train_data2,order=c(0,1,2),seasonal=list(order=c(1,1,1), period=12)) #  SARIMA(0,1,
2)×(1,1,1)12
```

```
# predict.Arima: Forecast from ARIMA fits
predict_m8 <- predict(m8_train,n.ahead = 12)
accuracy(predict_m8$pred, test_data2)
```

```
##                ME     RMSE      MAE        MPE      MAPE      ACF1 Theil's U
## Test set -24.3739 79.72052 71.99888 -0.2704733 0.7685606 0.8044557  1.752412
```

```
predict_m9 <- predict(m9_train,n.ahead = 12)
accuracy(predict_m9$pred, test_data2)
```

```
##                ME     RMSE      MAE        MPE      MAPE      ACF1 Theil's U
## Test set -24.8917 78.99092 71.29351 -0.2758236 0.7610887 0.8034666  1.737961
```

Above two test results is related to the forecast errors of the two models namely m8 and the m9. Lesser values for forecast error is given by the m9 model which is Seasonal ARIMA(0,1,2)×(1,1,1)12. Hence, based on the analysis of forecast errors, it assume Seasonal ARIMA(0,1,2)×(1,1,1)12 as the best model out of the tested models.

1.17 Prediction comparrison

```
#Values of fitted model
test_data2
```

```
##       Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec
## 2020 9292 9241 9230 9271 9298 9342 9437 9500 9550 9569 9555 9590
```

```
predict_m9$pred
```

```
##           Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
## 2020 9337.155 9349.996 9370.231 9378.460 9382.661 9406.719 9462.890 9474.714
##           Sep      Oct      Nov      Dec
## 2020 9485.902 9491.936 9518.544 9514.493
```

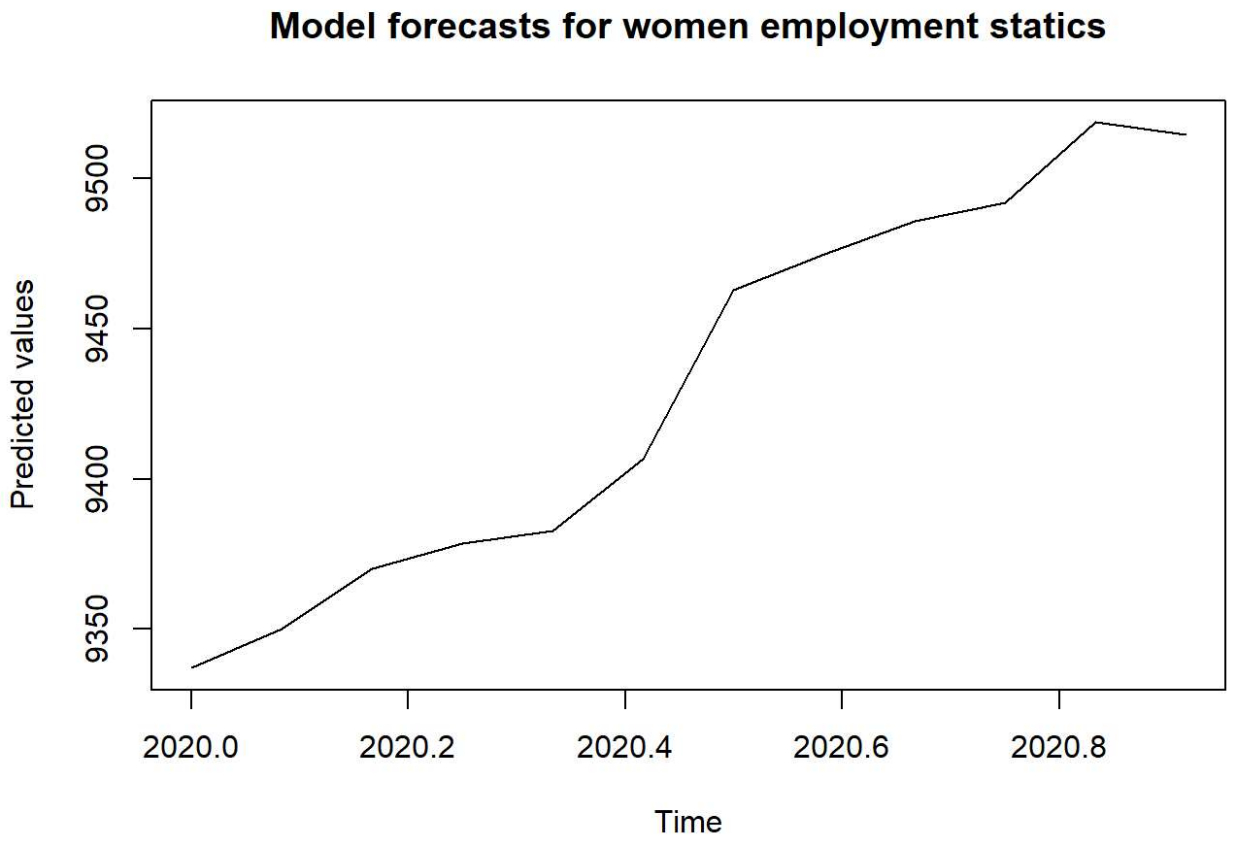
```
# Comparison among test and forecast data
aa <- test_data2
bb <- predict_m9$pred
cc <- data.frame(aa,bb)

names(cc)<- c("Test data", "Forecast")
cc
```

```
##    Test data Forecast
## 1       9292 9337.155
## 2       9241 9349.996
## 3       9230 9370.231
## 4       9271 9378.460
## 5       9298 9382.661
## 6       9342 9406.719
## 7       9437 9462.890
## 8       9500 9474.714
## 9       9550 9485.902
## 10      9569 9491.936
## 11      9555 9518.544
## 12      9590 9514.493
```

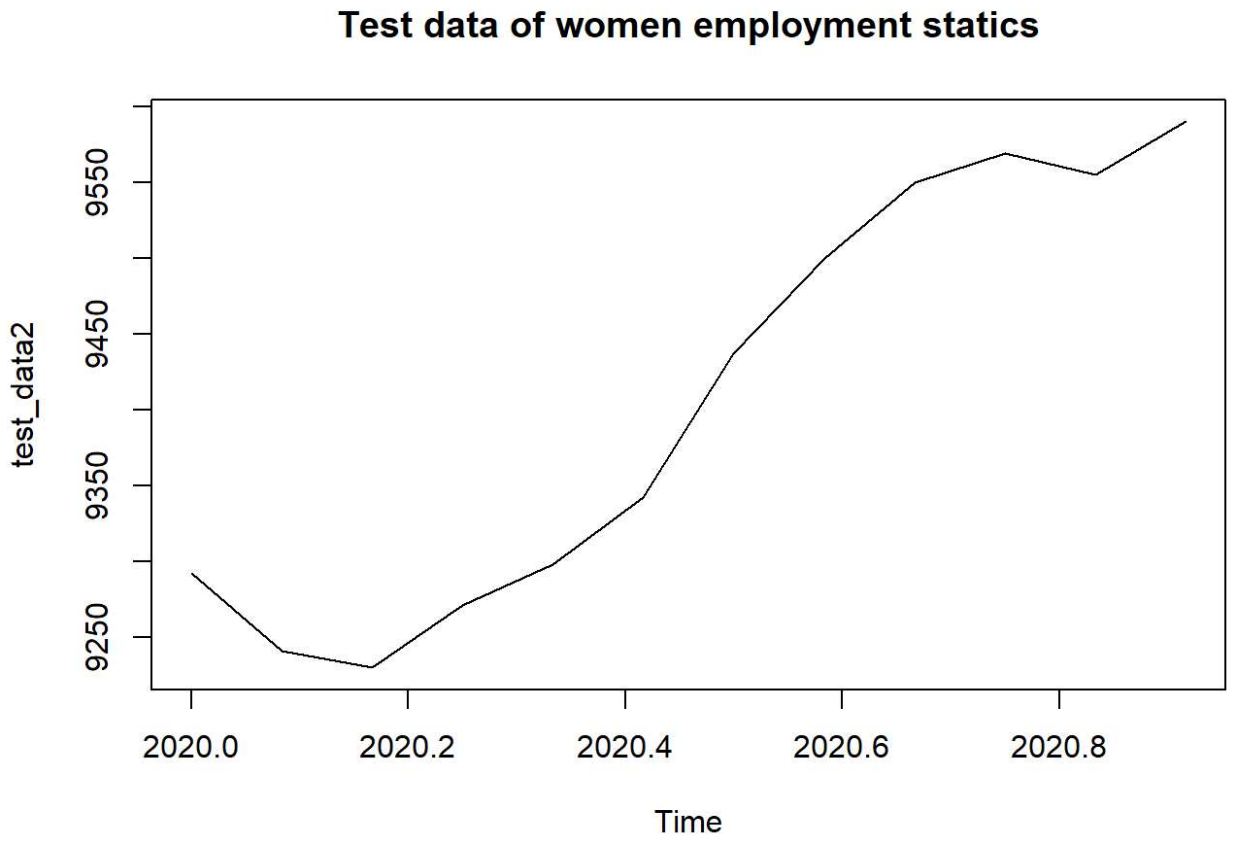
Above output illustrate the values of test data and predicted value of Seasonal ARIMA(0,1,2)×(1,1,1)12 model for the test data on the months of year 2020. As such, it was observed that the predicted values of the model follows similar pattern to the previous data. Further, values predicted of the fitted model is closer to the actual values. Further, below plots illustrate the patterns relevant to the above data.

1.18 Visualisation of predictions

```
#Ploting
plot(predict_m9$pred, type = "l", ylab="Predicted values" , main = "Model forecasts for women emplo
yment statics" )
```




```
plot(test_data2,  main = "Test data of women employment statics" )
```



1.19 Conclusion:

Time series for Womens in full time employment statistics in UK was analysed in this study. As such, time series was convert to a stationary time series, develop list of possible models, model fitting and , residual analysis was performed. Finally, Seasonal ARIMA(0,1,2)×(1,1,1)12 was fitted to forecast the number of overseas visits to UK. Then, apply that model to train and test data set of the same time series and evaluate the prediction criteria. As such, the model follows the similar pattern to that of past data and predict values closer to the test value set.

Limitations and future developments:

This model can be improved by application of some machine learning prediction algorithms to predict the values almost similar to the test values. Further, this data set is related pre-covid situation. However, there may be variations during covid seasons and afterwards. The data set used in this analysis is pre-Covid period. Hence, another model could be develop to track the during_Covid and post_Covid forecasts.